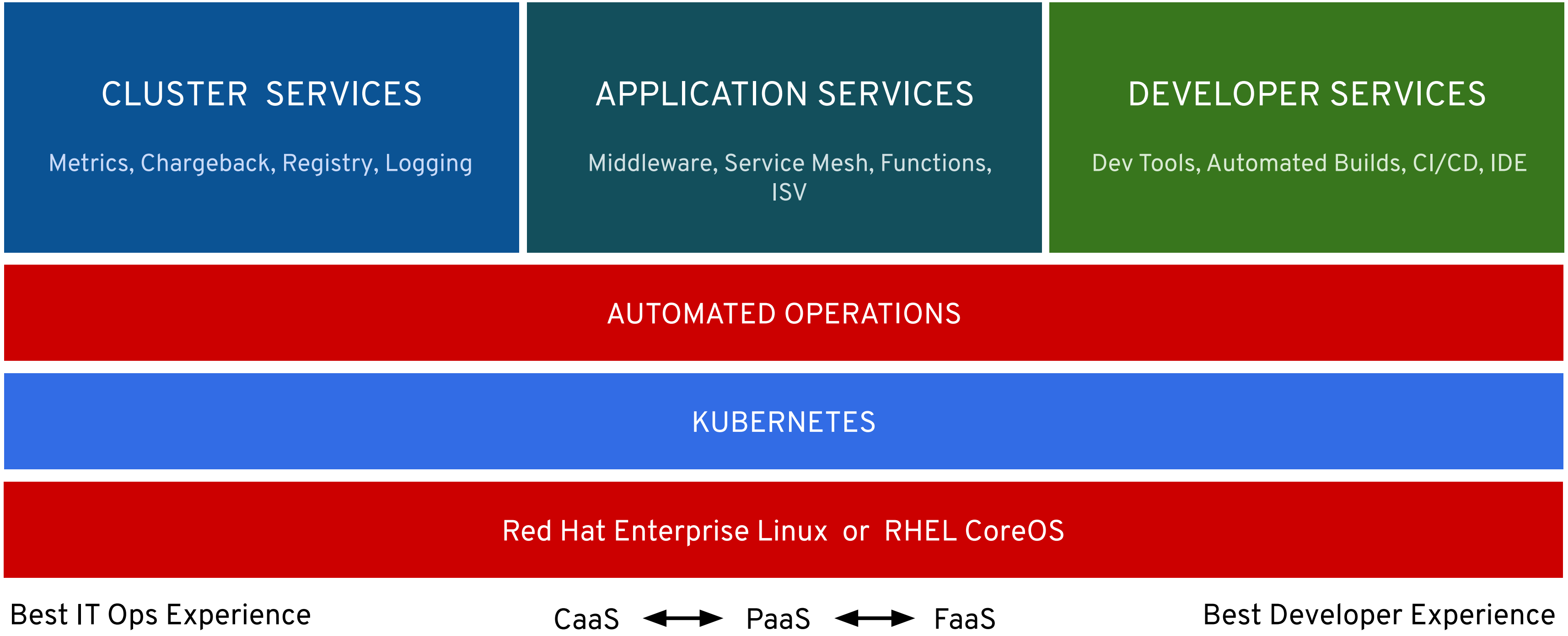
Red Hat

Red Hat
OpenShift

OpenShift 4 for Ops

Nick Schuetz
Principal Specialist Solution
Architect

OpenShift 4 Platform



CLUSTER SERVICES

Metrics, Chargeback, Registry, Logging

APPLICATION SERVICES

Middleware, Service Mesh, Functions, ISV

DEVELOPER SERVICES

Dev Tools, Automated Builds, CI/CD, IDE

AUTOMATED OPERATIONS

KUBERNETES

Red Hat Enterprise Linux or RHEL CoreOS

Best IT Ops Experience

CaaS ↔ PaaS ↔ FaaS

Best Developer Experience

OpenShift 4 Themes

DAY 2 OPERATIONS

Integrate CoreOS technology for a better install, re-config and upgrade experience.

Bring over-the-air upgrades to the platform.

IMMUTABLE INFRASTRUCTURE

Introduce Red Hat CoreOS as an immutable OS option. Enhance “infrastructure as code” throughout the platform.

OPERATOR FRAMEWORK

Provide tools, guidance and automation for customers and partners to deliver smart software on top of OpenShift

OpenShift 4.1 Workstreams Lifecycle

DAY 2 OPERATIONS

Installer + bootstrapping

Autoscale out of the box

MachineSet node pools

IMMUTABLE INFRASTRUCTURE

Red Hat Enterprise Linux CoreOS

Discourage SSH/node mutation

Ignition for Machine config

OPERATOR FRAMEWORK

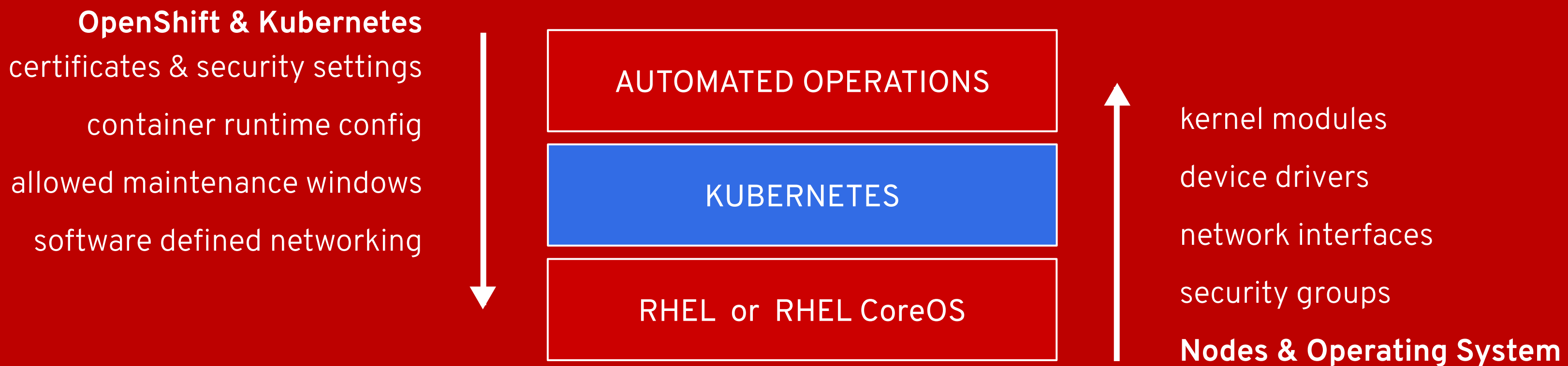
SDK & testing tools

OperatorHub for discovery

OLM delivers upper stack services

The New Platform Boundary

OpenShift 4 is aware of the entire infrastructure and brings the Operating System under management



OpenShift & Kubernetes

certificates & security settings

container runtime config

allowed maintenance windows

software defined networking

kernel modules

device drivers

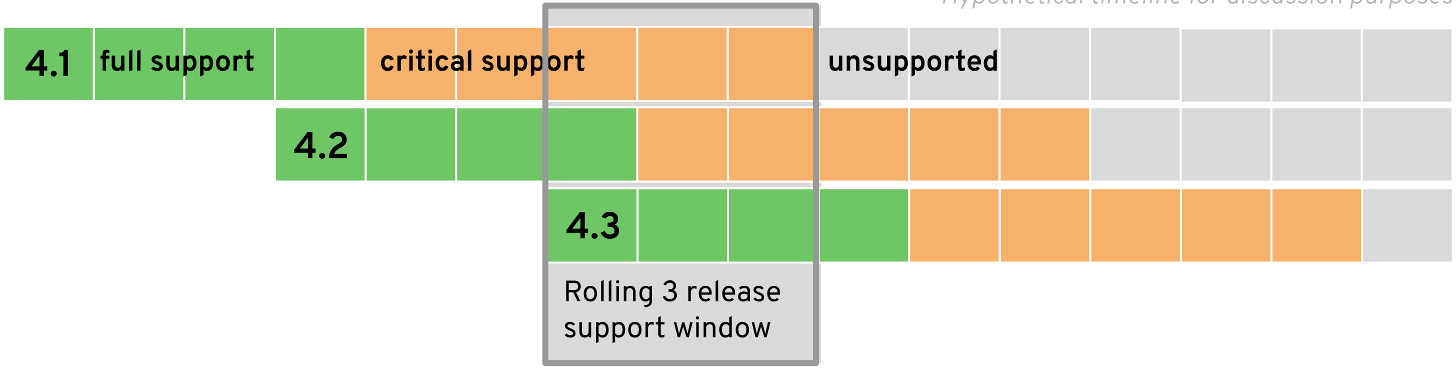
network interfaces

security groups

Nodes & Operating System

OpenShift 4 Lifecycle

** Hypothetical timeline for discussion purposes*



New model
Release based, not date based. Rolling three release window for support.

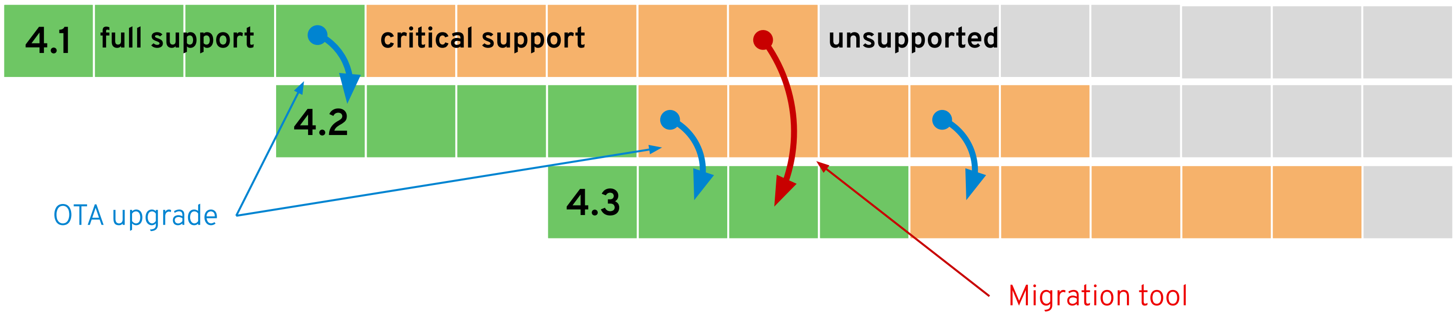
The overall 4 series will be supported for at least three years

- Minimum two years full support (likely more)
- One year maintenance past the end of full support

EUS release planned
Supported for 14 months of critical bug and critical security fixes instead of the normal 5 months. If you stay on the EUS for its entire life, you must use the application migration tooling to move to a new cluster

OpenShift 4 Upgrades

** Hypothetical timeline for discussion purposes*



OTA Upgrades
Works between two minor releases in a serial manner.

Happy path = migrate through each version
On a regular cadence, migrate to the next supported version.

Optional path = migration tooling
If you fall more than two releases behind, you must use the application migration tooling to move to a new cluster.

Current minor release
Full support for all bugs and security issues
1 month full support overlap with next release to aid migrations

Previous minor release
Fixes for critical bugs and security issues for 5 months

Installation Experiences

OPENSHIFT CONTAINER PLATFORM

Full Stack Automated

Simplified opinionated "Best Practices" for cluster provisioning

Fully automated installation and updates including host container OS.

Red Hat Enterprise Linux CoreOS

Pre-existing Infrastructure

Customer managed resources & infrastructure provisioning

Plug into existing DNS and security boundaries

Red Hat Enterprise Linux CoreOS

Red Hat Enterprise Linux

HOSTED OPENSHIFT

Azure Red Hat OpenShift

Deploy directly from the Azure console. Jointly managed by Red Hat and Microsoft Azure engineers.

OpenShift Dedicated

Get a powerful cluster, fully Managed by Red Hat engineers and support.

Generally Available

OpenShift 4 Supported Providers*

Full Stack Automated

Azure **

amazon web services

Pre-existing Infrastructure

Azure ***

amazon web services

Bare Metal

vmware vSphere

** Requires Internet connectivity; support for cluster-wide proxy & disconnected installation/updating tentatively planned for 4.2.*

*** Automated Installer for Azure, OSP, GCP tentatively planned for 4.2.*

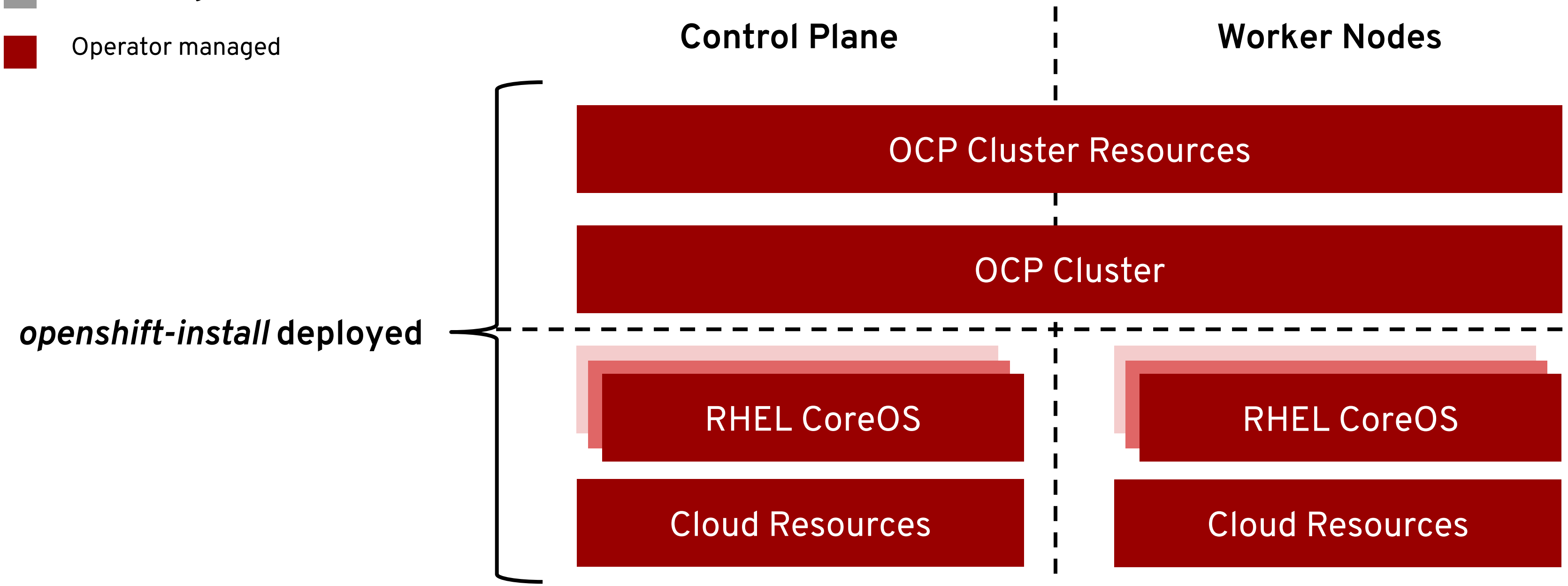
**** Azure UPI is currently in Dev Preview.*

Full Stack Automated Deployments

Day 1: openshift-install - Day 2: Operators

■ User managed

🟥 Operator managed



Generally Available

Full Stack Automated Deployments

Simplified Cluster Creation

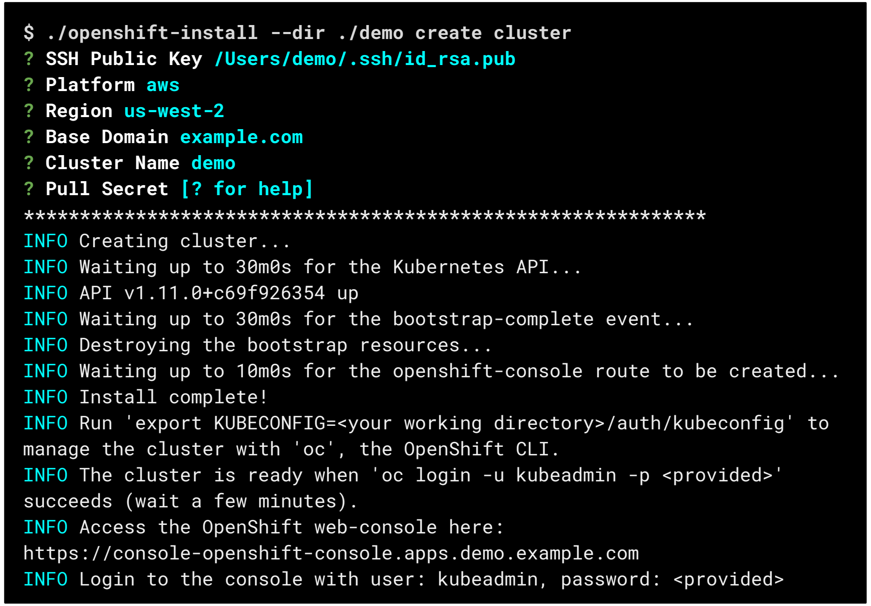
Designed to easily provision a “best practices” OpenShift cluster

- New CLI-based installer with interactive guided workflow that allows for customization at each step
- Installer takes care of provisioning the underlying Infrastructure significantly reducing deployment complexity
- Leverages RHEL CoreOS for all node types enabling full stack automation of installation and updates of both platform and host OS content

Faster Install

The installer typically finishes within 30 minutes

- Only minimal user input needed with all non-essential install config options now handled by component operator CRD’s
- 4.1 provides support for AWS deployments with additional provider support planned in future releases
- See the OpenShift documentation for more details

