



OpenShift AddOns

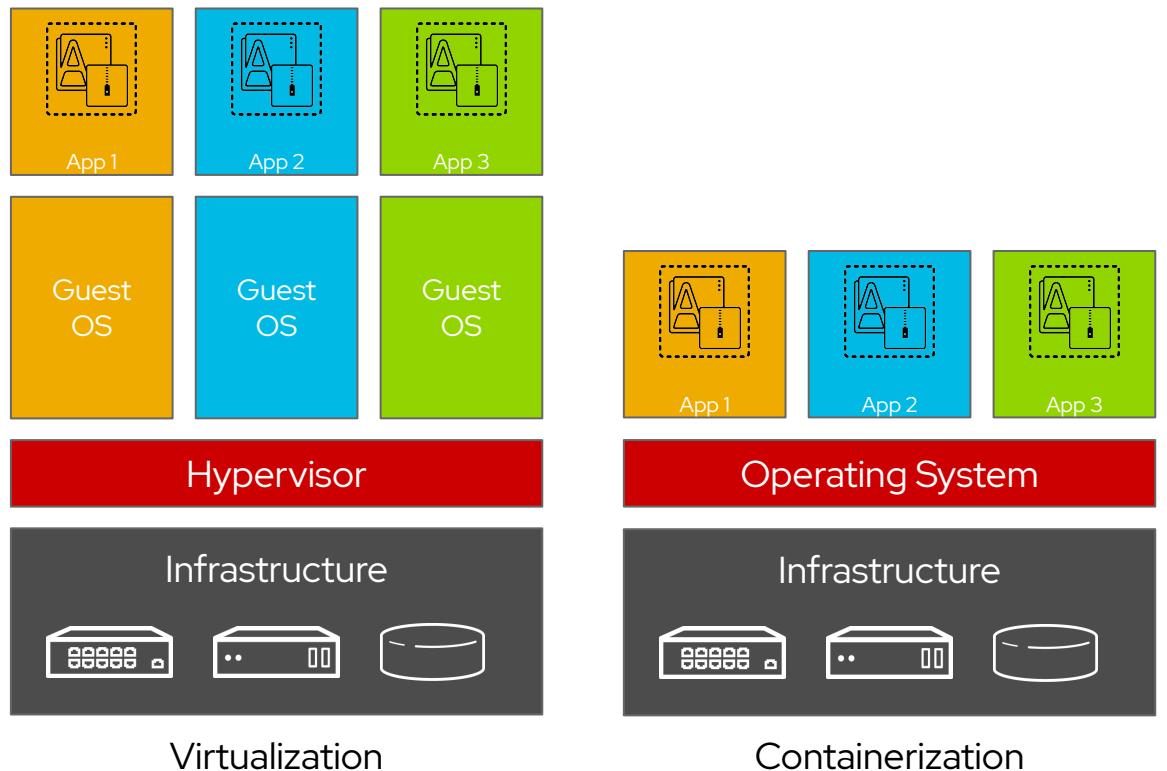


OpenShift Virtualization

Technical presentation

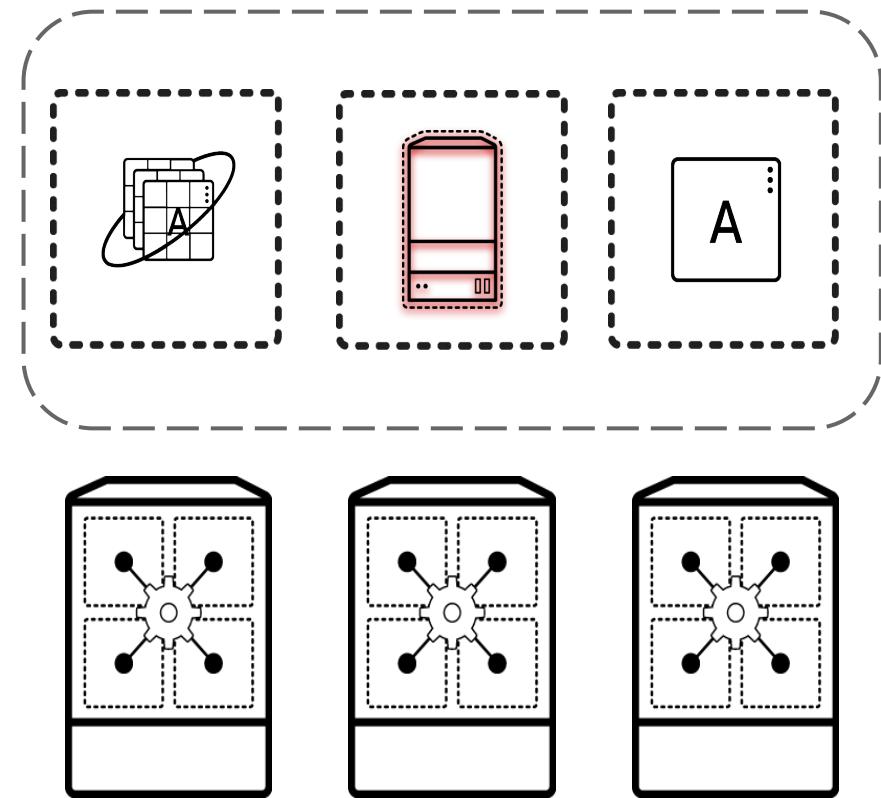
Containers are not virtual machines

- 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



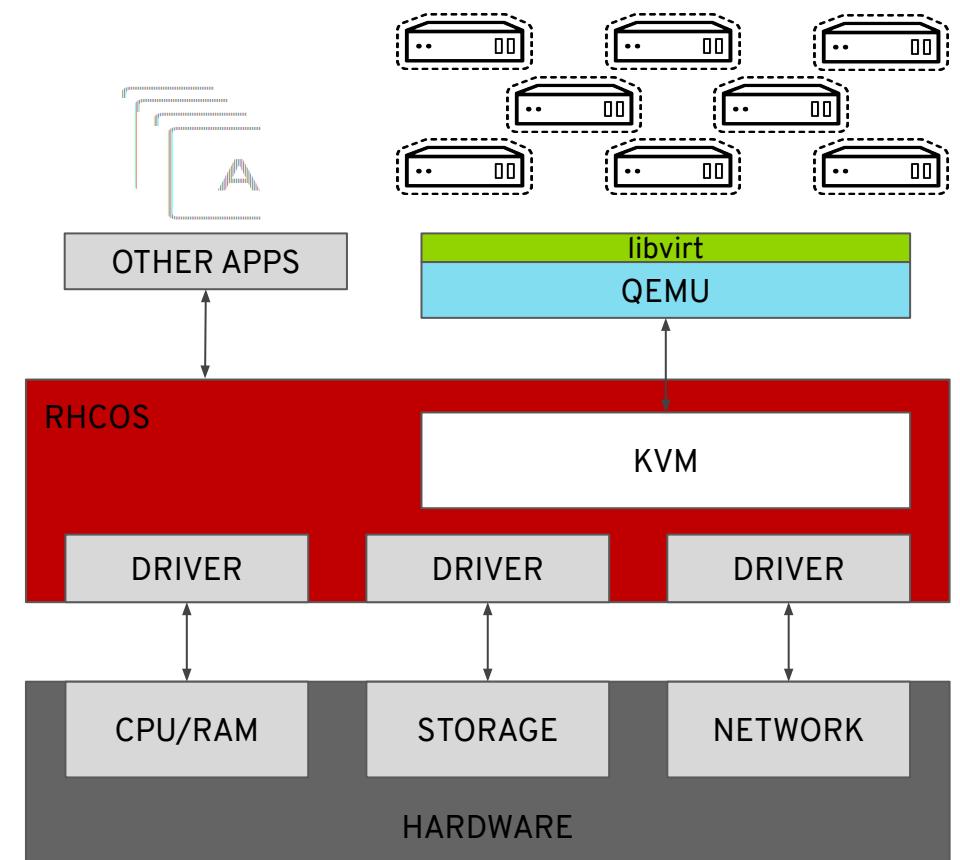
OpenShift Virtualization

- 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

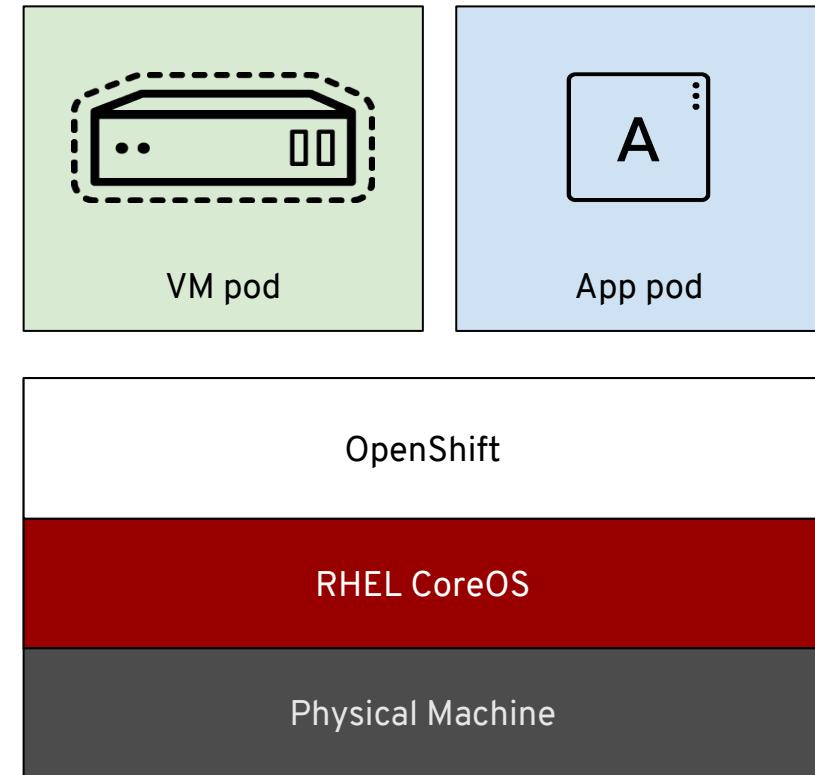
- OpenShift Virtualization uses KVM, the Linux kernel hypervisor
- KVM is a core component of the Red Hat Enterprise Linux kernel
 - KVM has 10+ years of production use: Red Hat Virtualization, Red Hat OpenStack Platform, and RHEL all leverage KVM, QEMU, and libvirt
- QEMU uses KVM to execute virtual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



