

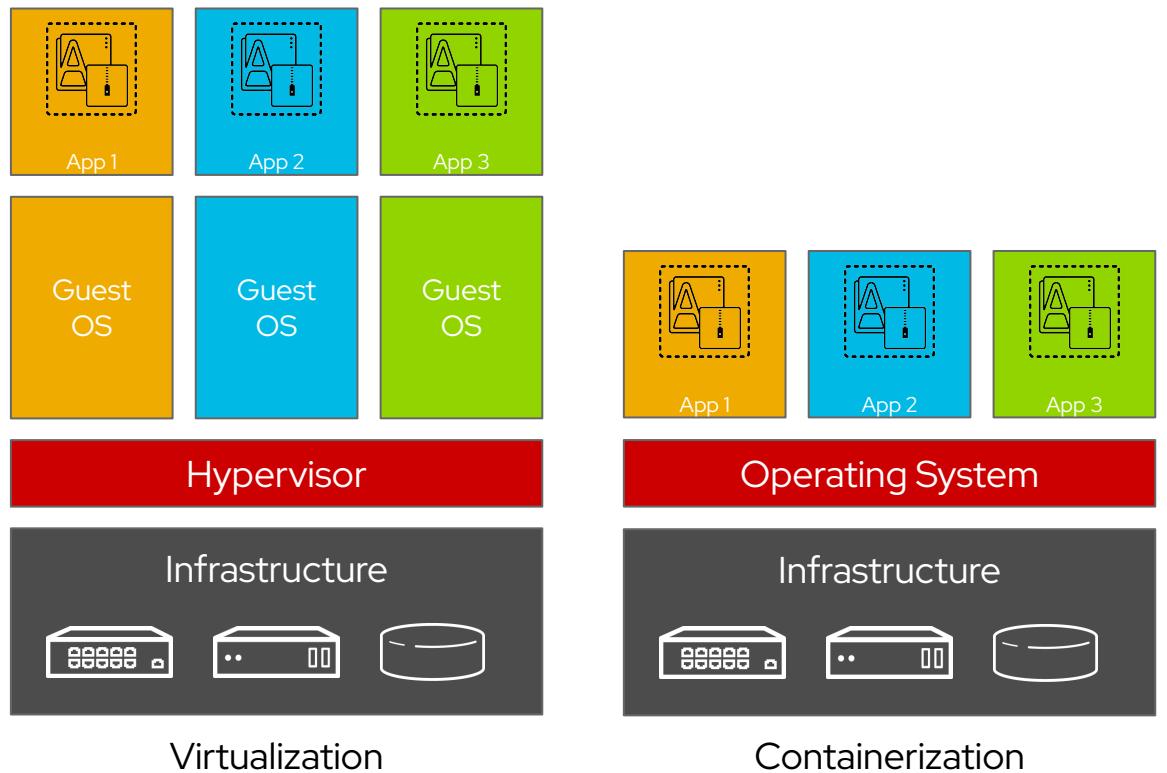


OpenShift Virtualization

Technical presentation

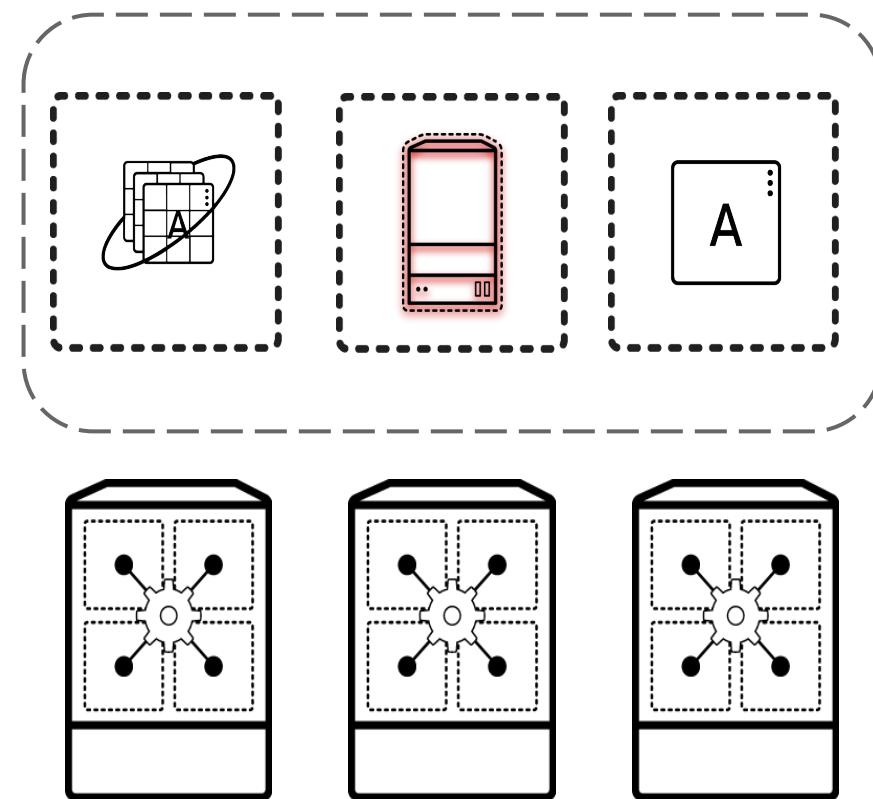
Containers are not virtual machines

- Containers are process isolation
- Kernel namespaces provide isolation and cgroups provide resource controls
- No hypervisor needed for containers
- Contain only binaries, libraries, and tools which are needed by the application
- Ephemeral



Virtual machines can be put into containers

- A KVM virtual machine is a process
- Containers encapsulate processes
- Both have the same underlying resource needs:
 - Compute
 - Network
 - (sometimes) Storage



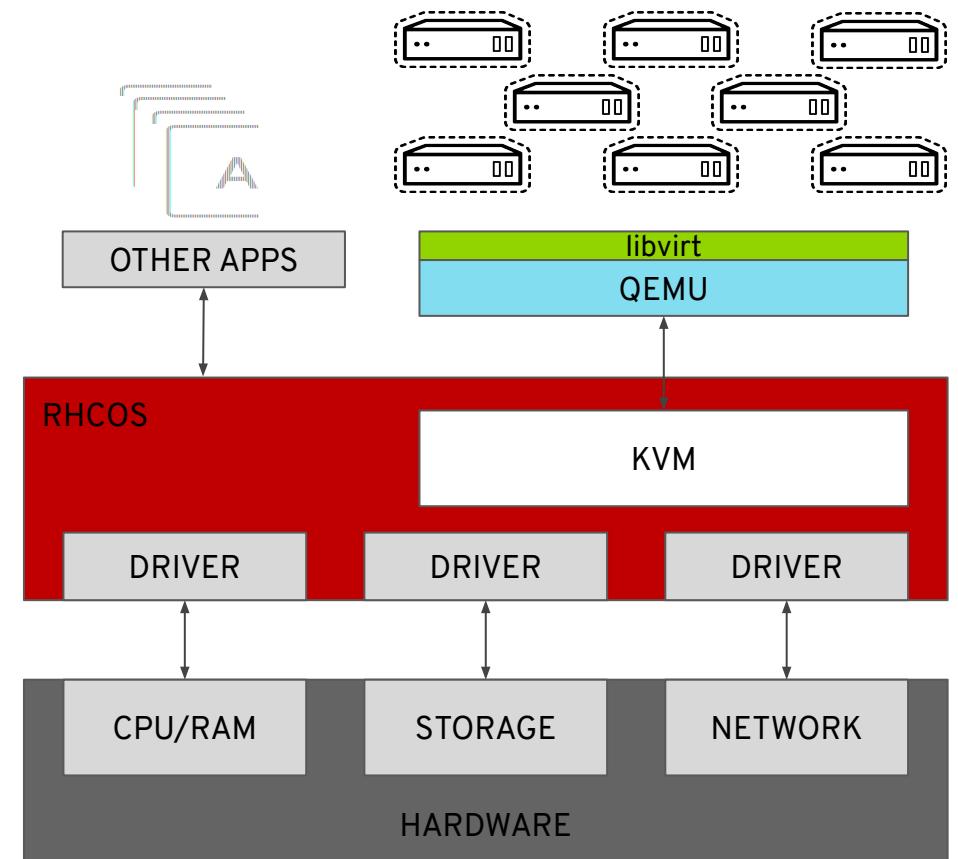
OpenShift Virtualization

- Virtual machines
 - Running in containers
 - Using the KVM hypervisor
- Scheduled, deployed, and managed by Kubernetes
- Integrated with container orchestrator resources and services
 - Traditional Pod-like SDN connectivity and/or connectivity to external VLAN and other networks via multus
 - Persistent storage paradigm (PVC, PV, StorageClass)



VM containers use KVM

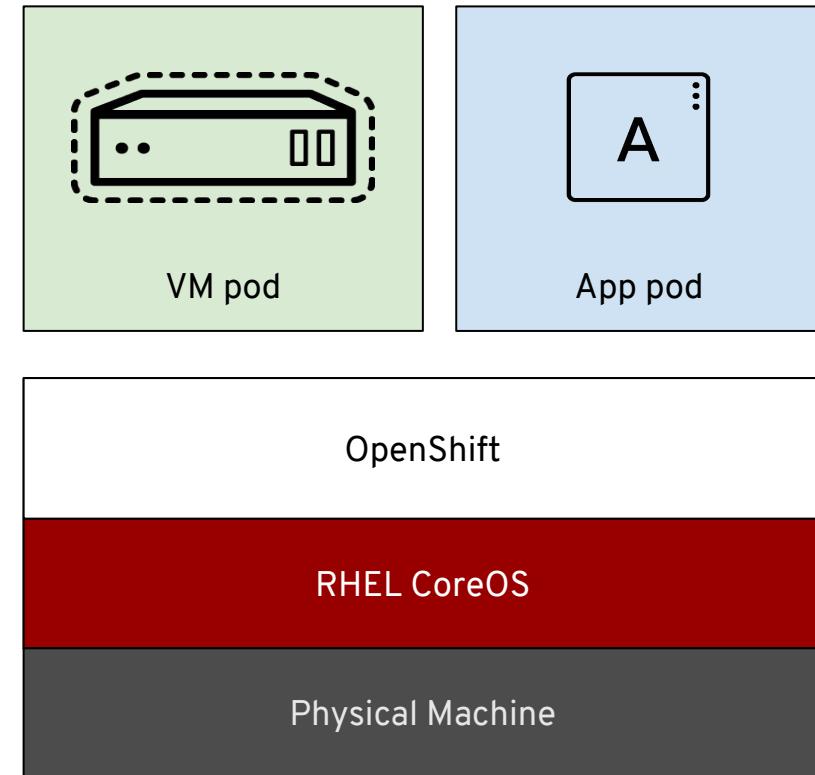
- OpenShift Virtualization uses KVM, the Linux kernel hypervisor
- KVM is a core component of the Red Hat Enterprise Linux kernel
 - KVM has 10+ years of production use: Red Hat Virtualization, Red Hat OpenStack Platform, and RHEL all leverage KVM, QEMU, and libvirt
- QEMU uses KVM to execute virtual machines
- libvirt provides a management abstraction layer



