

OpenShift Container Platform

ARCHITECTURAL OVERVIEW

Alfred Bach
Principal Solution Architect



linkedin.com/company/red-hat



youtube.com/user/RedHatVideos



facebook.com/redhatinc



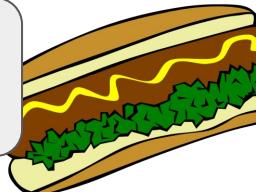
twitter.com/RedHat

OpenShift Architecture

Part 1
(9:30 - 11:00)



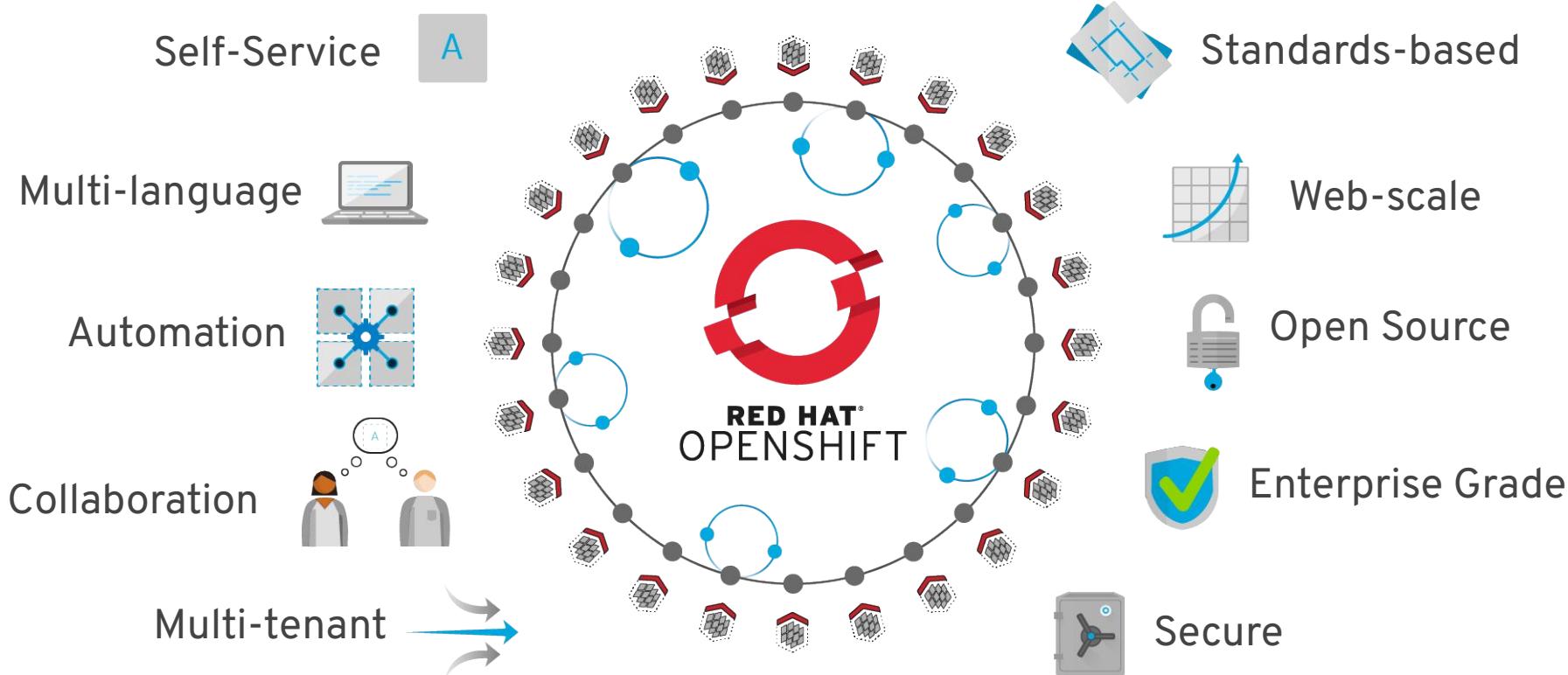
Part 2
(11:15 - 12:30)



Part 3 (13:30 - 14:30)



Functional overview



Value of OpenShift

Monitoring, Logging,
Registry, Router, Telemetry

Cluster Services

Service Mesh, Serverless,
Middleware/Runtimes, ISVs

Application Services

Dev Tools, CI/CD,
Automated Builds, IDE

Developer Services

Automated Operations

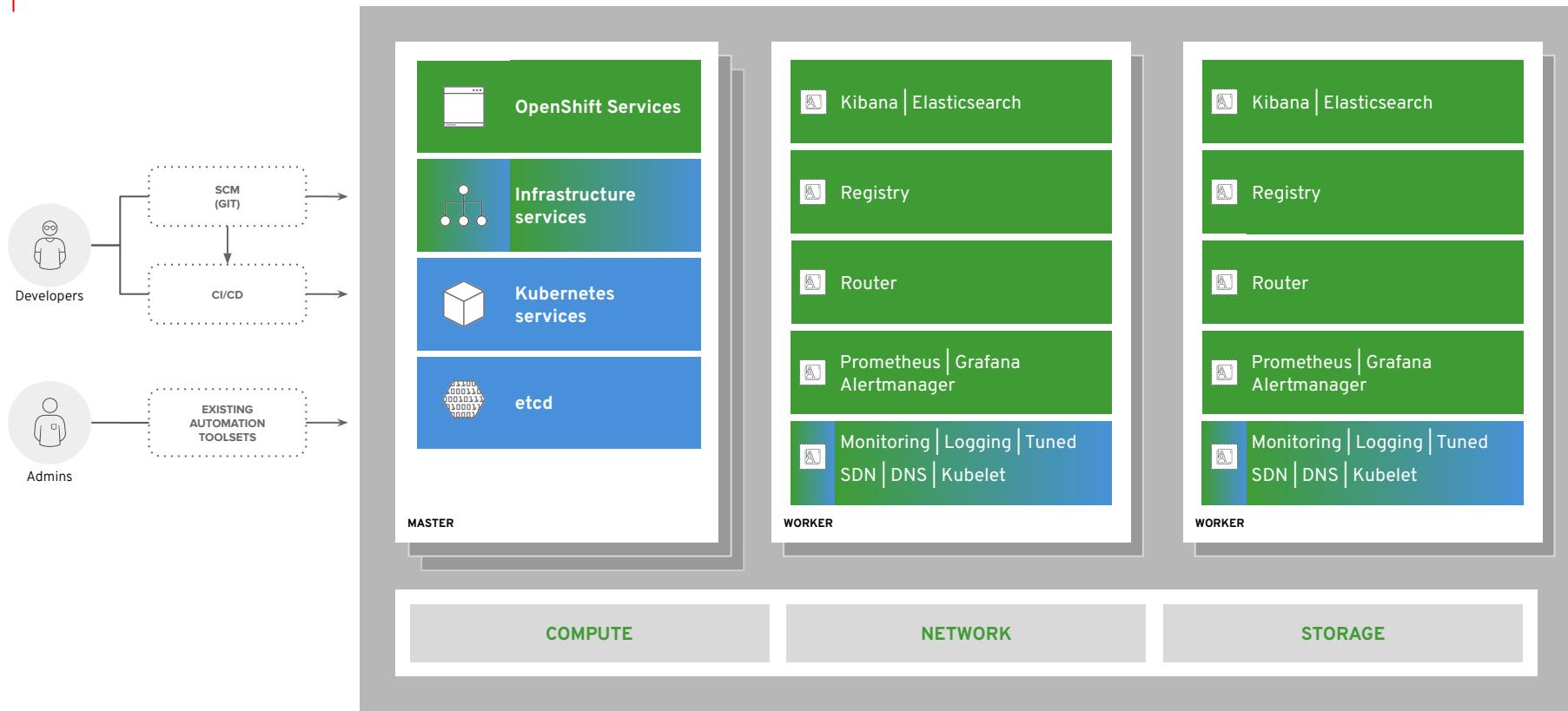
Kubernetes

Red Hat Enterprise Linux | RHEL CoreOS

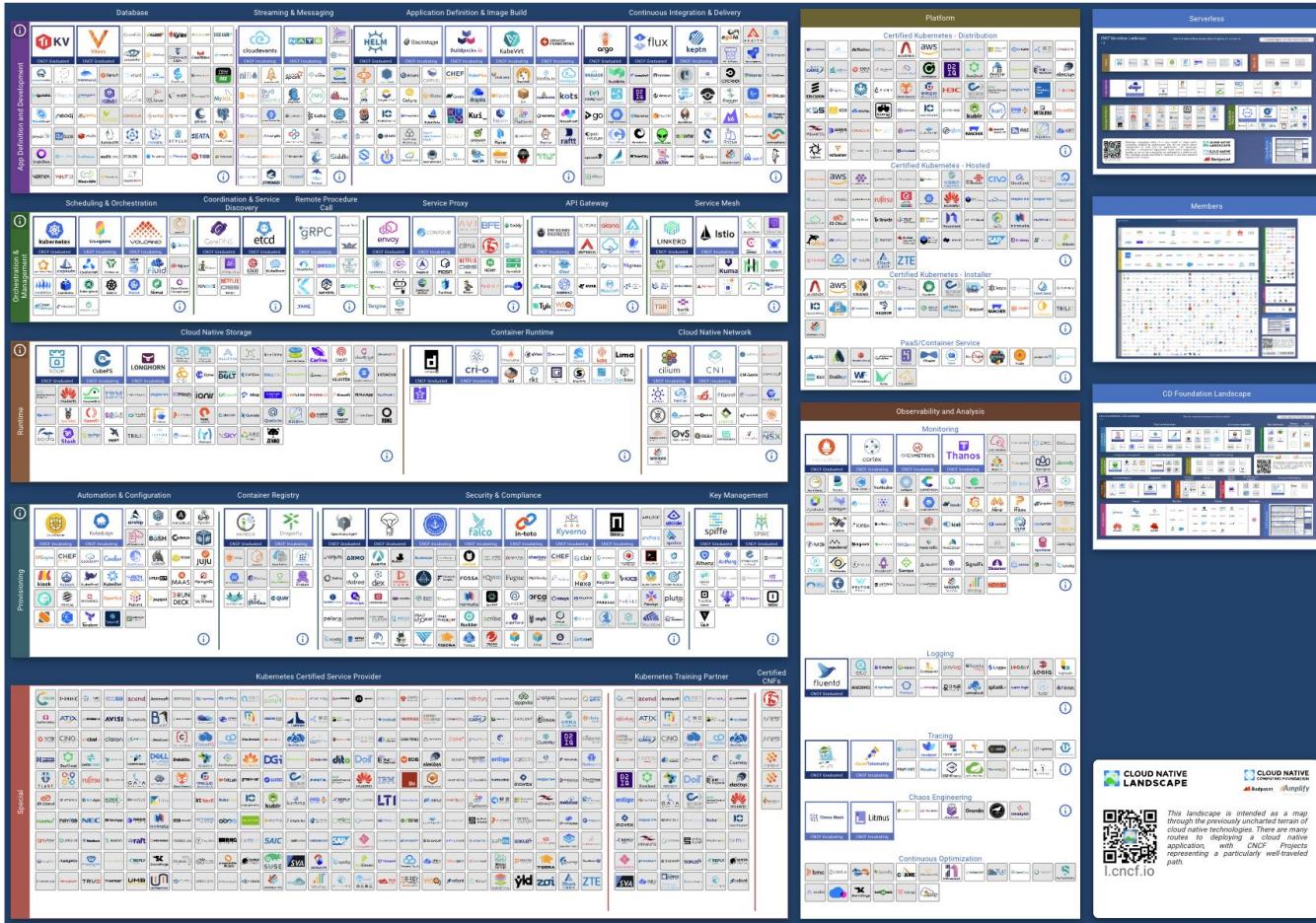
Best IT Ops Experience

CaaS \longleftrightarrow PaaS \longleftrightarrow FaaS

Best Developer Experience



CNCF Ecosystem Slide



CLOUD NATIVE
LANDSCAPE
CNCF

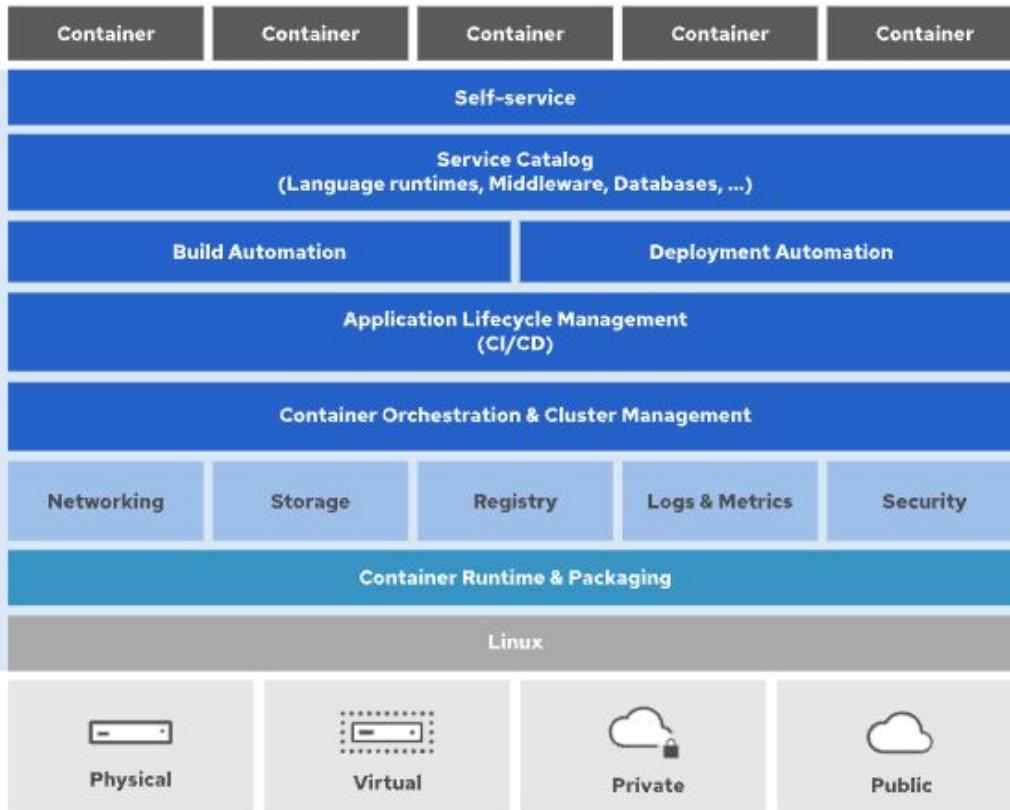
lcnf.io

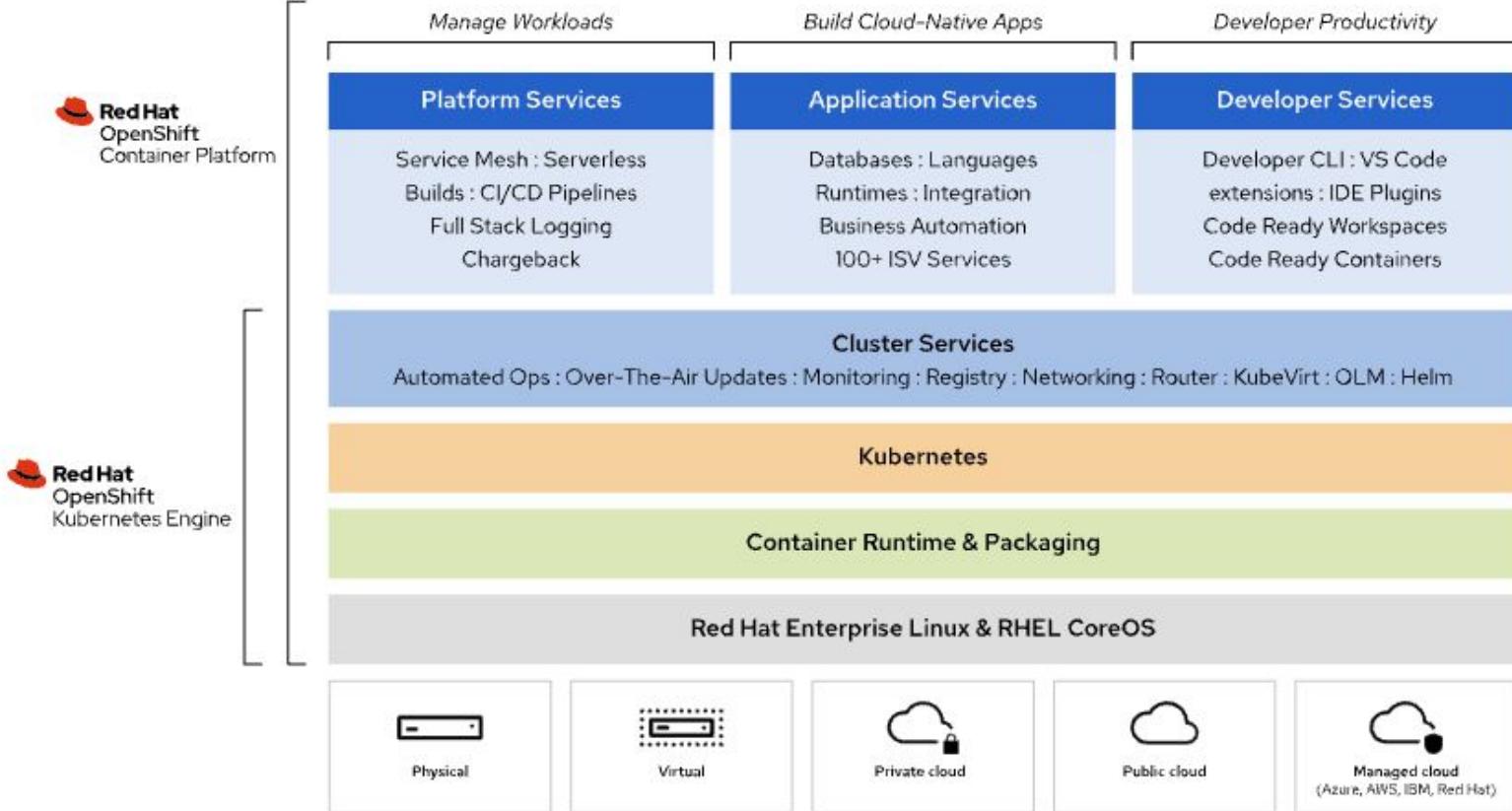
This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many more projects and paths to explore, with CNCF Projects representing a particularly well-traveled path.





