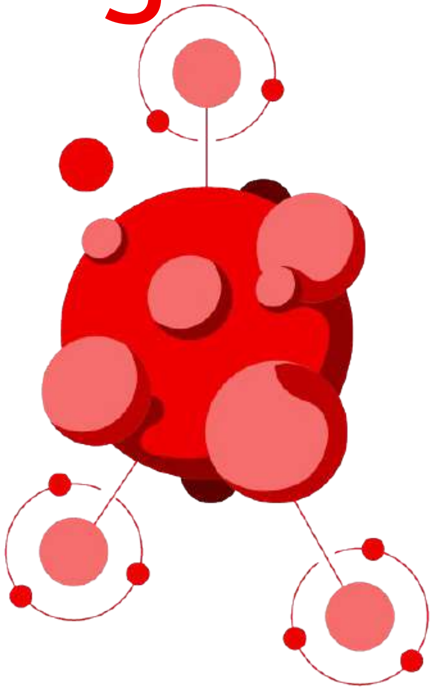
Red Hat | intel

Open Tour

Connecting people and solutions
to accelerate your business

Hybrid Cloud Development: 10 Best Practices using ARO and ROSA



Yury Titov

Senior BlackBelt for Managed Cloud Services,

Red Hat

Introduction

Yury Titov

- former senior EMEA Architect
- present: senior BlackBelt for Managed Cloud Services
- always: open source dude

What we’ll discuss today

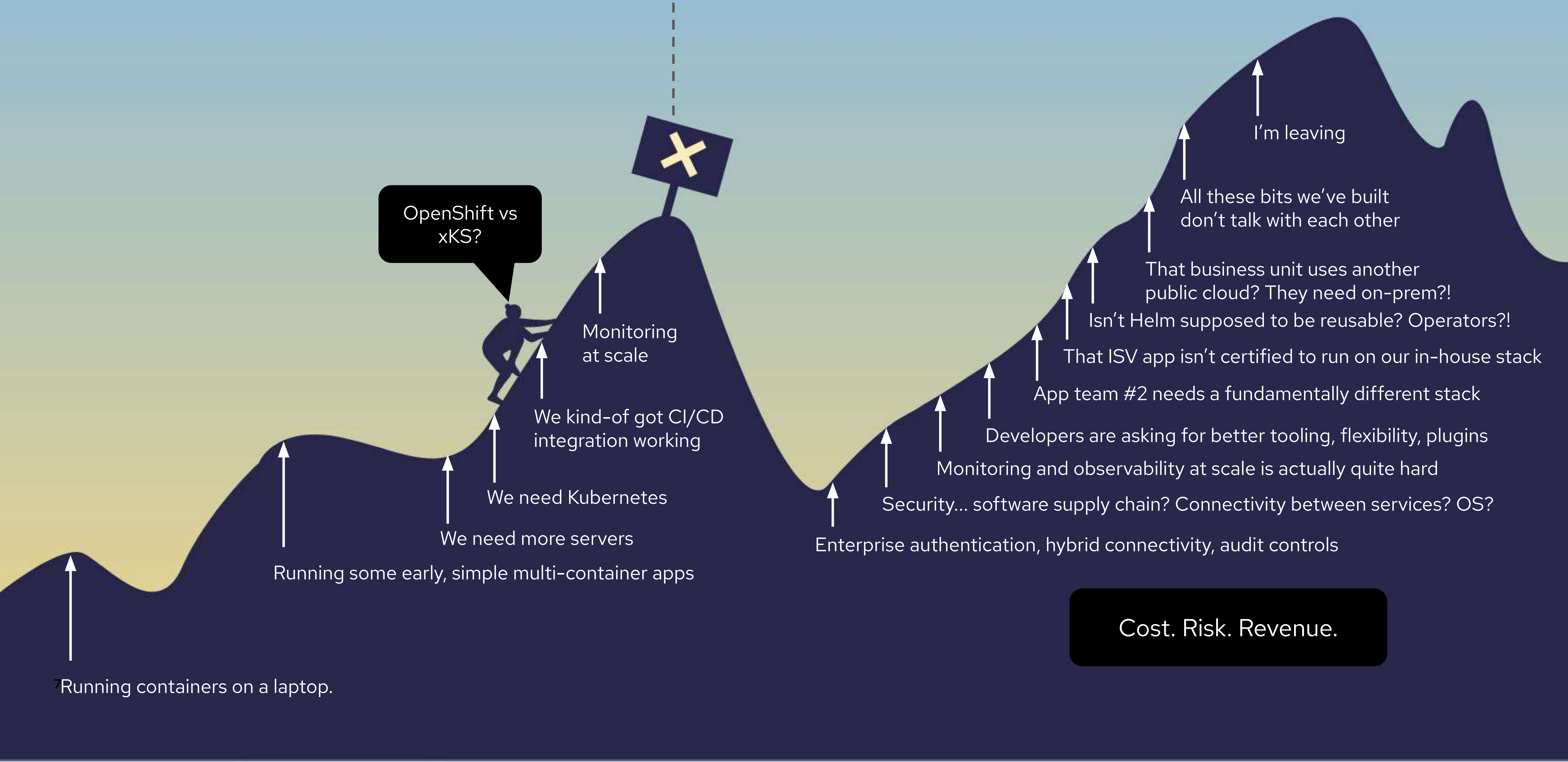
- OpenShift, ROSA, ARO: application platforms?
- Data: K8S is boring? Unique Value for the Hybrid Cloud
- Do not build CI/CD pipelines: Supply Chain Levels for Software Artifacts
- OpenShift is not an island: dev lifecycle with AWS/Azure Managed Services
- Microservices "patterns" using infrastructure?

What we'll discuss today

- Serverless, but across clouds?
- API Management vs. Service Mesh
- Shift left in practice?
- Mission critical apps?
- Where to find useful information for developers

OpenShift, ROSA, ARO: Application Platforms?

Cloud Based App Dev for a Developer



Running containers on a laptop.

Running some early, simple multi-container apps

We need more servers

We need Kubernetes

We kind-of got CI/CD integration working

Monitoring at scale

Enterprise authentication, hybrid connectivity, audit controls

Security… software supply chain? Connectivity between services? OS?

Monitoring and observability at scale is actually quite hard

Developers are asking for better tooling, flexibility, plugins

App team #2 needs a fundamentally different stack

That ISV app isn’t certified to run on our in-house stack

Isn’t Helm supposed to be reusable? Operators?!

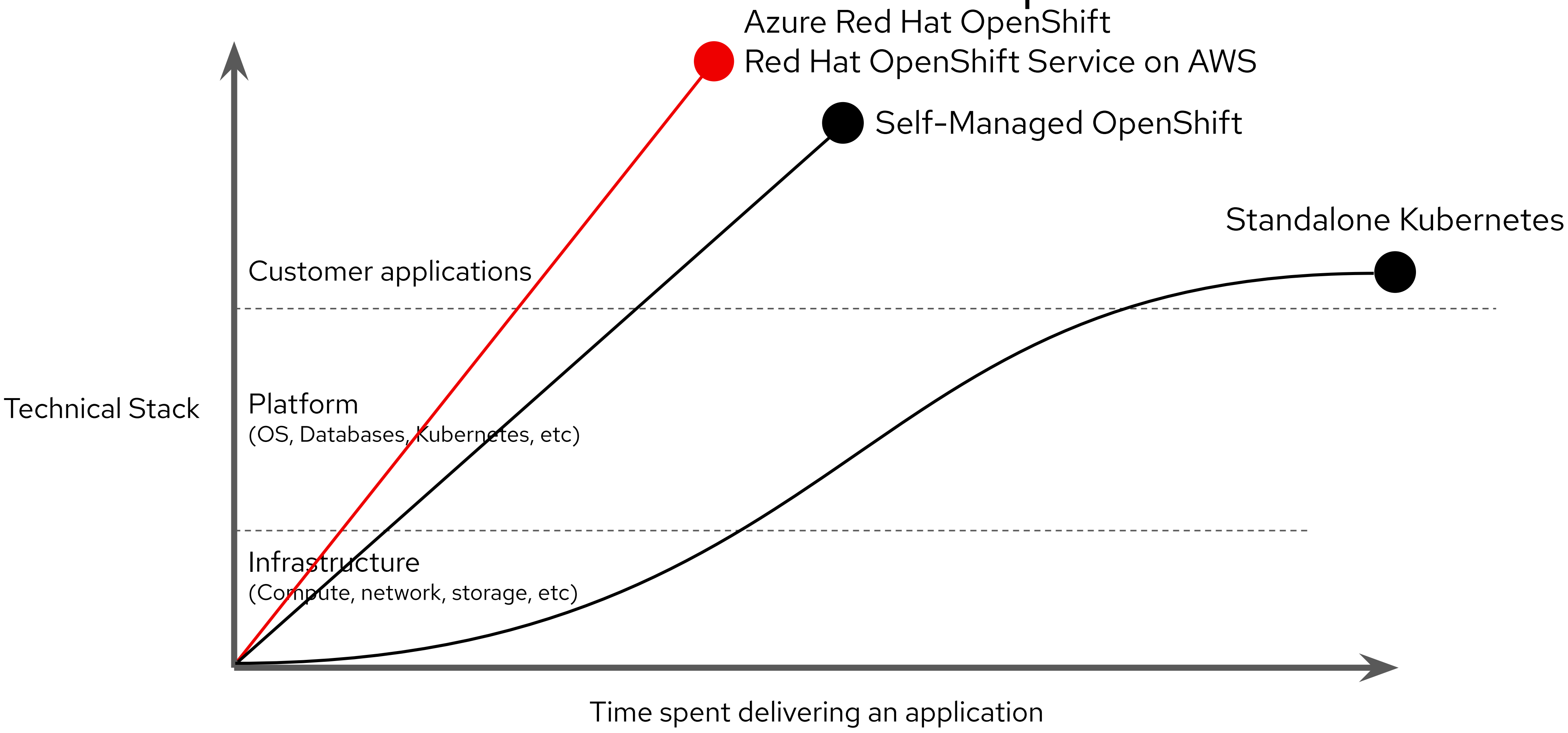
That business unit uses another public cloud? They need on-prem?!

All these bits we’ve built don’t talk with each other

I’m leaving

Cost. Risk. Revenue.

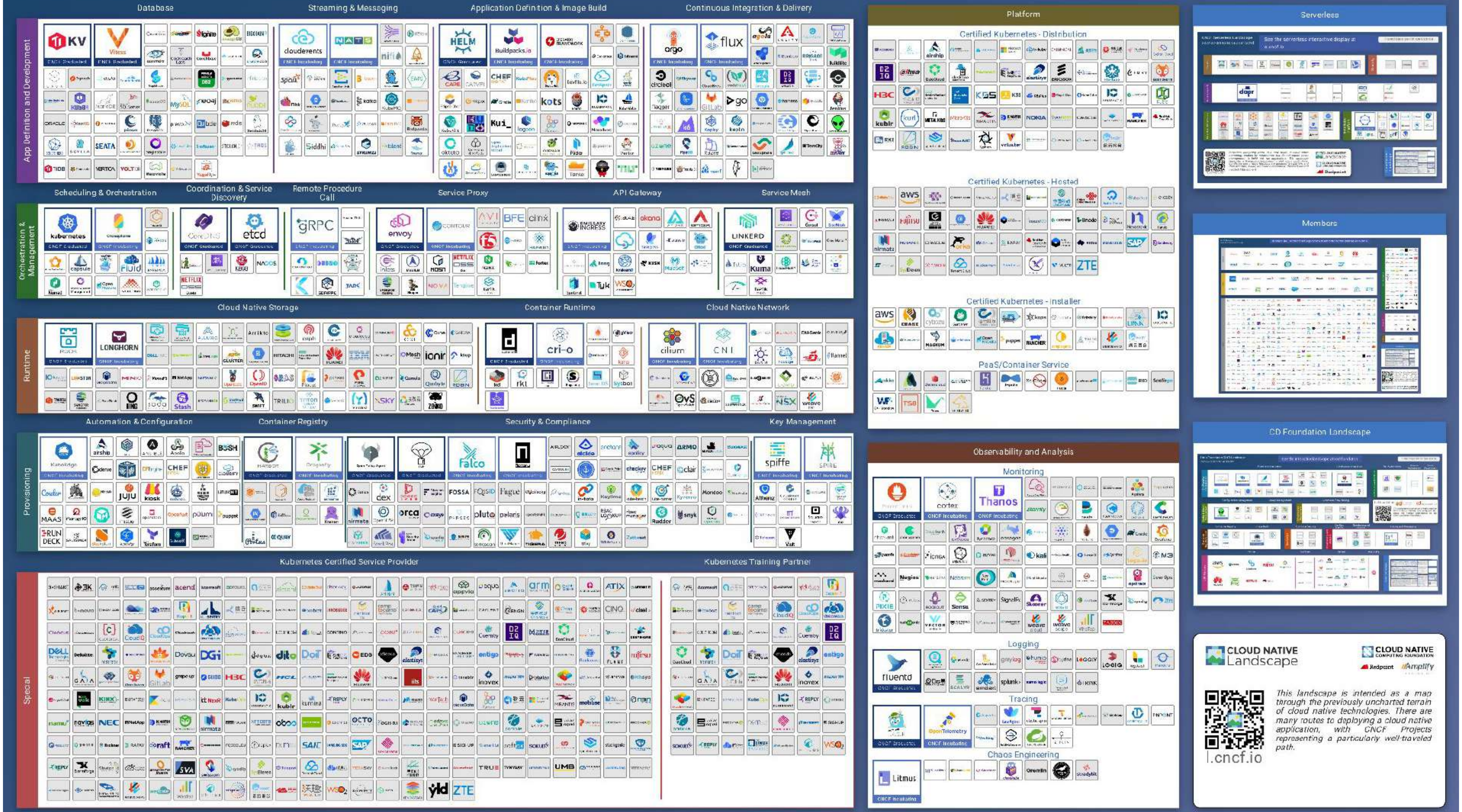
Time to value with OpenShift



CNCF Cloud Native Landscape
2022-02-03T13:00:56Z 031bcf4d

Overwhelmed? Please see the CNCF Trail Map. That and the interactive landscape are at l.cncf.io

Greyed logos are not open source



App Definition and Development: Database; Streaming & Messaging; Application Definition & Image Build; Continuous Integration & Delivery

Orchestration & Management: Scheduling & Orchestration; Coordination & Service Discovery; Remote Procedure Call; Service Proxy; API Gateway; Service Mesh

Runtime: Cloud Native Storage; Container Runtime; Cloud Native Network

Provisioning: Automation & Configuration; Container Registry; Security & Compliance; Key Management

Special: Kubernetes Certified Service Provider; Kubernetes Training Partner

Platform: Certified Kubernetes - Distribution; Certified Kubernetes - Hosted; Certified Kubernetes - Installer; PaaS/Container Service

