State of Serverless & Service Mesh

William Markito Oliveira
Product Manager
@william_markito

Giuseppe Bonocore
Solution Architect

Sep 2019

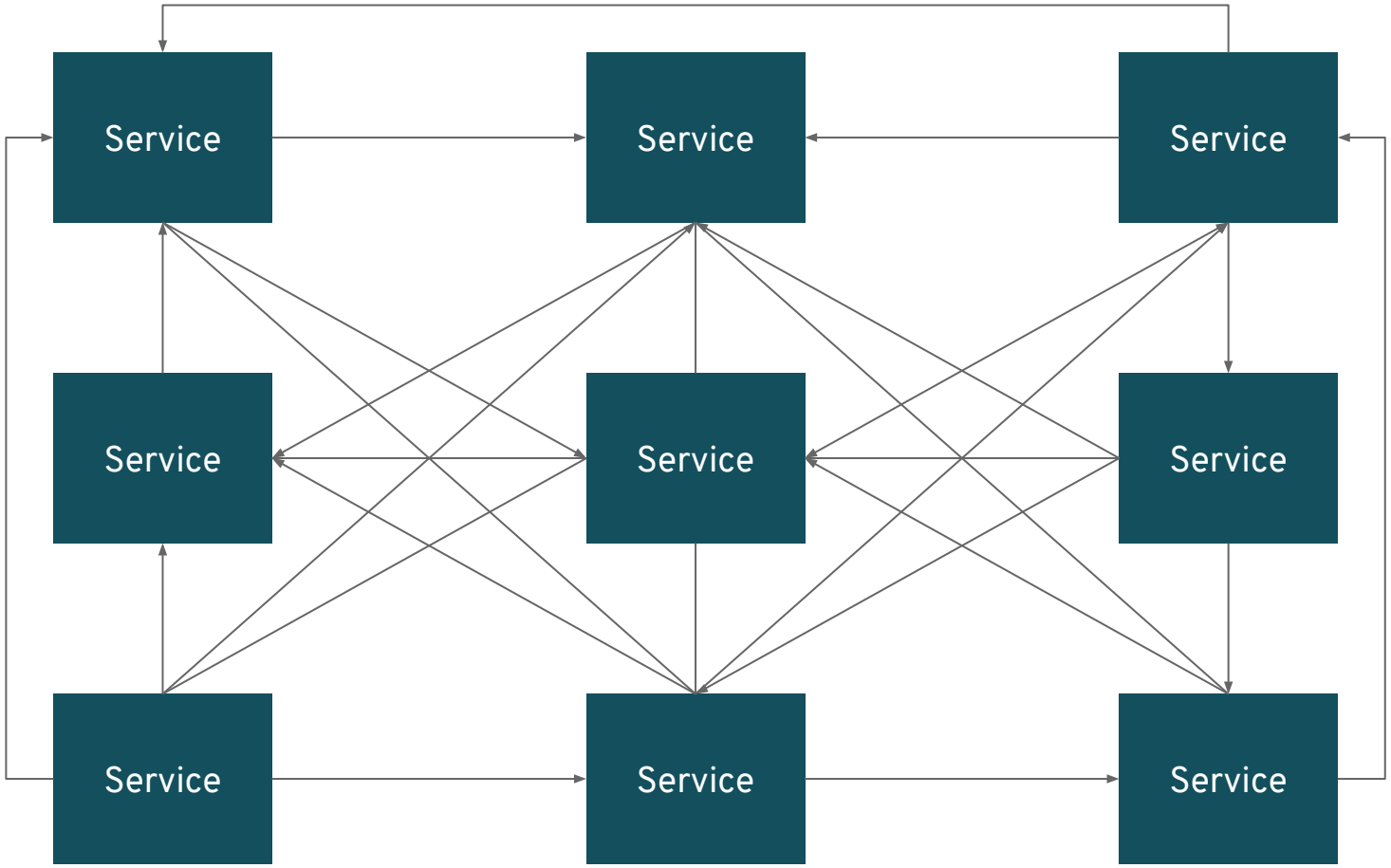
Red Hat

Sep 2019, Milano

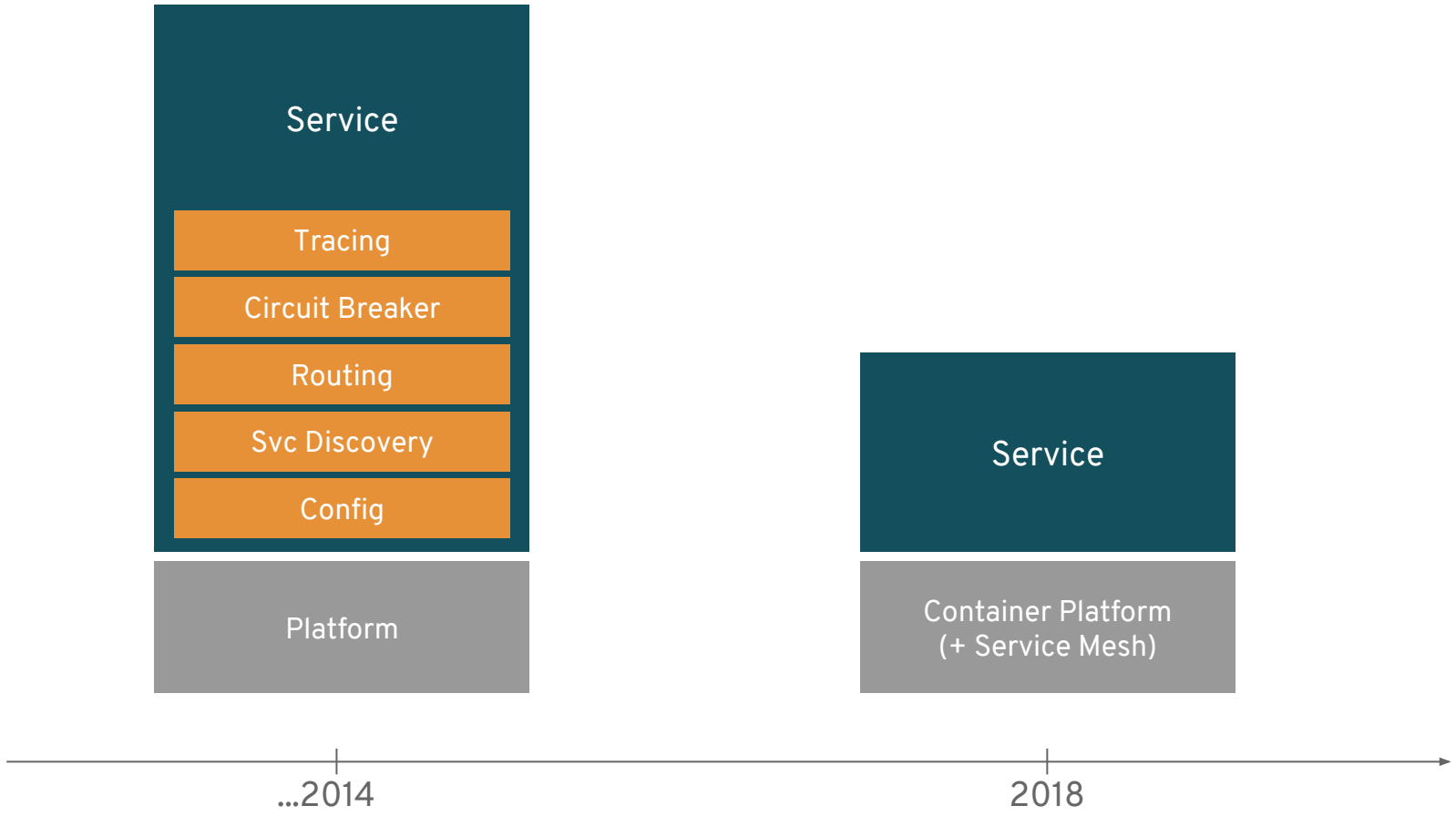
Red Hat

Service Mesh

Distributed Architecture



Microservices Evolution



Red Hat

Sidecars

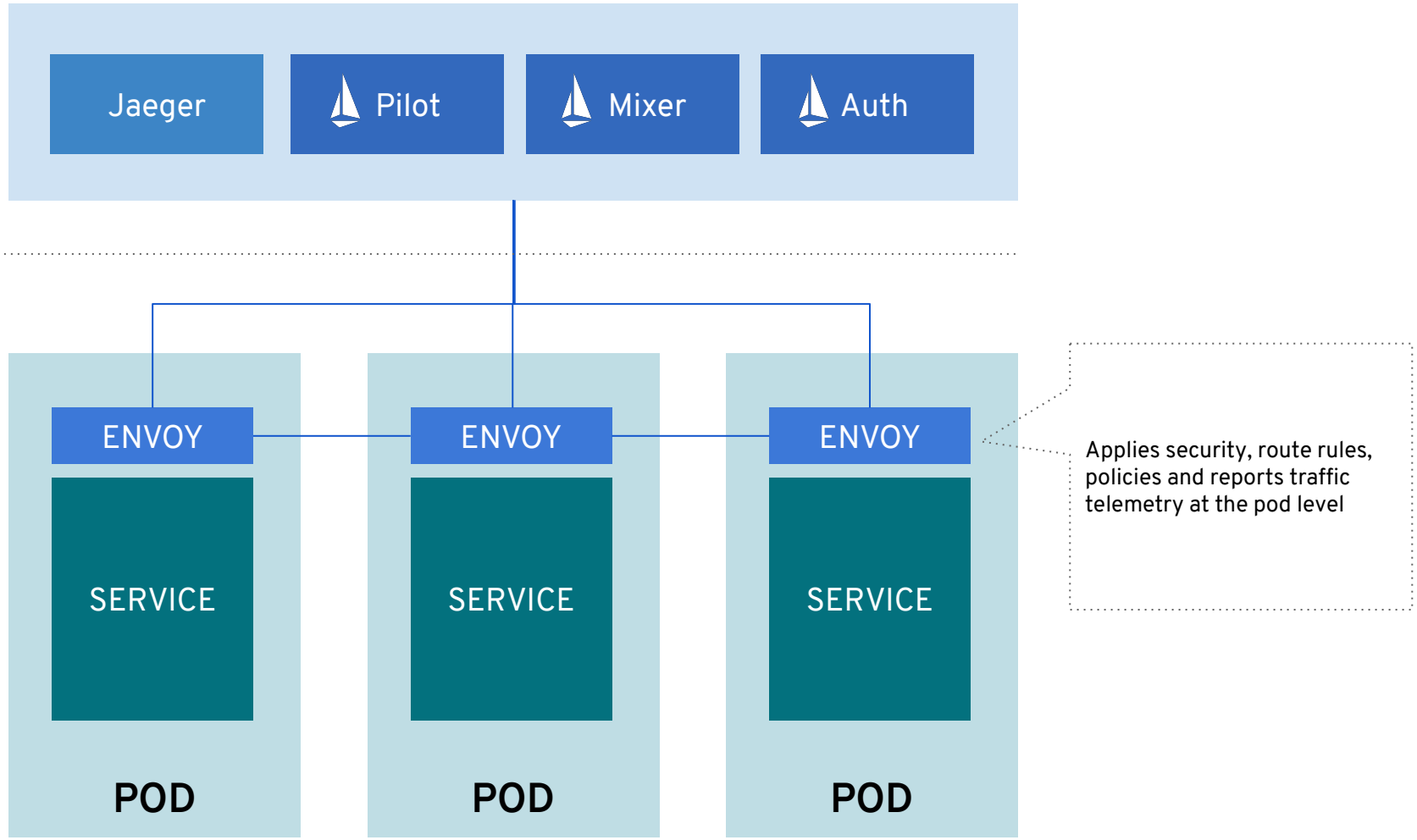
- Two or more containers deployed to same pod
- Share
  - Same
    - Namespace
    - Pod IP
  - Shared lifecycle
- Used to enhance the co-located containers
- Istio Proxy (L7 Proxy)
  - Proxy all network traffic in and out of the app container