Red Hat
OpenShift







OpenShift and Kubernetes core concepts

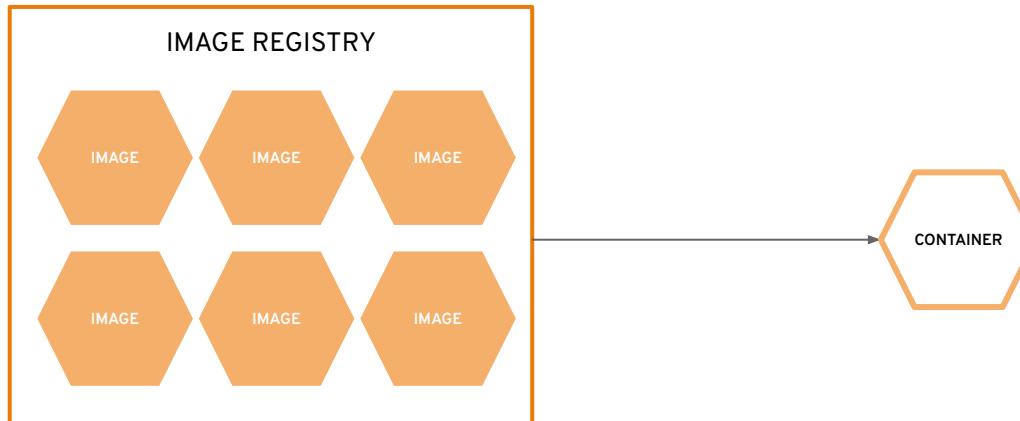
a container is the smallest compute unit



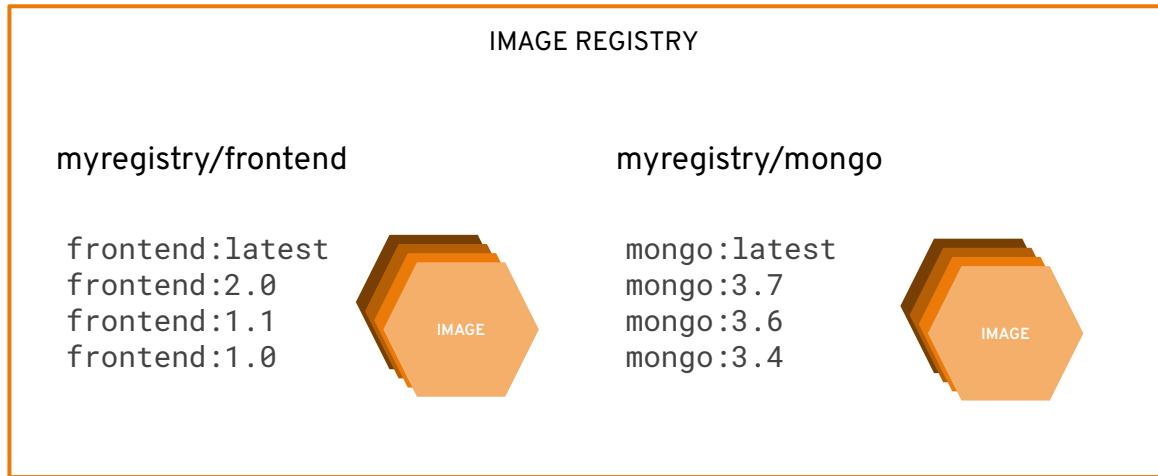
containers are created from container images



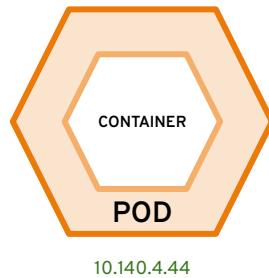
container images are stored in an image registry



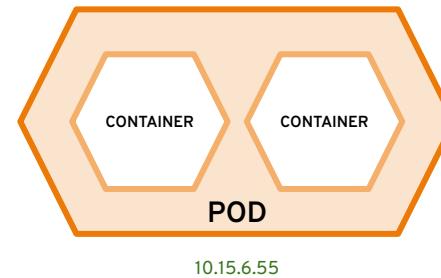
an image repository contains all versions of an image in the image registry



containers are wrapped in pods which are units of deployment and management

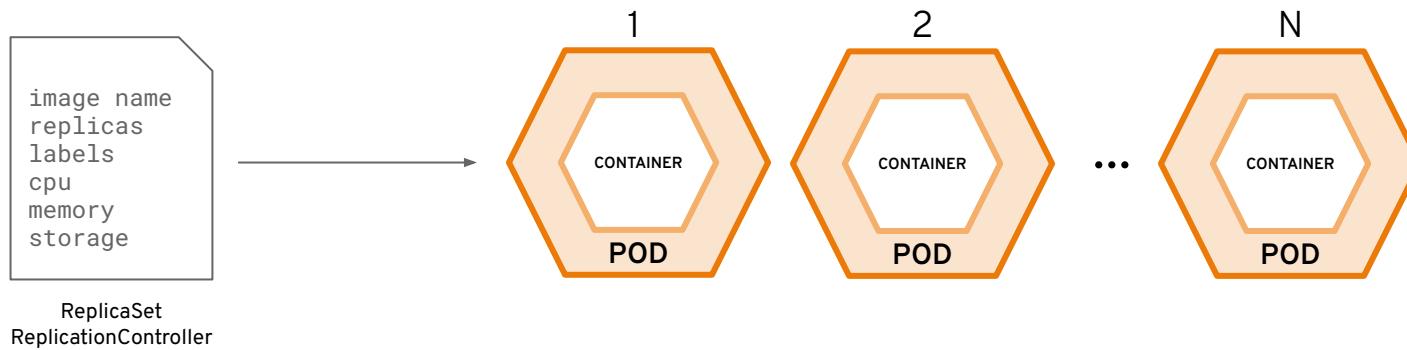


10.140.4.44

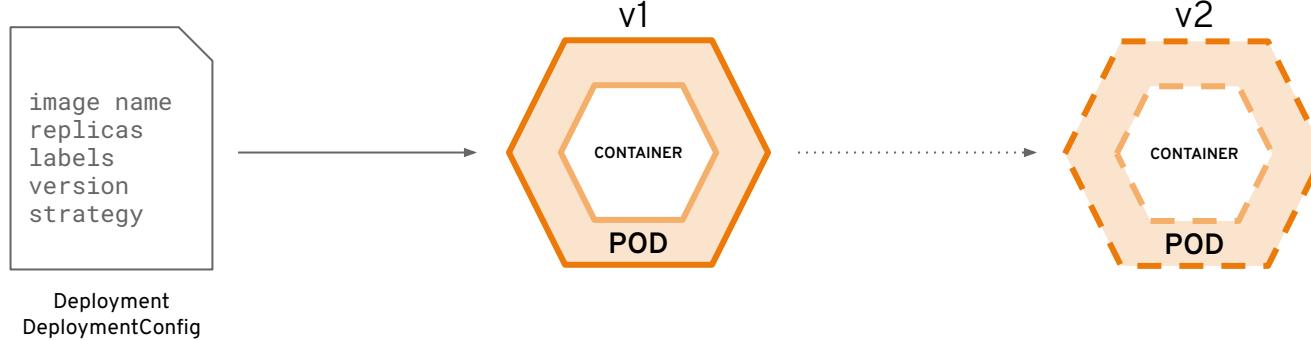


10.15.6.55

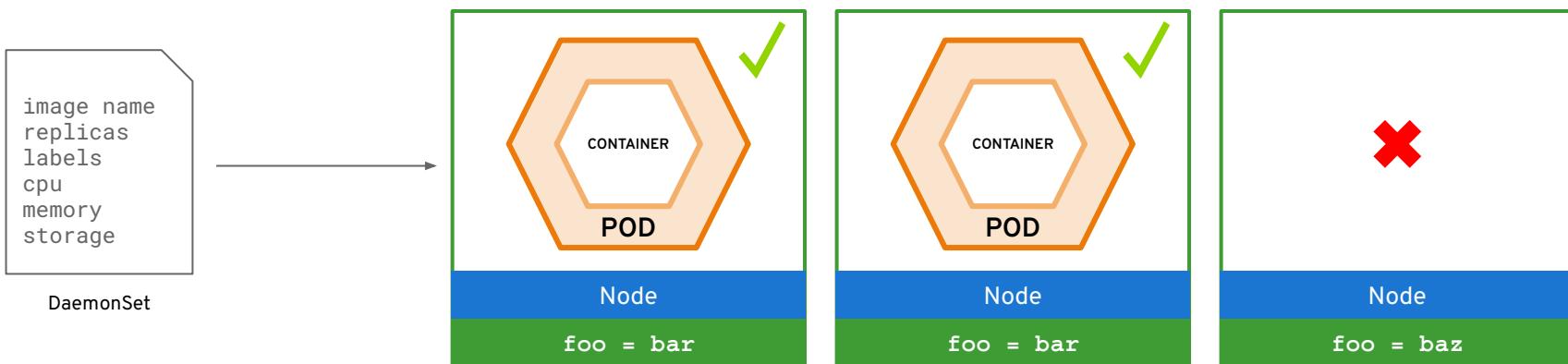
ReplicationControllers & ReplicaSets ensure a specified number of pods are running at any given time



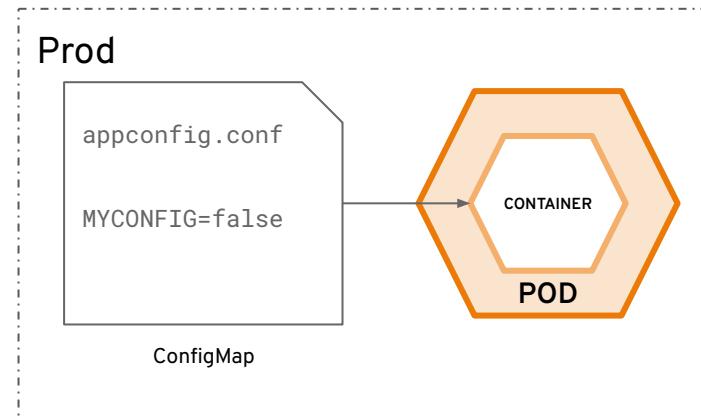
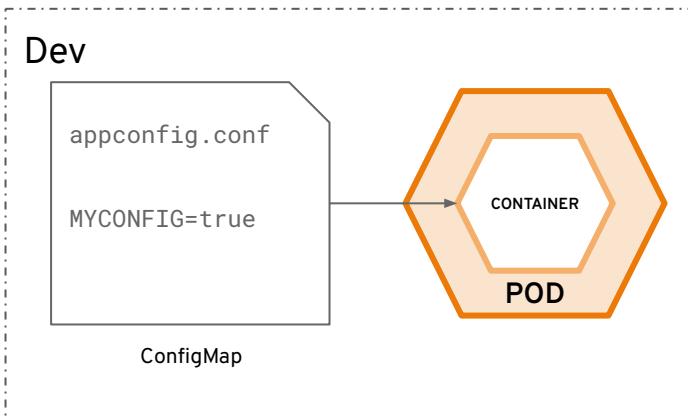
Deployments and DeploymentConfigurations define how to roll out new versions of Pods



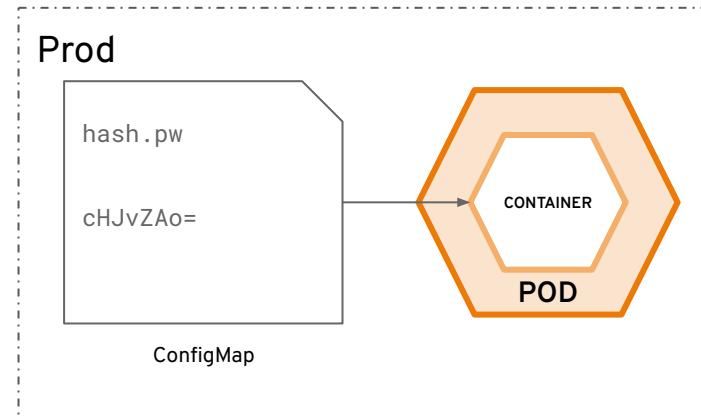
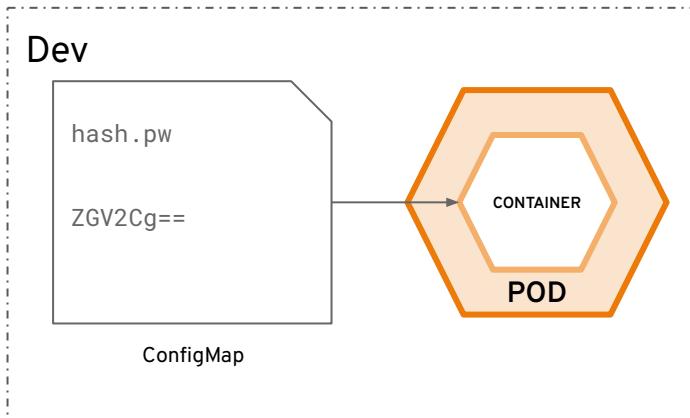
a daemonset ensures that all (or some) nodes run a copy of a pod



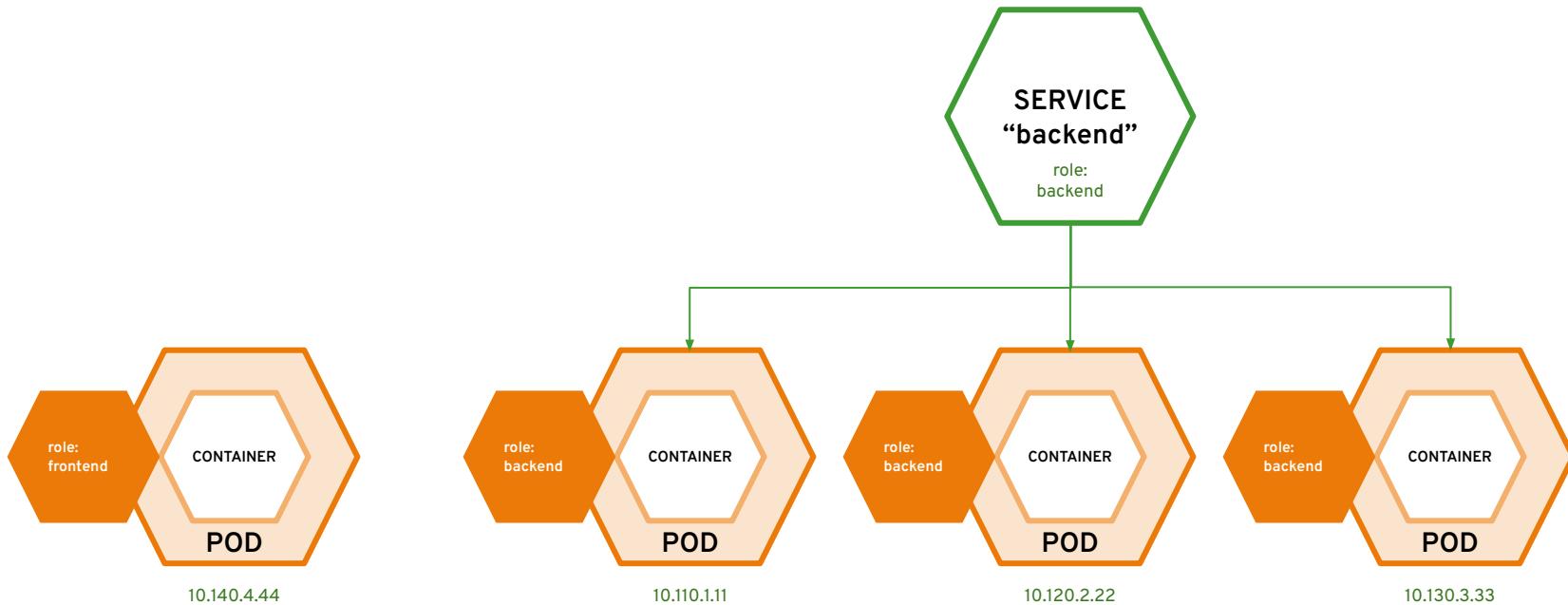
configmaps allow you to decouple configuration artifacts from image content



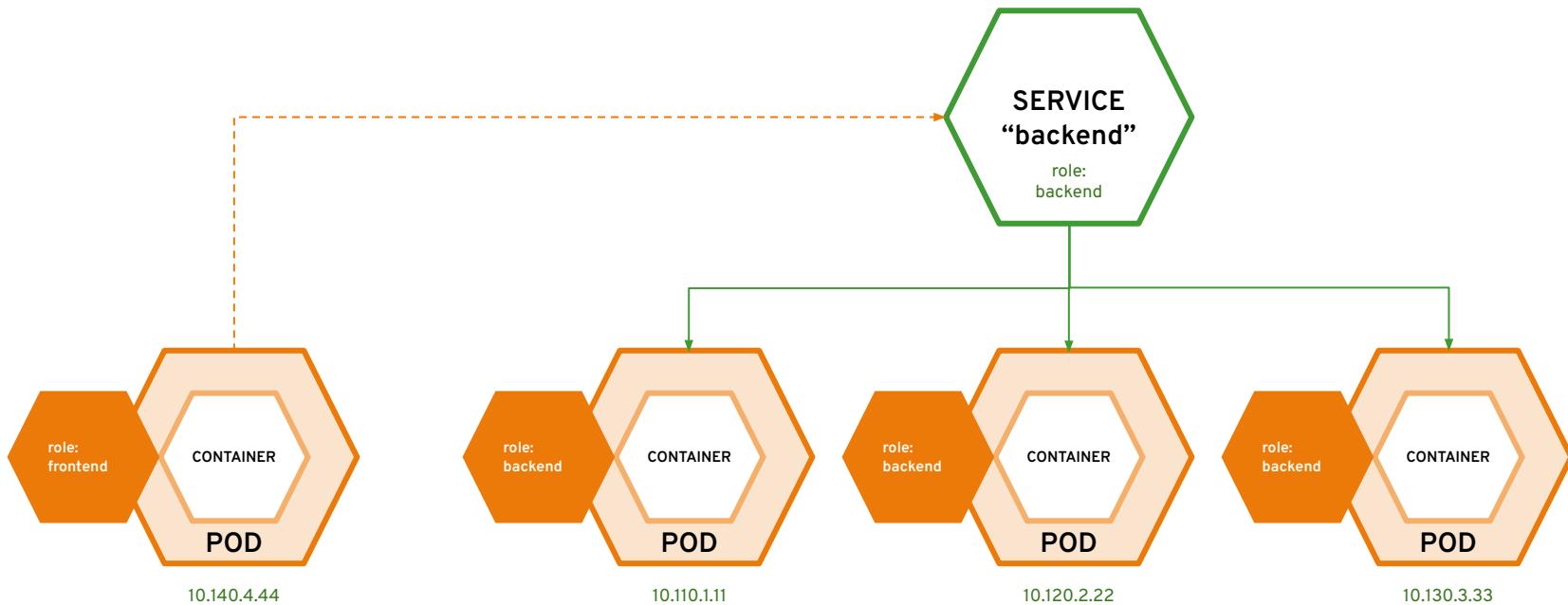
secrets provide a mechanism to hold sensitive information such as passwords



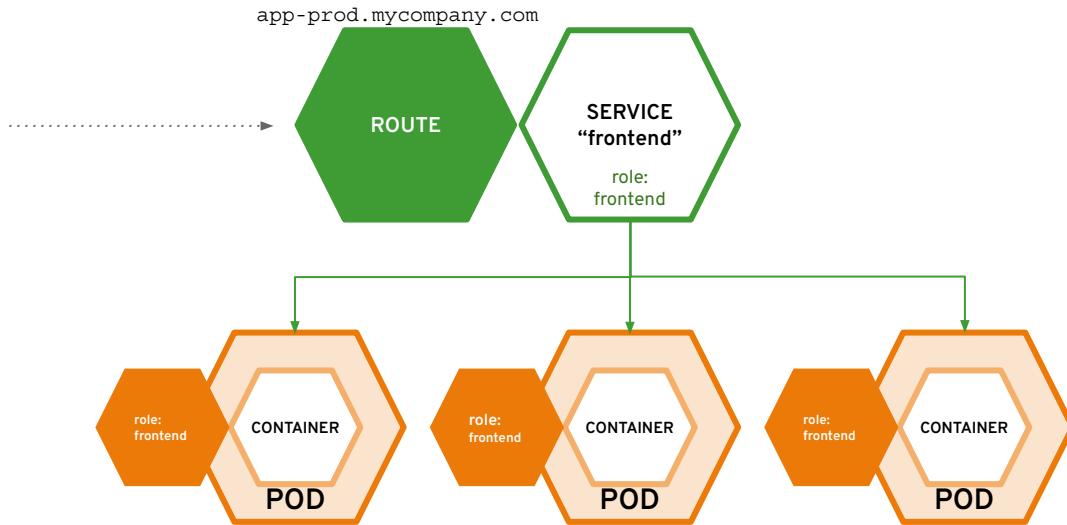
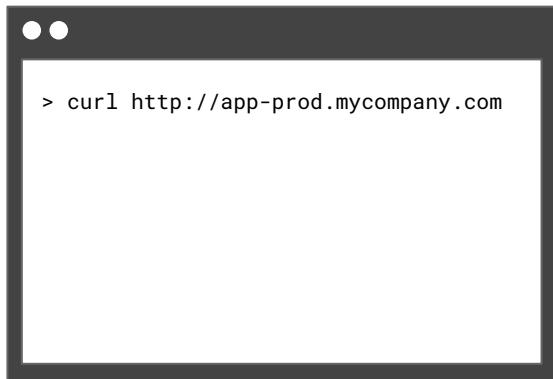
services provide internal load-balancing and service discovery across pods