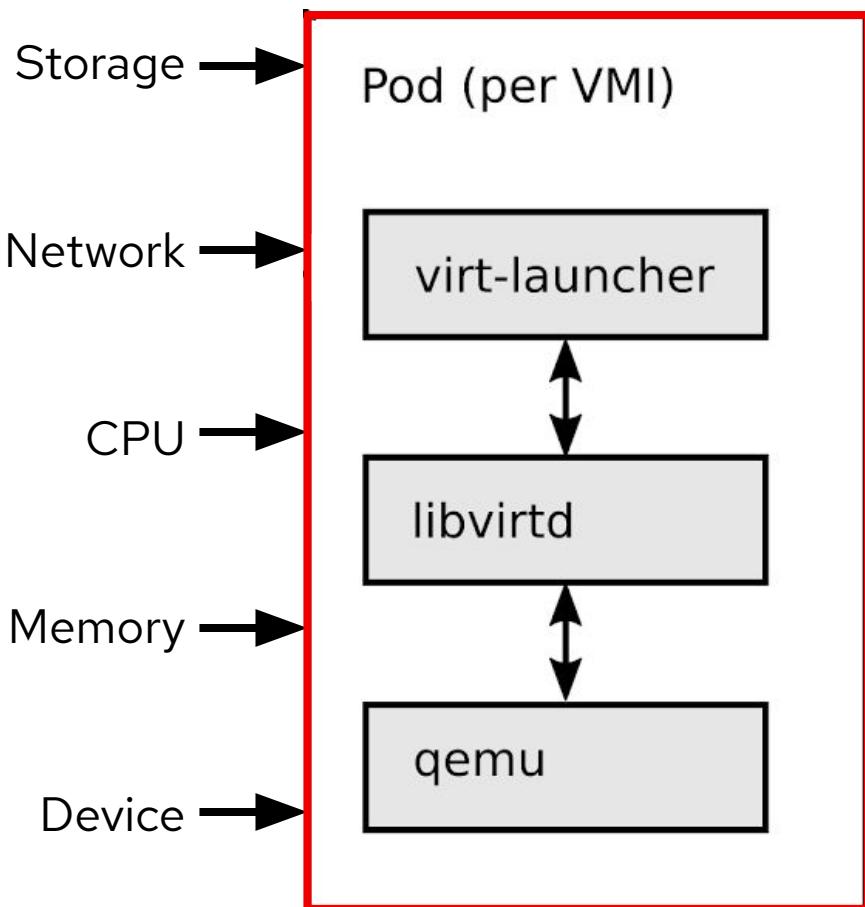
Built with Kubernetes

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



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

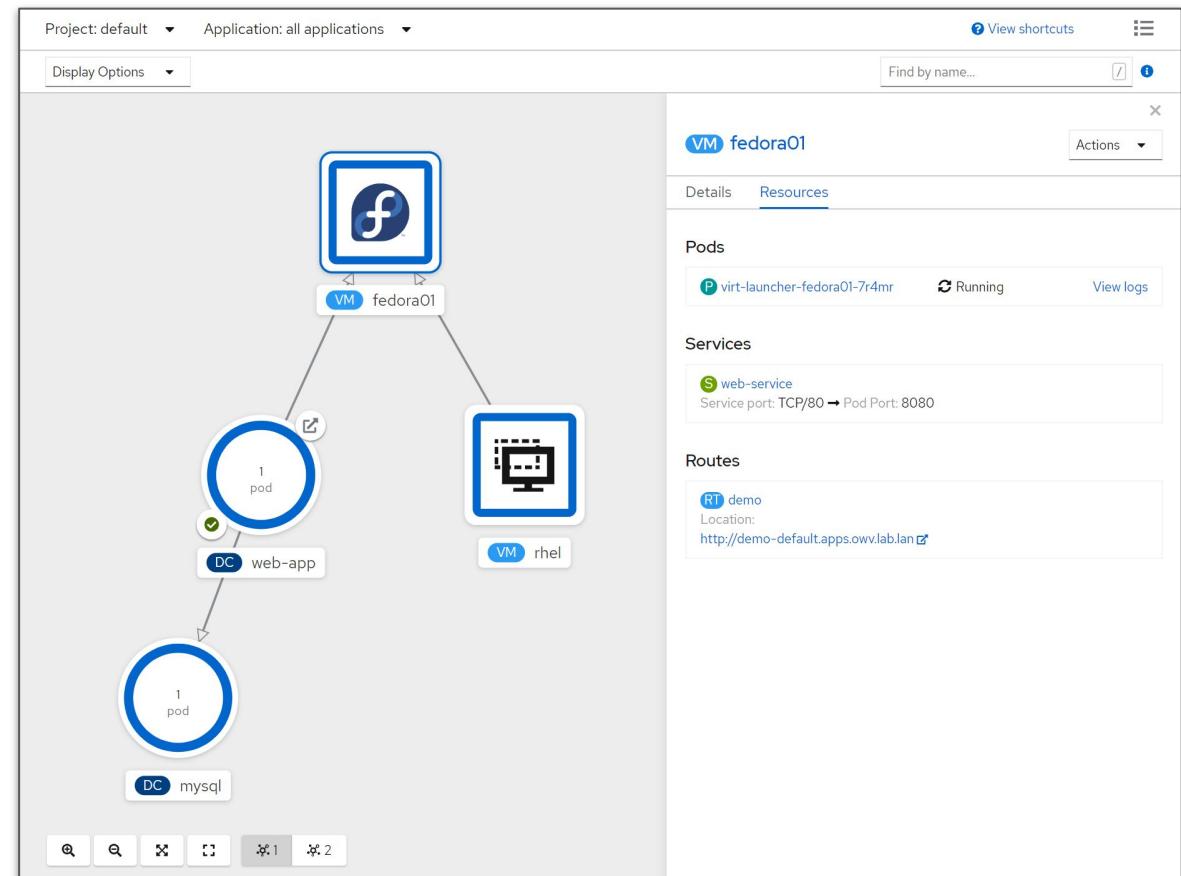
- libvirt and qemu from RHEL are mature, have high performance, provide stable abstractions, and have a minimal overhead.

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

The screenshot shows the Red Hat OpenShift Container Platform web interface. The left sidebar has a dark theme with white text. It includes sections for Administrator, Home, Operators, Workloads (with sub-options like Pods, Virtualization, Deployments, DeploymentConfigs, StatefulSets, Secrets, ConfigMaps, CronJobs, Jobs, DaemonSets, ReplicaSets, ReplicationControllers, and HorizontalPodAutoscalers), and Networking. The 'Virtualization' option under Workloads is currently selected and highlighted in blue. The main content area is titled 'Virtualization' and contains a table titled 'Virtual Machines'. The table lists five virtual machines with the following details:

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

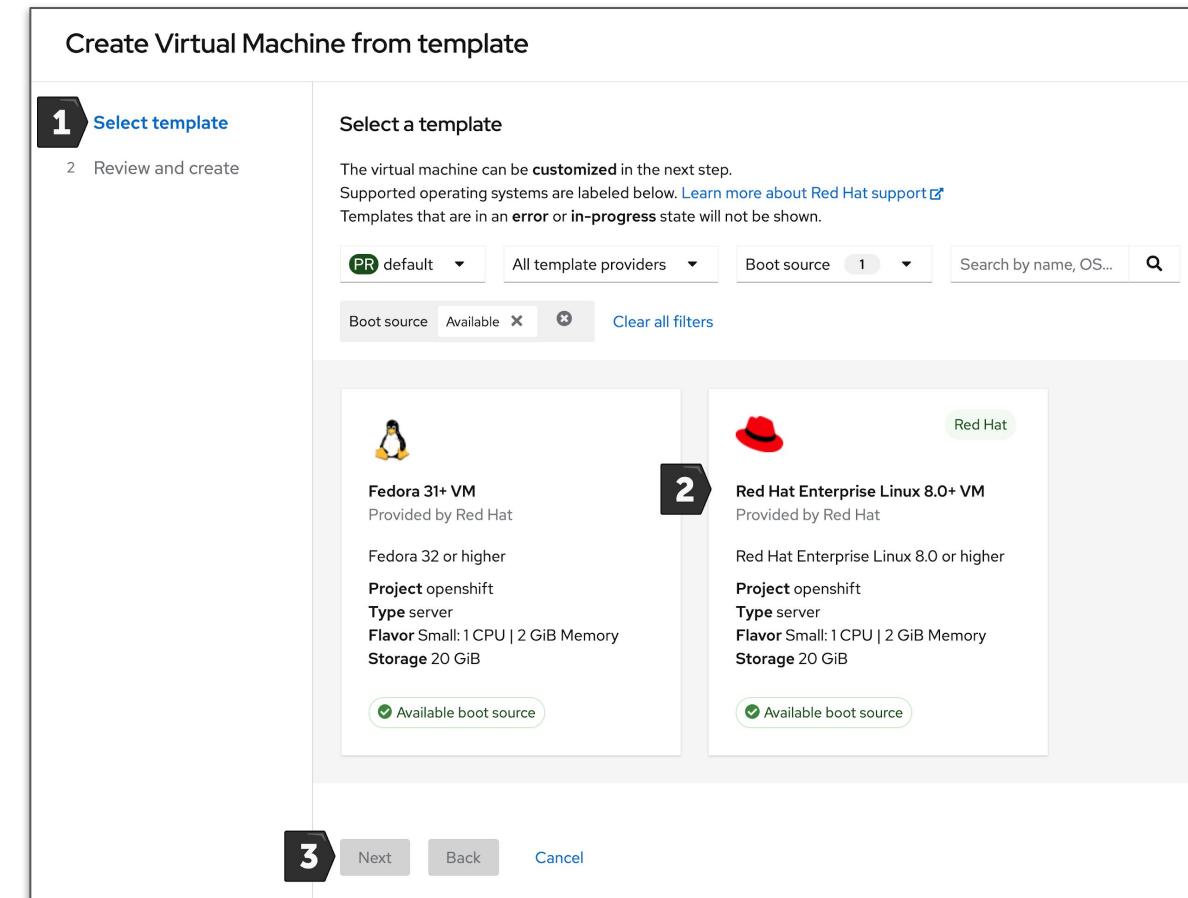
- Streamlined and simplified creation via the GUI or create VMs programmatically using YAML
- Full configuration options for compute, network, and storage resources
 - Clone VMs from templates or import disks using DataVolumes
 - Pre-defined and customizable presets for CPU/RAM allocations
 - Workload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

The screenshot shows the Red Hat OpenShift Container Platform web interface. The left sidebar has a dark theme with the Red Hat logo and "Red Hat OpenShift Container Platform". It includes links for "Administrator", "Home", "Operators", "Workloads" (with sub-options like "Pods", "Virtualization", "Deployments", "DeploymentConfigs", "StatefulSets", "Secrets", "ConfigMaps", "CronJobs", and "Jobs"), and "Templates". The main content area is titled "Virtualization" and "Virtual Machines". It lists several virtual machines with their names, namespaces, and IP addresses. A "Create" button is at the top right, with a dropdown menu showing two options: "Virtual Machine With Wizard" (marked with a red arrow) and "Virtual Machine With Import wizard" (marked with a red arrow). To the right of the dropdown, there is a "Template" section with "With Wizard" and "With YAML" options.

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 interface. At the top, there's a navigation bar with 'Project: default' and a 'Create' button. Below it, the title 'Virtualization' is followed by 'Virtual Machine' and a 'Templates' tab (which is highlighted). There are filters for 'Name' and a search bar. A note says 'Supported operating systems are labeled below. Learn more about Red Hat support'.

