The Path to Cloud

Kubernetes Advanced Concepts

Niklaus Hirt
DevOps Architect / Cloud Architect
nikh@ch.ibm.com

What we have seen so far....

Module 0: Prepare the Labs

Module 1: Microservices

Module 2: Containers with Docker

Module 3: Kubernetes

Module 4: Kubernetes Hands-On

Agenda

Module 5: Mesh Networking with ISTIO

Module 6: Mesh Networking Hands-On

Module 7: Serverless with Knative

Module 8: Serverless with Knative Hands-On (Optional)

Module 9: GitOps with ArgoCD (Optional)

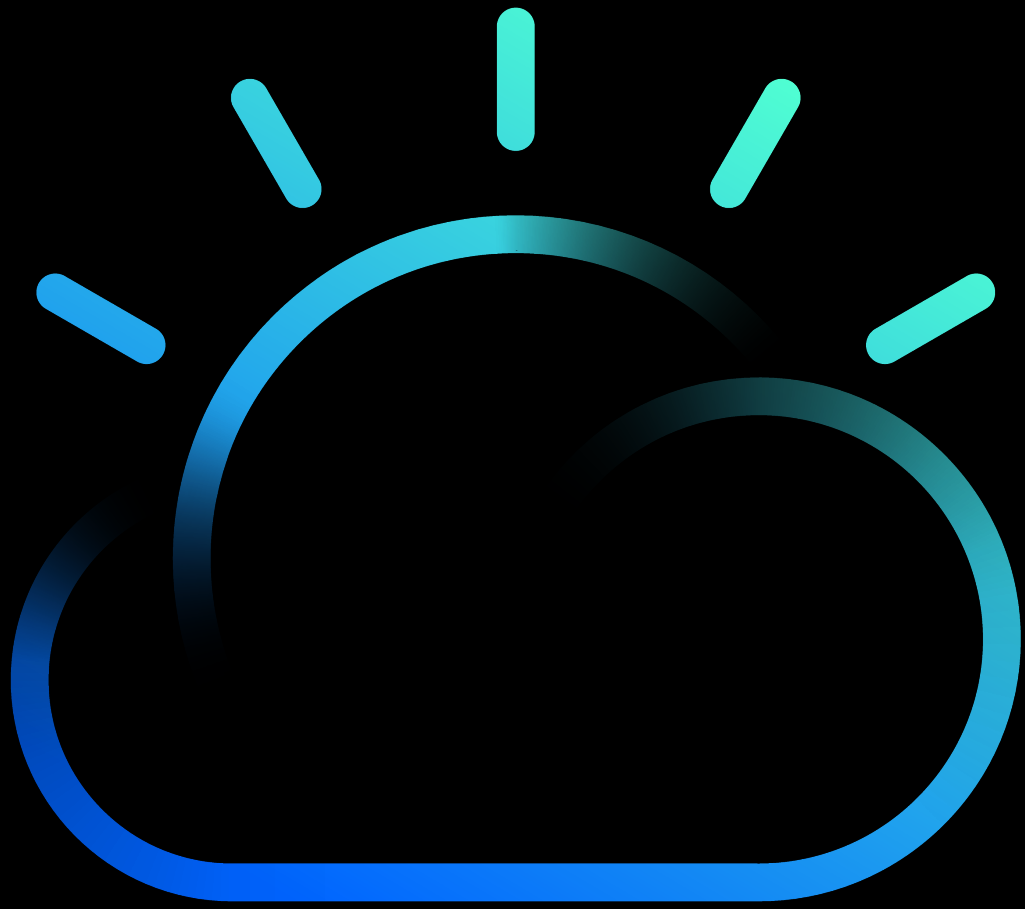
Module 10: Multi-Cloud Management (Optional)