Observability and Analysis: Monitoring; Logging; Tracing; Chaos Engineering

Serverless

Members

CD Foundation Landscape

CLOUD NATIVE Landscape

CLOUD NATIVE COMPUTING FOUNDATION

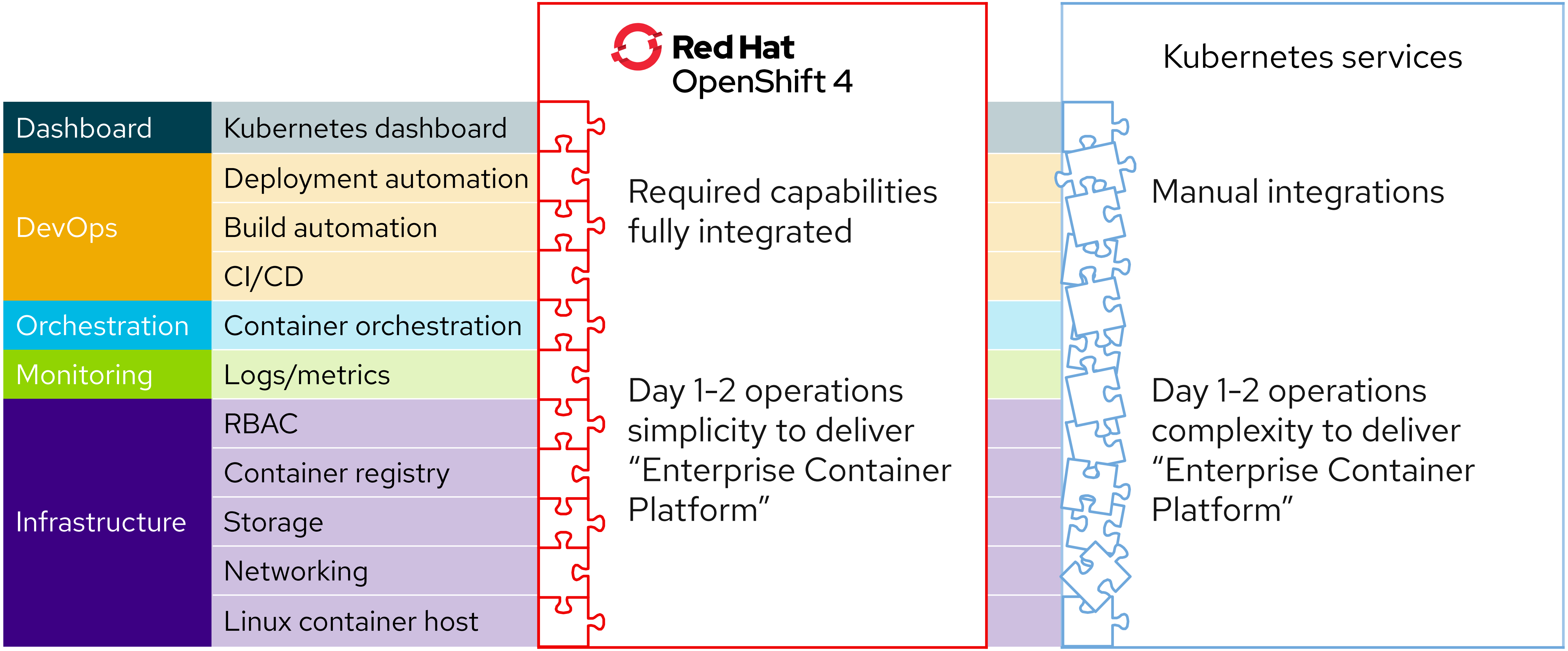
Redpoint

Amplify

This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.

l.cncf.io

OpenShift offers functionality fully integrated



Dashboard	Kubernetes dashboard
DevOps	Deployment automation
	Build automation
	CI/CD
Orchestration	Container orchestration
Monitoring	Logs/metrics
Infrastructure	RBAC
	Container registry
	Storage
	Networking
	Linux container host

Red Hat OpenShift 4

Required capabilities fully integrated

Day 1–2 operations simplicity to deliver “Enterprise Container Platform”

Kubernetes services

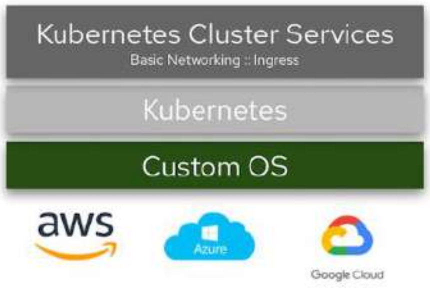
Manual integrations

Day 1–2 operations complexity to deliver “Enterprise Container Platform”

xKS vs OCP vs. Managed OpenShift

It’s important to understand the apples to apples comparison

The Engine



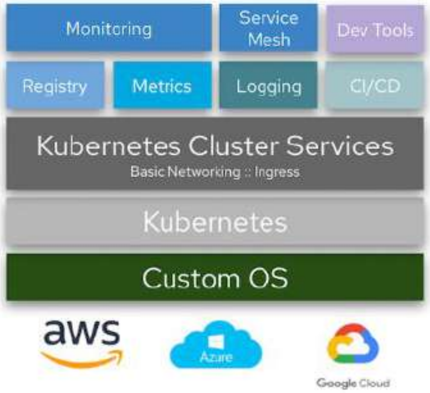
Kubernetes Cluster Services
Basic Networking :: Ingress

Kubernetes

Custom OS

xKS

The Parts



Monitoring | Service Mesh | Dev Tools

Registry | Metrics | Logging | CI/CD

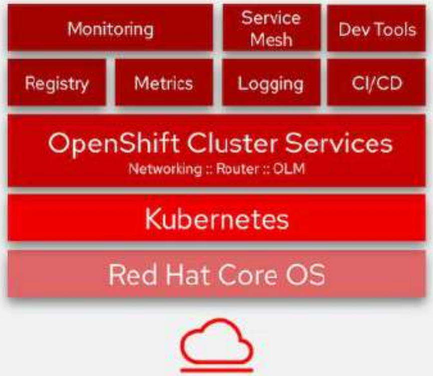
Kubernetes Cluster Services
Basic Networking :: Ingress

Kubernetes

Custom OS

xKS PLUS “NATIVE” SERVICES

The Assembled Car



Monitoring | Service Mesh | Dev Tools

Registry | Metrics | Logging | CI/CD

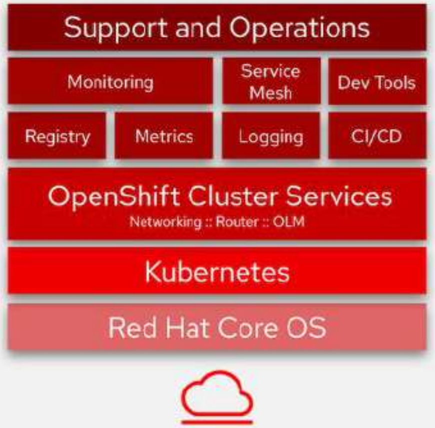
OpenShift Cluster Services
Networking :: Router :: OLM

Kubernetes

Red Hat Core OS

OPENSHIFT PLATFORM

The Full Service



Support and Operations

Monitoring | Service Mesh | Dev Tools

Registry | Metrics | Logging | CI/CD

OpenShift Cluster Services
Networking :: Router :: OLM

Kubernetes

Red Hat Core OS

MANAGED OPENSHIFT PLATFORM

Azure Red Hat OpenShift is a turnkey application platform

Integrated tools and services for faster application development and delivery

Red Hat OpenShift Service Mesh with **Istio** to connect, secure, and observe services

Red Hat OpenShift Serverless with Knative to provide hybrid serverless, FaaS, & event-driven architectures

Red Hat OpenShift pipelines with Tekton to provide Kubernetes-native CI/CD pipelines

argo

Red Hat OpenShift GitOps with **ArgoCD** to provide declarative GitOps based continuous delivery

Red Hat OpenShift builds with Shipwright to build images from code using S2I + other & integrate with Github actions

Red Hat Runtimes

Red Hat Runtimes, including Spring Boot, Quarkus, OpenJDK, JBoss SSO, node.js, Apache Tomcat, Apache HTTP, and .NET

Red Hat OpenShift developer console & CLI enhancements to improve dev experience

CodeReady Workspaces with Eclipse Che for cloud- native development & collaboration

Red Hat OpenShift **IDE plugin** integrations to meet the developer where they are

OpenShift developer sandbox and local cluster enhancements to improve access

Application level observability for developers to build and manage their apps

Kubernetes cluster services

Kubernetes (orchestration)

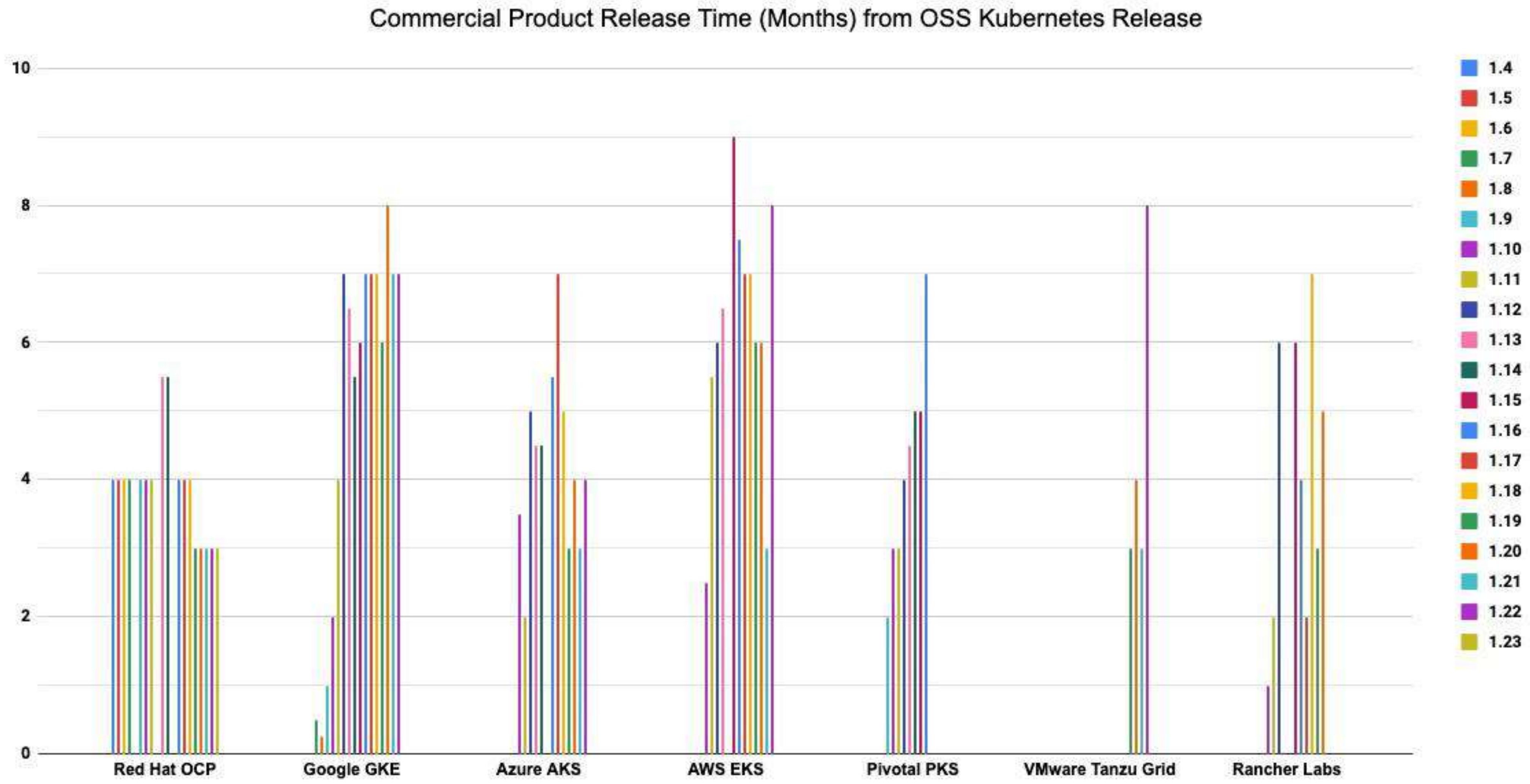
Linux (container host operating system)

Azure

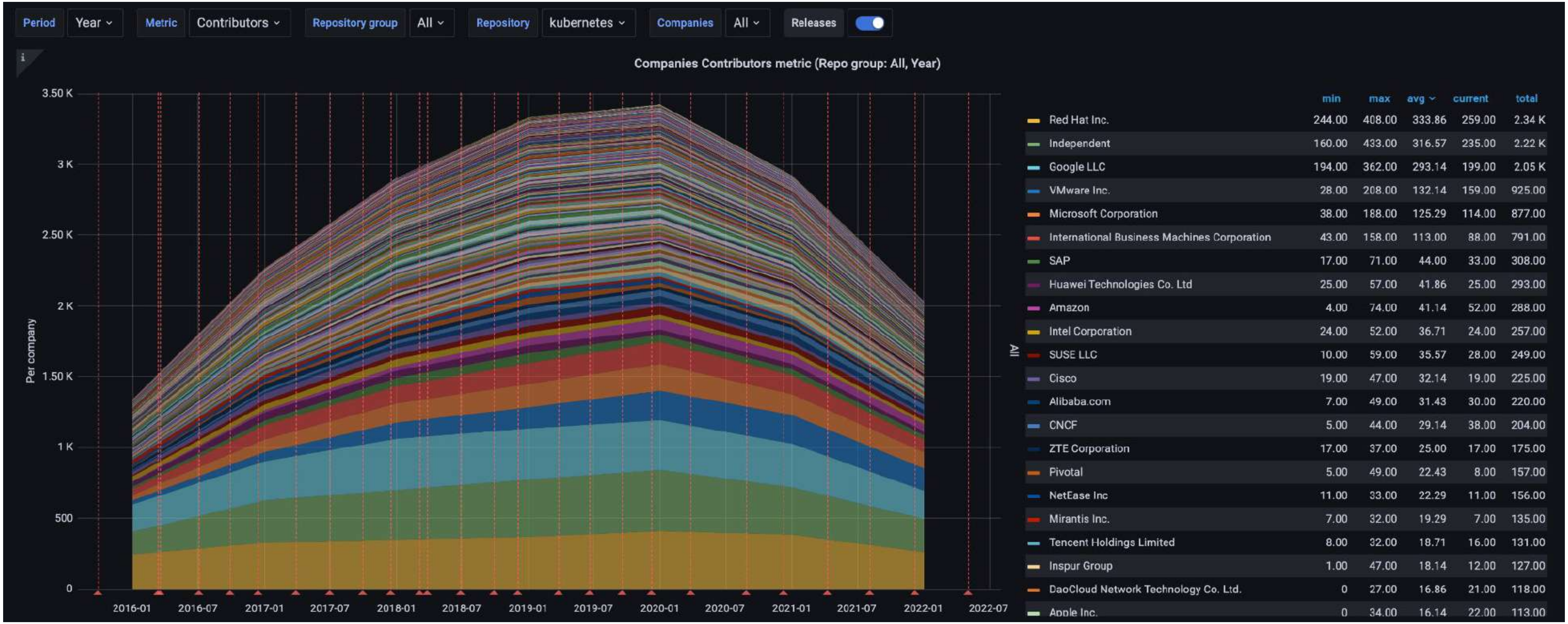
OpenShifts
Unique Value for the Hybrid Cloud from App Dev Perspective