The main area displays a table of templates:

Name	Provider	Boot source	Actions
fedora-33 ★	CPTMM	CPTMM	Details Create ⋮
2 rhel-83 ★	3 CPTMM	CPTMM	Details Create ⋮
Fedora 31+ VM	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 ⋮

Five large black arrows with white numbers indicate a workflow:

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

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 ---

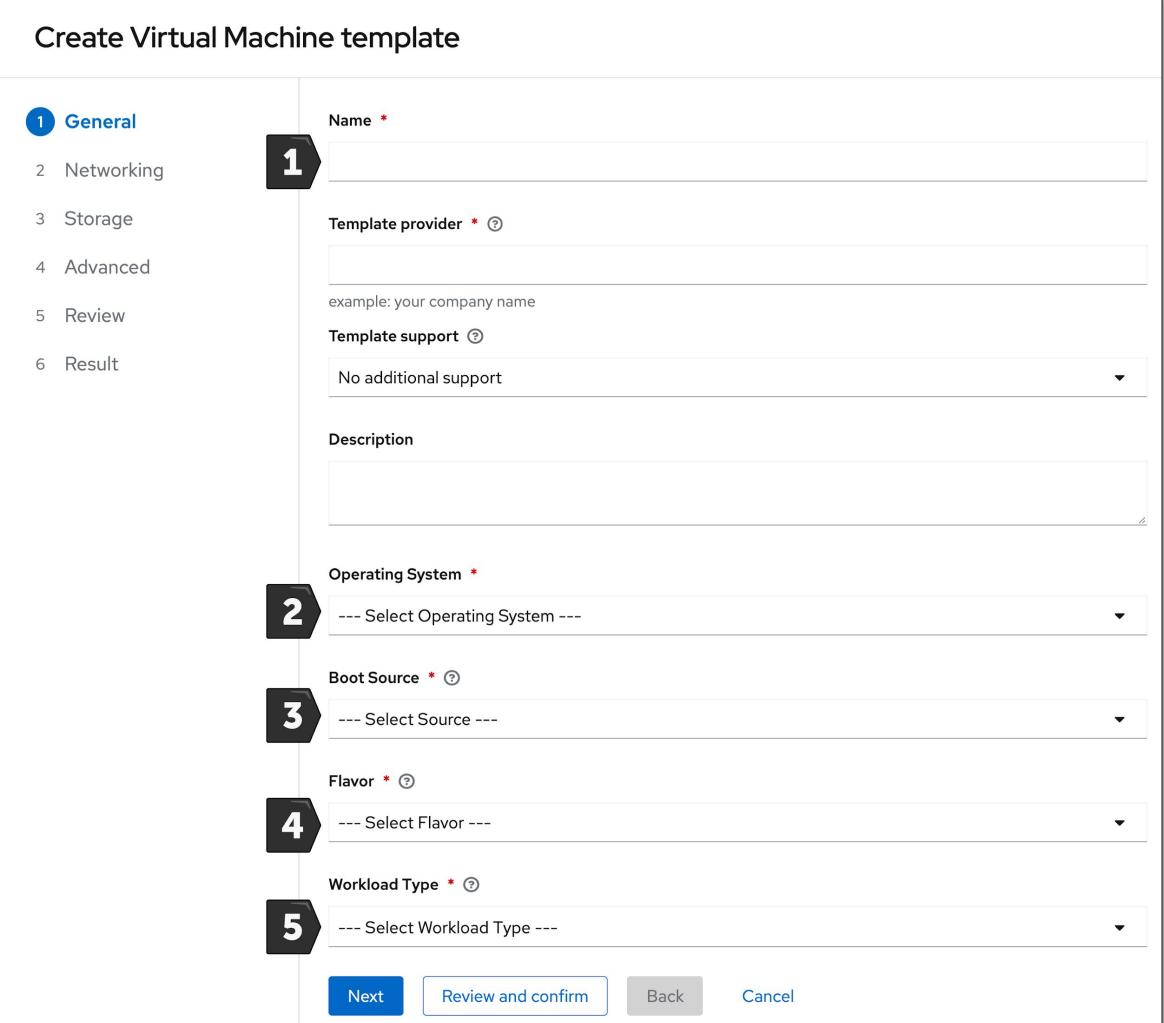
4 Flavor *

--- Select Flavor ---

5 Workload Type *

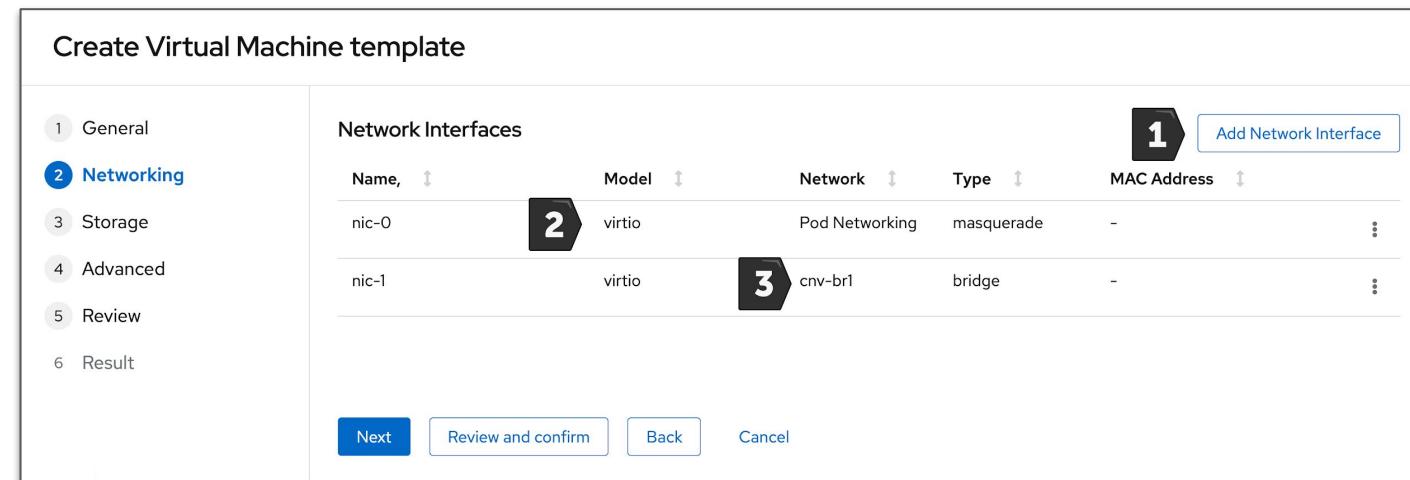
--- Select Workload Type ---

Next Review and confirm Back Cancel



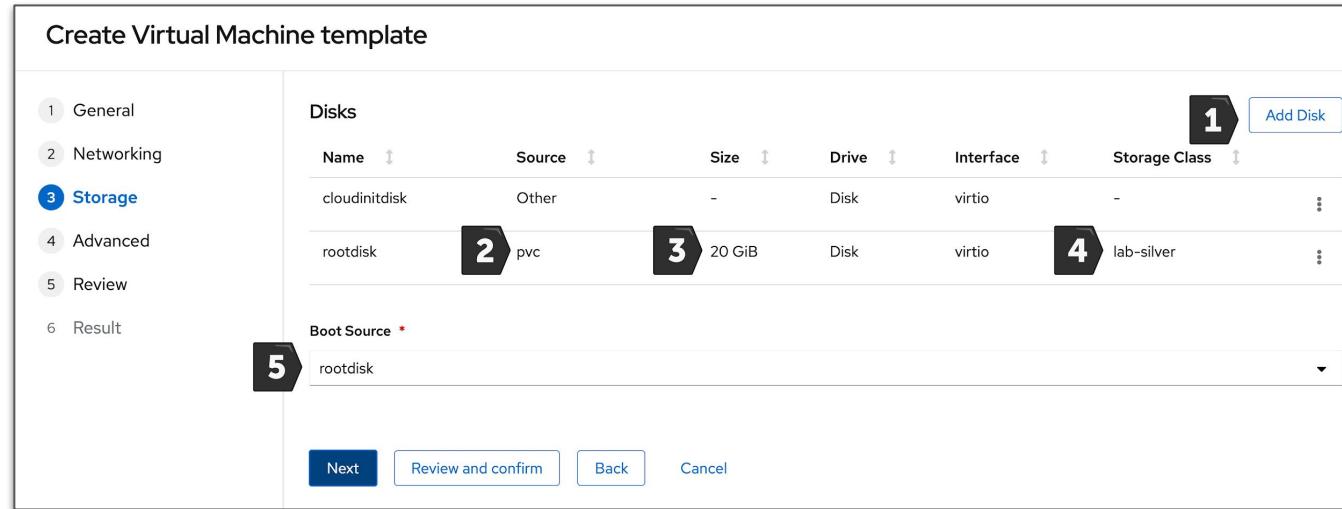
Create a template - Networks

- 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



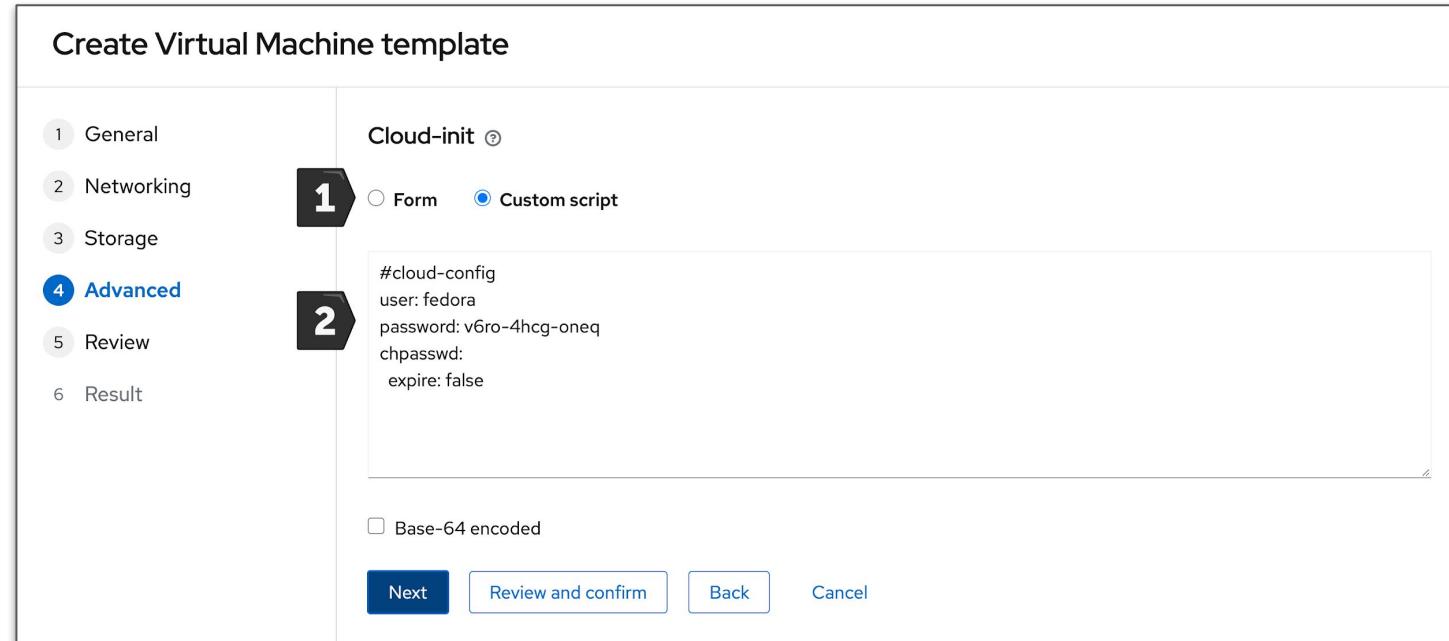
Create a template - Storage

- 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

Provider * Red Hat Virtualization (RHV)