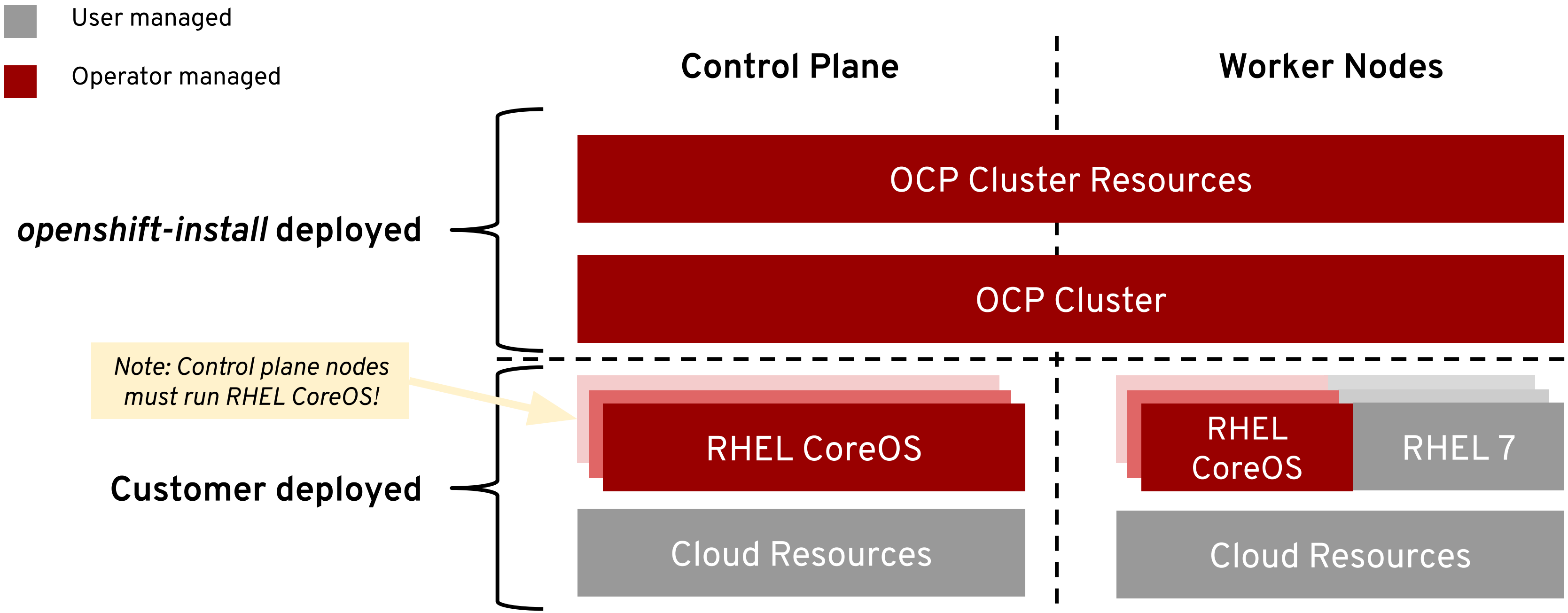


```
$ ./openshift-install --dir ./demo create cluster
? SSH Public Key /Users/demo/.ssh/id_rsa.pub
? Platform aws
? Region us-west-2
? Base Domain example.com
? Cluster Name demo
? Pull Secret [? for help]
*************************************************************
INFO Creating cluster...
INFO Waiting up to 30m0s for the Kubernetes API...
INFO API v1.11.0+c69f926354 up
INFO Waiting up to 30m0s for the bootstrap-complete event...
INFO Destroying the bootstrap resources...
INFO Waiting up to 10m0s for the openshift-console route to be created...
INFO Install complete!
INFO Run 'export KUBECONFIG=<your working directory>/auth/kubeconfig' to
manage the cluster with 'oc', the OpenShift CLI.
INFO The cluster is ready when 'oc login -u kubeadmin -p <provided>'
succeeds (wait a few minutes).
INFO Access the OpenShift web-console here:
https://console-openshift-console.apps.demo.example.com
INFO Login to the console with user: kubeadmin, password: <provided>
```

Generally Available

Deploying to Pre-existing Infrastructure

Day 1: openshift-install - Day 2: Operators + Customer Managed Infra & Workers



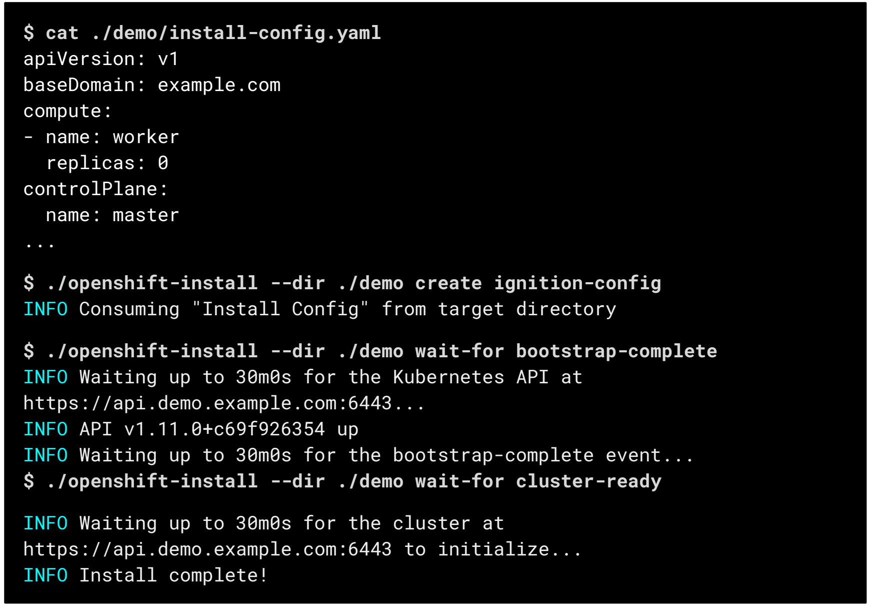
Generally Available

Deploying to Pre-existing Infrastructure

Customized OpenShift Deployments

Enables OpenShift to be deployed to user managed resources and pre-existing infrastructure.

- Customers are responsible for provisioning all infrastructure objects including networks, load balancers, DNS, hardware/VMs and performing host OS installation
- Deployments can be performed both on-premise and to the public cloud
- OpenShift installer handles generating cluster assets (such as node ignition configs and kubeconfig) and aids with cluster bring-up by monitoring for bootstrap-complete and cluster-ready events
- While RHEL CoreOS is mandatory for the control plane, either RHEL CoreOS or RHEL 7 can be used for the worker/infra nodes
- Node auto-scaling can be setup for providers with OpenShift Machine API support
- See the OpenShift documentation for more details



```
$ cat ./demo/install-config.yaml
apiVersion: v1
baseDomain: example.com
compute:
- name: worker
  replicas: 0
controlPlane:
  name: master
...

$ ./openshift-install --dir ./demo create ignition-config
INFO Consuming "Install Config" from target directory

$ ./openshift-install --dir ./demo wait-for bootstrap-complete
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.demo.example.com:6443...
INFO API v1.11.0+c69f926354 up
INFO Waiting up to 30m0s for the bootstrap-complete event...
$ ./openshift-install --dir ./demo wait-for cluster-ready

INFO Waiting up to 30m0s for the cluster at
https://api.demo.example.com:6443 to initialize...
INFO Install complete!
```

Generally Available

OpenShift Bootstrap Process: Self-Managed Kubernetes

How to boot a self-managed cluster:

- OpenShift 4 is unique in that management extends all the way down to the operating system
- Every machine boots with a configuration that references resources hosted in the cluster it joins enabling cluster to manage itself
- Downside is that every machine looking to join the cluster is waiting on the cluster to be created
- Dependency loop is broken using a bootstrap machine, which acts as a temporary control plane whose sole purpose is bringing up the permanent control plane nodes
- Permanent control plane nodes get booted and join the cluster leveraging the control plane on the bootstrap machine
- Once the pivot to the permanent control plane takes place, the remaining worker nodes can be booted and join the cluster

Bootstrapping process step by step:

1. Bootstrap machine boots and starts hosting the remote resources required for master machines to boot.
2. Master machines fetch the remote resources from the bootstrap machine and finish booting.
3. Master machines use the bootstrap node to form an etcd cluster.
4. Bootstrap node starts a temporary Kubernetes control plane using the newly-created etcd cluster.
5. Temporary control plane schedules the production control plane to the master machines.
6. Temporary control plane shuts down, yielding to the production control plane.
7. Bootstrap node injects OpenShift-specific components into the newly formed control plane.
8. Installer then tears down the bootstrap node or if user-provisioned, this needs to be performed by the administrator.

Comparison between deployments methods

	Full Stack Automation	Pre-existing Infrastructure
Build Network	Installer	User
Setup Load Balancers	Installer	User
Configure DNS	Installer	User
Hardware/VM Provisioning	Installer	User
OS Installation	Installer	User
Generate Ignition Configs	Installer	Installer
OS Support	Installer: RHEL CoreOS	User: RHEL CoreOS + RHEL 7
Node Provisioning / Autoscaling	Yes	Only for providers with OpenShift Machine API support
Customization & Provider Support	Best Practices: AWS	Yes: AWS, Bare Metal, & VMware
	Generally Available	

RHEL 7 Worker/Infra Nodes

Add RHEL 7.6 machines with Ansible

Use openshift-ansible to prepare, configure and join your RHEL 7 nodes to the cluster. After you have a functional control plane, nodes can be added to the cluster with the `scaleup` playbook.

- Pending certificates signing request (CSRs) for each RHEL machine must be approved before joining cluster
- See the OpenShift documentation for more details

Mixed clusters of RHEL 7 and RHEL CoreOS are ok

RHEL CoreOS is mandatory for the control plane, but mixed clusters of RHEL 7.6 and RHEL CoreOS are supported for any other node pools.

RHEL 8 is not yet supported for worker/infra nodes

Support will come in a later 4.x release

Upgrading OpenShift node components

With the `upgrade` playbook, RHEL 7 OpenShift components can be upgraded. Optionally, pre/post hooks are also available for performing custom tasks.

RHEL admins are responsible for:

Keeping host inventory up to date

Refresh your list of hosts before an upgrade, to make sure no machines are missed

Defining Ansible hooks

Run playbooks that will cordon/uncordon machines, along with any pre/post upgrade actions

Updating RHEL RPM content

Security, performance and regular updates from Red Hat

Partitioning disks

Configure, maintain and health check your disks

Configuring network interfaces

Configure, secure and maintain settings within your data center's specifications

Generally Available

Adding & Updating RHEL 7 Worker Nodes

System Requirements

- OpenShift 4 cluster deployed to pre-existing infrastructure (not supported on installer provisioned clusters)
- Latest release of RHEL 7 Server installed on every machine joining the cluster
  - Ansible >= 2.7.8, OpenShift Client (oc)
  - Necessary entitlements and YUM repo access pre-configured for every machine

Create an Ansible Inventory

- Inventory file with the appropriate groups and variables defined
  - An example inventory can be found in inventory/hosts.example
- Required variables include:
  - `openshift_kubeconfig_path:` Path to the kubeconfig for the cluster
  - `openshift_pull_secret_path:` Path to the pull secret to the image registry

Run RHEL 7 node 'scaleup' playbook

```
$ cd openshift-ansible; ansible-playbook -i inventory/hosts playbooks/scaleup.yml
```

- Pending certificates signing request (CSRs) for each RHEL machine must be approved before joining cluster

```
$ oc adm certificate approve <csr_name>
```

Upgrading RHEL 7 OpenShift node components

- Leverages upgrade section of Ansible Inventory to specify nodes
- Custom tasks can be performed during upgrades at different stages of the upgrade; refer to 'hooks' in the documentation for more information.

```
$ cd openshift-ansible; ansible-playbook -i inventory/hosts playbooks/upgrade.yml
```

Generally Available

Red Hat Enterprise Linux

	RED HAT® ENTERPRISE LINUX® **General Purpose OS**	RED HAT® ENTERPRISE LINUX CoreOS **Immutable container host**
BENEFITS	• 10+ year enterprise life cycle • Industry standard security • High performance on any infrastructure • Customizable and compatible with wide ecosystem of partner solutions	• Self-managing, over-the-air updates • Immutable and tightly integrated with OpenShift • Host isolation is enforced via Containers • Optimized performance on popular infrastructure
WHEN TO USE	When customization and integration with additional solutions is required	When cloud-native, hands-free operations are a top priority

Generally Available

Red Hat Enterprise Linux CoreOS

4.1 Image Availability:

- Amazon: AMIs
- vSphere: OVA
- Bare Metal: UEFI & BIOS

Installation Requirements:

- RHCOS image + ignition config (installer generated)

Bare Metal Installer (ISO or PXE):

```
coreos.inst=yes
coreos.inst.install_dev=sda
coreos.inst.image_url=http://10.10.10.1/rhcos-metal-uefi.raw.gz
coreos.inst.ignition_url=http://10.10.10.1/master.ign
```

RHCOS Details

- RHEL 8 bits (4.18 kernel)
- Includes all packages required for OpenShift
- Over-The-Air updates encompass OCP & RHCOS
- Transactional host updates
- Read-only OS binaries
- Preconfigured for most environments

Generally Available

Immutable Operating System

Red Hat Enterprise Linux CoreOS is versioned with OpenShift

CoreOS is tested and shipped in conjunction with the platform. Red Hat runs thousands of tests against these configurations.

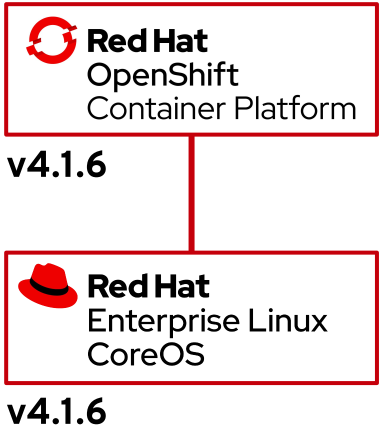
Red Hat Enterprise Linux CoreOS is managed by the cluster

The Operating system is operated as part of the cluster, with the config for components managed by Machine Config Operator:

- CRI-O config
- Kubelet config
- Authorized registries
- SSH config

RHEL CoreOS admins are responsible for:

Nothing. 😃 🙌



Generally Available

CRI-O Support in OpenShift

CRI-O tracks and versions identical to Kubernetes, simplifying support permutations

CRI-O 1.12	Kubernetes 1.12	OpenShift 4.0
CRI-O 1.13	Kubernetes 1.13	OpenShift 4.1
CRI-O 1.14	Kubernetes 1.14	OpenShift 4.2

Generally Available

Machine Config Operator (MCO)

Example with CRI-O:

- Simplified management across entire cluster
- Container Runtime Config (CRC) is specialized Machine Config (MC) for CRI-O
- Container Runtime Config for CRI-O exposes configuration knobs

Leverages

- Custom Resource Definitions
- Machine Config Pools
- Machine Config Pool Selector
- Machine Configs
- Container Runtime Config (specialized MC)

```
apiVersion: machineconfiguration.openshift.io/v1
kind: ContainerRuntimeConfig
metadata:
  name: set-log-and-pid
spec:
  machineConfigPoolSelector:
    matchLabels:
      debug-crio: config-log-and-pid
  containerRuntimeConfig:
    pidsLimit: 2048
    logLevel: debug
```

```
$ oc get ContainerRuntimeConfig

NAME             AGE
set-log-and-pid   22h
```

Generally Available

Discovering Configuration Options

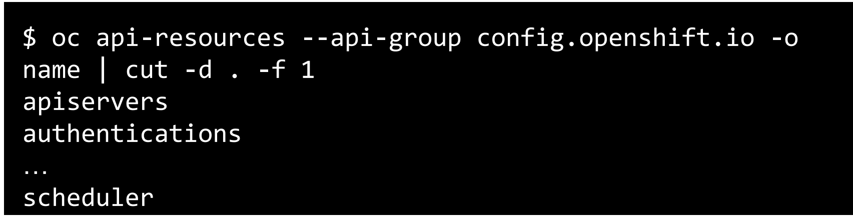
Cluster config objects are the human/machine APIs

These can be discovered with the commands to the right.

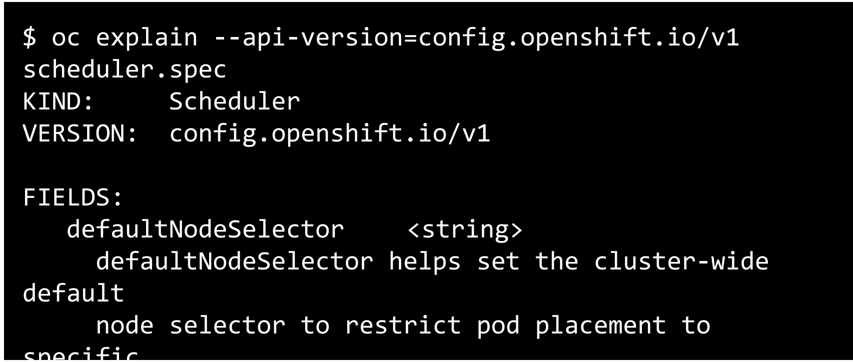
Configuration set via this mechanism will be carried forward through cluster updates.