Red Hat contributions extend across the CNCF ecosystem

Red Hat OpenShift has shipped consistently from the earliest Kubernetes releases



Kubernetes Stabilizing since 2020

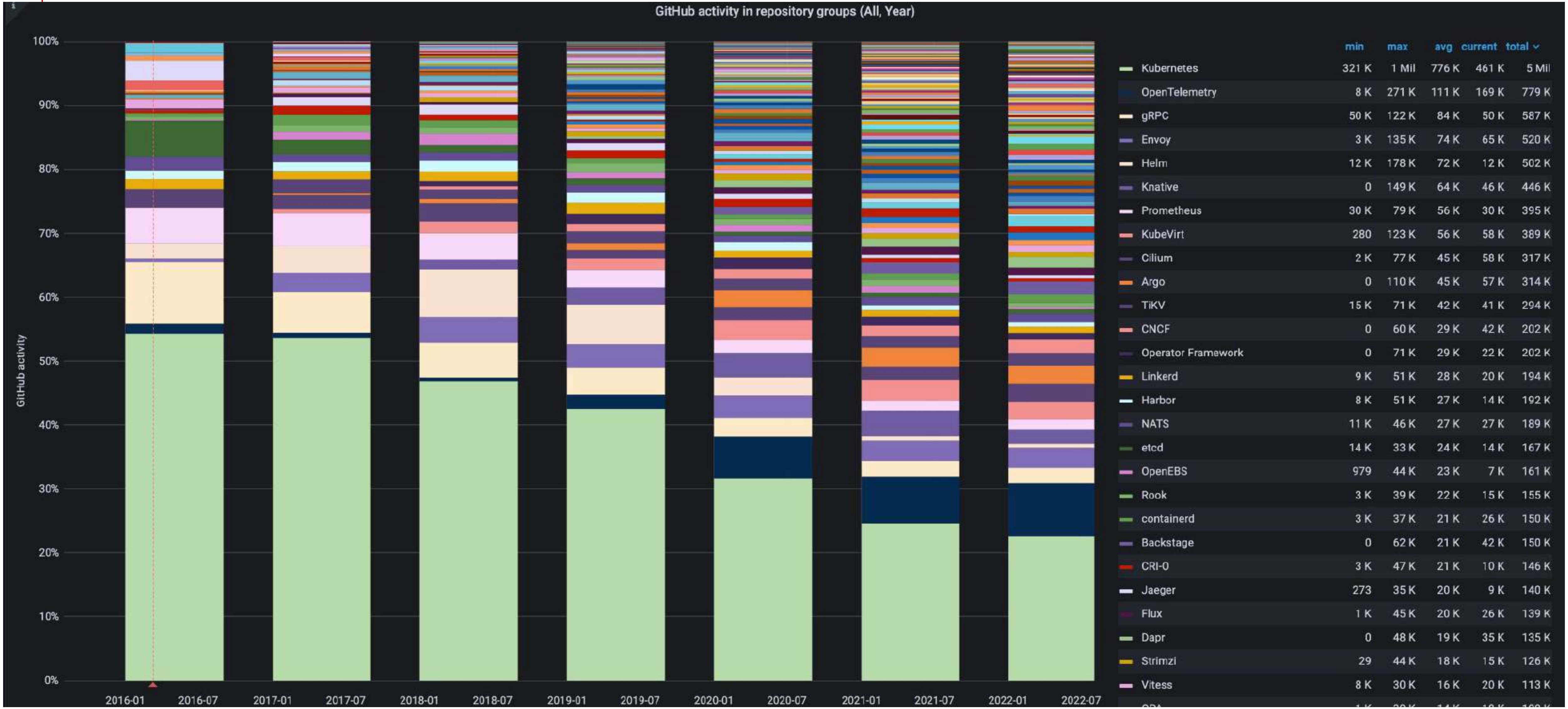


Period: Year · Metric: Contributors · Repository group: All · Repository: kubernetes · Companies: All · Releases: on

Companies Contributors metric (Repo group: All, Year)

	min	max	avg	current	total
▬ Red Hat Inc.	244.00	408.00	333.86	259.00	2.34 K
▬ Independent	160.00	433.00	316.57	235.00	2.22 K
▬ Google LLC	194.00	362.00	293.14	199.00	2.05 K
▬ VMware Inc.	28.00	208.00	132.14	159.00	925.00
▬ Microsoft Corporation	38.00	188.00	125.29	114.00	877.00
▬ International Business Machines Corporation	43.00	158.00	113.00	88.00	791.00
▬ SAP	17.00	71.00	44.00	33.00	308.00
▬ Huawei Technologies Co. Ltd	25.00	57.00	41.86	25.00	293.00
▬ Amazon	4.00	74.00	41.14	52.00	288.00
▬ Intel Corporation	24.00	52.00	36.71	24.00	257.00
▬ SUSE LLC	10.00	59.00	35.57	28.00	249.00
▬ Cisco	19.00	47.00	32.14	19.00	225.00
▬ Alibaba.com	7.00	49.00	31.43	30.00	220.00
▬ CNCF	5.00	44.00	29.14	38.00	204.00
▬ ZTE Corporation	17.00	37.00	25.00	17.00	175.00
▬ Pivotal	5.00	49.00	22.43	8.00	157.00
▬ NetEase Inc	11.00	33.00	22.29	11.00	156.00
▬ Mirantis Inc.	7.00	32.00	19.29	7.00	135.00
▬ Tencent Holdings Limited	8.00	32.00	18.71	16.00	131.00
▬ Inspur Group	1.00	47.00	18.14	12.00	127.00
▬ DaoCloud Network Technology Co. Ltd.	0	27.00	16.86	21.00	118.00
▬ Apple Inc.	0	34.00	16.14	22.00	113.00

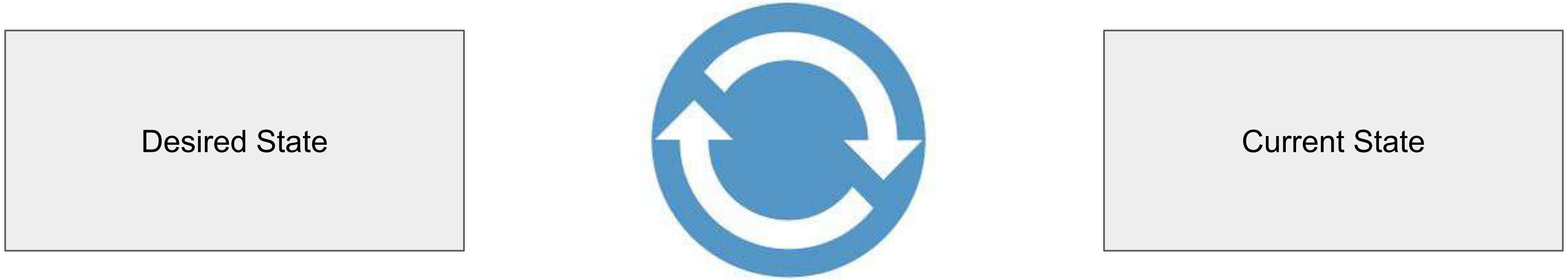
Innovation Focus on the Surrounding Areas



	min	max	avg	current	total ⌄
Kubernetes	321 K	1 Mil	776 K	461 K	5 Mil
OpenTelemetry	8 K	271 K	111 K	169 K	779 K
gRPC	50 K	122 K	84 K	50 K	587 K
Envoy	3 K	135 K	74 K	65 K	520 K
Helm	12 K	178 K	72 K	12 K	502 K
Knative	0	149 K	64 K	46 K	446 K
Prometheus	30 K	79 K	56 K	30 K	395 K
KubeVirt	280	123 K	56 K	58 K	389 K
Cilium	2 K	77 K	45 K	58 K	317 K
Argo	0	110 K	45 K	57 K	314 K
TiKV	15 K	71 K	42 K	41 K	294 K
CNCF	0	60 K	29 K	42 K	202 K
Operator Framework	0	71 K	29 K	22 K	202 K
Linkerd	9 K	51 K	28 K	20 K	194 K
Harbor	8 K	51 K	27 K	14 K	192 K
NATS	11 K	46 K	27 K	27 K	189 K
etcd	14 K	33 K	24 K	14 K	167 K
OpenEBS	979	44 K	23 K	7 K	161 K
Rook	3 K	39 K	22 K	15 K	155 K
containerd	3 K	37 K	21 K	26 K	150 K
Backstage	0	62 K	21 K	42 K	150 K
CRI-O	3 K	47 K	21 K	10 K	146 K
Jaeger	273	35 K	20 K	9 K	140 K
Flux	1 K	45 K	20 K	26 K	139 K
Dapr	0	48 K	19 K	35 K	135 K
Strimzi	29	44 K	18 K	15 K	126 K
Vitess	8 K	30 K	16 K	20 K	113 K

Return to the Beginning

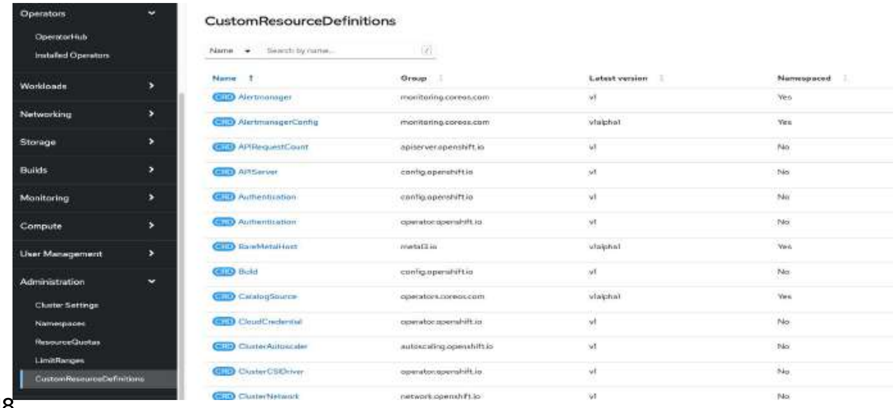
Kubernetes Declarative State


Desired State

Current State

Third Party Resource Definition Red Hat 2016
Operators CoreOS 2016
Custom Resource Definition Red Hat 2017
Red Hat Acquires CoreOS 2018
Operator SDK 2018
OpenShift 4 Released 2019

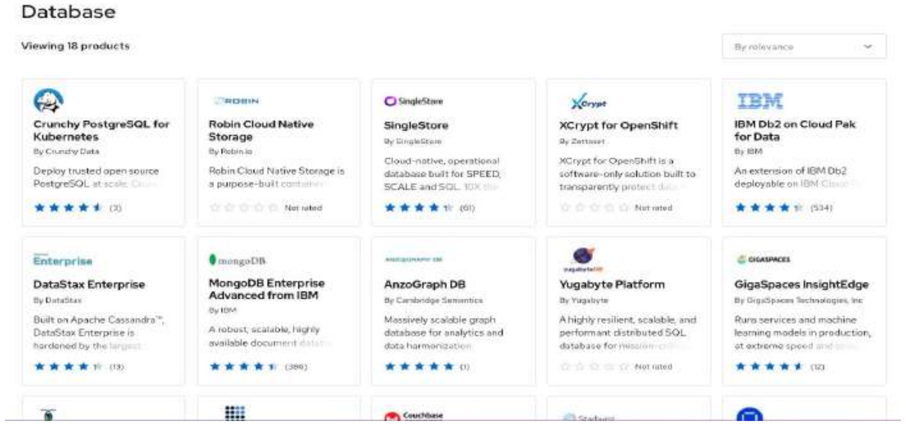
Platform



CustomResourceDefinitions

Name	Group	Latest version	Namespaced
CRD Alertmanager	monitoring.coreos.com	v1	Yes
CRD AlertmanagerConfig	monitoring.coreos.com	v1alpha1	Yes
CRD APIRequestCount	apiserver.openshift.io	v1	No
CRD APIServer	config.openshift.io	v1	No
CRD Authentication	config.openshift.io	v1	No
CRD Authentication	operator.openshift.io	v1	No
CRD BareMetalHost	metal3.io	v1alpha1	Yes
CRD Build	config.openshift.io	v1	No
CRD CatalogSource	operators.coreos.com	v1alpha1	Yes
CRD CloudCredential	operator.openshift.io	v1	No
CRD ClusterAutoscaler	autoscaling.openshift.io	v1	No
CRD ClusterCSIDriver	operator.openshift.io	v1	No
CRD ClusterNetwork	network.openshift.io	v1	No

Applications



Database

Viewing 18 products

Crunchy PostgreSQL for Kubernetes
By Crunchy Data
Deploy trusted open source PostgreSQL at scale. Crun
★★★★½ (3)

Robin Cloud Native Storage
By Robin.io
Robin Cloud Native Storage is a purpose-built container
☆☆☆☆☆ Not rated

SingleStore
By SingleStore
Cloud-native, operational database built for SPEED, SCALE and SQL. 10X the
★★★★½ (61)

XCrypt for OpenShift
By Zettaset
XCrypt for OpenShift is a software-only solution built to transparently protect data
☆☆☆☆☆ Not rated

IBM Db2 on Cloud Pak for Data
By IBM
An extension of IBM Db2 deployable on IBM Cloud
★★★★½ (534)

DataStax Enterprise
By DataStax
Built on Apache Cassandra™, DataStax Enterprise is hardened by the largest
★★★★½ (13)

MongoDB Enterprise Advanced from IBM
By IBM
A robust, scalable, highly available document data
★★★★½ (386)

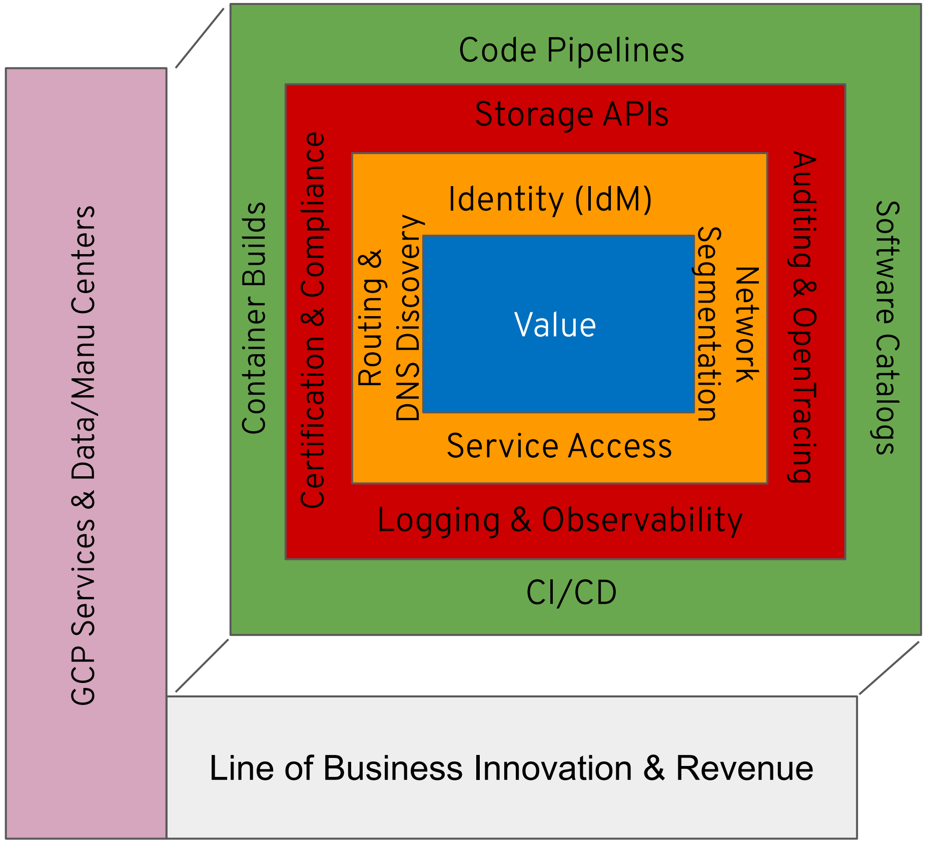
AnzoGraph DB
By Cambridge Semantics
Massively scalable graph database for analytics and data harmonization
★★★★★ (1)