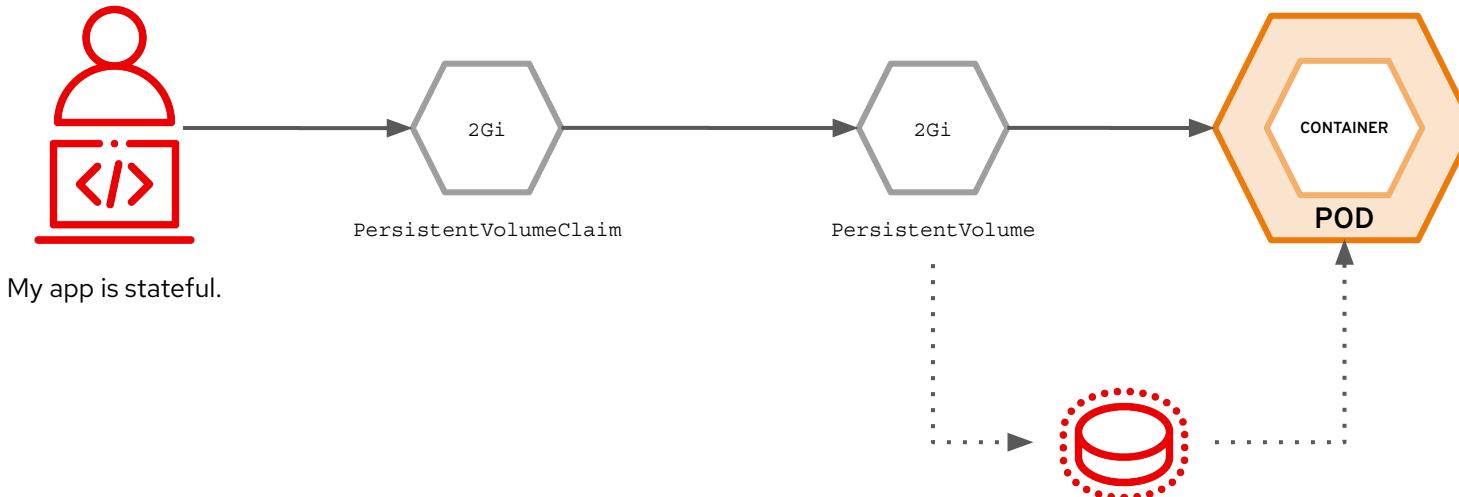
apps can talk to each other via services



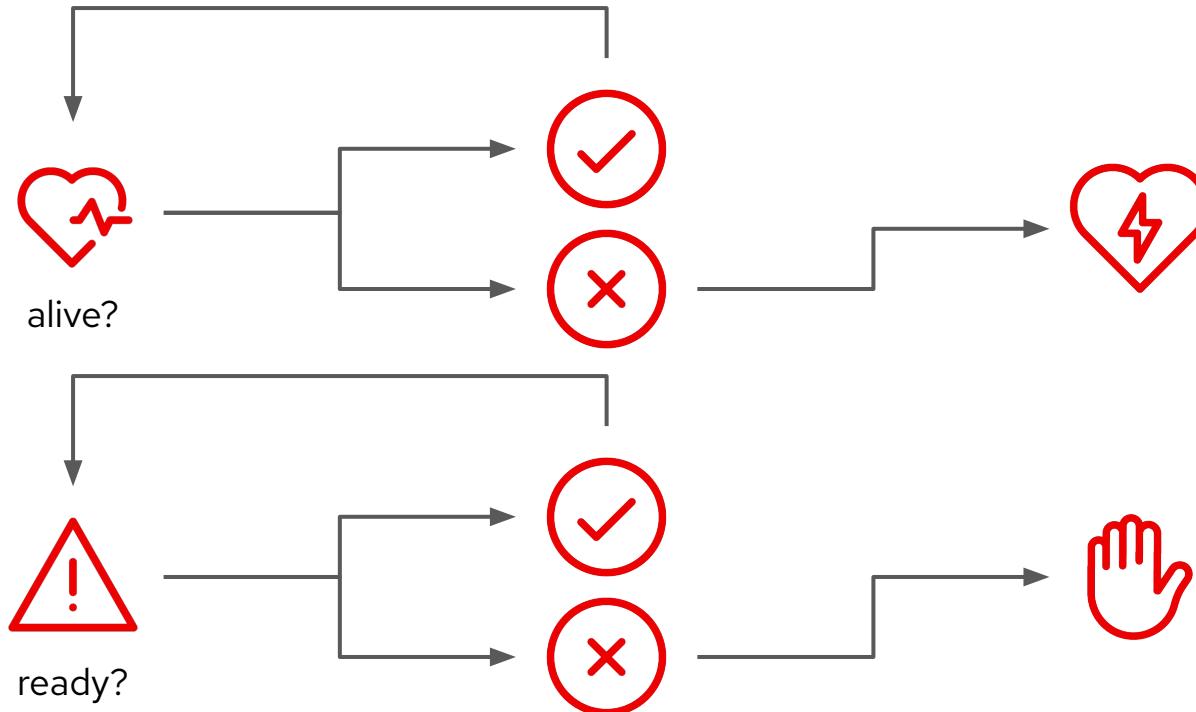
routes make services accessible to clients outside the environment via real-world urls



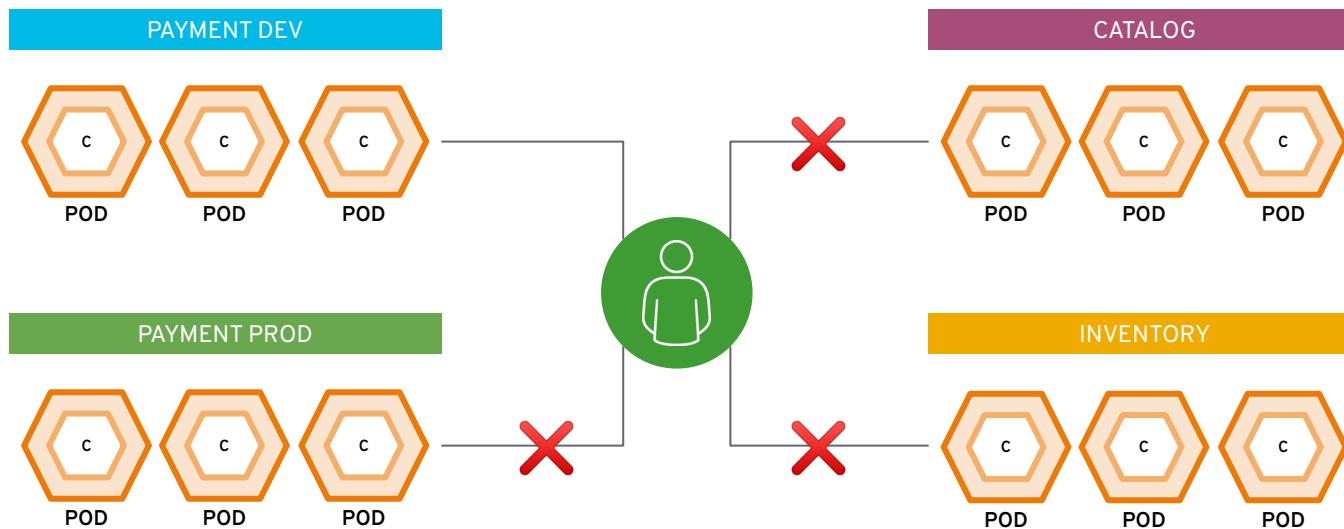
Persistent Volume and Claims



Liveness and Readiness



projects isolate apps across environments,
teams, groups and departments

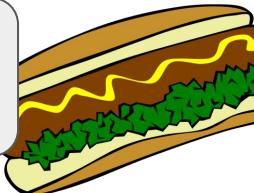


Coffee - Break

Part 1
(9:30 - 11:00)



Part 2
(11:15 - 12:30)



Part 3 (13:30 - 14:30)



OpenShift 4 Architecture

your choice of infrastructure

COMPUTE

NETWORK

STORAGE

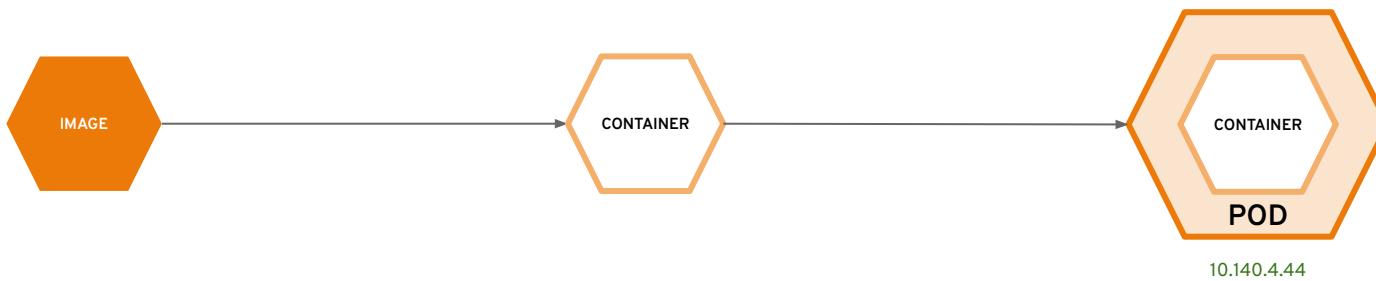
workers run workloads



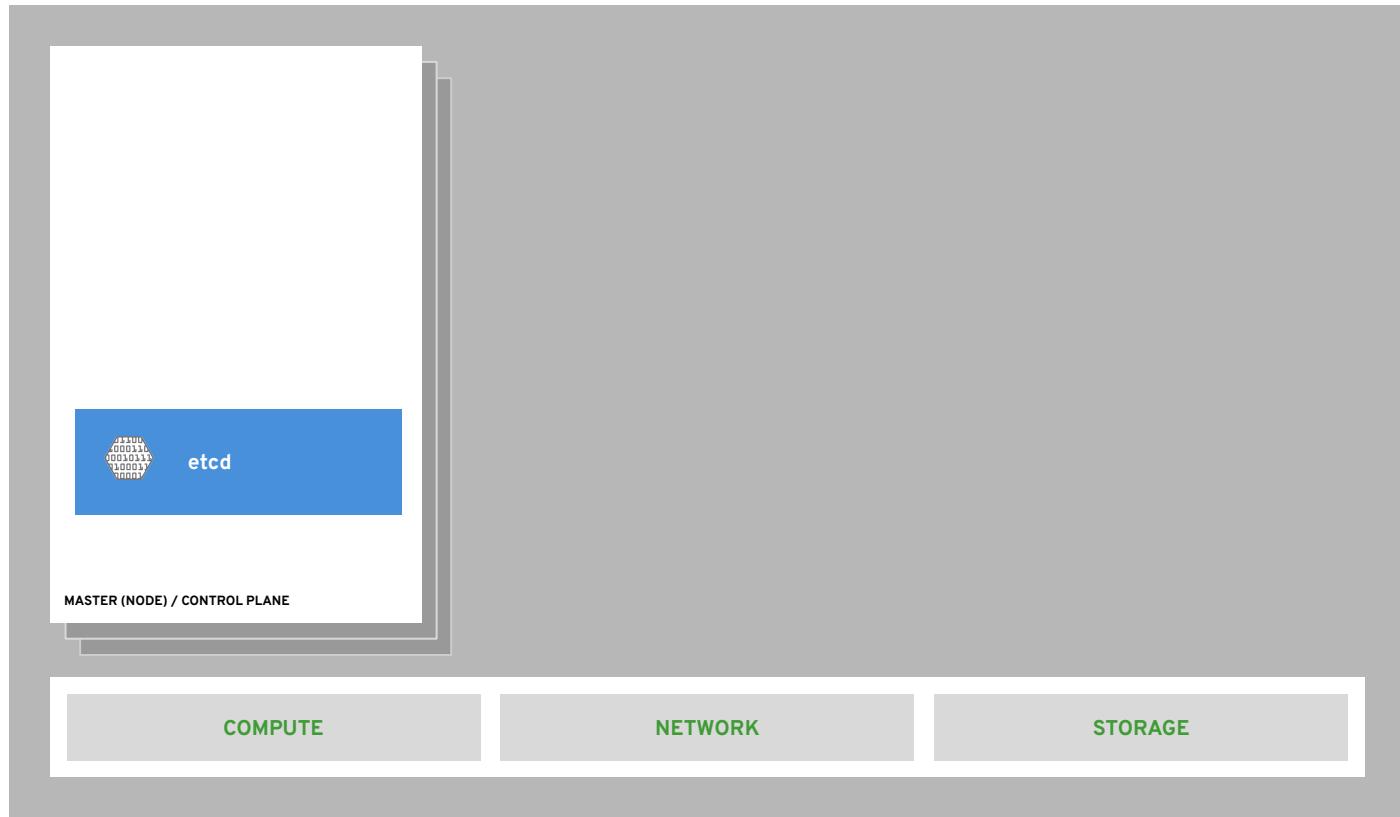
masters are the control plane



everything runs in pods



state of everything



core kubernetes components

Introducing the Declarative Architecture of Kubernetes

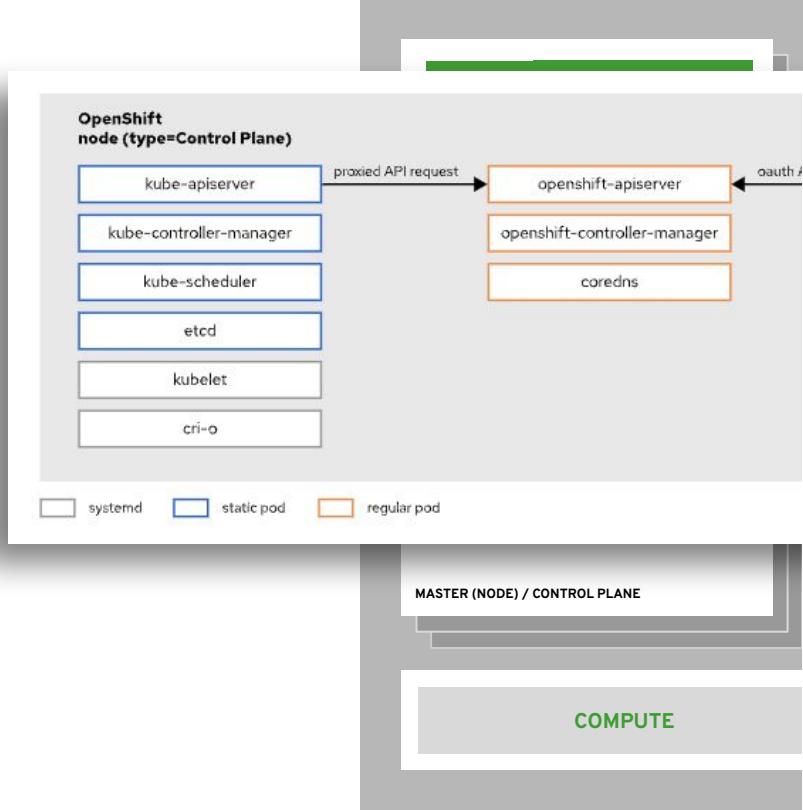
The architecture of OpenShift is based on the declarative nature of Kubernetes. Most system administrators are used to imperative architectures, where you perform actions that indirectly change the state of the system, such as starting and stopping containers on a given server. In a declarative architecture, you change the state of the system and the system updates itself to comply with the new state. For example, with Kubernetes, you define a pod resource that specifies that a certain container should run under specific conditions. Then Kubernetes finds a server (a node) that can run that container under these specific conditions.

Declarative architectures allow for self-optimizing and self-healing systems that are easier to manage than imperative architectures.

Kubernetes defines the state of its cluster, including the set of deployed applications, as a set of resources stored in the etcd database. Kubernetes also runs controllers that monitor these resources and compares them to the current state of the cluster. These controllers take any action necessary to reconcile the state of the cluster with the state of the resources, for example by finding a node with sufficient CPU capacity to start a new container from a new pod resource.

Kubernetes provides a REST API to manage these resources. All actions that an OpenShift user takes, either using the command-line interface or the web console, are performed by invoking this REST API.

core OpenShift components



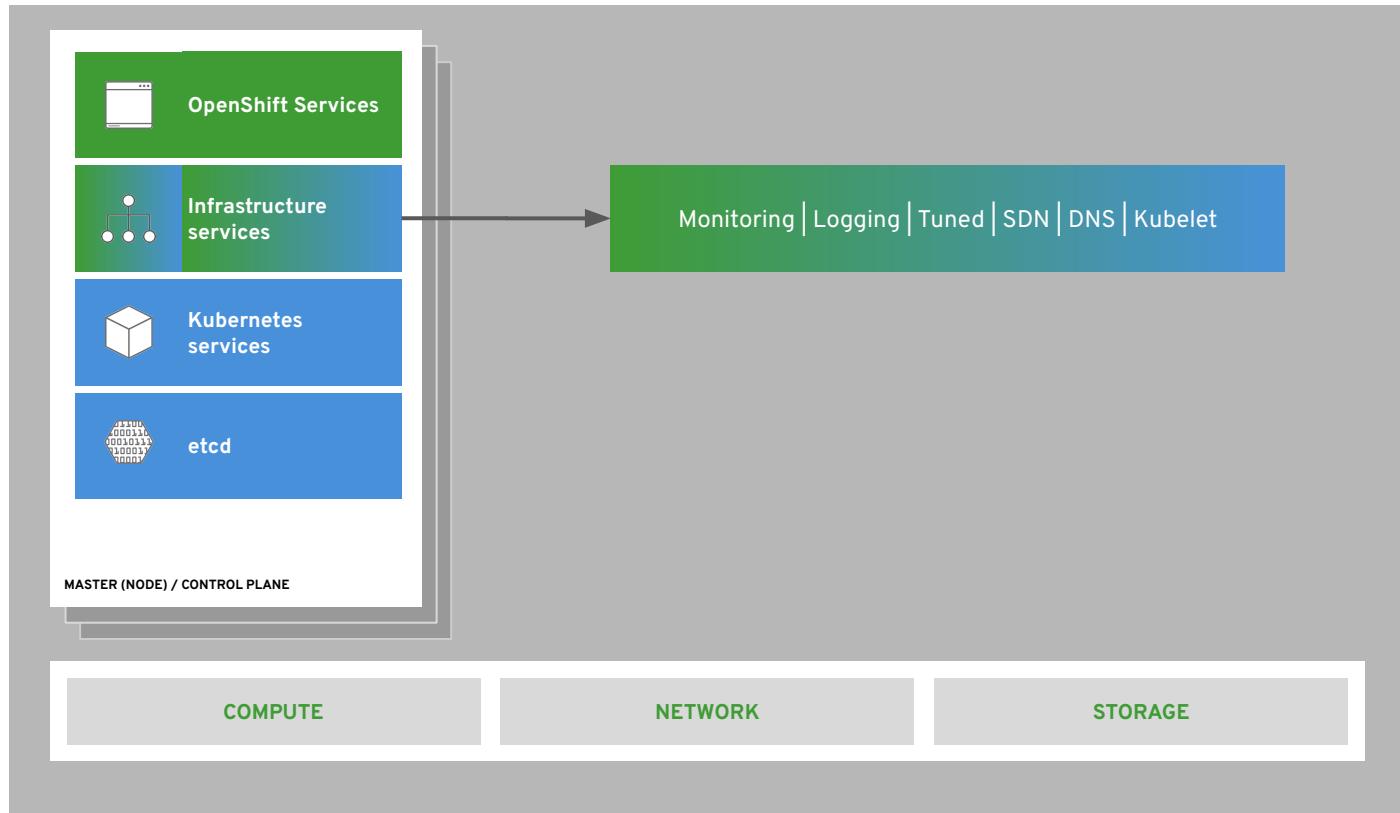
Describing OpenShift Extensions

A lot of functionality from Kubernetes depends on external components, such as ingress controllers, storage plug-ins, network plug-ins, and authentication plug-ins. Similar to Linux distributions, there are many ways to build a Kubernetes distribution by picking and choosing different components.