Configuration Resources

- Apiserver
- Authentication
- Build
- Clusterversion
- Console
- Dns
- Featuregate
- Image
- Infrastructure
- Ingress
- Network
- Oauth
- Project
- Scheduler

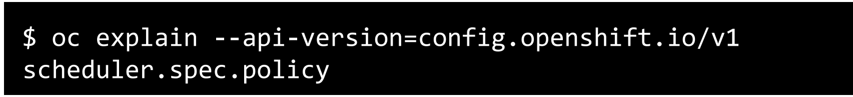


```
$ oc api-resources --api-group config.openshift.io -o
name | cut -d . -f 1
apiservers
authentications
…
scheduler
```



```
$ oc explain --api-version=config.openshift.io/v1
scheduler.spec
KIND:     Scheduler
VERSION:  config.openshift.io/v1

FIELDS:
   defaultNodeSelector    <string>
     defaultNodeSelector helps set the cluster-wide
default
     node selector to restrict pod placement to
specific
```



```
$ oc explain --api-version=config.openshift.io/v1
scheduler.spec.policy
```

Graphical Re-configuration

Global Configuration

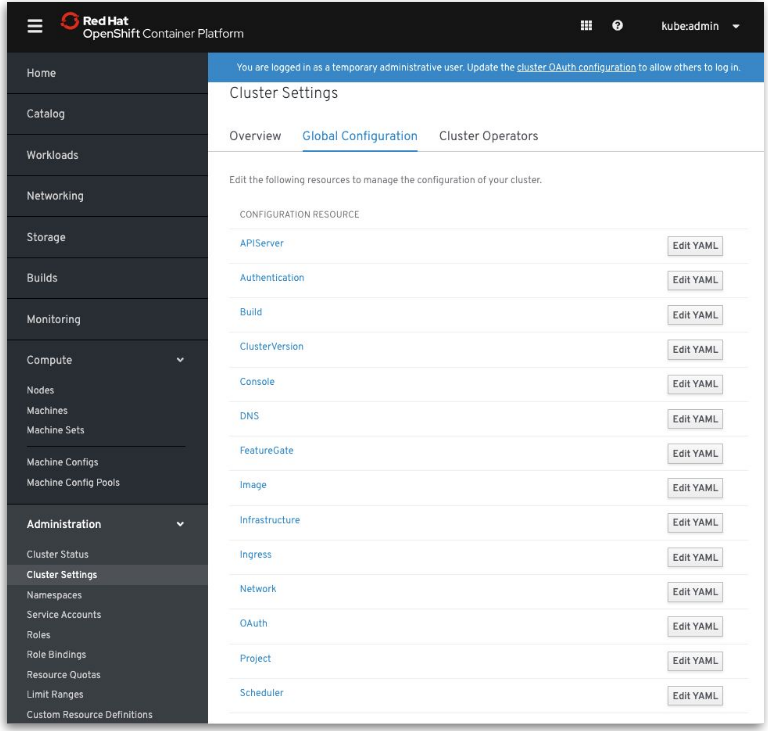
You complete most of the cluster configuration and customization after you deploy your OpenShift Container Platform cluster.

Change via Cluster Settings screen

Once you have discovered your desired settings (prev. slide), changes can be made via Console or CLI.

Operators apply these updates

One or more Operators are responsible for propagating these settings through the infrastructure



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Cluster Settings

Edit the following resources to manage the configuration of your cluster.

CONFIGURATION RESOURCE	
APIServer	Edit YAML
Authentication	Edit YAML
Build	Edit YAML
ClusterVersion	Edit YAML
Console	Edit YAML
DNS	Edit YAML
FeatureGate	Edit YAML
Image	Edit YAML
Infrastructure	Edit YAML
Ingress	Edit YAML
Network	Edit YAML
OAuth	Edit YAML
Project	Edit YAML
Scheduler	Edit YAML

Generally Available

Network Configuration

Example #1: Operator-Assisted Ingress Ctrlr “Sharding”

In 4.1, the way you create a router to work with a shard is different (API call versus ‘oc adm’ command). A simple config (example to right) acted upon by the ingress operator automatically integrates sharding with the external (front-end) DNS/LB configured at install-time,.

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  namespace: openshift-ingress-operator
  name: internal-apps
spec:
  domain: internal-apps.dmace.devcluster.openshift.com
  routeSelector:
    matchLabels:
      environment: internal
```

Example #2: Create a Second Router

Ingress controller configuration is now a first-class object, meaning additional Ingress controllers can be created by making multiple Ingress objects. This is the preferred method for giving teams their own subdomains, replacing the ‘oc adm’ method (see right).

```
$ cat <<EOF | oc create -f -
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  namespace: openshift-ingress-operator
  name: finance-apps
spec:
  domain: finance-apps.openshift.example.com
EOF
```

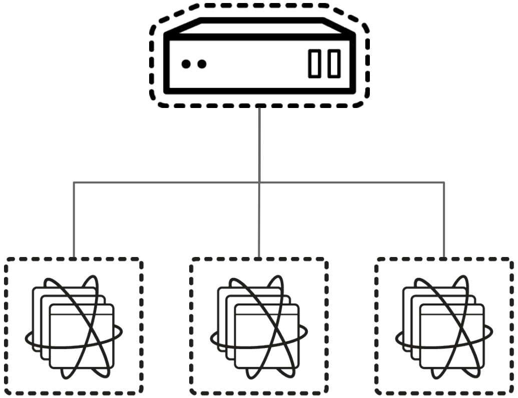
Generally Available

HAProxy Optimization

HAProxy Large Performance Increase

OCP 4.1 deploys 2 router replicas by default (improving the observed performance ~2x), and sets 4 router threads by default (also improving observed performance another ~2x).

We increased these defaults for HAProxy so that the cluster can serve more routes.

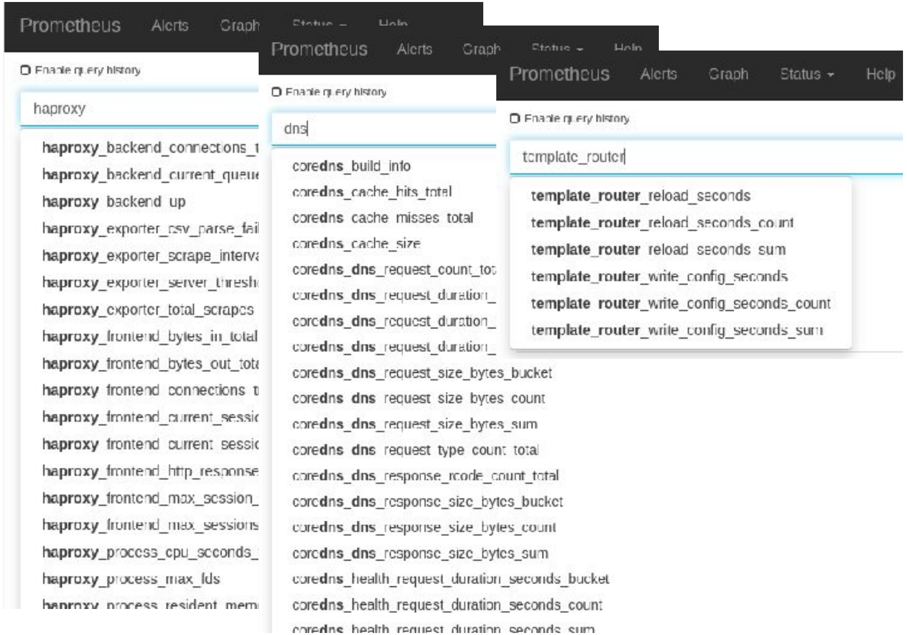


Generally Available

Networking Metrics

Networking Metrics Visibility in Prometheus

Networking metrics (HAproxy/Router, CoreDNS, etc.) for consumption by Prometheus are GA at 4.1.



```
haproxy
haproxy_backend_connections_t
haproxy_backend_current_queu
haproxy_backend_up
haproxy_exporter_csv_parse_fail
haproxy_exporter_scrape_interva
haproxy_exporter_server_thresh
haproxy_exporter_total_scrapes
haproxy_frontend_bytes_in_total
haproxy_frontend_bytes_out_tota
haproxy_frontend_connections_t
haproxy_frontend_current_sessic
haproxy_frontend_current_sessic
haproxy_frontend_http_response
haproxy_frontend_max_session_
haproxy_frontend_max_sessions
haproxy_process_cpu_seconds_
haproxy_process_max_fds
haproxy_process_resident_mem
```

```
dns
coredns_build_info
coredns_cache_hits_total
coredns_cache_misses_total
coredns_cache_size
coredns_dns_request_count_tot
coredns_dns_request_duration_
coredns_dns_request_duration_
coredns_dns_request_duration_
coredns_dns_request_size_bytes_bucket
coredns_dns_request_size_bytes_count
coredns_dns_request_size_bytes_sum
coredns_dns_request_type_count_total
coredns_dns_response_rcode_count_total
coredns_dns_response_size_bytes_bucket
coredns_dns_response_size_bytes_count
coredns_dns_response_size_bytes_sum
coredns_health_request_duration_seconds_bucket
coredns_health_request_duration_seconds_count
coredns_health_request_duration_seconds_sum
```

```
template_router
template_router_reload_seconds
template_router_reload_seconds_count
template_router_reload_seconds_sum
template_router_write_config_seconds
template_router_write_config_seconds_count
template_router_write_config_seconds_sum
```

Networking Re-configuration

Networking Advanced Settings

These are the OpenShift SDN settings that can be tweaked at install-time:

- Mode: NetworkPolicy, Multitenant, Subnet
- VXLAN Port Number
- MTU (autodetected, once)
- External OpenVSwitch

[How to Modify Advanced Network Configuration Parameters](#)

```
spec:
  defaultNetwork:
    type: OpenShiftSDN
    openshiftSDNConfig:
      mode: NetworkPolicy
      vxlanPort: 4789
      mtu: 1450
      useExternalOpenvswitch: false
```

NOTE: *Most network settings cannot be changed safely and affect the entire cluster.* The operator will prevent unsafe changes. If you need to force a change to a *non-production* cluster, see the operator README for the command, but a cluster re-install is likely to be the better choice.

Product Manager: Marc Curry

Generally Available

kube-proxy Re-configuration

kube-proxy Advanced Settings

These are the kube-proxy settings that can be tweaked at install-time:

- iptablesSyncPeriod
- bindAddress
- proxyArguments (a list of kube-proxy command-line flags)

How to Modify Advanced Network Configuration Parameters

```
spec:
kubeProxyConfig:
   iptablesSyncPeriod: 30s
   bindAddress: 0.0.0.0
   proxyArguments:
     iptables-min-sync-period: ["30s"]
```

NOTE: *Most network settings cannot be changed safely and affect the entire cluster.* The operator will prevent unsafe changes. If you need to force a change to a *non-production* cluster, see the operator README for the command, but a cluster re-install is likely to be the better choice.

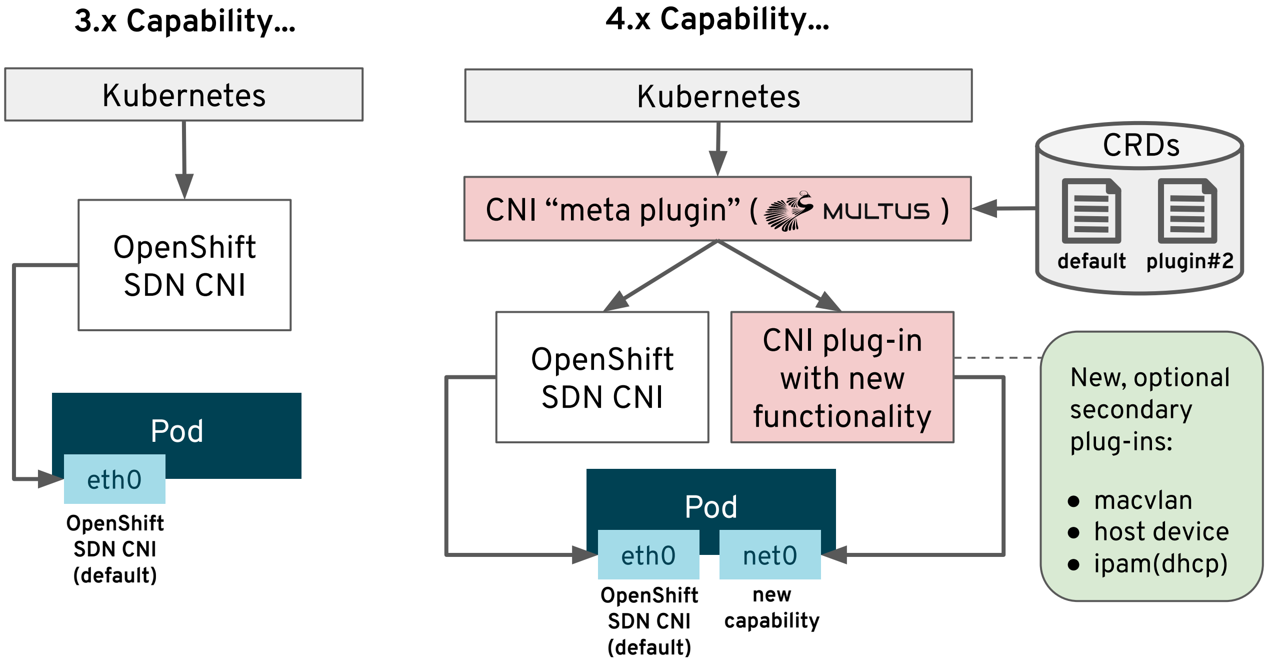
Generally Available

Networking Plug-ins

Multus Enables Multiple Networks & New Functionality to Existing Networking

The Multus CNI “meta plugin” for Kubernetes enables one to create multiple network interfaces per pod, and assign a CNI plugin to each interface created.

1. Create pod annotation(s) to call out a list of intended network attachments…
2. …each pointing to CNI network configurations packed inside CRD objects



New, optional secondary plug-ins:

- macvlan
- host device
- ipam(dhcp)

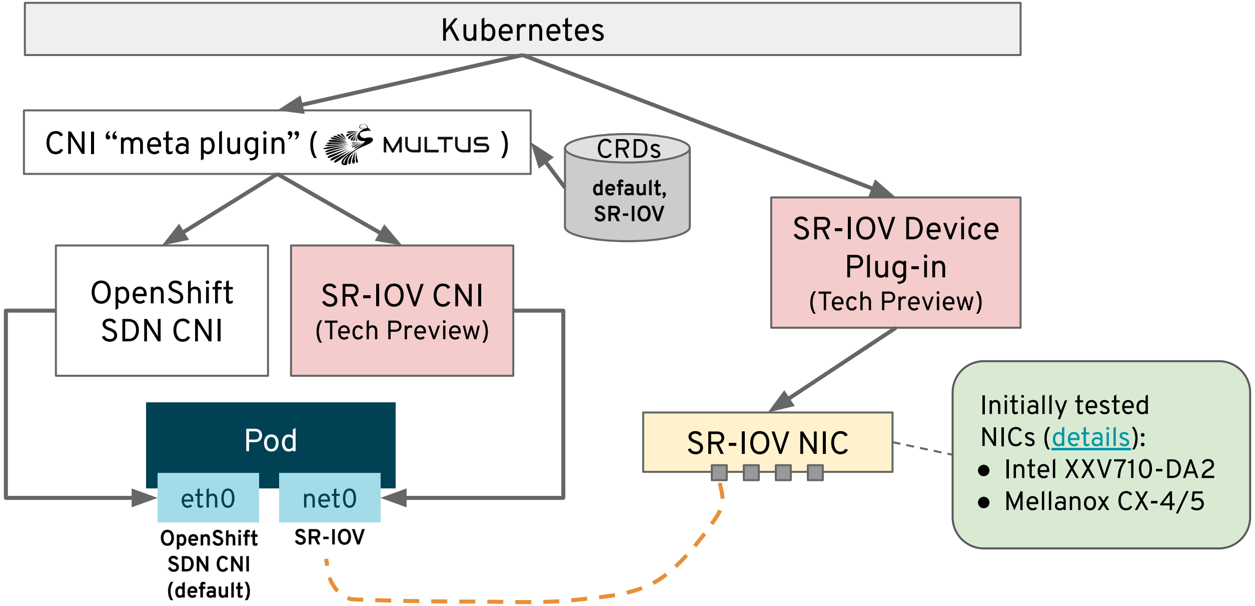
Generally Available

High-Performance Networking (Tech Preview)

Approach Line-Rate Performance to the Pod

OCP 4.1 includes the capability to use specific SR-IOV hardware on cluster nodes to brings near-line rate network performance to cluster pods.

Configuring SR-IOV


Kubernetes
CNI “meta plugin” (MULTUS)
CRDs
default, SR-IOV
SR-IOV Device Plug-in (Tech Preview)
OpenShift SDN CNI
SR-IOV CNI (Tech Preview)
Pod
eth0
net0
OpenShift SDN CNI (default)
SR-IOV
SR-IOV NIC

Initially tested NICs (details):

- Intel XXV710-DA2
- Mellanox CX-4/5

Storage

Storage Focus

- Cluster Storage Operator
  - Sets up the default storage class
  - Looks through cloud provider and sets up the correct storage class
- Drivers themselves remain in-tree for now.
- CSI coming in the future.

Supported	Dev Preview
AWS EBS	Snapshot*
VMware vSphere Disk	EFS*
NFS	Local Volume*
iSCSI	Raw Block
Fibre Channel	
HostPath	

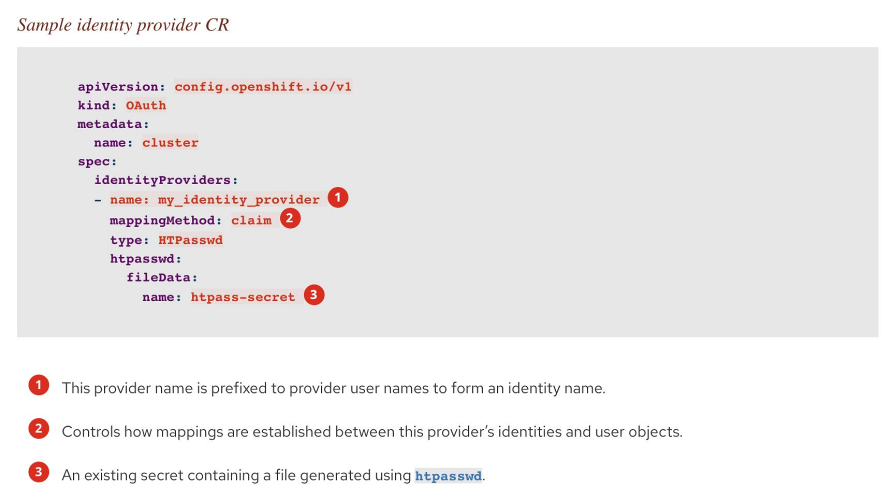
* via external provisioner

General Availability

Configuring an Identity Provider

The Cluster Authentication Operator

- Use the ***cluster-authentication-operator*** to configure an Identity Provider. The configuration is stored in the ***oauth/cluster*** custom resource object inside the cluster.
- Once that's done, you may choose to remove kubeadmin (warning: there's no way to add it back).
- All the identity providers supported in 3.11 are supported in 4.1: LDAP, GitHub, GitHub Enterprise, GitLab, Google; OpenID Connect, HTTP request headers (for SSO), Keystone, Basic authentication.
- For more information: [Understanding identity provider configuration](#) [cluster-authentication-operator](#)



Sample identity provider CR