Yugabyte Platform
By Yugabyte
A highly resilient, scalable, and performant distributed SQL database for mission-cri
☆☆☆☆☆ Not rated

GigaSpaces InsightEdge
By GigaSpaces Technologies, Inc
Runs services and machine learning models in production, at extreme speed and
★★★★½ (12)

Evolving Your Platform

Inside-Out Vs Outside-In



Spend More Time Here

- Istio - Service Mesh to connect, secure and observe services
- Knative - Kubernetes-Native Serverless to enable hybrid FaaS
- Tekton - Kubernetes-Native CI/CD for app build & deployment pipelines
- Eclipse Che - Kubernetes-Native IDE for development & collaboration
- Quarkus - Kubernetes-Native Java stack for next-generation apps
- Operator Framework for building managed services on Kubernetes

More than Kubernetes

Kubernetes is Boring (™)

Building a Kubernetes Cloud Native DevOps Services Stack

OpenShift Service Mesh with Istio to connect, secure and observe services

OpenShift Serverless with Knative to enable hybrid Serverless, FaaS & EDA

OpenShift Pipelines with Tekton to provide Kubernetes-Native CI/CD pipelines

GitHub Actions to automate container build and deployments to OpenShift

OpenShift Builds with Shipwright to build images from code using S2I, Buildpacks, and buildah

OpenShift GitOps with ArgoCD to enable declarative GitOps based continuous delivery

Building World Class Developer Tools & Developer Experience in OpenShift

Helm Charts for packaging and distributing applications on OpenShift

OpenShift Developer Console & CLI enhancements to improve dev experience

CodeReady Workspaces with Eclipse Che for cloud native development & collaboration

Complete IDE plugin integrations to meet the developer where they are

OpenShift developer sandbox and local cluster enhancements to improve access

Observability that enables app monitoring for developers on OpenShift

Install and Form Factors

Pick Your Operational Stance

IPI

- Most like *KS
- Carves out what it needs
- Tries to load all Infra Automations
- Let’s LOB get self service

UPI

- Old school unlimited options
- You choose Infra automations
- Integrate ISV solutions
- Bring your own hosts

Assisted

- Hosted Q&A
- Designed for Appliances
- Agnostic to Infra
- ISO Driven

ACM-Hive

- Install 1,000 of clusters
- Manage them from gitOps
- CR/Yaml Driven with ACM UX
- Automatically flow into governance and policy

HyperShift

(GA Target July 2022)

- Cloud Provider Level
- Control Plane Pods in Namespaces
- External to the Cluster Resources
- Not self managed

Your Pick of Kubernetes Design:

3-Node HA Cluster *(GA)*

Laptop Cluster *(GA)*

Single Node Cluster *(GA)*

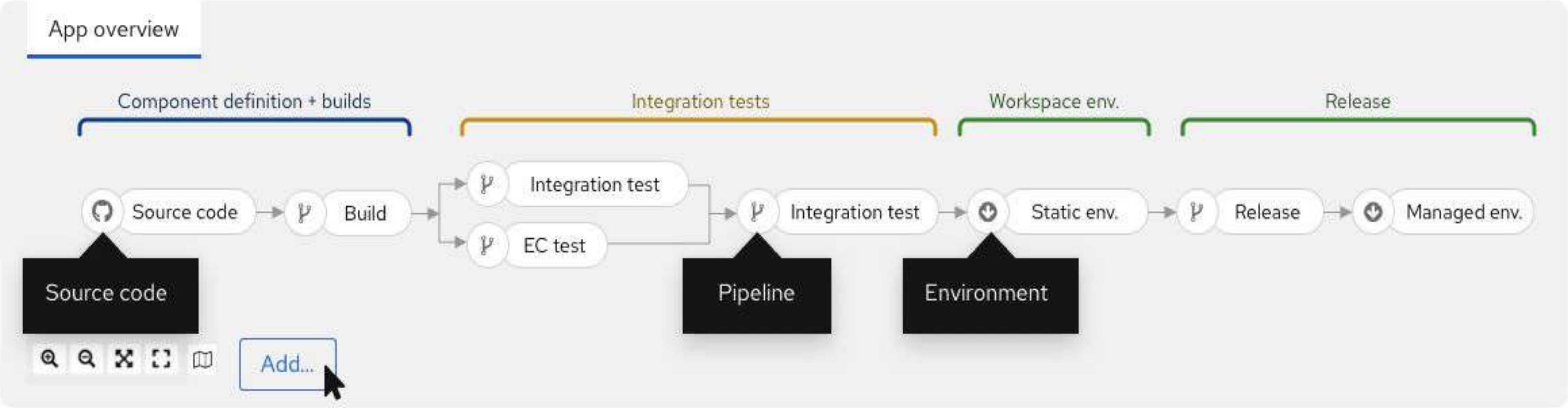
ROSA (AWS) *(GA)*

ARO (AWS) *(GA)*

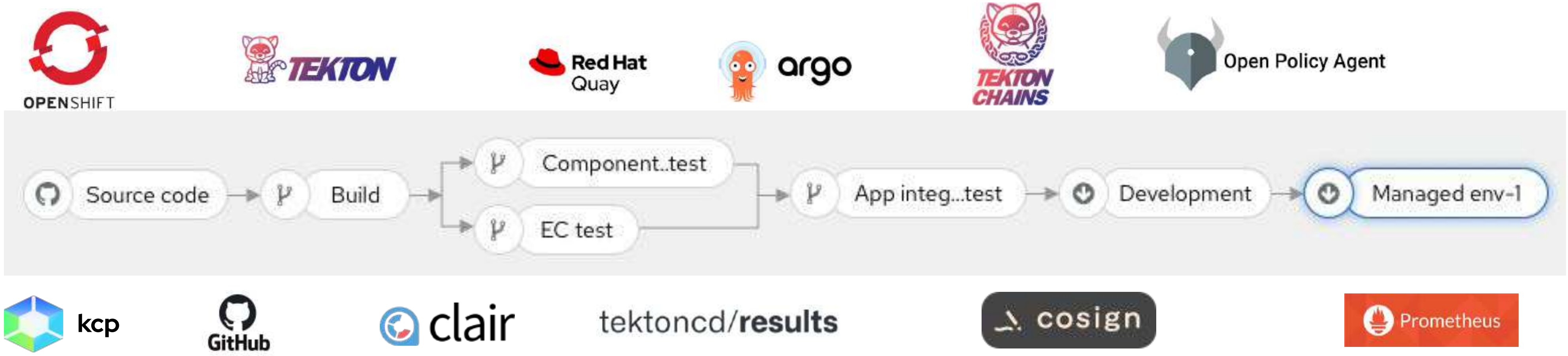
Focus on building applications

instead of building CI / CD systems.

Build Service gives you an out of the box workflow designed to flex for small or large applications.



Easy to use



Because Build Service is a managed service, you can be up and running in minutes. Complicated product integrations are handled for you. Upgrades are continuous and seamless.

Deliver **securely-built images** to a registry, deploy applications to the cloud or to your on-prem OpenShift cluster with just a few steps.

Enterprise Contract

How does is work?

Proof is provided by:

- Tekton chains is used to obtain proof of what happened in a user-defined pipeline.
- Rekor transparency log is used for serialization of TaskRun proof.
- Tekton Chains provides a mechanism to automatically upload signed payloads to a transparency log for off-system verification.

Build Service analyzes records in in the transparency log to verify that

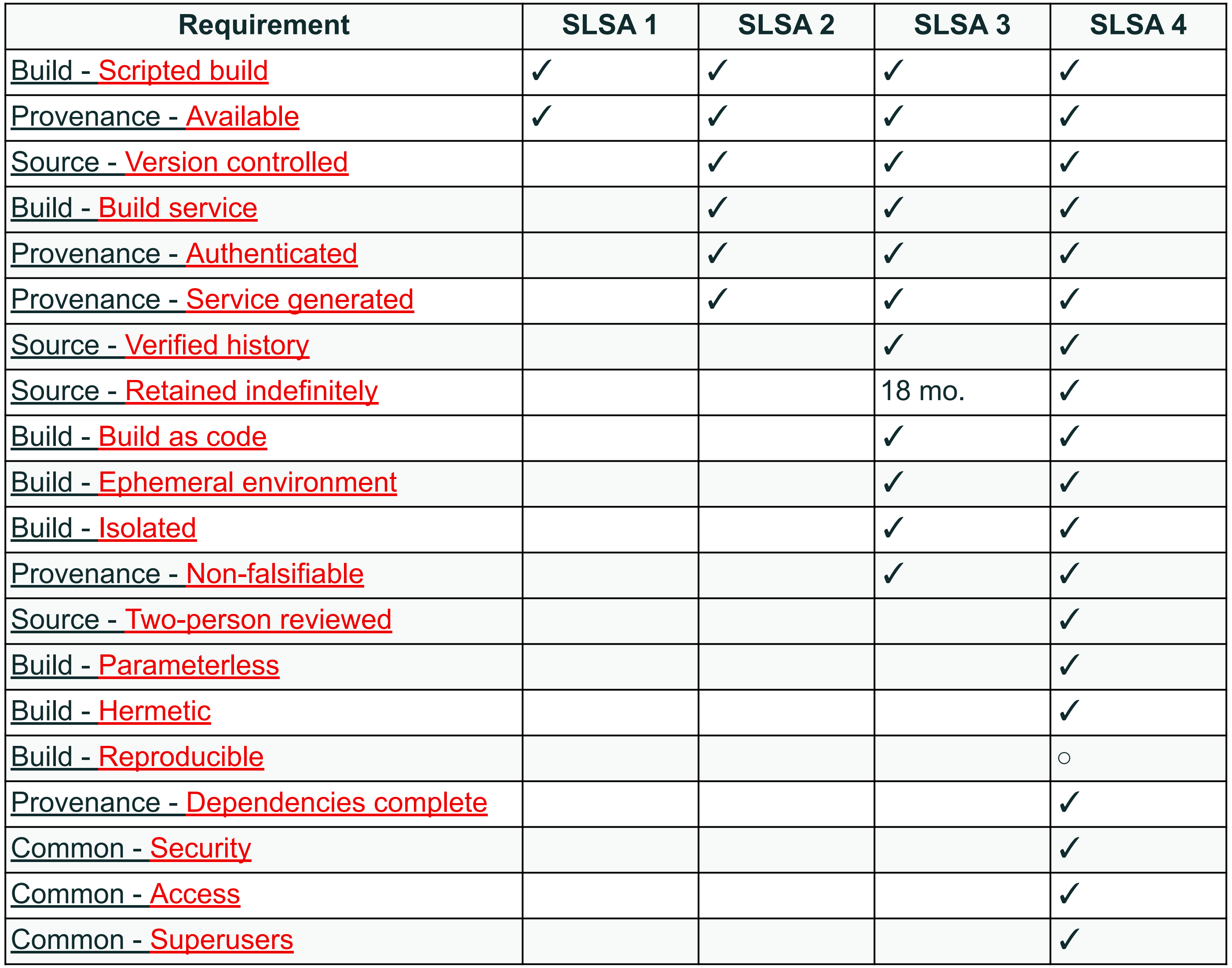
- a particular OCI image was produced by a valid pipeline,
- which was in turn composed of valid TaskRuns,
- which in turn were composed of valid Tasks,
- which in turn were compliant with the organization's enterprise contract.

Supply Chain Levels for Software Artifacts (SLSA)

"SLSA’s four levels are designed to be incremental and actionable, and to protect against specific integrity attacks.

SLSA 4 represents the ideal end state, and the lower levels represent milestones with corresponding integrity guarantees."

Source: https://slsa.dev/spec/v0.1/levels

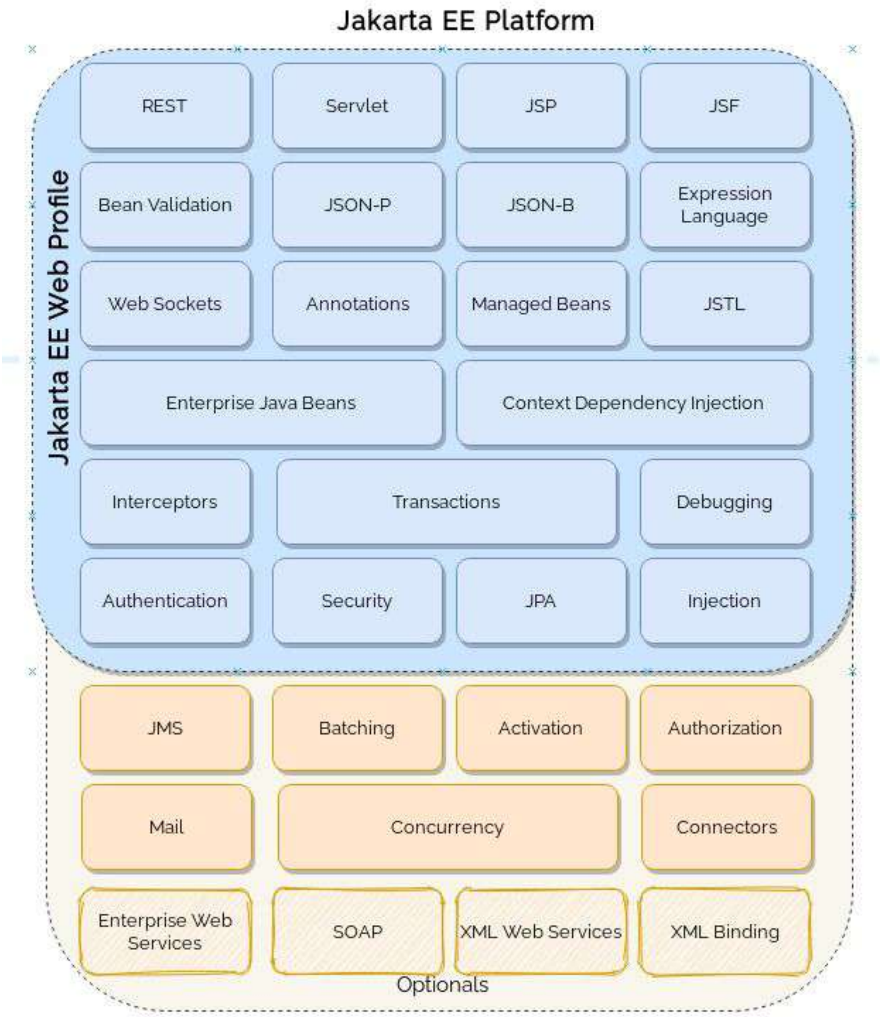


Requirement	SLSA 1	SLSA 2	SLSA 3	SLSA 4
Build - Scripted build	✓	✓	✓	✓
Provenance - Available	✓	✓	✓	✓
Source - Version controlled		✓	✓	✓
Build - Build service		✓	✓	✓
Provenance - Authenticated		✓	✓	✓
Provenance - Service generated		✓	✓	✓
Source - Verified history			✓	✓
Source - Retained indefinitely			18 mo.	✓
Build - Build as code			✓	✓
Build - Ephemeral environment			✓	✓
Build - Isolated			✓	✓
Provenance - Non-falsifiable			✓	✓
Source - Two-person reviewed				✓
Build - Parameterless				✓
Build - Hermetic				✓
Build - Reproducible				○
Provenance - Dependencies complete				✓
Common - Security				✓
Common - Access				✓
Common - Superusers				✓

OpenShift is not an island: use best parts of AWS and Azure

Microservices "patterns" using programming language lock-in?