Built with Kubernetes

Virtual machines in a container world

- Provides a way to transition application components which can't be directly containerized into a Kubernetes system
 - Integrates directly into existing k8s clusters
 - Follows Kubernetes paradigms:
 - Container Networking Interface (CNI)
 - Container Storage Interface (CSI)
 - Custom Resource Definitions (CRD, CR)
- Schedule, connect, and consume VM resources as container-native

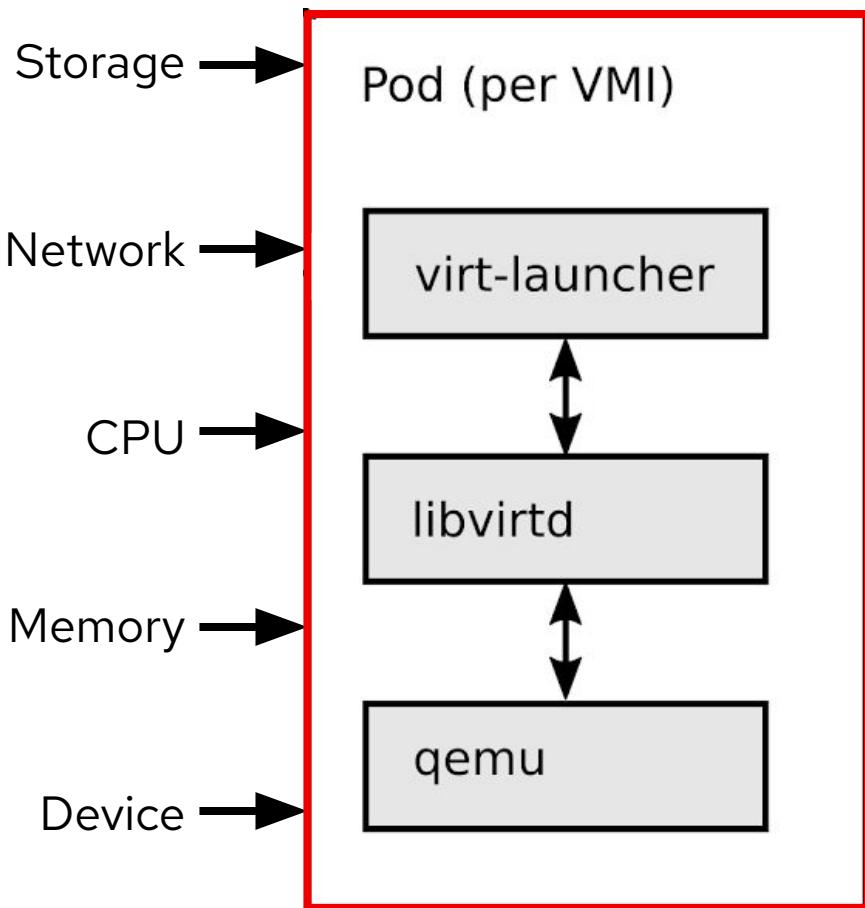


Virtualization native to Kubernetes

- Operators are a Kubernetes-native way to introduce new capabilities
- New CustomResourceDefinitions (CRDs) for native VM integration, for example:
 - `VirtualMachine`
 - `VirtualMachineInstance`
 - `VirtualMachineInstanceMigration`
 - `DataVolume`

```
apiVersion: kubevirt.io/v1alpha3
kind: VirtualMachine
metadata:
  labels:
    app: demo
    flavor.template.kubevirt.io/small: "true"
  name: rhel
spec:
  dataVolumeTemplates:
  - apiVersion: cdi.kubevirt.io/v1alpha1
    kind: DataVolume
    metadata:
      creationTimestamp: null
      name: rhel-rootdisk
    spec:
      pvc:
        accessModes:
        - ReadWriteMany
      resources:
        requests:
          storage: 20Gi
      storageClassName: managed-nfs-storage
      volumeMode: Filesystem
```

Containerized virtual machines



Kubernetes resources

- Every VM runs in a launcher pod. The launcher process will supervise, using libvirt, and provide pod integration.

Red Hat Enterprise Linux

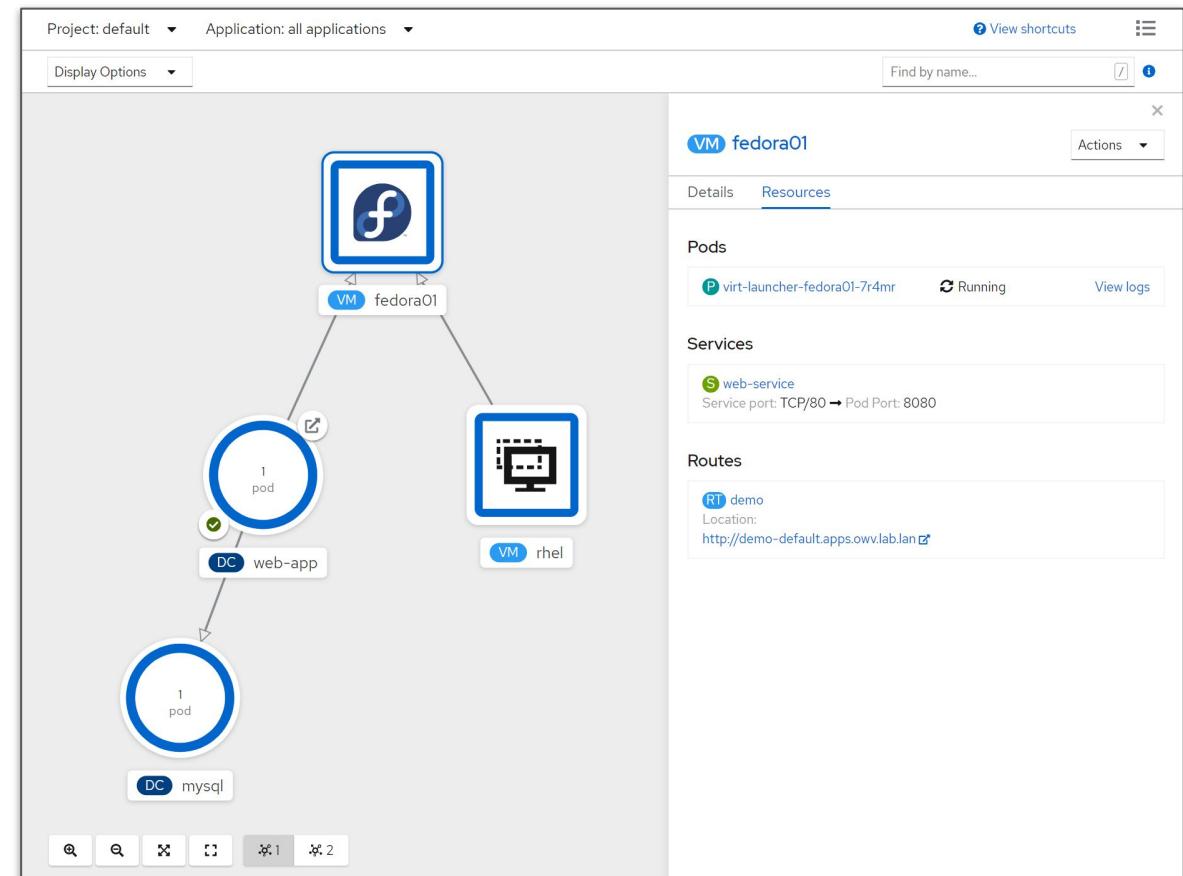
- libvirt and qemu from RHEL are mature, have high performance, provide stable abstractions, and have a minimal overhead.

Security - Defense in depth

- Immutable RHCOS by default, SELinux MCS, plus KVM isolation - inherited from the Red Hat Portfolio stack

Using VMs and containers together

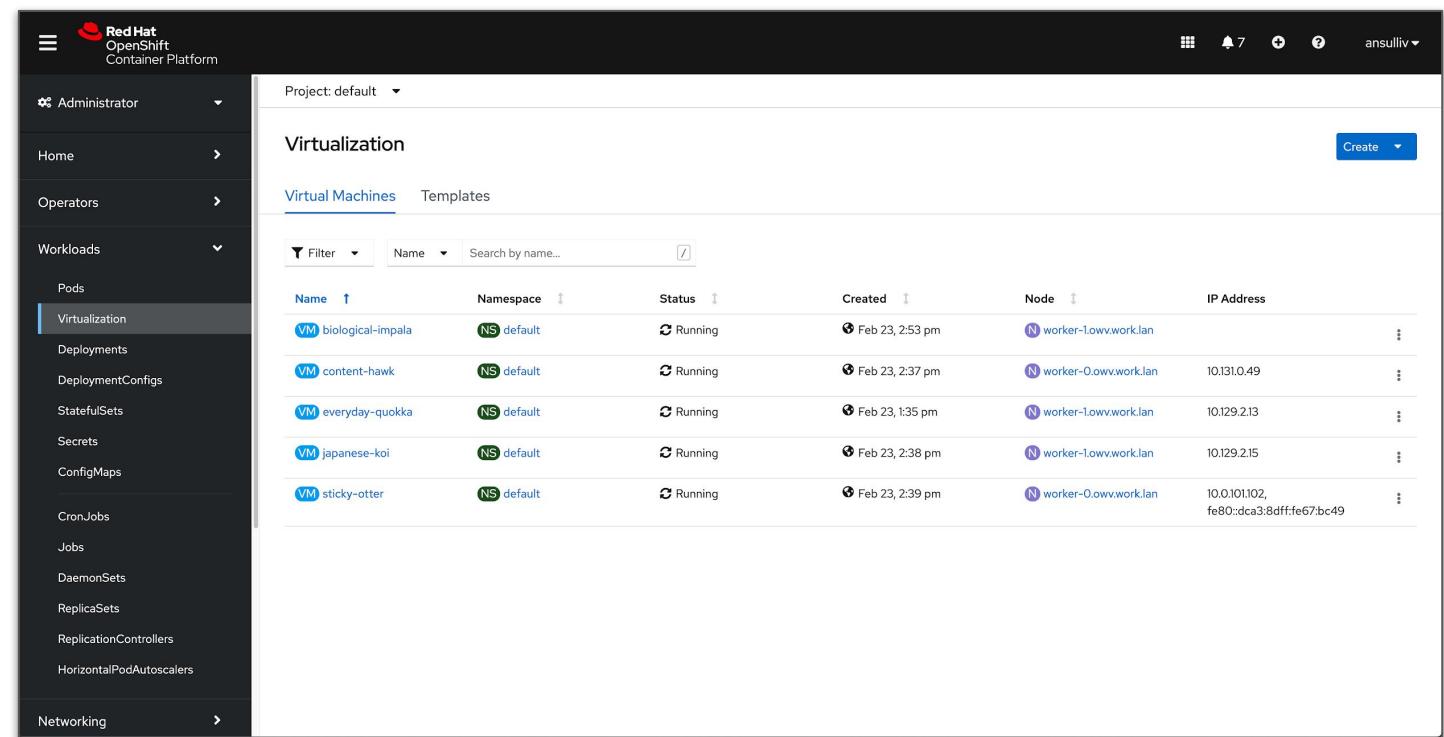
- Virtual Machines connected to pod networks are accessible using standard Kubernetes methods:
 - Service
 - Route
 - Ingress
- Network policies apply to VM pods the same as application pods
- VM-to-pod, and vice-versa, communication happens over SDN or ingress depending on network connectivity