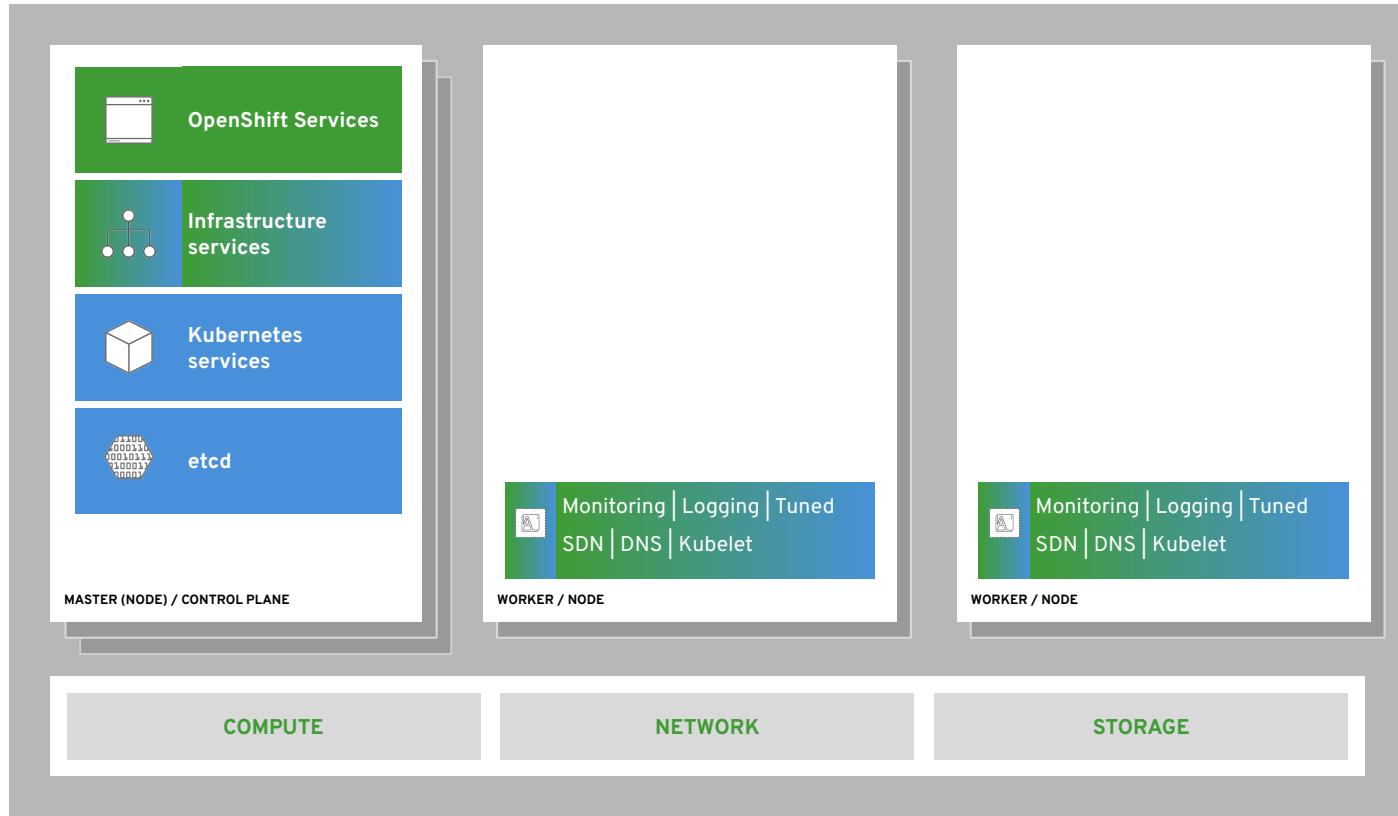
A lot of functionality from Kubernetes also depends on extension APIs, such as access control and network isolation.

OpenShift is a Kubernetes distribution that provides many of these components already integrated and configured, and managed by operators. OpenShift also provides preinstalled applications, such as a container image registry and a web console, managed by operators.

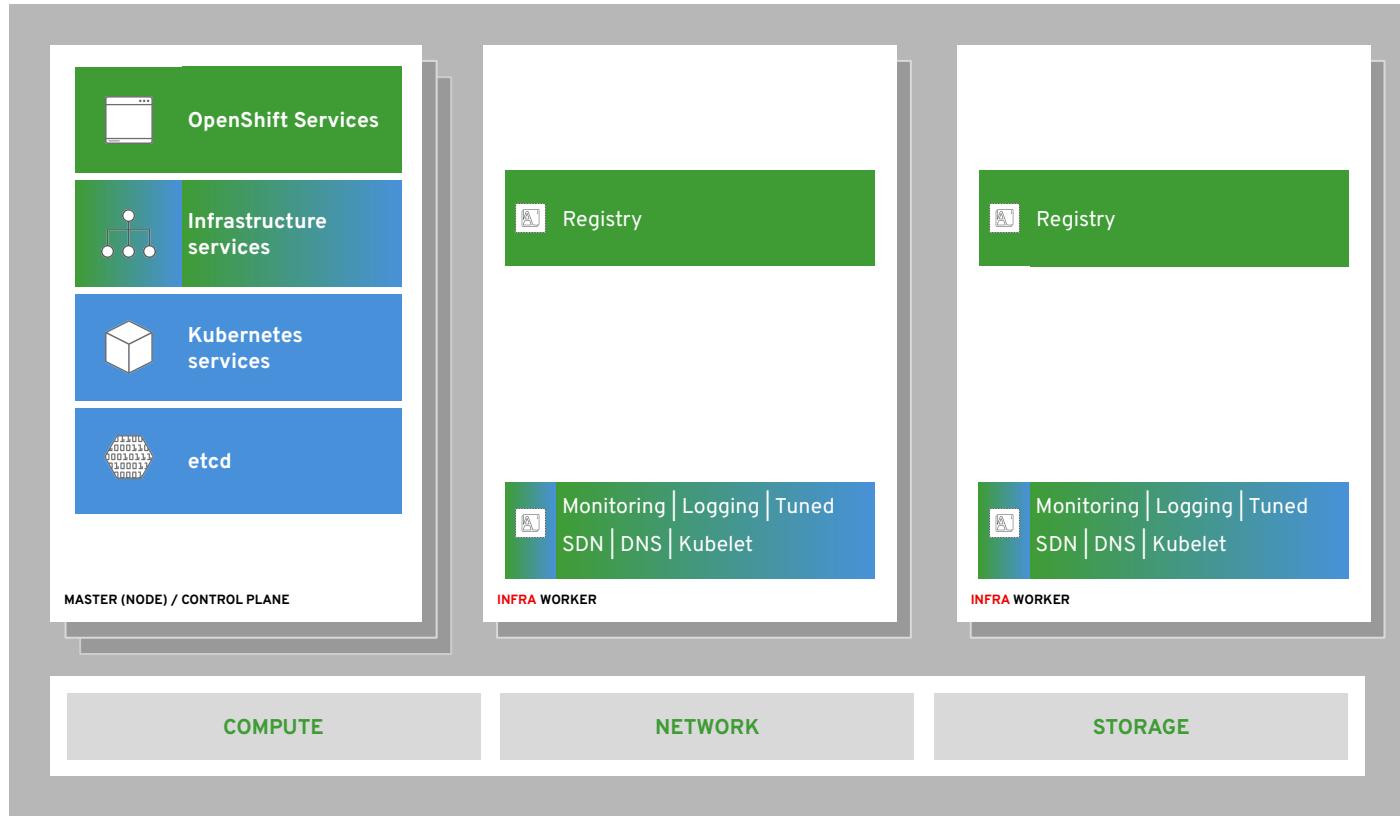
internal and support infrastructure services