Monolithic application



Jakarta EE Platform

Jakarta EE Web Profile

REST | Servlet | JSP | JSF

Bean Validation | JSON-P | JSON-B | Expression Language

Web Sockets | Annotations | Managed Beans | JSTL

Enterprise Java Beans | Context Dependency Injection

Interceptors | Transactions | Debugging

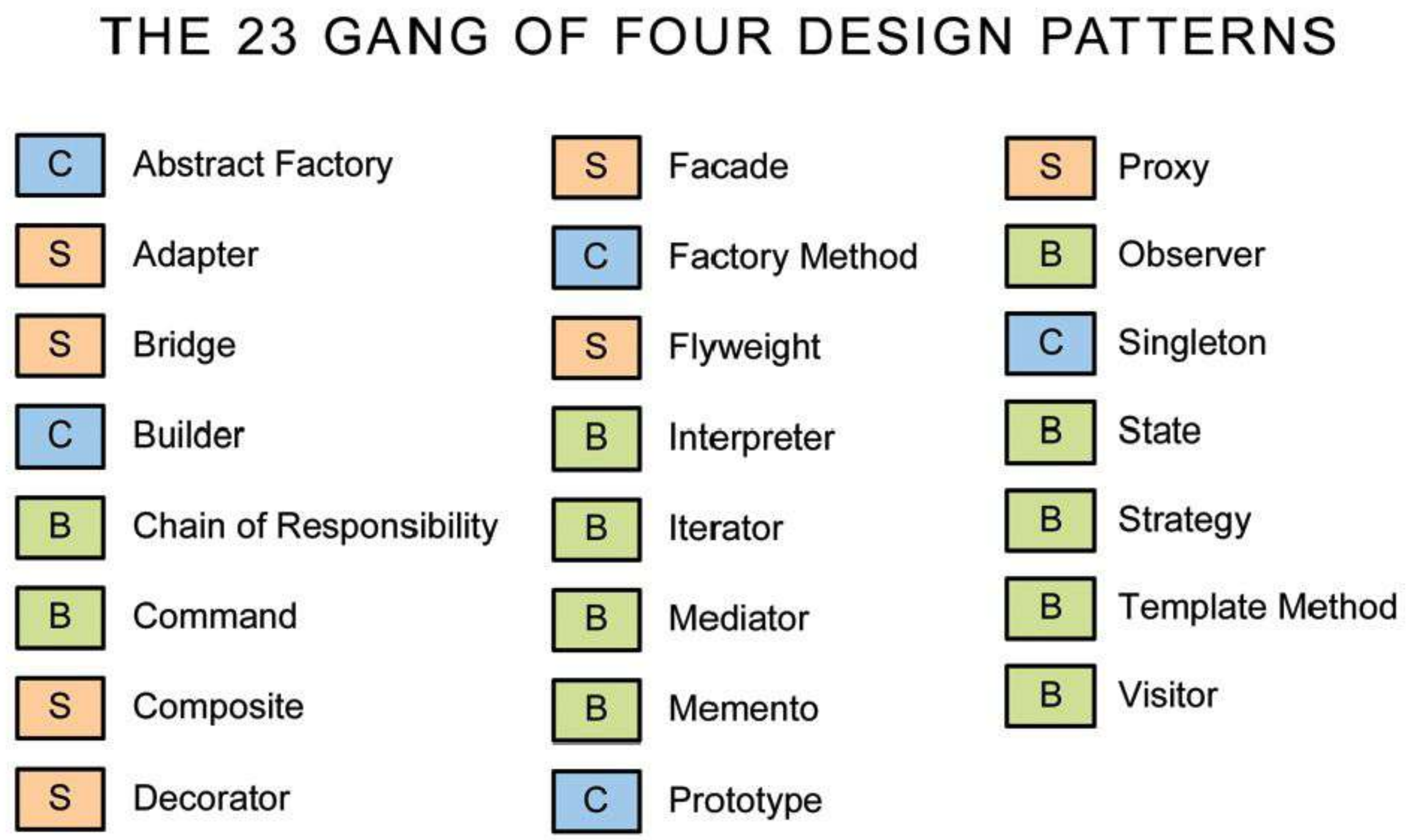
Authentication | Security | JPA | Injection

JMS | Batching | Activation | Authorization

Mail | Concurrency | Connectors

Enterprise Web Services | SOAP | XML Web Services | XML Binding

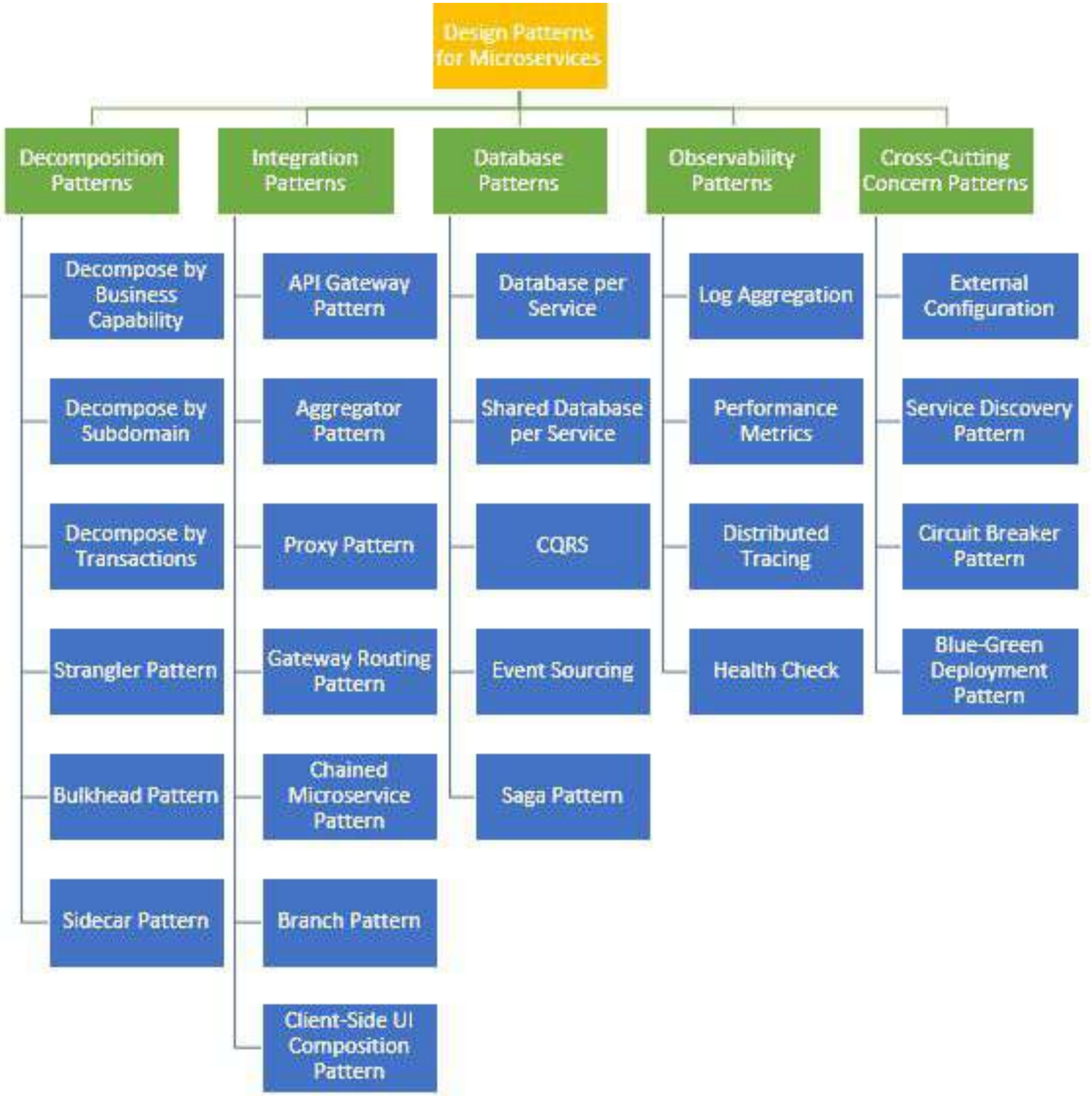
Optionals



THE 23 GANG OF FOUR DESIGN PATTERNS

C	Abstract Factory
S	Adapter
S	Bridge
C	Builder
B	Chain of Responsibility
B	Command
S	Composite
S	Decorator
S	Facade
C	Factory Method
S	Flyweight
B	Interpreter
B	Iterator
B	Mediator
B	Memento
C	Prototype
S	Proxy
B	Observer
C	Singleton
B	State
B	Strategy
B	Template Method
B	Visitor

Microservices in 2022



- Design Patterns for Microservices
  - Decomposition Patterns
    - Decompose by Business Capability
    - Decompose by Subdomain
    - Decompose by Transactions
    - Strangler Pattern
    - Bulkhead Pattern
    - Sidecar Pattern
  - Integration Patterns
    - API Gateway Pattern
    - Aggregator Pattern
    - Proxy Pattern
    - Gateway Routing Pattern
    - Chained Microservice Pattern
    - Branch Pattern
    - Client-Side UI Composition Pattern
  - Database Patterns
    - Database per Service
    - Shared Database per Service
    - CQRS
    - Event Sourcing
    - Saga Pattern
  - Observability Patterns
    - Log Aggregation
    - Performance Metrics
    - Distributed Tracing
    - Health Check
  - Cross-Cutting Concern Patterns
    - External Configuration
    - Service Discovery Pattern
    - Circuit Breaker Pattern
    - Blue-Green Deployment Pattern

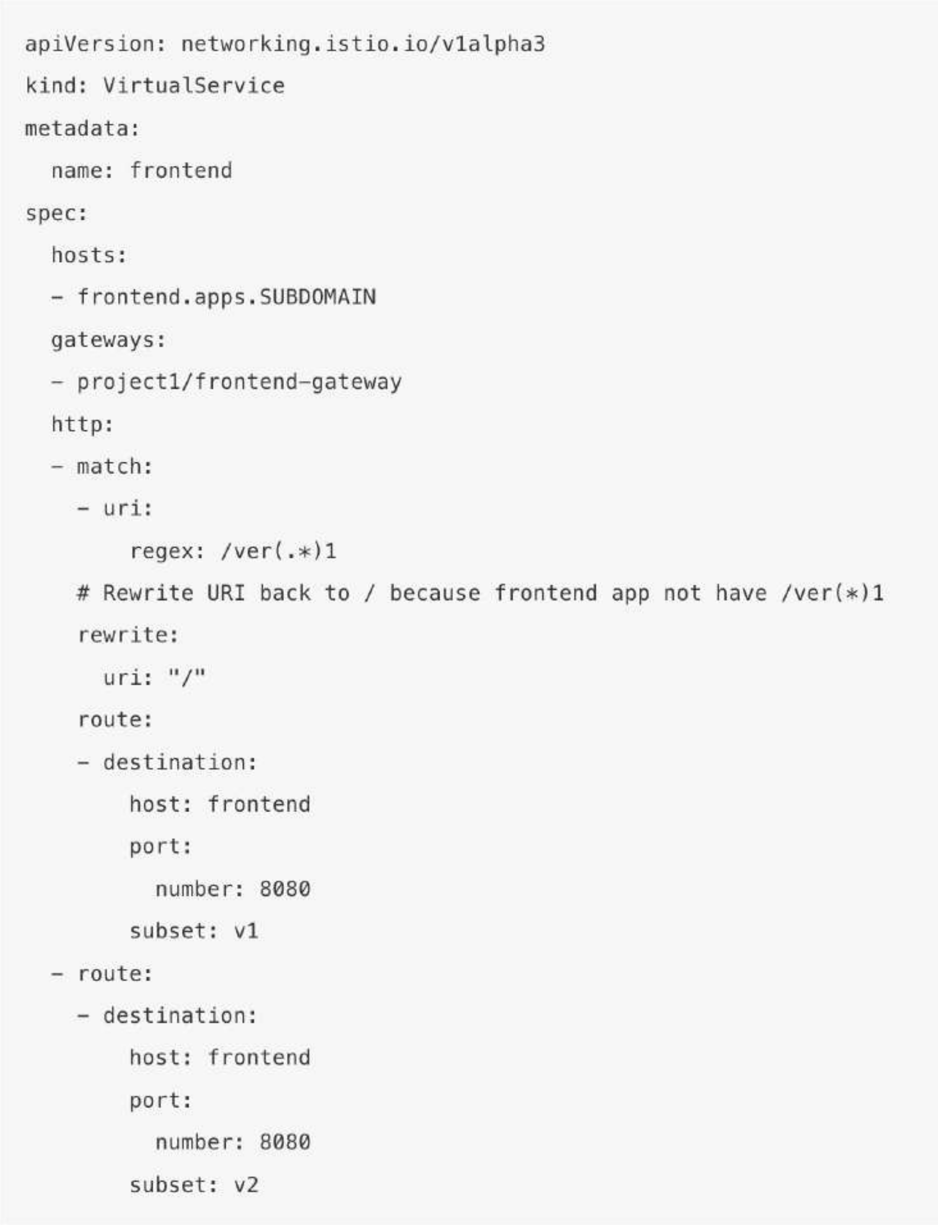
Reduce complexity? Speed up development? Polyglot programming?