Managed with
OpenShift

Virtual Machine Management

- Create, modify, and destroy virtual machines, and their resources, using the OpenShift web interface or CLI
- Use the `virtctl` command to simplify virtual machine interaction from the CLI



Name	Namespace	Status	Created	Node	IP Address
VM biological-impala	NS default	Running	Feb 23, 2:53 pm	worker-1.oww.work.lan	
VM content-hawk	NS default	Running	Feb 23, 2:37 pm	worker-0.oww.work.lan	10.131.0.49
VM everyday-quokka	NS default	Running	Feb 23, 1:35 pm	worker-1.oww.work.lan	10.129.2.13
VM japanese-koi	NS default	Running	Feb 23, 2:38 pm	worker-1.oww.work.lan	10.129.2.15
VM sticky-otter	NS default	Running	Feb 23, 2:39 pm	worker-0.oww.work.lan	10.0.101.102, fe80::dca3:8dff:fe67:bc49

Create VMs

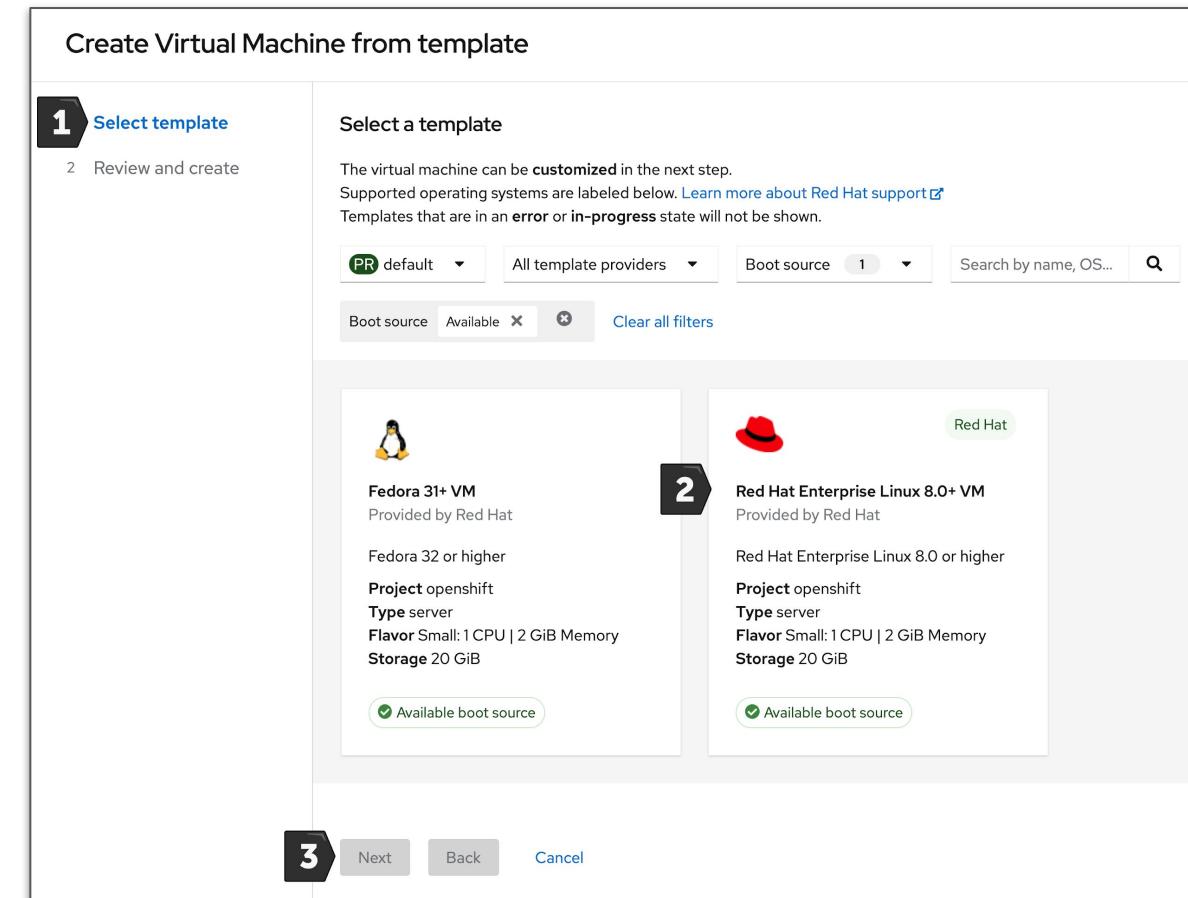
Virtual Machine creation

- Streamlined and simplified creation via the GUI or create VMs programmatically using YAML
- Full configuration options for compute, network, and storage resources
 - Clone VMs from templates or import disks using DataVolumes
 - Pre-defined and customizable presets for CPU/RAM allocations
 - Workload profile to tune KVM for expected behavior
- Import VMs from VMware vSphere or Red Hat Virtualization

Name	Namespace	IP Address
VM biological-impala	openlan	
VM content-hawk	openlan	10.131.0.49
VM everyday-quokka	openlan	10.129.2.13
VM japanese-koi	openlan	10.129.2.15
VM sticky-otter	openlan	10.0.101.102, fe80::dca3:8dff:fe67:bc49

Using templates for virtual machines

- Simplified and streamlined virtual machine creation experience for VM consumers
- Administrators configure templates with an OS disk, consumers select from the options



Creating a virtual machine

- Flavor represents the preconfigured CPU and RAM assignments
- Storage, including PVC size and storage class, are determined by the template administrator
- A default network configuration is defined in the template
- Workload profile defines the category of workload expected and is used to set KVM performance flags
- The VM can be further customized by selecting the option, or immediately created and deployed
 - Additional customization includes CPU/memory, storage, network, cloud-init, and more

Create Virtual Machine from template

1 Select template 2 Review and create 3 4 Create virtual machine 5 Customize virtual machine

Review and create
You are creating a virtual machine from the **Red Hat Enterprise Linux 8.0+ VM** template.

Project *
PR default

Virtual Machine Name * [?](#)
rhel8-pleasant-ladybug

Flavor *
Small: 1 CPU | 2 GiB Memory

Storage
20 GiB **Workload profile** [?](#)
server

Boot source
Clone and boot from disk
PVC rhel8

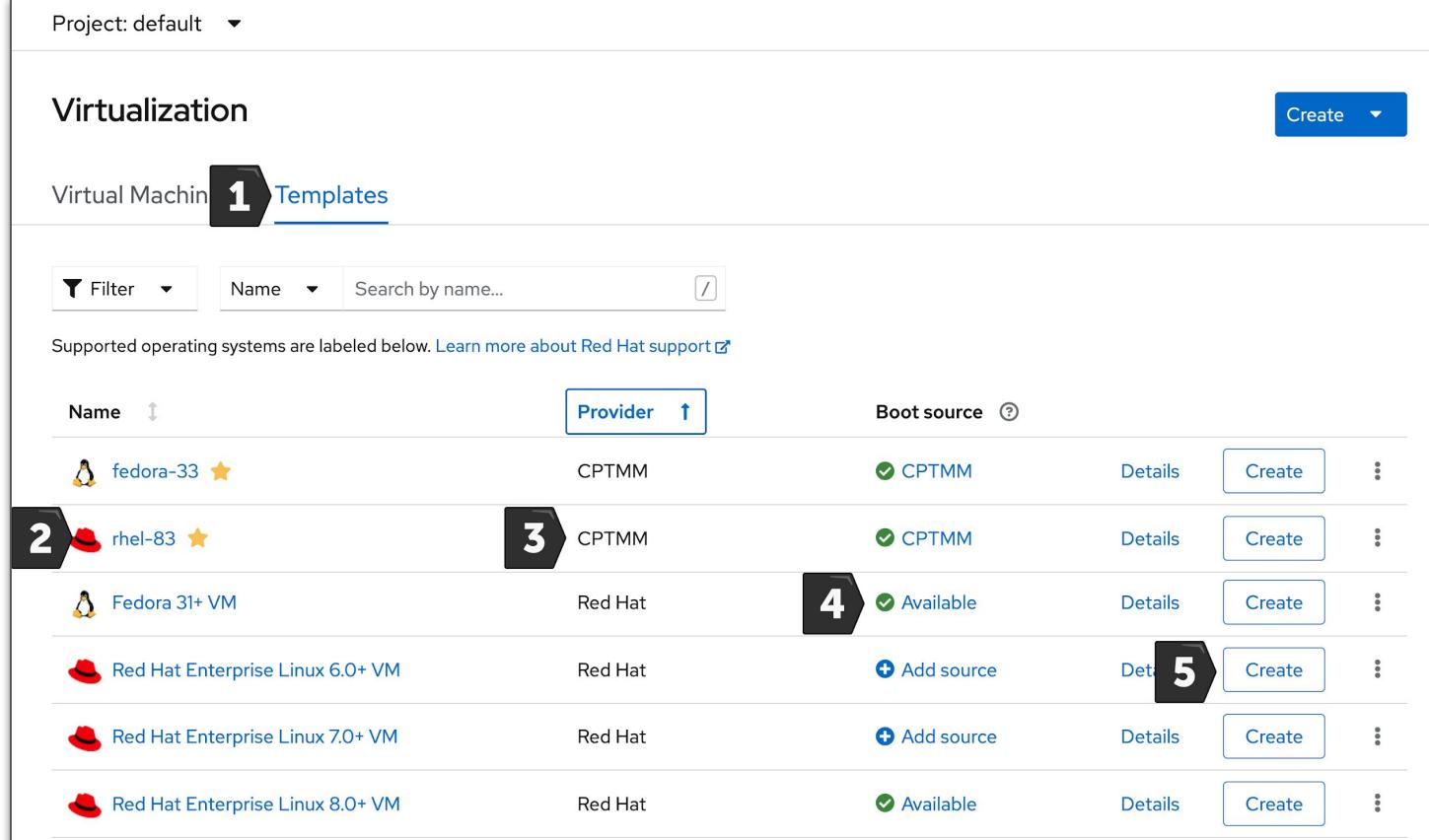
Start this virtual machine after creation