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

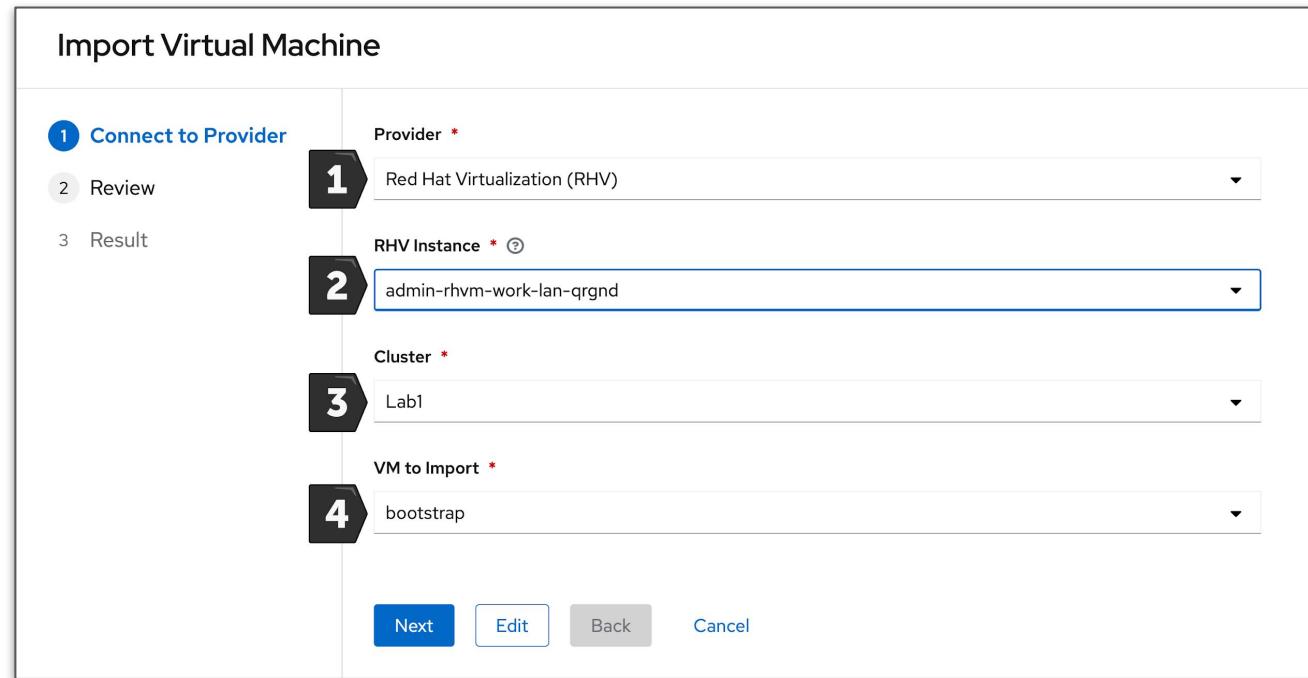
Cluster * Lab1

VM to Import * bootstrap

2 Review

3 Result

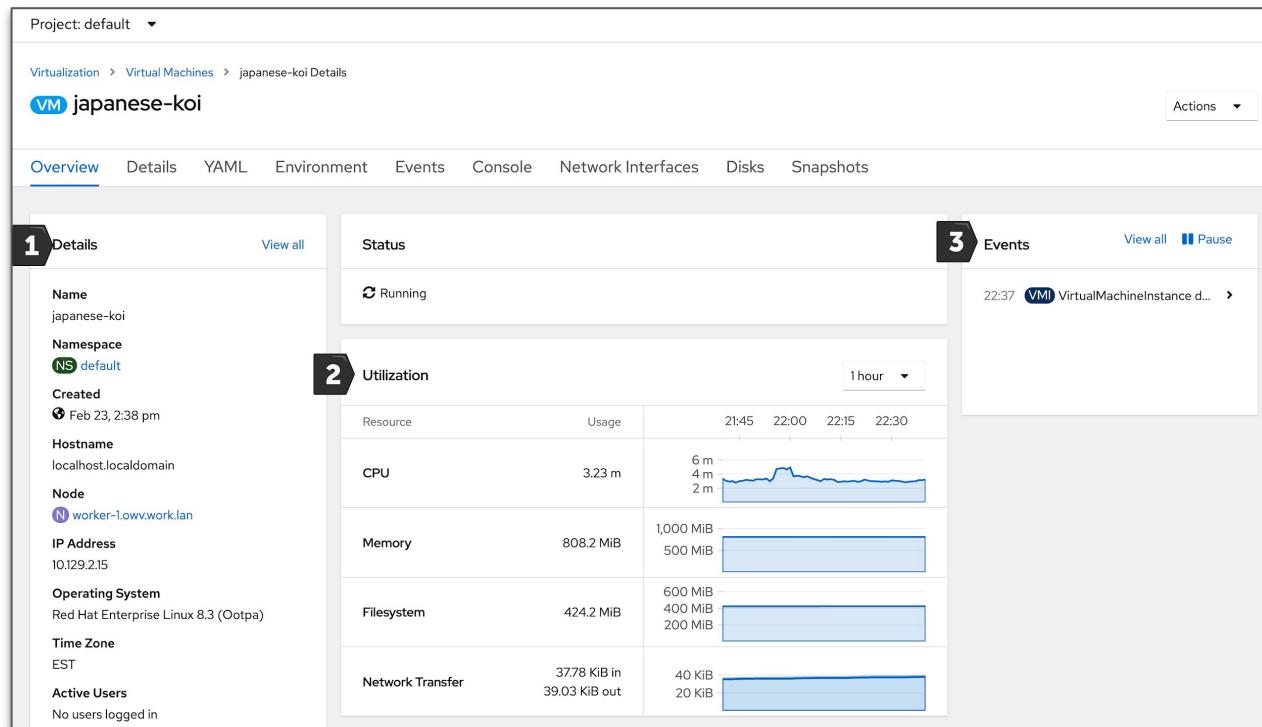
Next Edit Back Cancel



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 Details' page in the Red Hat Virtualization interface. The 'Actions' menu is open, displaying various management options. A red box highlights the 'Delete Virtual Machine' option at the bottom of the list.

VM japanese-koi

Overview Details YAML Environment Events Console Network Interfaces Disks Snapshots

Details

- Name: japanese-koi
- Namespace: default
- Created: Feb 23, 2:38 pm
- Hostname: localhost.localdomain
- Node: worker-1.oww.work.lan
- IP Address: 10.129.2.15
- Operating System: Red Hat Enterprise Linux 8.3 (Ootpa)

Status

Running

Utilization

Resource	Usage	22:15	22:30
CPU	3.9 m	8 m 6 m 4 m 2 m	8 m 6 m 4 m 2 m
Memory	808.2 MiB	1,000 MiB 500 MiB	1,000 MiB 500 MiB
Filesystem	424.8 MiB	600 MiB 400 MiB 200 MiB	600 MiB 400 MiB 200 MiB

Virtual Machine - Disks and NICs

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

The screenshot shows two views of a virtual machine named "japanese-koi".

Network Interfaces View:

- 1** Add Network Interface
- 2** Pod Networking
- 3** lab-silver

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

Disks View:

- 1** Add Disk
- 2** rootdisk
- 3** lab-silver

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

File Systems View:

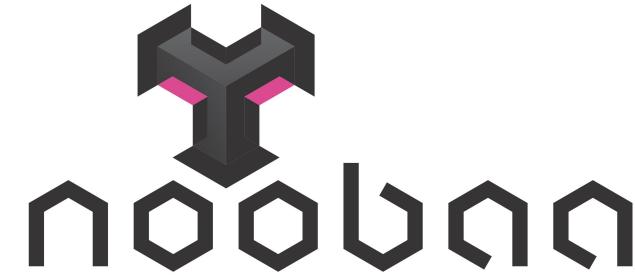
Name	File System Type	Mount Point	Total Bytes	Used Bytes
vdb2	vfat	/boot/efi	99.79 MiB	6.84 MiB
vdb3	xfs	/	19.89 GiB	1.41 GiB

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



OpenShift Storage Foundation (OSF)



Persistent Volume

Block

- Primary for DB and Transactional workloads
- Low latency
- Messaging

Provided by Rook-Ceph

Shared File System

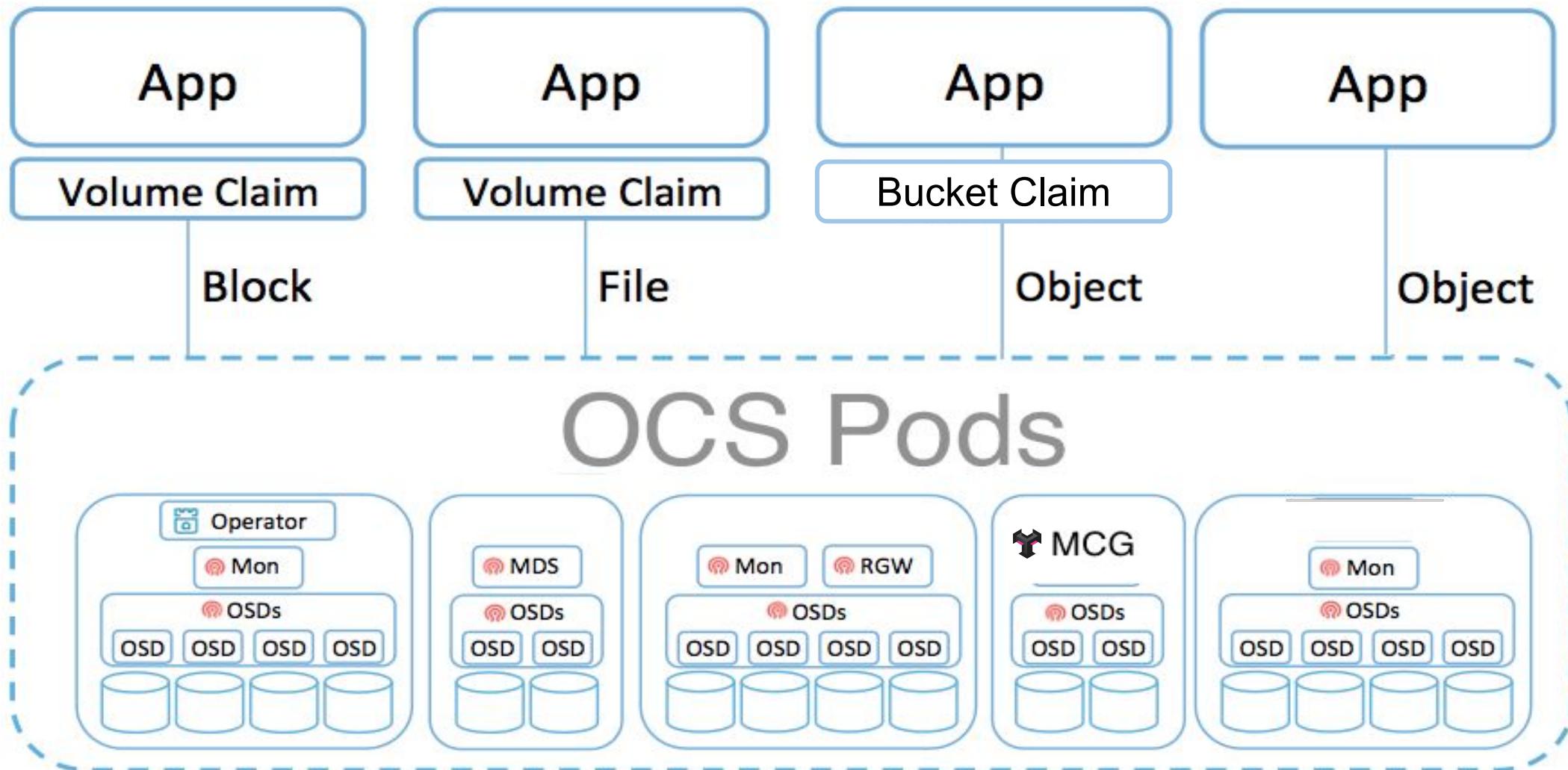
- POSIX-compliant shared file system
- Interface for legacy workloads
- CI/CD Pipelines
- AI/ML Data Aggregation
- Messaging