Yes, they are parts of the platform

+ Service mesh is **multitenant** in OpenShift (incl. ROSA and ARO)

Example:

Conditional Routing

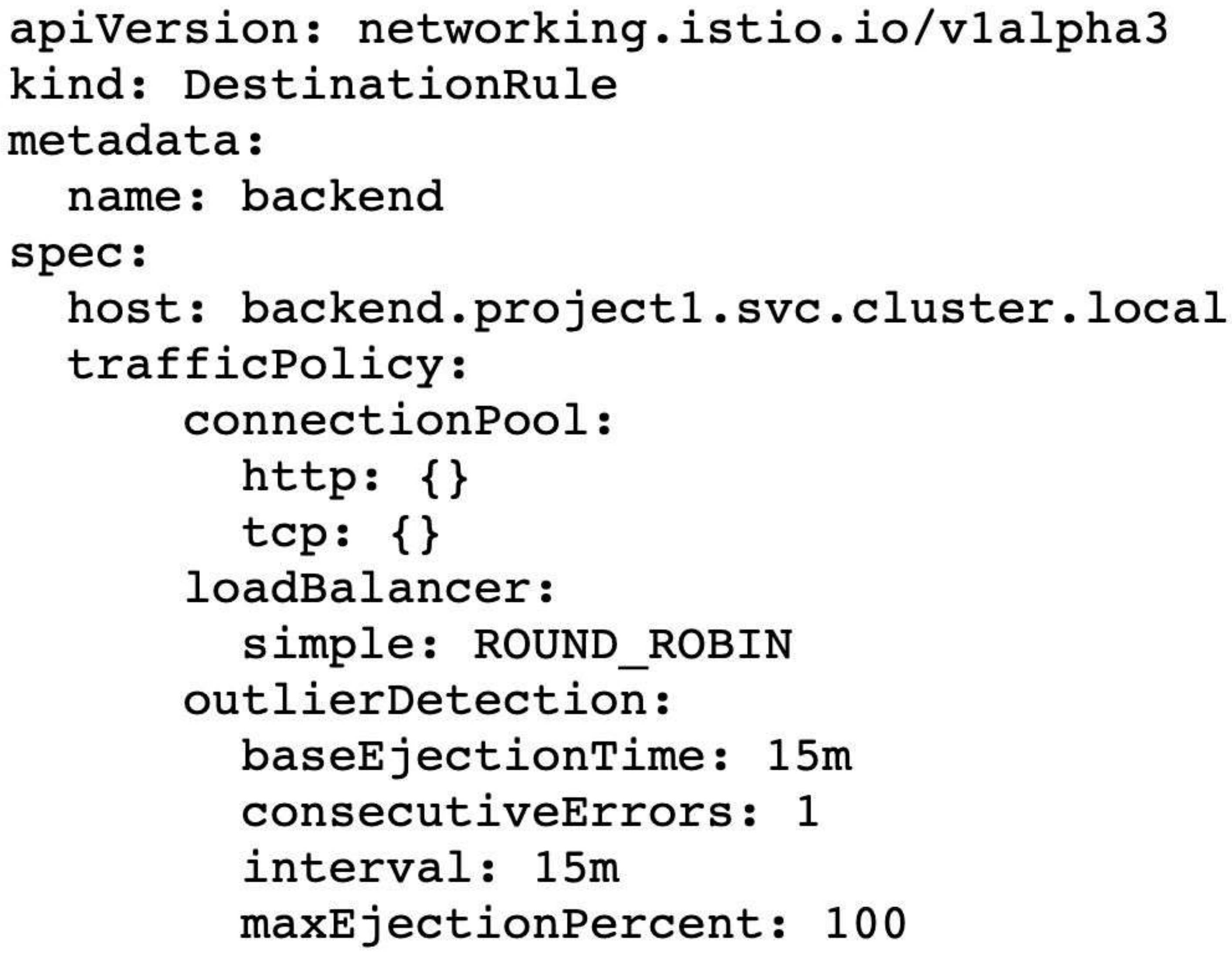


```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: frontend
spec:
  hosts:
  - frontend.apps.SUBDOMAIN
  gateways:
  - project1/frontend-gateway
  http:
  - match:
    - uri:
        regex: /ver(.*)1
    # Rewrite URI back to / because frontend app not have /ver(*)1
    rewrite:
      uri: "/"
    route:
    - destination:
        host: frontend
        port:
          number: 8080
        subset: v1
  - route:
    - destination:
        host: frontend
        port:
          number: 8080
        subset: v2
```

Example:

Circuit Breaker

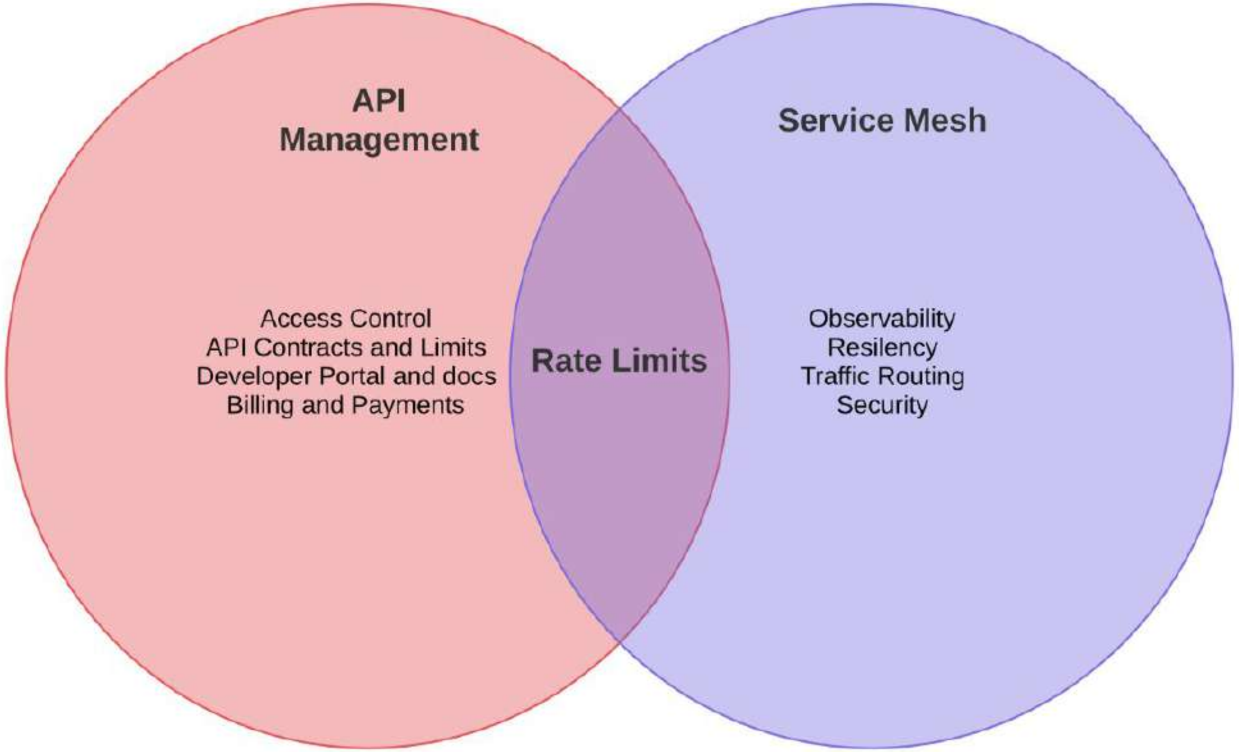
- If found error 1 times (consecutiveErrors)
- then eject that pod from pool for 15 mintues (baseEjectionTime)
- Maximum number of pod that can be ejected is 100% (maxEjectionPercent)
- Check this every 15 min (interval)



```yaml
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: backend
spec:
  host: backend.project1.svc.cluster.local
  trafficPolicy:
      connectionPool:
        http: {}
        tcp: {}
      loadBalancer:
        simple: ROUND_ROBIN
      outlierDetection:
        baseEjectionTime: 15m
        consecutiveErrors: 1
        interval: 15m
        maxEjectionPercent: 100
```

Api-Management VS. Service Mesh?

API Management vs. Service Mesh



https://itnext.io/api-management-and-service-mesh-e7f0e686090e

Whitepaper: https://www.redhat.com/en/resources/api-management-and-service-mesh-checklist

From Microservices to Serverless

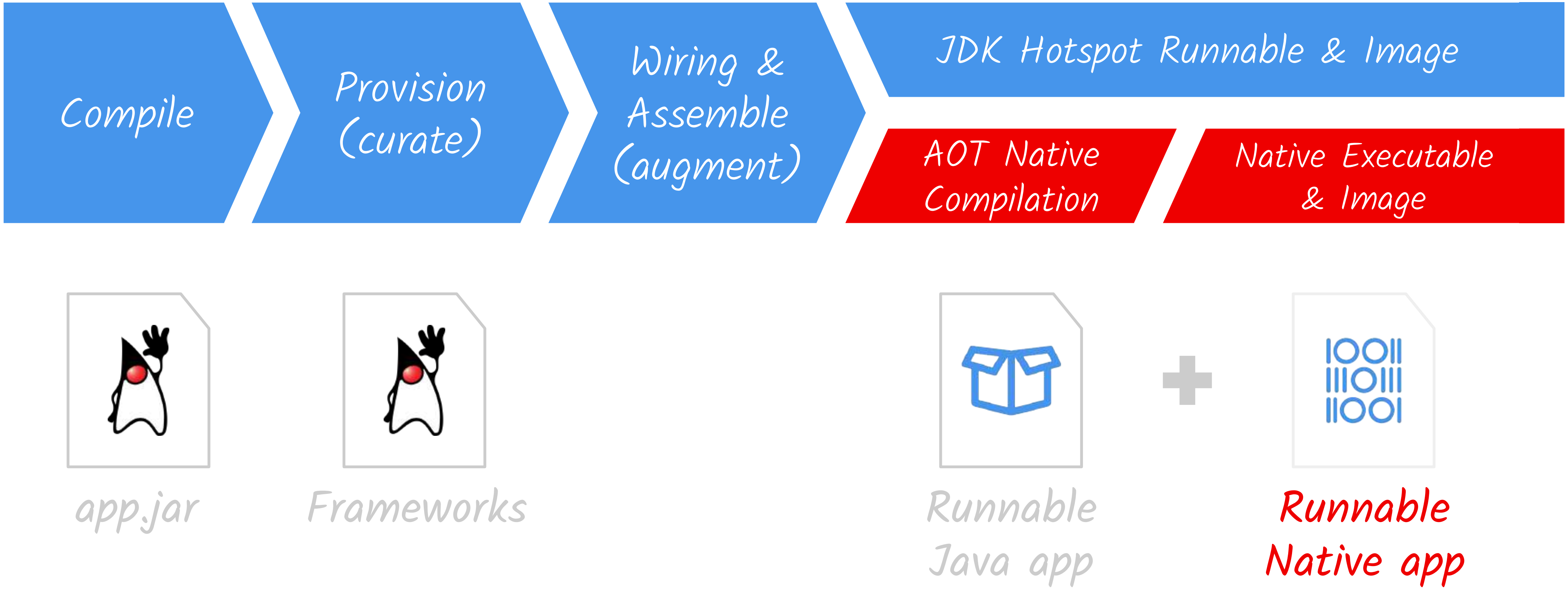
- Use serverless capabilities of the platform
- Code known language and style
- Make sure your function starts fast!*
- *If it is not fast rewrite to make it fast in other language

https://github.com/redhat-mw-demos/serverless-runtimes-demo

Serverless, but across clouds?

reduced lock-in

Quarkus Native Compilation



Compile

Provision (curate)

Wiring & Assemble (augment)

JDK Hotspot Runnable & Image

AOT Native Compilation

Native Executable & Image

app.jar

Frameworks

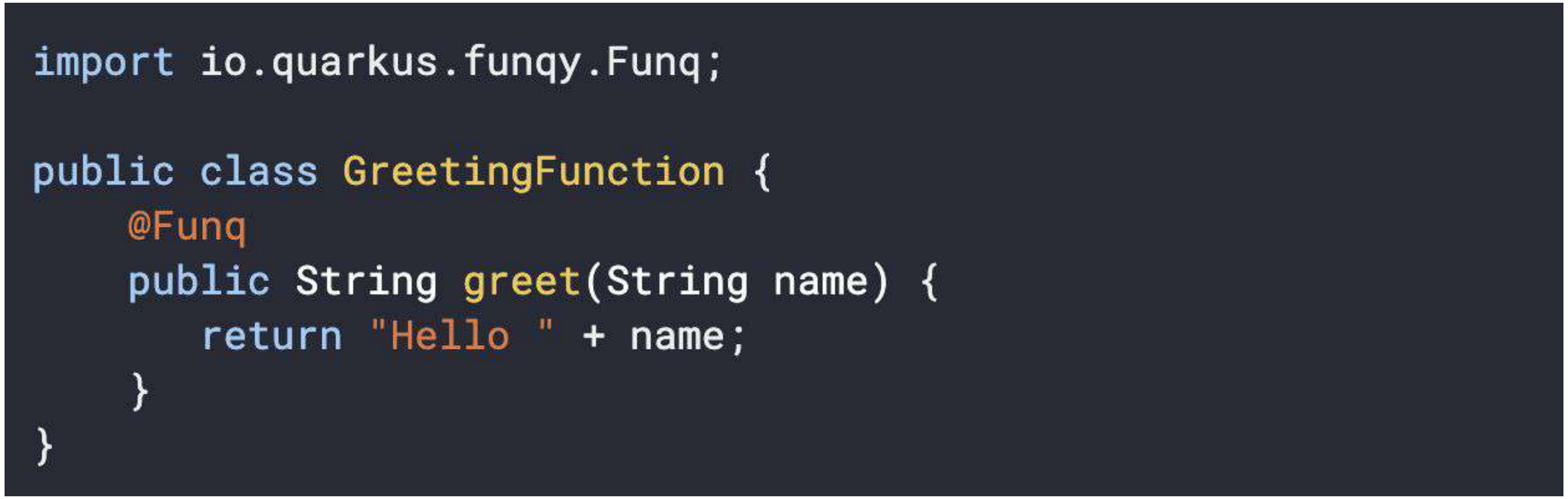
Runnable Java app

Runnable Native app

Quarkus Funqy

🤘 A portable Java API to write functions

🤘 Deployable to various FaaS environments or a standalone service



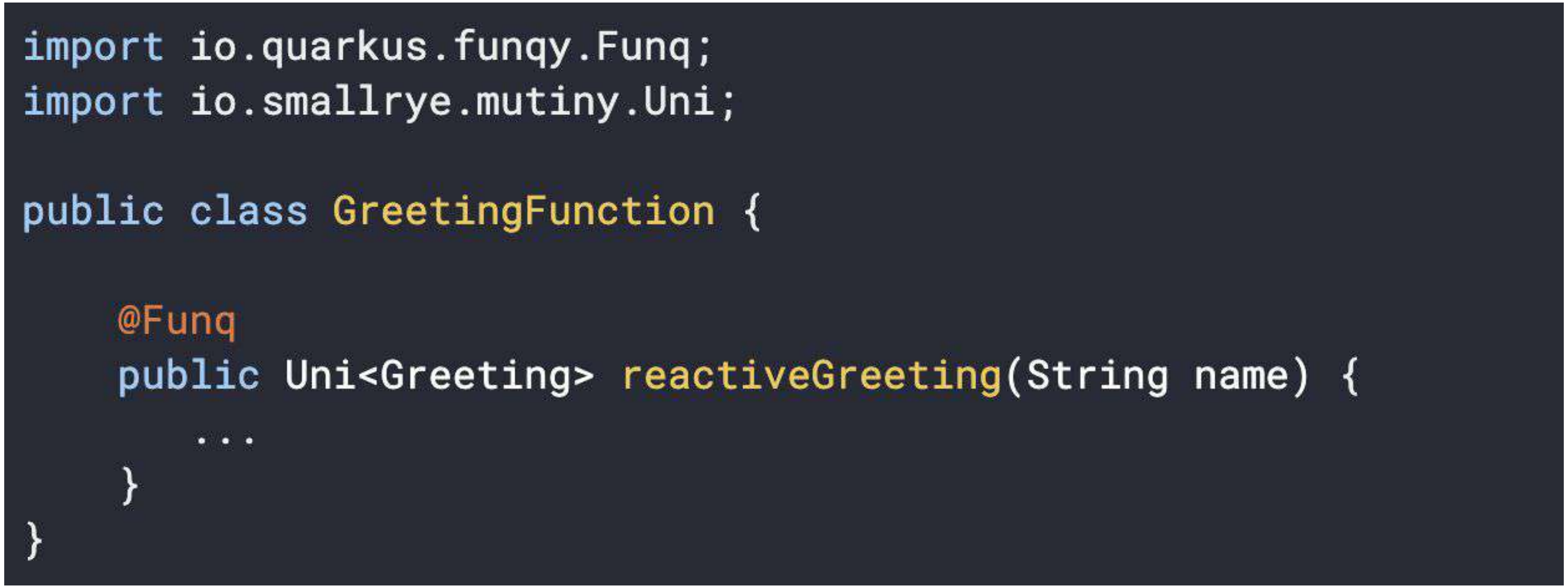
```
import io.quarkus.funqy.Funq;

public class GreetingFunction {
    @Funq
    public String greet(String name) {
        return "Hello " + name;
    }
}
```