Back Cancel

VM Templates

Templates

- Templates are a core concept for virtual machine creation with OpenShift Virtualization 2.6
- Red Hat provides default templates, administrators can create and customize additional as needed
- Boot sources provide disk images for the template
- Creating VMs can be done from the template page in just a few clicks



The screenshot shows the 'Virtualization' section of the OpenShift Virtualization 2.6 interface. The 'Virtual Machine' tab is selected, and the 'Templates' tab is highlighted. The page displays a list of templates, each with a name, provider, boot source, and actions. The templates listed are: fedora-33 (CPTMM), rhel-83 (CPTMM), Fedora 31+ VM (Red Hat), Red Hat Enterprise Linux 6.0+ VM (Red Hat), Red Hat Enterprise Linux 7.0+ VM (Red Hat), and Red Hat Enterprise Linux 8.0+ VM (Red Hat). Annotations are as follows:

- 1: Points to the 'Templates' tab in the navigation bar.
- 2: Points to the 'rhel-83' template.
- 3: Points to the 'CPTMM' provider for the 'rhel-83' template.
- 4: Points to the 'Available' status for the 'Fedora 31+ VM' template.
- 5: Points to the 'Create' button for the 'Red Hat Enterprise Linux 8.0+ VM' template.

Name	Provider	Boot source	Actions
fedora-33	CPTMM	CPTMM	Details Create
rhel-83	2 CPTMM	CPTMM	Details Create
Fedora 31+ VM	3 Red Hat	Available	Details Create
Red Hat Enterprise Linux 6.0+ VM	Red Hat	Add source	Details Create
Red Hat Enterprise Linux 7.0+ VM	Red Hat	Add source	Details Create
Red Hat Enterprise Linux 8.0+ VM	Red Hat	Available	Details Create

Create a template - General

- In addition to unique names, each template is associated with a provider
 - Providers represent who created the template, with optional support information
- The guest operating system and source boot disk are provided. A boot disk can be imported during the process, or an ISO can be used to boot and install the OS
- A default flavor, representing CPU and memory allotments, is provided
- Workload type determines KVM optimization to balance between performance and efficiency

Create Virtual Machine template

1 General

2 Networking

3 Storage

4 Advanced

5 Review

6 Result

1 Name *

Template provider *

example: your company name

Template support

No additional support

Description

2 Operating System *

--- Select Operating System ---

3 Boot Source *

--- Select Source ---

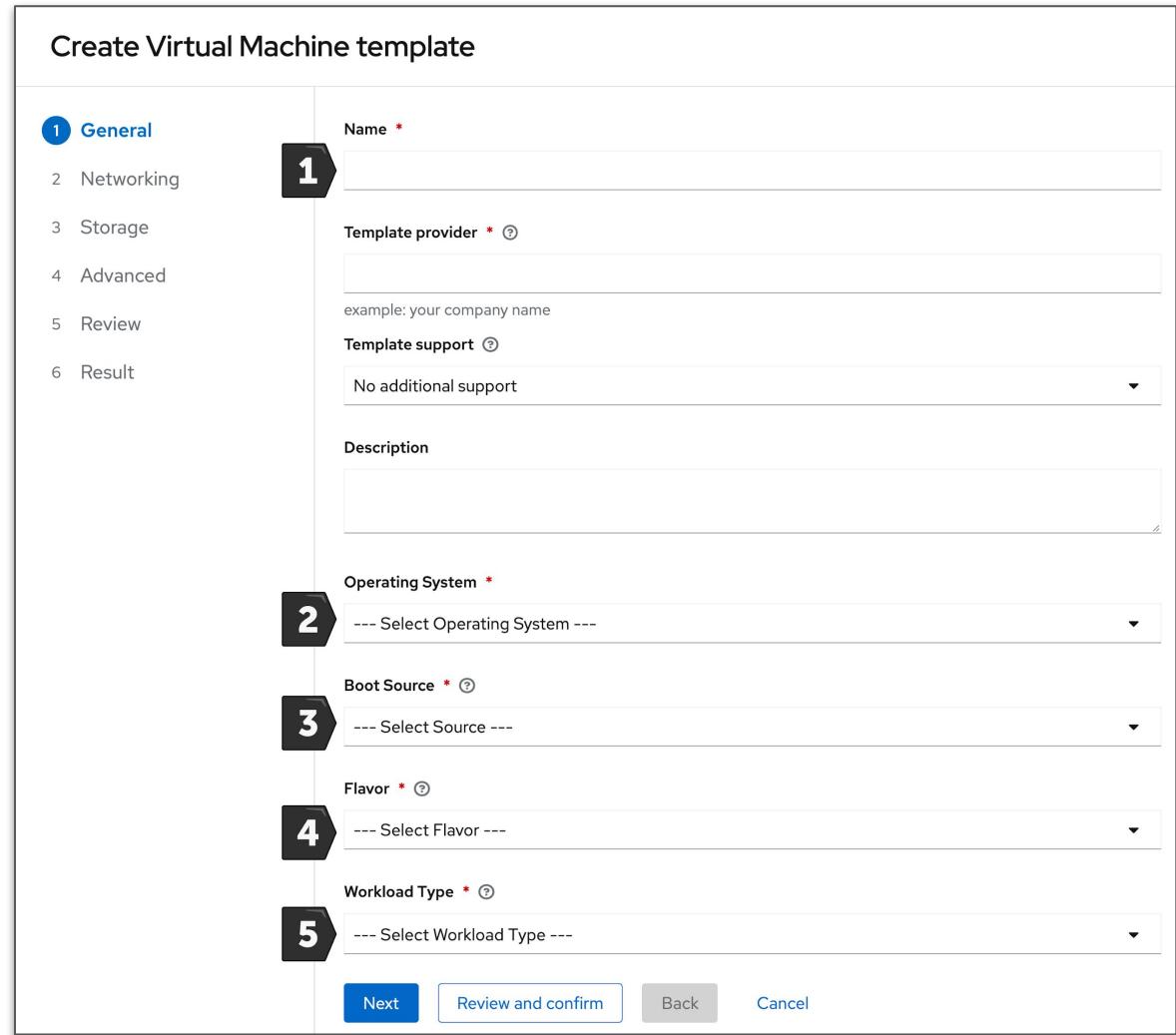
4 Flavor *

--- Select Flavor ---

5 Workload Type *

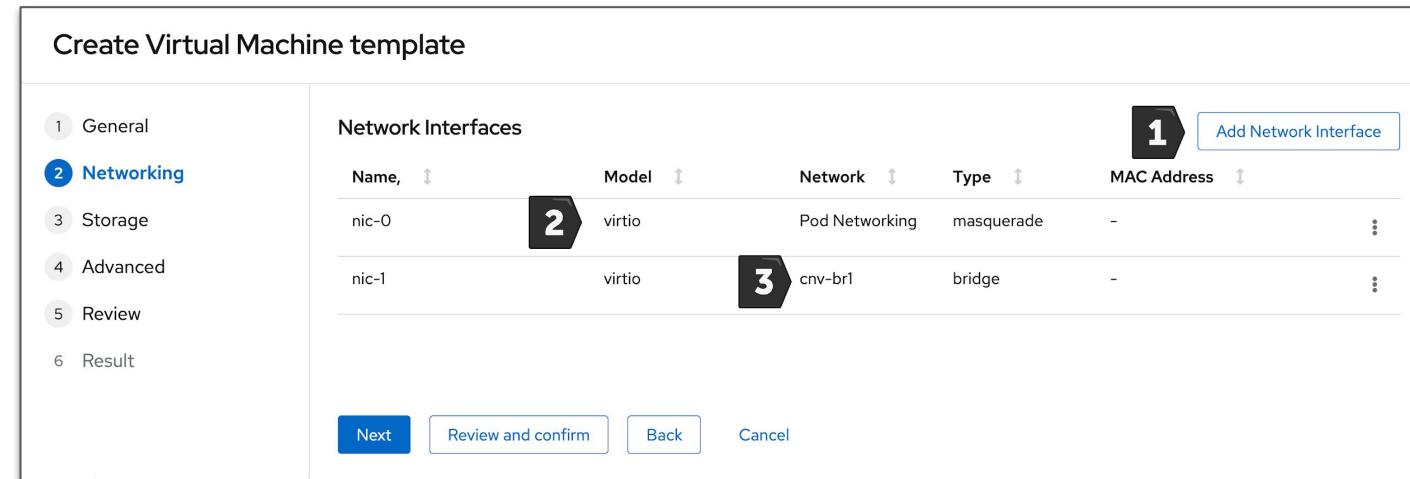
--- Select Workload Type ---

Next Review and confirm Back Cancel



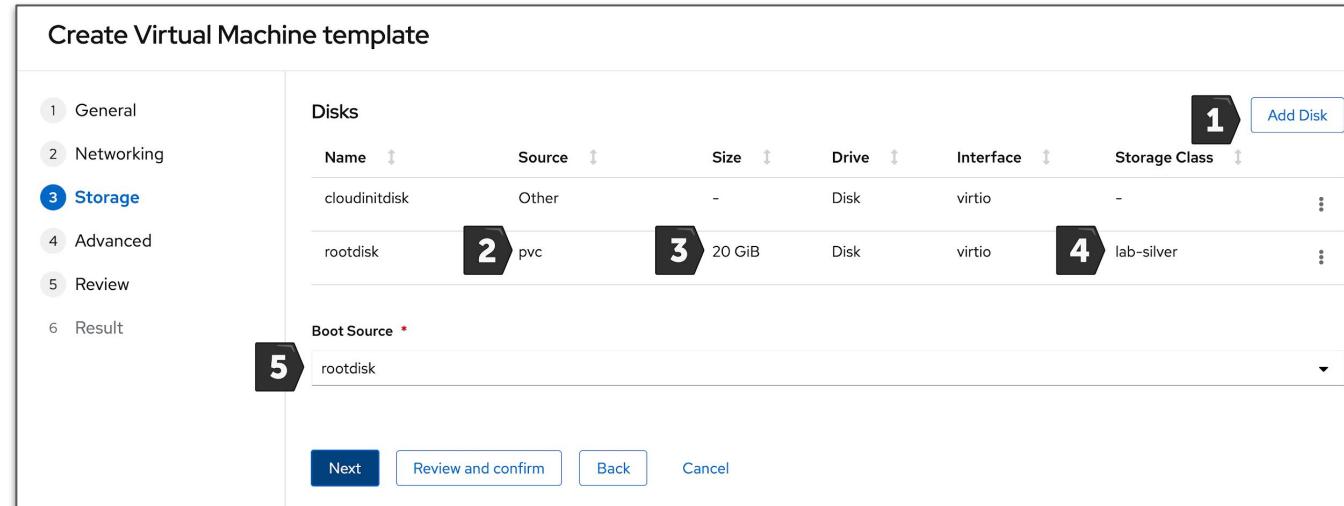
Create a template - Networks

- Add or edit network adapters
- One or more network connections
 - Pod network for the default SDN
 - Additional multus-based interfaces for specific connectivity
- Multiple NIC models for guest OS compatibility or paravirtualized performance with VirtIO
- Masquerade, bridge, or SR-IOV connection types
- MAC address customization if desired