Wrap-up

IBM

The path to Cloud
Mesh Networking

05



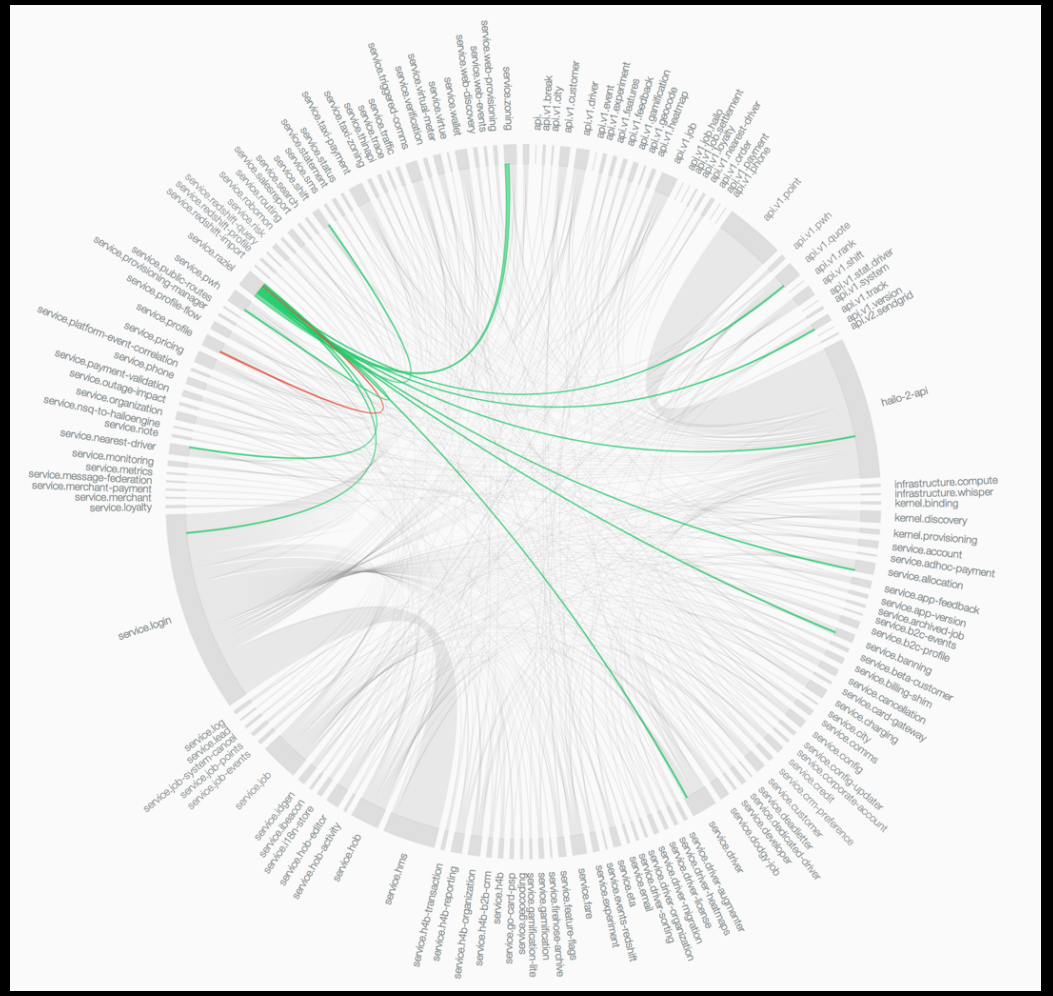
IBM **Cloud**

The trade off

Improved delivery velocity

in exchange for

increased operational complexity



Common DevOps Challenge 1

How do I **roll out** a newer version of my microservice **without down time**?

How do I **ensure traffic** continue to go to the current version before the newer version is tested and ready?

Common DevOps Challenge 2

How do I do **canary testing**?

How do I proceed to a **full rollout** after satisfactory testing of the new version?

Common DevOps Challenge 3

How do I do **A/B testing**?

- Release a new version to a subset of users in a precise way

I want to leverage crowdsourced testing. How do I **test** the new version **with a subset of users**?

I have **launched B in the dark**, but how can I keep B to myself or a small testing group?

Other common DevOps Challenges

4. Things don’t always go correctly in production… How do I **inject fault** to my microservices to prepare myself?

5. My services can only **handle certain rate**, how can I limit rate for some of my services?

Other common DevOps Challenges

6. I need to **view and monitor** what is going on with each of my services when crisis arises.

7. How can I **secure my services**.

Service Mesh

Dedicated infrastructure layer
to make
service-to-service communication
fast, safe and reliable

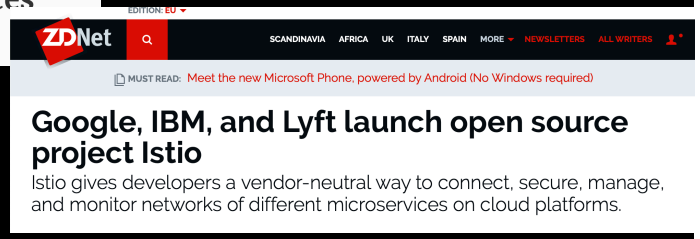
Istio

A service mesh designed to connect, manage and secure micro services


Forbes

3,859 views | May 25, 2017, 01:39am

Google, IBM And Lyft Want To Simplify Microservices Management With Istio

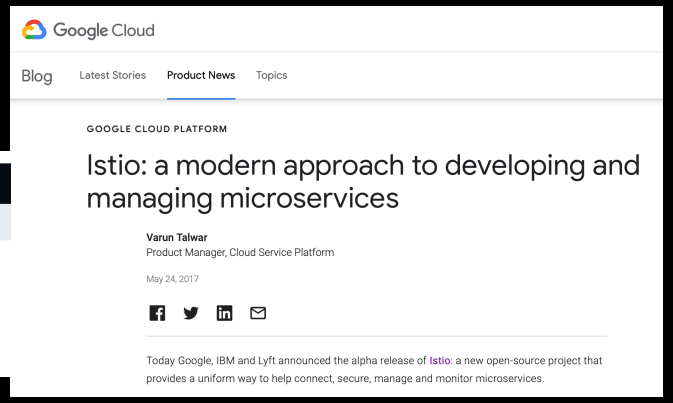

ZDNet

SCANDINIVIA AFRICA UK ITALY SPAIN MORE NEWSLETTERS ALL WRITERS

MUST READ: Meet the new Microsoft Phone, powered by Android (No Windows required)

Google, IBM, and Lyft launch open source project Istio

Istio gives developers a vendor-neutral way to connect, secure, manage, and monitor networks of different microservices on cloud platforms.