Quarkus Funqy

🤘 Async Reactive Types

🤘 Supports the Smallrye Mutiny Uni reactive type as a return typ



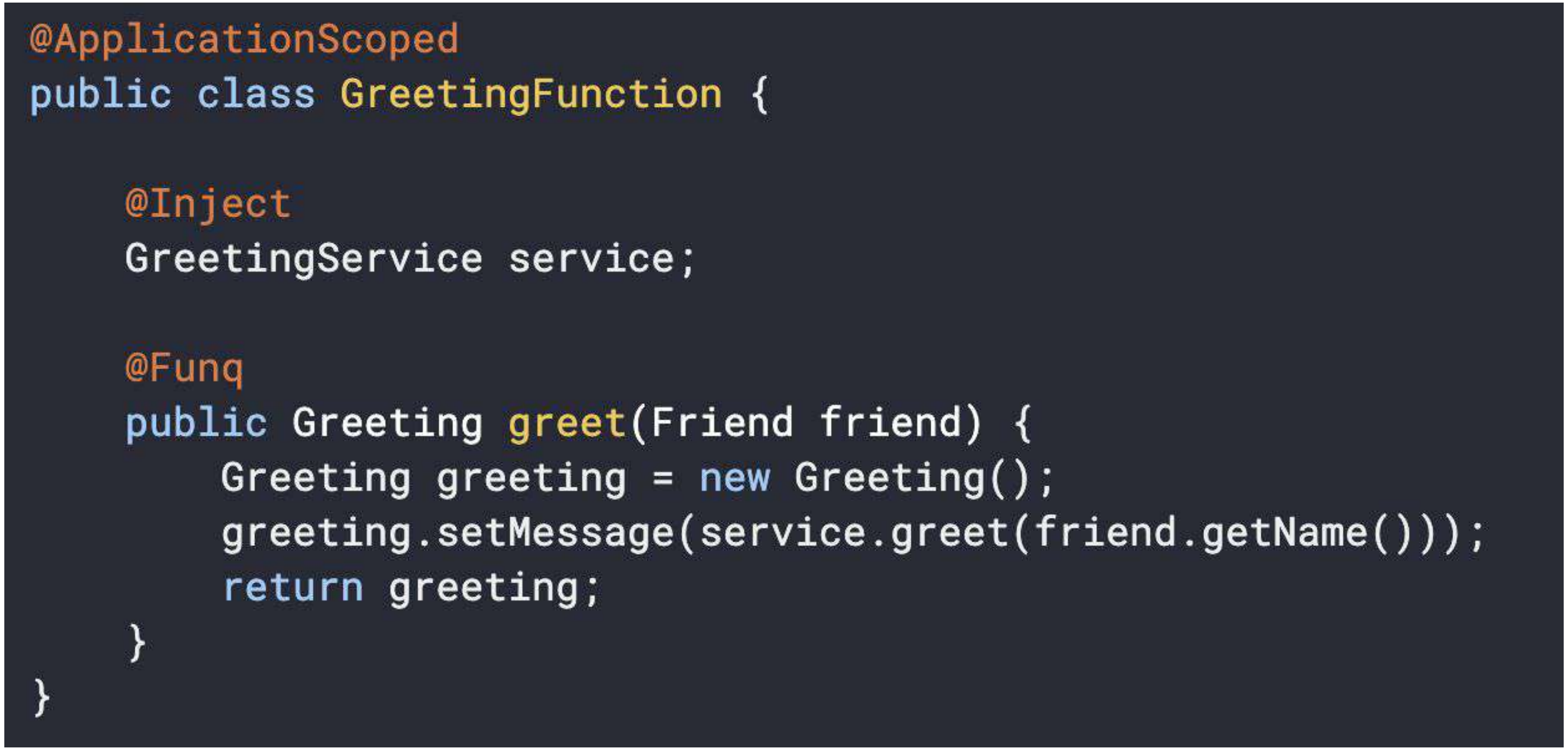
```
import io.quarkus.funqy.Funq;
import io.smallrye.mutiny.Uni;

public class GreetingFunction {

    @Funq
    public Uni<Greeting> reactiveGreeting(String name) {
        ...
    }
}
```

Quarkus Funqy

🤘 Supports dependency injection through CDI or Spring DI

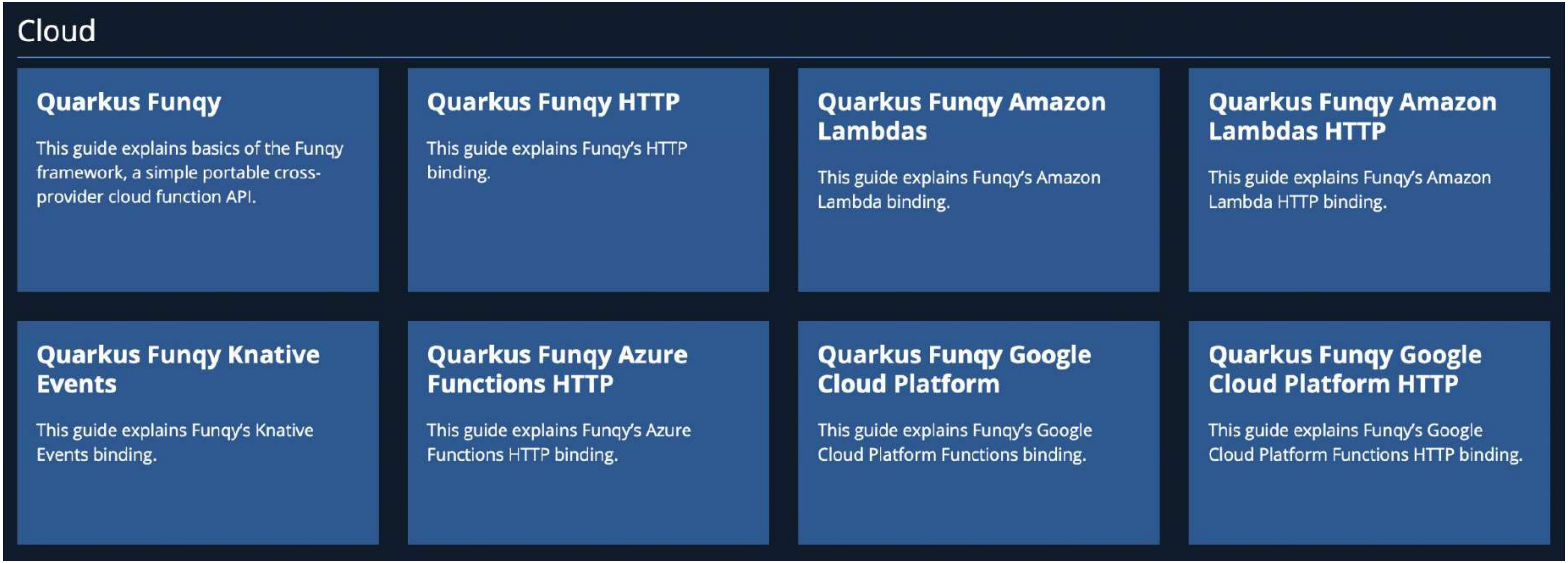


```
@ApplicationScoped
public class GreetingFunction {

    @Inject
    GreetingService service;

    @Funq
    public Greeting greet(Friend friend) {
        Greeting greeting = new Greeting();
        greeting.setMessage(service.greet(friend.getName()));
        return greeting;
    }
}
```

Choose a serverless platform to deploy the Funqy function



Cloud

Quarkus Funqy

This guide explains basics of the Funqy framework, a simple portable cross-provider cloud function API.

Quarkus Funqy HTTP

This guide explains Funqy's HTTP binding.

Quarkus Funqy Amazon Lambdas

This guide explains Funqy's Amazon Lambda binding.

Quarkus Funqy Amazon Lambdas HTTP

This guide explains Funqy's Amazon Lambda HTTP binding.

Quarkus Funqy Knative Events

This guide explains Funqy's Knative Events binding.

Quarkus Funqy Azure Functions HTTP

This guide explains Funqy's Azure Functions HTTP binding.

Quarkus Funqy Google Cloud Platform

This guide explains Funqy's Google Cloud Platform Functions binding.

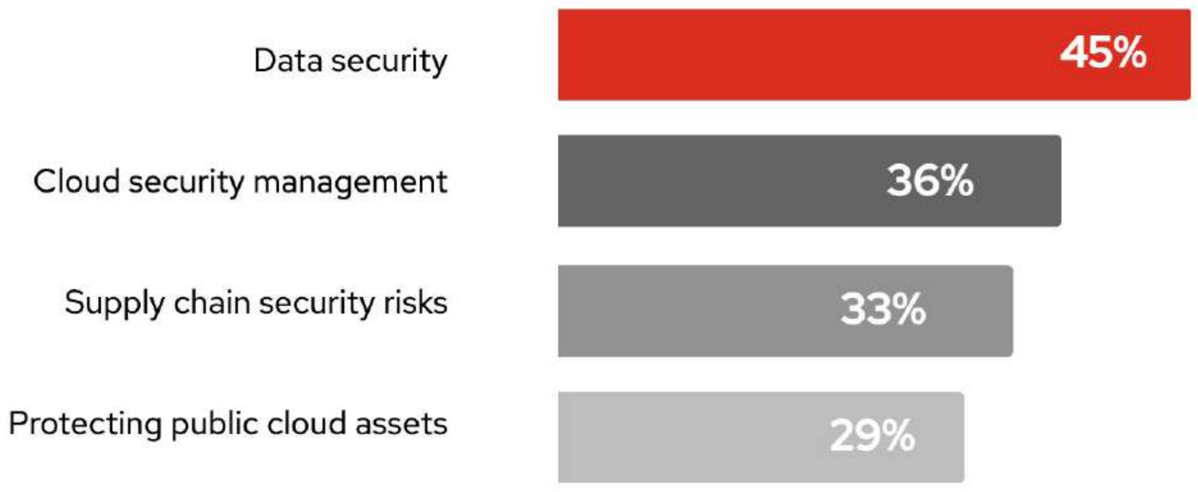
Quarkus Funqy Google Cloud Platform HTTP

This guide explains Funqy's Google Cloud Platform Functions HTTP binding.

“Shift Left” for App Dev on Public Cloud

Develop and **Ship** Governance Policies **as part of your application**!

Top four areas of concern as AppDev shift toward the cloud include*:



*https://www.redhat.com/en/topics/security/devsecops/approach

In a typical (and simplified) software development process:

- requirements phase
- design/development/ DevSecOps
- testing
- deployment.

“Shift Left” for App Dev on Public Cloud

Develop and **Ship** Governance Policies **as part of your application**!

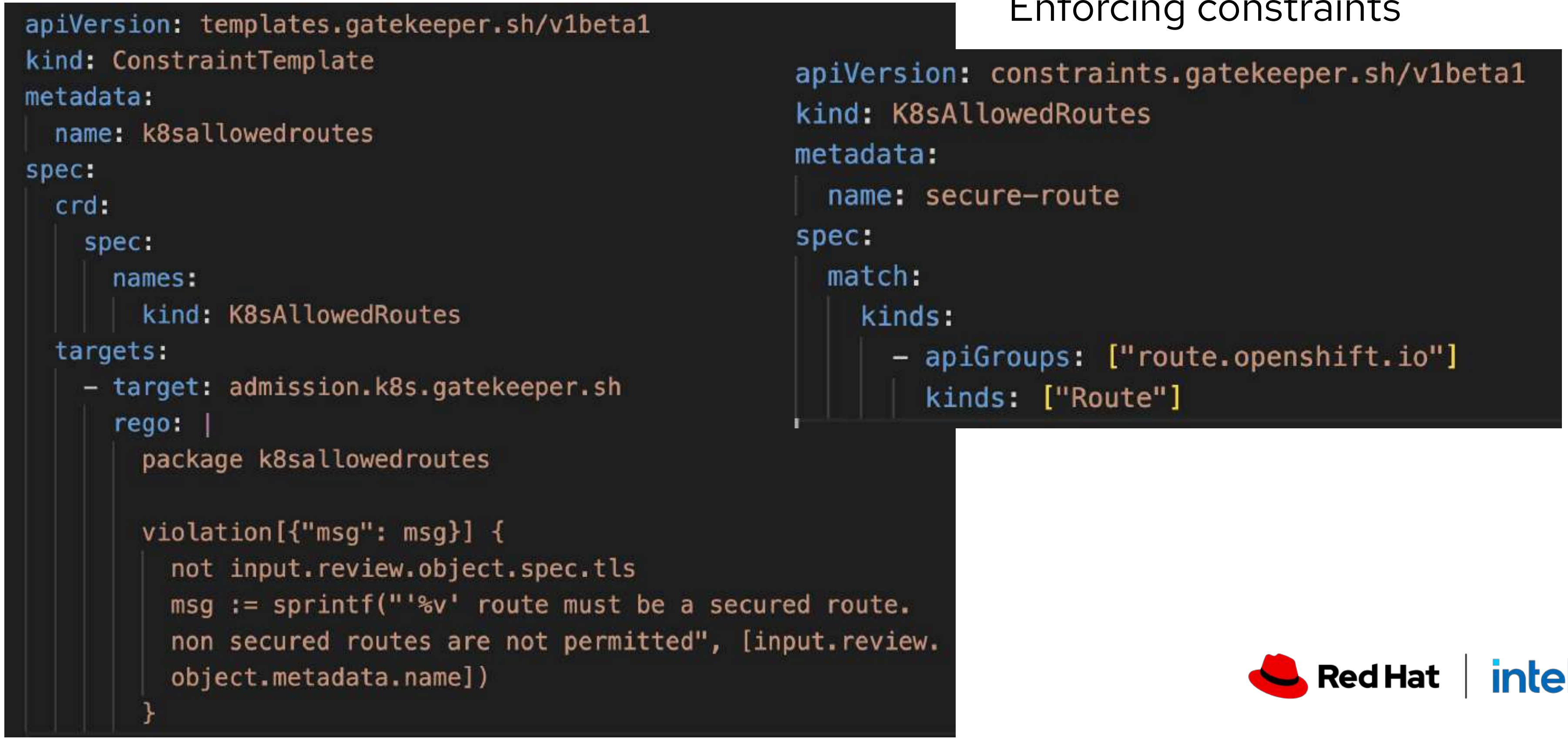
Programmed logic for any CRD (stored in Git)!