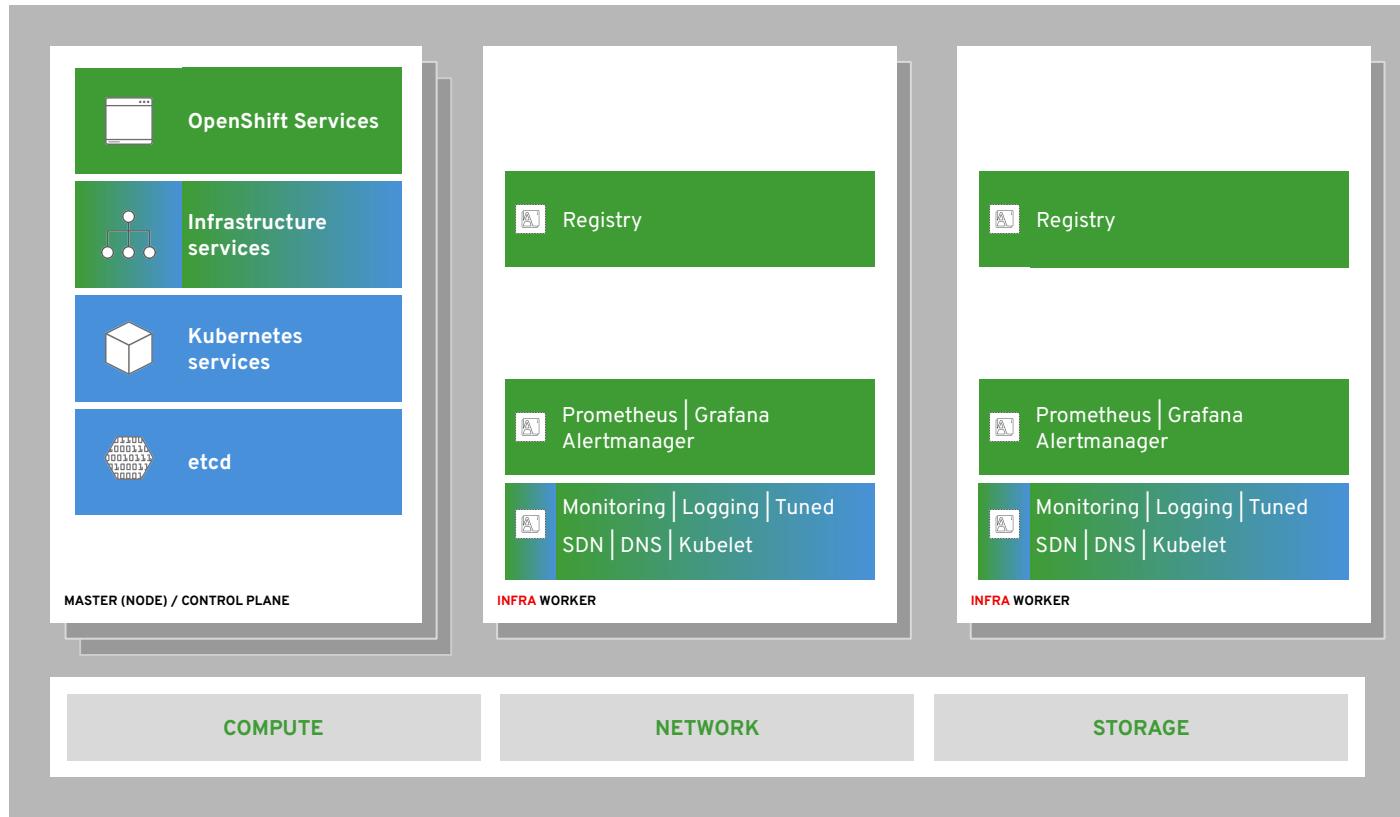
run on all hosts



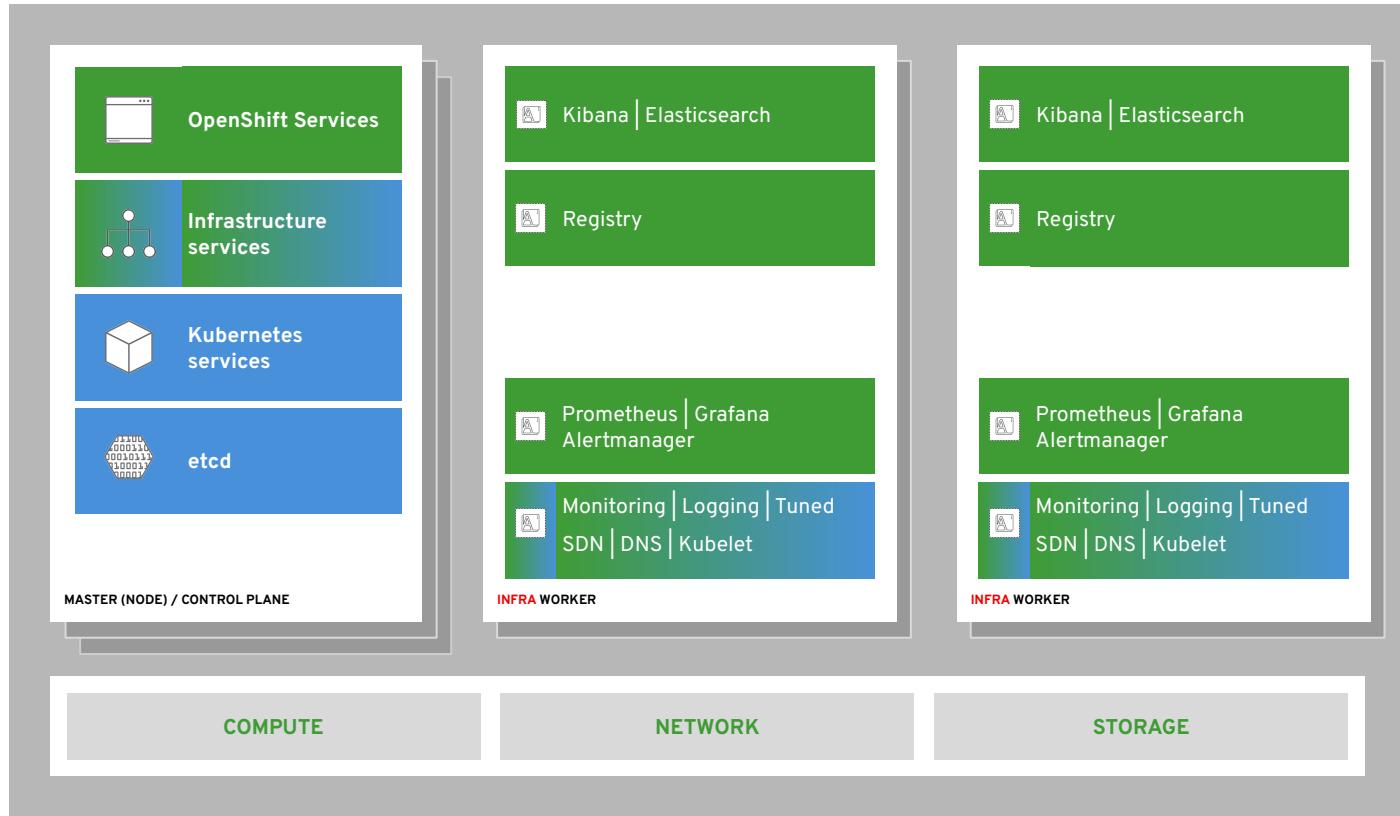
integrated image registry



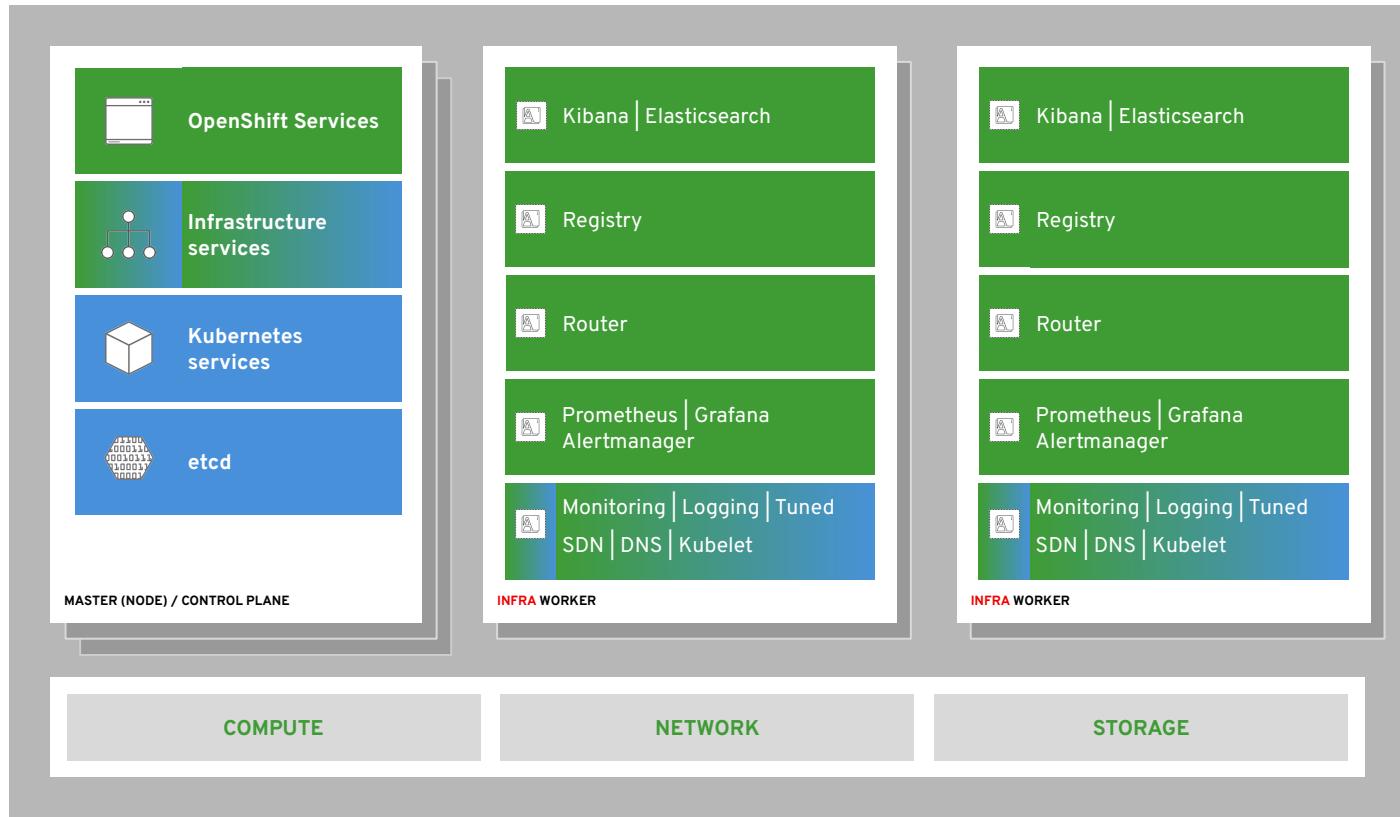
cluster monitoring



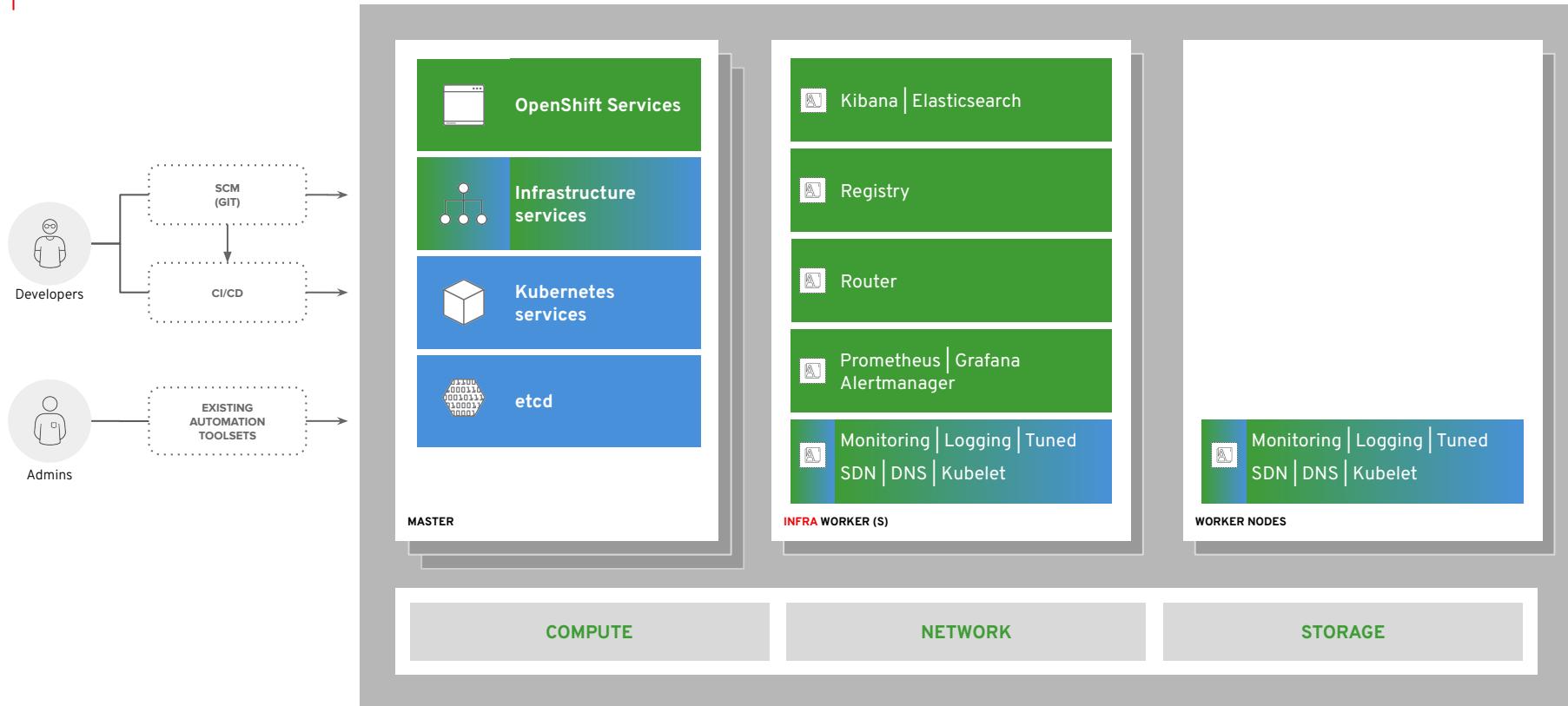
log aggregation



integrated routing



dev and ops via web, cli, API, and IDE



Quiz: Describing the Architecture of OpenShift

1. OpenShift is based on which of the following container orchestration technologies?

- A Docker Swarm
- B Rancher
- C Kubernetes
- D Mesosphere Marathon
- E CoreOS Fleet

CHECK **RESET** **SHOW SOLUTION**

2. Which two of the following statements are true of OpenShift Container Platform? (Choose two.)

- A OpenShift provides an OAuth server that authenticates calls to its REST API.
- B OpenShift requires the CRI-O container engine.
- C Kubernetes follows a declarative architecture, but OpenShift follows a more traditional imperative architecture.
- D OpenShift extension APIs run as system services.

CHECK **RESET** **SHOW SOLUTION**

3. Which of the following servers runs Kubernetes API components?

- A Compute nodes
- B Nodes
- C Control plane nodes

CHECK **RESET** **SHOW SOLUTION**

4. Which of the following components does OpenShift add to upstream Kubernetes?

- A The etcd database
- B A container engine
- C A registry server
- D A scheduler
- E The Kubelet

CHECK **RESET** **SHOW SOLUTION**

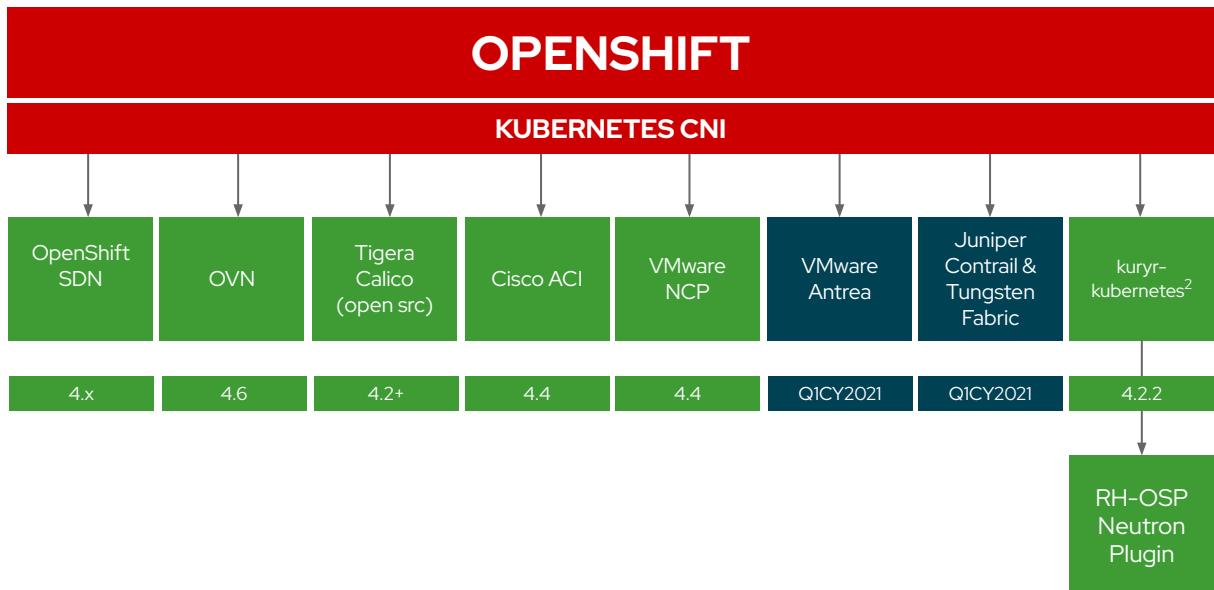
Networking

A pluggable model for network interface controls in kubernetes

OpenShift Networking Plug-ins

3rd-party Kubernetes CNI plug-in certification primarily consists of:

1. Formalizing the partnership
2. Certifying the container(s)
3. Certifying the Operator
4. Successfully passing the same Kubernetes networking conformance tests that OpenShift uses to validate its own SDN



OpenShift SDN

An Open Virtual Network OVN
Software Defined Network for
kubernetes

OpenShift implements a software-defined network (SDN) to manage the network infrastructure of the cluster and user applications. Software-defined networking is a networking model that allows you to manage network services through the abstraction of several networking layers. It decouples the software that handles the traffic, called the *control plane*, and the underlying mechanisms that route the traffic, called the *data plane*. Among the many features of SDN, open standards enable vendors to propose their solutions, centralized management, dynamic routing, and tenant isolation.

OpenShift SDN high-level architecture

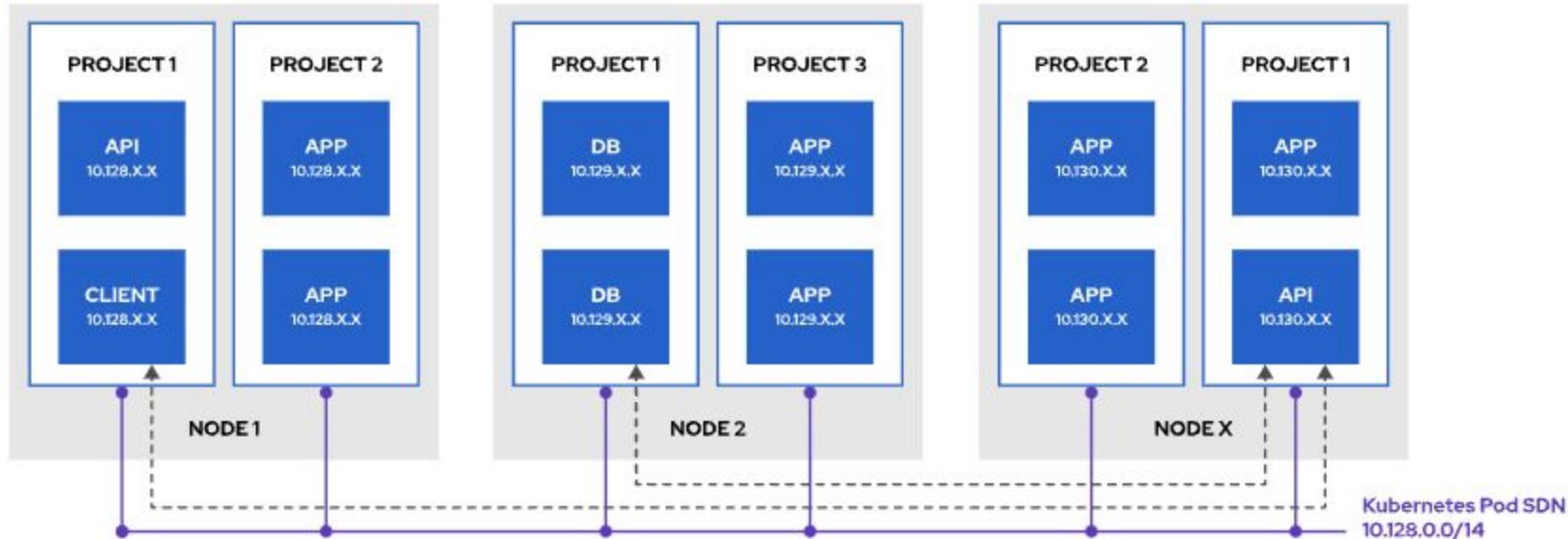


Figure 5.1: Kubernetes basic networking

Using Services for Accessing Pods

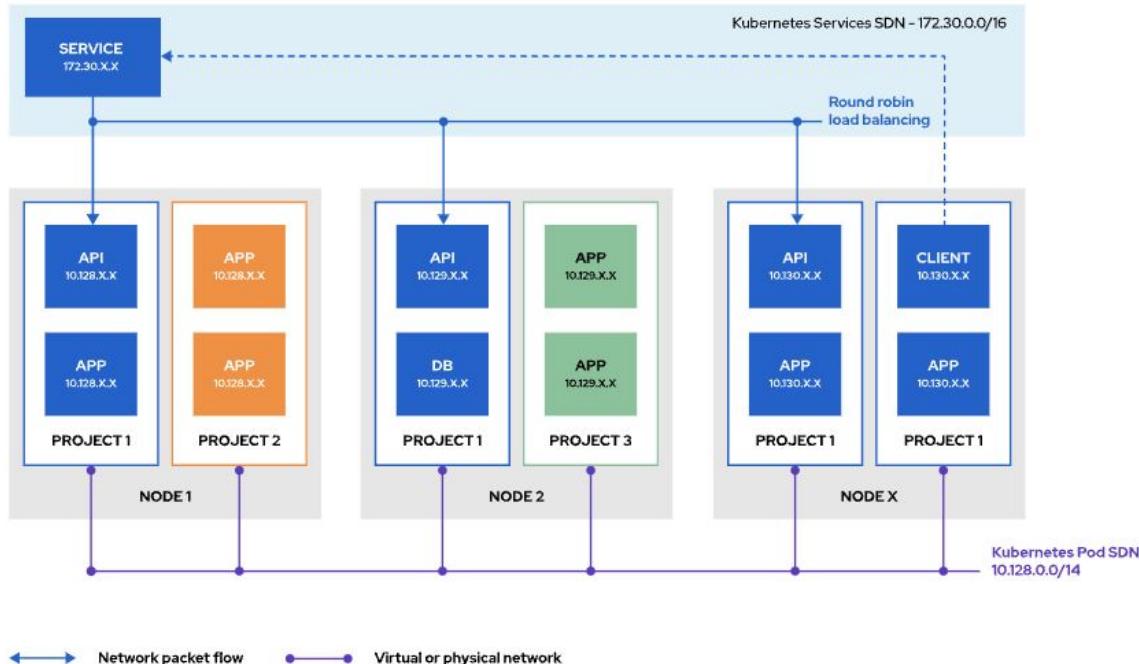
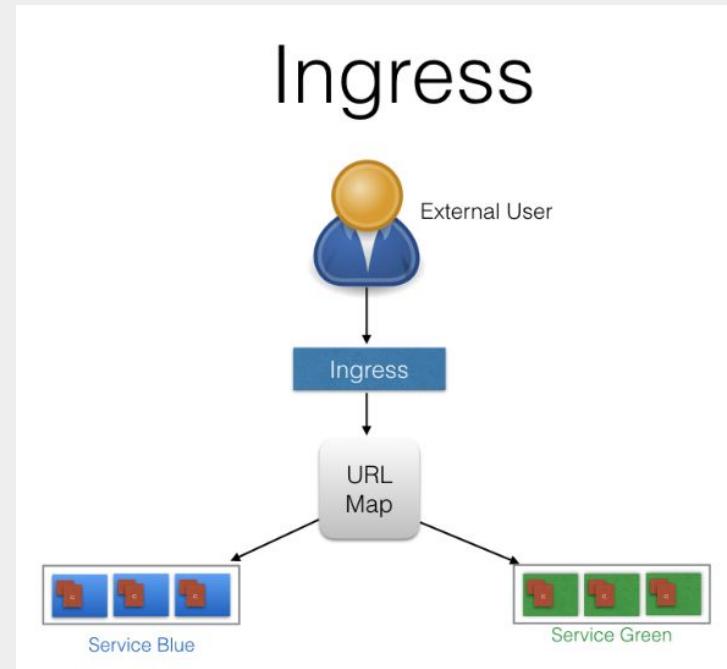


Figure 5.2: Using services for accessing applications

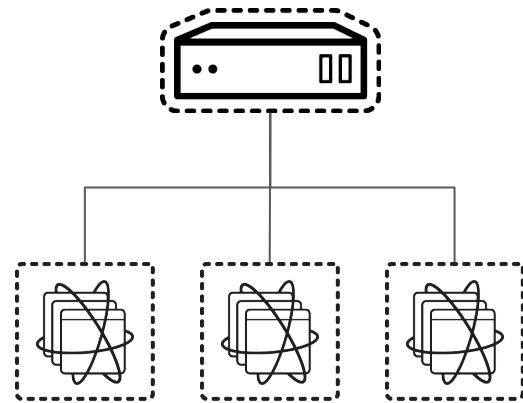
routes and ingress

How traffic enters the cluster

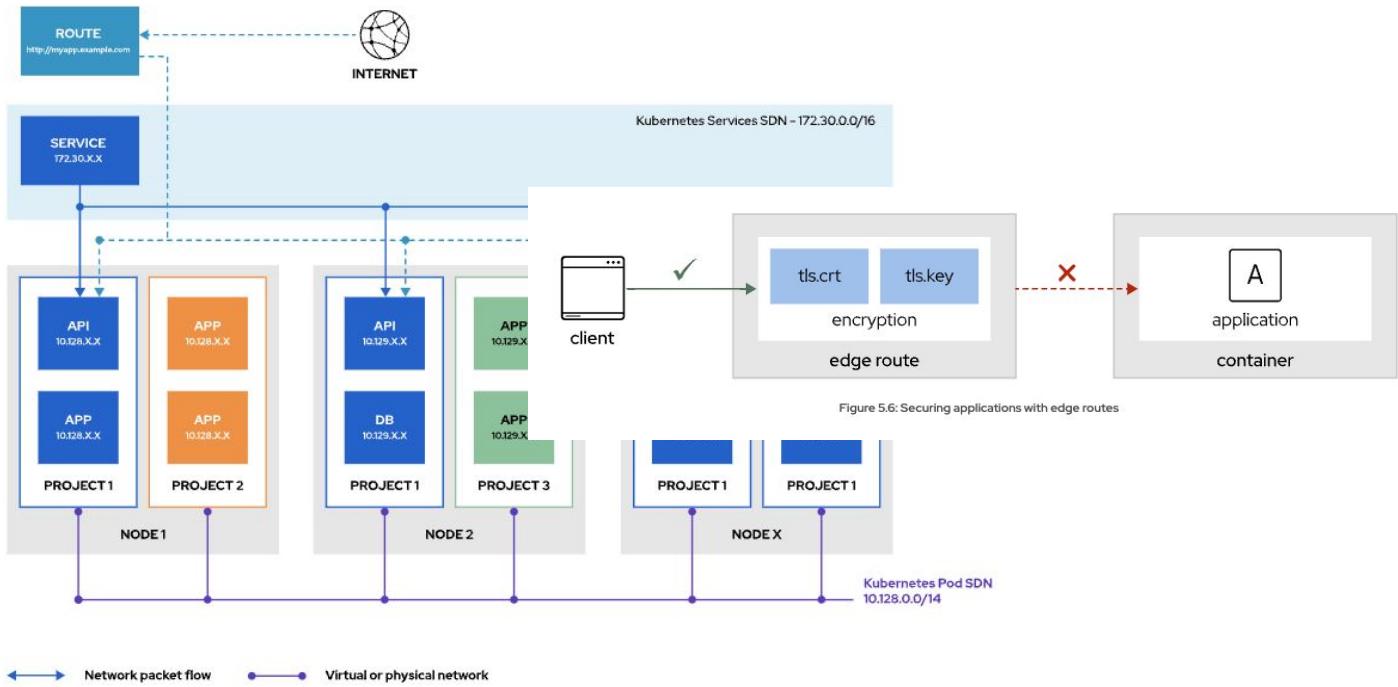


Routing and Load Balancing

- Pluggable routing architecture
 - HAProxy Router
 - F5 Router
- Multiple-routers with traffic sharding
- Router supported protocols
 - HTTP/HTTPS
 - WebSockets
 - TLS with SNI
- Non-standard ports via cloud load-balancers, external IP, and NodePort



Guided Exercise: Exposing Applications for External Access

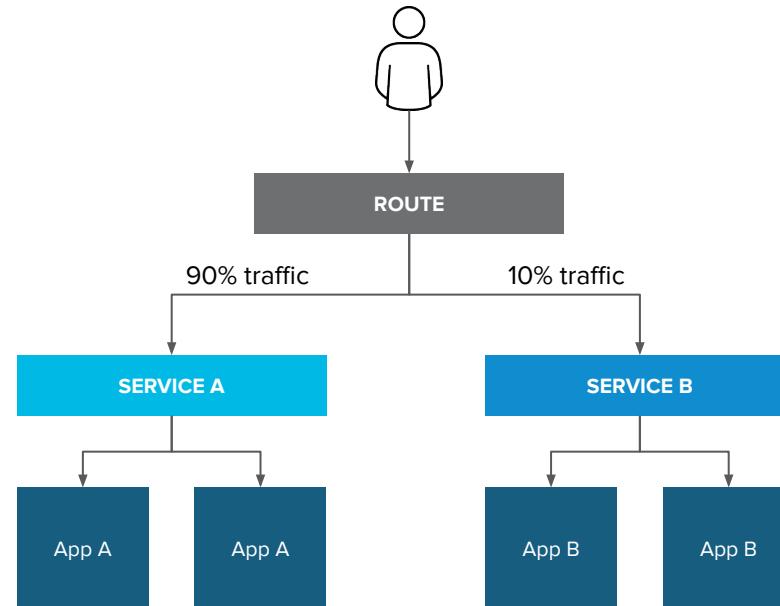


Routes vs Ingress

Feature	Ingress	Route
Standard Kubernetes object	X	
External access to services	X	X
Persistent (sticky) sessions	X	X
Load-balancing strategies (e.g. round robin)	X	X
Rate-limit and throttling	X	X
IP whitelisting	X	X
TLS edge termination	X	X
TLS re-encryption	X	X
TLS passthrough	X	X
Multiple weighted backends (split traffic)		X
Generated pattern-based hostnames		X
Wildcard domains		X