Provided by Rook-Ceph

Object Service

- Media, AI/ML training data, Archiving, Backup, Health Records
- Great Bandwidth performance
- Object API (S3/Blob)

Provided by NooBaa



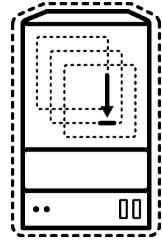


MULTI-CLOUD OBJECT GATEWAY



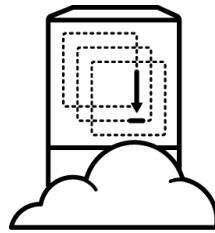
Start lean

A single lightweight pod for basic development and tests



Scale locally

Scale with local volumes or Red Hat Ceph Storage



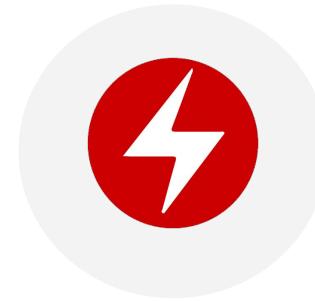
Workload portability

Easily mirror data to other cluster or native cloud storage



NOOBAA CORE

Managing the data flow
Providing the object service



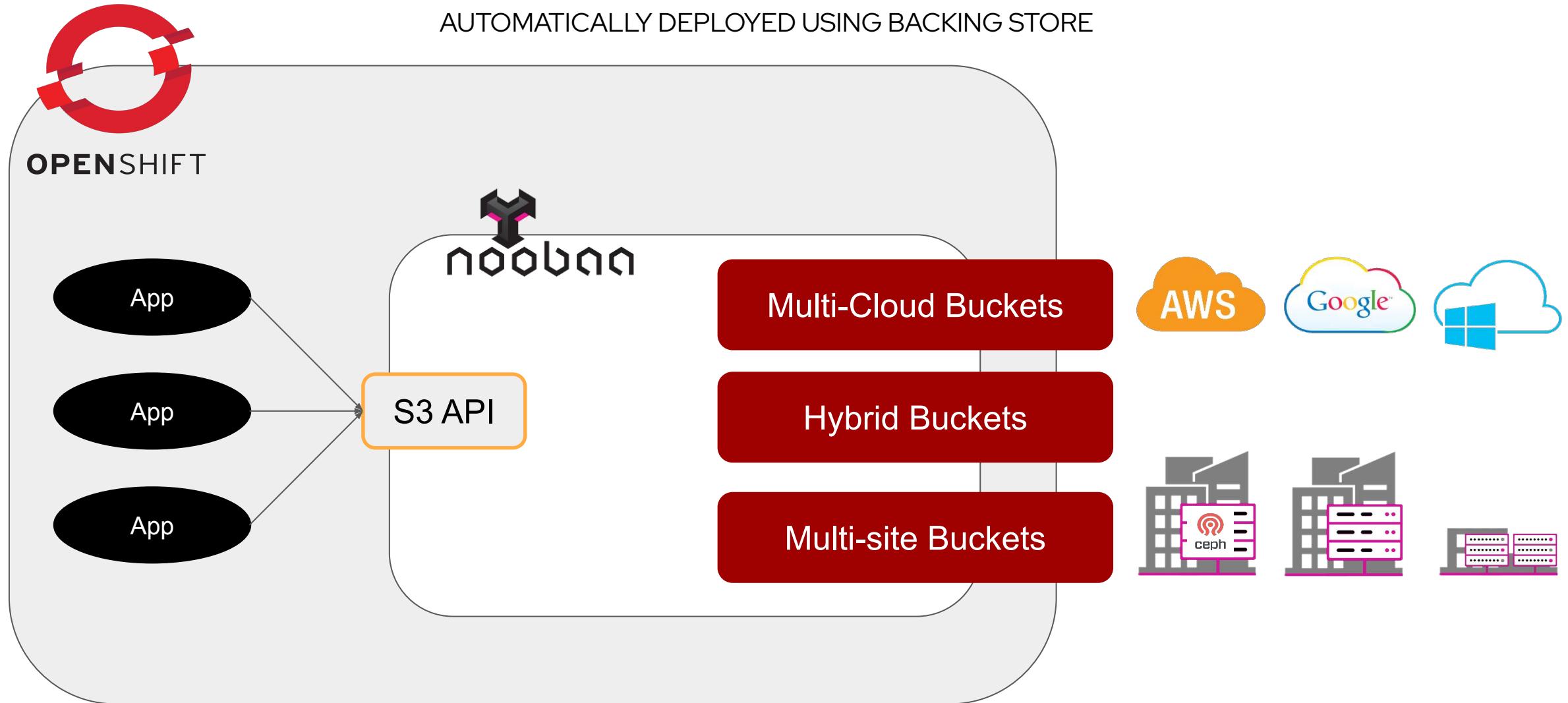
NOOBAA OPERATOR

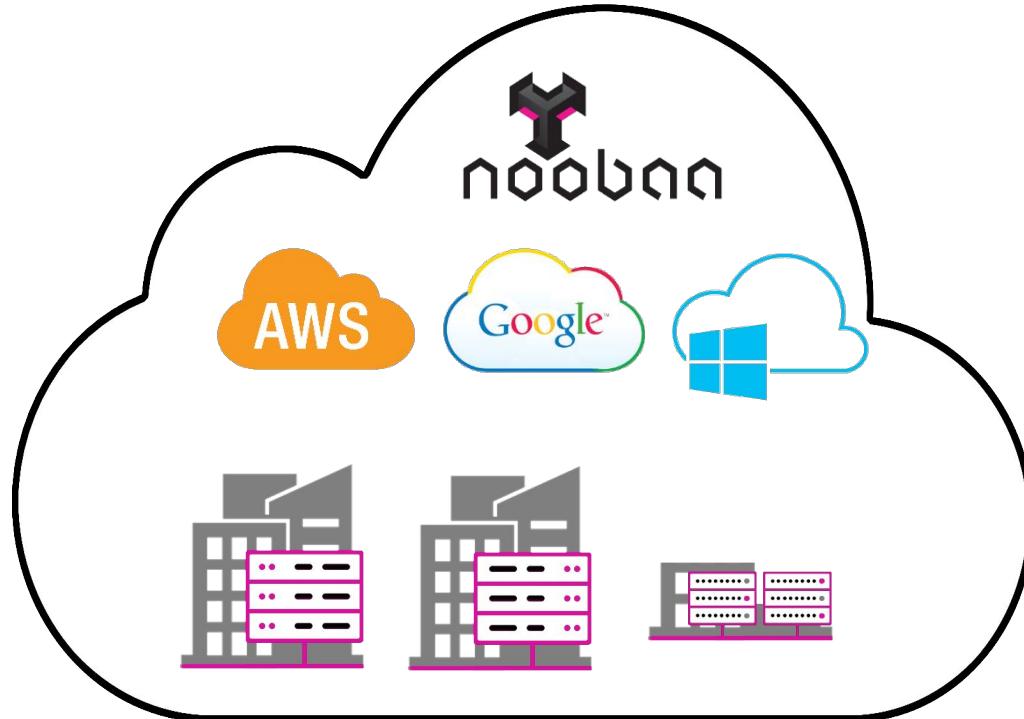
Deployment and second day
operations



COMMAND LINE

Bringing user experience to data
management





HIGH STACK - DATA MANAGEMENT

- ❖ Active Active – Keep your data available
- ❖ Flexible – Any resource, on premises or cloud
- ❖ Secured – All data encrypted, managed

LOW STACK - DATA EFFICIENCY LAYER

- ❖ Compression
- ❖ Inline Deduplication

ENABLING THE OPEN HYBRID CLOUD

**ENCRYPTION AT REST AND IN MOTION**

Every chunked data is encrypted with its own key using AES 256. Separation between the key management and the data.

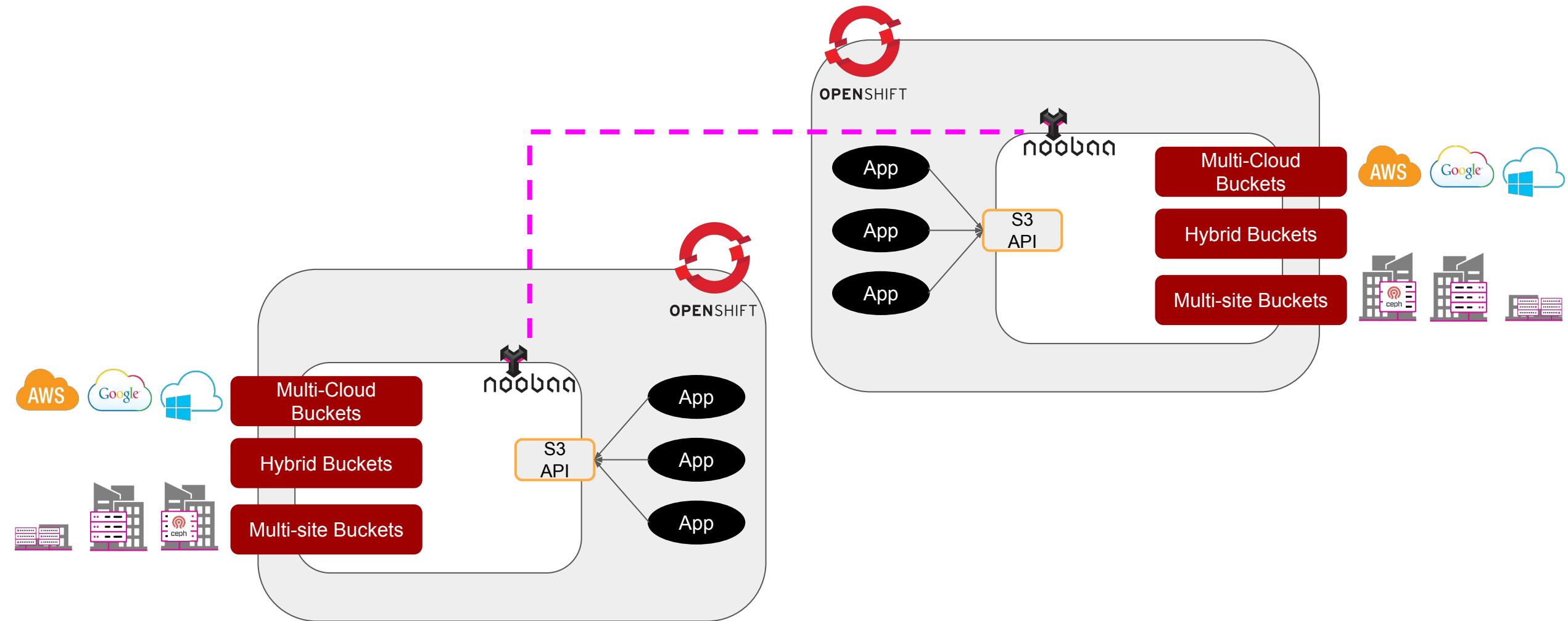
**ANONYMIZATION & MASKING**

Unique customizable data flow, that can anonymize or mask your own data before storing or collaborating