Open Policy Agent

Gatekeeper

Defining constraints

Enforcing constraints



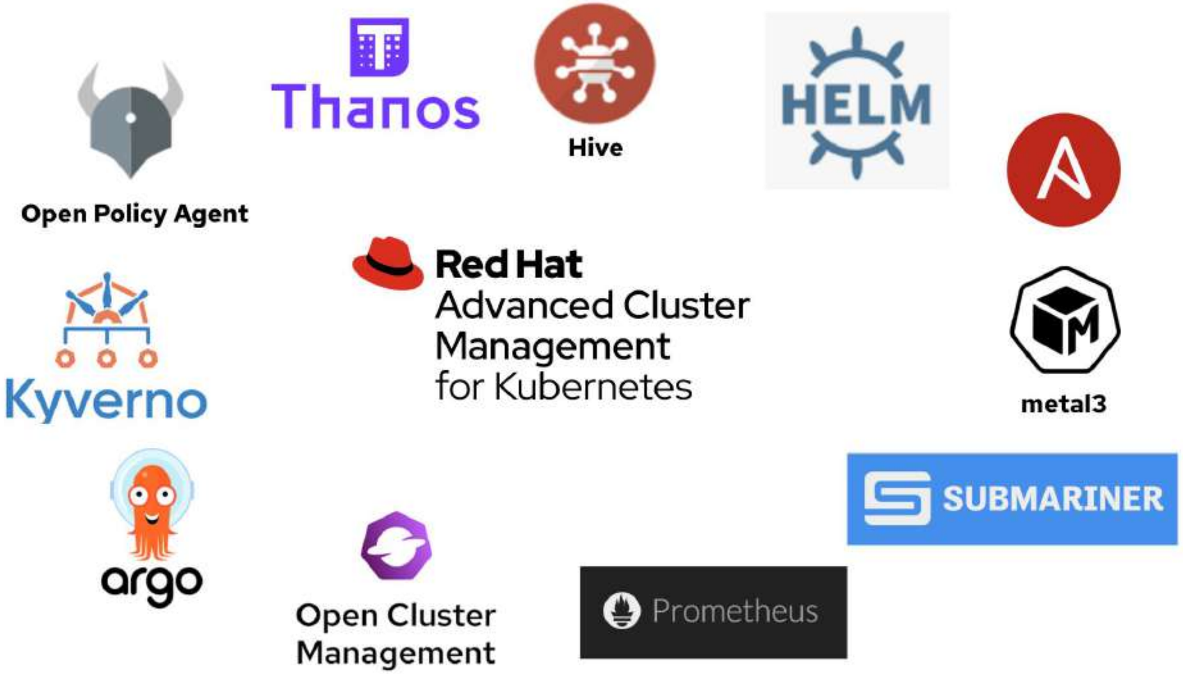
```
apiVersion: templates.gatekeeper.sh/v1beta1
kind: ConstraintTemplate
metadata:
  name: k8sallowedroutes
spec:
  crd:
    spec:
      names:
        kind: K8sAllowedRoutes
  targets:
    - target: admission.k8s.gatekeeper.sh
      rego: |
        package k8sallowedroutes

        violation[{"msg": msg}] {
          not input.review.object.spec.tls
          msg := sprintf("'%v' route must be a secured route.
          non secured routes are not permitted", [input.review.
          object.metadata.name])
        }
```

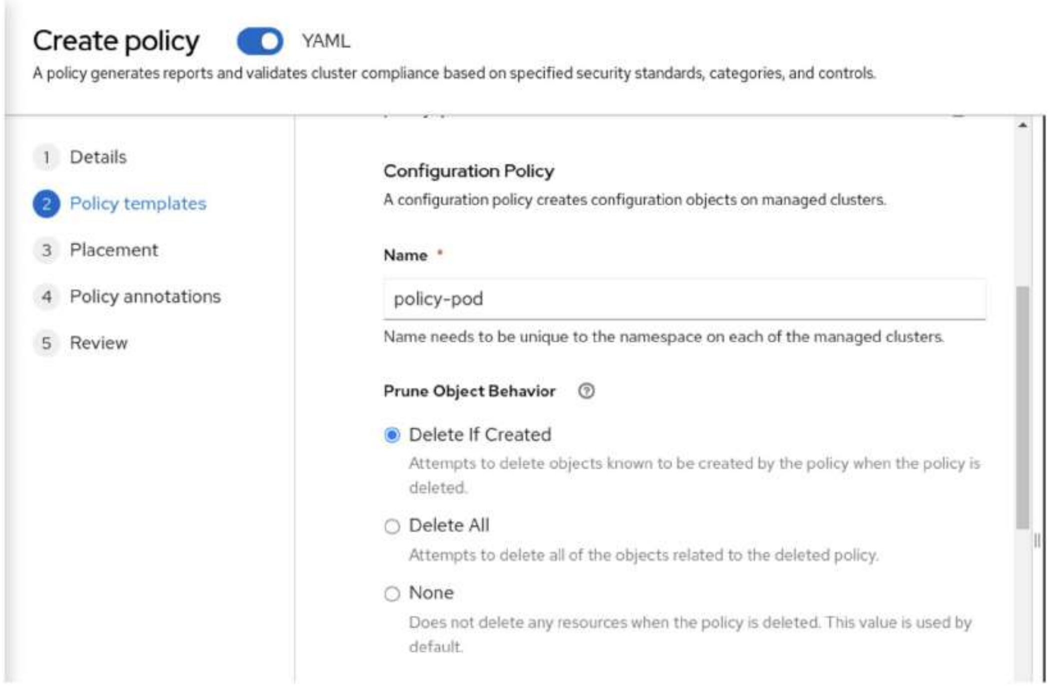
```
apiVersion: constraints.gatekeeper.sh/v1beta1
kind: K8sAllowedRoutes
metadata:
  name: secure-route
spec:
  match:
    kinds:
      - apiGroups: ["route.openshift.io"]
        kinds: ["Route"]
```

More Compliance Needed?

Additional Policy Engines and GitOps



Policy creation wizard



Create policy ● YAML

A policy generates reports and validates cluster compliance based on specified security standards, categories, and controls.

1 Details
2 Policy templates
3 Placement
4 Policy annotations
5 Review

Configuration Policy

A configuration policy creates configuration objects on managed clusters.

Name *

policy-pod

Name needs to be unique to the namespace on each of the managed clusters.

Prune Object Behavior

● Delete If Created

Attempts to delete objects known to be created by the policy when the policy is deleted.

○ Delete All

Attempts to delete all of the objects related to the deleted policy.

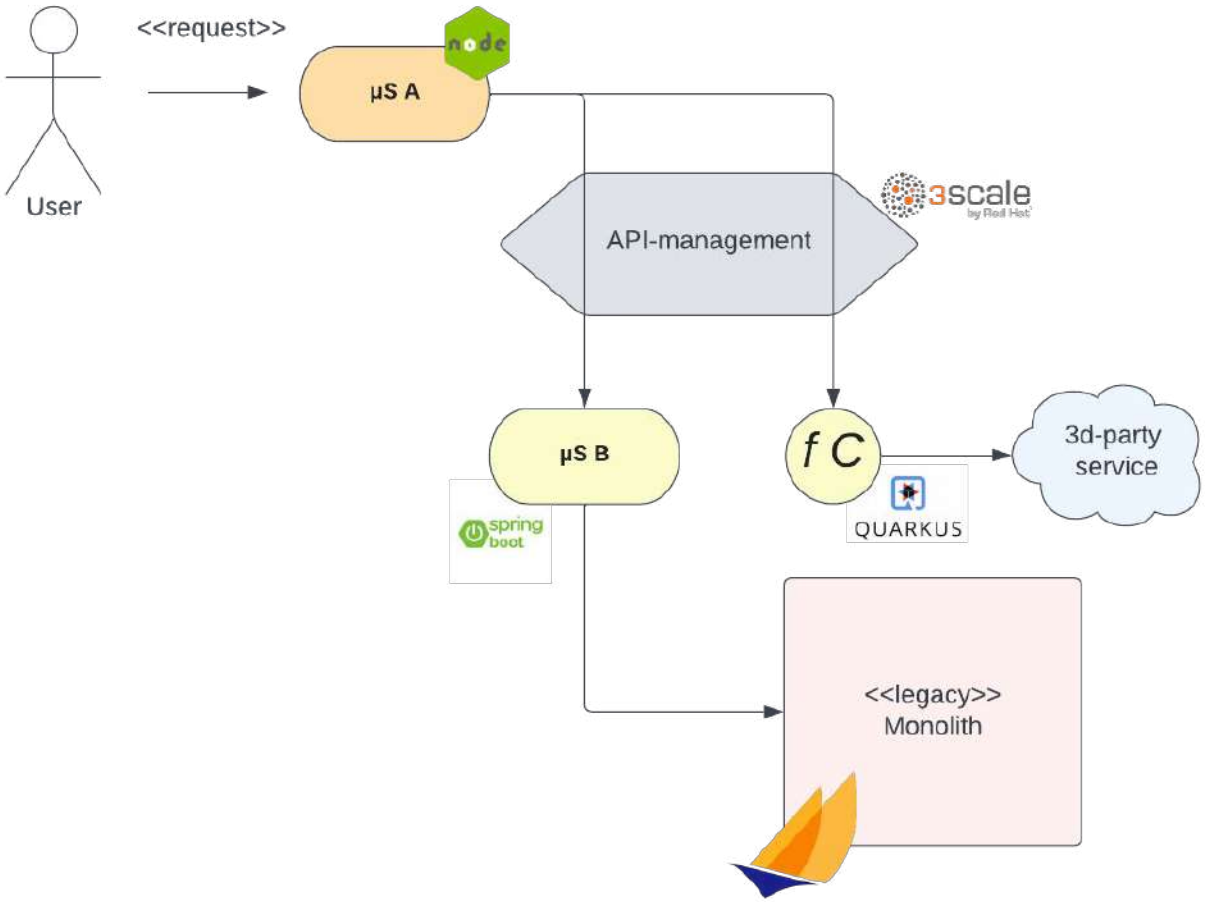
○ None

Does not delete any resources when the policy is deleted. This value is used by default.

*used rather by security specialists rather developers

Centralised Authorization for Enterprise Orchestra

Example cloud based app orchestra



- Need: time based access (not just role-based)?
- Second Factor Authentication?
- How to handle different Deployments?

KEYCLOAK

Part of ARO and ROSA with Red Hat support

Use Authorization Services!
Do not programm authz logic,
configure it via KC API!

Example

spring boot

KEYCLOAK

Part of ARO and ROSA with Red Hat support

https://github.com/redhat-developer/redhat-sso-quickstarts/tree/7.4.x/app-authz-rest-springboot

Where to find useful information for app developers?

https://developers.redhat.com/e-books

Summary

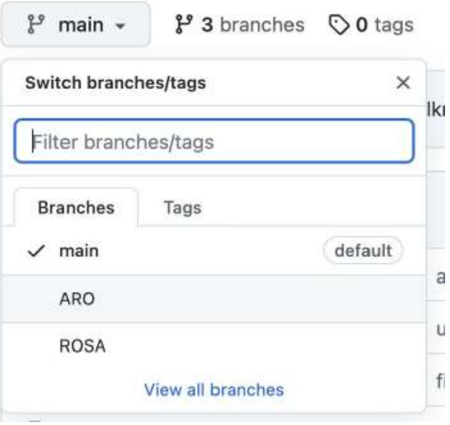
- ROSA, ARO are application platforms
- K8S is boring. Look for everything on top.
- Do not build CI/CD pipelines, build applications. Using Supply Chain Levels for Software Artifacts
- OpenShift is not an island: dev lifecycle with AWS/Azure Managed Services
- Use microservices patterns with inbuilt ARO/ROSA parts
- Serverless, but across clouds? Funqy
- API Management vs. Service Mesh
- Shift left in practice!
- Keycloak is part of ARO/ROSA: Centralised Authorisation
- Keep learning! (with RedHat Books)

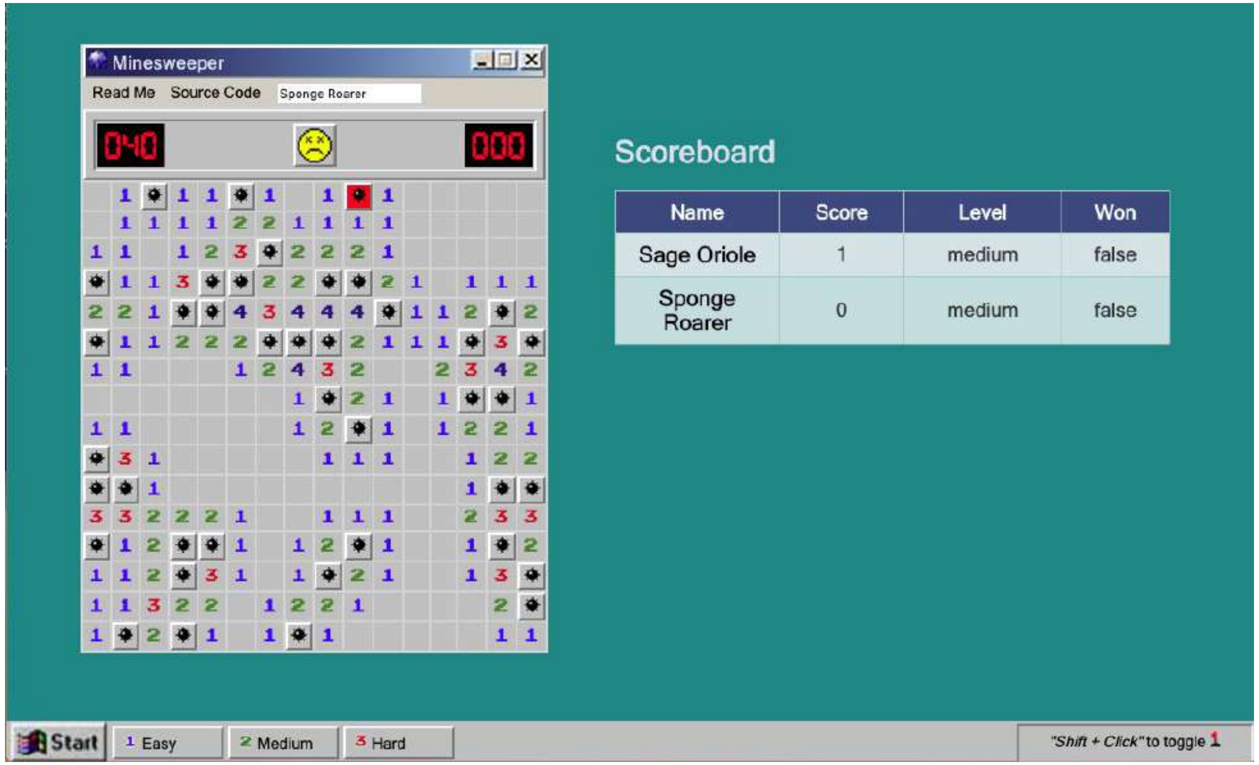
Bonus

Microsweeper

examples for ARO and ROSA with

- ext. DynamoDB (ROSA)
- ext. Azure PostgreSQL (ARO)





Scoreboard

Name	Score	Level	Won
Sage Oriole	1	medium	false
Sponge Roarer	0	medium	false

https://github.com/redhat-mw-demos/microsweeper-quarkus/tree/ROSA

Learn. Code. Play! :D

Thank you!

Yury Titov

Email: ytitov@redhat.com

Red Hat

Red Hat Developer

Join Red Hat Developer.

Build here. Go anywhere.

youtube.com/RedHatDevelopers

linkedin.com/showcase/red-hat-developer

facebook.com/RedHatDeveloperProgram

twitter.com/rhdevelopers

```
[your@sandbox ~]$ lscpu
RAM: 7GB
Storage: 15GB
Time limit: 30 days
Awesome: YES
```

Learn containers, Kubernetes, and OpenShift in your browser.

Start exploring in the OpenShift Sandbox.

Try Red Hat's products and technologies
without setup or configuration.

developers.redhat.com/developer-sandbox

Red Hat

Red Hat
Developer