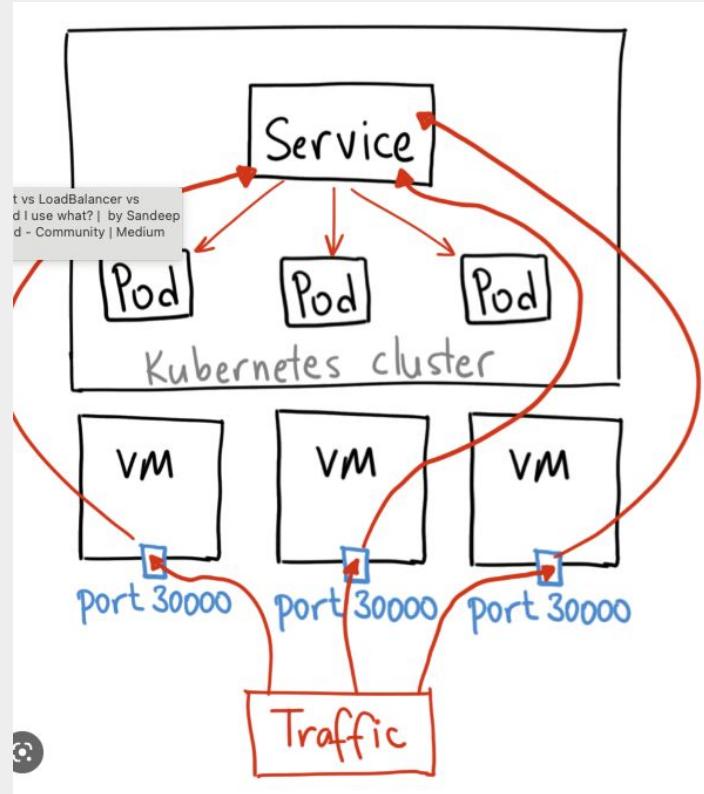
Router-based deployment methodologies

Split Traffic Between
Multiple Services For A/B
Testing, Blue/Green and
Canary Deployments



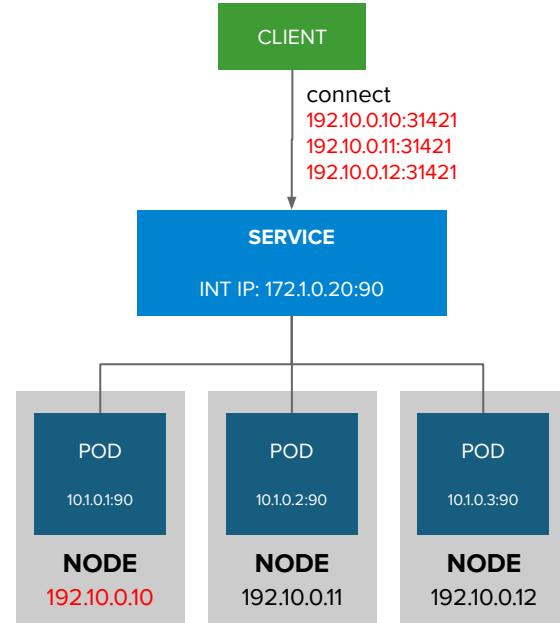
Alternative methods for ingress

Different ways that traffic can enter the cluster without the router



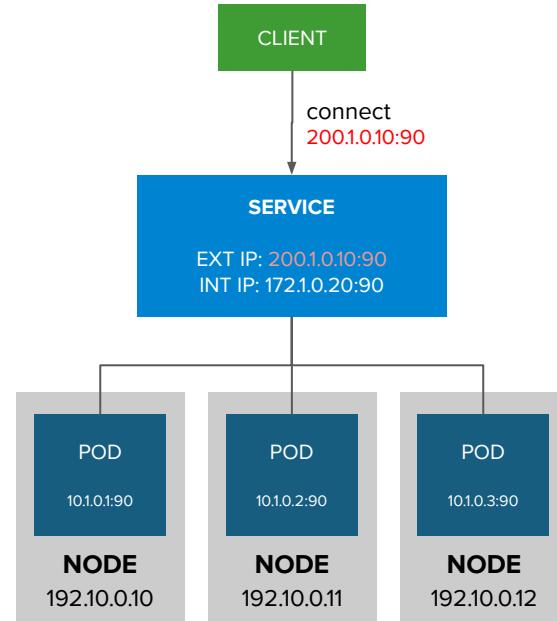
Entering the cluster on a random port with service nodeports

- NodePort binds a service to a unique port on all the nodes
- Traffic received on any node redirects to a node with the running service
- Ports in 30K-60K range which usually differs from the service
- Firewall rules must allow traffic to all nodes on the specific port



External traffic to a service on any port with external IP

- Access a service with an external IP on any TCP/UDP port, such as
 - Databases
 - Message Brokers
- Automatic IP allocation from a predefined pool using Ingress IP Self-Service
- IP failover pods provide high availability for the IP pool (fully supported in 4.8)



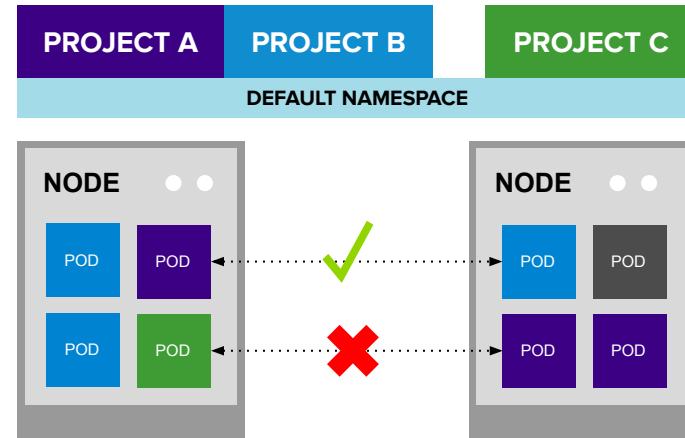
OpenShift SDN “flavors”

OPEN NETWORK (Default)

- All pods can communicate with each other across projects

MULTI-TENANT NETWORK

- Project-level network isolation
- Multicast support
- Egress network policies



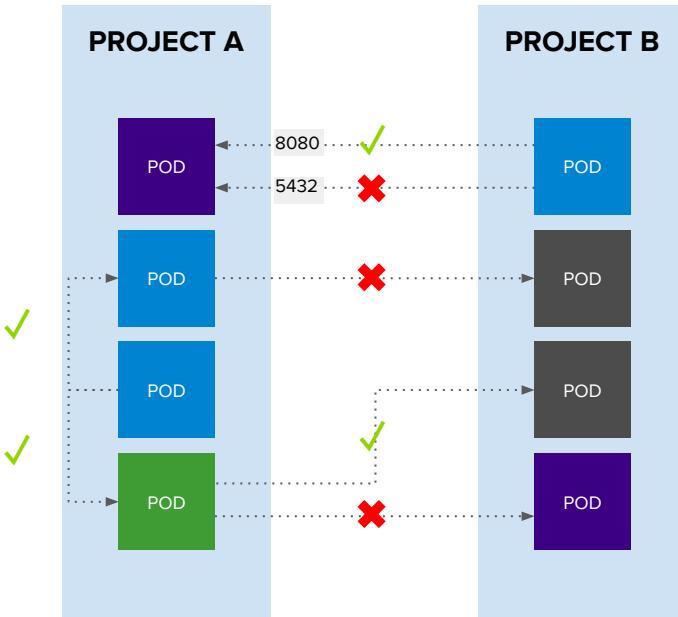
Multi-Tenant Network

Managing Network Policies in OpenShift

Network policies allow you to configure isolation policies for individual pods. Network policies do not require administrative privileges, giving developers more control over the applications in their projects. You can use network policies to create logical zones in the SDN that map to your organization network zones. The benefit of this approach is that the location of running pods becomes irrelevant because network policies allow you to segregate traffic regardless of where it originates.

The screenshot shows the Red Hat OpenShift Container Platform web interface. The left sidebar menu is visible, with 'Administrator' selected. Under the 'Networking' section, 'Network Policies' is highlighted. The main content area is titled 'Create Network Policy' and shows a form for defining a new policy. The 'Policy name' field contains 'my-policy'. The 'Policy namespace' field contains 'my-project'. A note states: 'The namespace containing the pods that the network policy applies.' Below this is a 'PodSelector' section with a radio button for 'Add pod selector' and a 'Policy type' section with a 'Deny' icon. Under 'Select default ingress and egress deny rules', there are two checkboxes: 'Deny all ingress traffic' (unchecked) and 'Deny all egress traffic' (unchecked). Below this are sections for 'Ingress' and 'Egress', each with a 'Remove all' button and an 'Add [type] rule' button. The 'Ingress' section has a note: 'List of ingress rules to be applied to the selected pods. Traffic is allowed from a pod if the traffic matches at least one rule.' The 'Egress' section has a note: 'List of egress rules to be applied to the selected pods. Traffic is allowed to a pod if the traffic matches at least one rule.' At the bottom are 'Create' and 'Cancel' buttons. On the right side, there are tabs for 'Network Policy', 'Schema', and 'Samples', with 'Network Policy' currently selected. A detailed description of the 'NetworkPolicy' schema is provided on the right, including fields like 'apiVersion', 'kind', and 'status'.

NetworkPolicy



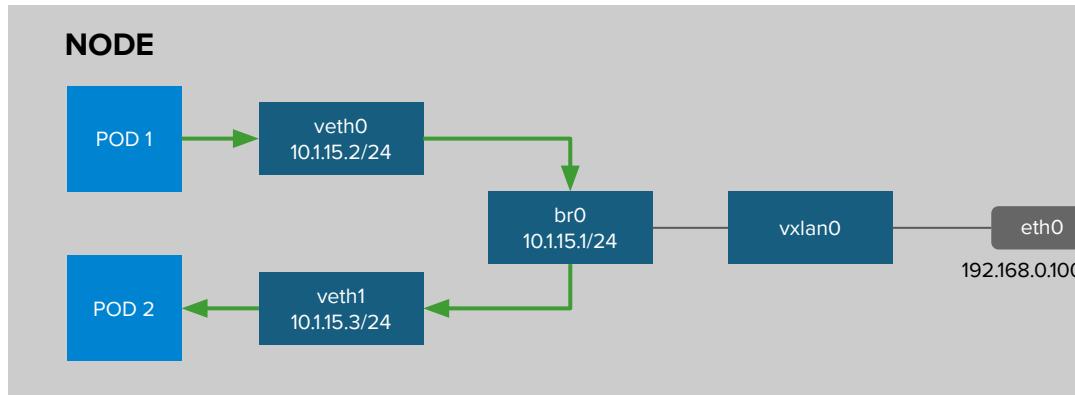
Example Policies

- Allow all traffic inside the project
- Allow traffic from green to gray
- Allow traffic to purple on 8080

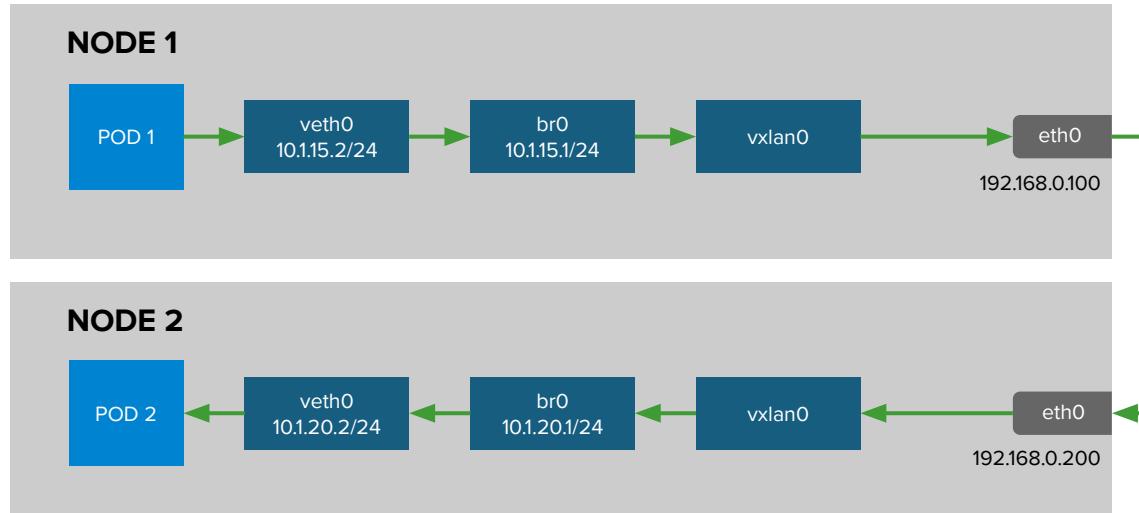
```
apiVersion: extensions/v1beta1
kind: NetworkPolicy
metadata:
  name: allow-to-purple-on-8080
spec:
  podSelector:
    matchLabels:
      color: purple
  ingress:
  - ports:
    - protocol: tcp
      port: 8080
```

OpenShift SDN packet flows

container-container on same host

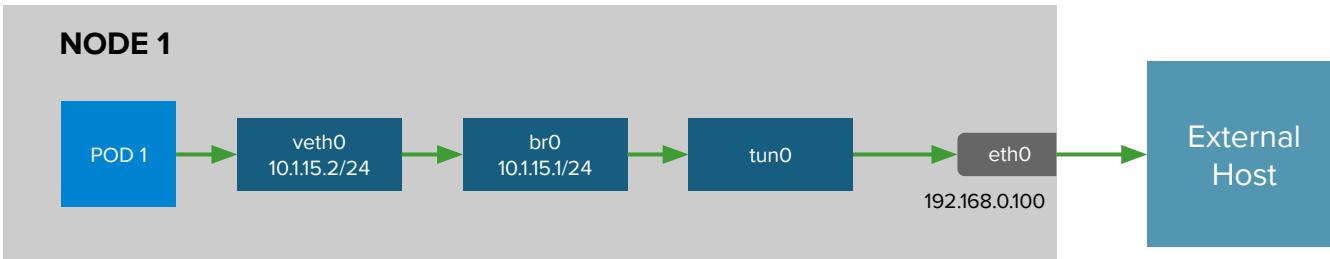


OpenShift SDN packet flows container-container across hosts



OpenShift SDN packet flows

container leaving the host



Cluster DNS

An automated system for
providing hostname resolution
within kubernetes



CoreDNS

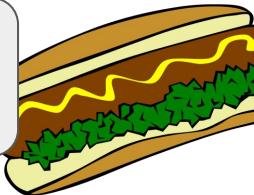
- Built-in internal DNS to reach services by a (fully qualified) hostname
- Split DNS is used with CoreDNS
 - CoreDNS answers DNS queries for internal/cluster services
 - Other defined “upstream” name servers serve the rest of the queries

Lunch Break

Part 1
(9:30 - 11:00)



Part 2
(11:15 - 12:30)

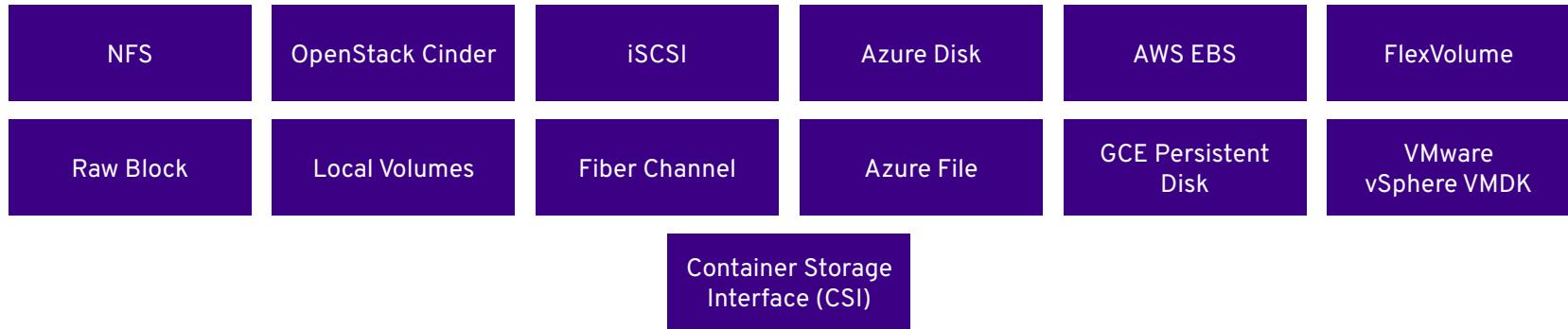


Part 3 (13:30 - 14:30)