```
apiVersion: config.openshift.io/v1
kind: OAuth
metadata:
  name: cluster
spec:
  identityProviders:
  - name: my_identity_provider ①
    mappingMethod: claim ②
    type: HTPasswd
    htpasswd:
      fileData:
        name: htpass-secret ③
```

① This provider name is prefixed to provider user names to form an identity name.

② Controls how mappings are established between this provider's identities and user objects.

③ An existing secret containing a file generated using `htpasswd`.

Generally Available

Service Certificates and the Service CA

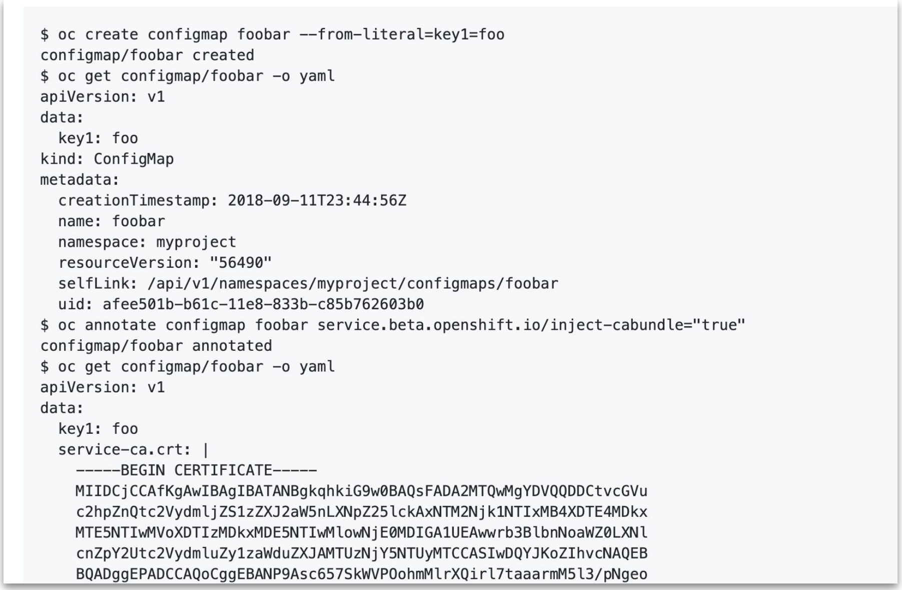
The Service CA Operator

- The Service Certificate Authority (CA) can sign server certificates for services running in the OpenShift cluster.

- The **service-ca operator** manages 3 controllers:
  service-ca controller
  configmap-cabundle-injector controller
  apiservice-cabundle-injector controller

- The Service CA is valid for one year after installation. Manual steps to refresh the Service CA are documented. Automation is targeted for 4.2.

- For more information:
  Service Serving Certificate Secrets
  openshift-service-ca-operator

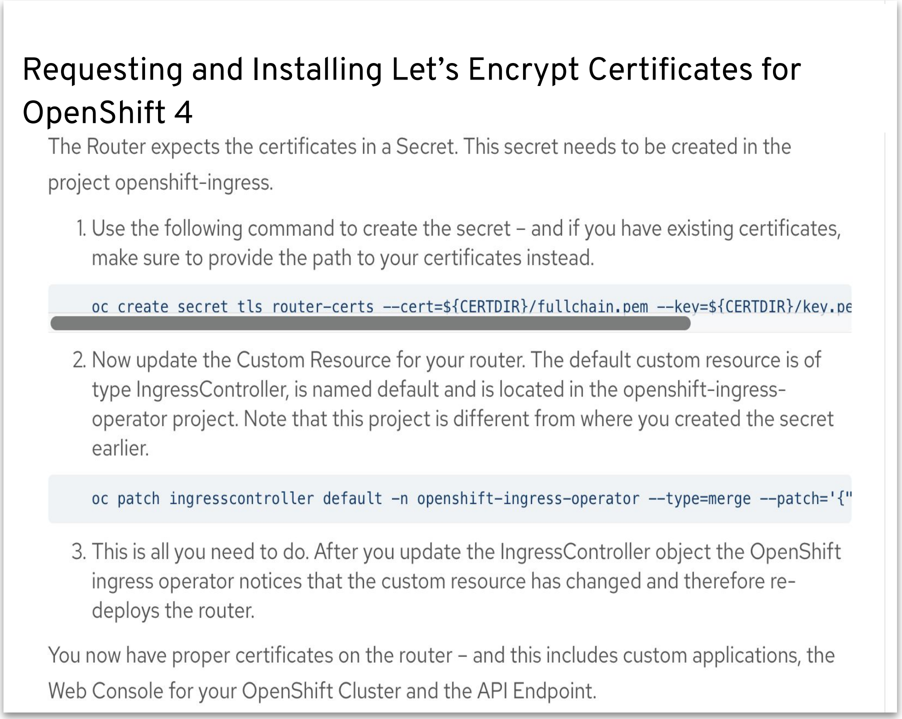


```
$ oc create configmap foobar --from-literal=key1=foo
configmap/foobar created
$ oc get configmap/foobar -o yaml
apiVersion: v1
data:
  key1: foo
kind: ConfigMap
metadata:
  creationTimestamp: 2018-09-11T23:44:56Z
  name: foobar
  namespace: myproject
  resourceVersion: "56490"
  selfLink: /api/v1/namespaces/myproject/configmaps/foobar
  uid: afee501b-b61c-11e8-833b-c85b762603b0
$ oc annotate configmap foobar service.beta.openshift.io/inject-cabundle="true"
configmap/foobar annotated
$ oc get configmap/foobar -o yaml
apiVersion: v1
data:
  key1: foo
  service-ca.crt: |
    -----BEGIN CERTIFICATE-----
    MIIDCjCCAfKgAwIBAgIBATANBgkqhkiG9w0BAQsFADA2MTQwMgYDVQQDDCtvcGVu
    c2hpZnQtc2VydmljZS1zZXJ2aW5nLXNpZ25lckAxNTM2Njk1NTIxMB4XDTE4MDkx
    MTE5NTIwMVoXDTIzMDkxMDE5NTIwMlowNjE0MDIGA1UEAwwrb3BlbnNoaWZ0LXNl
    cnZpY2Utc2VydmluZy1zaWduZXJAMTUzNjY5NTUyMTCCASIwDQYJKoZIhvcNAQEB
    BQADggEPADCCAQoCggEBANP9Asc657SkWVPOohmMlrXQirl7taaarmM5l3/pNgeo
```

Custom Certificates for External Endpoints

Custom certificates for Ingress Controllers

- In OpenShift 4 adding custom certificates for external endpoints is a post-installation task.
- Cluster ingress is managed by the Ingress Operator which configures Ingress Controllers to route traffic as specified by OpenShift Route and Kubernetes Ingress resources.
- Updating the certificate for the ingress controller(s) covers the Web console, the API endpoint, the internal Registry and custom applications. For the Registry, you can choose to define a separate route with certificate.
- For example:
  Requesting and Installing Let's Encrypt Certificates for OpenShift 4



Requesting and Installing Let's Encrypt Certificates for OpenShift 4

The Router expects the certificates in a Secret. This secret needs to be created in the project openshift-ingress.

1. Use the following command to create the secret – and if you have existing certificates, make sure to provide the path to your certificates instead.

```
oc create secret tls router-certs --cert=${CERTDIR}/fullchain.pem --key=${CERTDIR}/key.pe
```

2. Now update the Custom Resource for your router. The default custom resource is of type IngressController, is named default and is located in the openshift-ingress-operator project. Note that this project is different from where you created the secret earlier.

```
oc patch ingresscontroller default -n openshift-ingress-operator --type=merge --patch='{"
```

3. This is all you need to do. After you update the IngressController object the OpenShift ingress operator notices that the custom resource has changed and therefore re-deploys the router.

You now have proper certificates on the router – and this includes custom applications, the Web Console for your OpenShift Cluster and the API Endpoint.

Generally Available

Registry Configuration

Image Registry Operator Configuration Parameters

Several parameters support changing the registry configuration. Those are defined in the **configs.imageregistry.operator.openshift.io** resource

Additional configuration by ConfigMap and Secret resources located within the **openshift-image-registry** namespace

Documentation:

https://docs.openshift.com/container-platform/4.1/registry/configuring-registry-operator.html#registry-operator-configuration-resource-overview_configuring-registry-operator

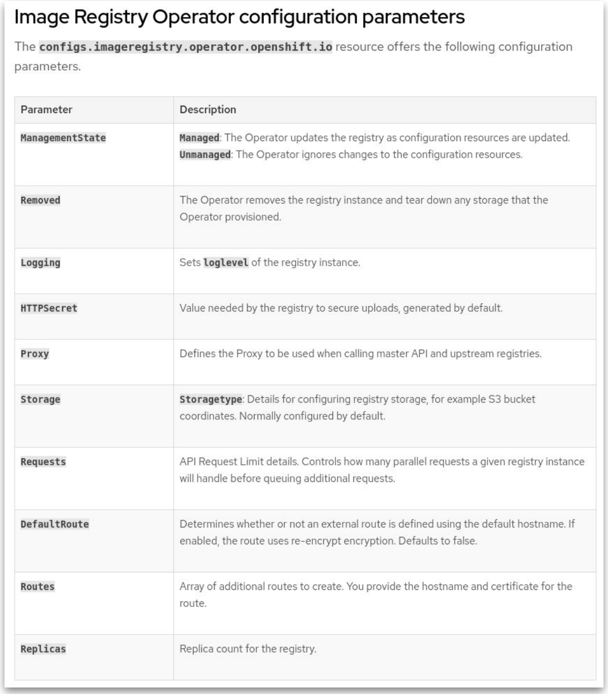


Image Registry Operator configuration parameters

The `configs.imageregistry.operator.openshift.io` resource offers the following configuration parameters.

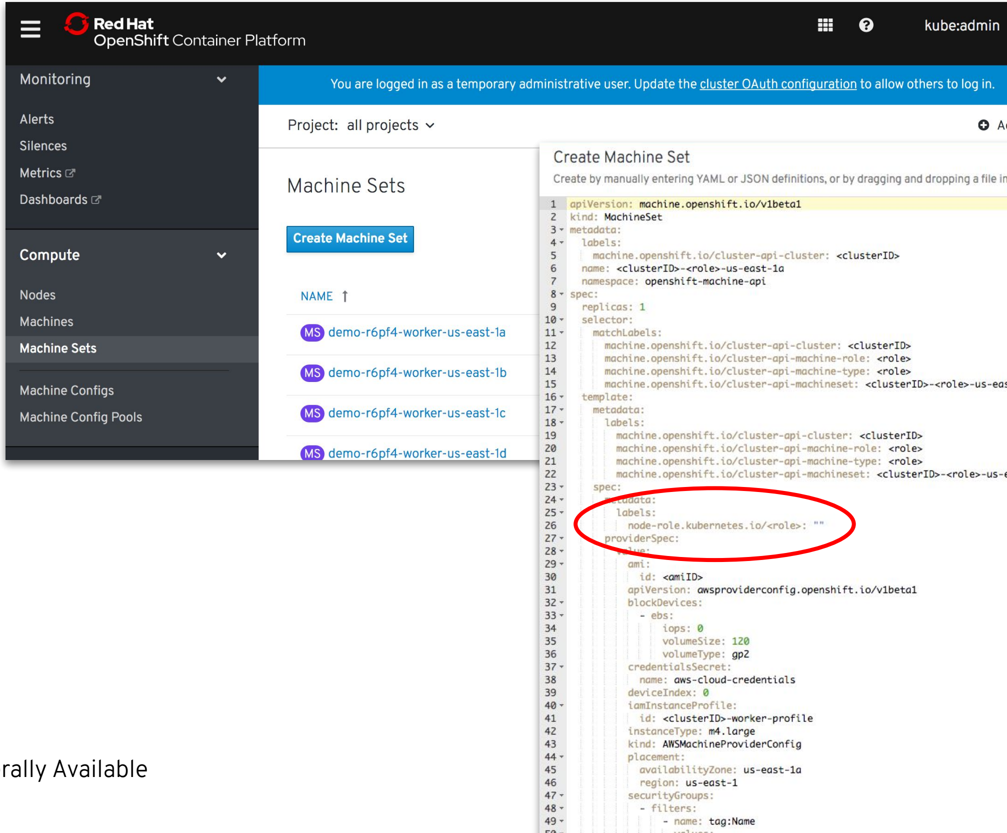
Parameter	Description
`ManagementState`	`Managed`: The Operator updates the registry as configuration resources are updated. `Unmanaged`: The Operator ignores changes to the configuration resources.
`Removed`	The Operator removes the registry instance and tear down any storage that the Operator provisioned.
`Logging`	Sets `loglevel` of the registry instance.
`HTTPSecret`	Value needed by the registry to secure uploads, generated by default.
`Proxy`	Defines the Proxy to be used when calling master API and upstream registries.
`Storage`	`Storagetype`: Details for configuring registry storage, for example S3 bucket coordinates. Normally configured by default.
`Requests`	API Request Limit details. Controls how many parallel requests a given registry instance will handle before queuing additional requests.
`DefaultRoute`	Determines whether or not an external route is defined using the default hostname. If enabled, the route uses re-encrypt encryption. Defaults to false.
`Routes`	Array of additional routes to create. You provide the hostname and certificate for the route.
`Replicas`	Replica count for the registry.

Generally Available

Infrastructure MachineSets

MachineSets that are purpose built for Infrastructure Services

- Elasticsearch, Prometheus, Router, Registry
- Out-of-box installer does not create a MachineSets dedicated for Infra services
- Create a MachineSet via console or cli and label them with desired roles
- Redeploy Infra Services with nodeSelector set to the designated role



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Create Machine Set

Create by manually entering YAML or JSON definitions, or by dragging and dropping a file in

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <clusterID>
  name: <clusterID>-<role>-us-east-1a
  namespace: openshift-machine-api
spec:
  replicas: 1
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: <clusterID>
      machine.openshift.io/cluster-api-machine-role: <role>
      machine.openshift.io/cluster-api-machine-type: <role>
      machine.openshift.io/cluster-api-machineset: <clusterID>-<role>-us-eas
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: <clusterID>
        machine.openshift.io/cluster-api-machine-role: <role>
        machine.openshift.io/cluster-api-machine-type: <role>
        machine.openshift.io/cluster-api-machineset: <clusterID>-<role>-us-e
    spec:
      metadata:
        labels:
          node-role.kubernetes.io/<role>: ""
      providerSpec:
        value:
          ami:
            id: <amiID>
          apiVersion: awsproviderconfig.openshift.io/v1beta1
          blockDevices:
            - ebs:
                iops: 0
                volumeSize: 120
                volumeType: gp2
          credentialsSecret:
            name: aws-cloud-credentials
          deviceIndex: 0
          iamInstanceProfile:
            id: <clusterID>-worker-profile
          instanceType: m4.large
          kind: AWSMachineProviderConfig
          placement:
            availabilityZone: us-east-1a
            region: us-east-1
          securityGroups:
            - filters:
                - name: tag:Name
```

[Documentation: Creating Infrastructure MachineSets]

Product Manager: Tushar Katarki

Generally Available

Moving components to dedicated Node pools

Create a new Node pool

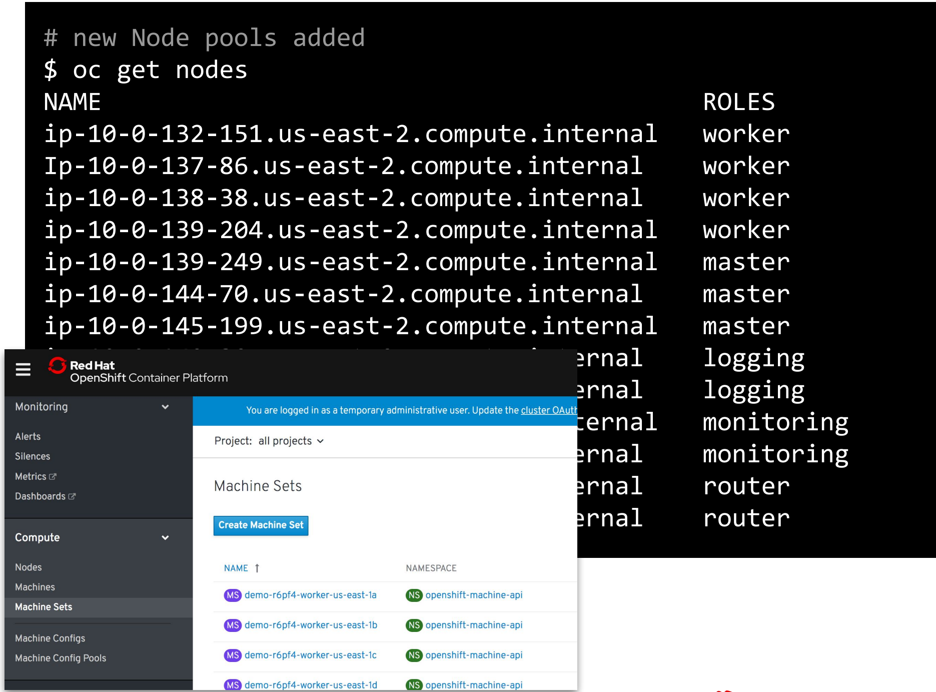
First, make a new MachineSet with a template that contains a custom label, like `role=logging`. Optionally customize the resources or security groups on this set of Nodes.