**FINGERPRINTING**

Automatic data fingerprinting with self-healing and trust management of the storage nodes

MULTI-CLUSTER ENVIRONMENT





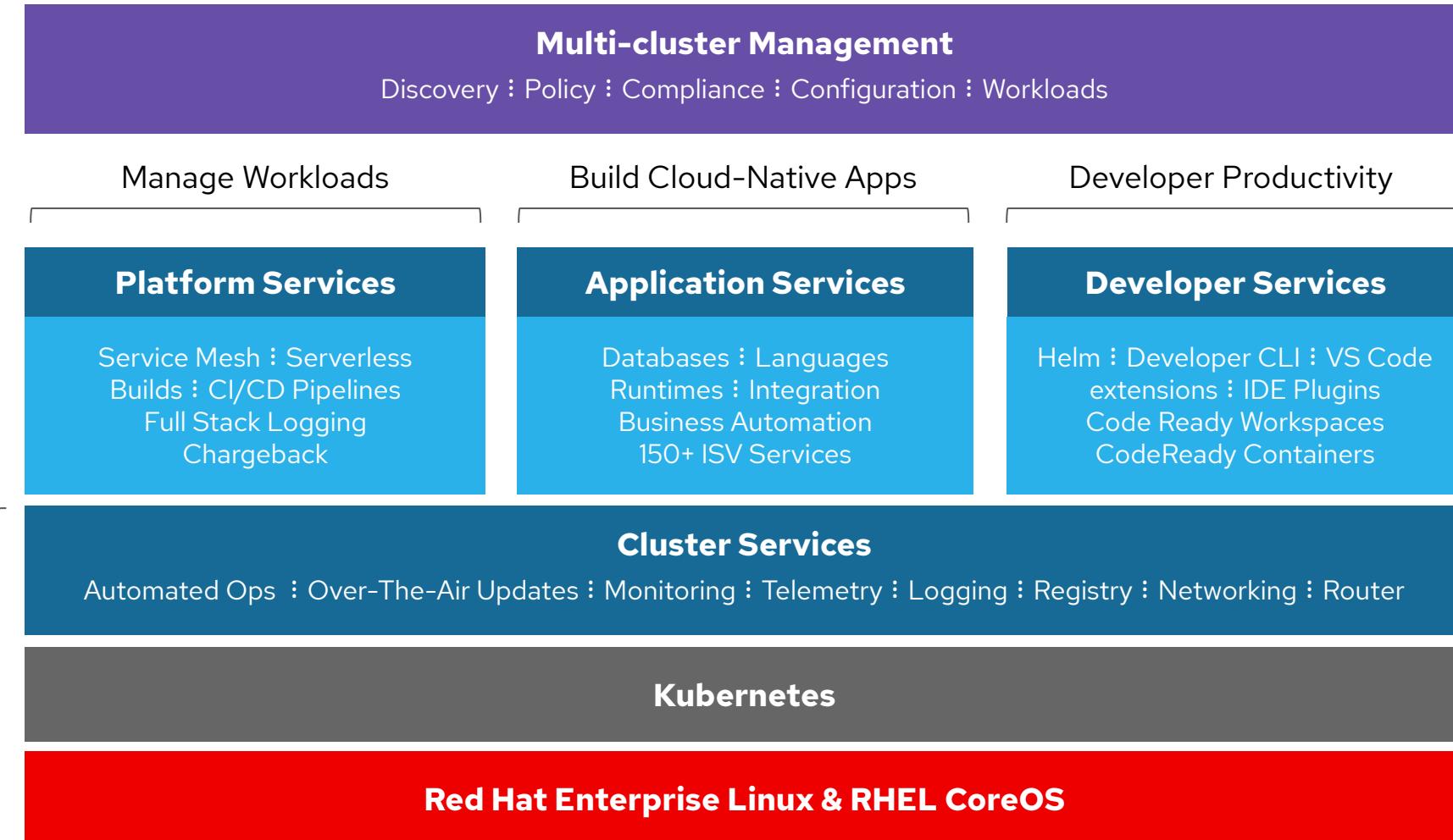
OpenShift Serverless

Alfred Bach
abach@redhat.com

Updated: April 2020



OpenShift Container Platform



Physical



Virtual

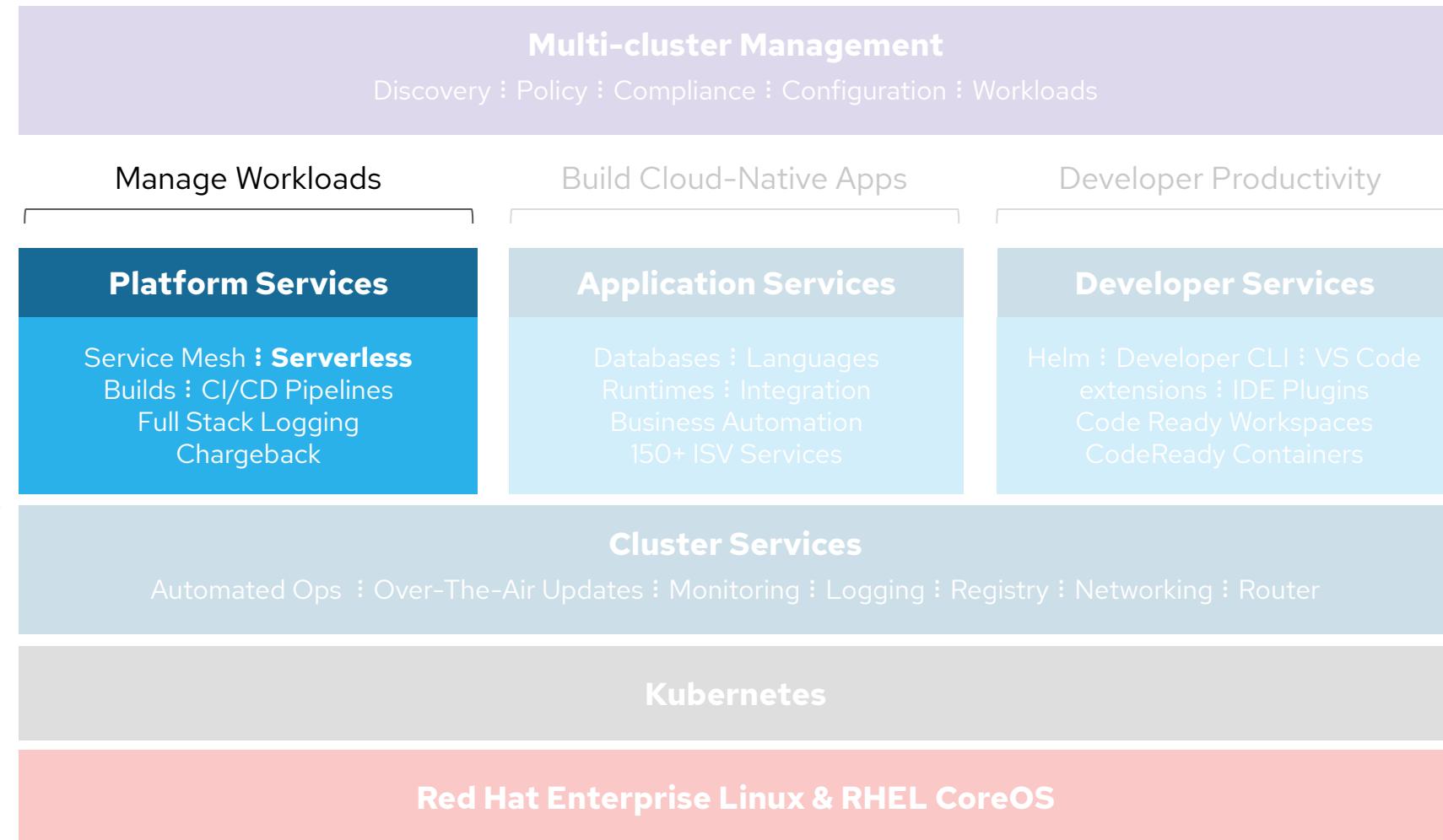


Private cloud



Public cloud

OpenShift Container Platform



Serverless Market Trends

"Use Serverless To optimize The Benefits of The cloud"²

40%

of enterprises adopted Serverless technologies or practices with expected growth coming in the next 12 to 18 months.¹

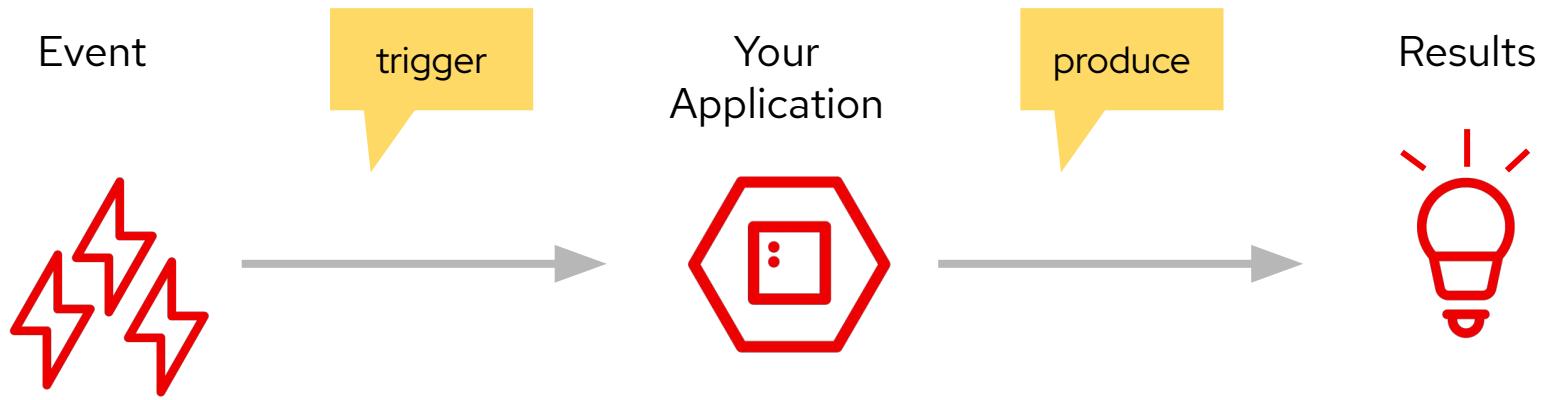


Vendor lock-in is the second biggest concern when adopting Serverless technologies.¹