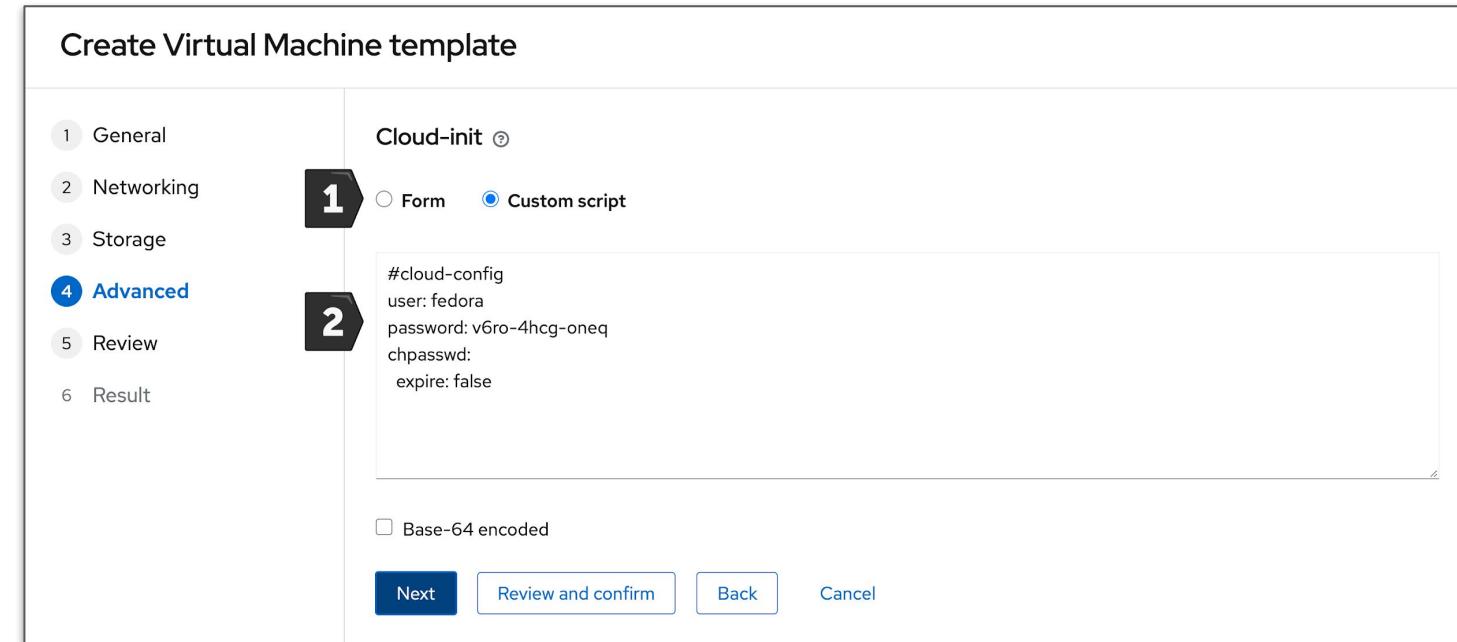
Create a template - Storage

- Add or edit persistent storage
- Disks can be sourced from
 - Imported QCOW2 or raw images
 - New or existing PVCs
 - Clone existing PVCs
- Use SATA/SCSI interface for compatibility or VirtIO for paravirtual performance
- For new or cloned disks, select from available storage classes
 - Customize volume and access mode as needed
 - RWX PVCs are required for live migration



Create a template - Advanced

- Customize the operating system deployment using cloud-init scripts
 - Guest OS must have cloud-init installed
 - RHEL, Fedora, etc. cloud images
 - Default templates will auto-generate a simple **cloud-init** to set the password



Import VMs

Virtual Machine Import

- Wizard supports importing from VMware or Red Hat Virtualization
 - Single-VM workflow
- VMware import uses VDDK to expedite the disk import process
 - User is responsible for downloading the VDDK from VMware and adding it to a container image
- Credentials stored as Secrets
- **ResourceMapping** CRD configures default source -> destination storage and network associations

Import Virtual Machine

1 Connect to Provider

2 Review

3 Result

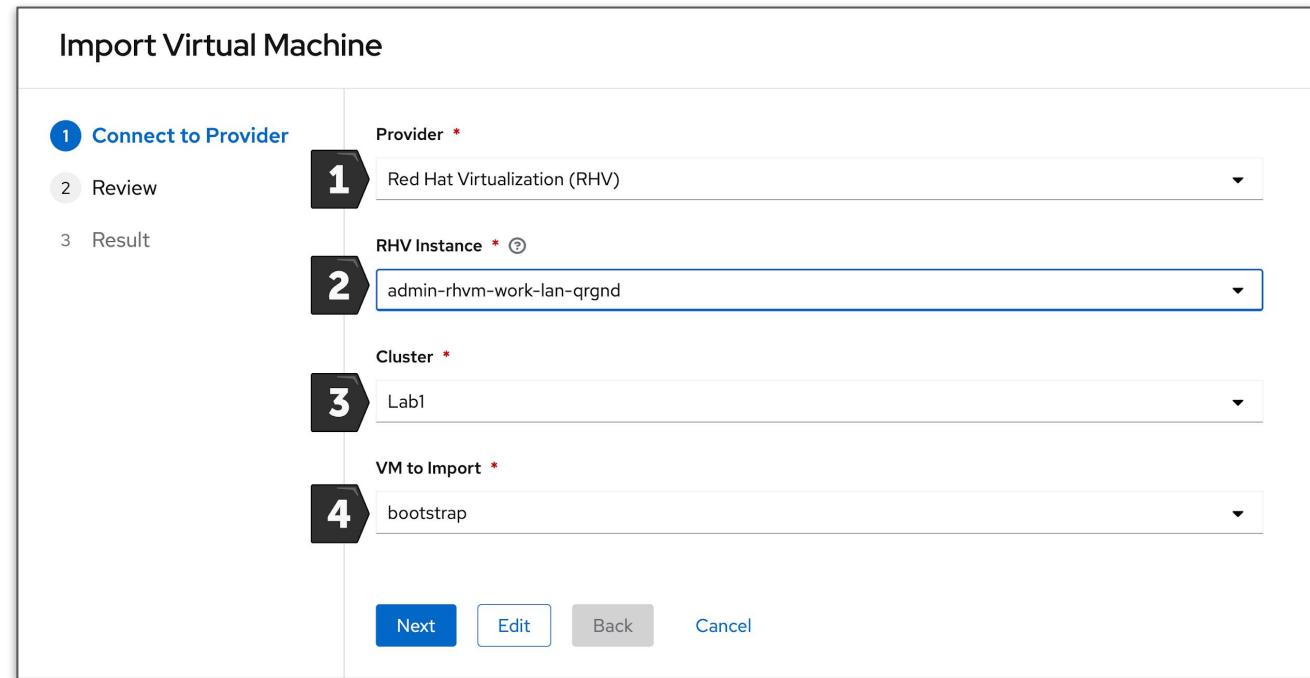
Provider * 1 Red Hat Virtualization (RHV)