[Documentation: Roles in OpenShift]

Update Node selectors

Change the node selectors to your new Node pool labels. This process uses different CRDs based on the component:

- Cluster Monitoring (docs)
- Router (lab)
- Registry (lab)
- Logging (docs)



```
# new Node pools added
$ oc get nodes
NAME                                           ROLES
ip-10-0-132-151.us-east-2.compute.internal     worker
Ip-10-0-137-86.us-east-2.compute.internal      worker
ip-10-0-138-38.us-east-2.compute.internal      worker
ip-10-0-139-204.us-east-2.compute.internal     worker
ip-10-0-139-249.us-east-2.compute.internal     master
ip-10-0-144-70.us-east-2.compute.internal      master
ip-10-0-145-199.us-east-2.compute.internal     master
                                         ernal     logging
                                         ernal     logging
                                        ternal     monitoring
                                         ernal     monitoring
                                         ernal     router
                                         ernal     router
```

You are logged in as a temporary administrative user. Update the cluster OAuth

Project: all projects

Machine Sets

NAME	NAMESPACE
MS demo-r6pf4-worker-us-east-1a	NS openshift-machine-api
MS demo-r6pf4-worker-us-east-1b	NS openshift-machine-api
MS demo-r6pf4-worker-us-east-1c	NS openshift-machine-api
MS demo-r6pf4-worker-us-east-1d	NS openshift-machine-api

Generally Available

Additional Build Configurations

Default build configurations for buildconfigs

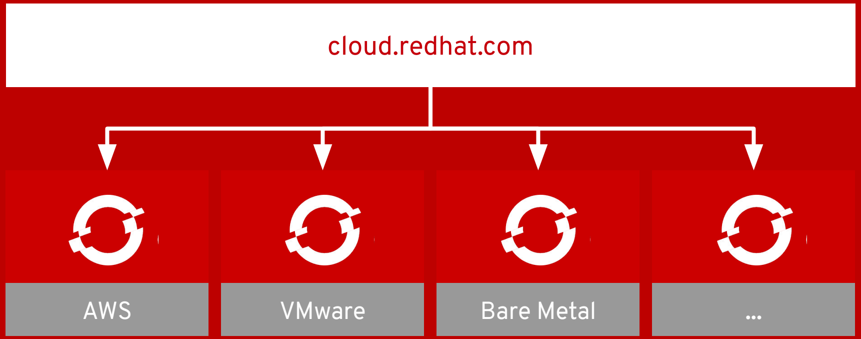
- Additional CAs to be trusted for image pull/push
- Proxy setting for image pull/push and source download
- Proxy settings for git commands
- Environment variables to set on all builds
- Docker labels to apply to resulting images
- Build resource requirements
- Default values to override on builds even if user has provided values on the buildconfig

```
apiVersion: config.openshift.io/v1
kind: Build
metadata:
 name: cluster
spec:
 additionalTrustedCA:
   name: trustedCAsConfigMap
 buildDefaults:
   defaultProxy: # http, https and no proxy
   gitProxy: # http, https and no proxy
   env: # key-values
   imageLabels:
   resources:
     limits: # cpu, memory
     requests: # cpu, memory
 buildOverrides:
   imageLabels:
   nodeSelector:
   tolerations:
```

```
oc edit build.config.openshift.io/cluster
```

Cloud-like Simplicity, Everywhere

Full-stack automated operations across any on-premises, cloud, or hybrid infrastructure



CLOUD-LIKE SIMPLICITY, EVERYWHERE

OpenShift Cluster Manager on cloud.redhat.com

Automatic registration of OpenShift clusters

View cluster versions and capacity in one place, no matter what infrastructure you are running on. Integrated with RHSM.

OpenShift Dedicated cluster management

Self-service cluster deployment, scaling, and management for OpenShift Dedicated coming soon.

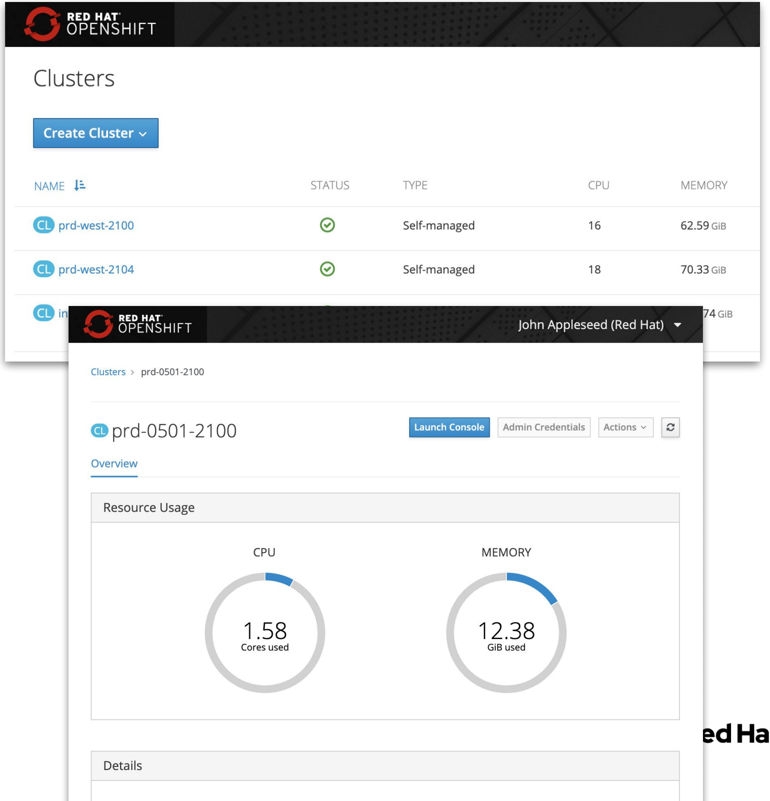
Azure Red Hat OpenShift

Information about these clusters will be coming at a later date.

Hosted in the United States

Other geographies may come later. You can opt-out too.

Generally Available



Clusters

NAME	STATUS	TYPE	CPU	MEMORY
CL prd-west-2100	✓	Self-managed	16	62.59 GiB
CL prd-west-2104	✓	Self-managed	18	70.33 GiB
CL in				74 GiB

Clusters › prd-0501-2100

CL prd-0501-2100

Resource Usage

CPU: 1.58 Cores used

MEMORY: 12.38 GiB used

Details

OpenShift Subscription Management

Moves from node management to cluster management

Entitle clusters and not nodes. Nodes too dynamic. We do not block on usage. Requires telemeter Opt-In.

Dynamically adds and removes nodes

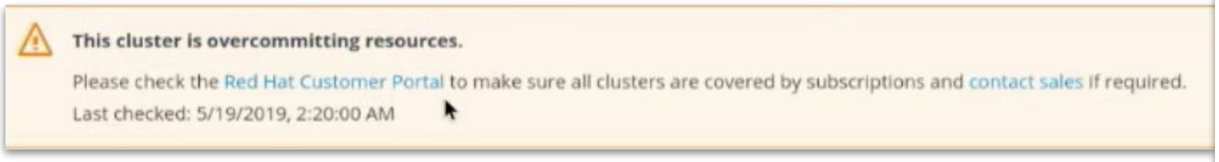
UHC will dynamically add and remove nodes from your subscription allocations to the cluster in 24 hour intervals. This will move to instantaneous across the next several releases.

Connected to the same backend as Subscription Portal and Satellite

Allocation numbers you see at cloud.redhat.com for OCP can be also seen on the subscription portal at access.redhat.com

Removes OCP Infrastructure from the count

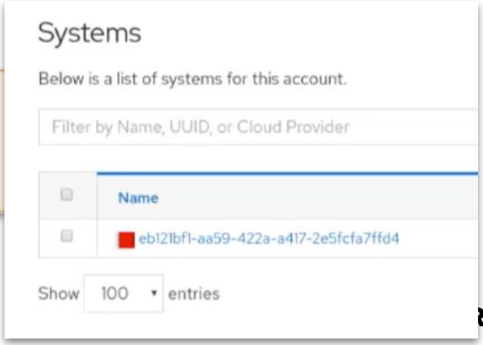
UHC will figure out which pods are your OCP infra pods and subtract out their usage from your core count so you are not charged.



This cluster is overcommitting resources.

Please check the Red Hat Customer Portal to make sure all clusters are covered by subscriptions and contact sales if required.

Last checked: 5/19/2019, 2:20:00 AM



Systems

Below is a list of systems for this account.

Filter by Name, UUID, or Cloud Provider

□	Name
□	🟥 eb121bf1-aa59-422a-a417-2e5fcfa7ffd4

Show 100 ▾ entries

Generally Available

Automated Container Operations

Fully automated day-1 and day-2 operations

AUTOMATED OPERATIONS

INSTALL	DEPLOY	HARDEN	OPERATE
Infra provisioning	Full-stack deployment	Secure defaults	Multi-cluster aware
Embedded OS	On-premises and cloud	Network isolation	Monitoring and alerts
	Unified experience	Audit and logs	Full-stack patch & upgrade
		Signing and policies	Zero downtime upgrades
			Vulnerability scanning

Generally Available

Smarter Software Updates

No downtime for well behaving apps

Applications with multiple replicas, using liveness probes, health checks and taints/tolerations

Node Pools with more than one worker and slack resources

Maintenance window for entire cluster

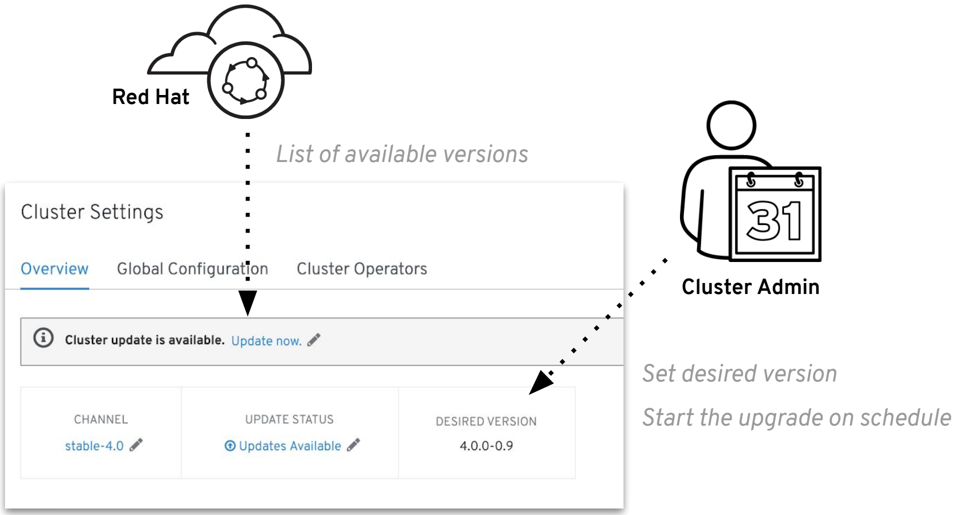
No need for separate windows for each component

Upgrade runs completely on the cluster

No more long running processes on a workstation

Constant health checking from each Operator

Operators are constantly looking for incompatibilities and issues that might arise



Cluster Settings

Overview | Global Configuration | Cluster Operators

ⓘ **Cluster update is available.** Update now.

CHANNEL	UPDATE STATUS	DESIRED VERSION
stable-4.0	Updates Available	4.0.0-0.9

Generally Available

Rolling Machine Updates

Single-click updates

- RHEL CoreOS version & config
- Kubernetes core components
- OpenShift cluster components

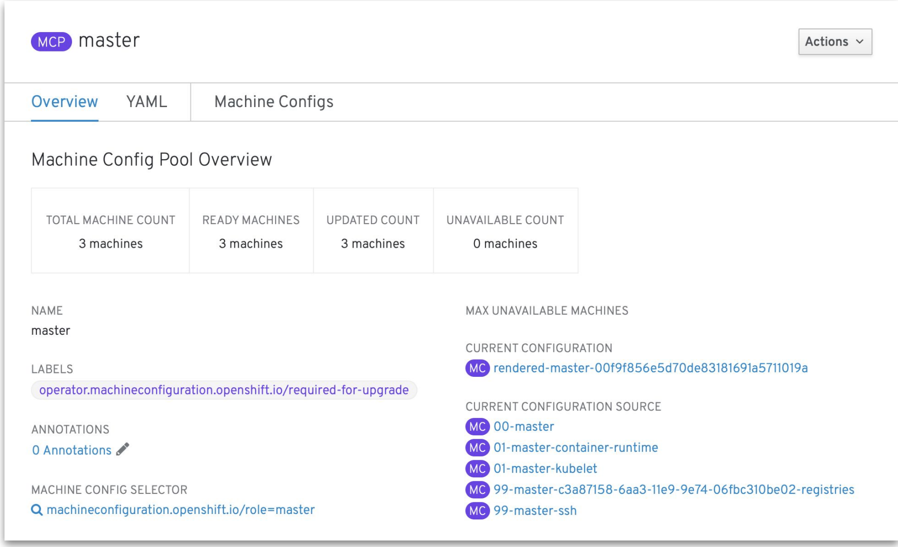
Configure how many machines can be unavailable

Set the “maxUnavailable” setting in the MachineConfigPool to maintain high availability while rolling out updates.

The default is 1.

Machine Config Operator (MCO) controls updates

This is a DaemonSet that runs on all Nodes in the cluster. When you upgrade with `oc adm upgrade`, the MCO executes these changes.



MCP master

Machine Config Pool Overview

TOTAL MACHINE COUNT	READY MACHINES	UPDATED COUNT	UNAVAILABLE COUNT
3 machines	3 machines	3 machines	0 machines

NAME
master

LABELS
operator.machineconfiguration.openshift.io/required-for-upgrade

ANNOTATIONS
0 Annotations

MACHINE CONFIG SELECTOR
machineconfiguration.openshift.io/role=master

MAX UNAVAILABLE MACHINES

CURRENT CONFIGURATION
MC rendered-master-00f9f856e5d70de83181691a5711019a

CURRENT CONFIGURATION SOURCE
MC 00-master
MC 01-master-container-runtime
MC 01-master-kubelet
MC 99-master-c3a87158-6aa3-11e9-9e74-06fbc310be02-registries
MC 99-master-ssh

Generally Available

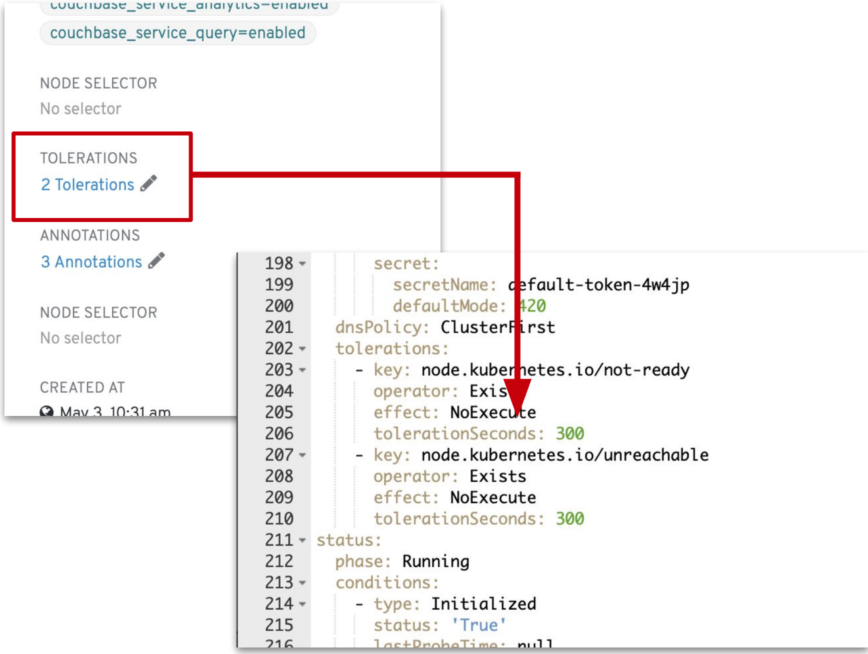
Pod Eviction Behavior

Eviction has been moved from Node Conditions to Taints

A new `DefaultTolerationSeconds` mutating admission plugin will add a 5 min eviction timeout unless specifically set by the Pod. This gives users more control vs the previous cluster-wide setting.