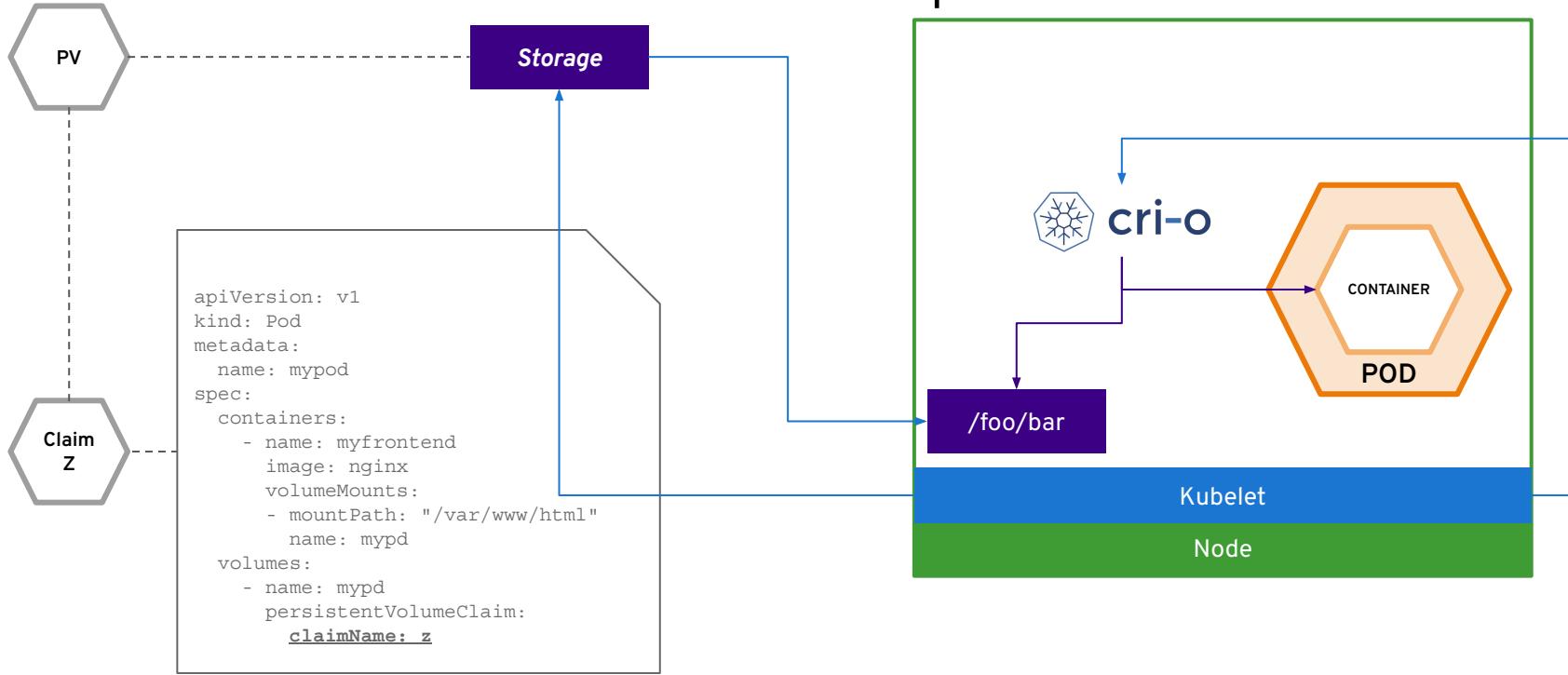


Persistent Storage

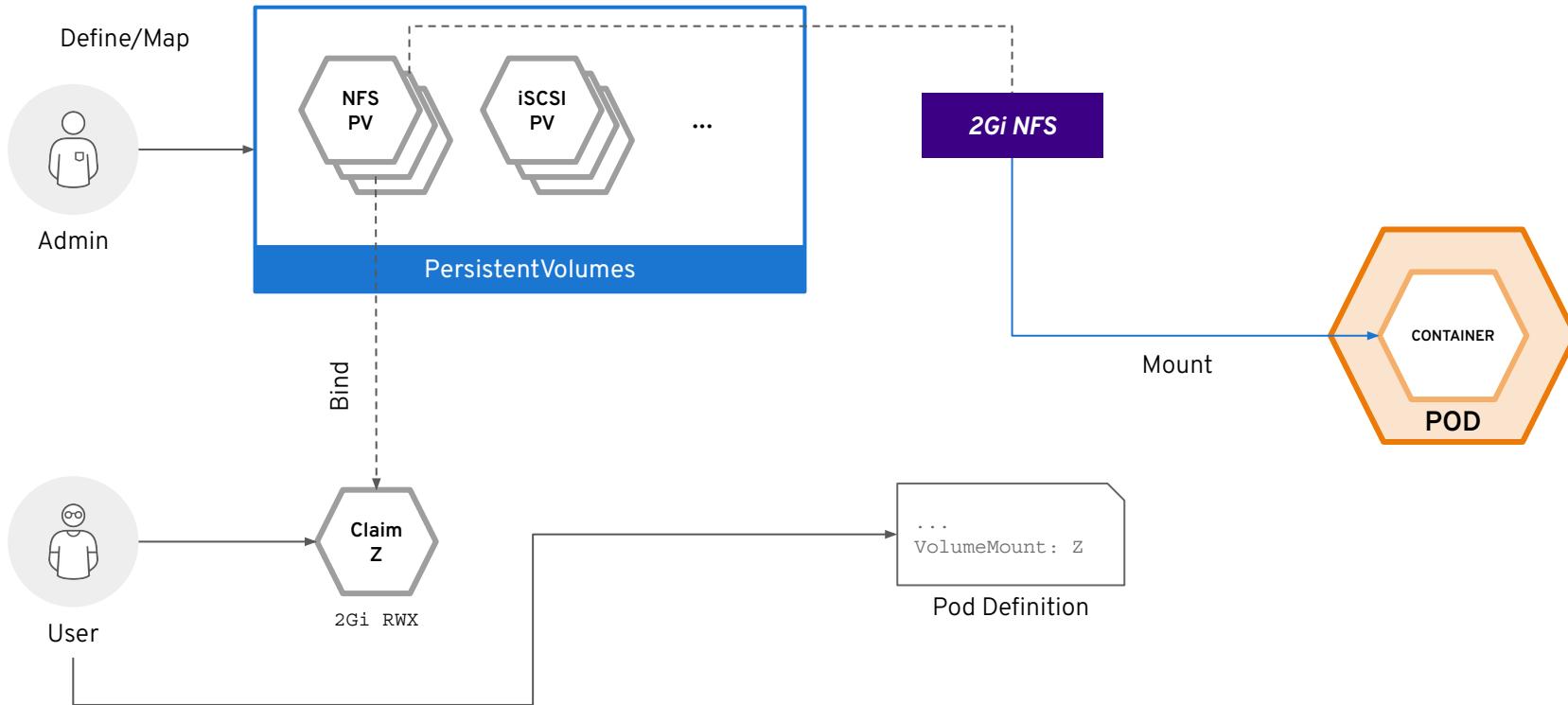
A broad spectrum of static and dynamic storage endpoints



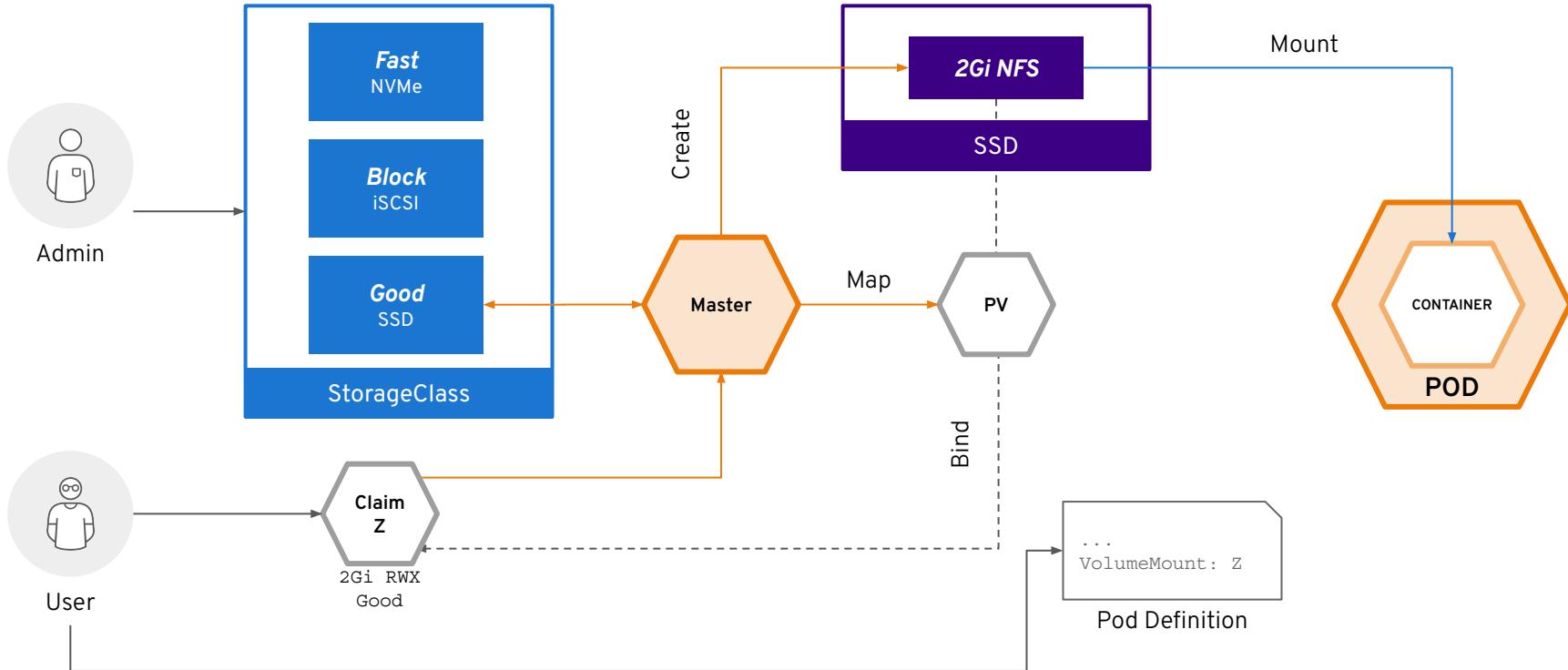
PV Consumption

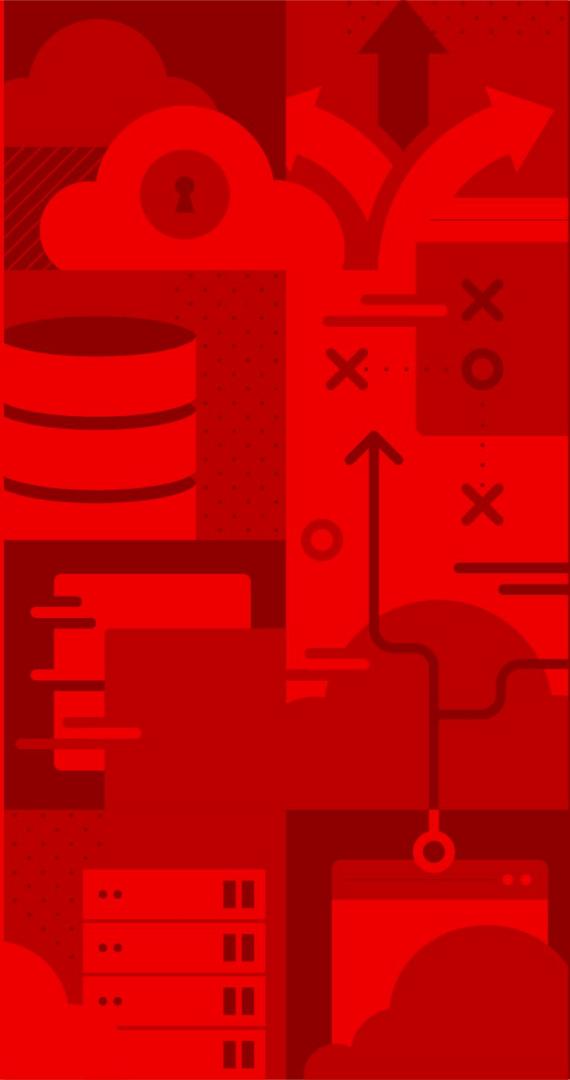


Static Storage Provisioning



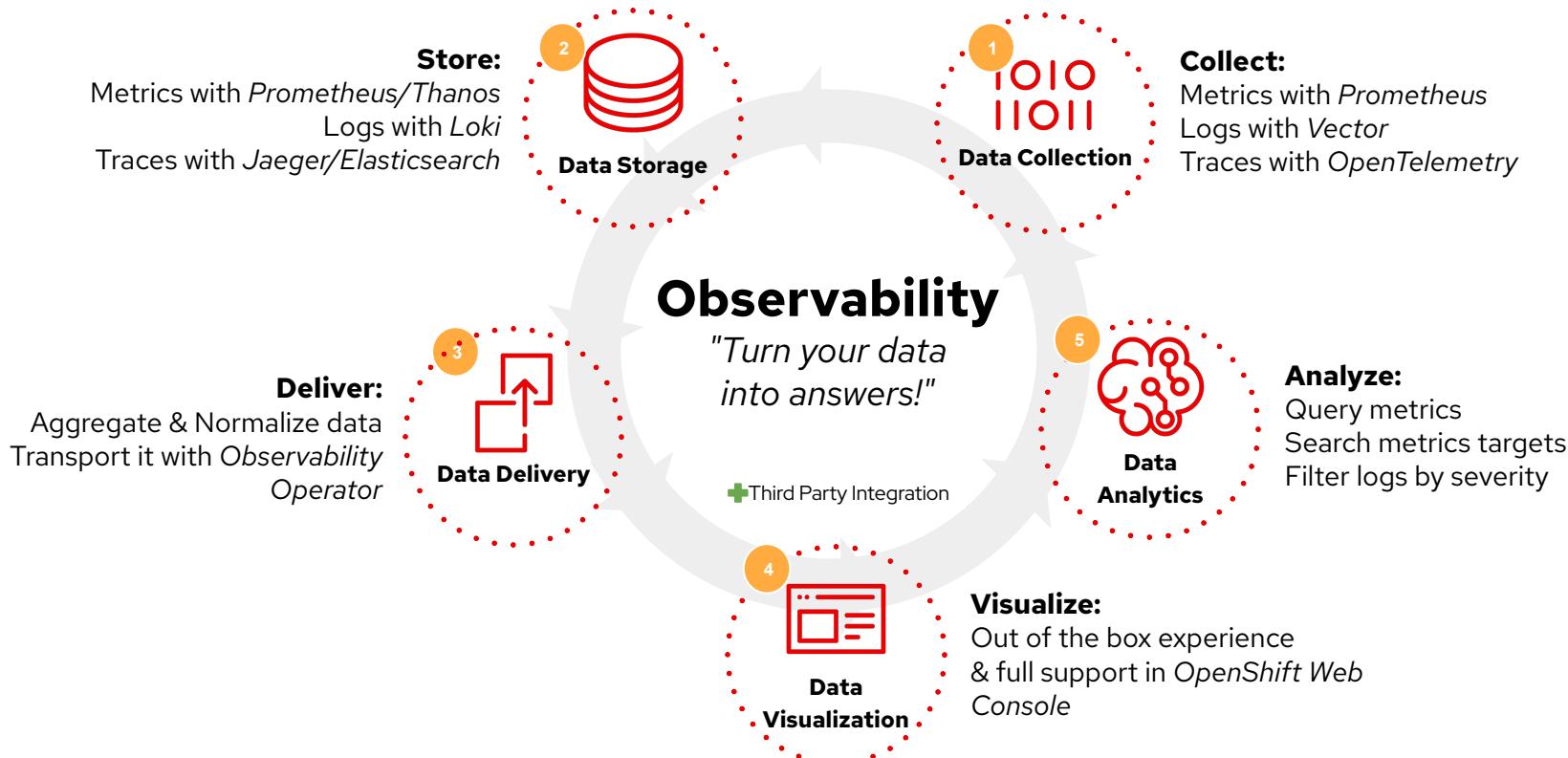
Dynamic Storage Provisioning

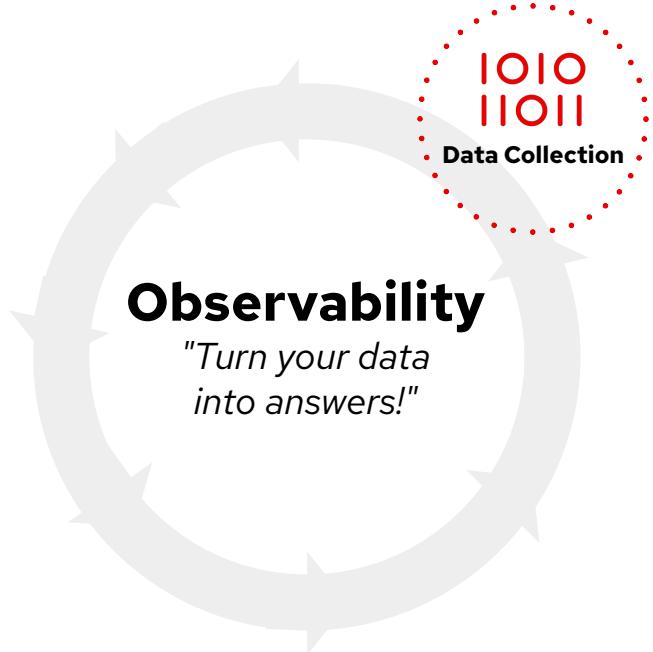




Observability

OpenShift Observability: Five Pillars





OpenShift 4.12 Monitoring

- ▶ Option to specify Topology Spread Constraints for Prometheus, Alertmanager & Thanos Ruler
- ▶ Option to improve consistency of prometheus-adapter CPU and RAM time series

Logging 5.6

- ▶ GA release of Vector as an alternate collector to Fluentd



OpenShift 4.12 Monitoring

- ▶ Version updates to Monitoring stack components & dependencies

Logging 5.6

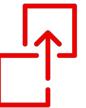
- ▶ Exposed stream-based retention capabilities in the Loki Stack custom resource for OpenShift Application owners and OpenShift Administrators



Observability

*"Turn your data
into answers!"*

75



Data Delivery



OpenShift 4.12 Monitoring

- ▶ Tech Preview: Allow admin users to create new alerting rules based on platform metrics

Logging 5.6

- ▶ Support for forwarding logs to Splunk



76

OpenShift 4.12 Monitoring

- ▶ Improved UX experience in OpenShift Web Console:
Easier selection of records in Metrics UI
- ▶ Support for Alertmanager's negative matchers

Logging 5.6

- ▶ Log Exploration UI available in OpenShift Web Console
 - Developer Perspective: Observe > Aggregated Logs
- ▶ Improved UX experience in OpenShift Web Console:
Custom time range & Predefined filters to easily search logs (namespace, pod, container)

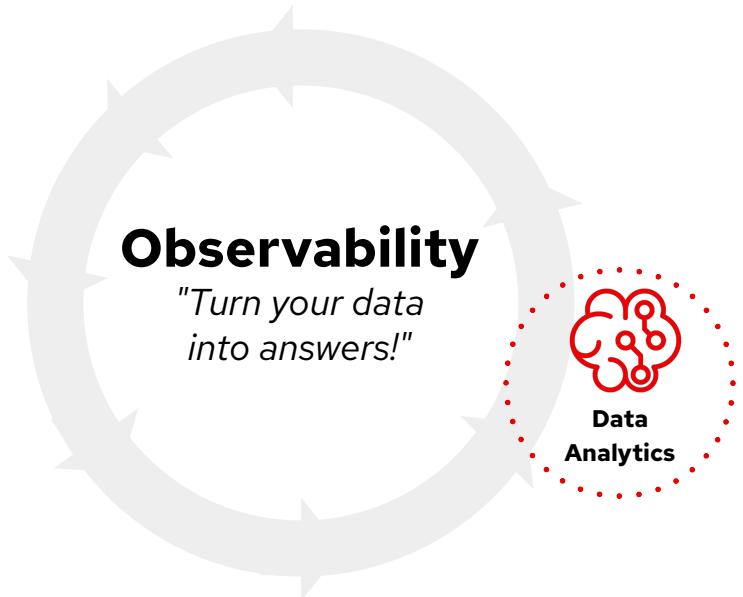
OpenShift Observability

The screenshot shows the Red Hat OpenShift Developer Console interface. The top navigation bar includes the Red Hat OpenShift logo, a developer icon, and a dropdown for 'dev-user'. The left sidebar has a 'Developer' section with '+Add', 'Topology', 'Observe', 'Search', 'Builds', 'Helm', 'Project', 'ConfigMaps', and 'Secrets'. The main content area is titled 'Project: test' and 'Observe'. It features tabs for 'Dashboard', 'Metrics', 'Events', and 'Aggregated Logs', with 'Aggregated Logs' currently selected. Below the tabs is a histogram titled 'Logs' showing log levels over time from 12:09 to 13:09. A red arrow points to this histogram. Below the histogram is a table of log entries with columns for 'Date' and 'Message'. A red arrow points to the 'Run Query' button at the top of this table. The table contains several log entries, including one with a long message about solving the Travelling Salesman problem.