RHV Instance * 2 admin-rhvm-work-lan-qrgnd

Cluster * 3 Lab1

VM to Import * 4 bootstrap

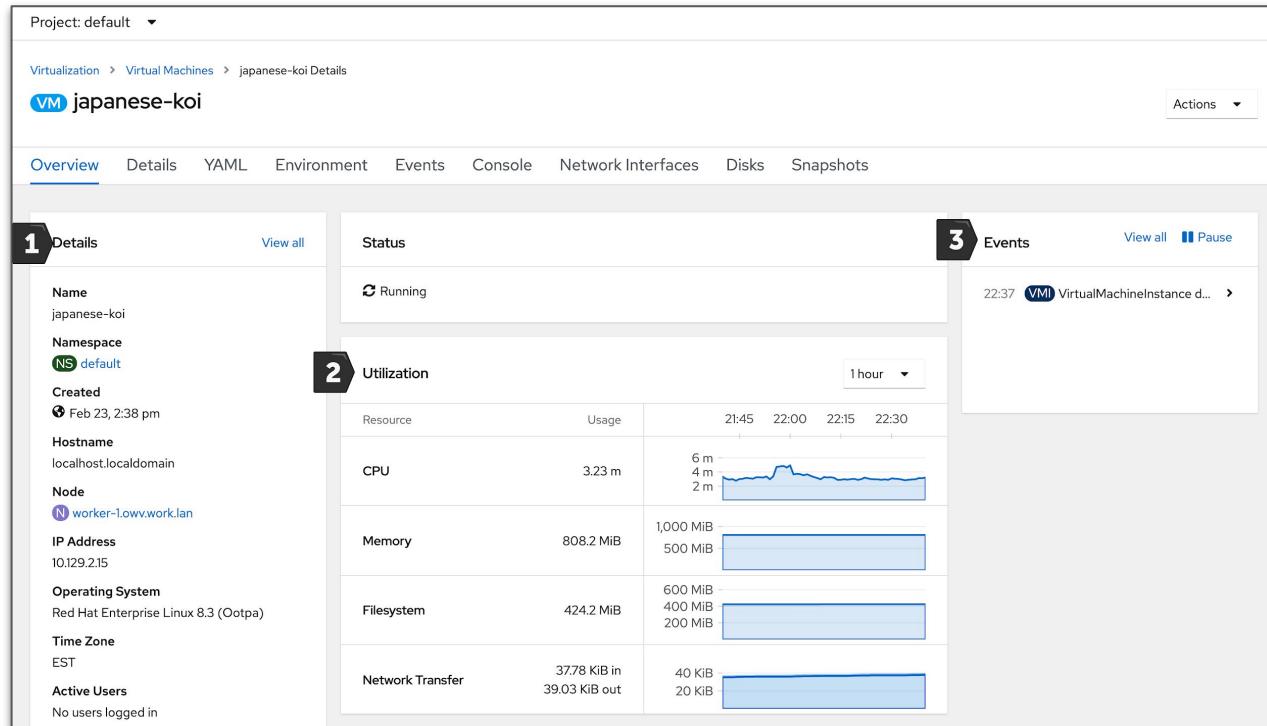
Next Edit Back Cancel



View / manage VMs

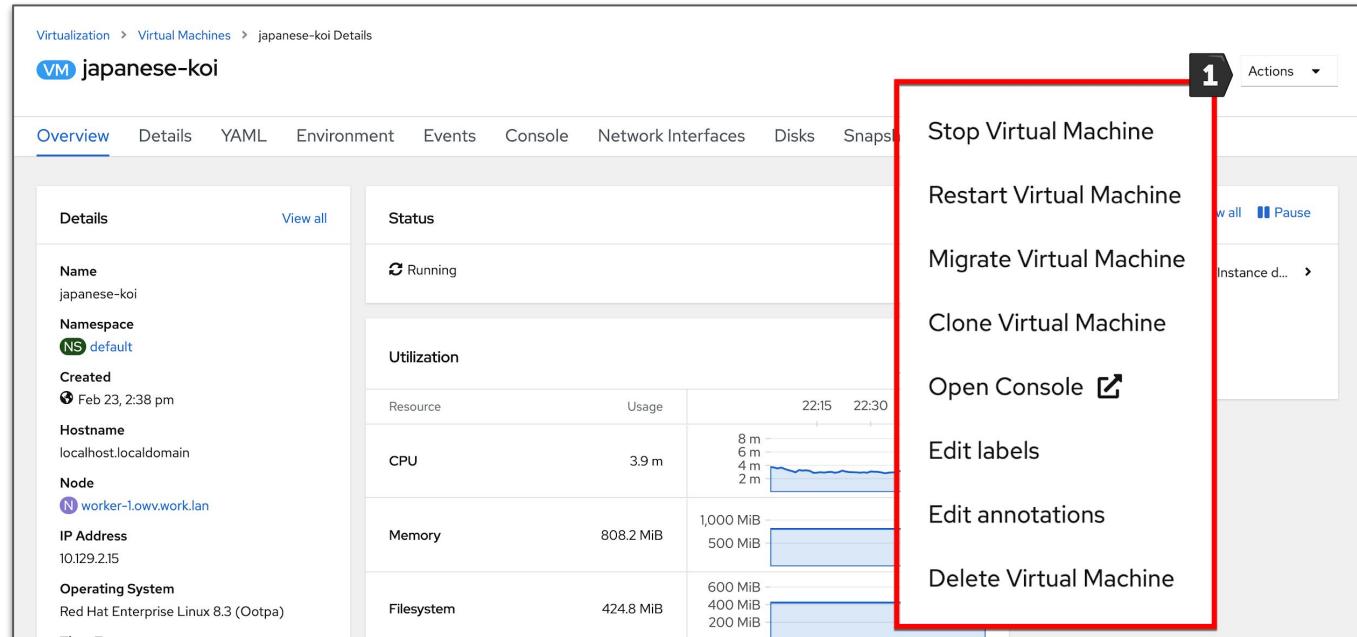
Virtual Machine - Overview

- General overview about the virtual machine
- Information populated from guest when integrations are available
 - IP address
- Inventory quickly shows configured hardware with access to view/manage
- Utilization reporting for CPU, RAM, disk, and network
- Events related to the Pod, scheduling, and resources are displayed



Virtual Machine - Actions

- Actions menu allows quick access to common VM tasks
 - Start/stop/restart
 - Live migration
 - Clone
 - Edit application group, labels, and annotations
 - Delete
- Accessible from all tabs of VM details screen and the VM list



The screenshot shows the 'japanese-koi' VM details page. The 'Actions' menu is open, displaying the following options:

- Stop Virtual Machine
- Restart Virtual Machine
- Migrate Virtual Machine
- Clone Virtual Machine
- Open Console
- Edit labels
- Edit annotations
- Delete Virtual Machine

Virtual Machine - Details

- Details about the virtual machine
 - Labels, annotations
 - Configured OS
 - Template used, if any
 - Configured boot order
 - Associated workload profile
 - Flavor
- Additional details about scheduling
 - Node selector, tolerations, (anti)affinity rules
- Services configured for the VM

Project: default

Virtualization > Virtual Machines > rhel Details

VM rhel

Overview Details YAML Environment Events Console Network Interfaces Disks Actions

Virtual Machine Details

1 Labels
app-rhel flavor:template.kubevirt.io/small=true ostemplate.kubevirt.io/rhel8.2=true
vm.kubevirt.io/template=rhs8-desktop-small-v0.10.0
vm.kubevirt.io/template.namespace=openshift
vm.kubevirt.io/template.revision=1
vm.kubevirt.io/template.version=v0.11.2 workload.template.kubevirt.io/desktop=true

2 Operating System
Red Hat Enterprise Linux 8.0 or higher

3 Template
Not available

4 Boot Order
1.rootdisk (Disk)

CD-ROMs
Not available

5 Workload Profile
desktop

6 Flavor
Small:1 vCPU, 2 GiB Memory

Dedicated Resources
No Dedicated resources applied

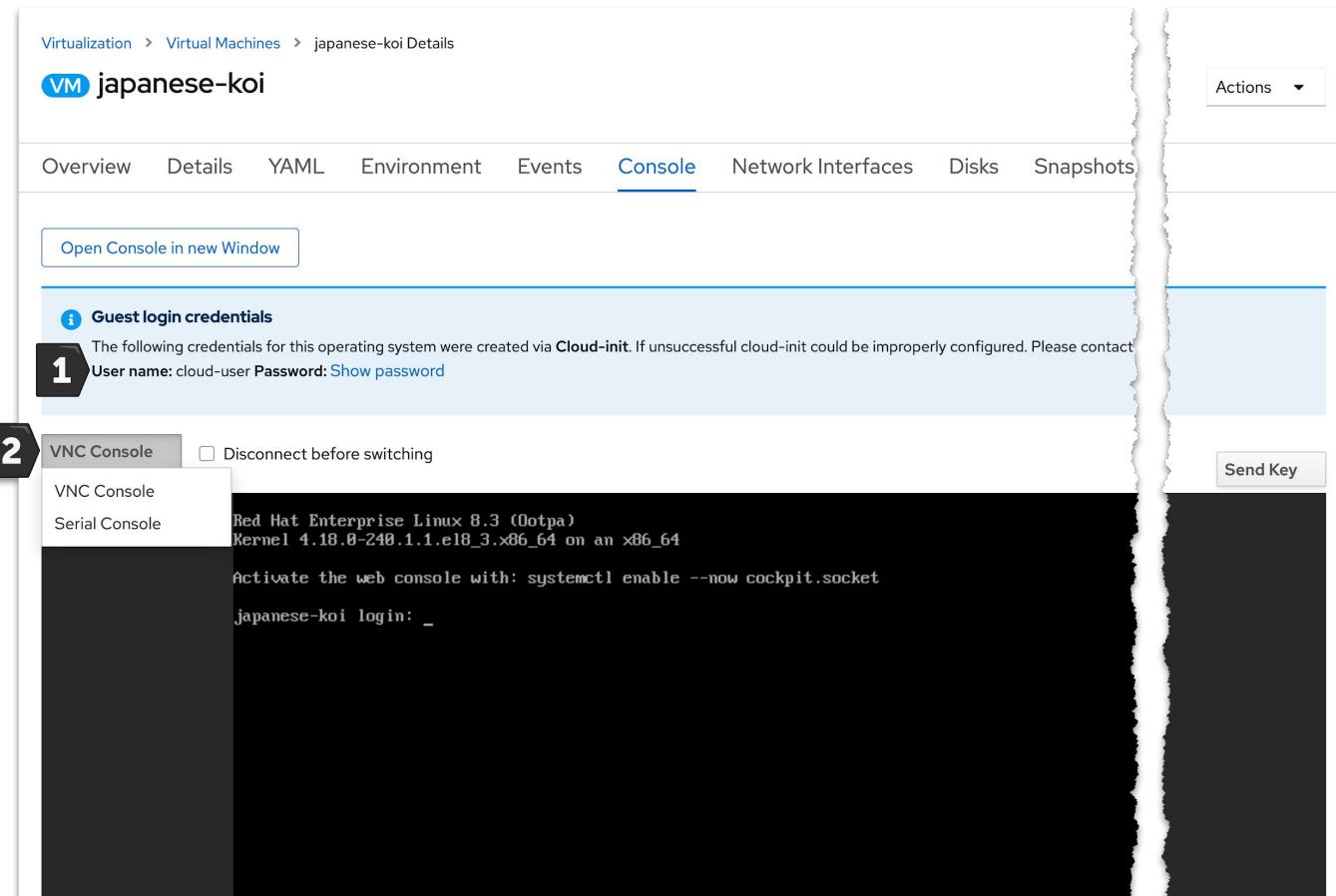
7 Scheduling and resources requirements

- Node Selector: No selector
- Tolerations: No Tolerations rules
- Affinity Rules: No Affinity rules

8 Services
No Services Found

Virtual Machine - Console

- Browser-based access to the serial and graphical console of the virtual machine
- Access the console using native OS tools, e.g. `virt-viewer`, using the `virtctl` CLI command
 - `virtctl console vmname`
 - `virtctl vnc vmname`



Virtual Machine - Disks and NICs

- Add, edit, and remove NICs and disks for non-running virtual machines

The screenshot shows a virtual machine configuration interface with two main sections: Network Interfaces and Disks.