60%

of the serverless practitioners reported "*reduction of operational costs*" with the second biggest benefit being "*scale with demand automatically*"

The "Serverless Pattern"



HTTP Requests

Kafka Messages

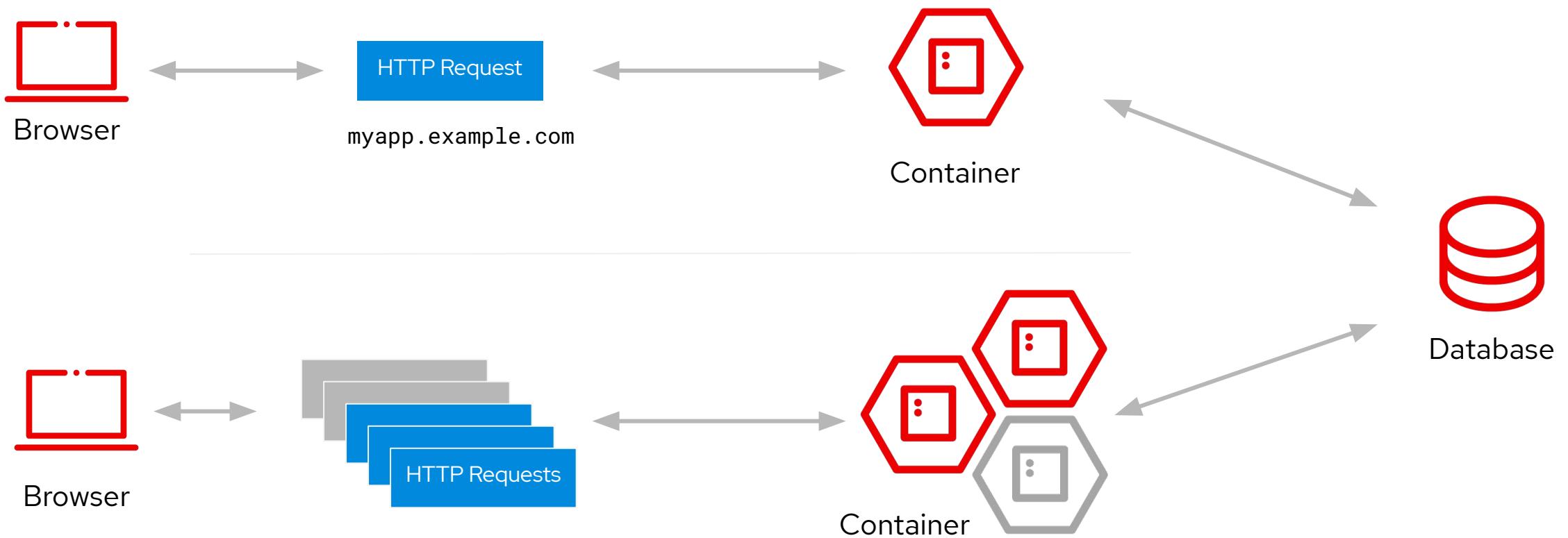
Image Uploaded

New Order

Login from user

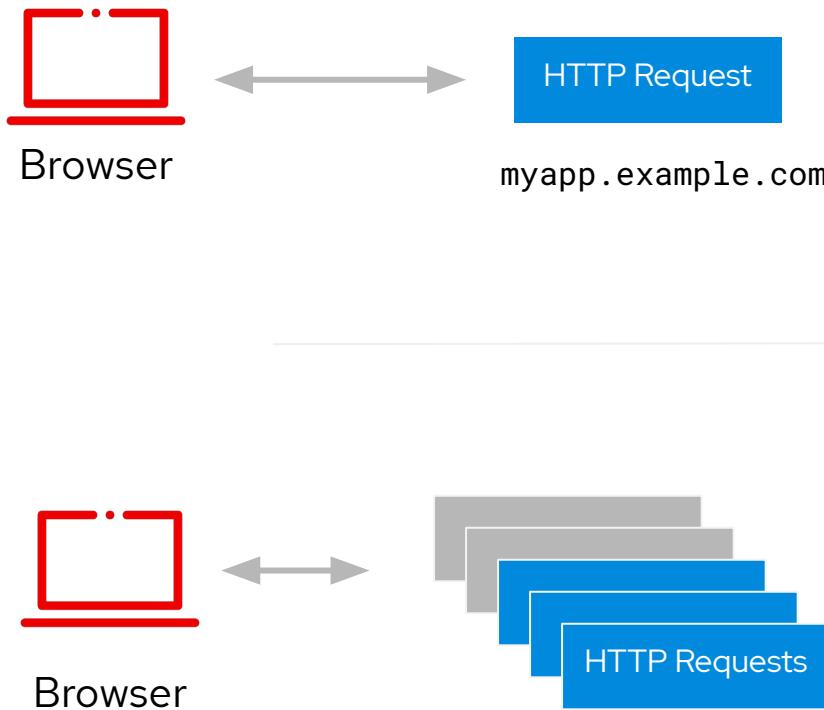
The "Serverless Pattern"

A serverless web application



The "Serverless Pattern"

A serverless web application

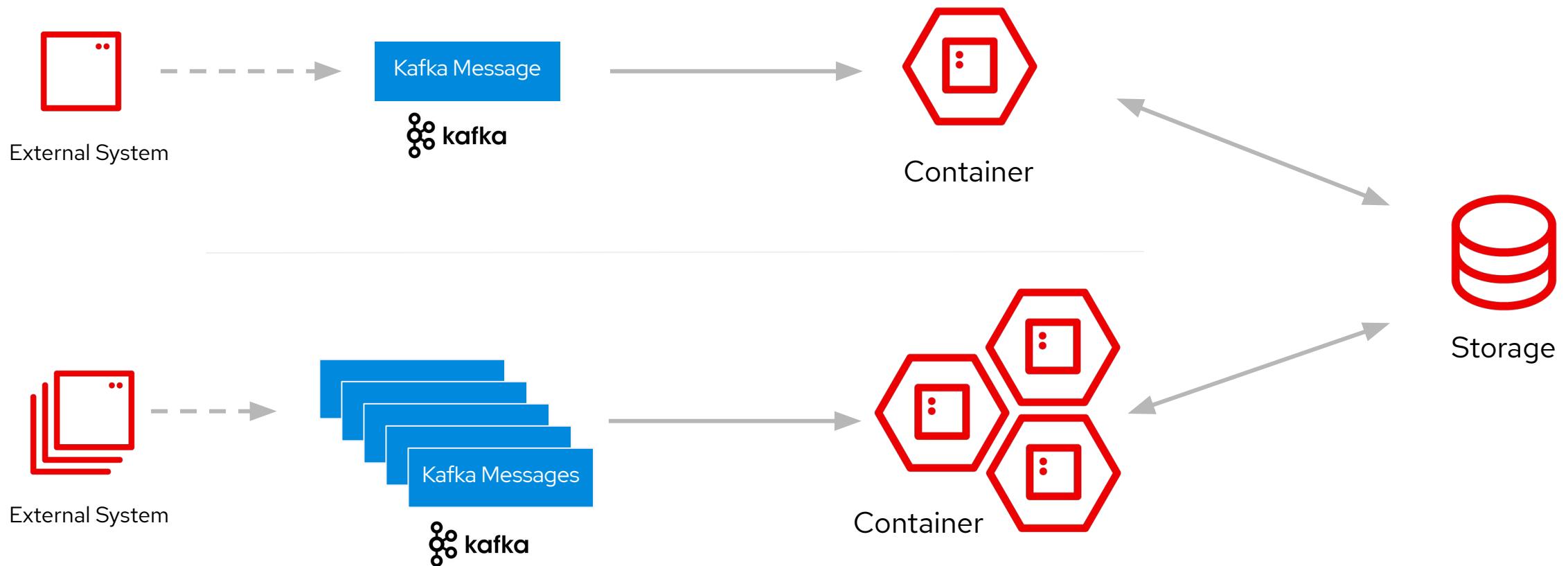


Benefits of this model:

- No need to setup auto-scaling and load balancers
 - Scale down and save resources when needed.
 - Scale up to meet the demand.
- No tickets to configure SSL for applications
- Enable Event Driven Architectures (EDA) patterns
- Enable teams to associate cost with IT
- Modernize existing applications to run as serverless containers

The "Serverless Pattern"