Date	Message
21 Nov 2022, 13:09:35.999	ts=2022-11-21T12:09:35.999567044Z stream=stderr host=logger-77f7648b76-s8lsw.0.00000000000000008CBD706BDE6FD2F3 level=error count=1711 msg="could not compute the last digit of PI"
21 Nov 2022, 13:09:34.999	ts=2022-11-21T12:09:34.999378931Z stream=stderr host=logger-77f7648b76-s8lsw.0.00000000000000008CBD706BDE6FD2F3 level=error count=1710 msg="failed to get an error message"
21 Nov 2022, 13:09:33.999	ts=2022-11-21T12:09:33.99928267Z stream=stderr host=logger-77f7648b76-s8lsw.0.00000000000000008CBD706BDE6FD2F3 level=info count=1709 msg="John Doe solved the Travelling Salesman problem in O(1) time. Here's the pseudo-code: Break salesman into N pieces. Kick each piece to a different city."
21 Nov 2022, 13:09:32.999	ts=2022-11-21T12:09:32.999176279Z stream=stderr host=logger-77f7648b76-s8lsw.0.00000000000000008CBD706BDE6FD2F3 level=warn count=1708 msg="failing to cook potatoes"
21 Nov 2022, 13:09:31.999	ts=2022-11-21T12:09:31.999070109Z stream=stdout host=logger-77f7648b76-s8lsw.0.00000000000000008CBD706BDE6FD2F3 level=warn count=1707 msg="change stuff and see what happens"
21 Nov 2022, 13:09:30.999	ts=2022-11-21T12:09:30.998969983Z stream=stderr host=logger-77f7648b76-s8lsw.0.00000000000000008CBD706BDE6FD2F3 level=error count=1706 msg="failed to reach the cloud, try again on a rainy day"

New entry:
Aggregated Logs
view in Developer
Console

Improved UX:
Filter by content
(namespace, pod,
container) AND
Search by content
AND Filter by
severity



OpenShift 4.12 Monitoring

- ▶ Runbooks URLs enabled in the Alerting UI of OpenShift Console

Logging 5.6

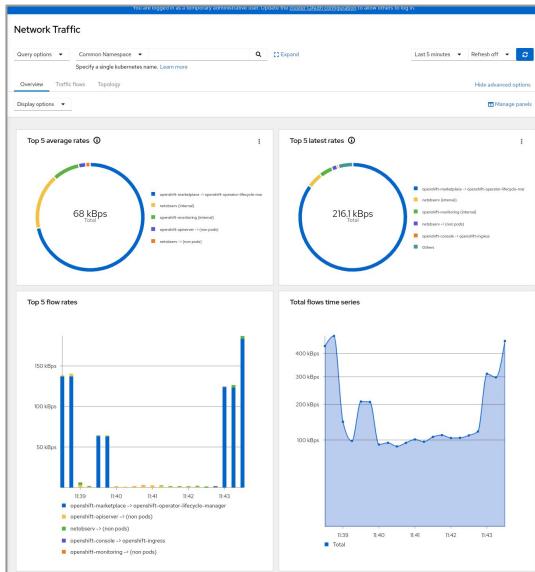
- ▶ Add the OpenShift cluster ID to log records so that clusters can be uniquely identified in aggregated logs



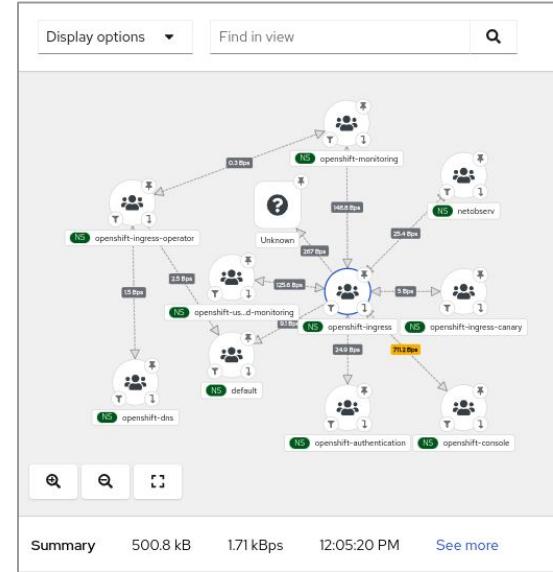
Network Observability

Network Observability GAs at 4.12 for all supported versions of OpenShift at 4.10 or newer

- Integrated with the larger Observability ecosystem, this optional Operator focuses on networking information for a single cluster
- Uses an **eBPF-based** agent on cluster nodes to collect metrics
- Provides observable network traffic metrics, flows, topology and tracing



End Time	Source	Namespace	Port
Nov 22, 2022, 11:51:25.209 AM	P installer-5-ip-10-0-131-74.us-east...	openshift-kube-scheduler	8443
Nov 22, 2022, 11:51:25.206 AM	N ip-10-0-131-74.us-east...	n/a	39616
Nov 22, 2022, 11:51:25.206 AM	N ip-10-0-203-192.us-east...	n/a	30377
Nov 22, 2022, 11:51:25.205 AM	P apiserver-7764cf65b-kkb4d	openshift-apiserver	8443
Nov 22, 2022, 11:51:25.205 AM	P apiserver-7764cf65b-r5lqg	openshift-apiserver	8443
Nov 22, 2022, 11:51:25.204 AM	N ip-10-0-203-192.us-east...	n/a	53633
Nov 22, 2022, 11:51:25.203 AM	P apiserver-7764cf65b-r5lqg	openshift-apiserver	8443
Nov 22, 2022, 11:51:25.203 AM	N ip-10-0-131-74.us-east...	n/a	22033
Nov 22, 2022, 11:51:25.202 AM	N ip-10-0-203-192.us-east...	n/a	2380



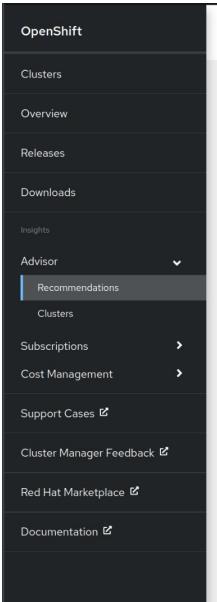


Insights Advisor for OpenShift

- ▶ **Free service leveraging Red Hat experience with supporting and operating OpenShift**
- ▶ New recommendations based on analysis of Kubernetes YAML files (available for managed OpenShift only ATM)
- ▶ Alerts in OpenShift WebConsole for most critical recommendations
- ▶ **New recommendations focused on storage performance, etcd issues etc.**
- ▶ Improved internal integrations for more stable upgrades

<https://console.redhat.com/openshift/advisor>

<https://console.redhat.com/settings/notifications/openshift>

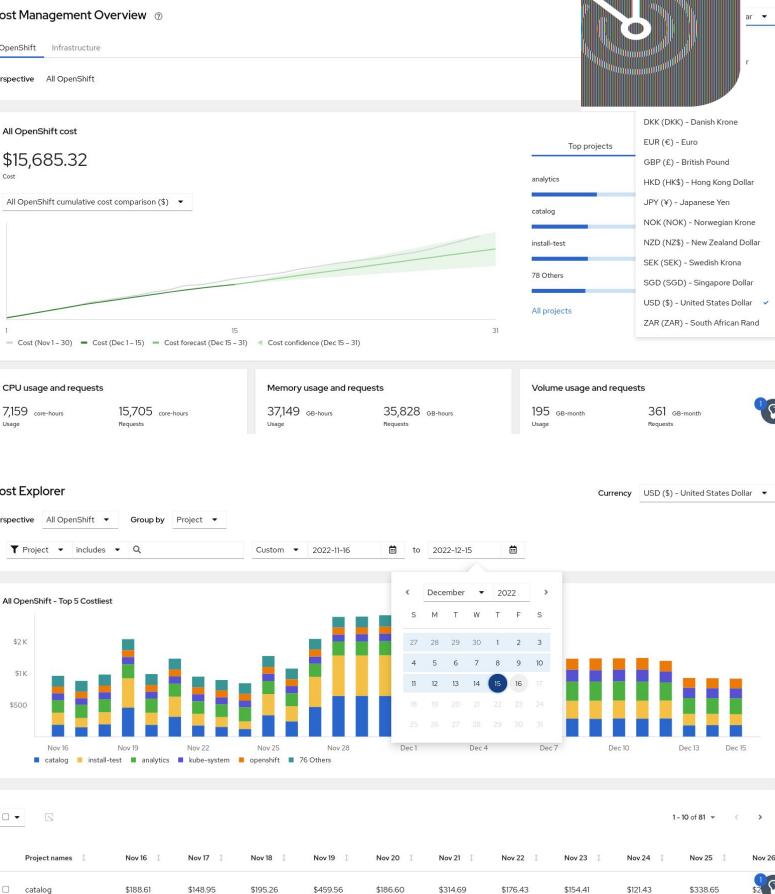


Advisor recommendations

Name	Added	Category	Total risk	Clusters
Cluster upgrade will fail when default SCC gets changed	2 years ago	Service Availability	Important	1
Workloads are still using the deprecated APIs which will be removed in the next release	5 months ago	Service Availability	Important	6
SystemMemoryExceedsReservation alerts when the system daemons memory usage on nodes exceeds 90% of the reservation for them	4 months ago	Service Availability	Important	1
Workloads are using the deprecated PodSecurityPolicy API	5 months ago	Performance	Moderate	3
CVE-2021-30465: runc vulnerable to privilege escalation	9 months ago	Security	Moderate	1
Nodes will become Not Ready due to a CRI-O PID leak in the running OpenShift Container Platform version	5 months ago	Service Availability	Moderate	13
The running OpenShift version has reached its End of Life	2 years ago	Service Availability	Moderate	1
Pods could fail to start if openshift-samples is degraded due to FailedImageImport which is caused by a hiccup while talking to the Red Hat registry	2 years ago	Service Availability	Moderate	1
Prometheus metrics data will be lost when the Prometheus pod is restarted or recreated	1 year ago	Service Availability	Moderate	50
The authentication operator is degraded when cluster is configured to use a cluster-wide proxy	2 years ago	Security	Moderate	1
An OCP node behaves unexpectedly when it doesn't meet the minimum resource requirements	2 years ago	Performance	Moderate	2

Insights Cost Management

- ▶ **Free service to monitor per-project and per-cluster spending**
- ▶ Currency support
- ▶ Marketplace services reported including ROSA, ARO, RHEL, ODF, 3rd parties, etc
- ▶ ROSA and ARO costs distributed to projects
- ▶ Costs now distributed according to the same resource consumption criteria in every view
- ▶ Cost of unallocated capacity accounted (both workers and platform)
- ▶ Filtering gained exclude capabilities ("negative filtering")
- ▶ AWS costs default to amortized when Savings Plans are involved
- ▶ Previous month report and custom date picker in Cost Explorer
- ▶ Performance improvements for OCP clusters running on GCP
- ▶ Integration with console.redhat.com notifications



<https://console.redhat.com/openshift/cost-management>

<https://console.redhat.com/settings/applications/cost-management>

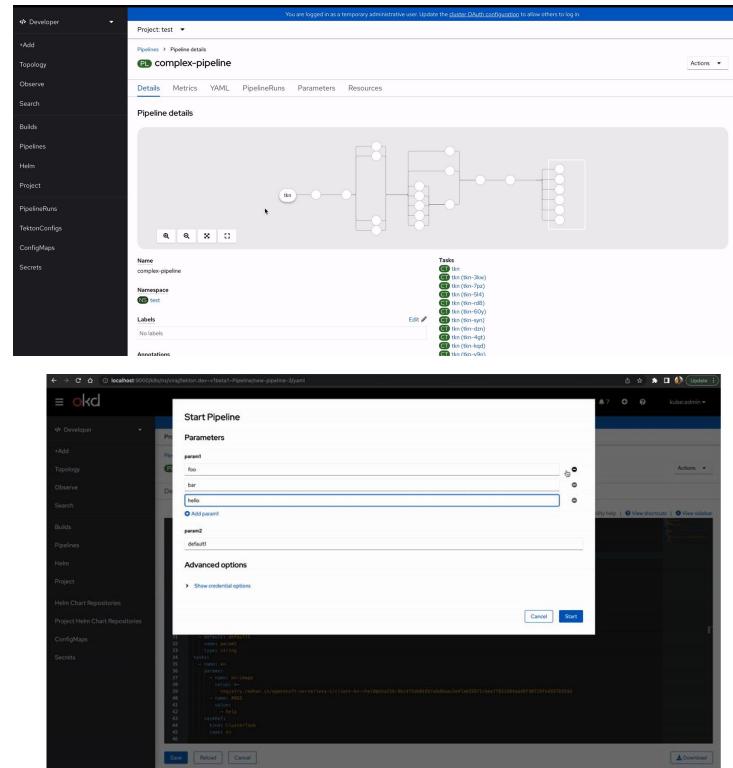
<https://console.redhat.com/settings/notifications/openshift>



Platform Services

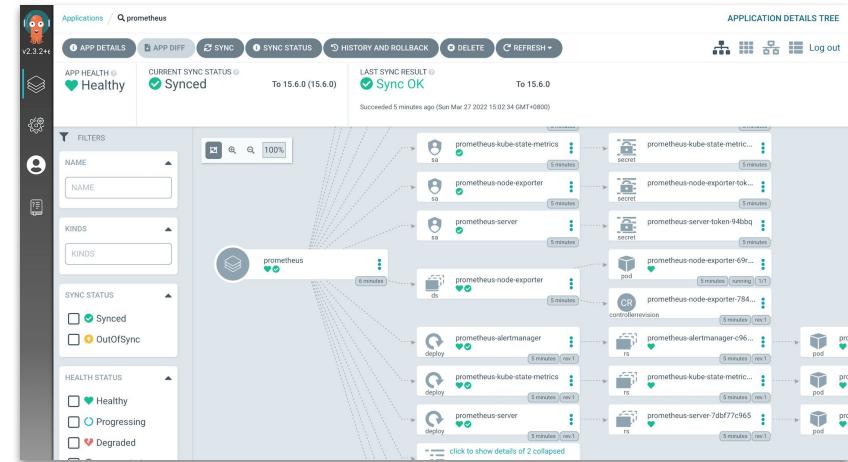
OpenShift Pipelines

- ▶ OpenShift Pipelines 1.9
- ▶ Reference pipelines/tasks in Git, TektonHub, ArtifactHub, etc (Tech Preview)
- ▶ Pipelines as code GA
 - ▶ PAC concurrency control
 - ▶ Support for advanced event matching on filepath/PR title
 - ▶ Ability to enable pac for all [new] repos in a GitHub org
 - ▶ Better errors tooling in Pipelines as Code CLI
 - ▶ Rich PipelineRun details in GitHub Checks UI
- ▶ Support for CSI and projected volume for workspace
- ▶ New CLI: Openshift Pipelines CLI (opc) - Tech Preview
- ▶ Pipelines on Dev Sandbox
- ▶ Dev Console UX improvements : Pipeline topology view, Support of array in Param



OpenShift GitOps

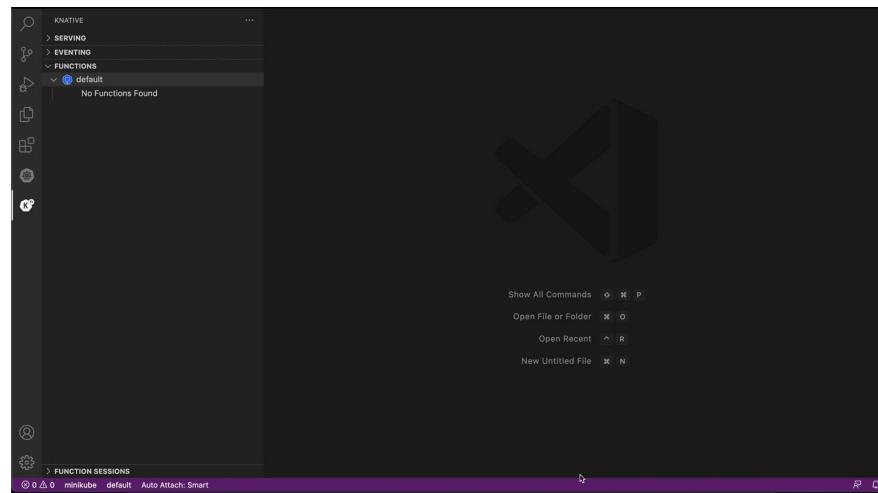
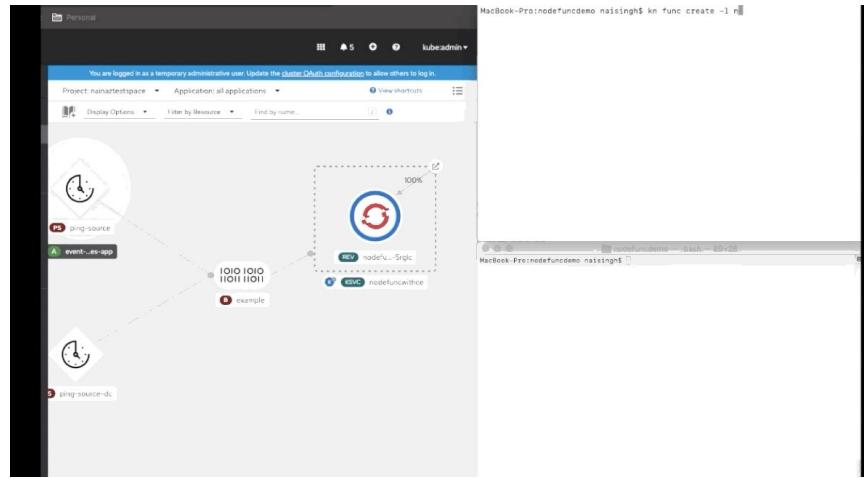
- ▶ OpenShift GitOps 1.7
- ▶ Includes Argo CD 2.6
- ▶ Patching existing resources with Server Side Apply
- ▶ Applications in non-control plane namespaces (TP)
- ▶ Operator improvements:
 - ▶ Custom node selectors
 - ▶ RBAC match mode 'regex'
 - ▶ Sub-keys for resource customizations
 - ▶ Enable/Disable cluster Argo CD console link



OpenShift Serverless

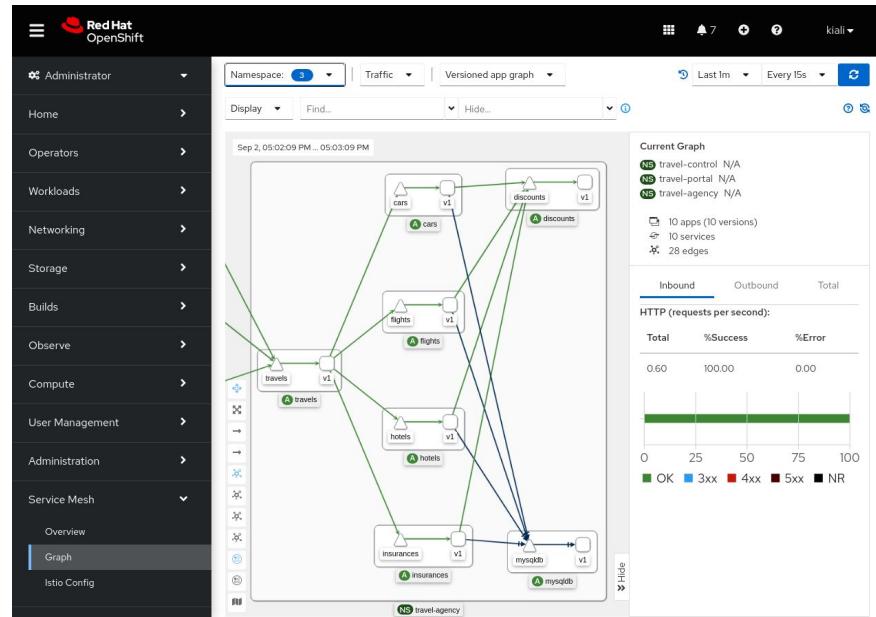
Key Features & Updates

- ▶ Update to Knative 1.6
- ▶ [Serverless functions](#) with Quarkus (GA)
 - ▶ In Cluster build using OpenShift Pipelines
 - ▶ Local experience with CLI and IDE (VScode and IntelliJ)
- ▶ [Knative Kafka Broker](#) & [Knative Kafka Sink](#) (GA)
- ▶ Support for [Init Containers](#) and [PVC](#) (GA)
- ▶ mTLS natively in Serverless (Tech Preview)
- ▶ Serverless Logic (Dev Preview)
 - ▶ Orchestration for Functions and Services
 - ▶ CLI and Workflow Editor(UX)



OpenShift Service Mesh

- ▶ OpenShift Service Mesh 2.3 is now available
- ▶ Based on **Istio 1.14** and **Kiali 1.57**
- ▶ Introduces GA support for **Gateway Injection**
- ▶ New Technology Preview features:
 - ▶ OpenShift **Console Service Mesh Plugin**
 - ▶ **Cluster-wide mesh** installation option
 - ▶ Kubernetes **Gateway API** (Kiali support added)
- ▶ Service Mesh federation is now supported on Azure Red Hat OpenShift (ARO)





[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