Network Interfaces Section:

- 1** Add Network Interface
- 2** Network: Pod Networking
- 3** Type: masquerade

Name	Model	Network	Type	MAC Address
default	virtio	Pod Networking	masquerade	-

Disks Section:

- 1** Add Disk
- 2** Disk: rootdisk
- 3** Storage Class: lab-silver

Name	Source	Size	Drive	Interface	Storage Class
cloudinitdisk	Other	-	Disk	virtio	-
rootdisk	pvc	20 GiB	Disk	virtio	lab-silver

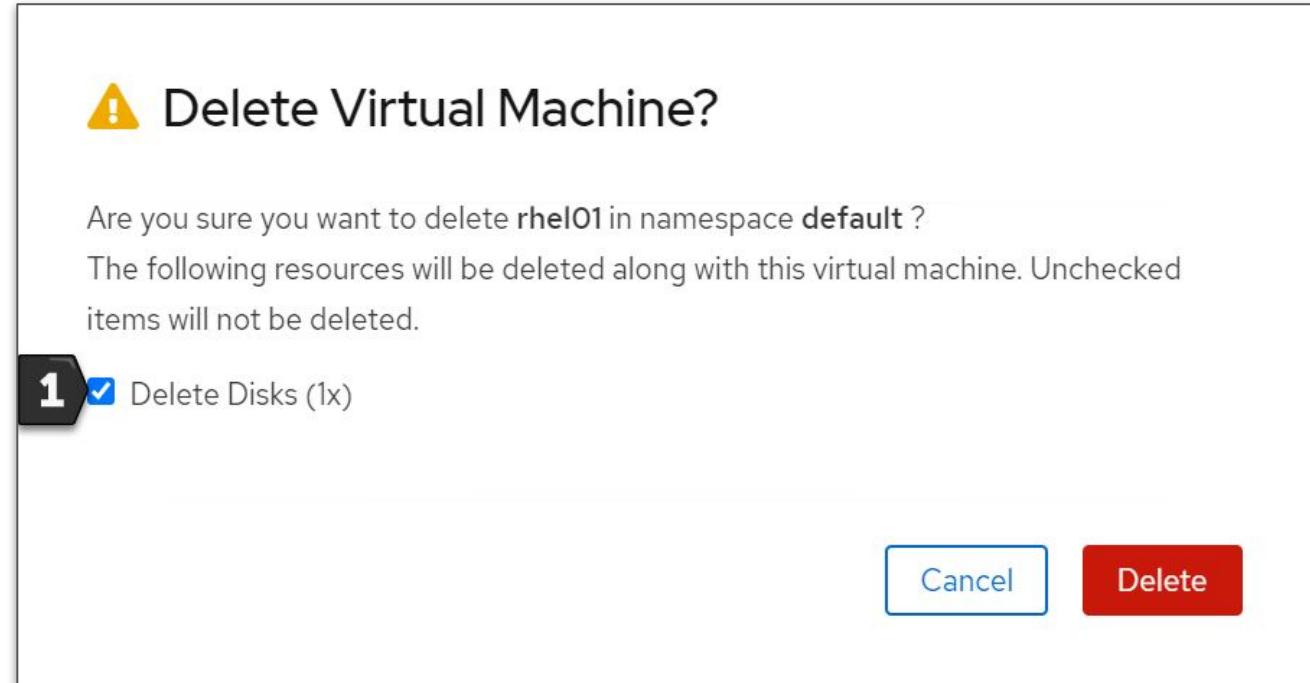
File Systems Section:

Name	File System Type	Mount Point	Total Bytes	Used Bytes
vdb2	vfat	/boot/efi	99.79 MiB	6.84 MiB
vdb3	xfs	/	19.89 GiB	1.41 GiB

Destroy VMs

Destroying a Virtual Machine

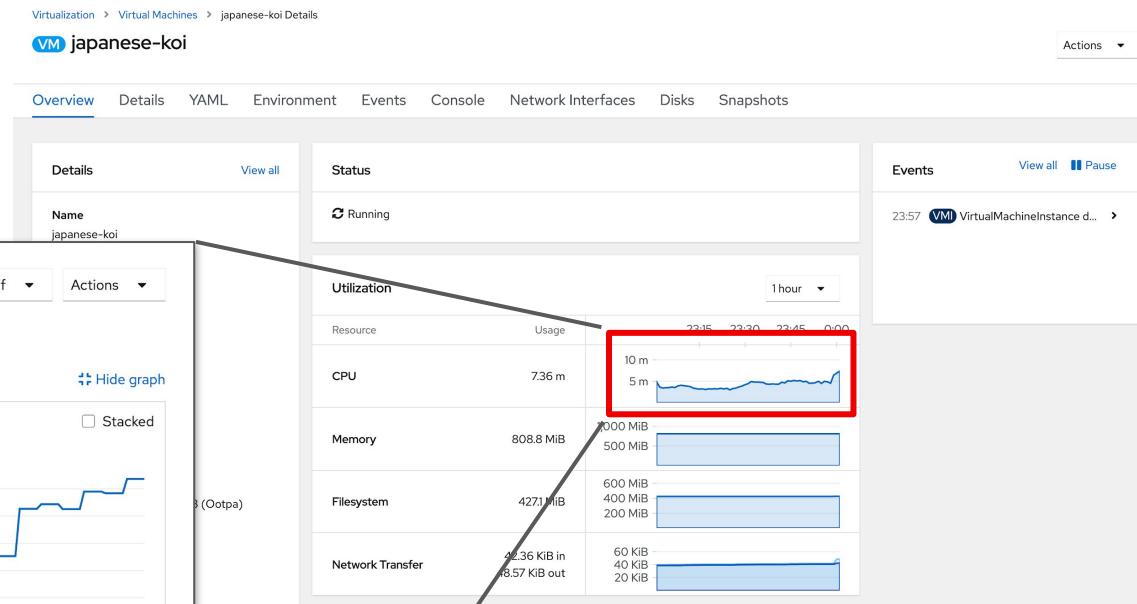
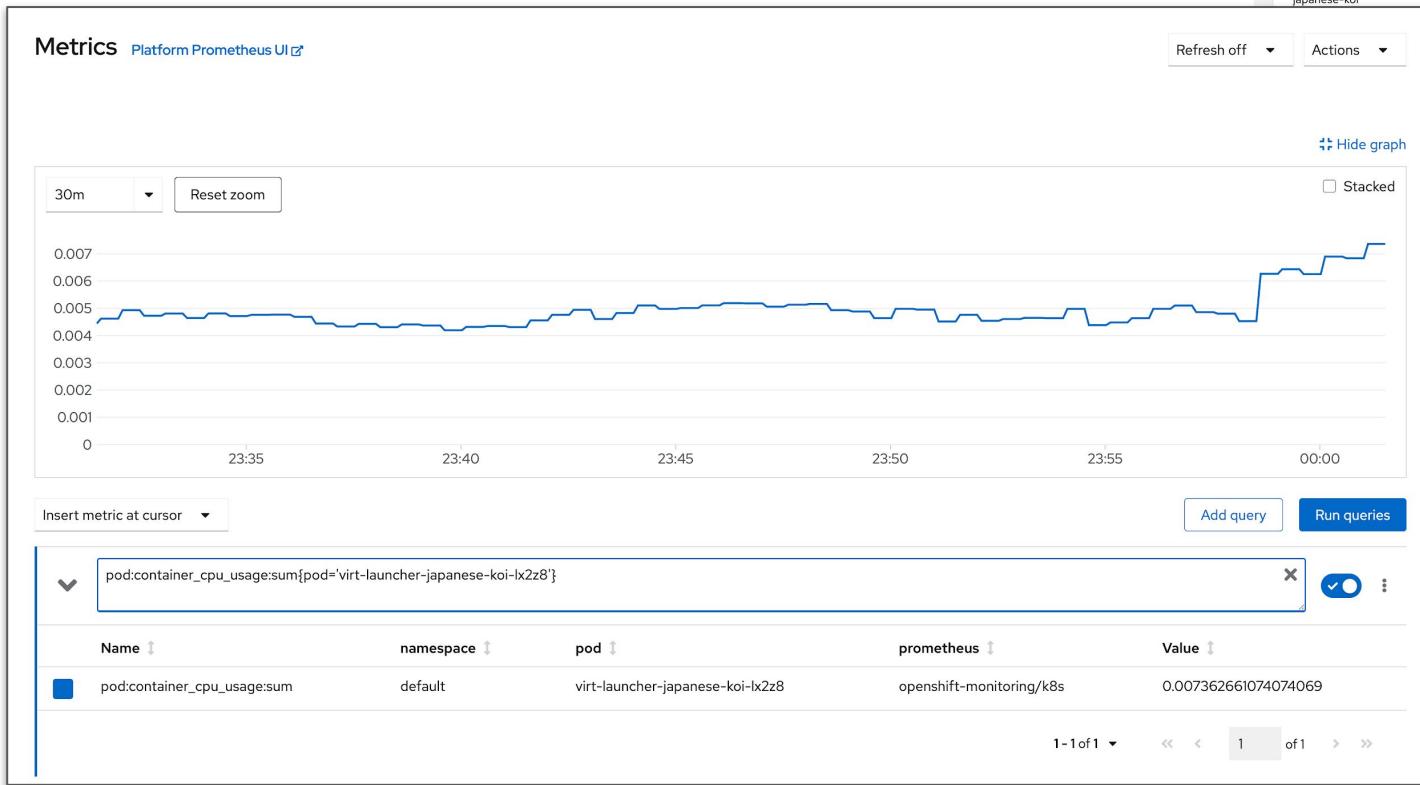
- Deleting a VM removes the VM definition
 - Optionally delete PVC-backed disks associated with the VM
- Running VMs are terminated first
- Other associated resources, e.g. Services, are not affected