Don’t tolerate the node.kubernetes.io/unreachable taint

This taint is used by the cluster to evict and reschedule Nodes. If your Pod tolerates this, it will not be rescheduled on Node failure.



couchbase_service_analytics=enabled

couchbase_service_query=enabled

NODE SELECTOR
No selector

TOLERATIONS
2 Tolerations

ANNOTATIONS
3 Annotations

NODE SELECTOR
No selector

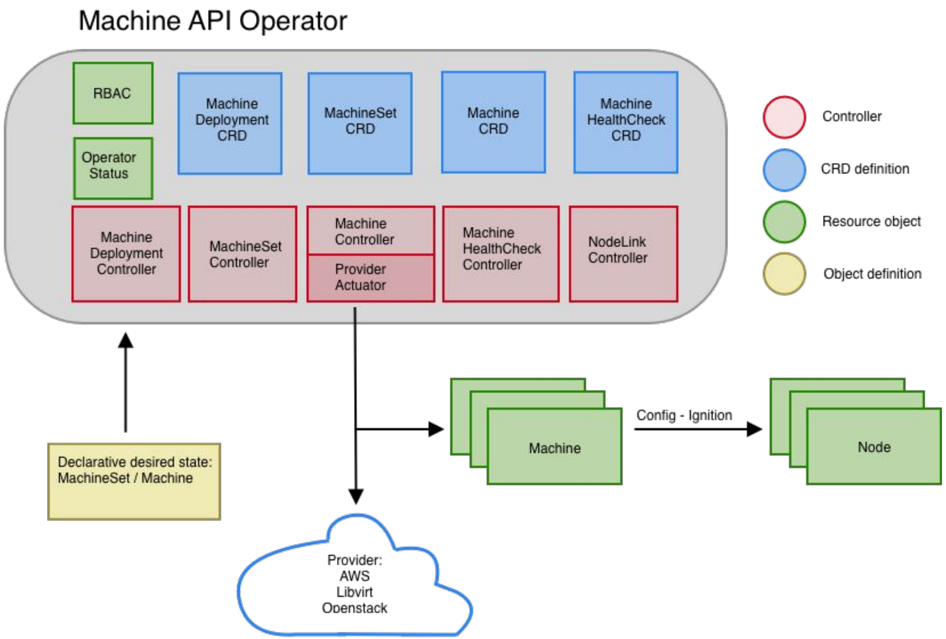
CREATED AT
May 3, 10:31 am

```
        secret:
          secretName: default-token-4w4jp
          defaultMode: 420
    dnsPolicy: ClusterFirst
    tolerations:
      - key: node.kubernetes.io/not-ready
        operator: Exists
        effect: NoExecute
        tolerationSeconds: 300
      - key: node.kubernetes.io/unreachable
        operator: Exists
        effect: NoExecute
        tolerationSeconds: 300
  status:
    phase: Running
    conditions:
      - type: Initialized
        status: 'True'
        lastProbeTime: null
```

Generally Available

Cloud API

- Provide a single view and control across multiple cluster types
- *Machine API:*
  - Set up definitions via CRDs
  - Machine: a node
  - MachineSet: think ReplicaSet
  - Actuators roll definitions across clusters
  - Nodes are drained before deletion
- *Cluster Autoscaler:* provide/remove additional nodes on demand
- AWS (4.1), Azure/GCP (target 4.2), VMWare (Future)

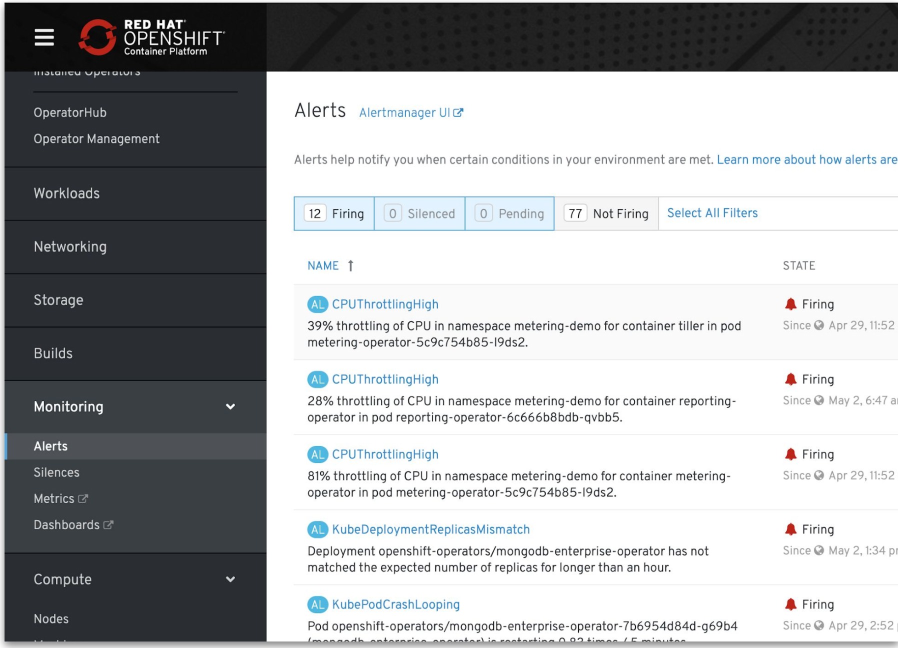


Generally Available

Cluster Monitoring

Cluster monitoring is installed by default

- Exposes resource metrics for Horizontal Pod Autoscaling (HPA) by default
    - HPA based on custom metric is tech preview
- No manual etcd monitoring configuration anymore
- New screens for managing Alerts & Silences
- More metrics available for troubleshooting purposes (e.g. HAproxy)
- Configuration via ConfigMaps and Secrets



Alerts Alertmanager UI

Alerts help notify you when certain conditions in your environment are met. Learn more about how alerts are

12 Firing | 0 Silenced | 0 Pending | 77 Not Firing | Select All Filters

NAME	STATE
AL CPUThrottlingHigh 39% throttling of CPU in namespace metering-demo for container tiller in pod metering-operator-5c9c754b85-l9ds2.	Firing Since Apr 29, 11:52
AL CPUThrottlingHigh 28% throttling of CPU in namespace metering-demo for container reporting-operator in pod reporting-operator-6c666b8bdb-qvbb5.	Firing Since May 2, 6:47 a
AL CPUThrottlingHigh 81% throttling of CPU in namespace metering-demo for container metering-operator in pod metering-operator-5c9c754b85-l9ds2.	Firing Since Apr 29, 11:52
AL KubeDeploymentReplicasMismatch Deployment openshift-operators/mongodb-enterprise-operator has not matched the expected number of replicas for longer than an hour.	Firing Since May 2, 1:34 p
AL KubePodCrashLooping Pod openshift-operators/mongodb-enterprise-operator-7b6954d84d-g69b4 (mongodb-enterprise-operator) is restarting 0.83 times / 5 minutes.	Firing Since Apr 29, 2:52

Generally Available

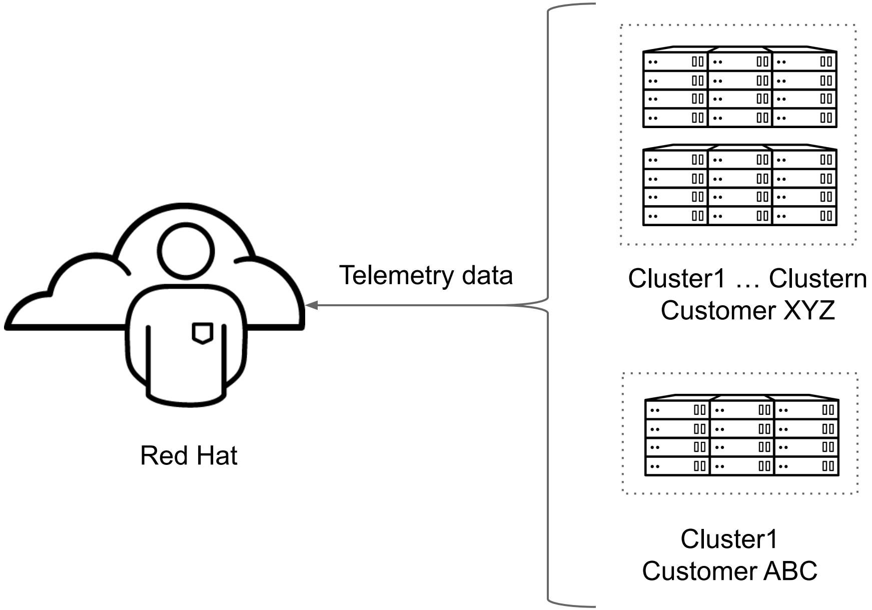
Telemetry

Collects anonymized data from any OpenShift 4 cluster deployment

- Red Hat gains quality assurance with anonymous data reporting faults encountered during upgrade
- Show utilization of all your clusters at cloud.redhat.com
- Perform subscription management at cloud.redhat.com

Opt-out is only available for self-managed OpenShift clusters but we strongly discourage that as you will lose all of the features described above.

Complete list of collected metrics

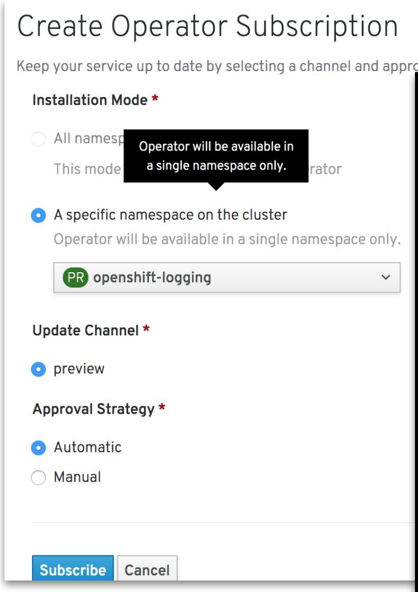


Generally Available

Cluster Logging

Cluster Logging is lifecycle managed via Operator Lifecycle Management

- Install the Elasticsearch and Cluster Logging Operators from OperatorHub
- Create an instance of Cluster Logging. fluentd, Elasticsearch and Kibana (with Operators) are created
- Changing the out-of-box configuration:
    - CPU, memory requests and limits, PVC sizes etc can be changed by editing the Cluster Logging Operator YAML
- Direct Elasticsearch and Kibana Deployments to dedicated Nodes (recommended for production usage)



Create Operator Subscription

Keep your service up to date by selecting a channel and approval strategy. The strategy determines either manual or automat

Installation Mode *

○ All namesp

This mode

Operator will be available in a single namespace only.

rator

● A specific namespace on the cluster

Operator will be available in a single namespace only.

PR openshift-logging

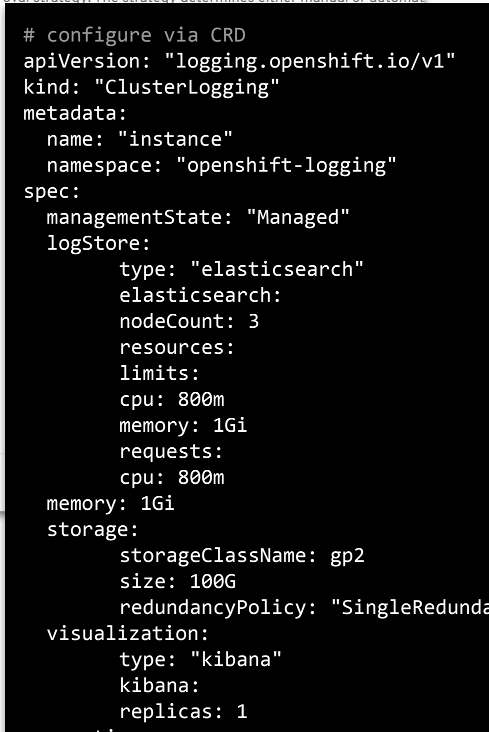
Update Channel *

● preview

Approval Strategy *

● Automatic

○ Manual

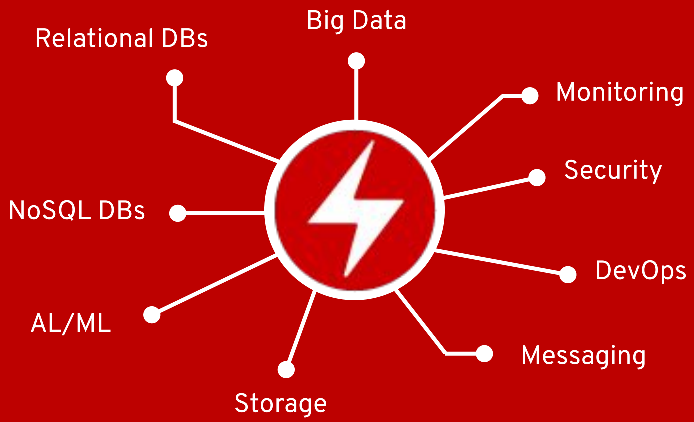


```
# configure via CRD
apiVersion: "logging.openshift.io/v1"
kind: "ClusterLogging"
metadata:
  name: "instance"
  namespace: "openshift-logging"
spec:
  managementState: "Managed"
  logStore:
        type: "elasticsearch"
        elasticsearch:
        nodeCount: 3
        resources:
        limits:
        cpu: 800m
        memory: 1Gi
        requests:
        cpu: 800m
  memory: 1Gi
  storage:
        storageClassName: gp2
        size: 100G
        redundancyPolicy: "SingleRedunda
  visualization:
        type: "kibana"
        kibana:
        replicas: 1
```

Generally Available

A broad ecosystem of workloads

Operator-backed services allow for a
SaaS experience on your own infrastructure



BROAD ECOSYSTEM OF WORKLOADS

Red Hat Certified Operators

DEVOPS	dynatrace
APM	APPDYNAMICS INSTANA New Relic. Sysdig
DATA SERVICES	GIGASPACES hazelcast PlanetScale
DATABASE	MEMSQL Couchbase mongoDB. NUODB PingCAP
SECURITY	aqua anchore BLACKDUCK BY SYNOPSYS TREMOLO SECURITY tufin
STORAGE	ROBIN STORAGEOS

Generally Available

OperatorHub data sources

Requires an online cluster

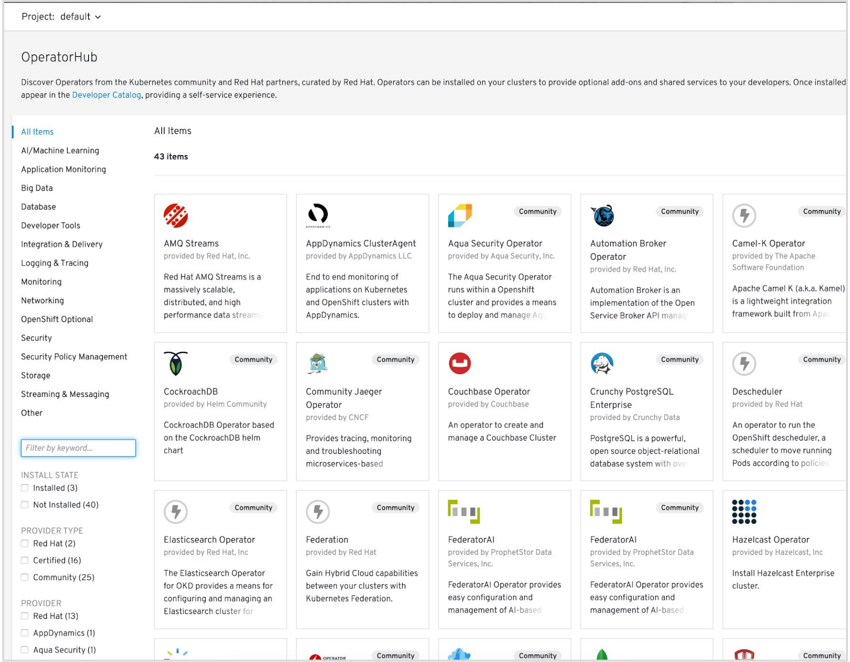
- For 4.1, the cluster must have connectivity to the internet
- Later 4.x releases will add offline capabilities

Operator Metadata

- Stored in quay.io
- Fetches channels and available versions for each Operator

Container Images

- Red Hat products and certified partners come from RHCC
- Community content comes from a variety of registries



Project: default ⌄

OperatorHub

Discover Operators from the Kubernetes community and Red Hat partners, curated by Red Hat. Operators can be installed on your clusters to provide optional add-ons and shared services to your developers. Once installed appear in the Developer Catalog, providing a self-service experience.

All Items

43 items

AMQ Streams
provided by Red Hat, Inc.
Red Hat AMQ Streams is a massively scalable, distributed, and high performance data stream

AppDynamics ClusterAgent
provided by AppDynamics LLC
End to end monitoring of applications on Kubernetes and OpenShift clusters with AppDynamics.

Community
Aqua Security Operator
provided by Aqua Security, Inc.
The Aqua Security Operator runs within a Openshift cluster and provides a means to deploy and manage Aq

Community
Automation Broker Operator
provided by Red Hat, Inc.
Automation Broker is an implementation of the Open Service Broker API manag

Community
Camel-K Operator
provided by The Apache Software Foundation
Apache Camel K (a.k.a. Kamel) is a lightweight integration framework built from Apa

Community
CockroachDB
provided by Helm Community
CockroachDB Operator based on the CockroachDB helm chart

Community
Community Jaeger Operator
provided by CNCF
Provides tracing, monitoring and troubleshooting microservices-based

Couchbase Operator
provided by Couchbase
An operator to create and manage a Couchbase Cluster

Community
Crunchy PostgreSQL Enterprise
provided by Crunchy Data
PostgreSQL is a powerful, open source object-relational database system with ove

Community
Descheduler
provided by Red Hat
An operator to run the OpenShift descheduler, a scheduler to move running Pods according to policie

Community
Elasticsearch Operator
provided by Red Hat, Inc
The Elasticsearch Operator for OKD provides a means for configuring and managing an Elasticsearch cluster for

Community
Federation
provided by Red Hat
Gain Hybrid Cloud capabilities between your clusters with Kubernetes Federation.

FederatorAI
provided by ProphetStor Data Services, Inc.
FederatorAI Operator provides easy configuration and management of AI-based

Community
FederatorAI
provided by ProphetStor Data Services, Inc.
FederatorAI Operator provides easy configuration and management of AI-based

Hazelcast Operator
provided by Hazelcast, Inc
Install Hazelcast Enterprise cluster.