Source: http://blog.kubernetes.io/2015/06/the-distributed-system-toolkit-patterns.html

Service Mesh Architecture

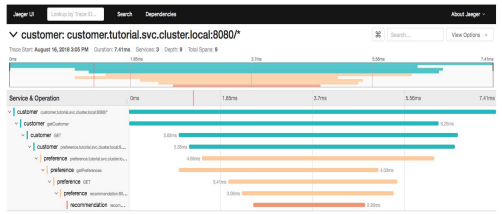
Control Plane

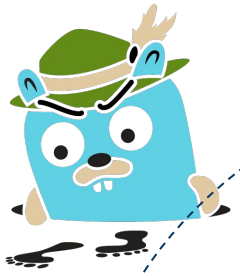
Data Plane



Red Hat

Service Mesh Ecosystem





Jaeger

Secure

Istio

Observe

Connect

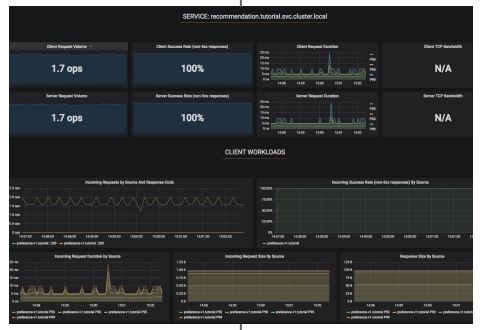
Control

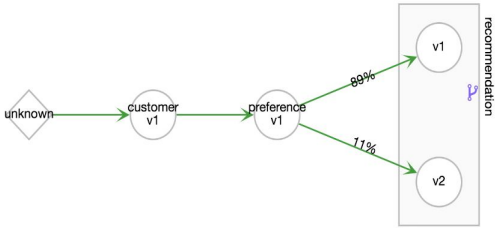
kiali

Kiali

Prometheus

Observe





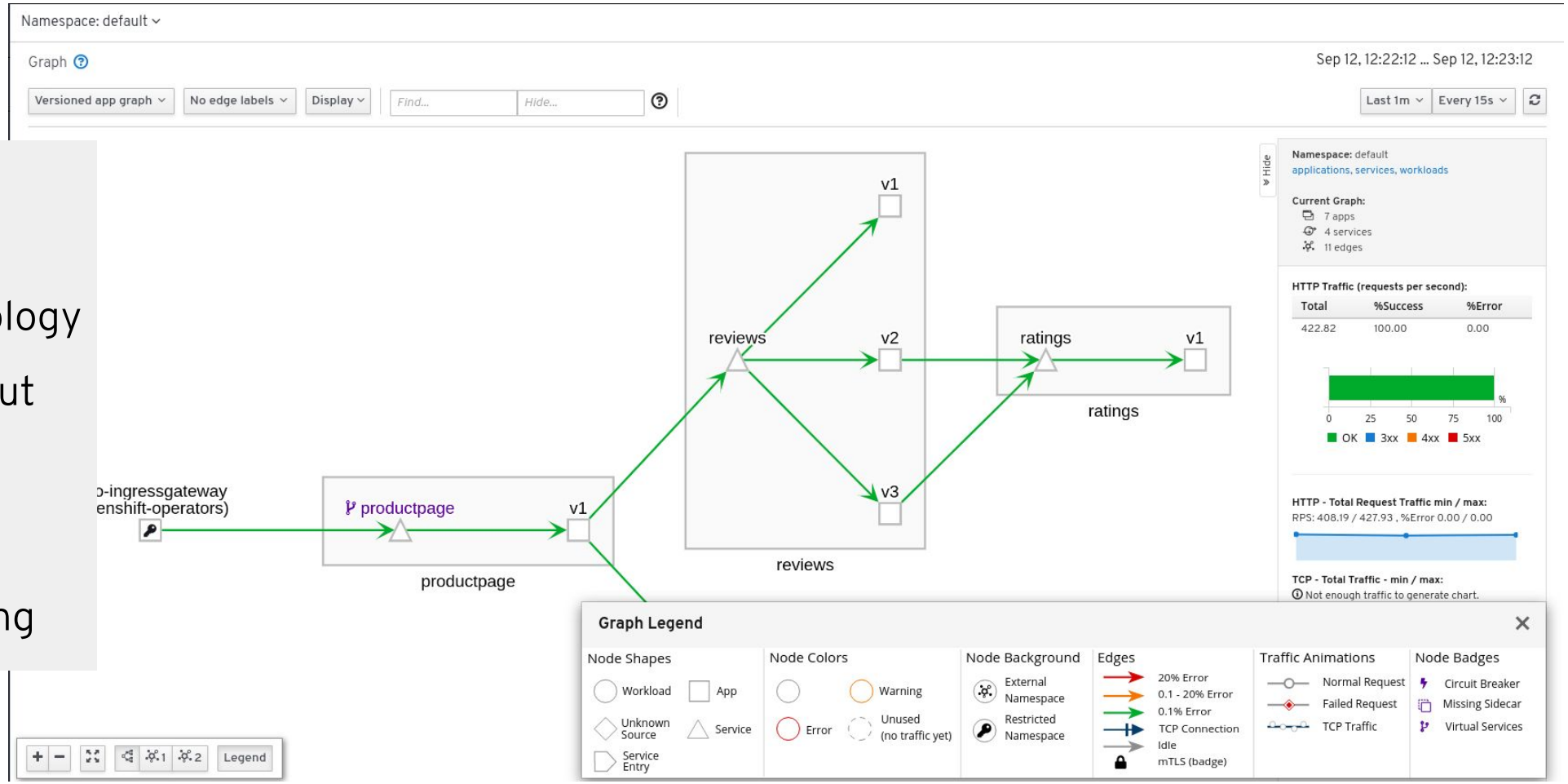
Grafana

Red Hat

Enhanced Visualization of Cluster Traffic With Kiali

Visualization of what Matters most:

- Application Topology
- Traffic throughput
- Error Rates
- Service Latency
- Service Versioning



Red Hat

Convenient Overviews of Individual Services

Services › Namespace: default › Service: reviews

reviews (Show on graph)

Overview | Traffic | Inbound Metrics

Last 1m | Actions

Labels
app reviews service reviews

Selectors
app reviews

Type ClusterIP
IP 172.30.17.223
Created at 9/12/2019, 12:06:32 PM
Resource Version 5094640

Ports
TCP http (9080)

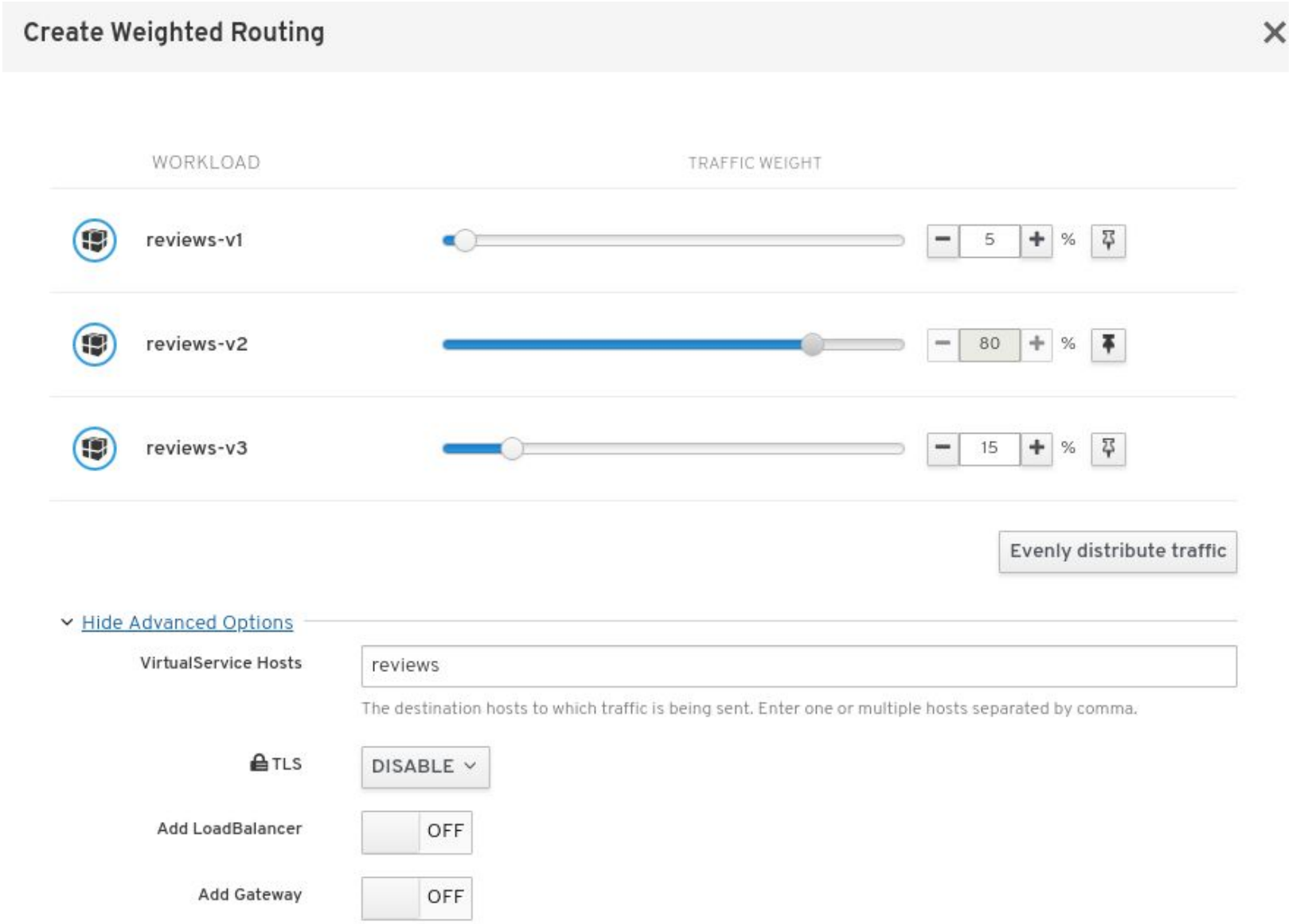
Endpoints
10.128.2.27 : reviews-v1-989d5ffdf-w8gmn
10.128.2.28 : reviews-v3-757c4f7849-rs7sw
10.131.0.35 : reviews-v2-6ff8648d69-tlqhn

Health
Healthy
Error Rate over last 1m: 0.00%

Workloads (3) | Virtual Services (0) | Destination Rules (0)

Name	Type	Labels	Created at	Resource version
reviews-v1	Deployment	app reviews version v1	9/12/2019, 12:06:32 PM	5095051
reviews-v2	Deployment	app reviews version v2	9/12/2019, 12:06:32 PM	5094998
reviews-v3	Deployment	app reviews version v3	9/12/2019, 12:06:32 PM	5095043