Metrics

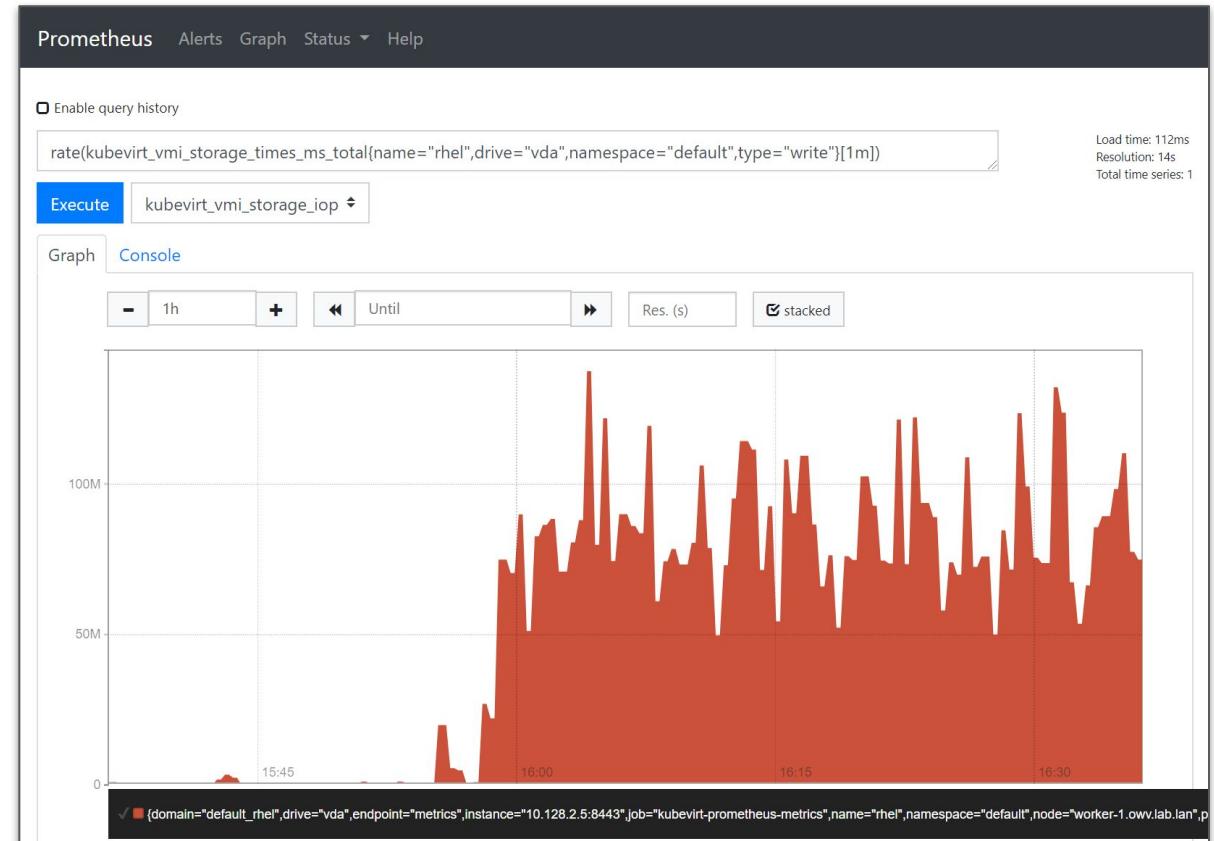
Overview Virtual Machine metrics

- Summary metrics for 1, 6, and 24 hour periods are quickly viewable from the VM overview page
- Clicking a graph will display it enlarged in the metrics UI



Detailed Virtual Machine metrics

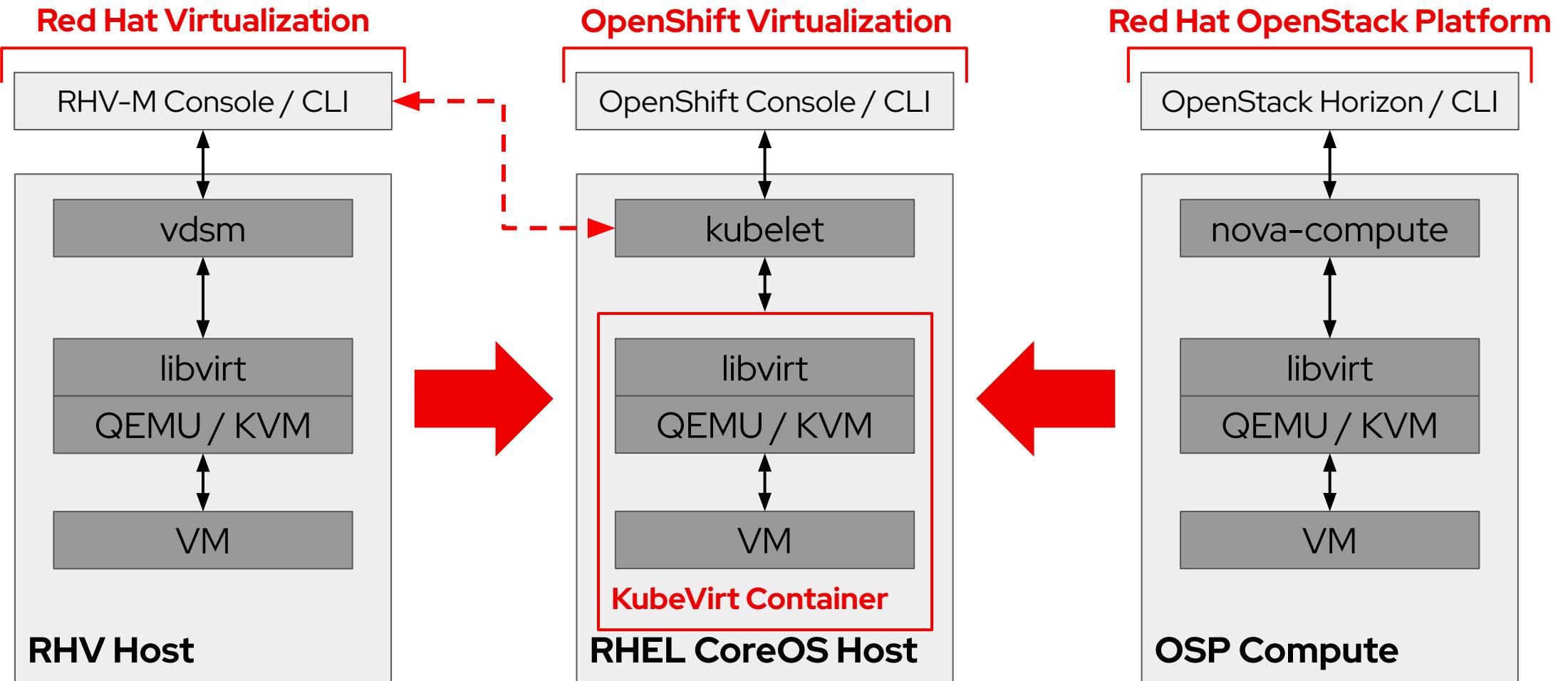
- Virtual machine, and VM pod, metrics are collected by the OpenShift metrics service
 - Available under the `kubevirt` namespace in Prometheus
- Available per-VM metrics include
 - Active memory
 - Active CPU time
 - Network in/out errors, packets, and bytes
 - Storage R/W IOPS, latency, and throughput
- VM metrics are for VMs, not for VM pods
 - Management overhead not included in output
 - Look at `virt-launcher` pod metrics for
- No preexisting Grafana dashboards



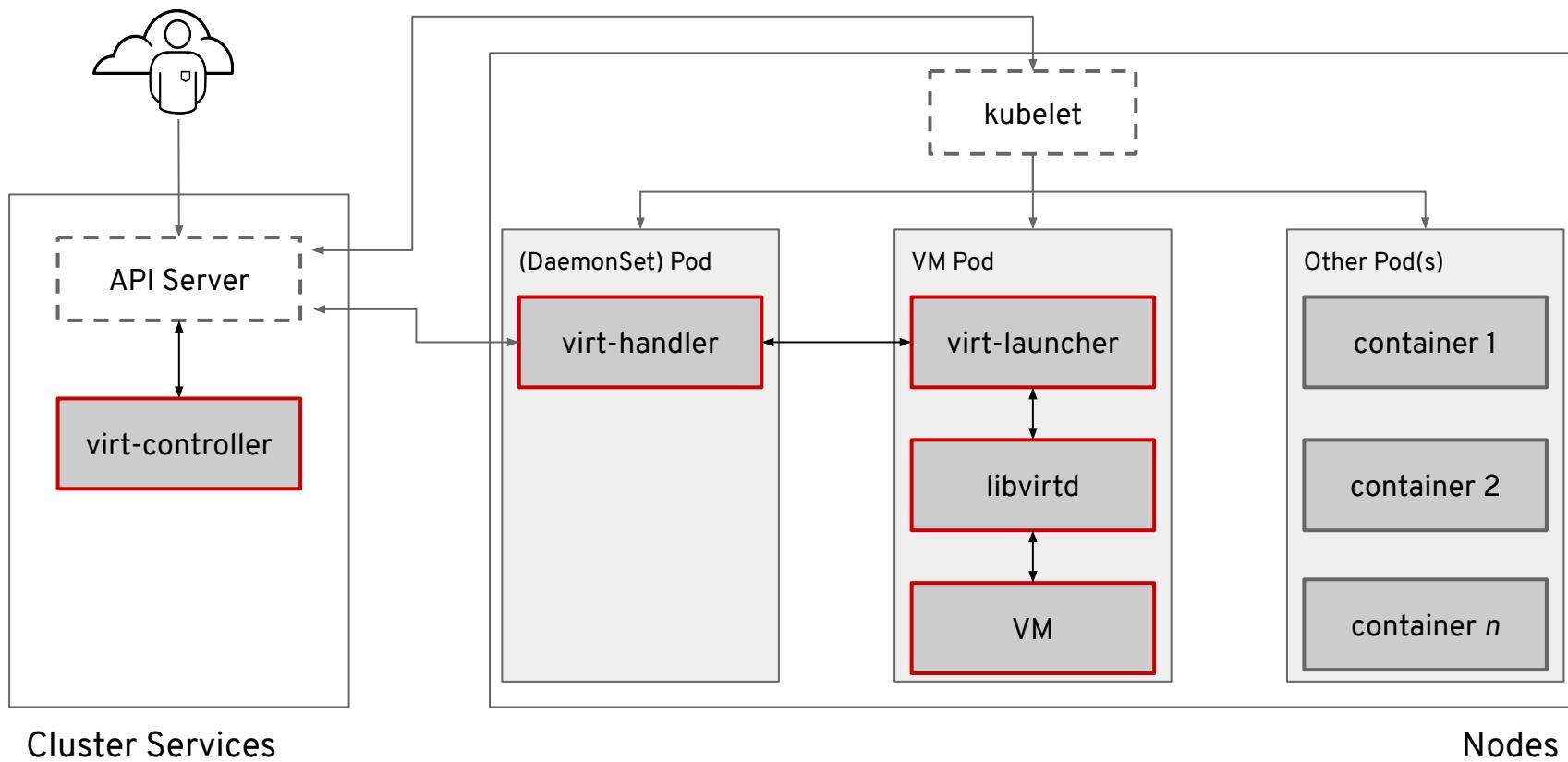
Deeper into the technology

Containerizing KVM

Trusted, mature KVM wrapped in modern management and automation



Architectural Overview



Terminology comparison

Feature	RHV	OpenShift Virtualization	vSphere
Where VM disks are stored	Storage Domain	PVC	datastore
Policy based storage	None	StorageClass	SPBM
Non-disruptive VM migration	Live migration	Live migration	vMotion
Non-disruptive VM storage migration	Storage live migration	N/A	Storage vMotion
Active resource balancing	Cluster scheduling policy	Pod eviction policy, descheduler	Dynamic Resource Scheduling (DRS)
Physical network configuration	Host network config (via nmstate w/4.4)	nmstate Operator, Multus	vSwitch / DvSwitch
Overlay network configuration	OVN	OCP SDN (OpenShiftSDN, OVNKubernetes, and partners), Multus	NSX-T
Host / VM metrics	Data warehouse + Grafana (RHV 4.4)	OpenShift Metrics, health checks	vCenter, vROps

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

 [linkedin.com/company/red-hat](https://www.linkedin.com/company/red-hat)

 [youtube.com/user/RedHatVideos](https://www.youtube.com/user/RedHatVideos)

 [facebook.com/redhatinc](https://www.facebook.com/redhatinc)

 twitter.com/RedHat