Community

Community

Community

Generally Available

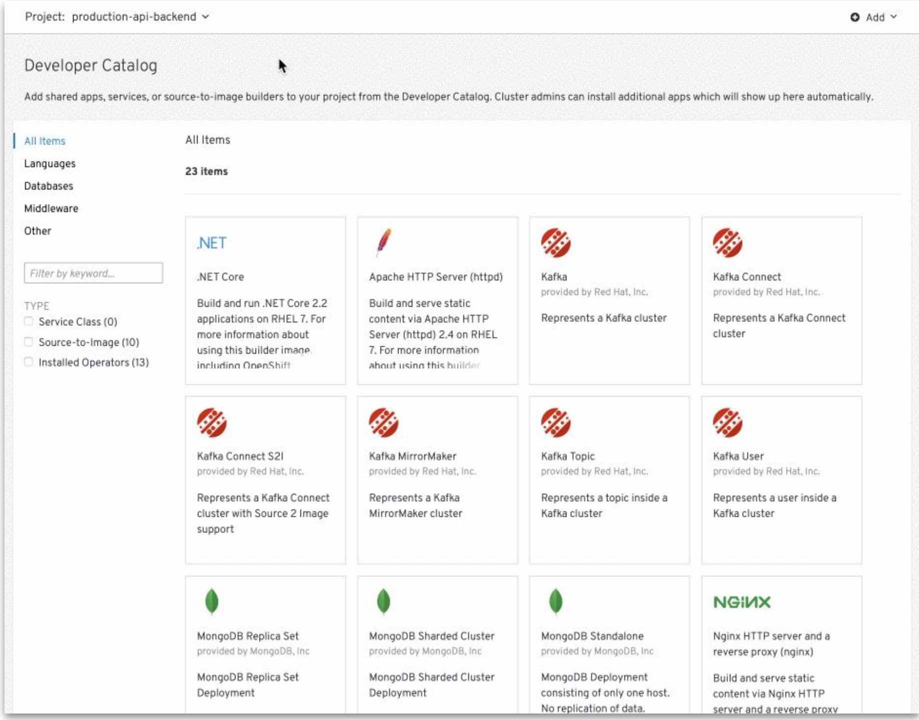
Services ready for your developers

New Developer Catalog aggregates apps

- Blended view of Operators, Templates and Broker backed services
- Operators can expose multiple CRDs. Example:
  - MongoDBReplicaSet
  - MongoDBSharded Cluster
  - MongoDBStandalone
- Developers can't see any of the admin screens

Self-service is key for productivity

- Developers with access can change settings and test out new services at any time



Project: production-api-backend ⌄ ⊕ Add ⌄

Developer Catalog

Add shared apps, services, or source-to-image builders to your project from the Developer Catalog. Cluster admins can install additional apps which will show up here automatically.

All Items
Languages
Databases
Middleware
Other

Filter by keyword...

TYPE
□ Service Class (0)
□ Source-to-Image (10)
□ Installed Operators (13)

All Items

23 items

.NET Core
Build and run .NET Core 2.2 applications on RHEL 7. For more information about using this builder image including OpenShift

Apache HTTP Server (httpd)
Build and serve static content via Apache HTTP Server (httpd) 2.4 on RHEL 7. For more information about using this builder

Kafka
provided by Red Hat, Inc.
Represents a Kafka cluster

Kafka Connect
provided by Red Hat, Inc.
Represents a Kafka Connect cluster

Kafka Connect S2I
provided by Red Hat, Inc.
Represents a Kafka Connect cluster with Source 2 Image support

Kafka MirrorMaker
provided by Red Hat, Inc.
Represents a Kafka MirrorMaker cluster

Kafka Topic
provided by Red Hat, Inc.
Represents a topic inside a Kafka cluster

Kafka User
provided by Red Hat, Inc.
Represents a user inside a Kafka cluster

MongoDB Replica Set
provided by MongoDB, Inc
MongoDB Replica Set Deployment

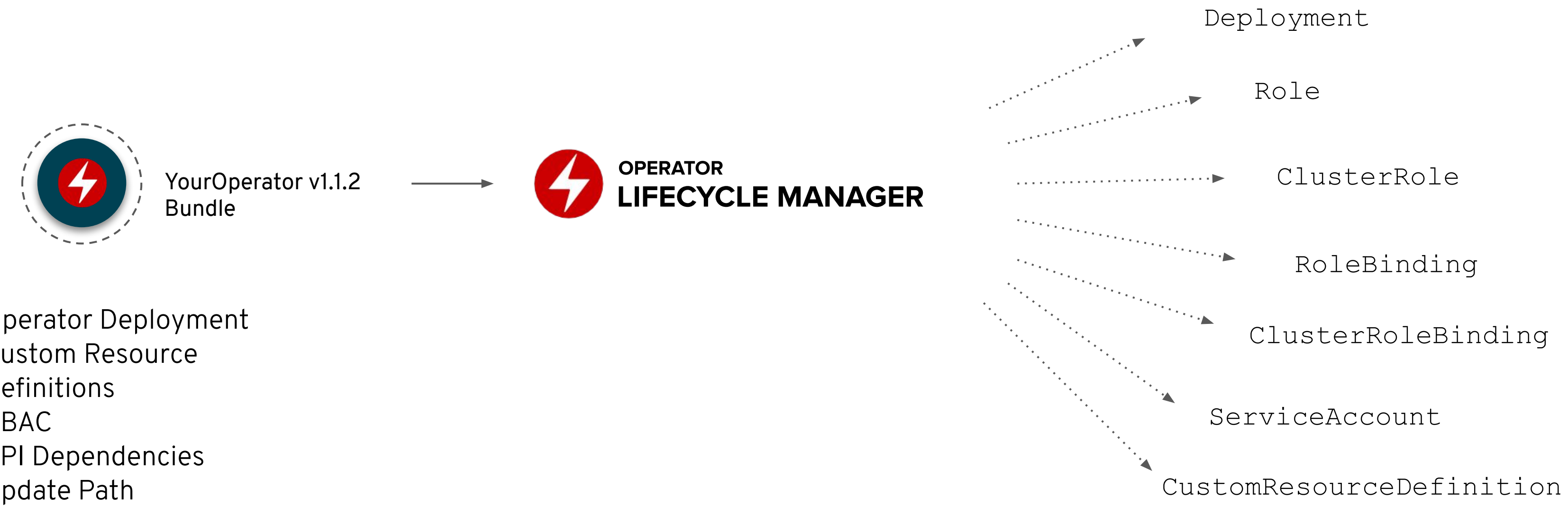
MongoDB Sharded Cluster
provided by MongoDB, Inc
MongoDB Sharded Cluster Deployment

MongoDB Standalone
provided by MongoDB, Inc
MongoDB Deployment consisting of only one host. No replication of data.

Nginx HTTP server and a reverse proxy (nginx)
Build and serve static content via Nginx HTTP server and a reverse proxy

Generally Available

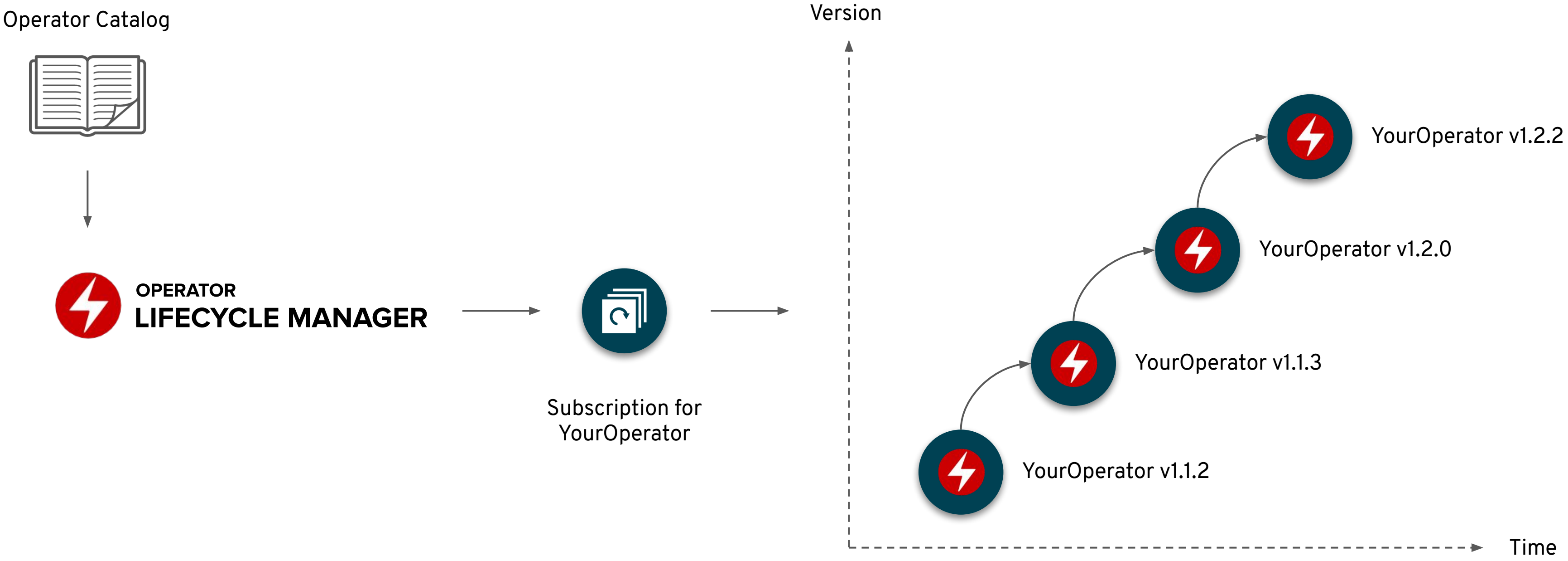
Operators as a First-Class Citizen



Operator Deployment
Custom Resource
Definitions
RBAC
API Dependencies
Update Path
Metadata

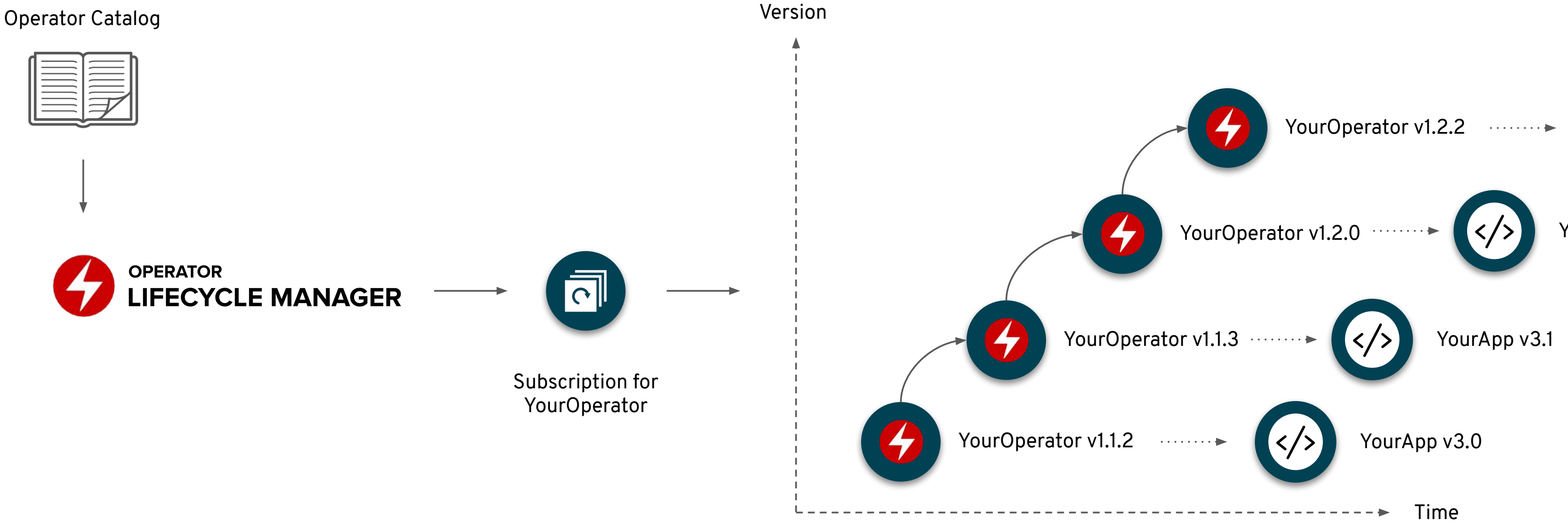
Generally Available

Operator Lifecycle Management



Generally Available

Operator Lifecycle Management



Operator Catalog

OPERATOR LIFECYCLE MANAGER

Subscription for YourOperator

Version

YourOperator v1.2.2

YourOperator v1.2.0

Y

YourOperator v1.1.3

YourApp v3.1

YourOperator v1.1.2

YourApp v3.0

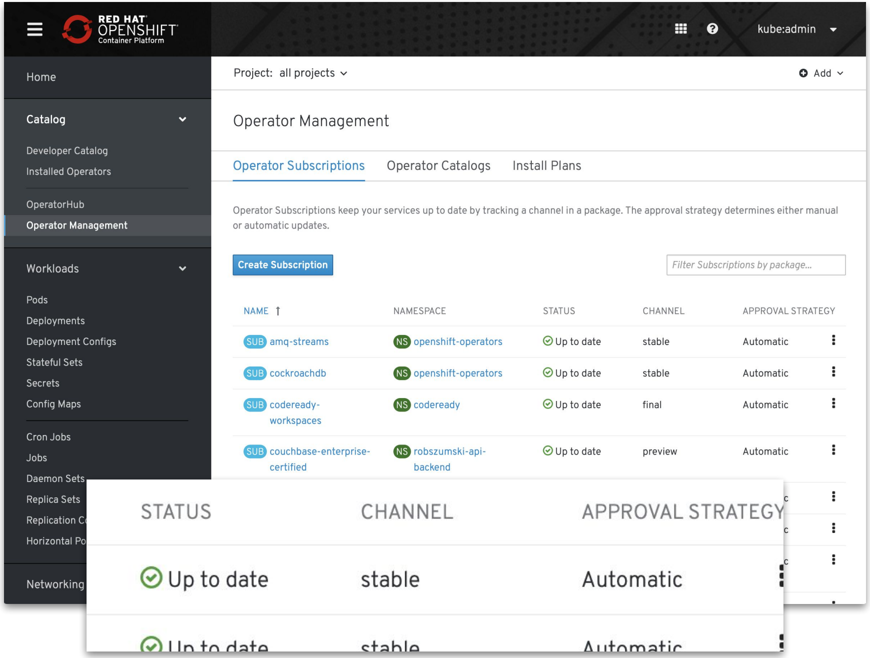
Time

Generally Available

Operator Upgrade in Detail

OperatorHub facilitates upgrades of installed Operators

- Manual or automatic modes can be chosen per Operator
- The Operator itself is upgraded by OLM via Deployment and a regular rolling upgrade
- The objects managed by the Operator use built in mechanisms to maintain HA
  - Deployments/StatefulSets
  - affinity/anti-affinity
  - taints/tolerations
  - PodDisruptionBudgets
- Behavior is dependent on the maturity of the Operator
- Optional cluster components like Cluster Logging are well behaved during upgrades



Operator Management

Operator Subscriptions | Operator Catalogs | Install Plans

Operator Subscriptions keep your services up to date by tracking a channel in a package. The approval strategy determines either manual or automatic updates.

Create Subscription

Filter Subscriptions by package...

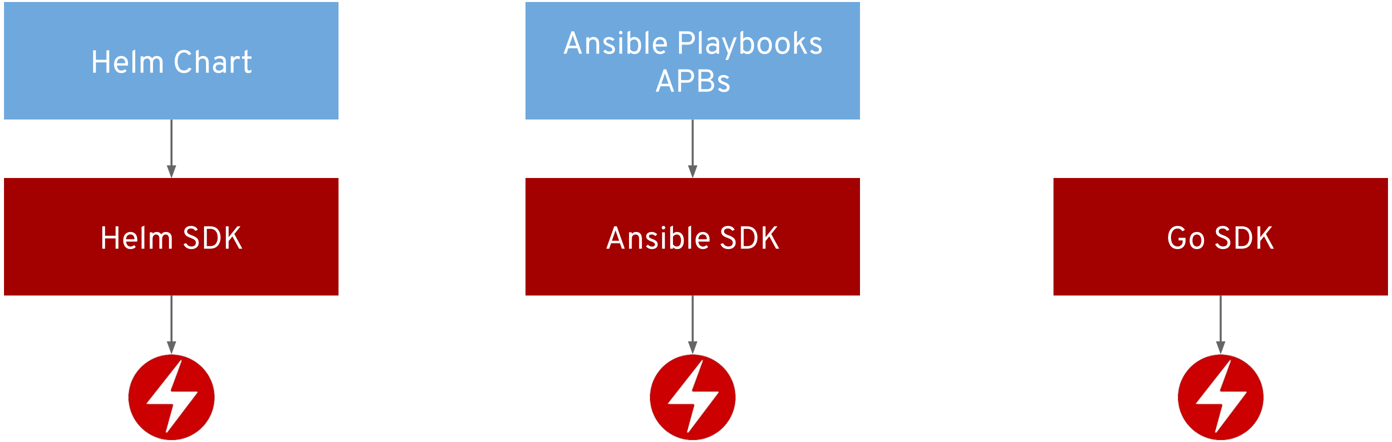
NAME ↑	NAMESPACE	STATUS	CHANNEL	APPROVAL STRATEGY
SUB amq-streams	NS openshift-operators	✓ Up to date	stable	Automatic
SUB cockroachdb	NS openshift-operators	✓ Up to date	stable	Automatic
SUB codeready-workspaces	NS codeready	✓ Up to date	final	Automatic
SUB couchbase-enterprise-certified	NS robszumski-api-backend	✓ Up to date	preview	Automatic

STATUS	CHANNEL	APPROVAL STRATEGY
✓ Up to date	stable	Automatic
✓ Up to date	stable	Automatic