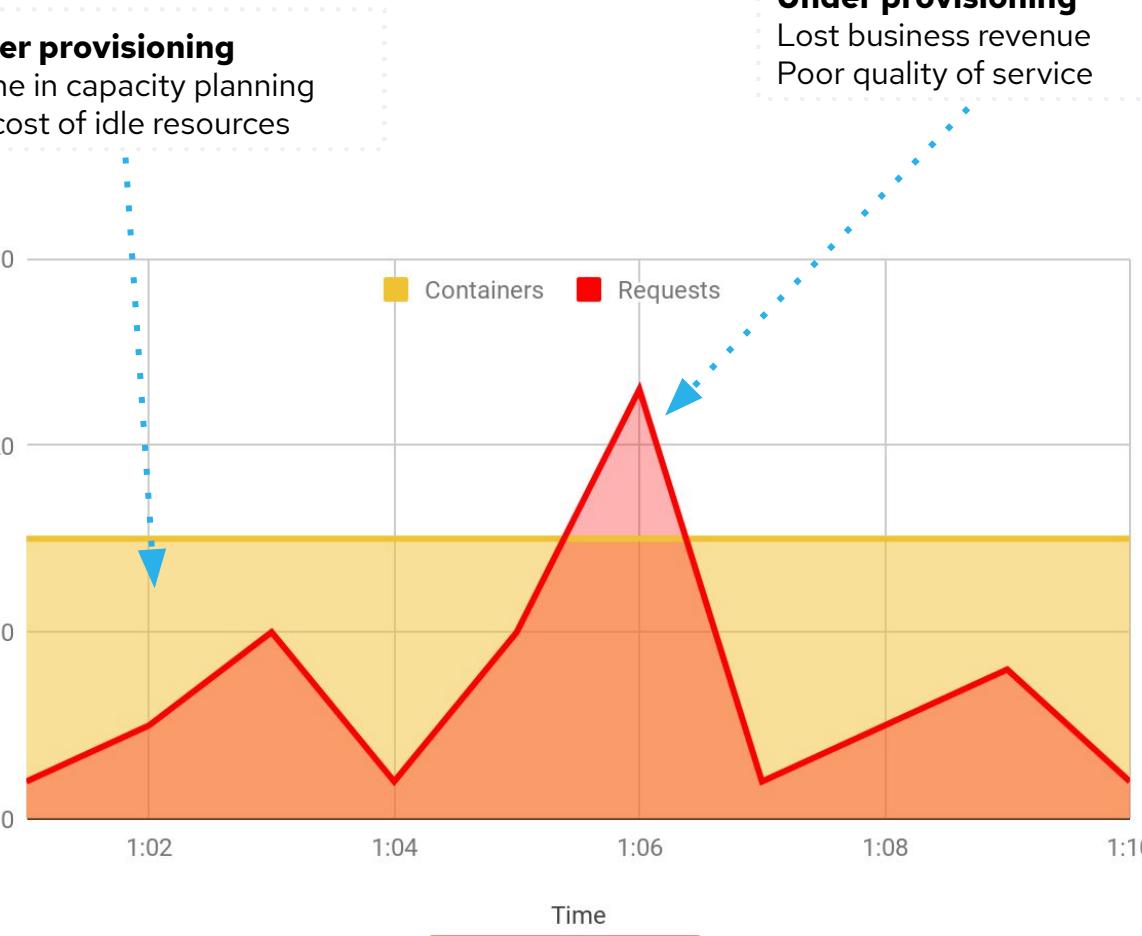
Processing a Kafka message



Serverless Operational Benefits

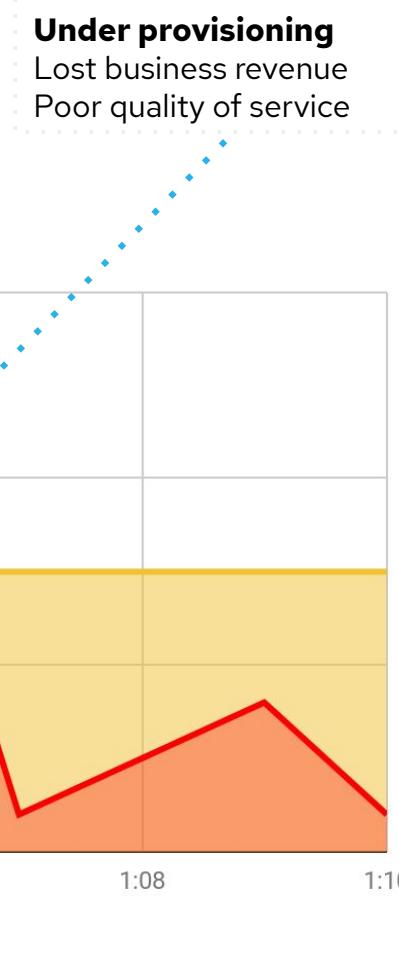
Over provisioning

Time in capacity planning
IT cost of idle resources



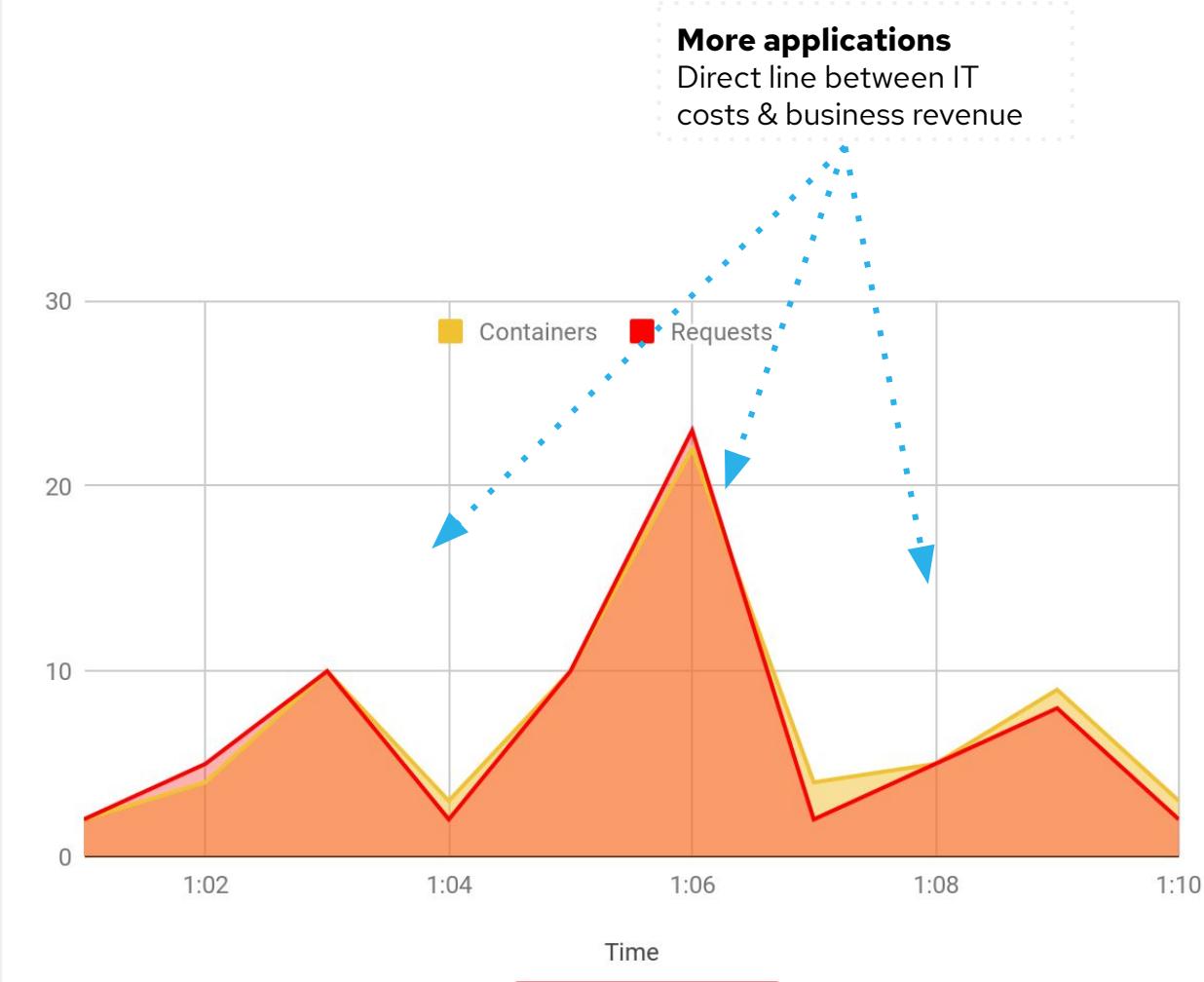
Under provisioning

Lost business revenue
Poor quality of service



More applications

Direct line between IT costs & business revenue



NOT Serverless

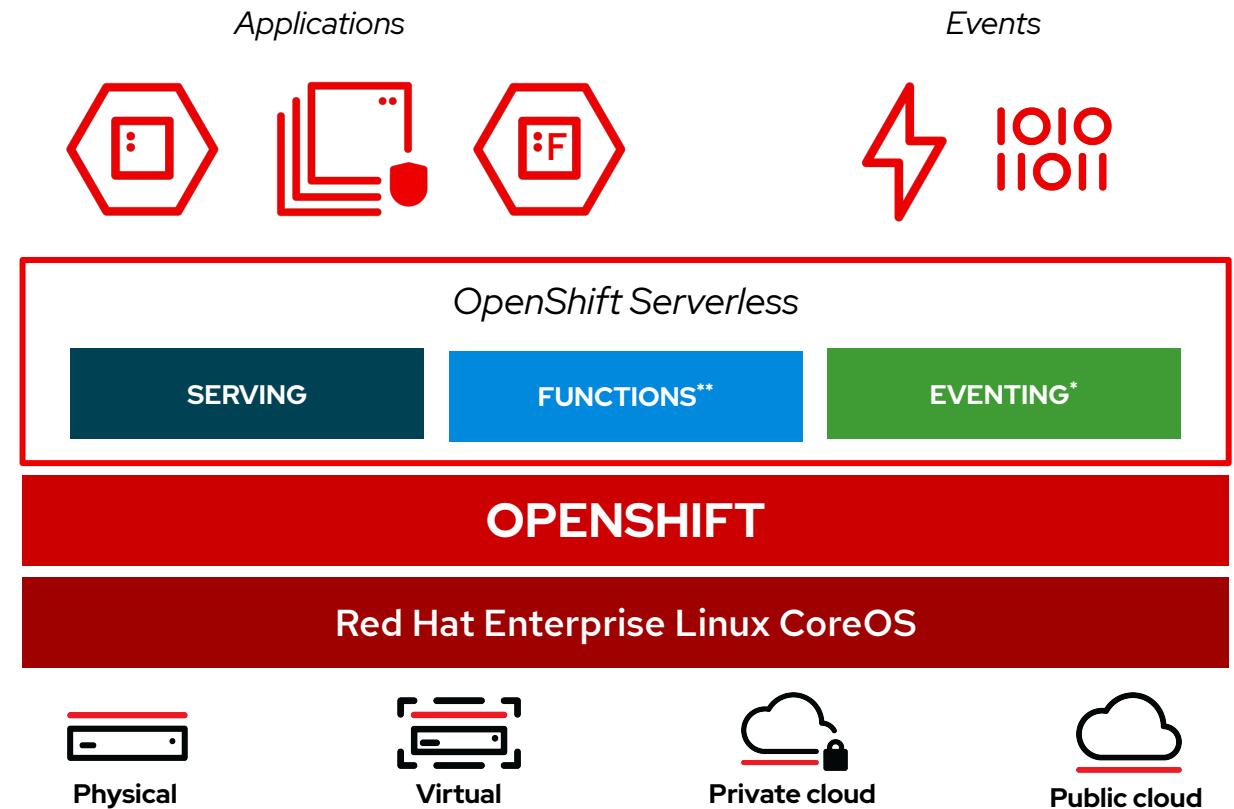
with Serverless



OpenShift Serverless

Event-driven serverless containers and functions

- Deploy and run **serverless containers**
- Use any programming language or runtime
- Modernize existing applications to run serverless
- Powered by a rich ecosystem of event sources
- Manage serverless apps natively in Kubernetes
- Based on open source project **Knative**
- Run anywhere OpenShift runs



* Eventing is currently in Technology Preview

** Functions are currently a work in progress initiative

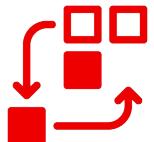
OpenShift Serverless

Key Features



Containers made easy

Simplified developer experience to deploy applications/code on serverless containers abstracting infrastructure & focusing on what matters.



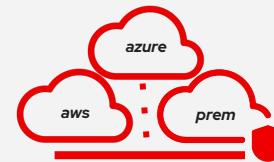
Immutable revisions

Deploy new features: performing canary, A/B or blue-green testing with gradual traffic rollout with no sweat and following best practices.



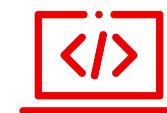
Automatic scaling

No need to configure number of replicas, or idling. Scale to zero when not in use, auto scale to thousands during peak, with built-in reliability and fault-tolerance.



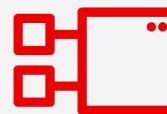
Ready for the Hybrid Cloud

Truly portable serverless running anywhere OpenShift runs, that is on-premises or on any public cloud. Leverage data locality and SaaS when needed.



Any programming language

Use any programming language or runtime of choice. From Java, Python, Go and JavaScript to Quarkus, SpringBoot or Node.js.



Event Driven Architectures

Build loosely coupled & distributed apps connecting with a variety of built-in or third-party event sources or connectors powered by Operators.

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