Guided Configuration of Traffic Policies



Red Hat

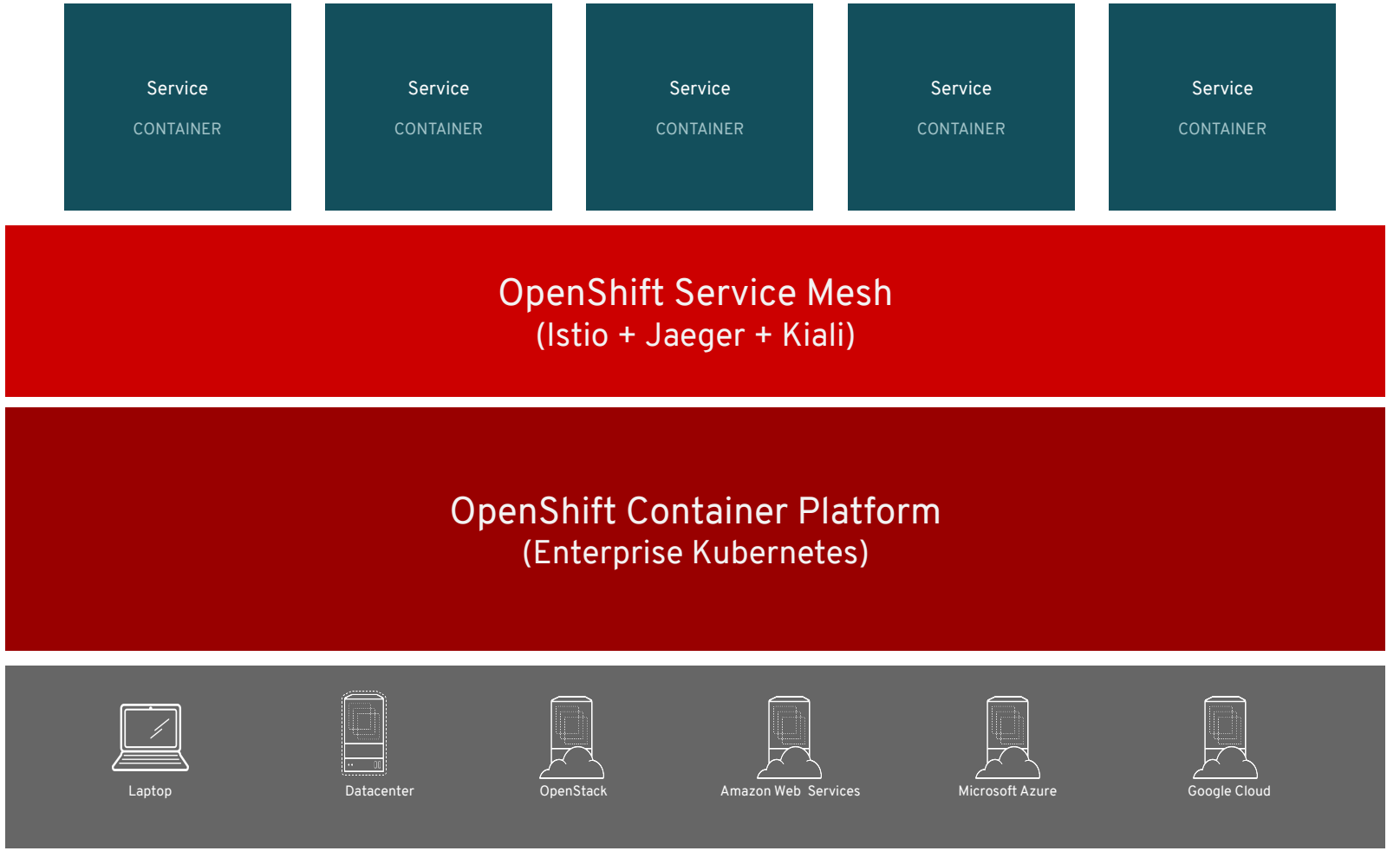
Distributed Services With Red Hat Openshift Service Mesh

SERVICE

SERVICE OPS

INFRA OPS

INFRA



ANY APPLICATION

RED HAT® OPENSHIFT

ANY INFRASTRUCTURE

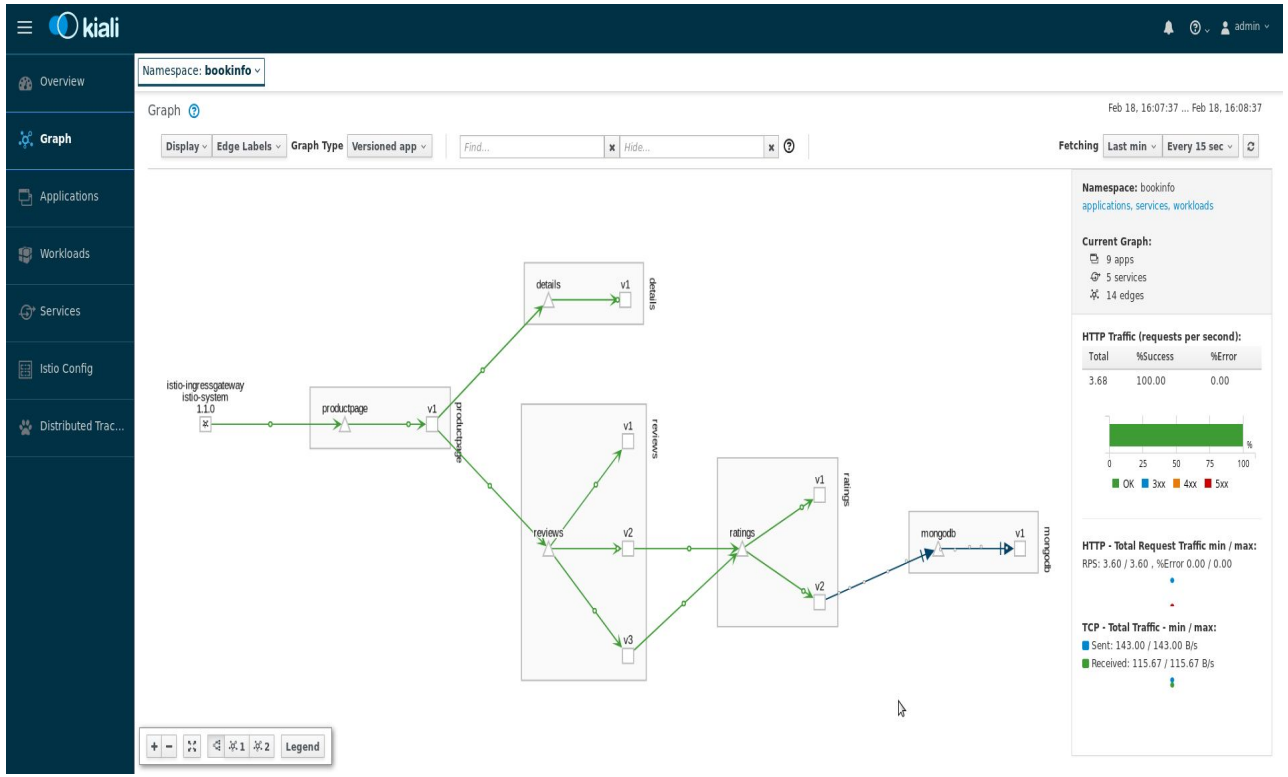
Red Hat

OpenShift Service Mesh

Generally Available

Key Features

- A dedicated network for service to service communications
- Observability and distributed tracing
- Policy-driven security
- Routing rules & chaos engineering
- Powerful visualization & monitoring
- Will be available via OperatorHub



Red Hat

Serverless

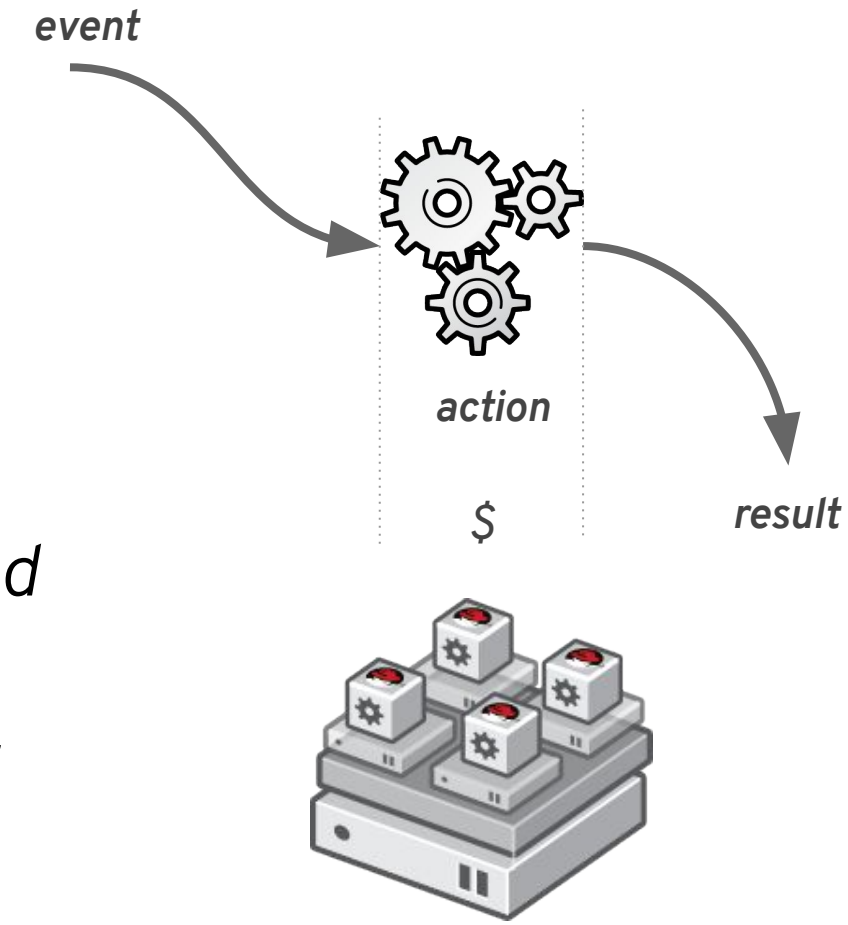
Wait… wat ?

https://github.com/ghuntley/serverless-to-cgi-bin

Red Hat

Serverless Defined

"computing execution model that depends on services to manage server-side logic and state where business logic run in stateless, event-triggered compute linux containers"



Serverless > Functions

Serverless > Microservices

Serverless > Containers

"Function as a Service (FaaS) is serverless in the same way a square is a rectangle"

CNCF Serverless Landscape

2019-05-19T02:14:19Z fe939f1

See the serverless interactive display at s.cncf.io

Greyed logos are not open source

Sep 2019, Milano

Tools

CLOUDZERO, dashbird, Epsagon, event gateway, HASURA GraphQL Engine, IO|pipe, Iron.io, Node Lambda, python-λ, SCAR, SIGMA, STACKERY, THUNDRA

Security

intrinsic, Protego, PURESEC, threat stack

Framework

APEX, .arc Architect, aws Chalice, AWS SAM, Claudia.js, FLOGO, gun.io, serverless, SPARTA, Spring Cloud Function

Hosted

Installable

Platform

ALGORITHMIA, Azure Functions, AWS Lambda, BINARIS, cloudboost, CLOUDFLARE Workers, Google Cloud Functions, HUAWEI FunctionStage, IBM Cloud Functions, kitsune, netlify, PubNub FUNCTIONS, spotinst, stdlib, Tencent Cloud, twilio Functions, ZEIT

Apache OpenWhisk, AppScale, DISPATCH, fission, fn, GALACTIC FOG, Knative, Kubeless, Kyma, LunchBadger, nuclio, OPENFAAS, PipelineAI, riff, Virtual Kubelet

s.cncf.io

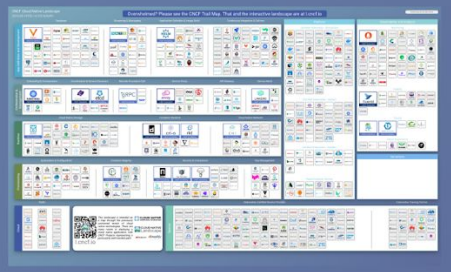
Serverless computing refers to a new model of cloud native computing, enabled by architectures that do not require server management to build and run applications. This landscape illustrates a finer-grained deployment model where applications, bundled as one or more functions, are uploaded to a platform and then executed, scaled, and billed in response to the exact demand needed at the moment

CLOUD NATIVE Landscape

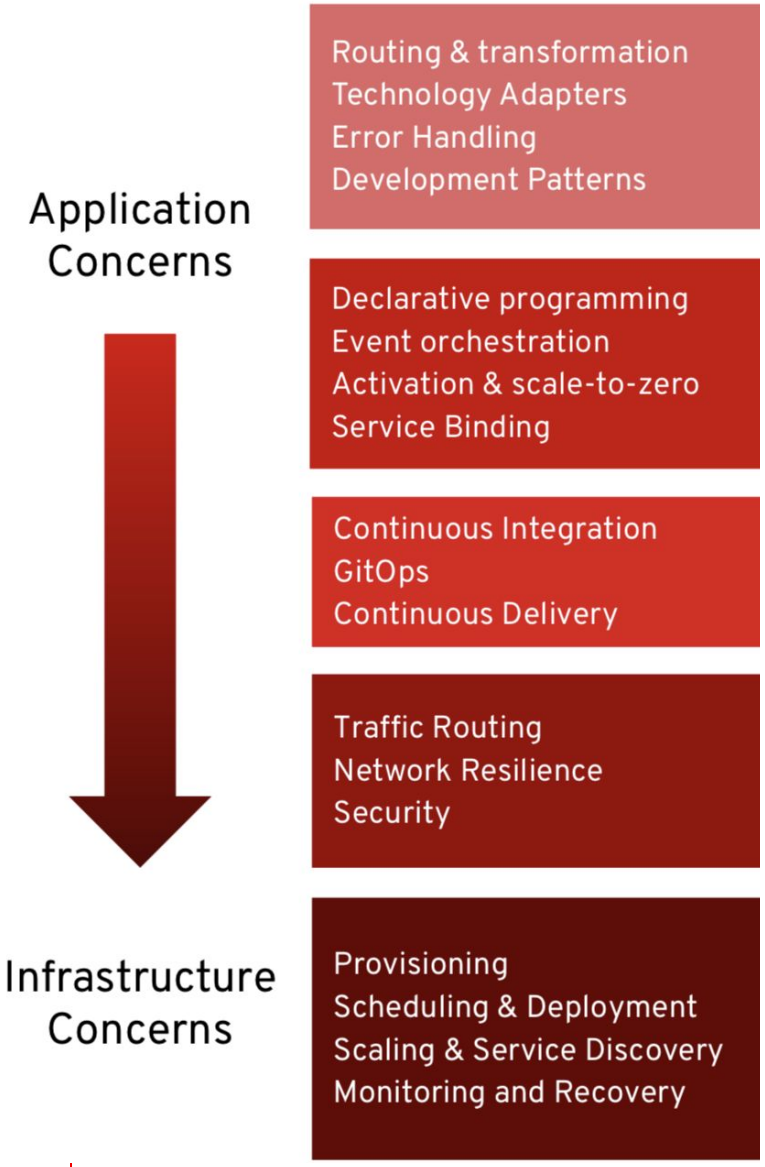
CLOUD NATIVE COMPUTING FOUNDATION

Redpoint

Cloud Native Landscape



Red Hat


Application Concerns
Routing & transformation
Technology Adapters
Error Handling
Development Patterns
Declarative programming
Event orchestration
Activation & scale-to-zero
Service Binding
Continuous Integration
GitOps
Continuous Delivery
Traffic Routing
Network Resilience
Security
Infrastructure Concerns
Provisioning
Scheduling & Deployment
Scaling & Service Discovery
Monitoring and Recovery

Application Concerns

- Routing & transformation
- Technology Adapters
- Error Handling
- Development Patterns

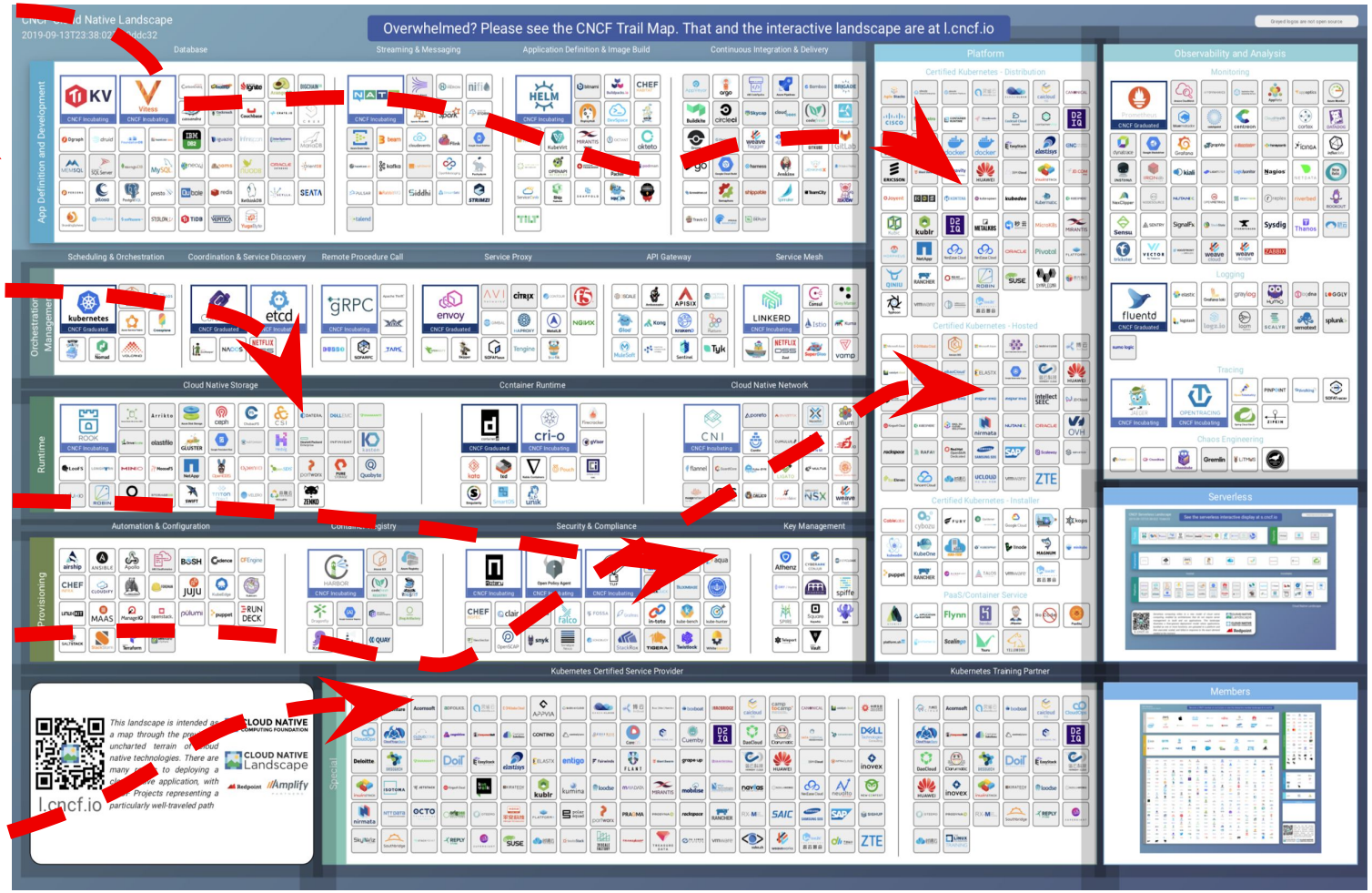
- Declarative programming
- Event orchestration
- Activation & scale-to-zero
- Service Binding

- Continuous Integration
- GitOps
- Continuous Delivery

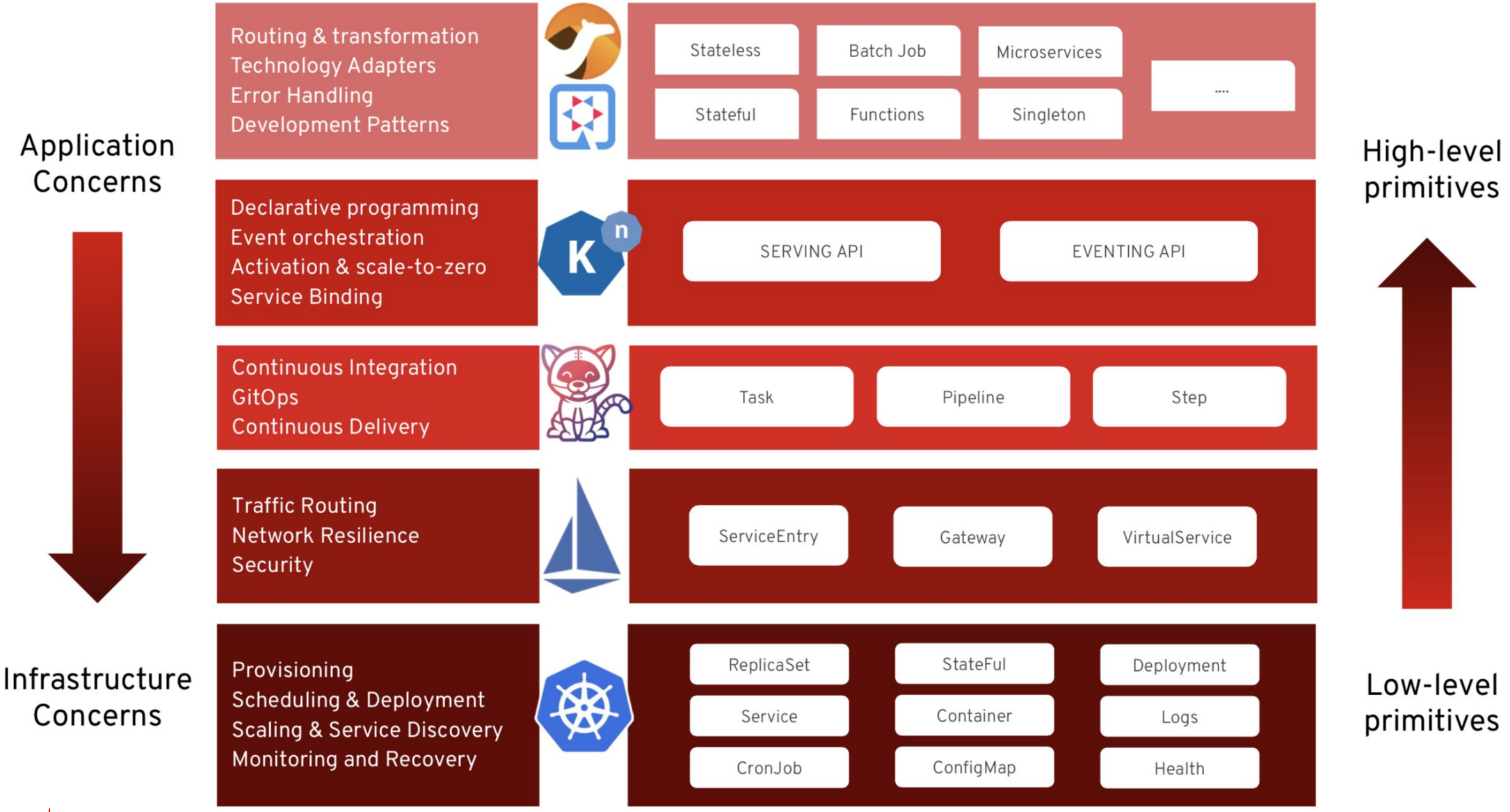
- Traffic Routing
- Network Resilience
- Security

Infrastructure Concerns

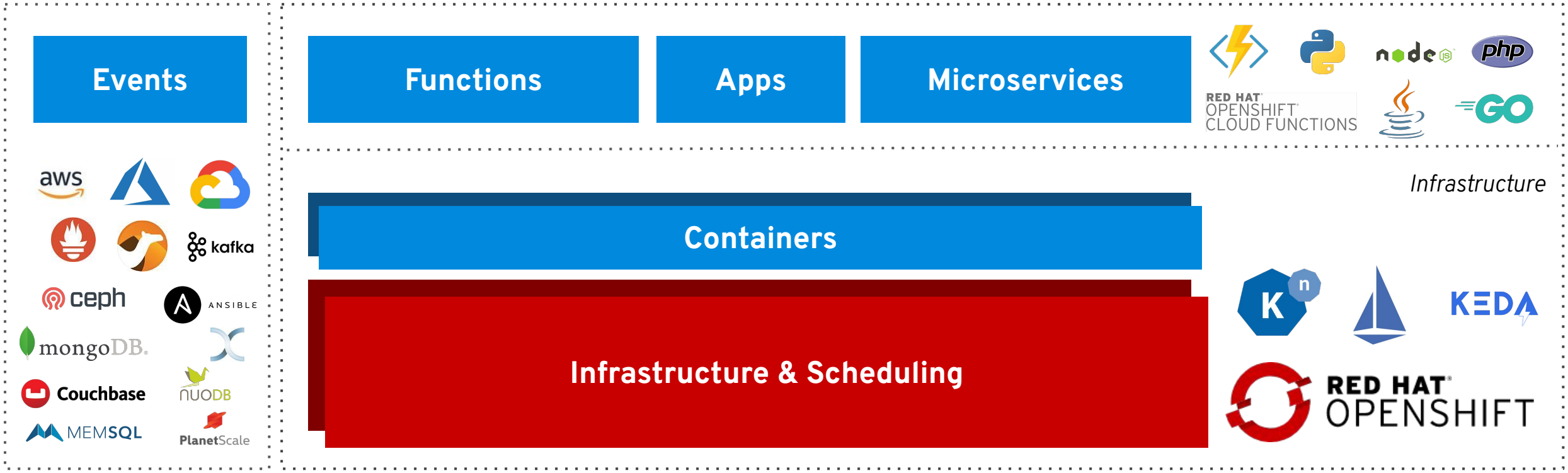
- Provisioning
- Scheduling & Deployment
- Scaling & Service Discovery
- Monitoring and Recovery



Red Hat

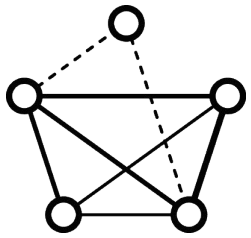

Application Concerns
Infrastructure Concerns
High-level primitives
Low-level primitives
Routing & transformation
Technology Adapters
Error Handling
Development Patterns
Stateless
Batch Job
Microservices
....
Stateful
Functions
Singleton
Declarative programming
Event orchestration
Activation & scale-to-zero
Service Binding
SERVING API
EVENTING API
Continuous Integration
GitOps
Continuous Delivery
Task
Pipeline
Step
Traffic Routing
Network Resilience
Security
ServiceEntry
Gateway
VirtualService
Provisioning
Scheduling & Deployment
Scaling & Service Discovery
Monitoring and Recovery
ReplicaSet
StateFul
Deployment
Service
Container
Logs
CronJob
ConfigMap
Health

Microservices, Functions and Apps + Events

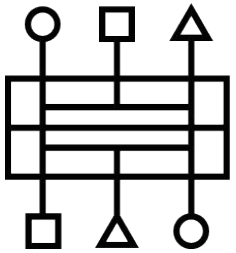


OperatorHub.io

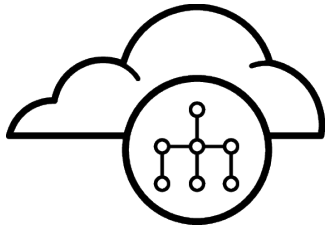
First Principles



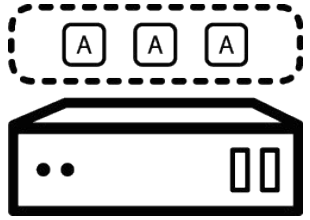
Distributed



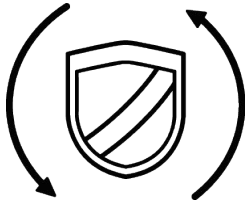
API Centric



Multi-cloud

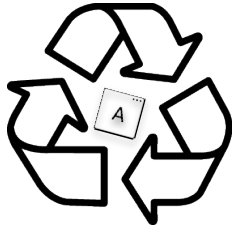


Scalable

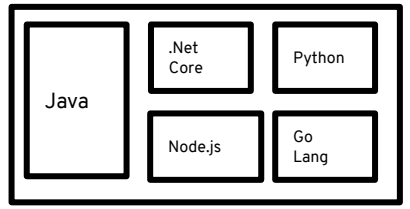


Secure

Event Driven



Disposable



Polyglot

Red Hat

Serverless & Knative

Range Last month Metric Contributions

Knative Companies statistics (Contributions, Range: Last month), bots excluded

Company	Number
All	8077
Google	5431
Red Hat	1001
IBM	457
Pivotal	103
Dropbox	45
TriggerMesh	27
Alfresco	21
Microsoft	20
Mirantis	7
SAP	6
Cisco	6
Ill	5
BoCloud	5
The Elegant Monkeys	2
Stark & Wayne	2
Schibsted	2
StatDNA	1
EVRY	1
China Mobile	1

Companies contributing to Knative

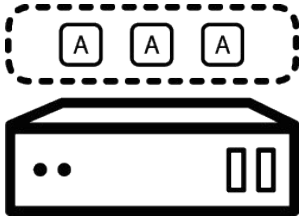
Source:
https://knative.teststats.cncf.io/d/8/dashboards?refresh=15m&orgId=1

Red Hat

What is Knative ?

SERVING

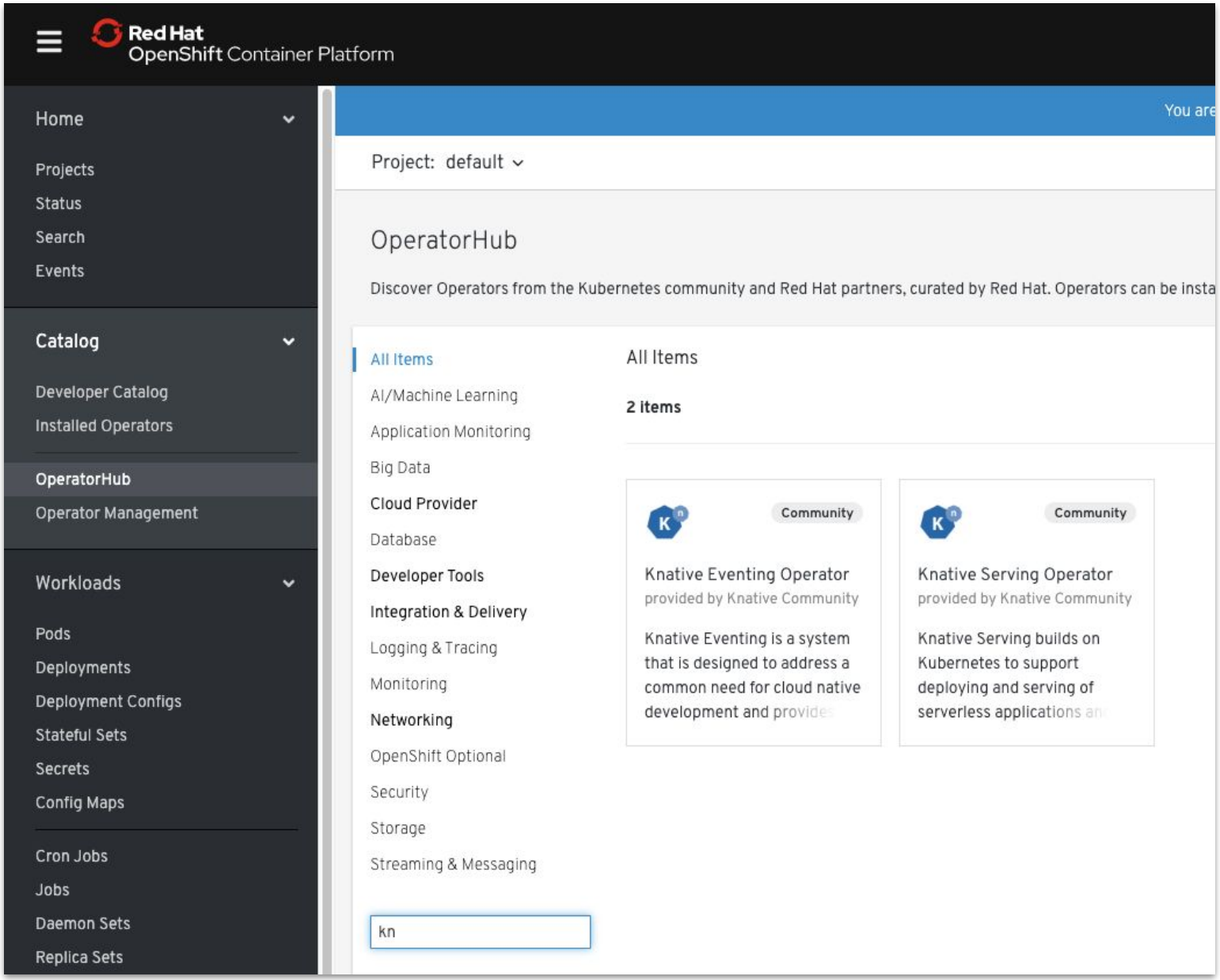
An event-driven model that serves the container with your application and can "scale to zero".



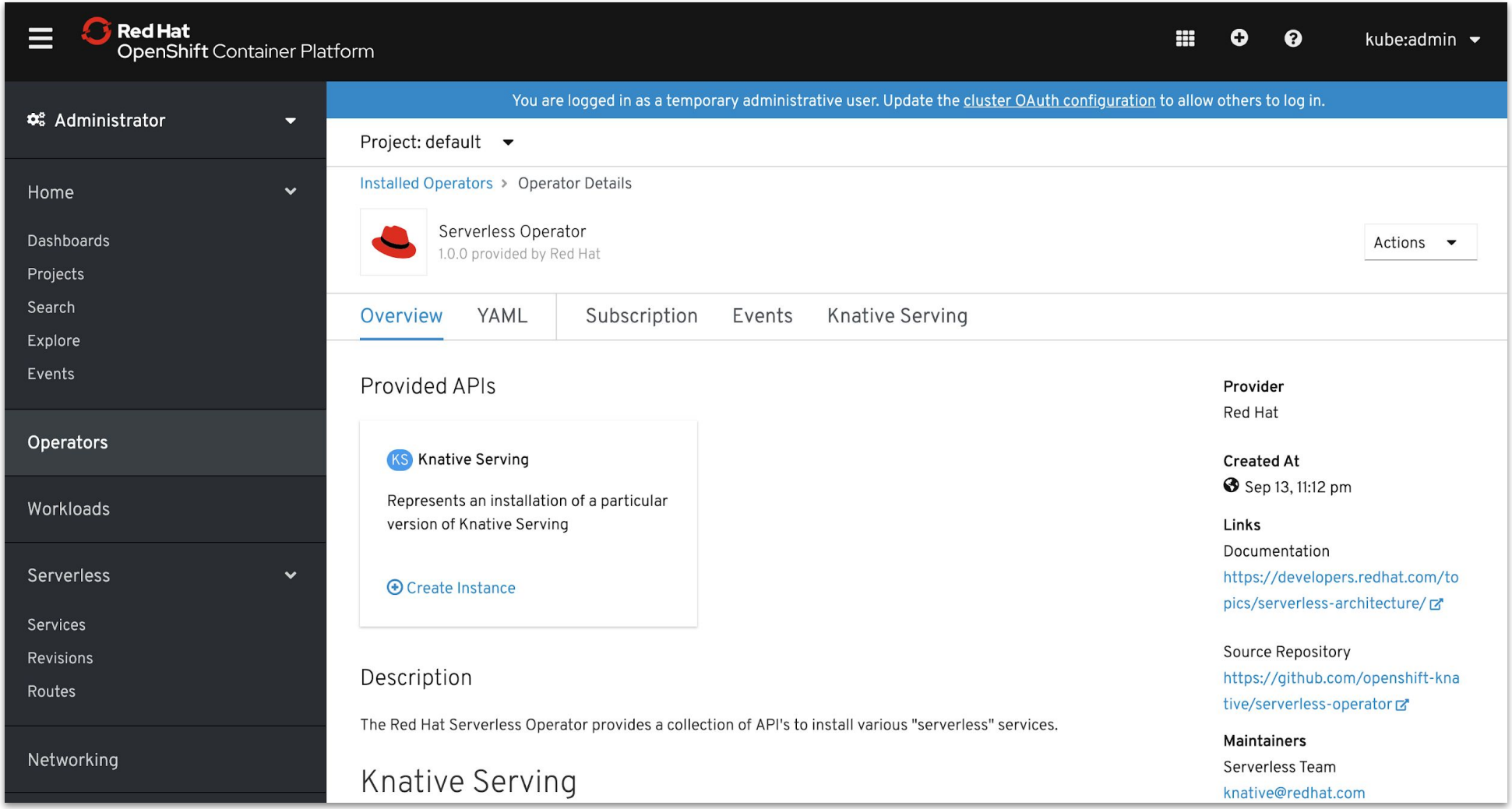
EVENTING

Common infrastructure for consuming and producing events that will stimulate applications.

Community Operators



Red Hat



Productized Operator

Red Hat

User Experience

```
kn service create hello --image gcr.io/knative-samples/helloworld-go --env NAME=value
Service 'hello' successfully created in namespace 'default'.
```

Options: `--limits-cpu 100m --limits-memory 1024m`

```
kn service get hello
NAME            DOMAIN                          GENERATION   AGE     CONDITIONS   READY
REASON
hello           hello.default.example.com       1            3m5s    3 OK / 3     True
```

Other commands available: `update, replace, get, describe`

More details at https://github.com/knative/client/tree/master/docs

"Lift & Shift" to Serverless

Kubernetes

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
 name: frontend
 labels:
   app: guestbook
spec:
 selector:
   matchLabels:
     app: guestbook
     tier: frontend
 replicas: 1
 template:
   metadata:
     labels:
       app: guestbook
       tier: frontend
   spec:
     containers:
     - image: markusthoemmes/guest
       name: guestbook
       resources:
         requests:
           cpu: 100m
           memory: 100Mi
       env:
       - name: GET_HOSTS_FROM
         value: dns
       ports:
       - containerPort: 80
```

```yaml
---
apiVersion: v1
kind: Service
metadata:
 name: frontend-service
 labels:
   app: guestbook
   tier: frontend
spec:
 ports:
 - port: 80
 selector:
   app: guestbook
   tier: frontend
---
apiVersion: route.openshift.io/v1
kind: Route
metadata:
 name: frontend-route
spec:
 to:
   kind: Service
   name: frontend-service
```

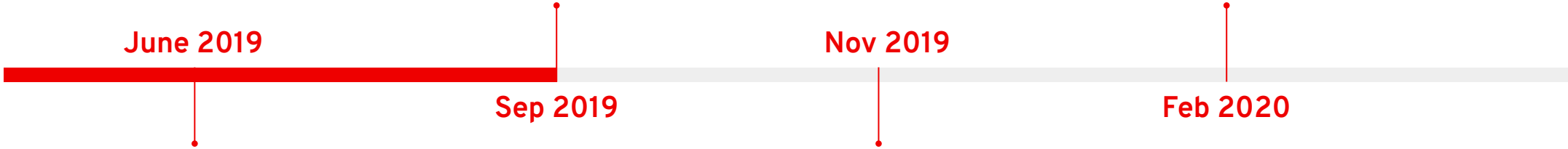
53 lines

Knative

```yaml
apiVersion: serving.knative.dev/v1alpha1
kind: Service
metadata:
 name: frontend
spec:
 template:
   metadata:
     labels:
       app: guestbook
       tier: frontend
   spec:
     containers:
     - image: markusthoemmes/guestbook
       resources:
         requests:
           cpu: 100m
           memory: 100Mi
       env:
       - name: GET_HOSTS_FROM
         value: dns
       ports:
       - containerPort: 80
```

22 lines

OpenShift Serverless Roadmap



Dev Preview 0.7

Knative v0.7.0 (v1alpha)
Operators-based Install
Domain config & Ingress
CLI (kn) for Serving
Camel-K event source

Tech Preview 1

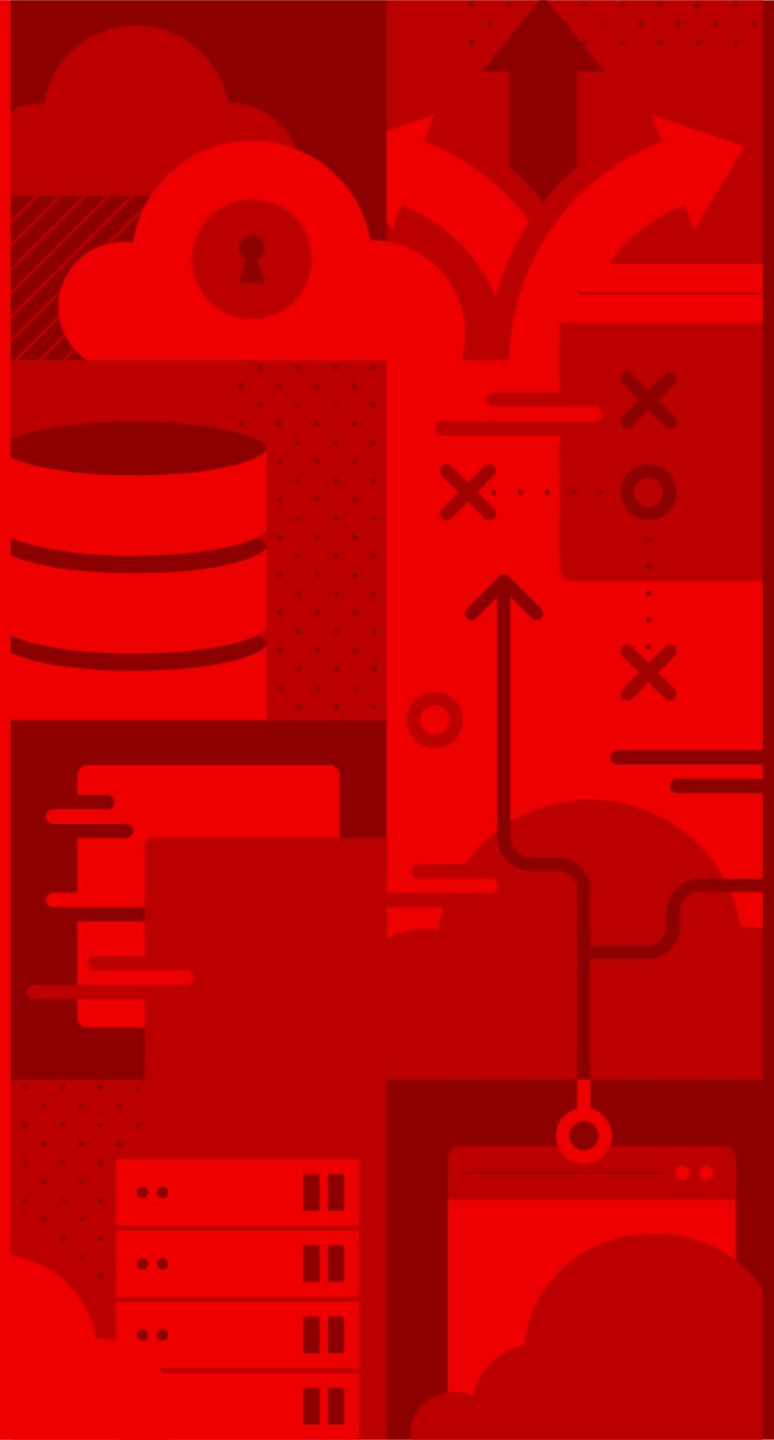
Knative v0.7.1 (v1alpha1)
Kafka event source
Installation docs
Metering
Logging & Monitoring
Developer Console (UI)

Tech Preview 2

Knative v0.9.0 (v1beta1)
Tekton Integration (Pipelines)
VS Code Plugins
CLI (kn) for Eventing
Support for GPU workloads
Rolling Updates (using OLM)
App Migration ("Knatify")

General Availability

Knative Eventing (TP)
Developer Console (Eventing)
Advanced Metering (Cost)
Functions Catalog (Kscout)
Program for ISV Event Sources
Camel-K productization



DEMO

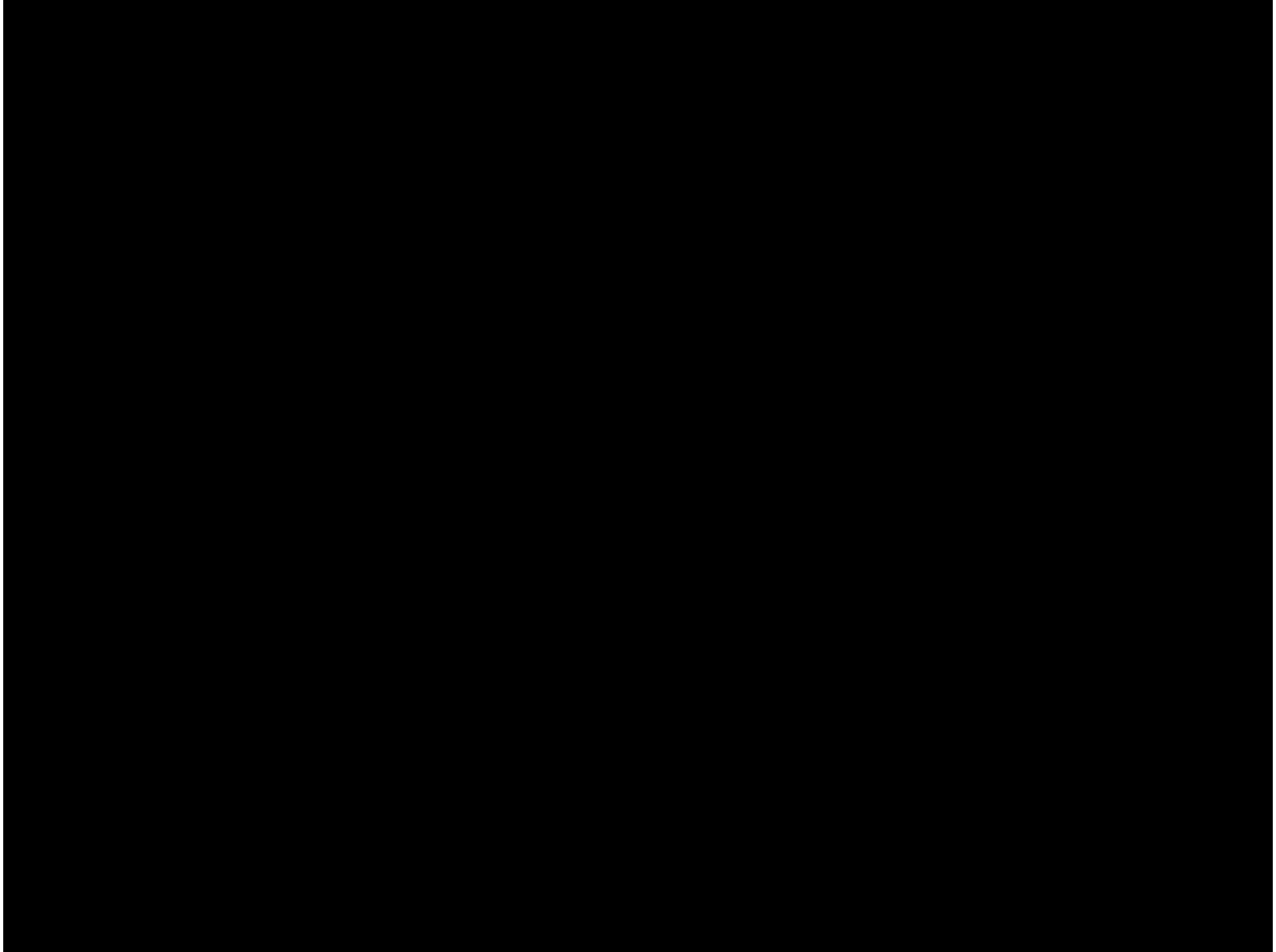
The Journey to Profit$! with Serverless

Kiosk App

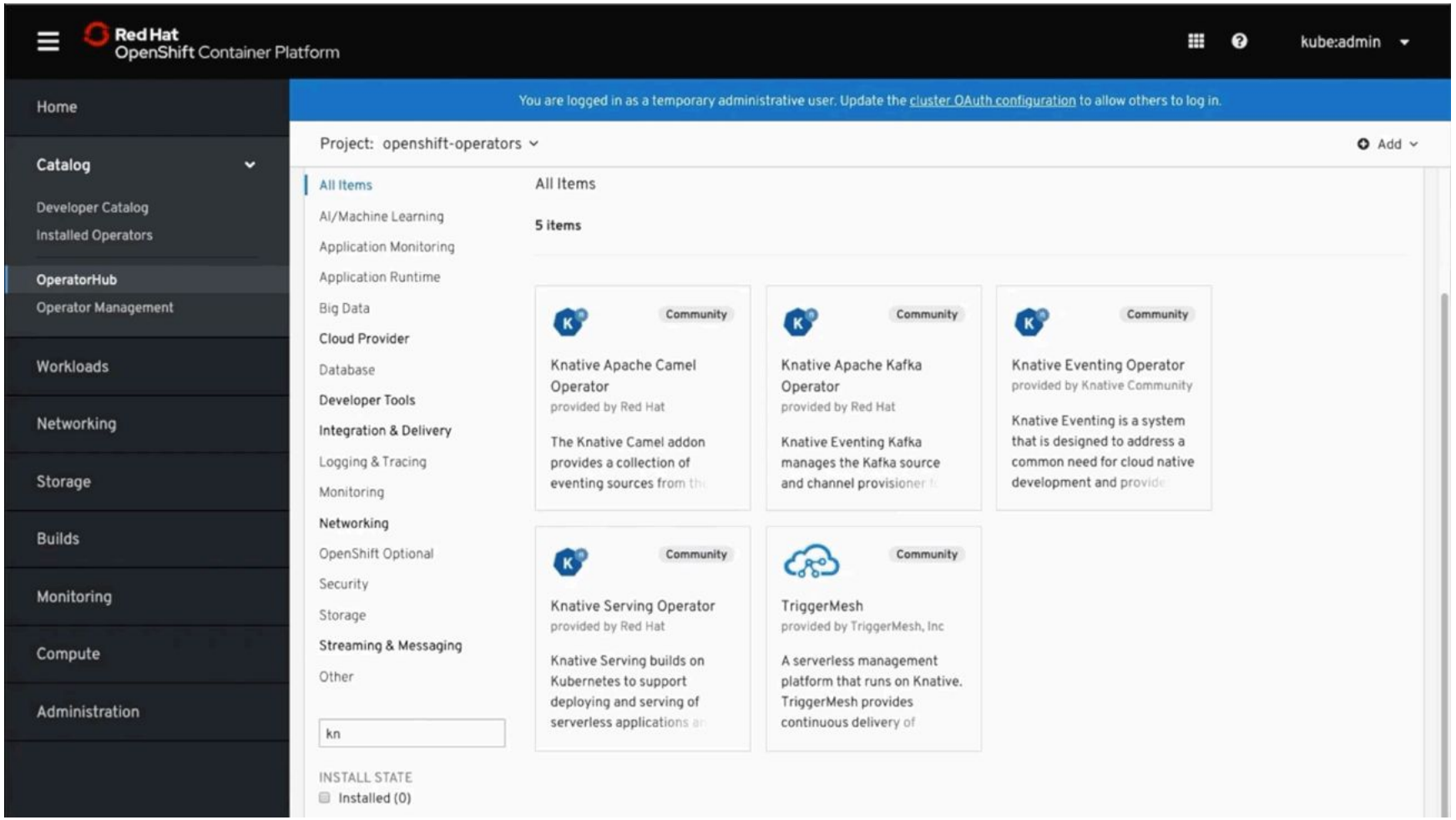
Store App

Order Routing

Payment



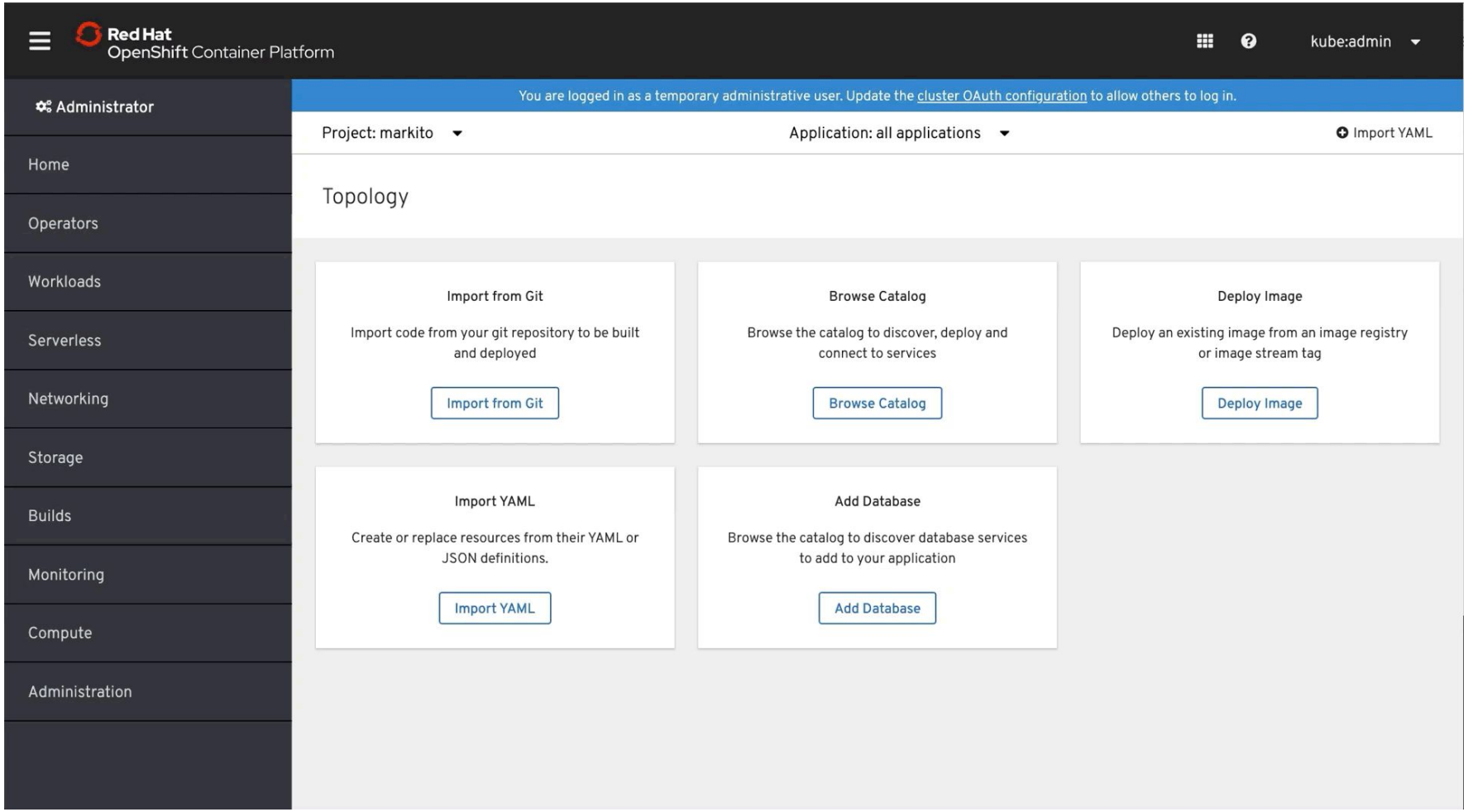
Admin User Experience



User Experience

Applies to:

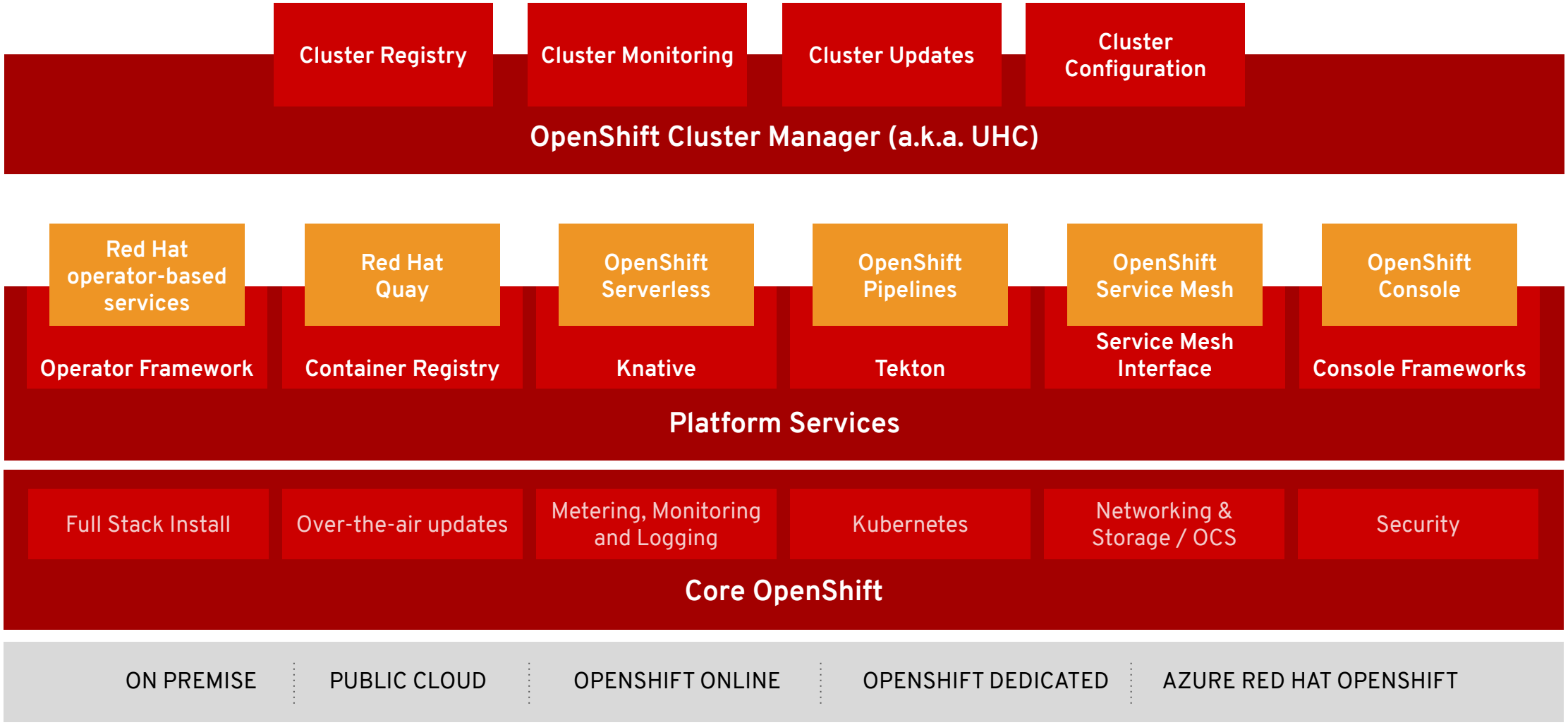
- Deploy Image
- Import from Git
- Create new App



* Developer Perspective coming in OpenShift 4.2

Summary & Roadmap

OpenShift Platform Services



Red Hat

Developer Preview

Azure Functions & KEDA

Key features

- Enable FaaS in OpenShift
- Familiar developer experience using VS Code and Azure CLI
- Polling based auto-scaling for Azure Queues, Kafka...
- Reuse Knative event sources, HTTP auto-scaling
- On premise or Any cloud.
- Familiar to Kubernetes users. Native.

Learn more

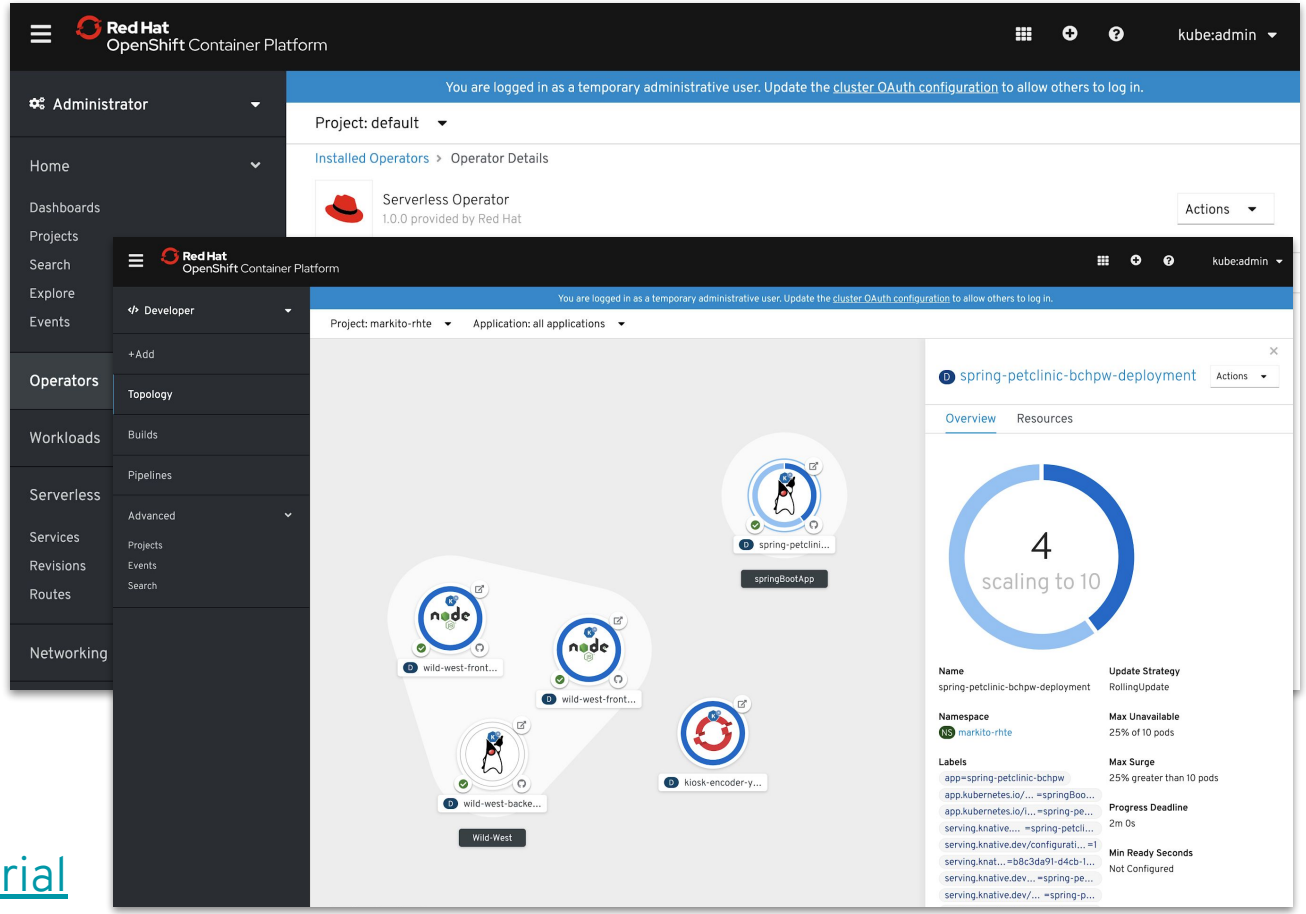
- https://github.com/kedacore/keda

In partnership with: Red Hat | Microsoft Azure

OpenShift Serverless

Key Features

- Familiar to Kubernetes users. Native.
- Scale to 0 and autoscale to N based on demand
- Applications and functions. Any container workload.
- Powerful eventing model with multiple event sources.
- Operator available via OperatorHub
- Knative v0.8 (v1beta1 APIs)
- No vendor lock in



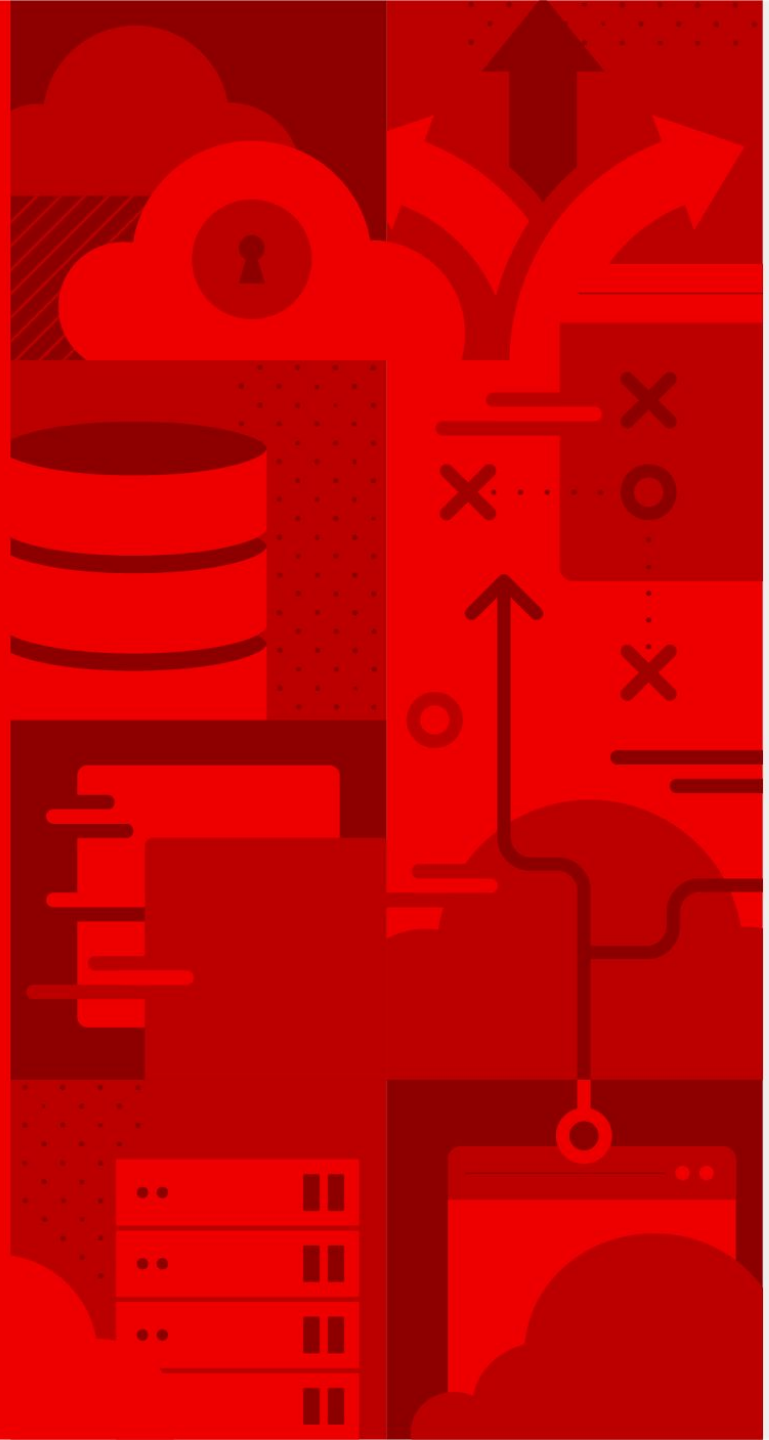
Learn more

https://openshift.com/learn/topics/serverless

https://redhat-developer-demos.github.io/knative-tutorial

See you soon at...

Red Hat



Thank you

- openshift.com/learn/topics/knative
- https://redhat-developer-demos.github.io/knative-tutorial
- github.com/kedacore/keda

linkedin.com/company/red-hat

facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

@william_markito

Red Hat