Generally Available

BROAD ECOSYSTEM OF WORKLOADS

Build Operators for your apps



Helm Chart

Helm SDK

Ansible Playbooks APBs

Ansible SDK

Go SDK

Build operators from Helm chart, without any coding

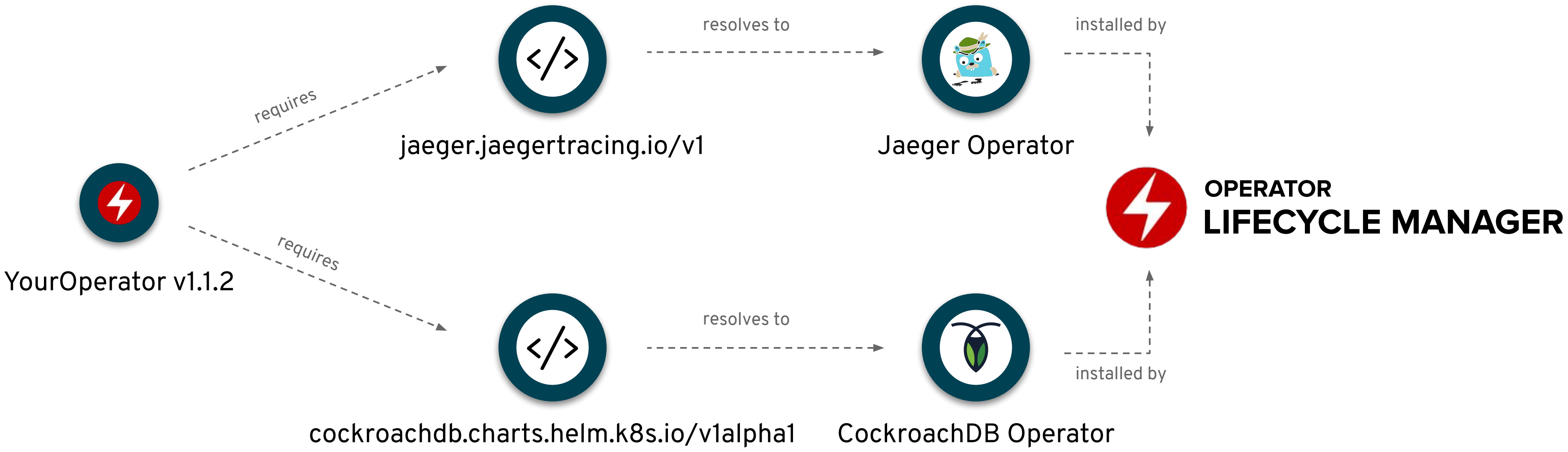
Build operators from Ansible playbooks and APBs

Build advanced operators for full lifecycle management

Generally Available

Depend on other Operators

Operator Framework Dependency Graphs



YourOperator v1.1.2

requires

jaeger.jaegertracing.io/v1

resolves to

Jaeger Operator

installed by

requires

cockroachdb.charts.helm.k8s.io/v1alpha1

resolves to

CockroachDB Operator

installed by

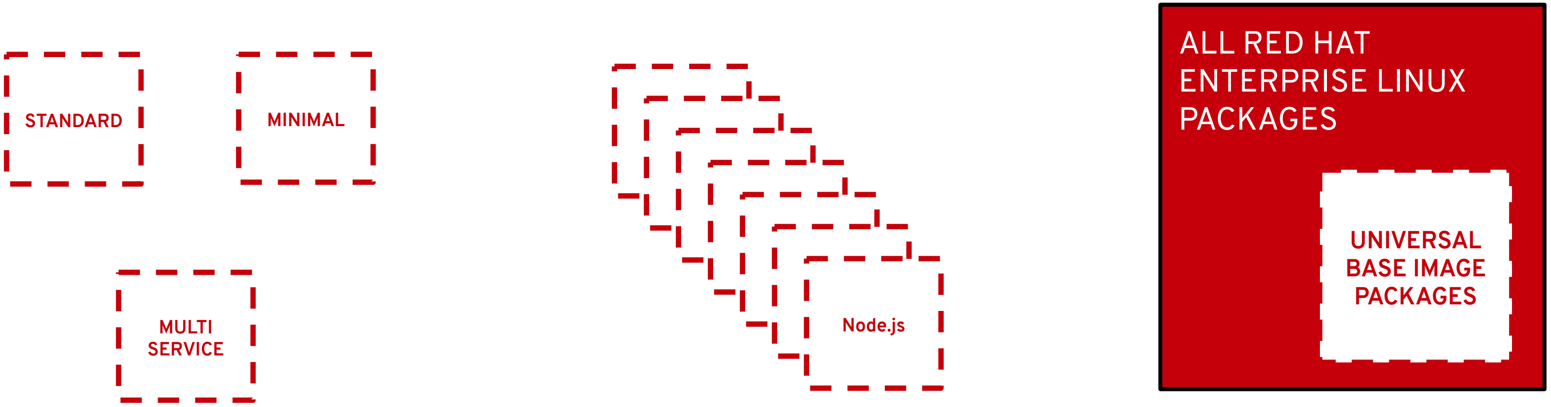
OPERATOR LIFECYCLE MANAGER

Generally Available

BROAD ECOSYSTEM OF WORKLOADS

Red Hat Universal Base Image

Enable an ecosystem of freely distributable operators for Kubernetes/OpenShift


STANDARD
MINIMAL
MULTI SERVICE
Node.js
ALL RED HAT ENTERPRISE LINUX PACKAGES
UNIVERSAL BASE IMAGE PACKAGES

Base Images

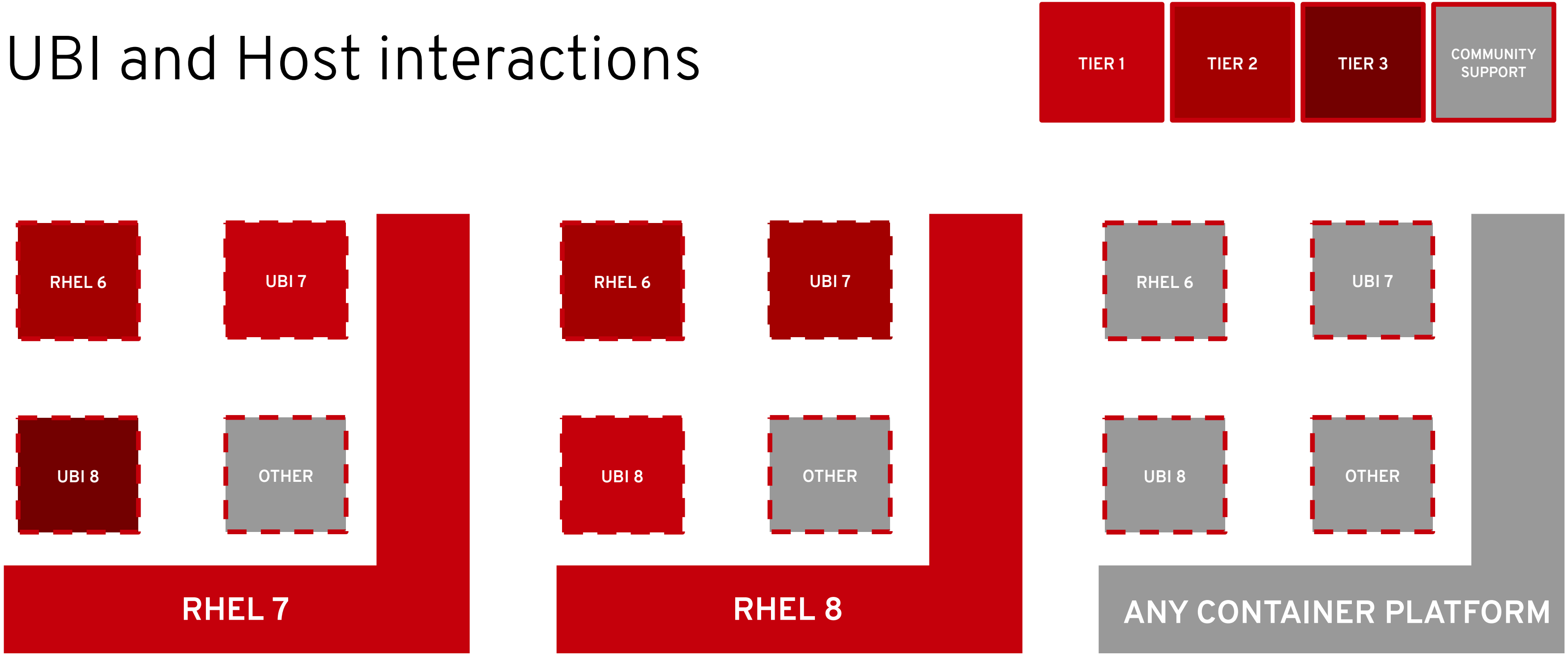
Pre-Built Language Images

Package Subset

Generally Available

BROAD ECOSYSTEM OF WORKLOADS

UBI and Host interactions



Red Hat Enterprise Linux 7

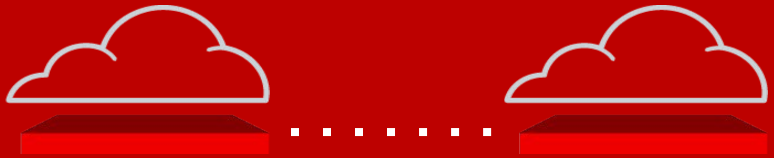
Red Hat Enterprise Linux 8

Like any community distro

Generally Available

Hosted OpenShift

Get the best of OpenShift without being on call



Hosted OpenShift Benefits

OPENSHIFT CONTAINER PLATFORM

Full Stack Automated

Simplified opinionated

cluster provisioning

Pre-existing Infrastructure

Customer managed

urces & infrastructure

provisioning

d

ries

Skip the on-call rotation

Red Hat engineers keep you up to date

Expand capacity without hassle

HOSTED OPENSHIFT

Azure Red Hat OpenShift

Deploy directly from the Azure console.

Jointly managed by Red Hat and Azure engineers.

Free your team from the distraction of ops

OpenShift Dedicated

Powerful cluster, no maintenance needed

Managed by Red Hat engineers and support

Free your team from the distraction of ops

Generally Available

Azure Red Hat OpenShift

Jointly engineered, operated, and supported by both Microsoft and Red Hat with an integrated support experience

Experience OpenShift as a native Microsoft Azure service.

- Create fully managed OpenShift clusters in minutes using `az openshift create`
- Add or remove compute nodes to match resource demand using `az openshift scale`
- 99.9% SLA
- Will soon inherit Azure regulatory compliance
- Pricing available at https://azure.microsoft.com/en-us/pricing/details/openshift/

Red Hat OpenShift

Microsoft Azure

Generally Available

OpenShift Dedicated

Dedicated with OpenShift 3

Available today, hosted on Amazon Web Services

Consumption based billing now available

Bring Your Own Cloud Account

Dedicated with OpenShift 4

Initial availability June 2019

Broader availability in fiscal Q2

OperatorHub

Red Hat products and certified Operators will be added in a curated catalog later in the year.

The Service Catalog and Brokers will not migrate to Dedicated due to their deprecation.

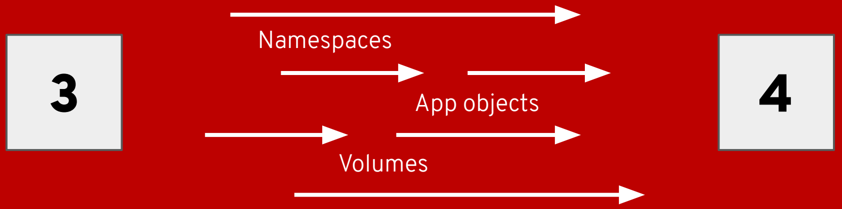
Connected to cloud.redhat.com

Clusters will appear beside other self-managed installs

Generally Available

Migrating to OpenShift 4

Tooling and advice for moving from OpenShift 3.x to 4.x



App migration experience

Using open source tooling based on Velero

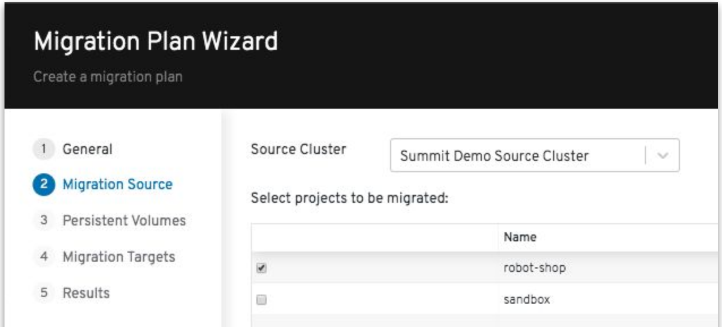
Velero is an upstream project previously known as Ark. Check out this video if you are curious and want to get a sneak peek at our capabilities.

What’s moved during a migration

- Namespaces
- Persistent Volumes (move or copy)
- All important resource objects (Deployments, StatefulSets, etc)

Available in OpenShift 4.2

Customers are anxious to get their hands on this, but we want to get it right. We would love to receive sample application workloads to test.



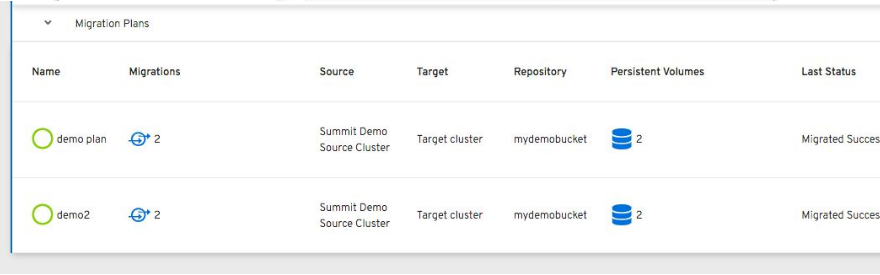
Migration Plan Wizard

Create a migration plan

Source Cluster: Summit Demo Source Cluster

Select projects to be migrated:

	Name
☑	robot-shop
□	sandbox



Migration Plans

Name	Migrations	Source	Target	Repository	Persistent Volumes	Last Status
○ demo plan	2	Summit Demo Source Cluster	Target cluster	mydemobucket	2	Migrated Succes
○ demo2	2	Summit Demo Source Cluster	Target cluster	mydemobucket	2	Migrated Succes

Product Manager: Maria Bracho

Not Available Yet

Why did we choose this migration strategy?

Reducing risk

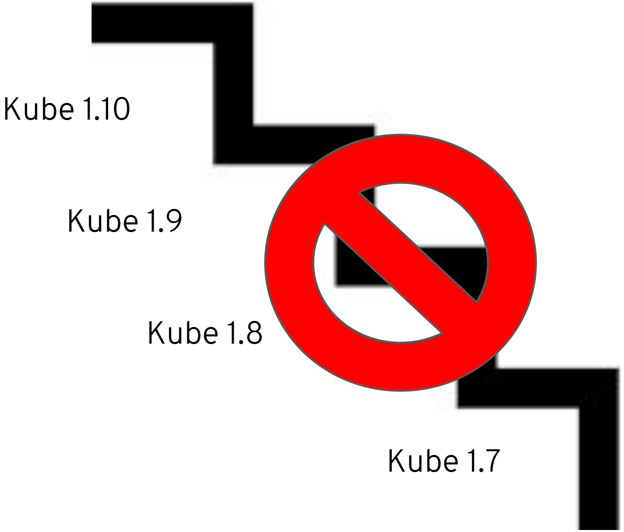
A ton of innovation went into OpenShift 4, and an in-place upgrade would have risk of failure in which there is no forwards or backwards remediation. It allows you to skip from 3.7/3.9/3.10/3.11 to 4.x. Skipping the need to install each one.

Useful for 4-to-4 migrations

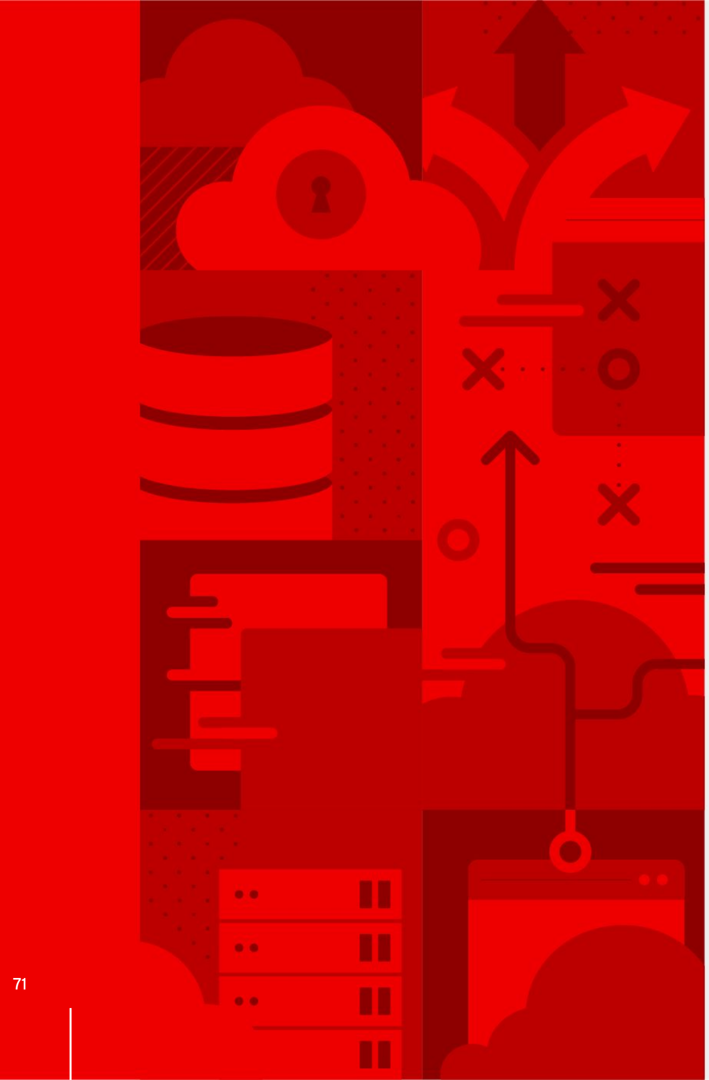
A general migration tool is frequently requested and a better long term investment. Helps you build a foundation towards making your cluster investments less fragile.

Allows for staging

Stage a mock migration before doing it live, on a Project by Project basis. Extremely useful for success.



Not Available Yet



Questions?

in linkedin.com/company/red-hat

f facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat