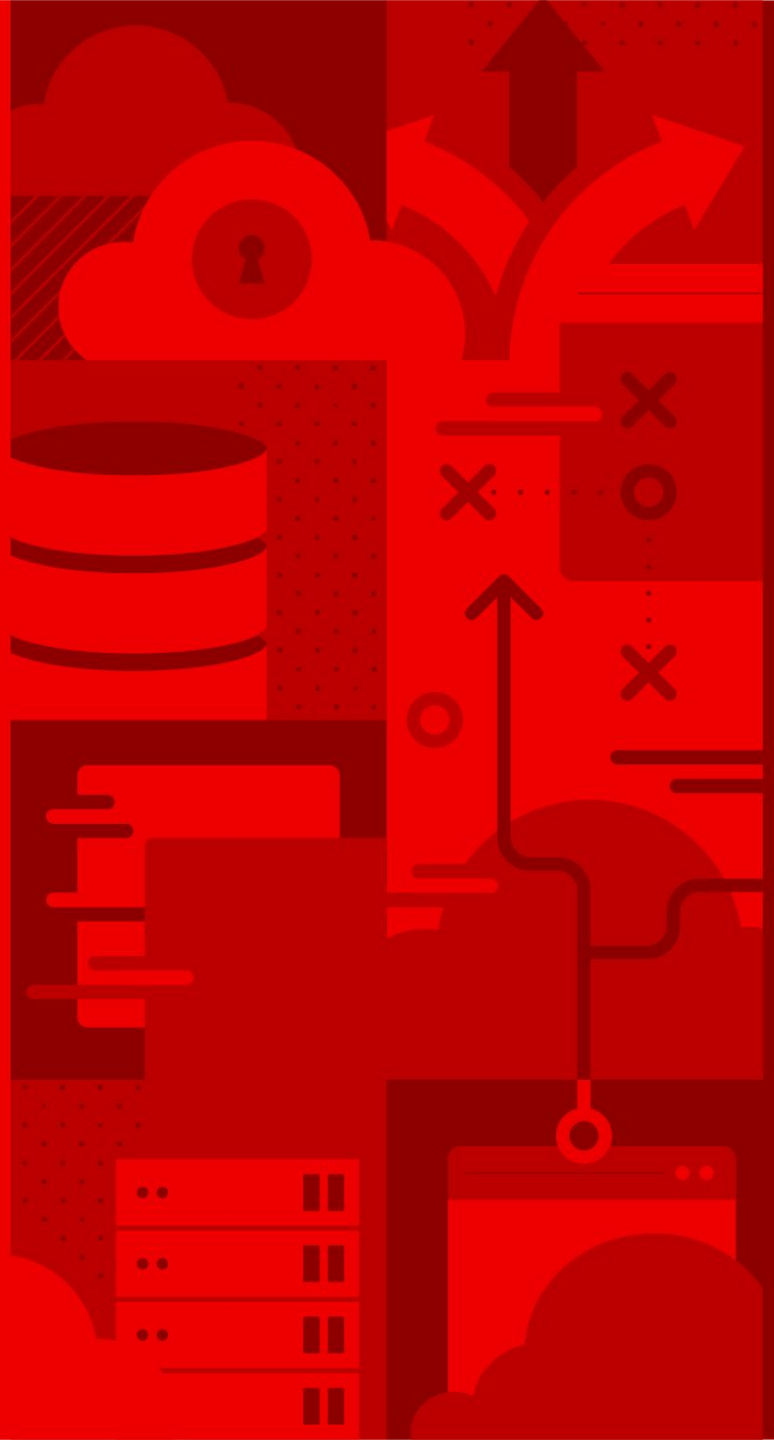

Red Hat
IBM

Red Hat OpenShift

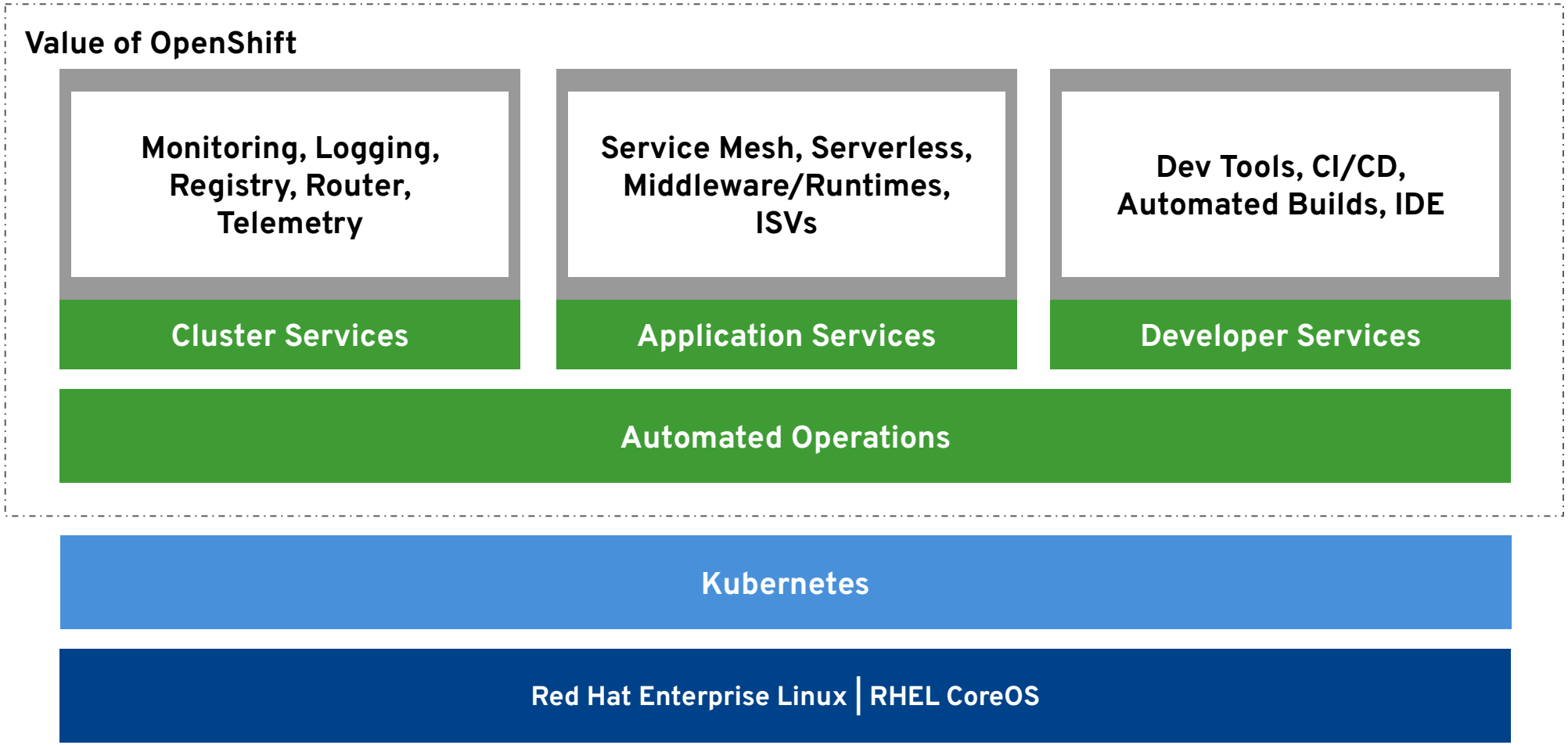
OpenShift Architecture

OpenShift Architecture Workshop for IBM Nordics

Alfred Bach
PESA RedHat EMEA
abach@redhat.com



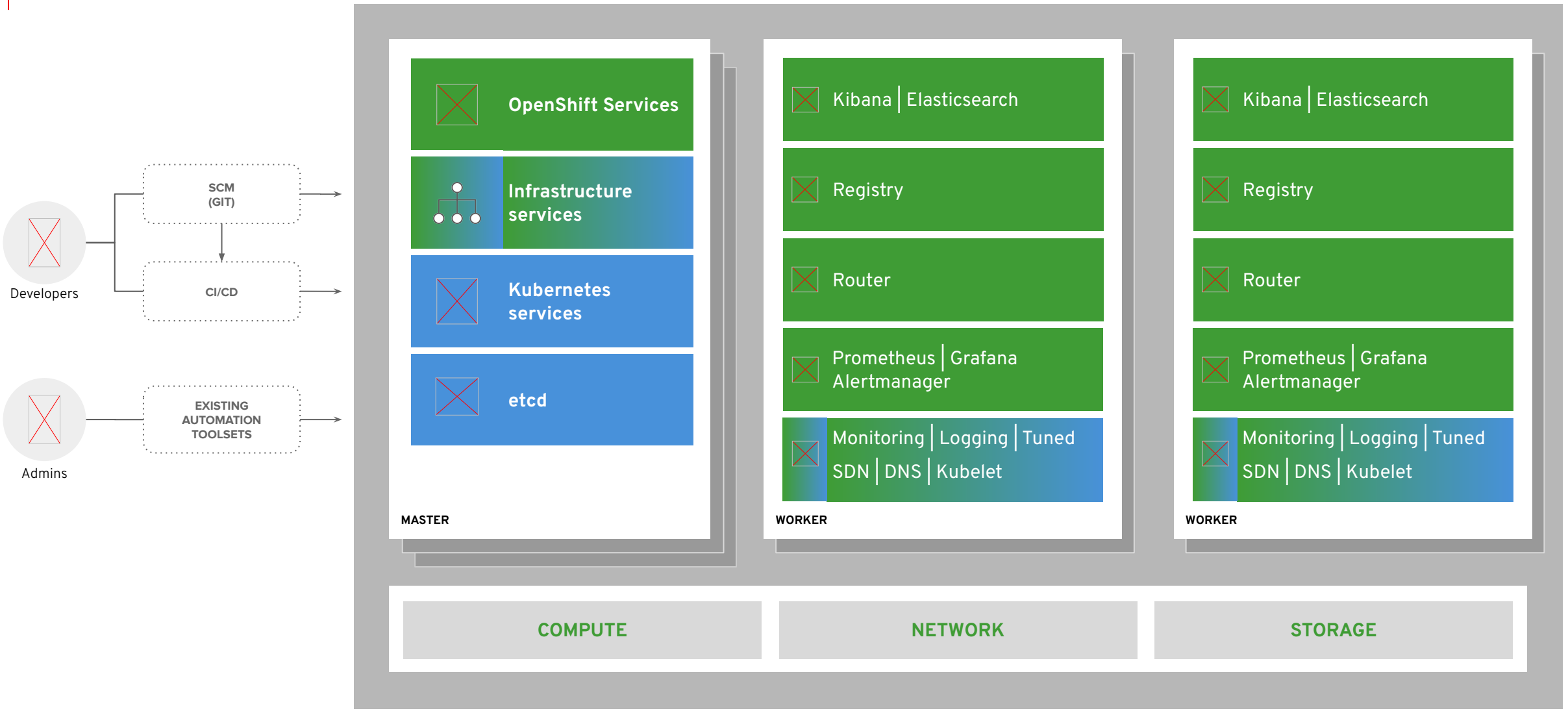
Functional overview

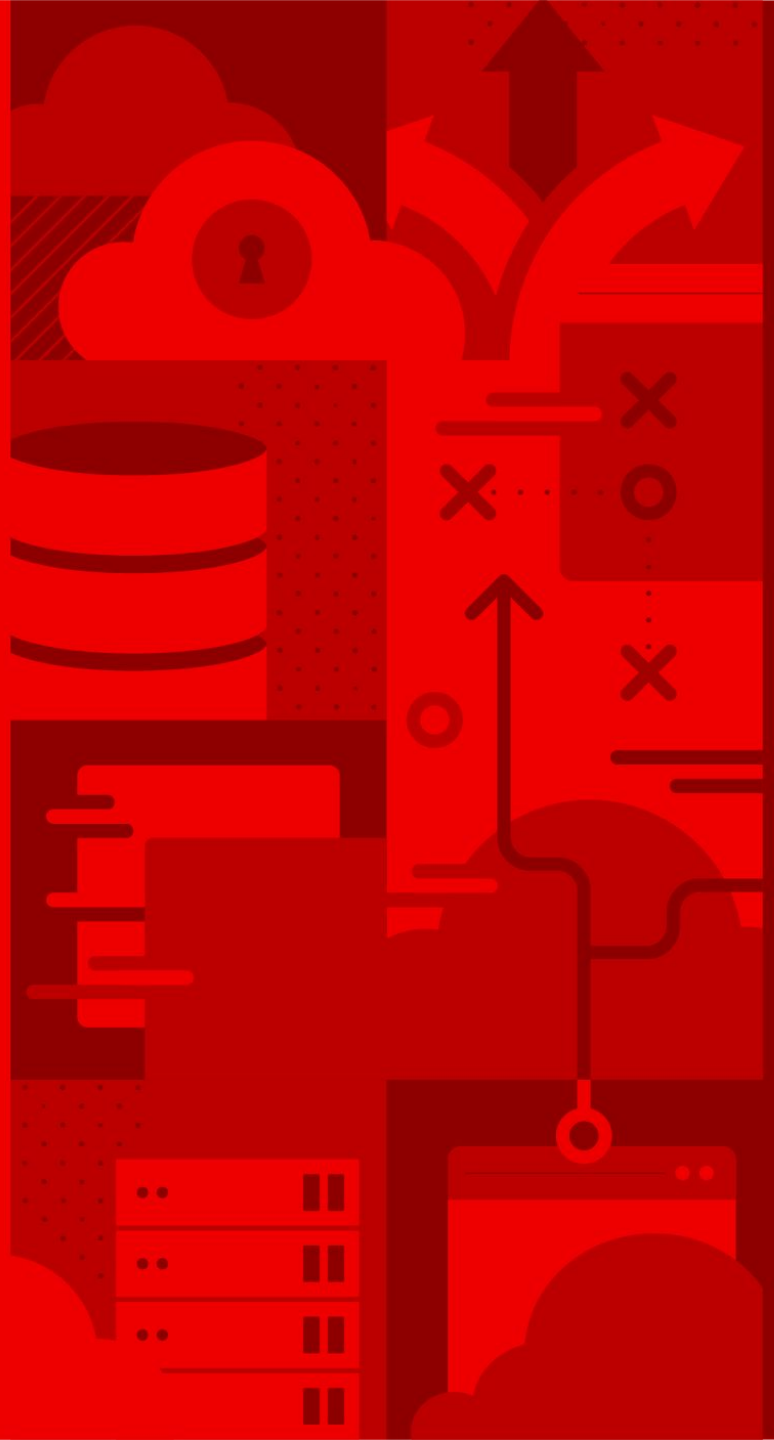


Best IT Ops Experience

CaaS ⟷ PaaS ⟷ FaaS

Best Developer Experience


SCM (GIT)
CI/CD
Developers
EXISTING AUTOMATION TOOLSETS
Admins
OpenShift Services
Infrastructure services
Kubernetes services
etcd
MASTER
Kibana | Elasticsearch
Registry
Router
Prometheus | Grafana Alertmanager
Monitoring | Logging | Tuned
SDN | DNS | Kubelet
WORKER
Kibana | Elasticsearch
Registry
Router
Prometheus | Grafana Alertmanager
Monitoring | Logging | Tuned
SDN | DNS | Kubelet
WORKER
COMPUTE
NETWORK
STORAGE

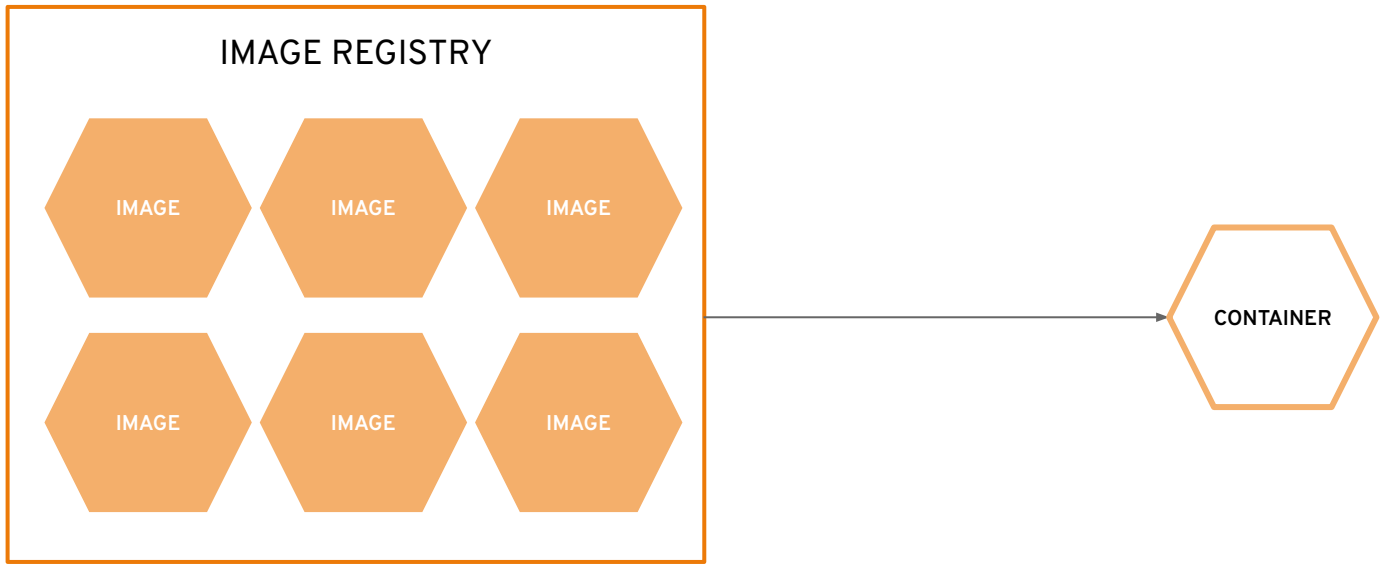


OpenShift and Kubernetes core concepts

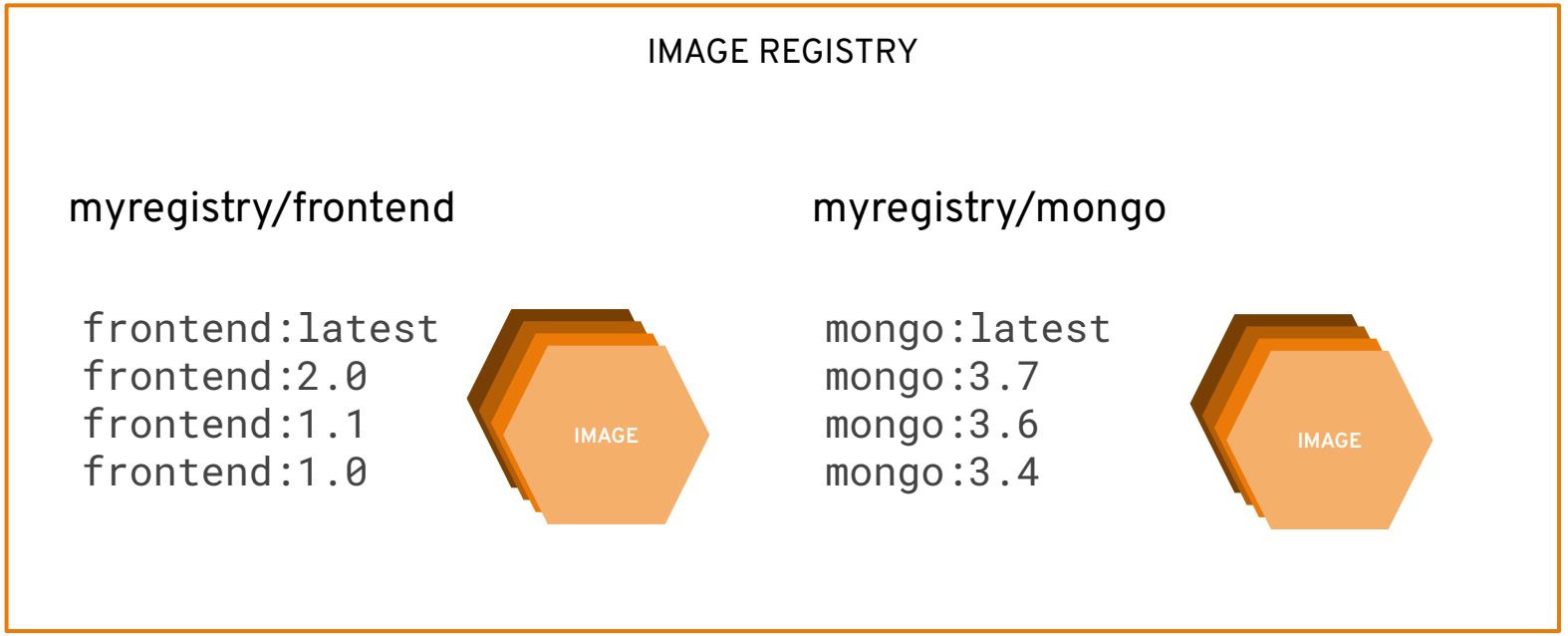
a container is the smallest compute unit

containers are created from container images

container images are stored in an image registry

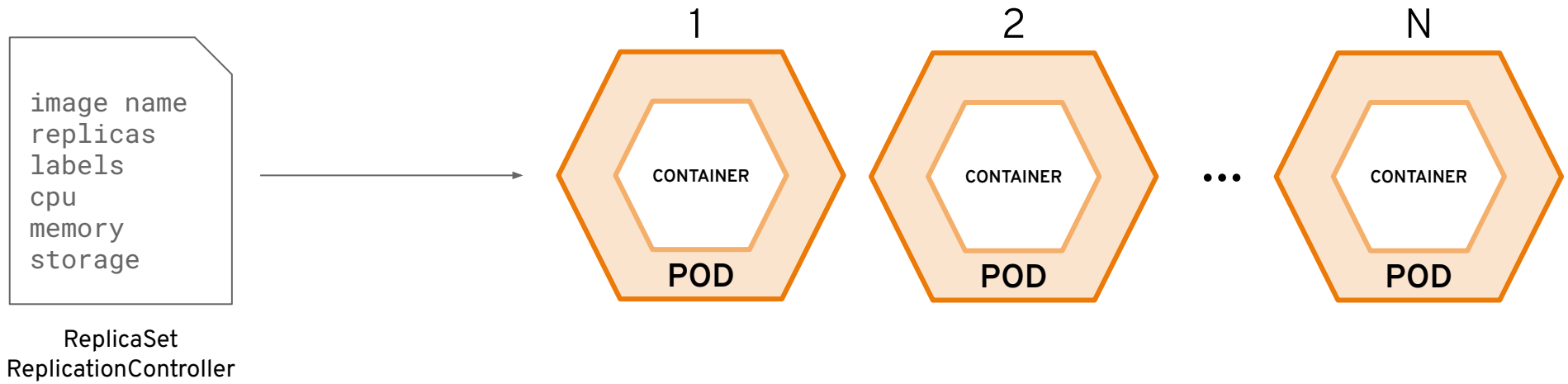


an image repository contains all versions of an image in the image registry

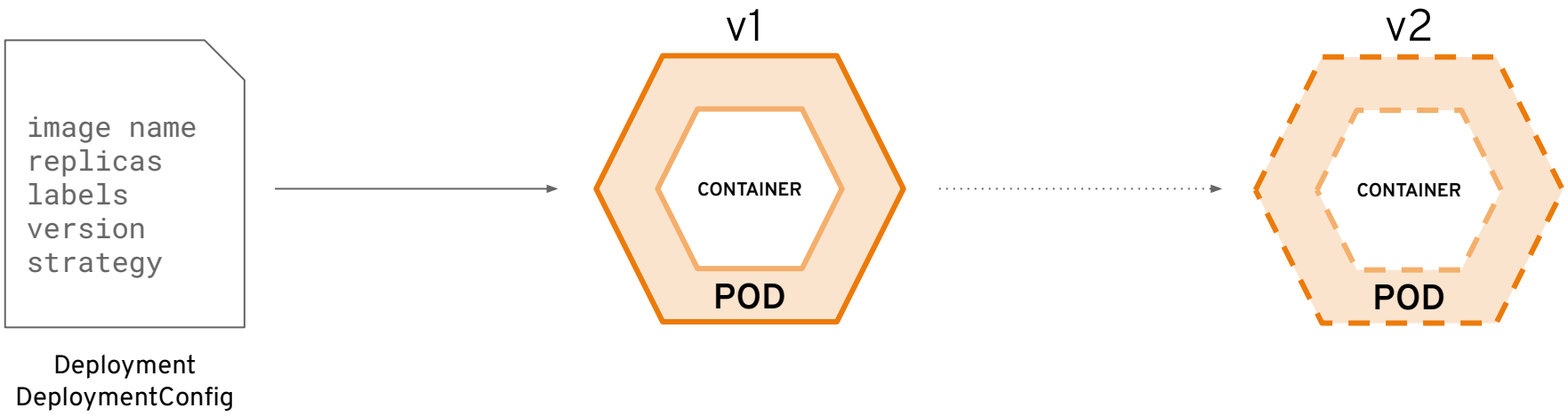


containers are wrapped in pods which are units of deployment and management

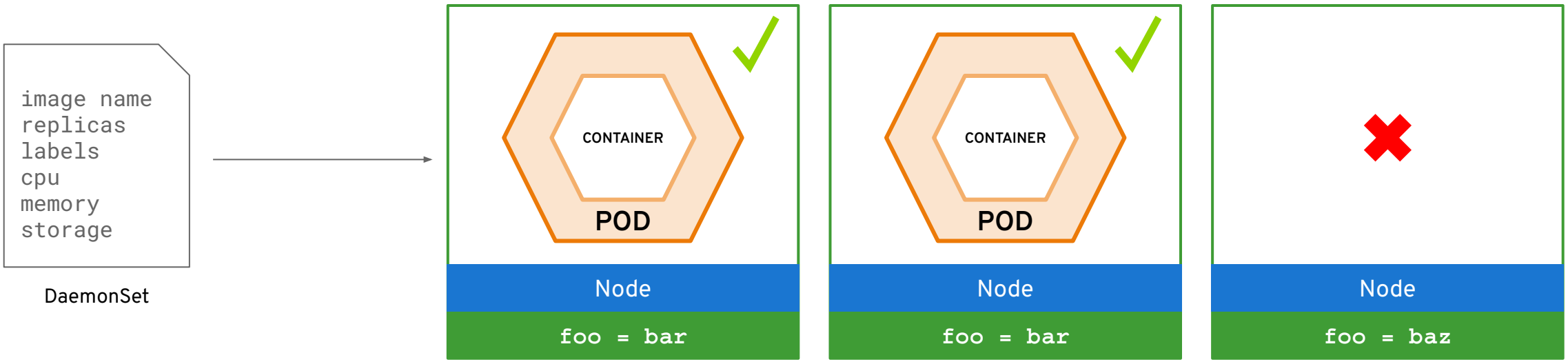
`ReplicationControllers` & `ReplicaSets` ensure a specified number of pods are running at any given time



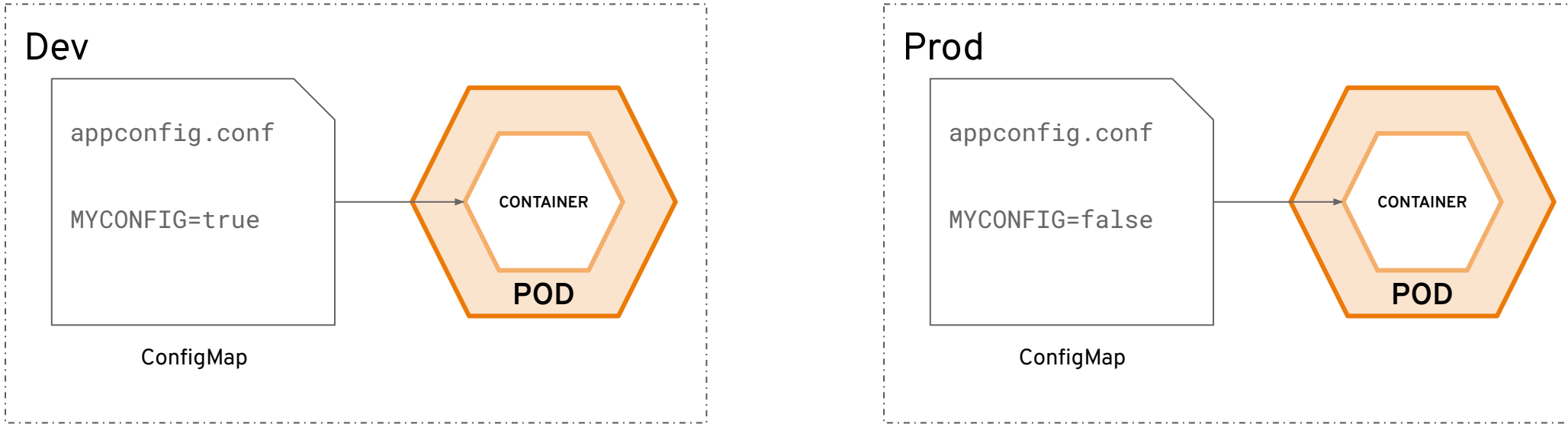
`Deployments` and `DeploymentConfigurations` define how to roll out new versions of Pods



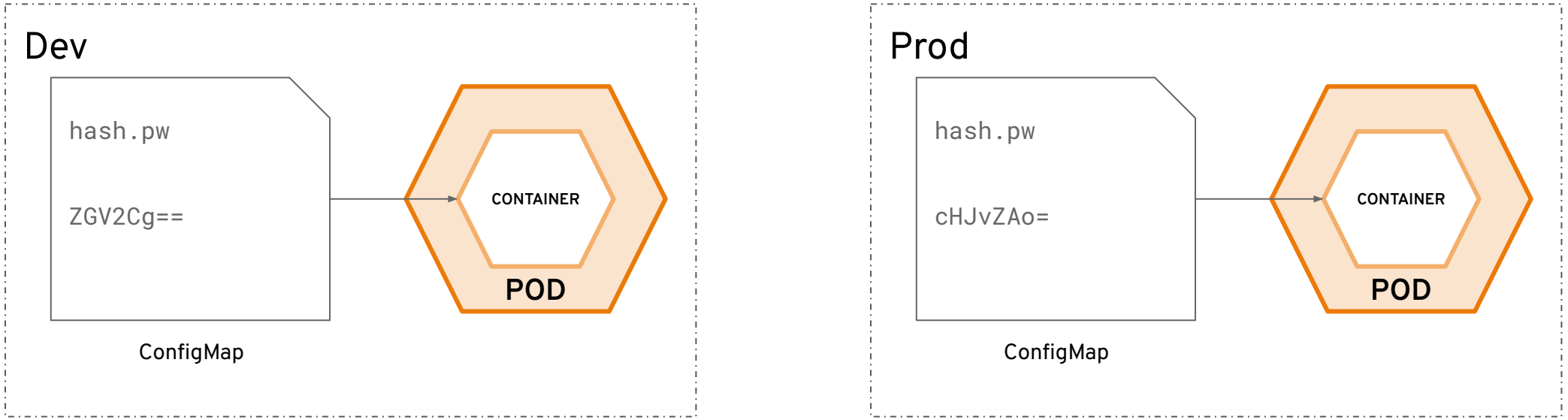
a daemonset ensures that all (or some) nodes run a copy of a pod



`configmaps` allow you to decouple configuration artifacts from image content

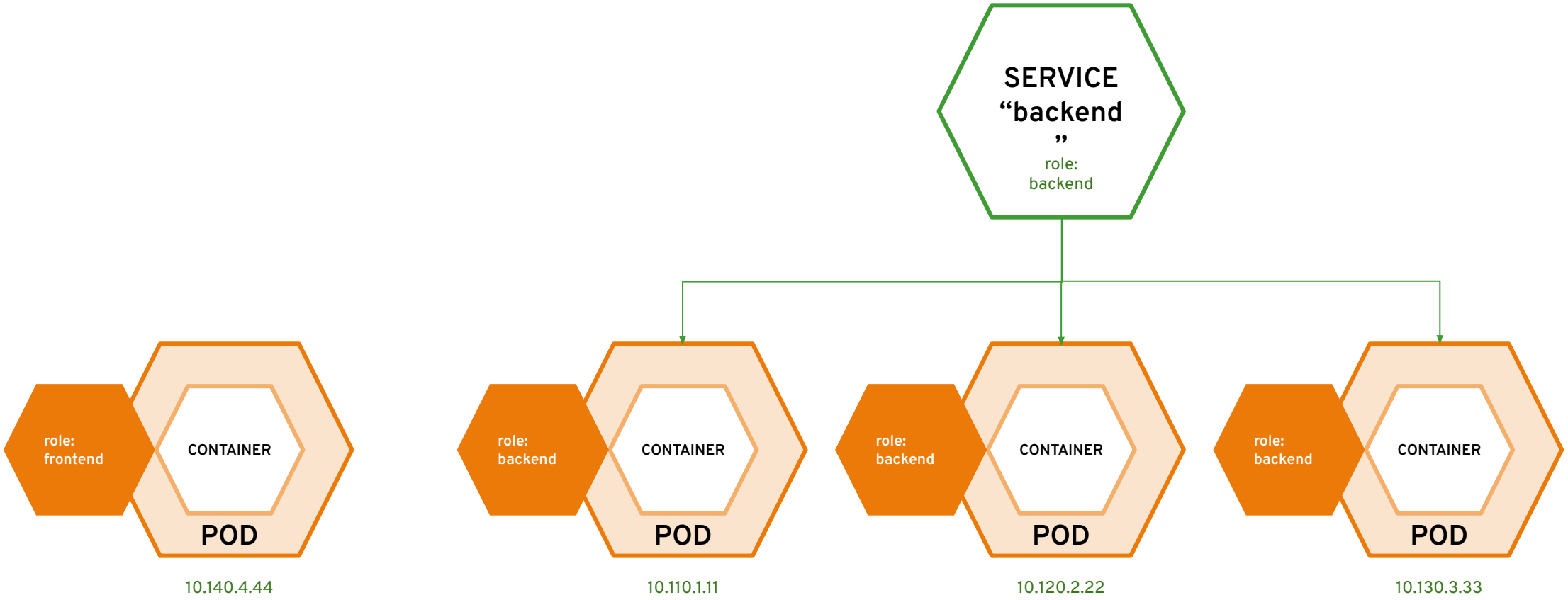


`secrets` provide a mechanism to hold sensitive information such as passwords

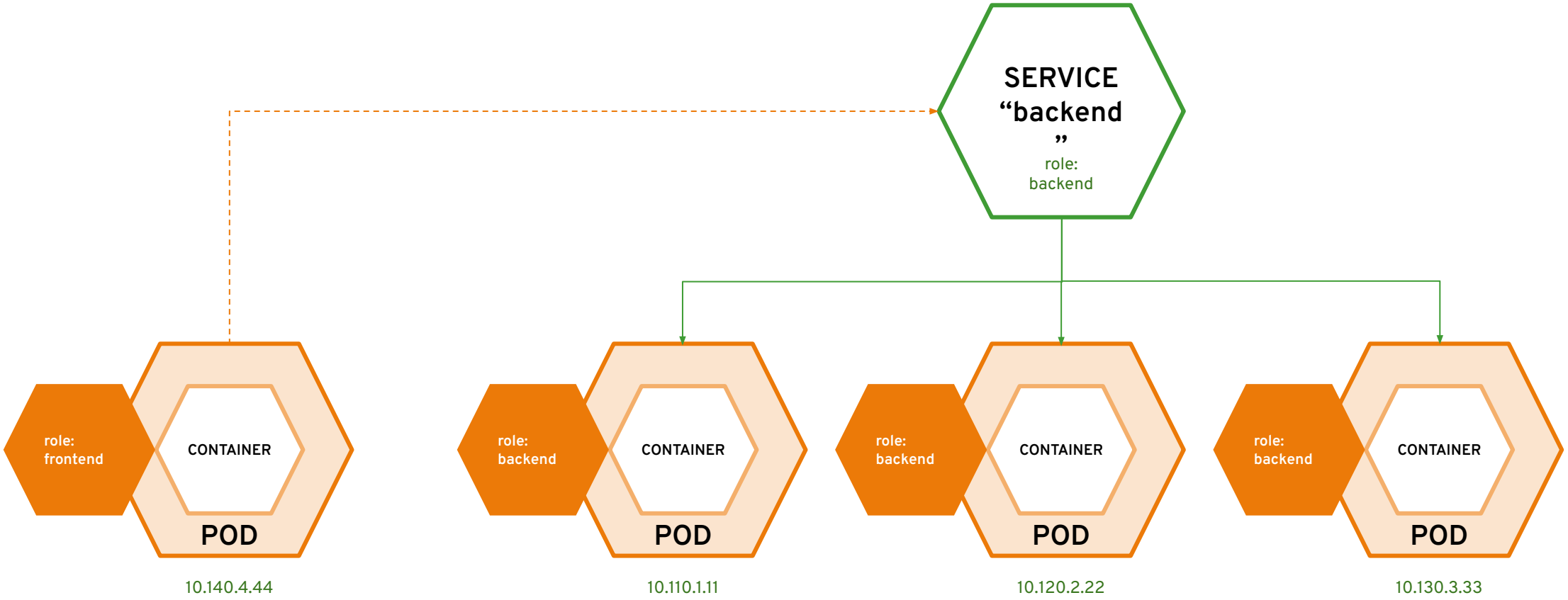


The etcd datastore can be encrypted for additional security
https://docs.openshift.com/container-platform/4.6/security/encrypting-etcd.html

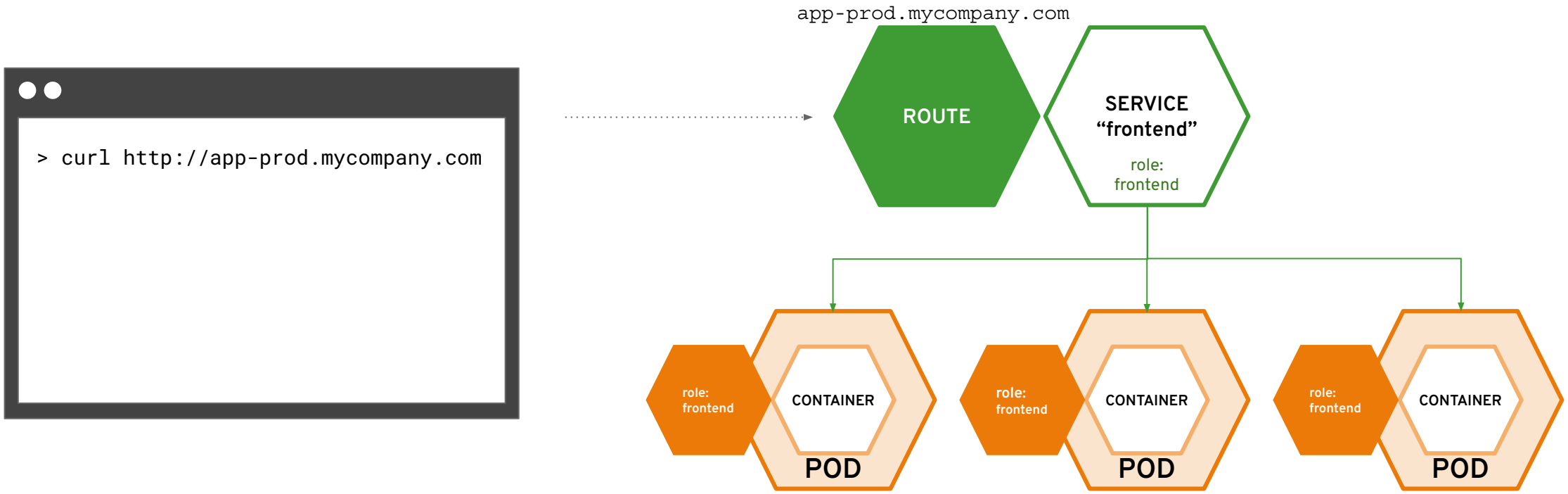
services provide internal load-balancing and service discovery across pods



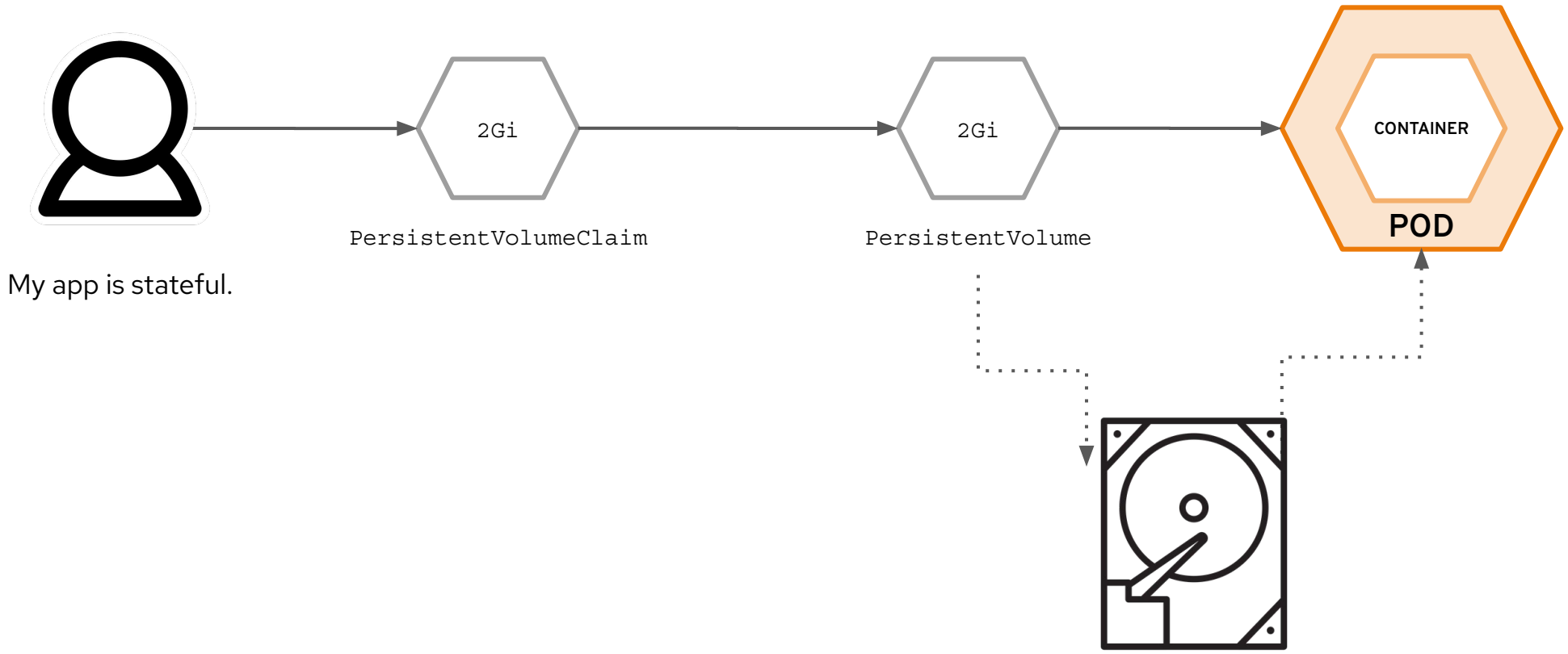
apps can talk to each other via services



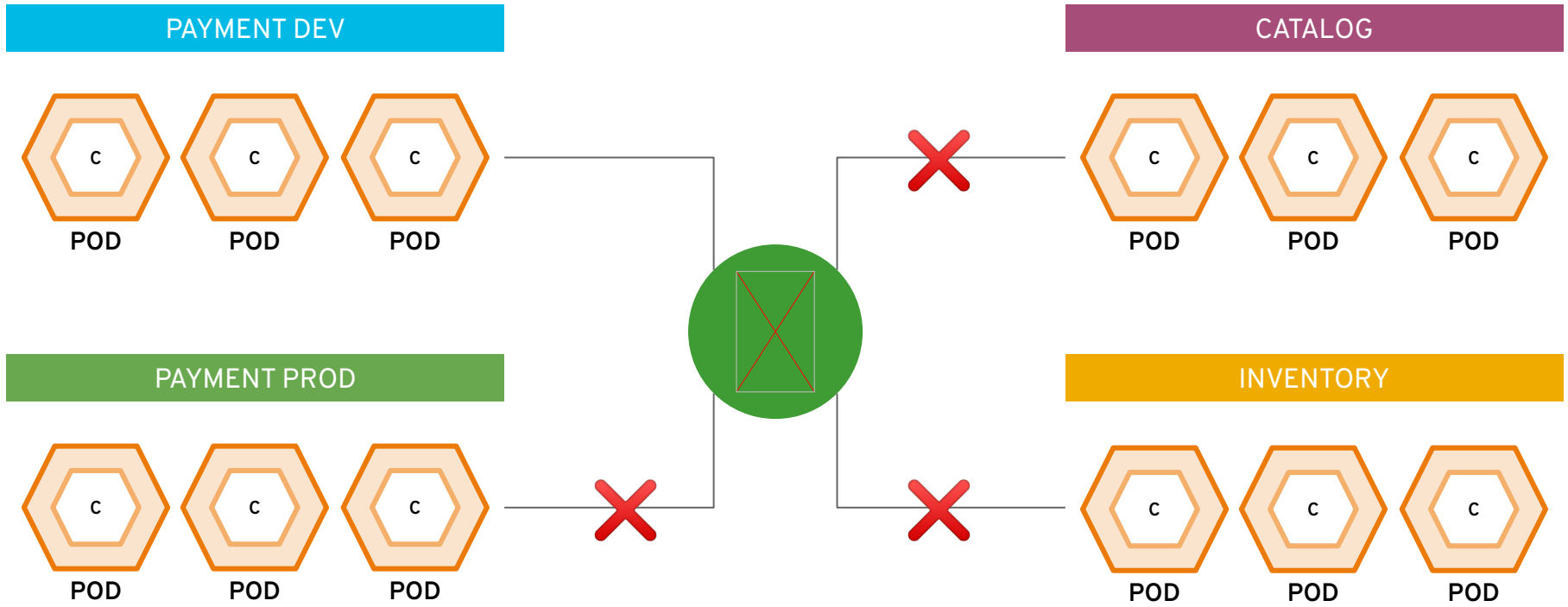
`routes` make services accessible to clients outside the environment via real-world urls

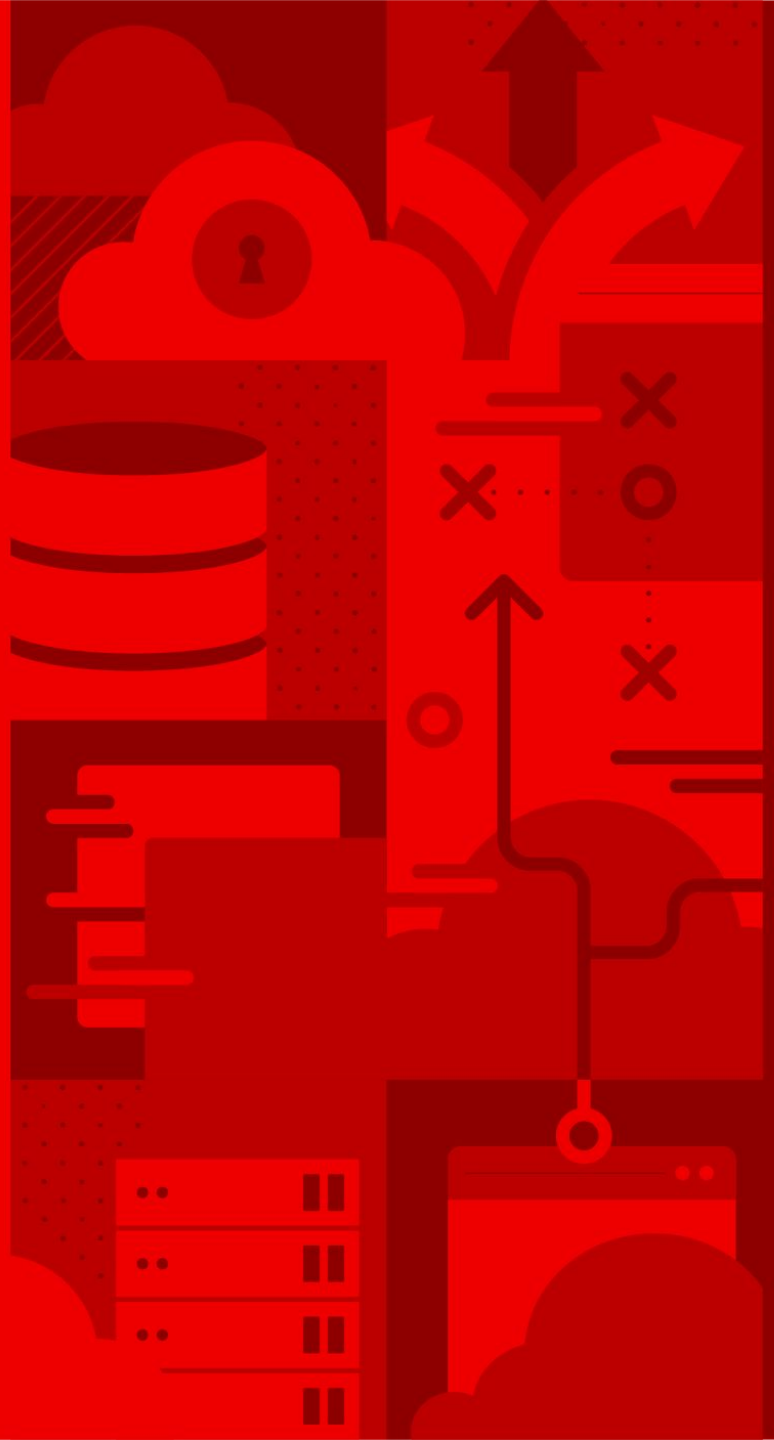


Persistent Volume and Claims



projects isolate apps across environments, teams, groups and departments





OpenShift 4 Architecture

your choice of infrastructure

COMPUTE

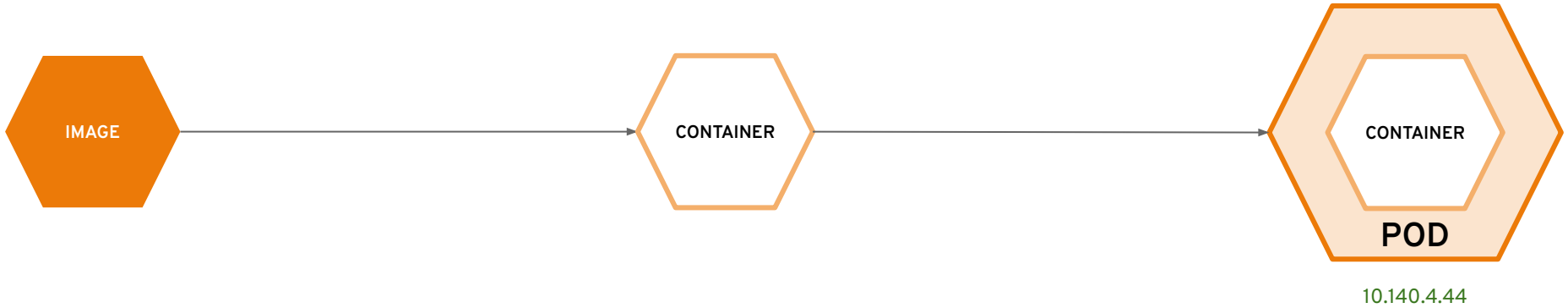
NETWORK

STORAGE

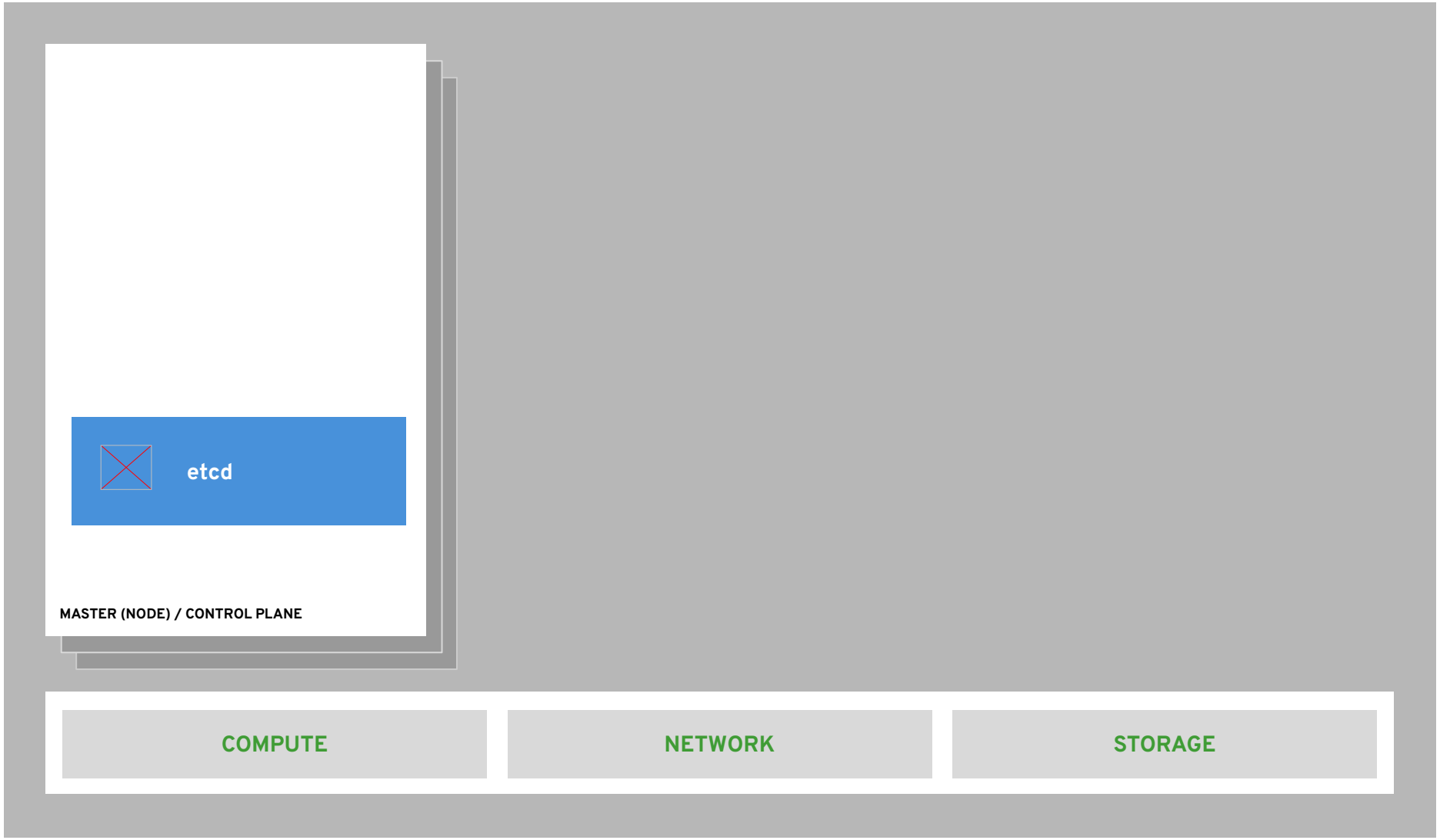
workers run workloads

masters are the control plane

everything runs in pods



state of everything



Introducing the Declarative Architecture of Kubernetes

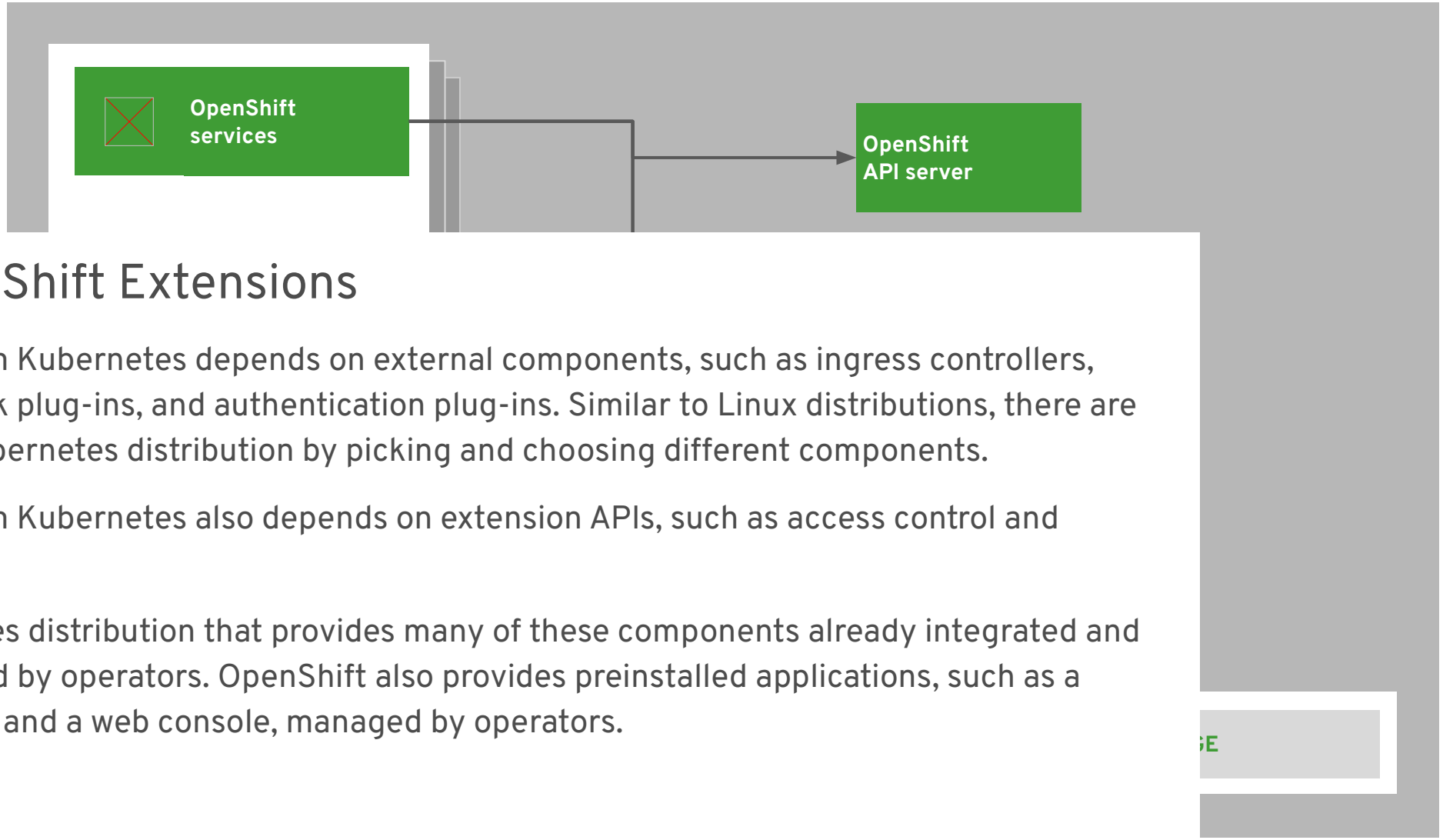
The architecture of OpenShift is based on the declarative nature of Kubernetes. Most system administrators are used to imperative architectures, where you perform actions that indirectly change the state of the system, such as starting and stopping containers on a given server. In a declarative architecture, 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



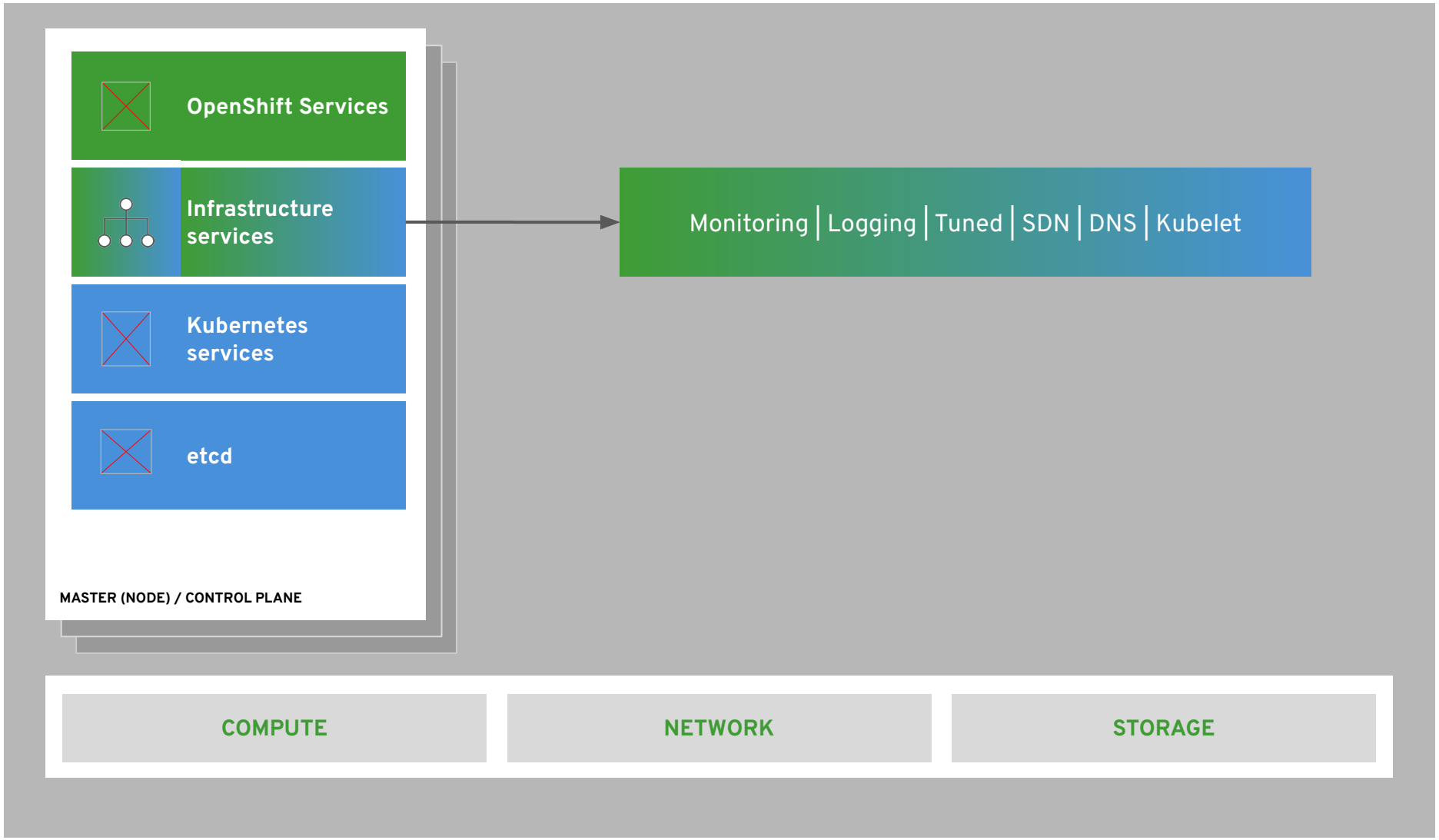
Describing OpenShift Extensions

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.

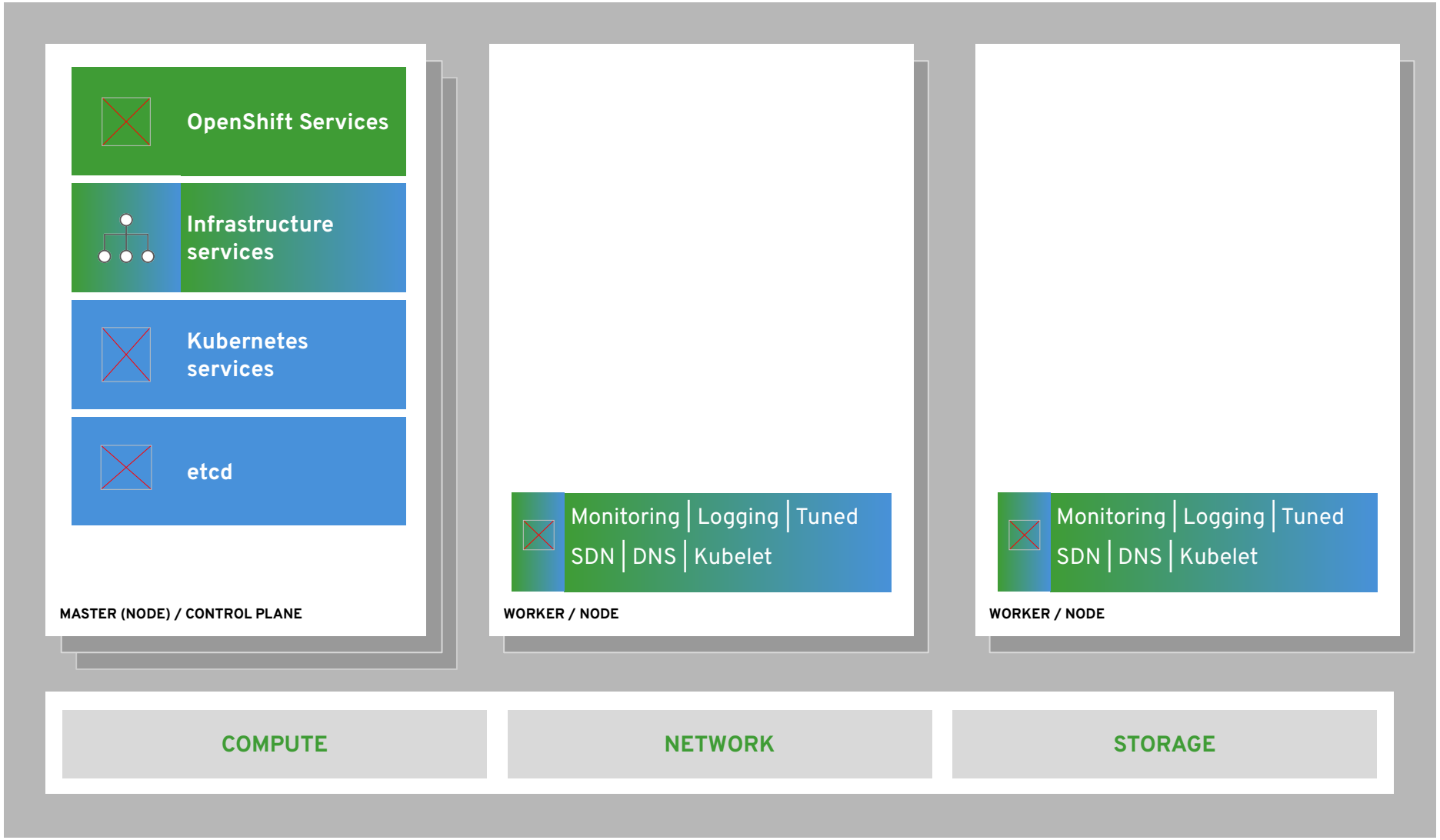
A lot of functionality from Kubernetes also depends on extension APIs, such as access control and network isolation.

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.

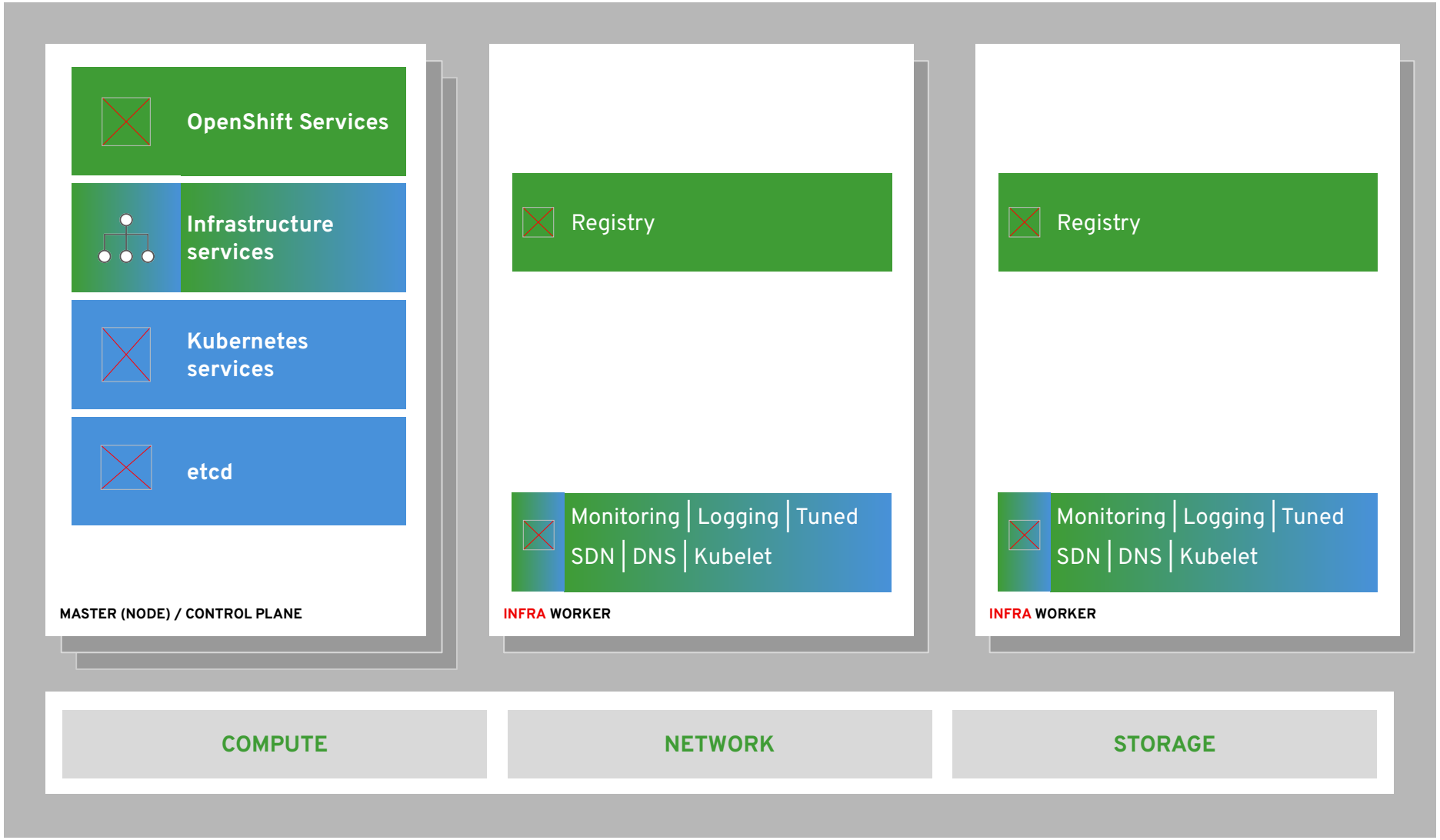
internal and support infrastructure services



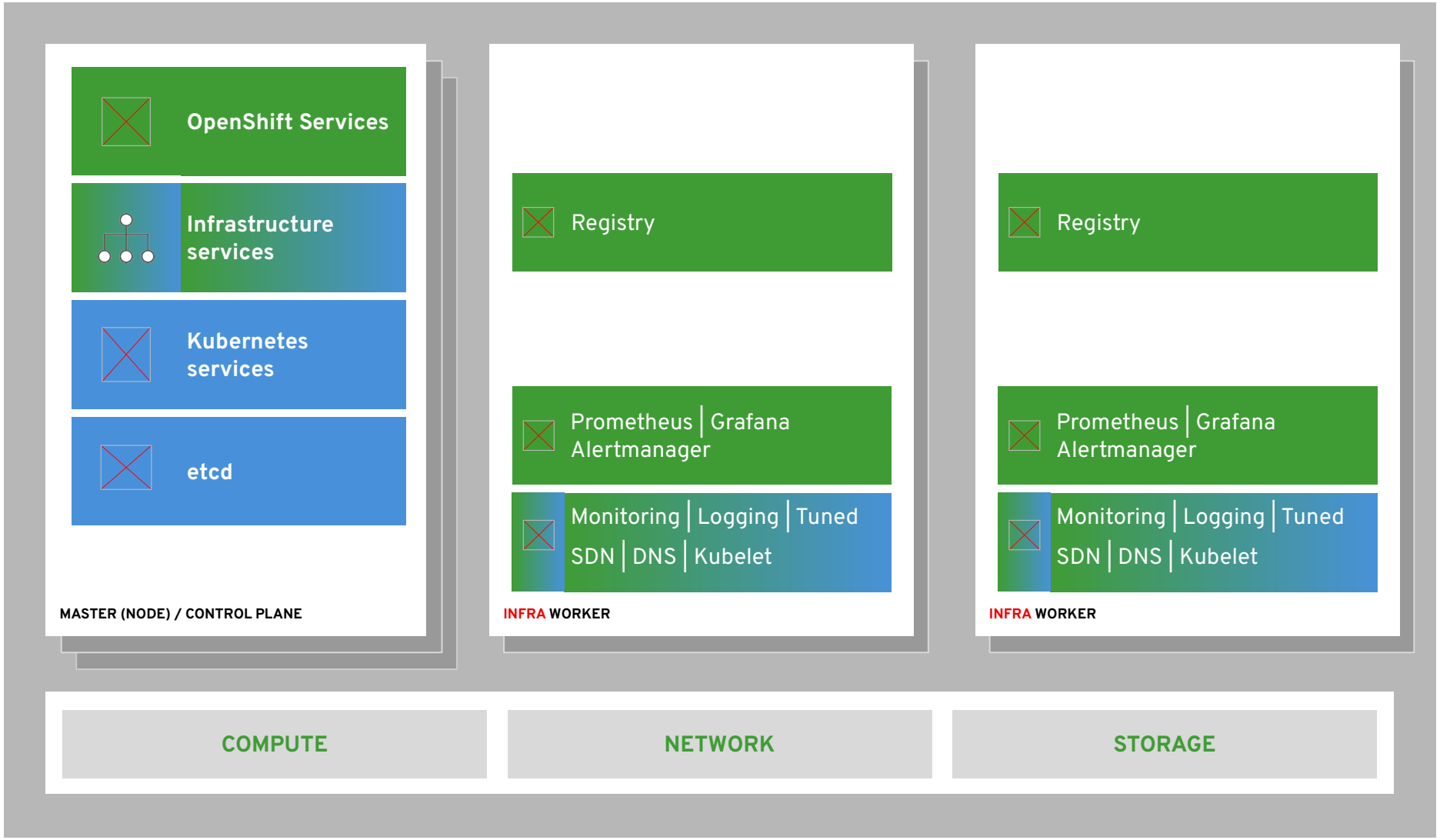
run on all hosts



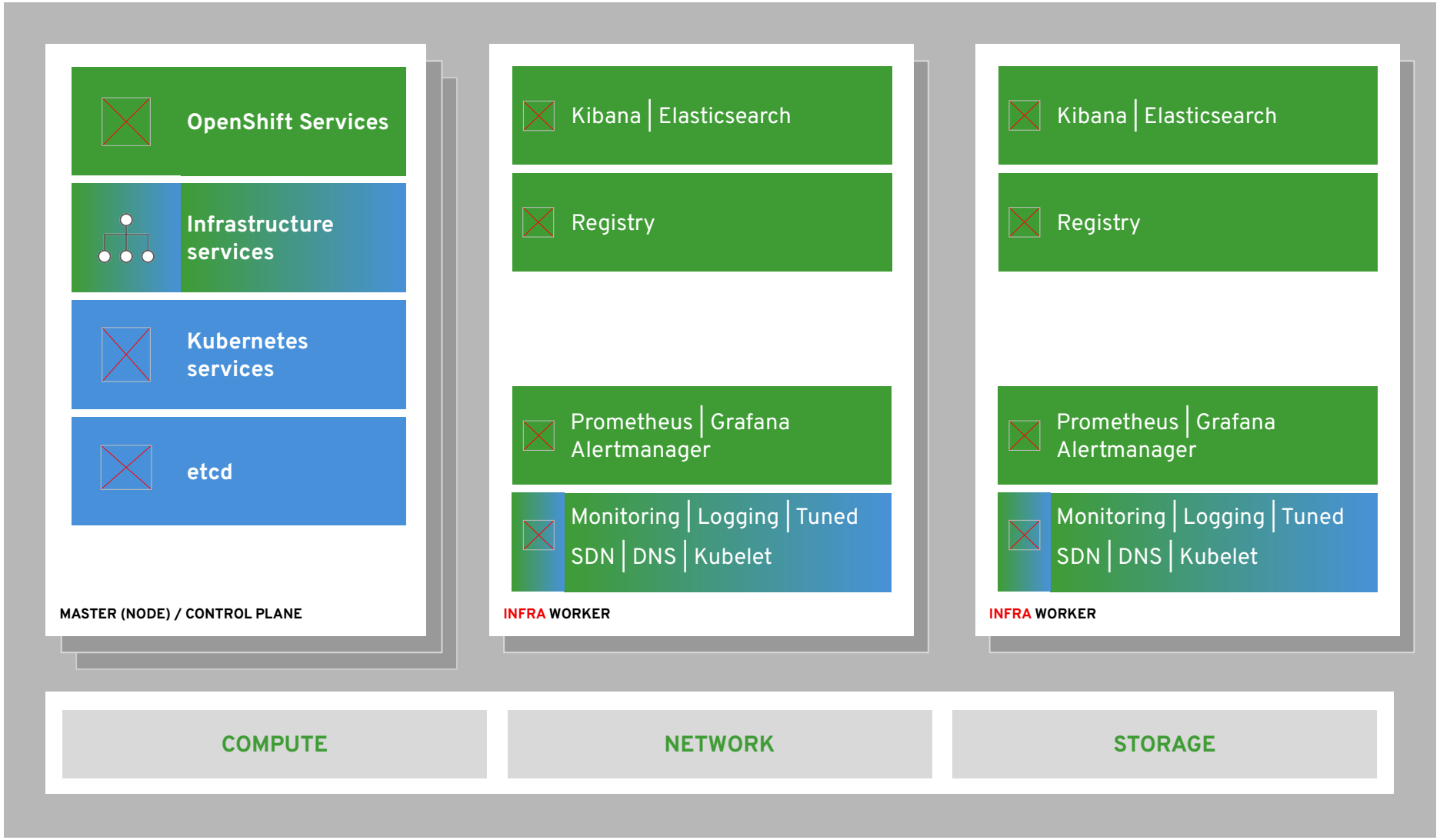
integrated image registry



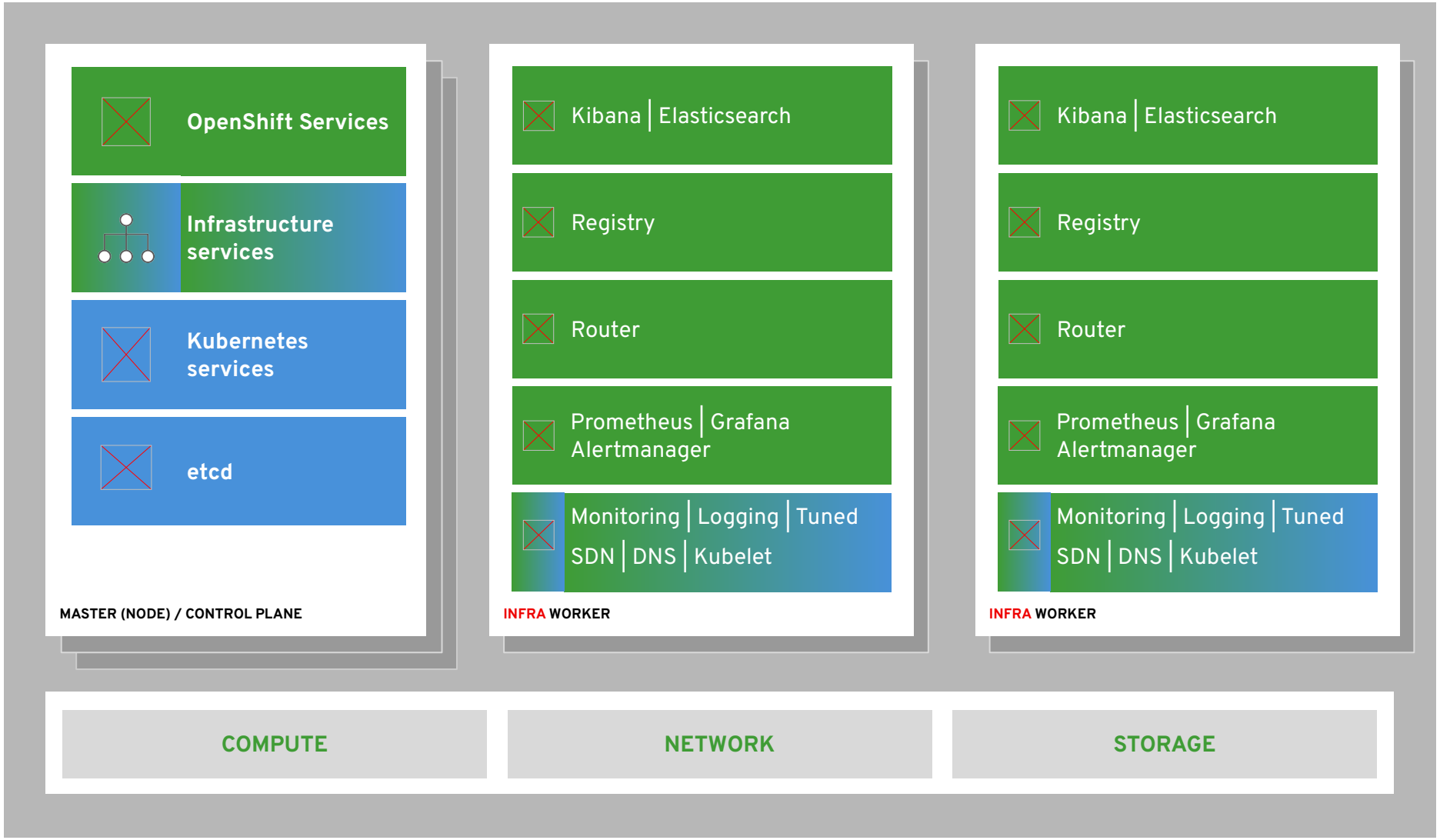
cluster monitoring



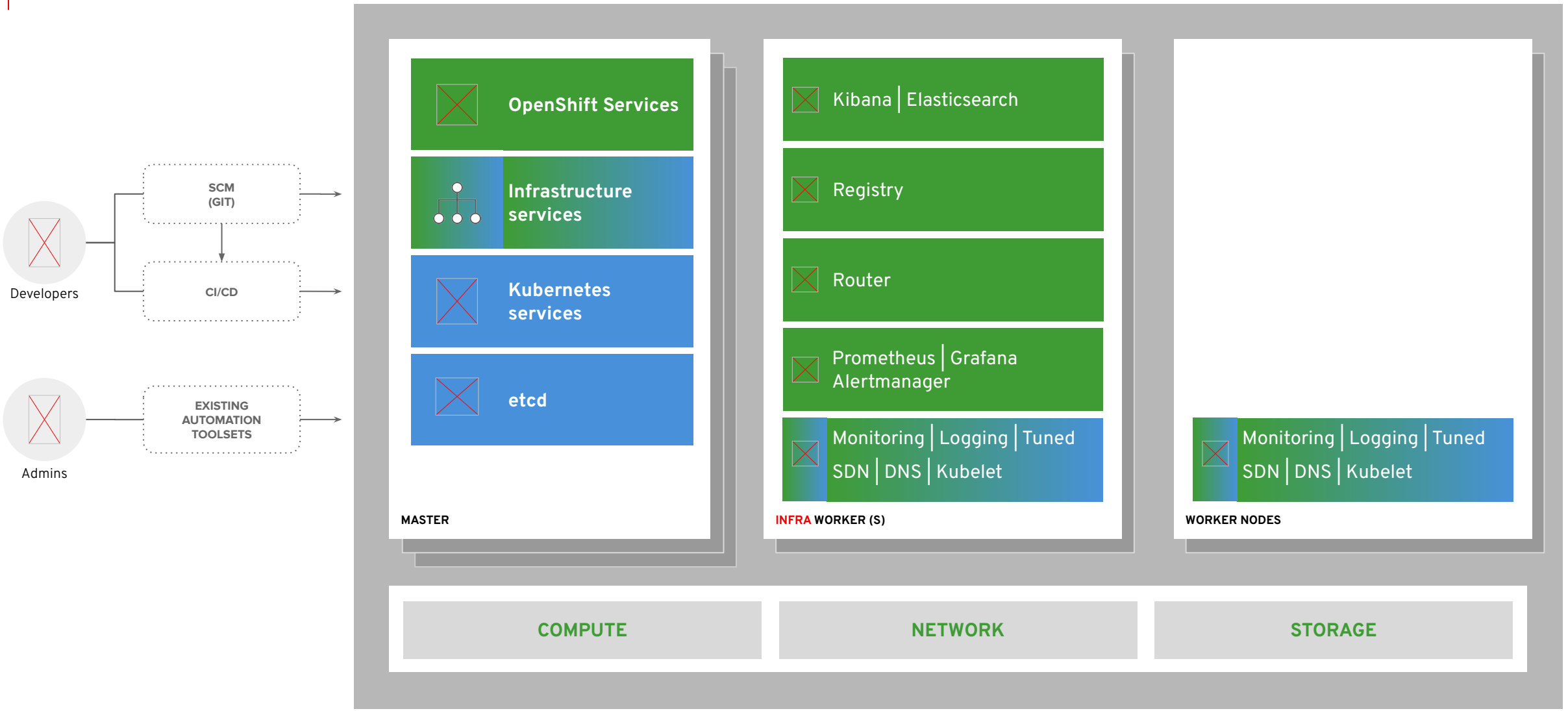
log aggregation



integrated routing



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



Quiz: Describing the Architecture of OpenShift

https://rol.redhat.com/rol/app/courses/do280-4.10/pages/ch01

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

5. Which of the following sentences is true regarding support for storage with OpenShift?

A ○ Users can only store persistent data in the etcd database.

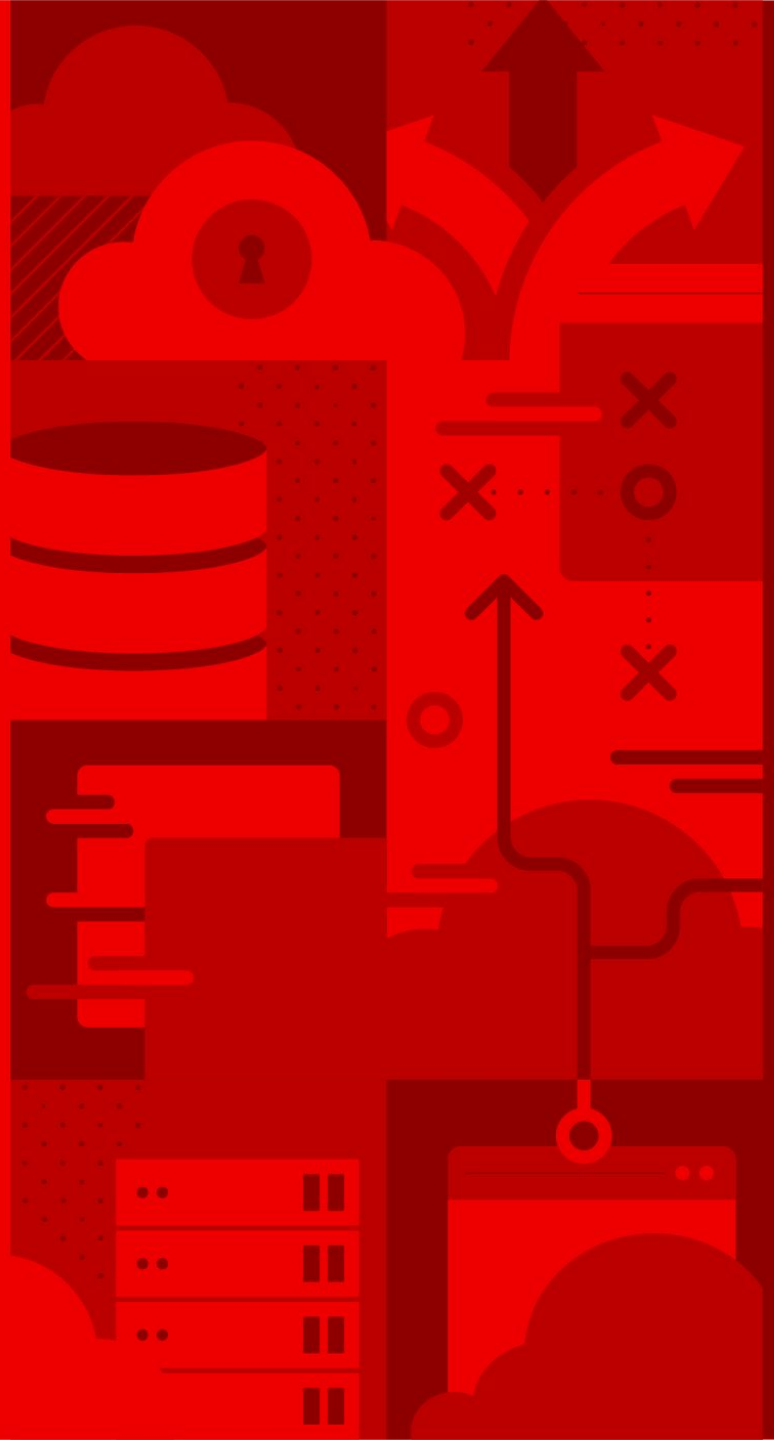
B ○ Users can only deploy on OpenShift cloud-native applications that conform to the Twelve-Factor App methodology.

C ○ Administrators must configure storage plug-ins appropriate for their cloud providers.

D ○ Administrators must define persistent volumes before any user can deploy applications that require persistent storage.

E ○ Users can deploy applications that require persistent storage by relying on the default storage class.

CHECK RESET SHOW SOLUTION



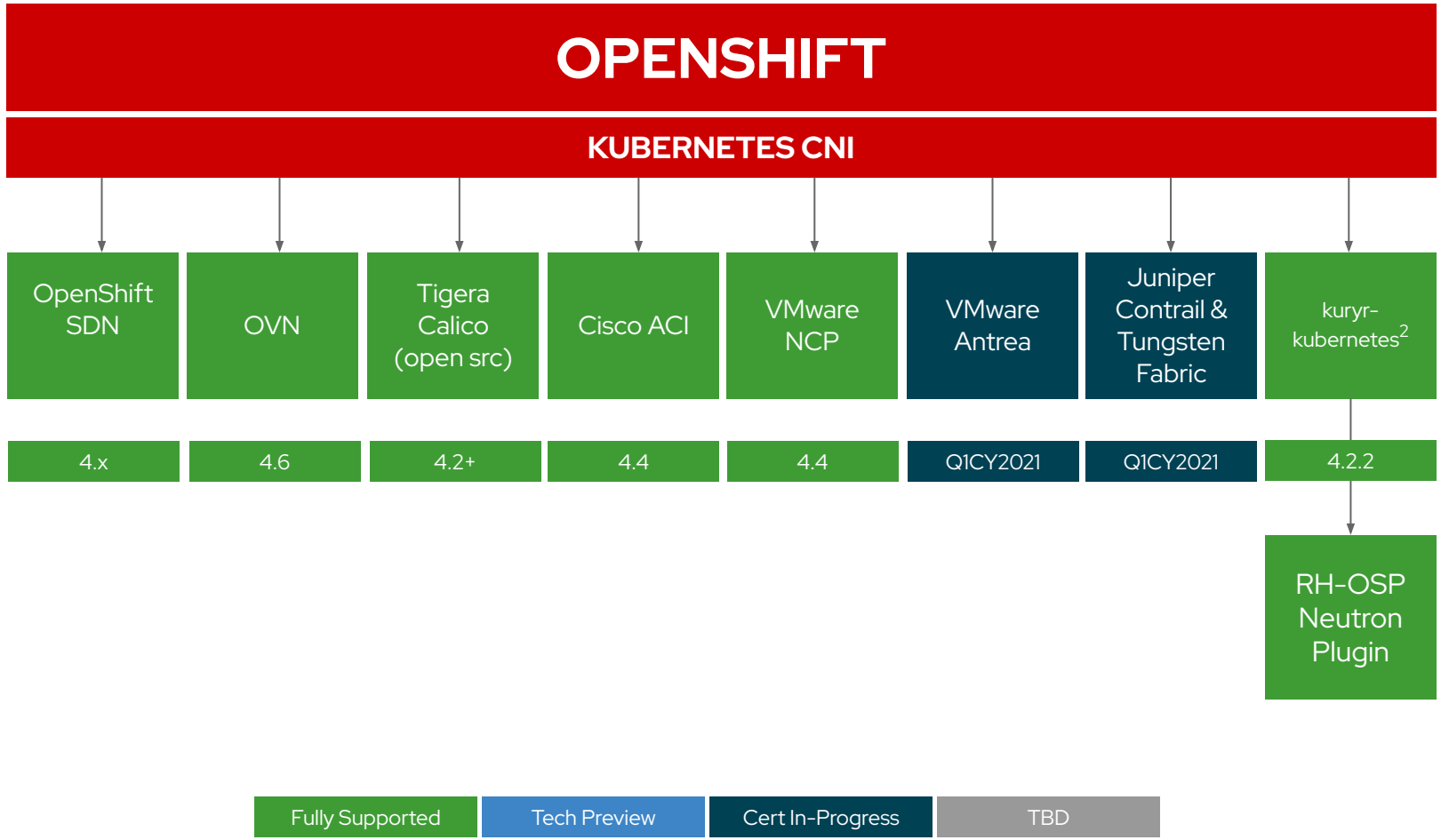
Networking

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



Red Hat

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

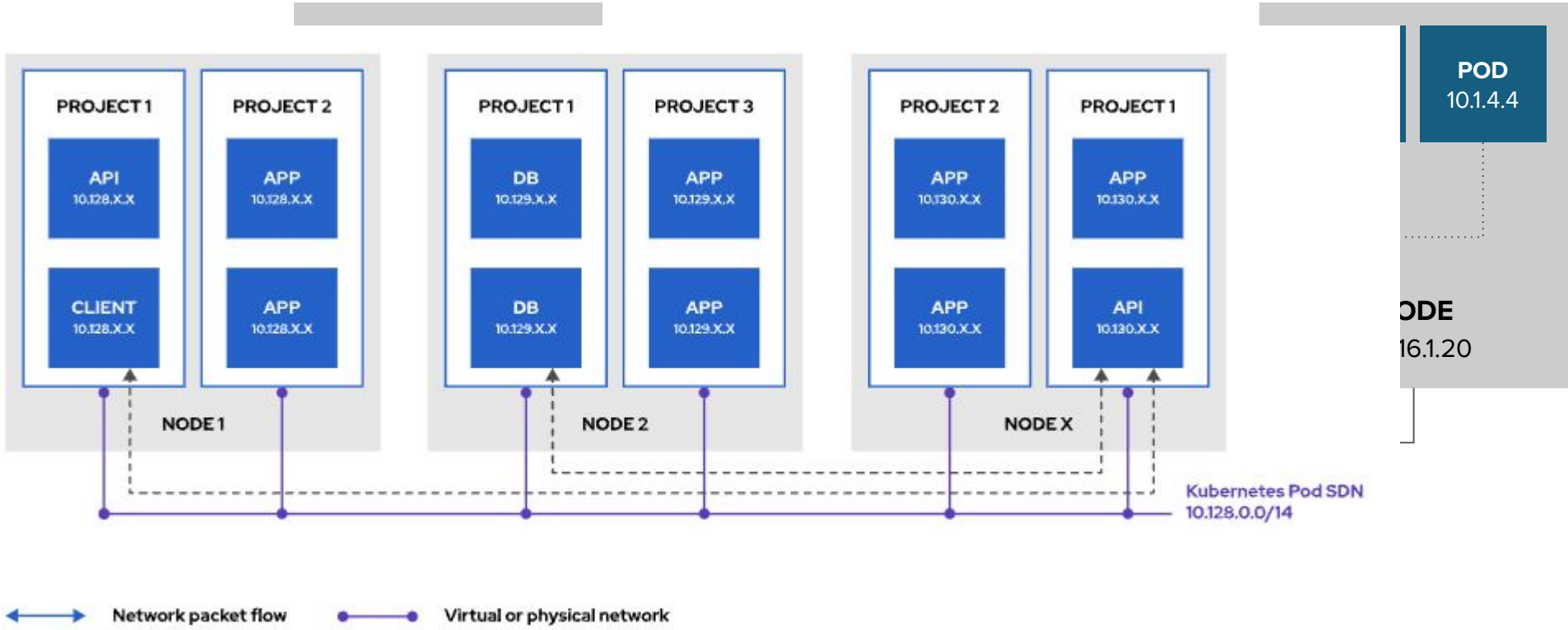


Figure 5.1: Kubernetes basic networking

Using Services for Accessing Pods

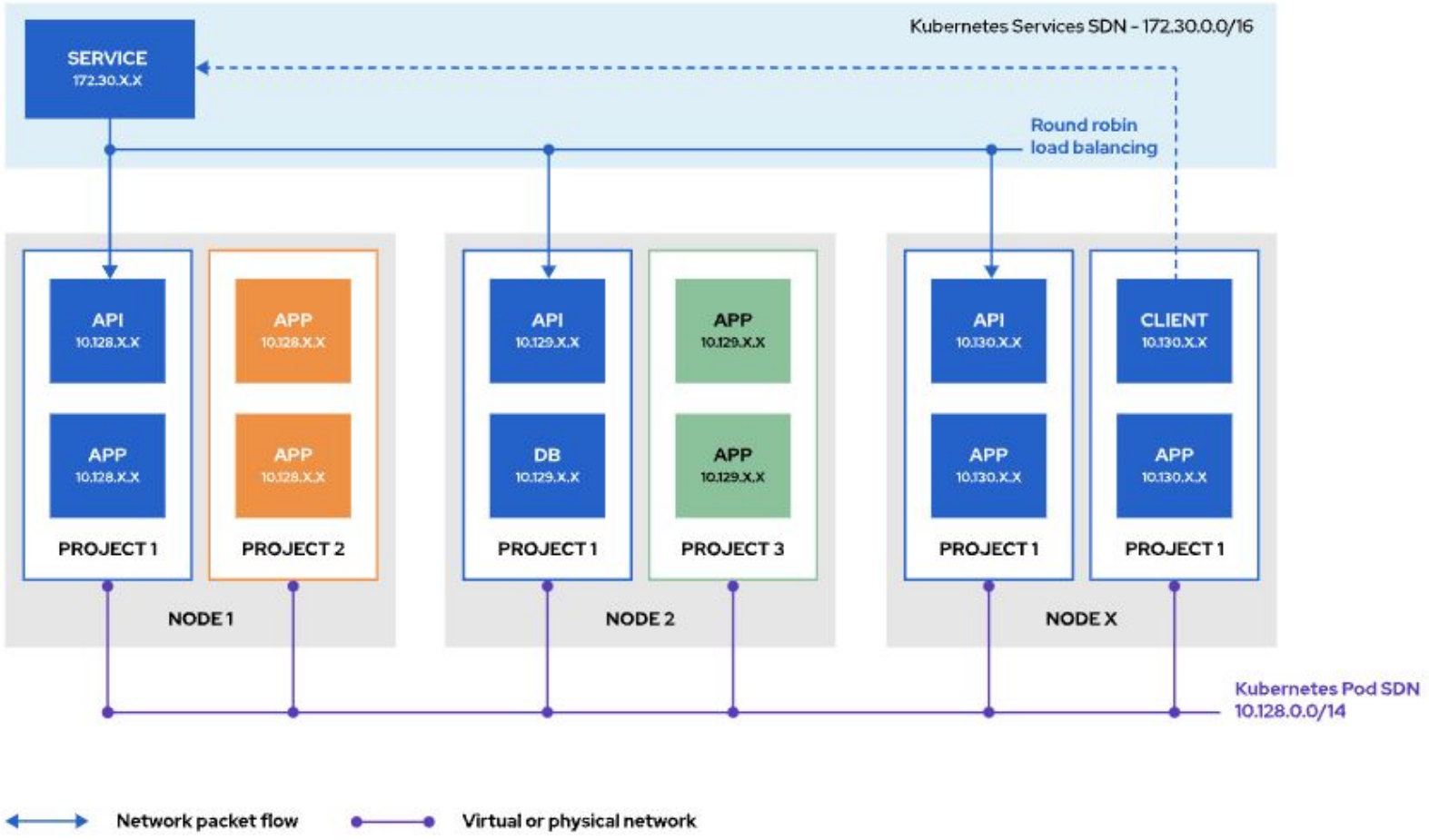
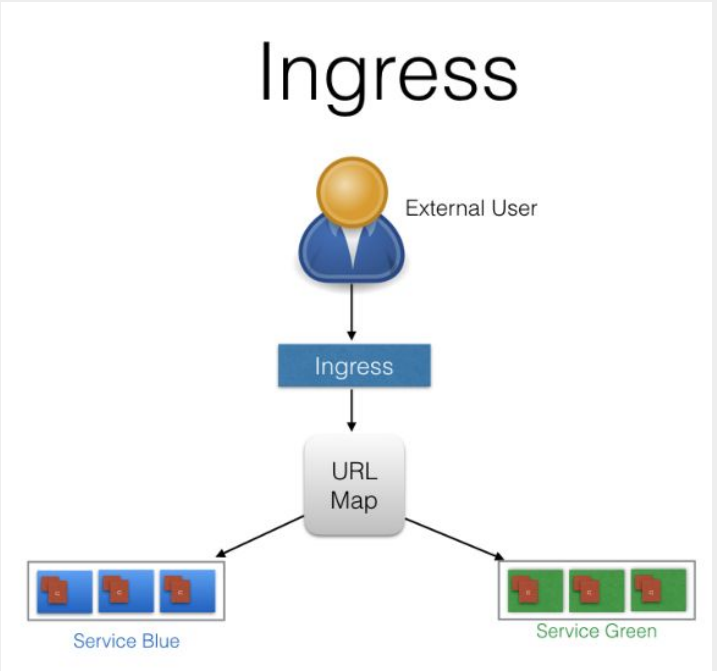


Figure 5.2: Using services for accessing applications

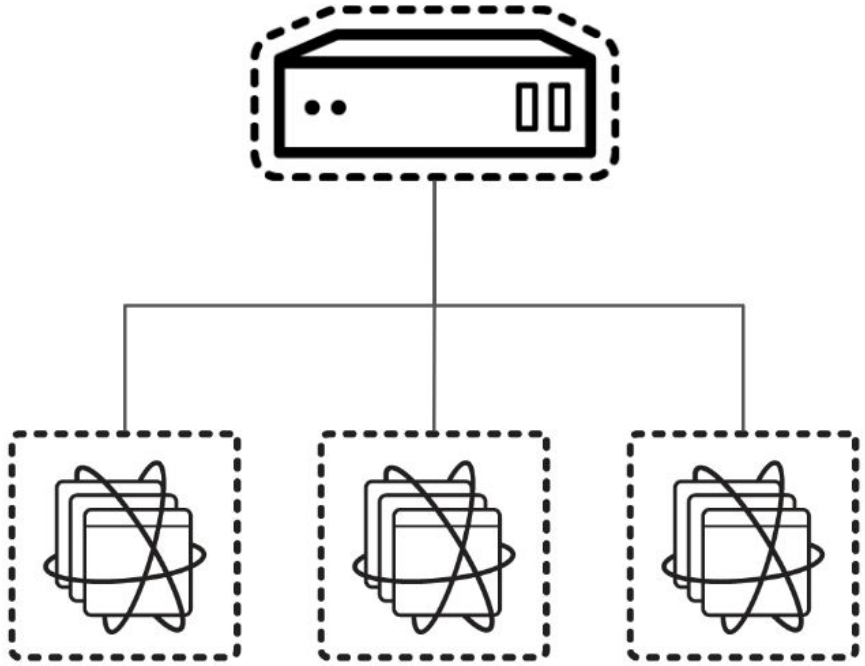
routes and ingress

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



Exposing Applications for External Access

Guided Exercise: Exposing Applications for External Access

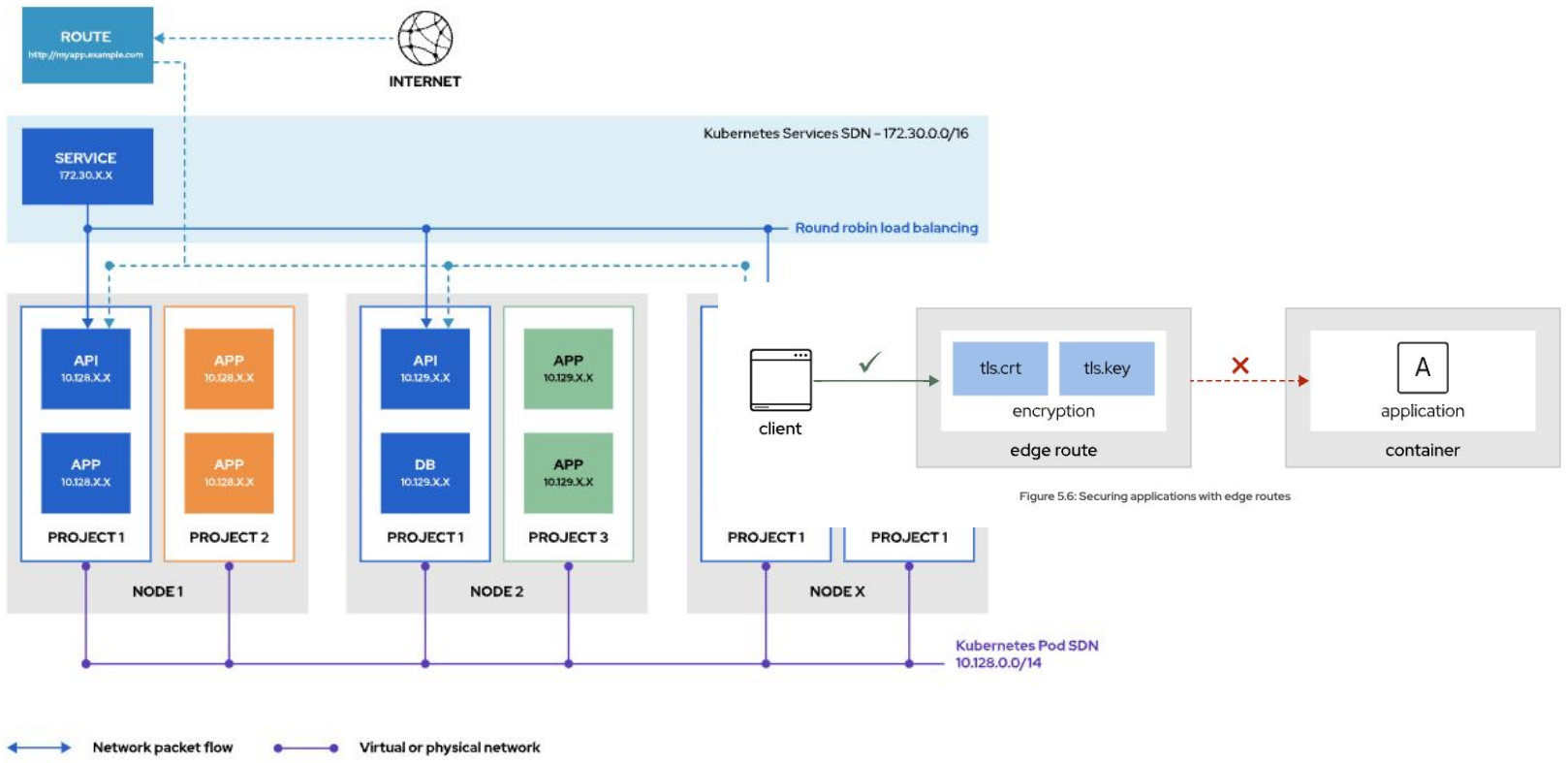


Figure 5.6: Securing applications with edge routes

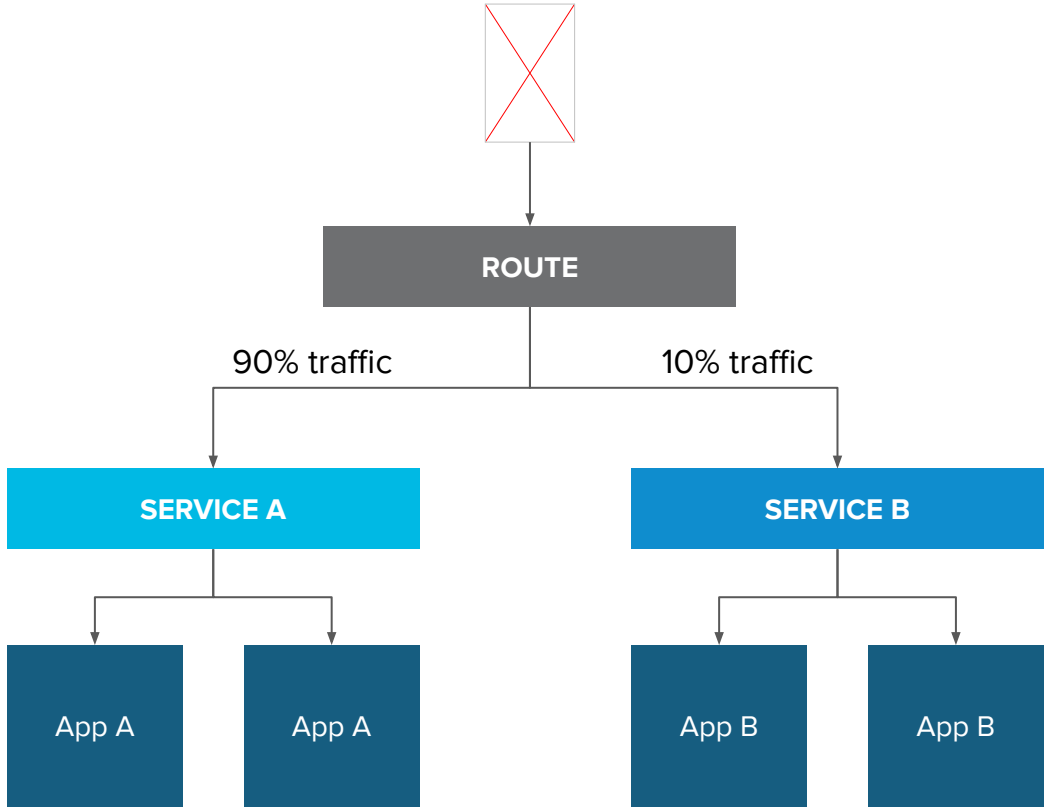
Figure 5.5: Using routes to expose applications

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

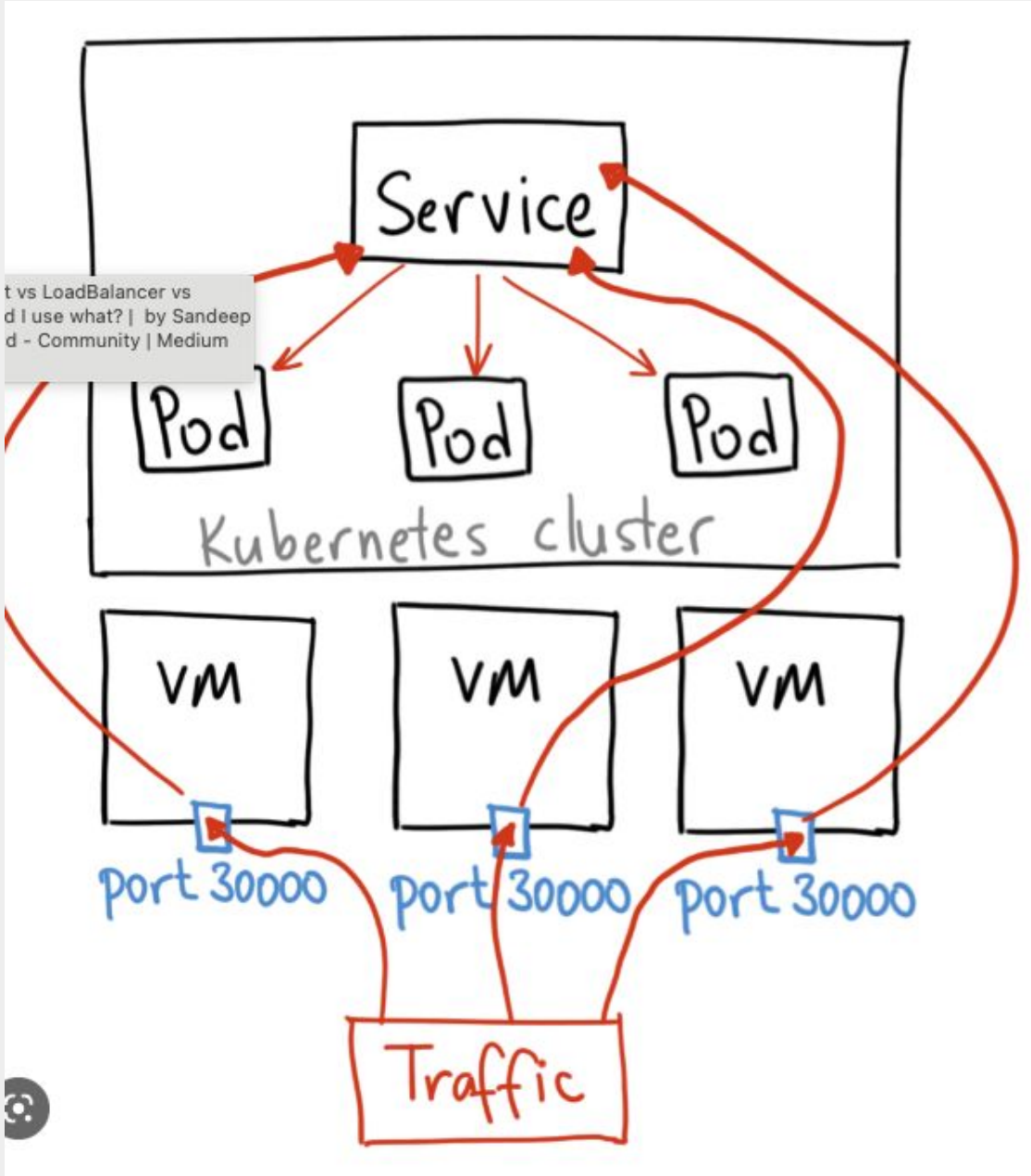
Router-based deployment methodologies

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



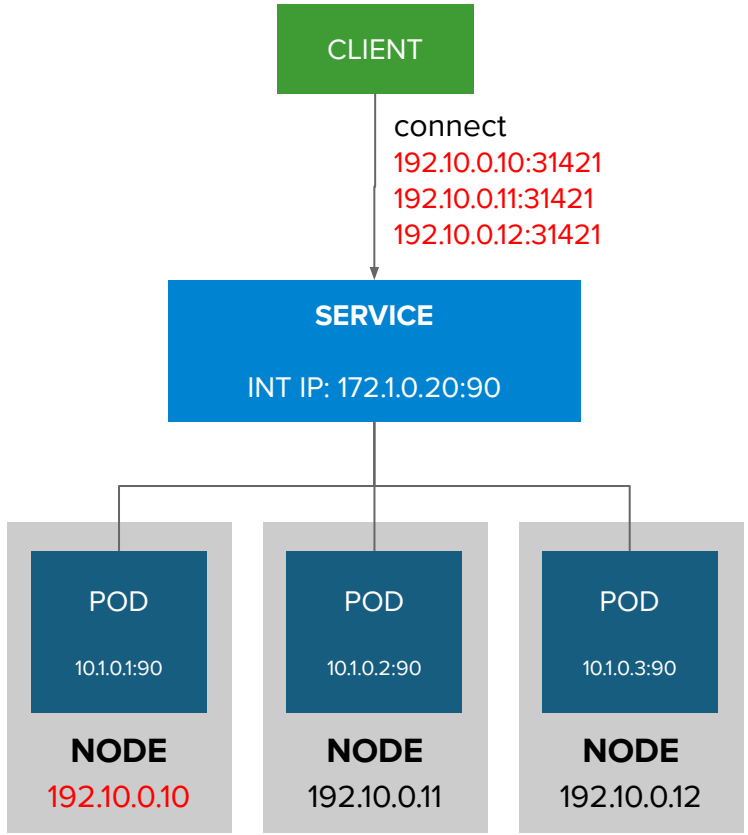
Alternative methods for ingress

Different ways that traffic can enter the cluster without the router



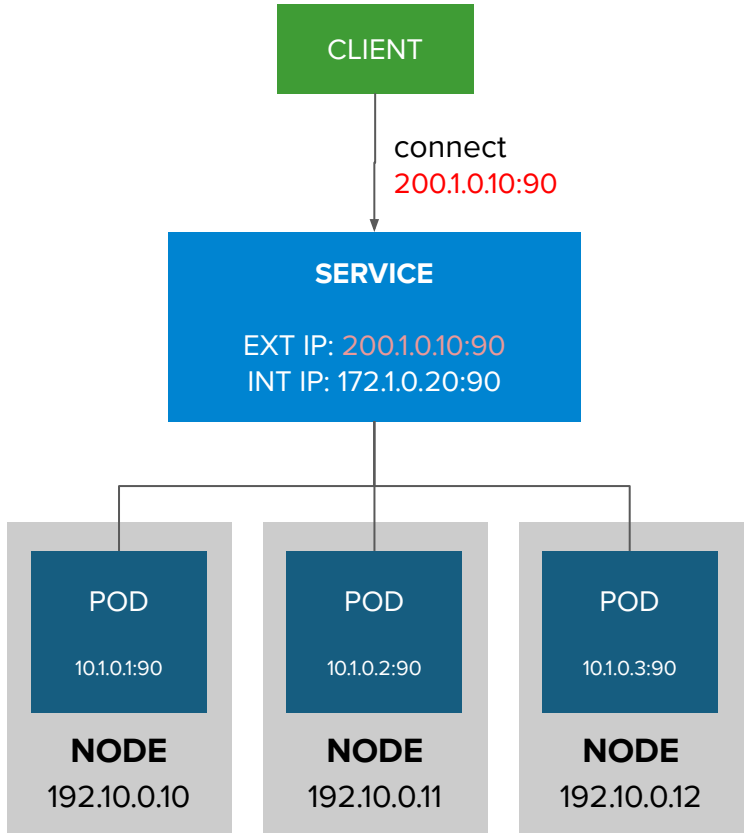
Entering the cluster on a random port with service nodeports

- 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)



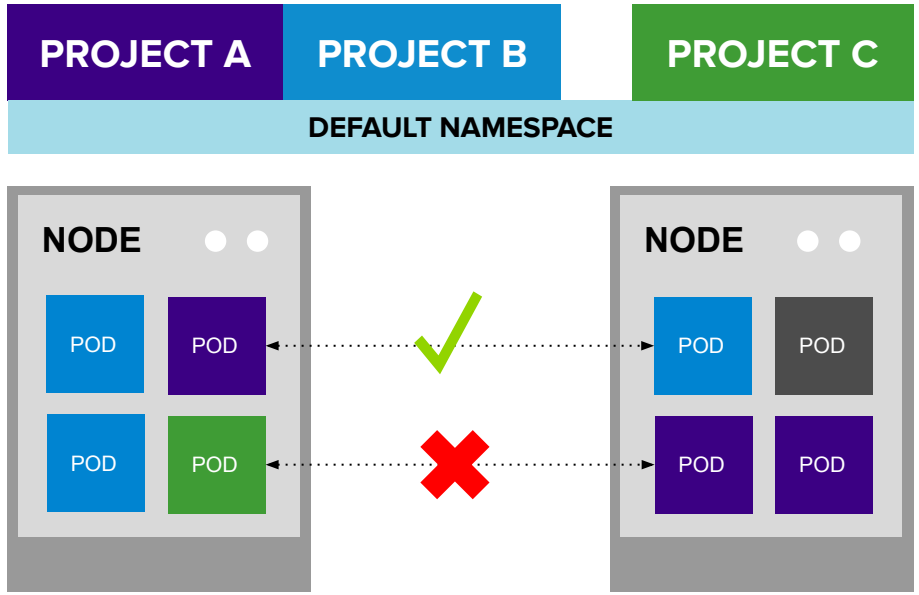
OpenShift SDN “flavors”

OPEN NETWORK (Default)

- All pods can communicate with each other across projects

MULTI-TENANT NETWORK

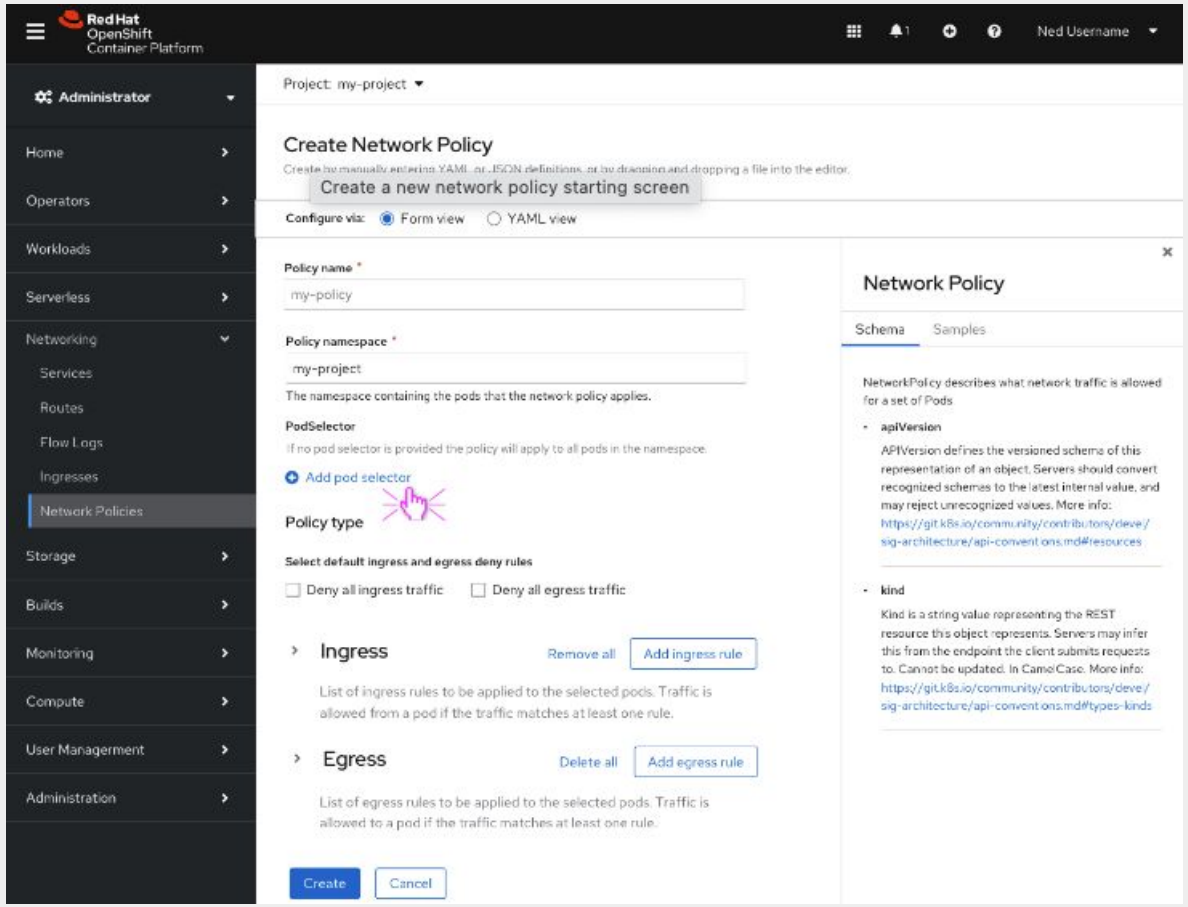
- Project-level network isolation
- Multicast support
- Egress network policies



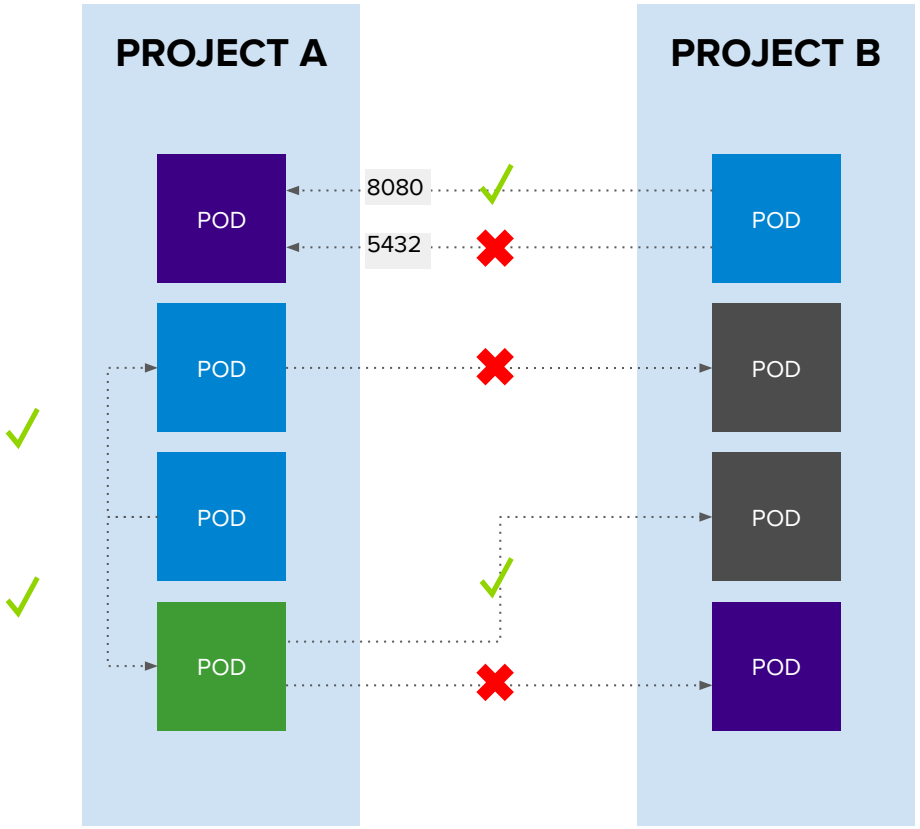
Multi-Tenant Network

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.



NetworkPolicy



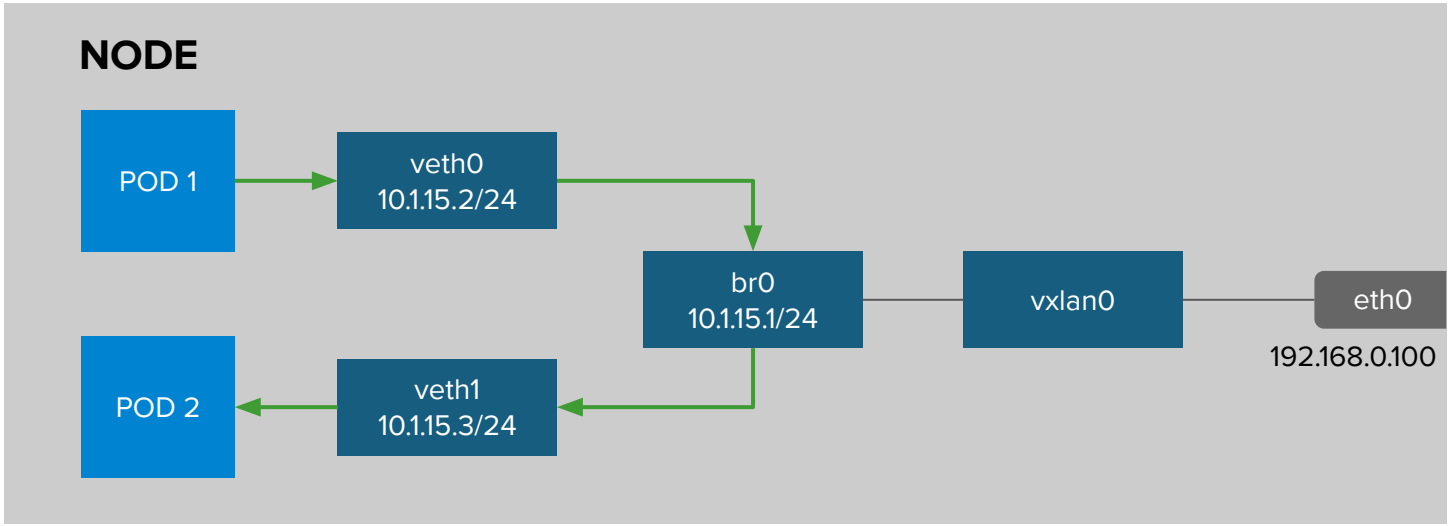
Example Policies

- 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
```

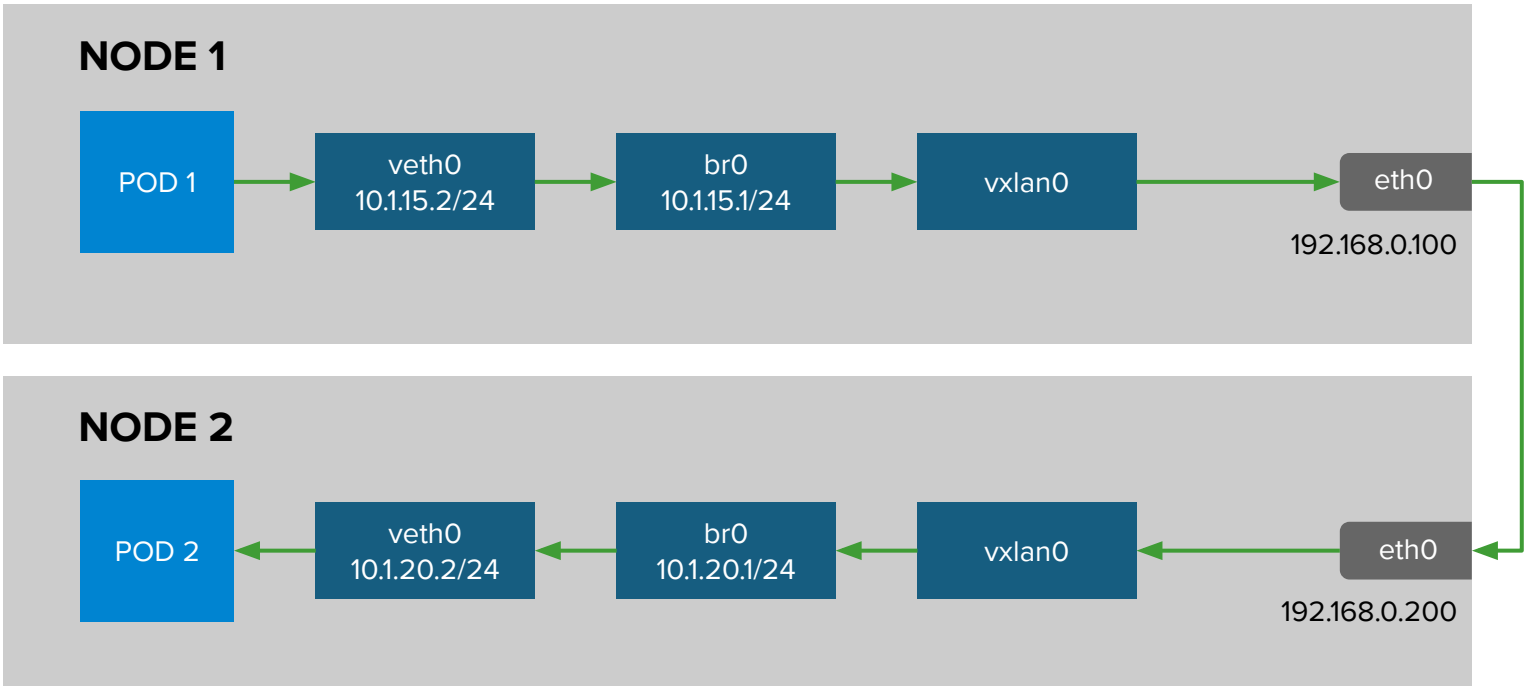

OpenShift SDN packet flows

container-container on same host



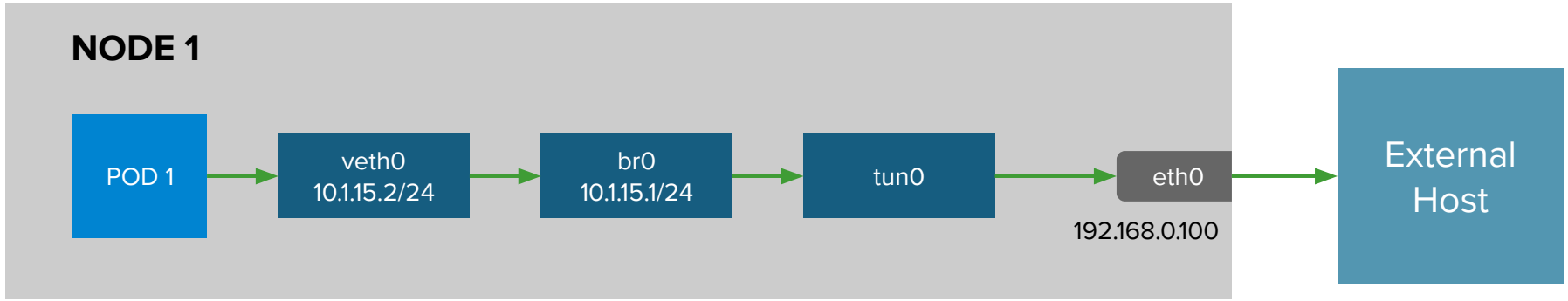
OpenShift SDN packet flows

container-container across hosts



OpenShift SDN packet flows

container leaving the host

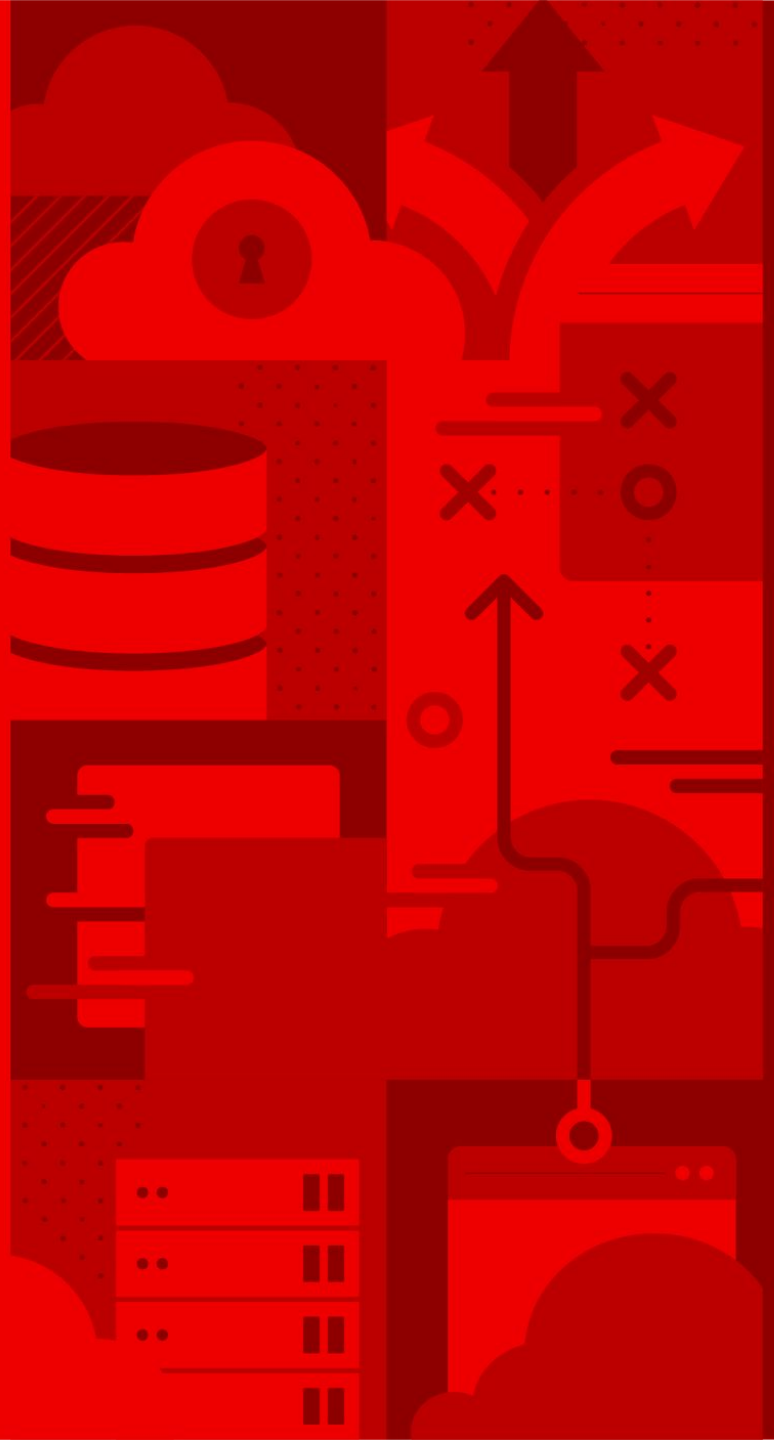


Cluster DNS

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



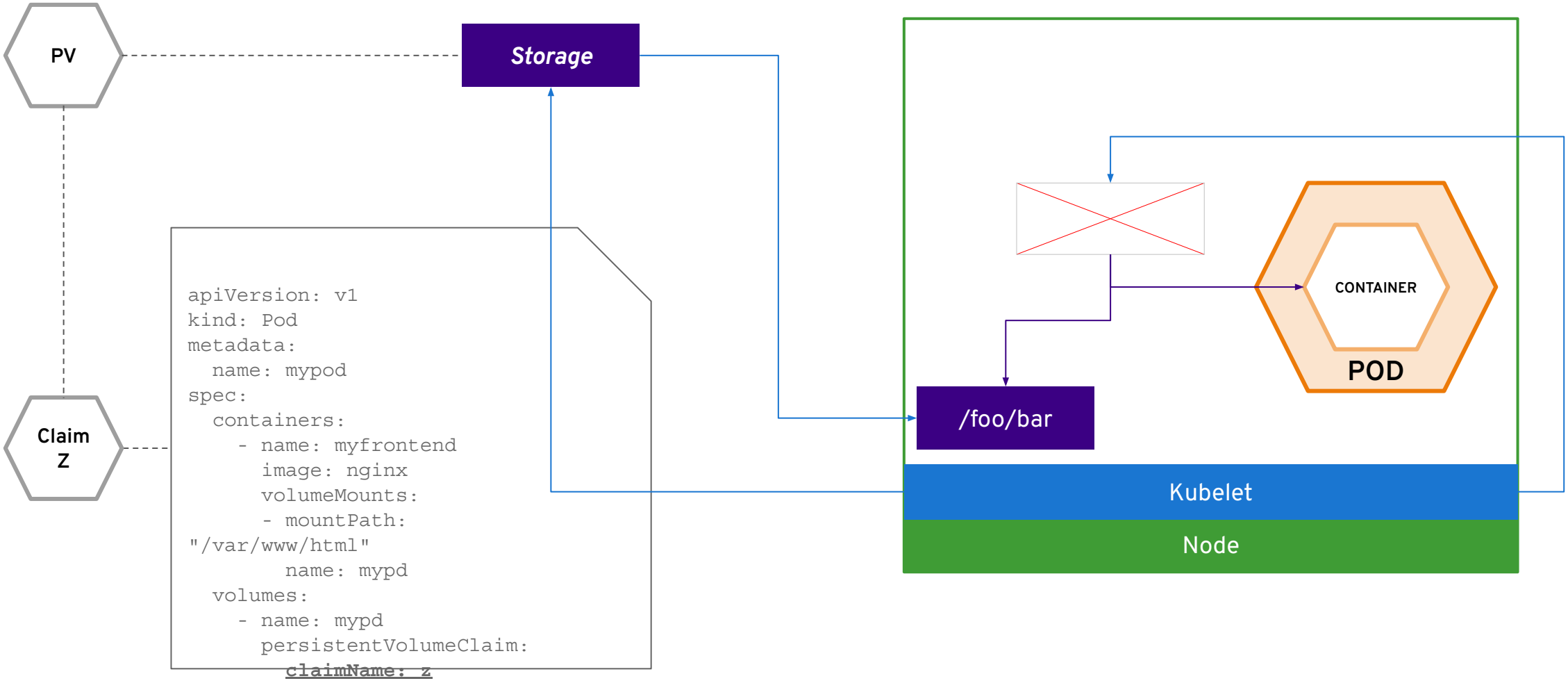
Persistent Storage

A broad spectrum of static and dynamic storage endpoints

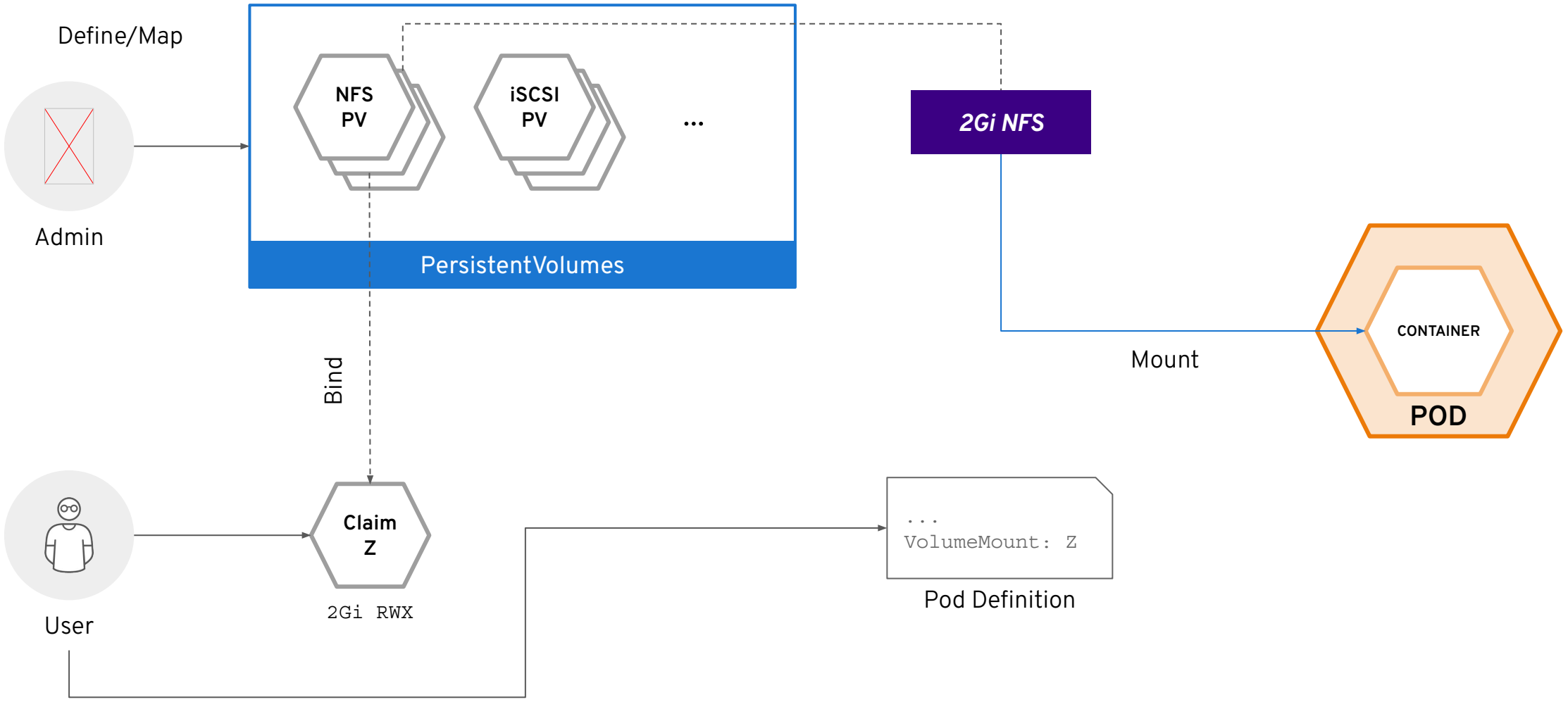
NFS	OpenStack Cinder	iSCSI	Azure Disk	AWS EBS	FlexVolume
Raw Block	Local Volumes	Fiber Channel	Azure File	GCE Persistent Disk	VMware vSphere VMDK

Container Storage Interface (CSI)

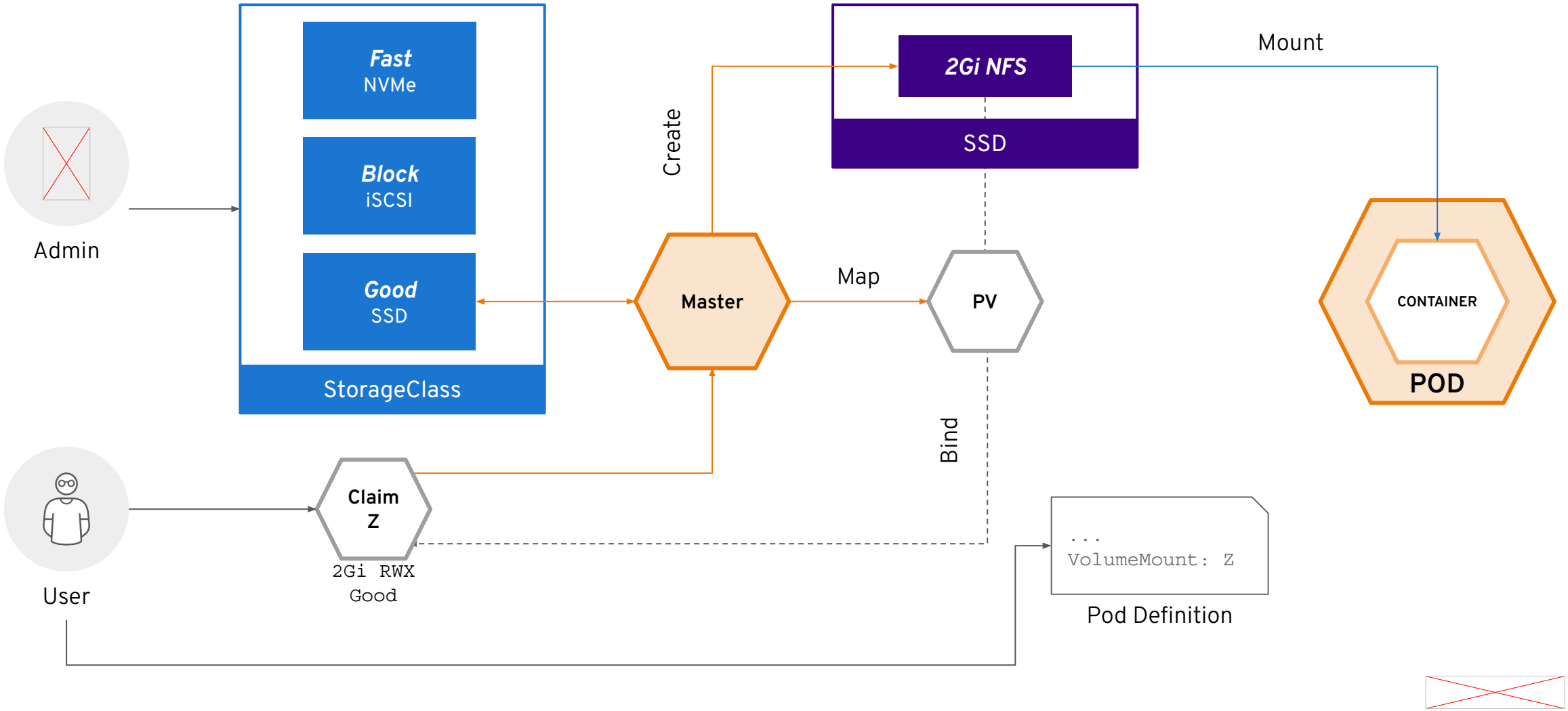
PV Consumption

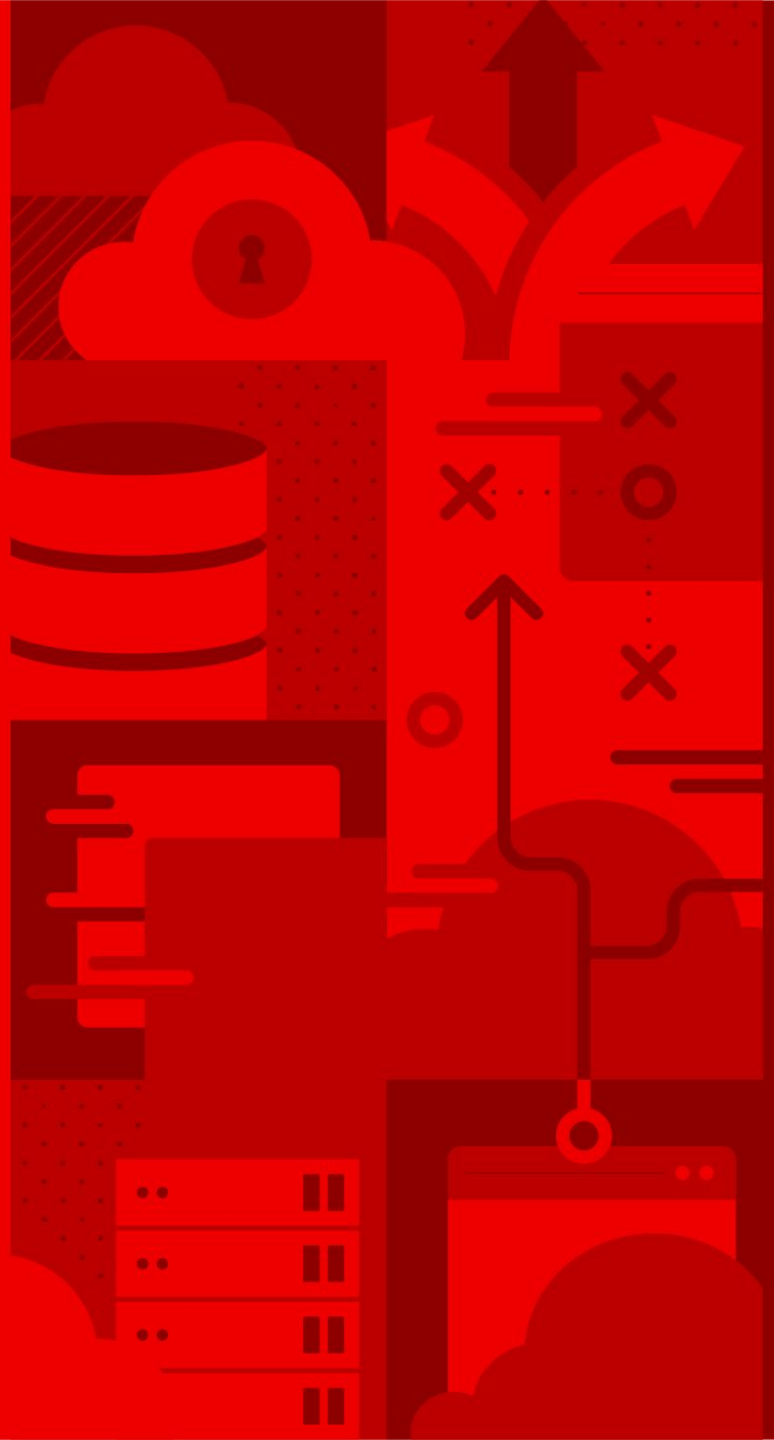


Static Storage Provisioning



Dynamic Storage Provisioning





Observability

OpenShift Observability: Five Pillars

Store:
Metrics with *Prometheus/Thanos*
Logs with *Loki*
Traces with *Jaeger/Elasticsearch*

2 Data Storage

1 Data Collection

Collect:
Metrics with *Prometheus*
Logs with *Vector*
Traces with *OpenTelemetry*

Observability

"Turn your data into answers!"

Third Party Integration

3 Data Delivery

Deliver:
Aggregate & Normalize data
Transport it with *Observability Operator*

5 Data Analytics

Analyze:
Query metrics
Search metrics targets
Filter logs by severity

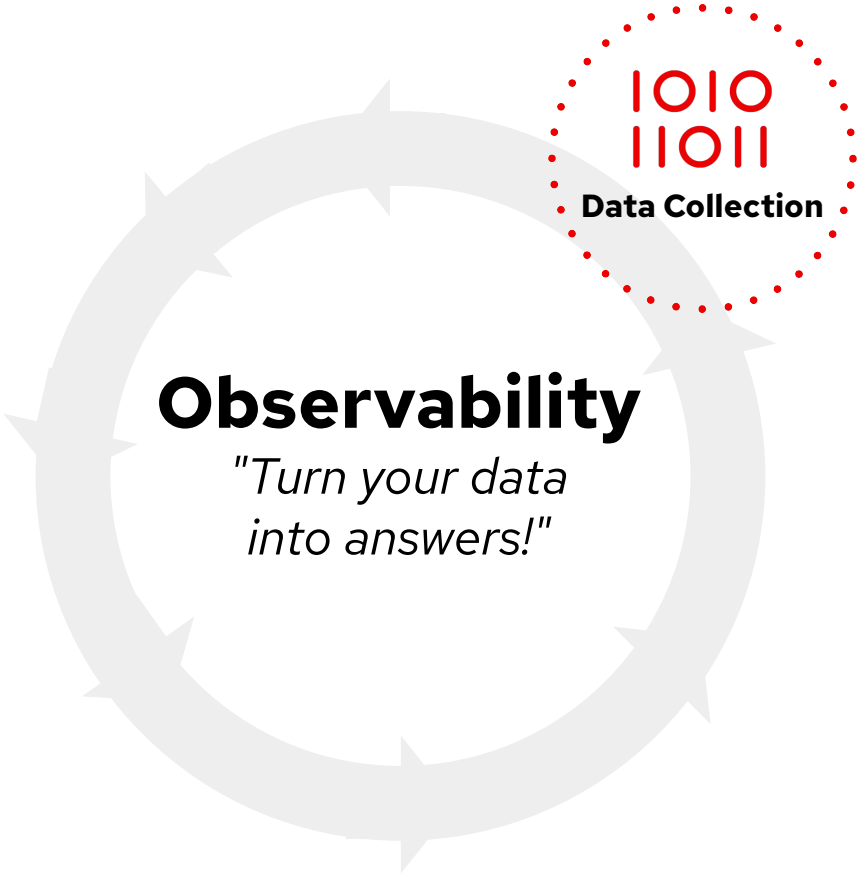
4 Data Visualization

Visualize:
Out of the box experience
& full support in *OpenShift Web Console*

PMs: Roger Floren, Jamie Parker & Vanessa Martini

Red Hat

OpenShift Observability



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

PMs: Roger Floren, Jamie Parker & Vanessa Martini

Red Hat

OpenShift Observability

Observability

"Turn your data into answers!"

OpenShift 4.12 Monitoring

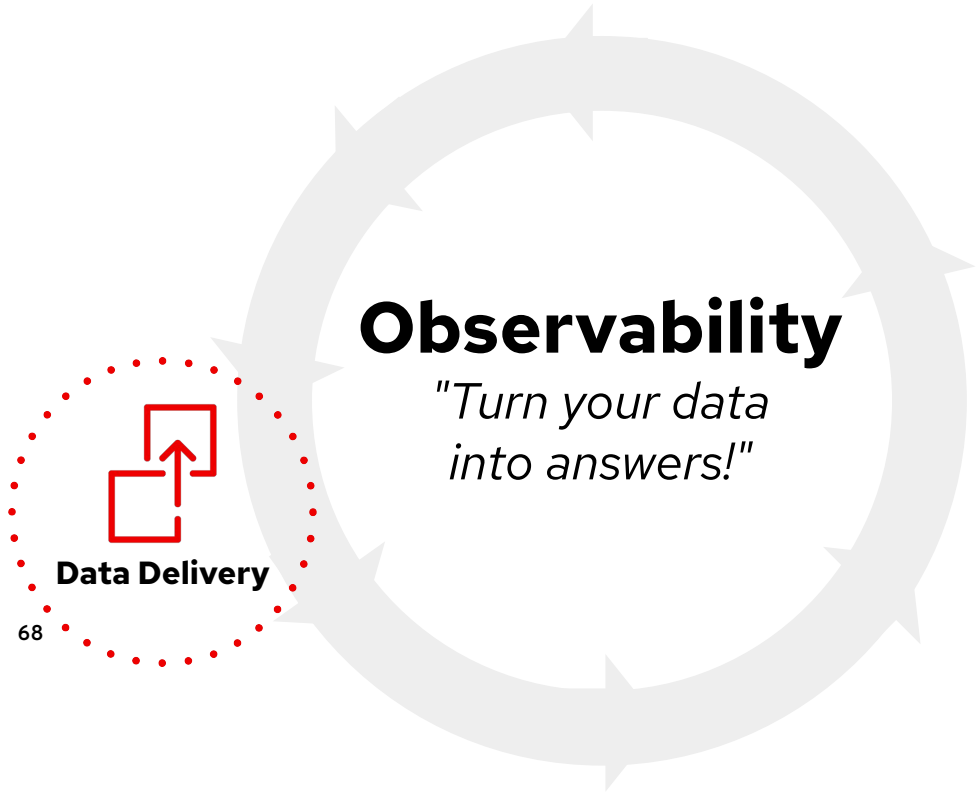
- Version updates to Monitoring stack components & dependencies

Logging 5.6

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

PMs: Roger Floren, Jamie Parker & Vanessa Martini

Red Hat



OpenShift 4.12 Monitoring

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

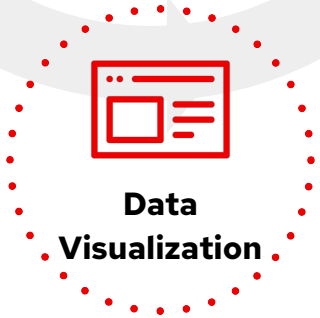
Logging 5.6

- Support for forwarding logs to Splunk

PMs: Roger Floren, Jamie Parker & Vanessa Martini

Observability

"Turn your data into answers!"



OpenShift 4.12 Monitoring

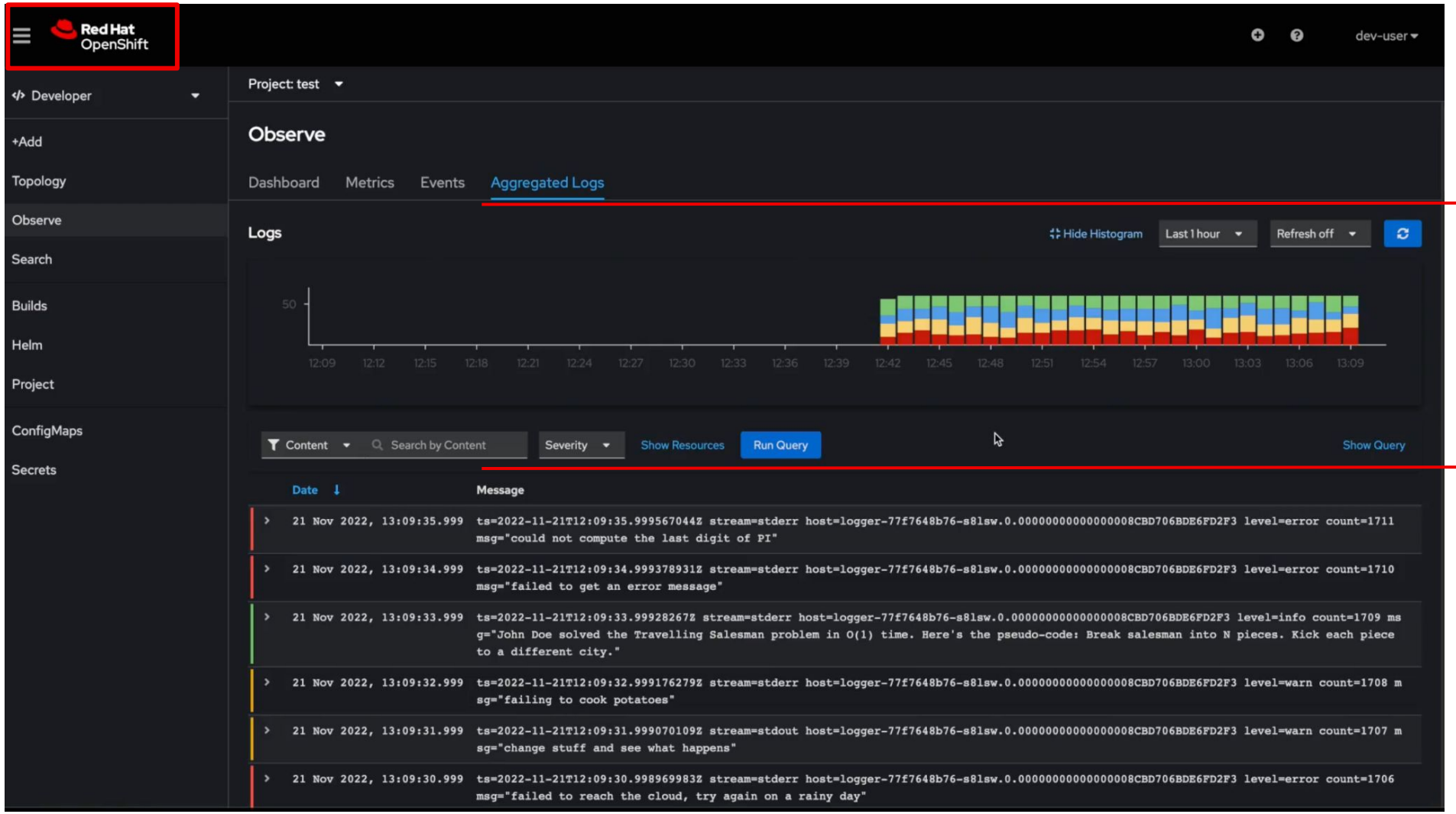
- Improved UX experience in OpenShift Web Console: Easier selection of records in Metrics UI
- Support for Alertmanager's negative matchers

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)

PMs: Roger Floren, Jamie Parker & Vanessa Martini

OpenShift Observability



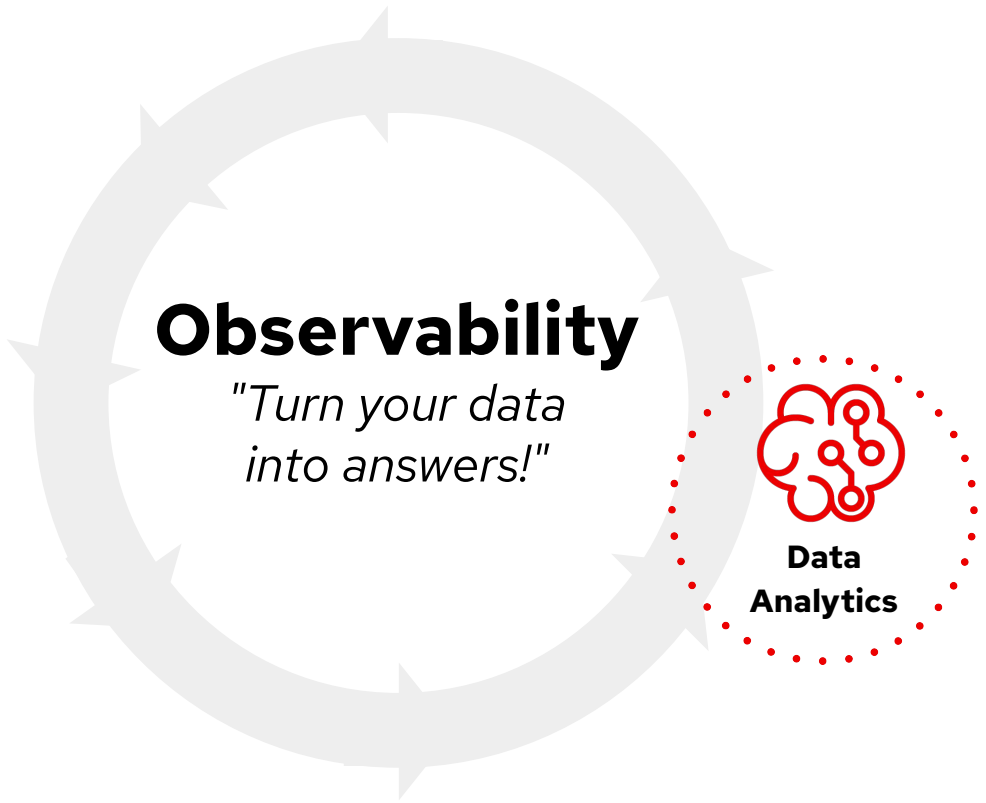
New entry: Aggregated Logs view in Developer Console

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

PMs: Roger Floren, Jamie Parker & Vanessa Martini

Red Hat

OpenShift Observability



OpenShift 4.12 Monitoring

- Runbooks URLs enabled in the Alerting UI of OpenShift Console

Logging 5.6

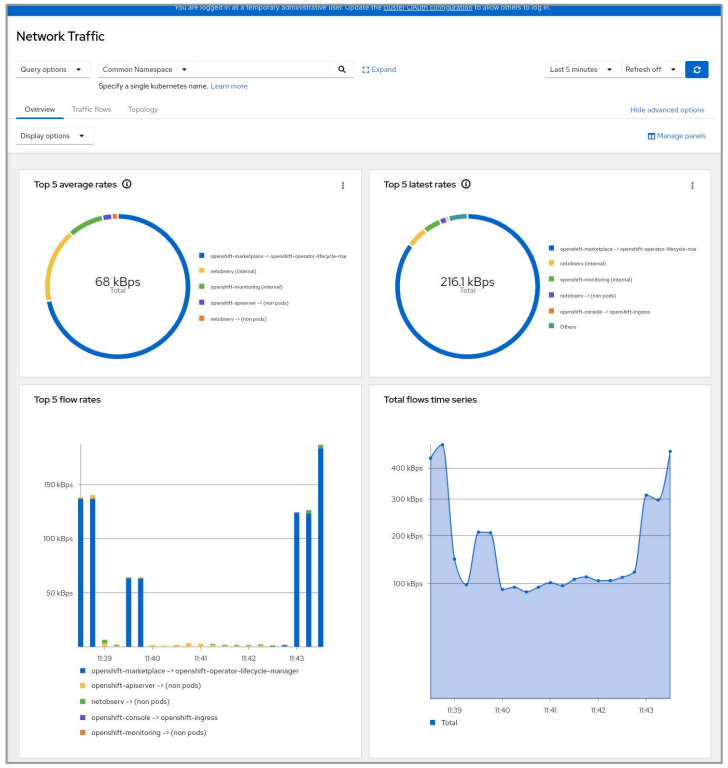
- Add the OpenShift cluster ID to log records so that clusters can be uniquely identified in aggregated logs

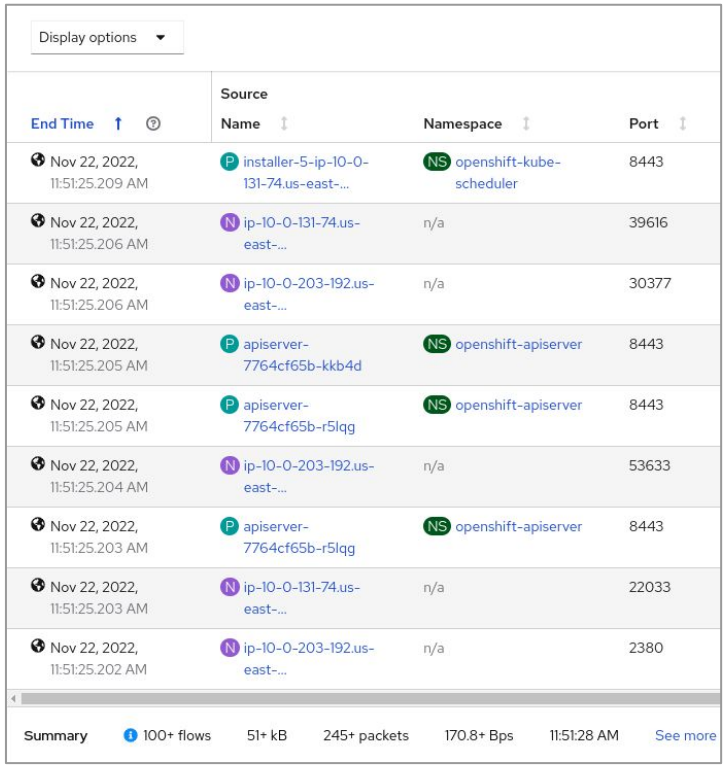
PMs: Roger Floren, Jamie Parker & Vanessa Martini

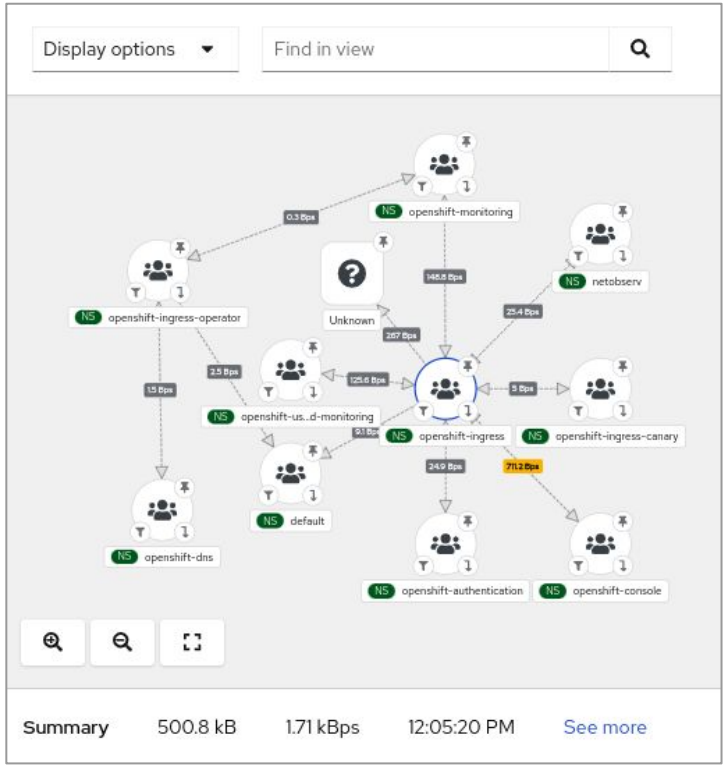
Network Observability

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





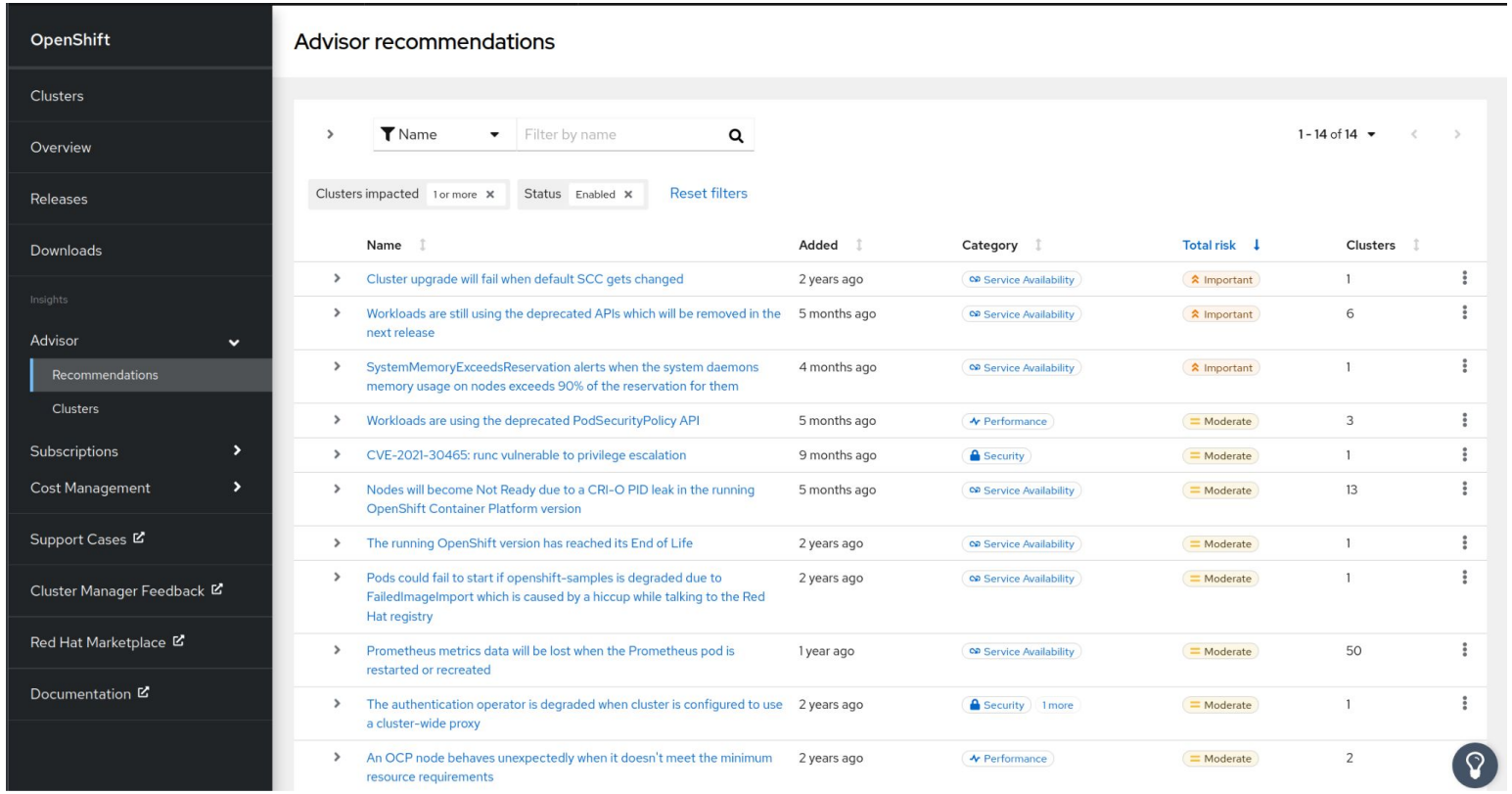


PM: Marc Curry, Deepthi Dharwar

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

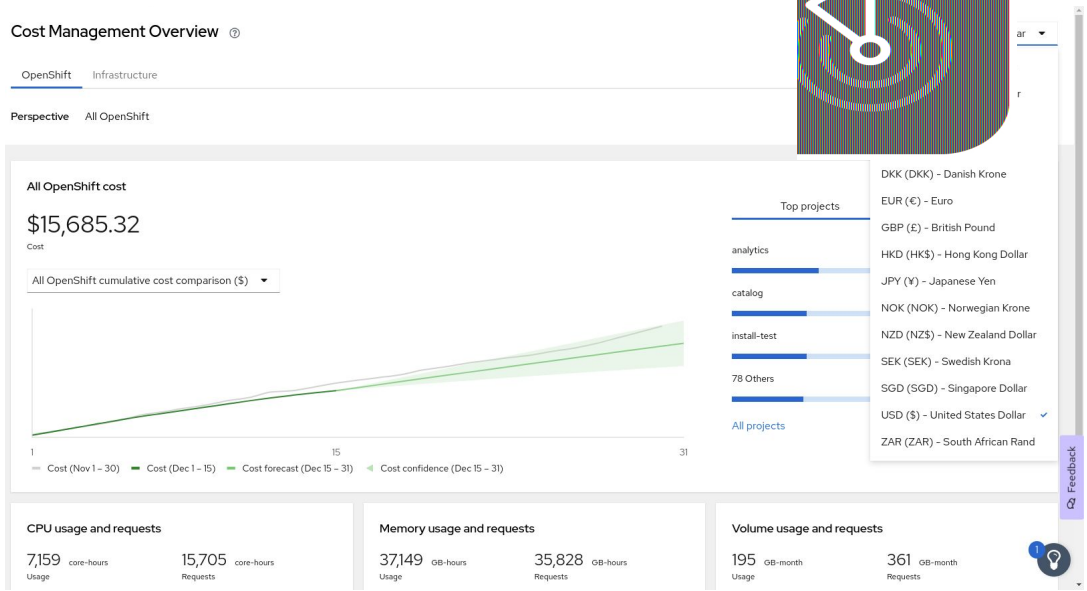
https://console.redhat.com/openshift/advisor
https://console.redhat.com/settings/notifications/openshift

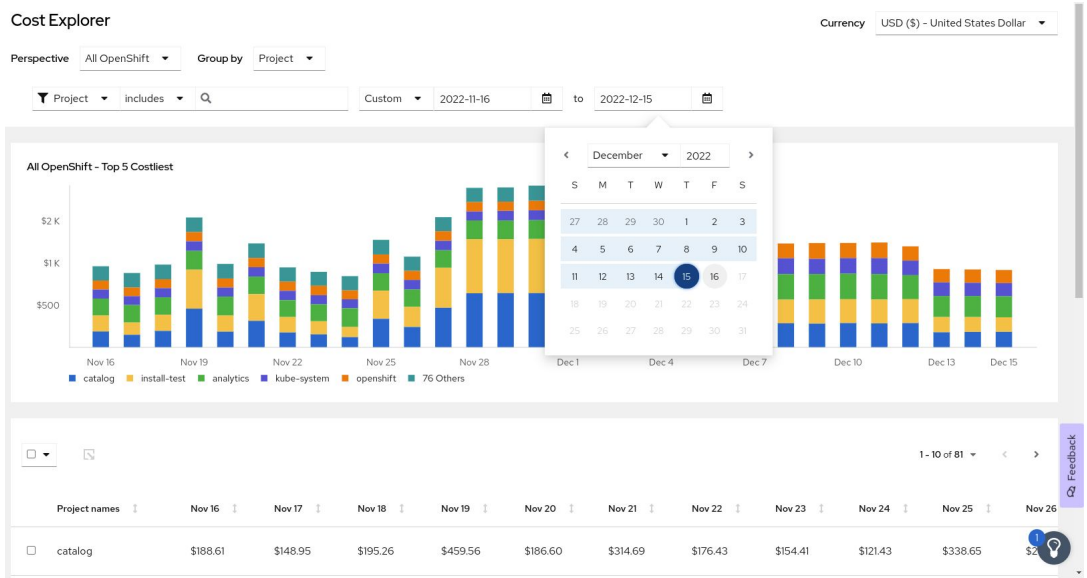


PM: Tomas Dosek

Insights Cost Management

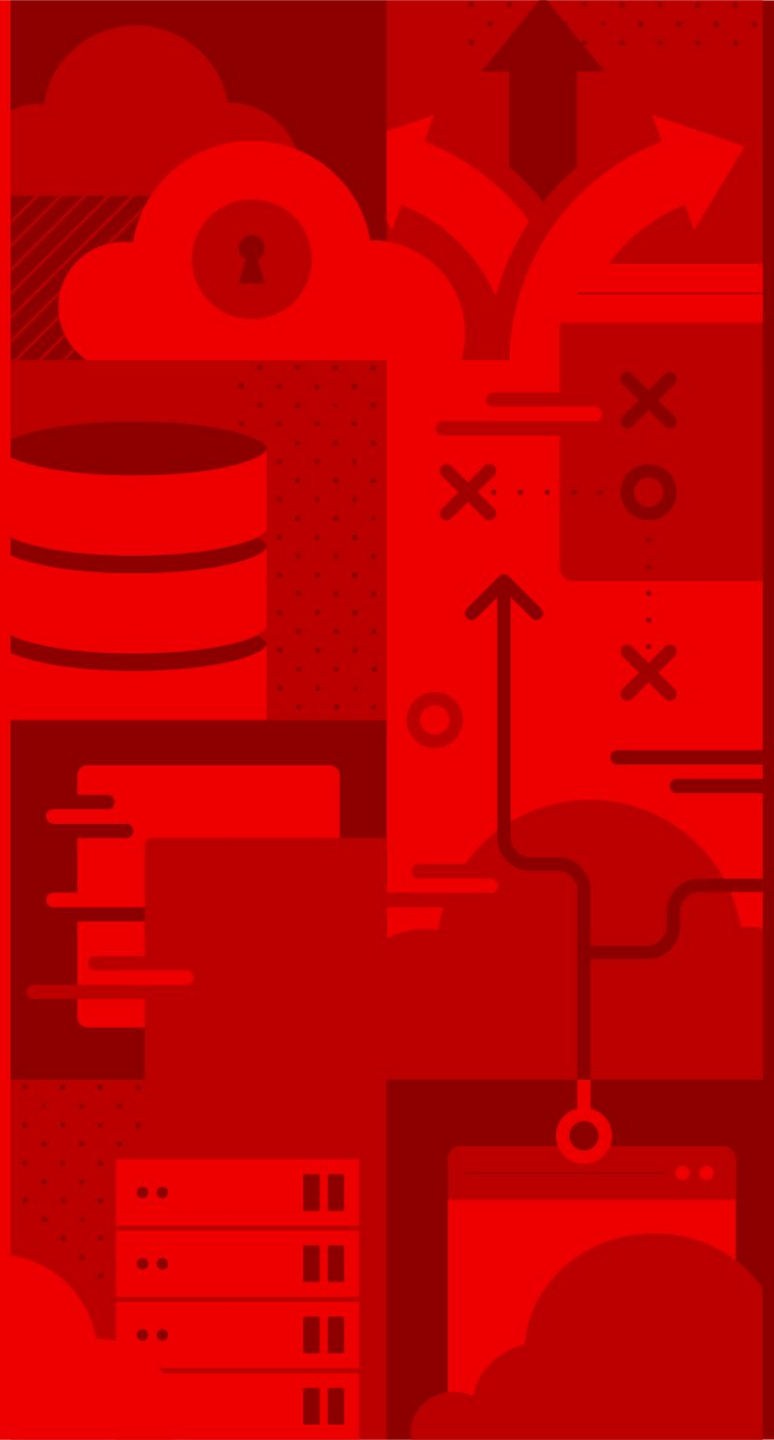
- **Free service to monitor per-project and per-cluster spending**
- Currency support
- Marketplace services reported including ROSA, ARO, RHEL, ODF, 3rd parties, etc
- ROSA and ARO costs distributed to projects
- Costs now distributed according to the same resource consumption criteria in every view
- 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

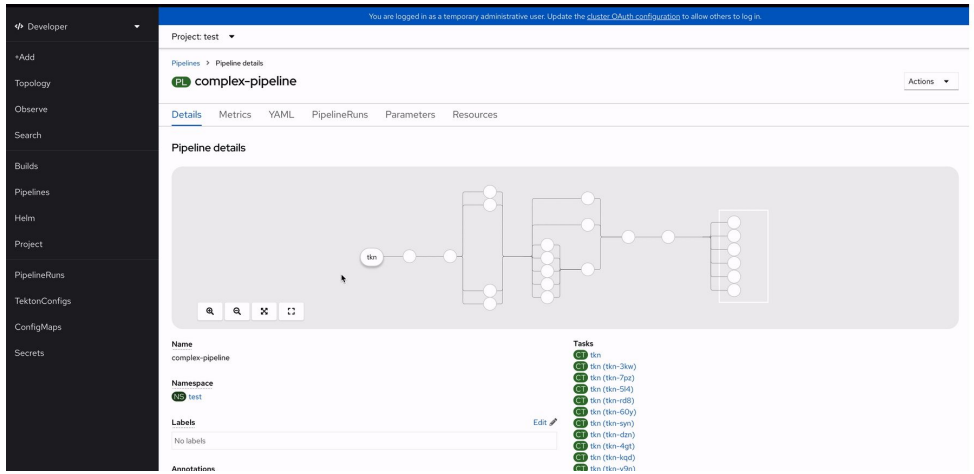
PM: Pau Garcia Quiles

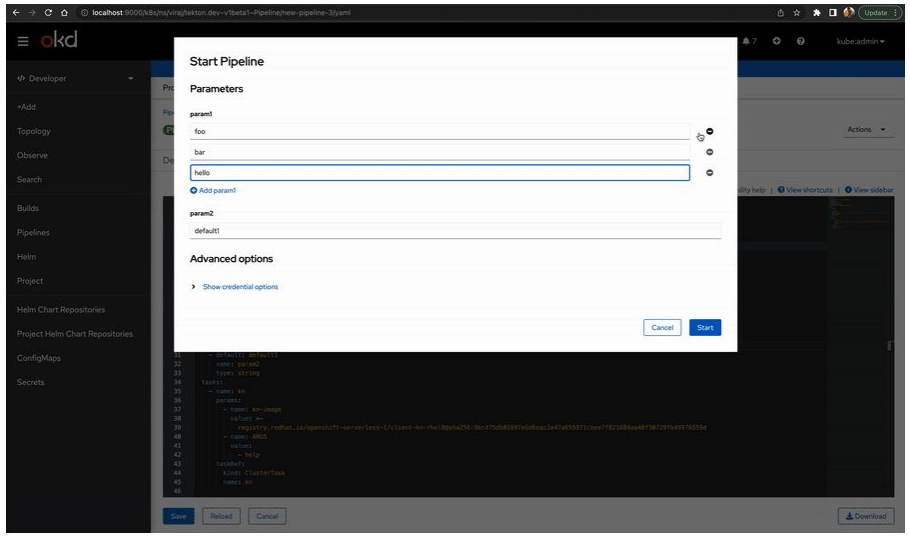


Platform Services

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

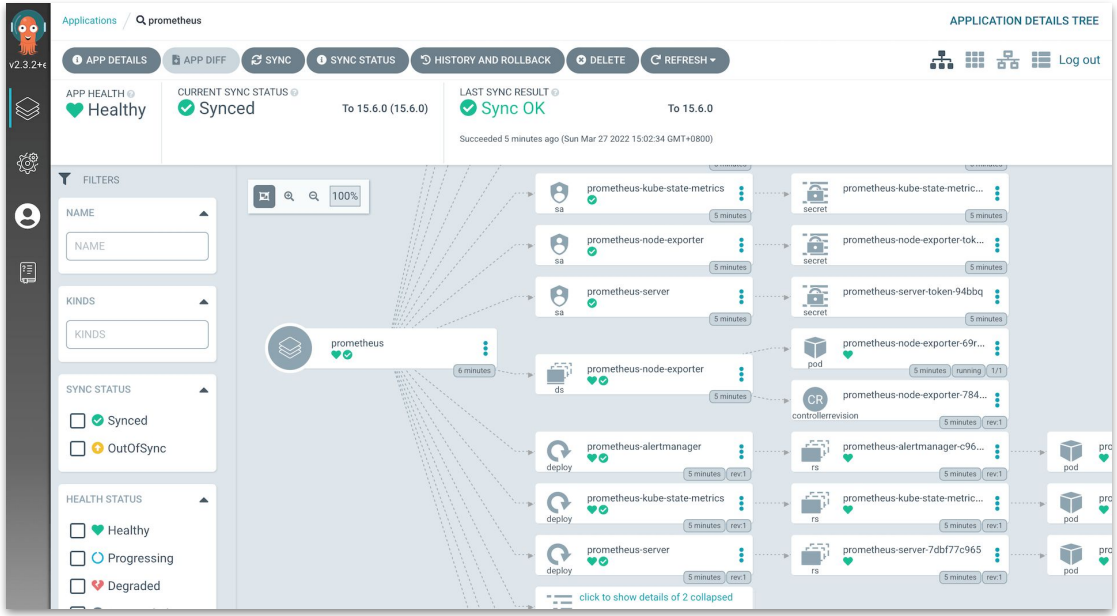




Red Hat

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

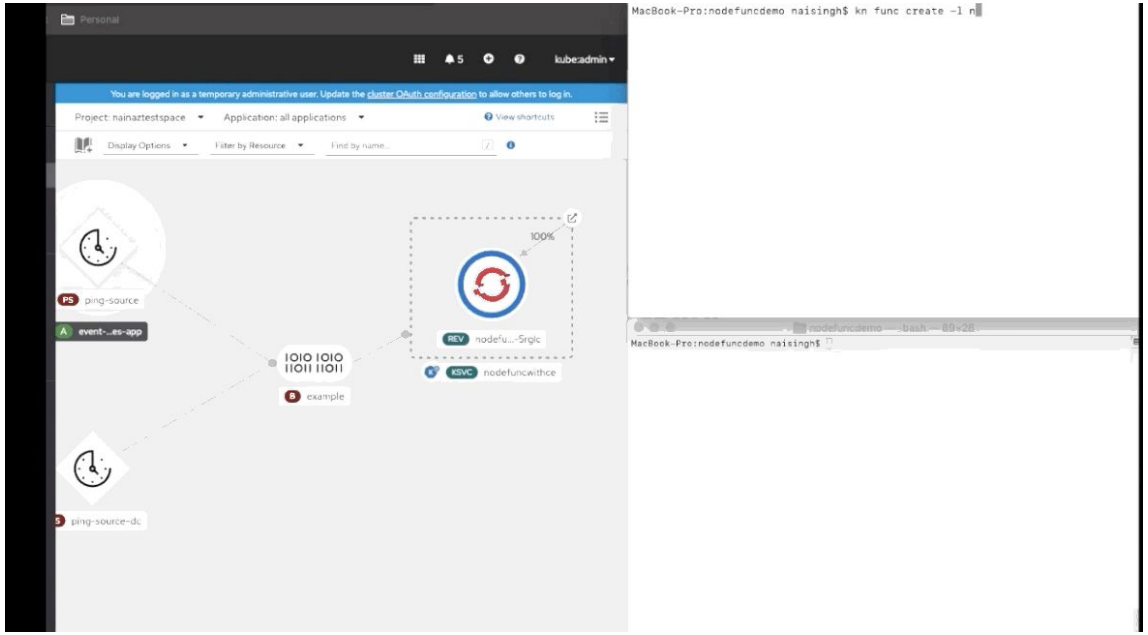


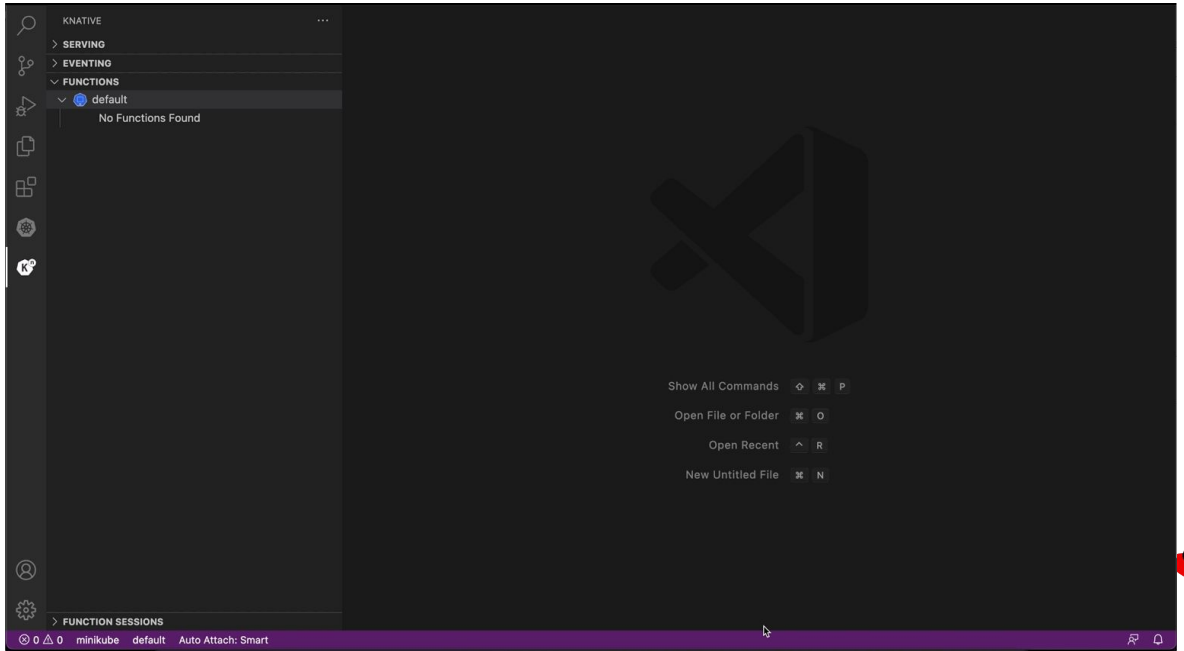
Red Hat

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)

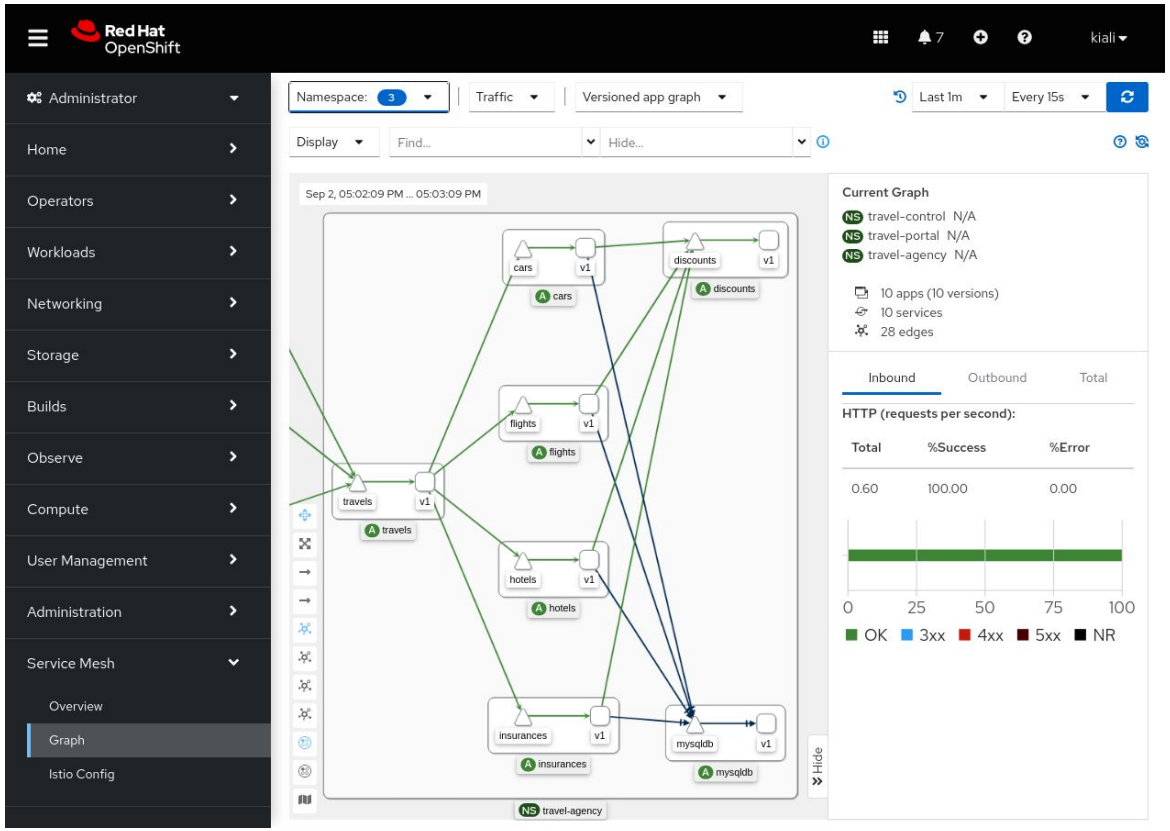




Red Hat

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)



Red Hat

Thank you

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


linkedin.com/company/red-hat


youtube.com/user/RedHatVideos


facebook.com/redhatinc


twitter.com/RedHat

Red Hat | IBM