Google Cloud

Blog Latest Stories Product News Topics

GOOGLE CLOUD PLATFORM

Istio: a modern approach to developing and managing microservices

Varun Talwar
Product Manager, Cloud Service Platform

May 24, 2017

Today Google, IBM and Lyft announced the alpha release of Istio: a new open-source project that provides a uniform way to help connect, secure, manage and monitor microservices.

Launched in May 2017 by Google, Lyft and IBM

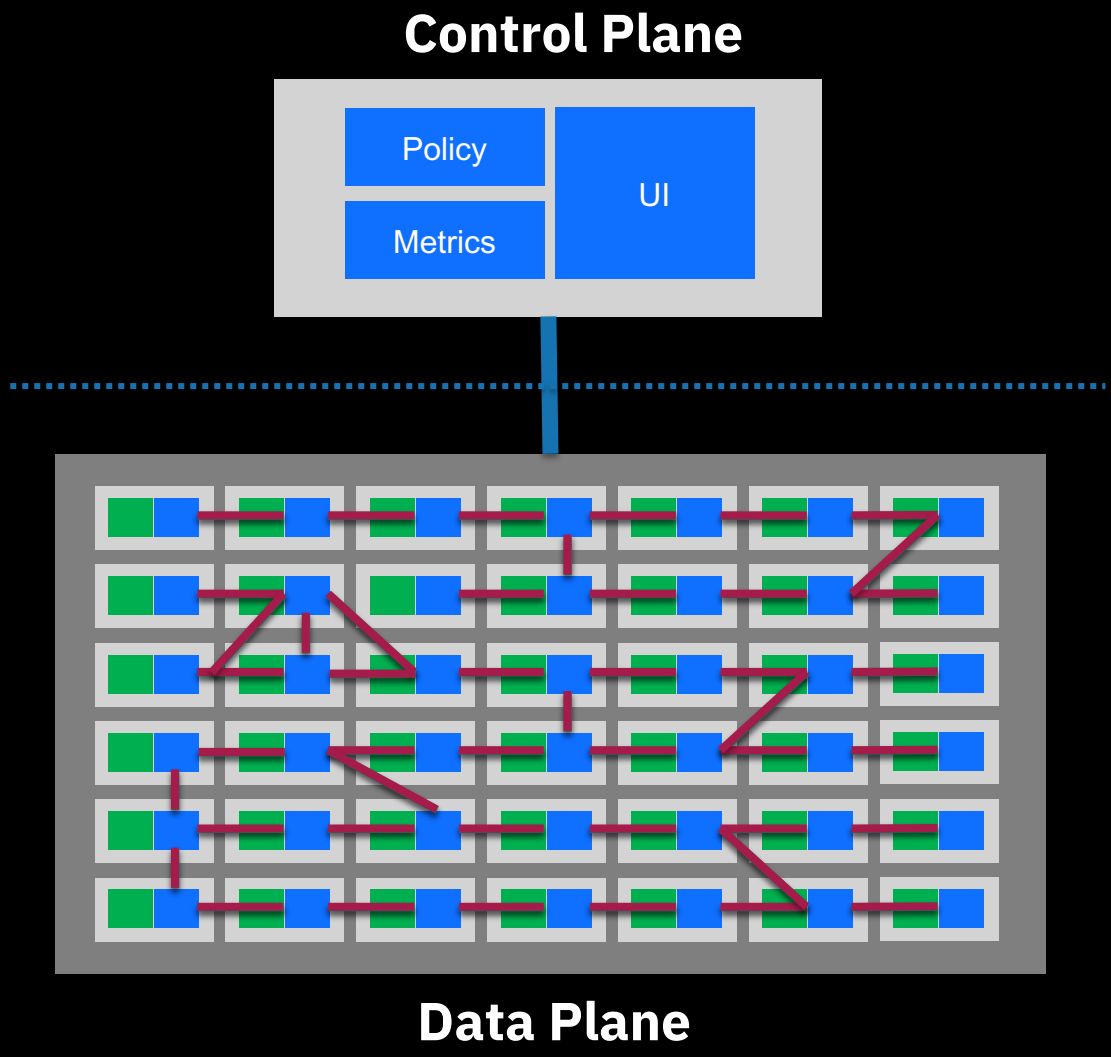
Open Source

Zero Code Changes

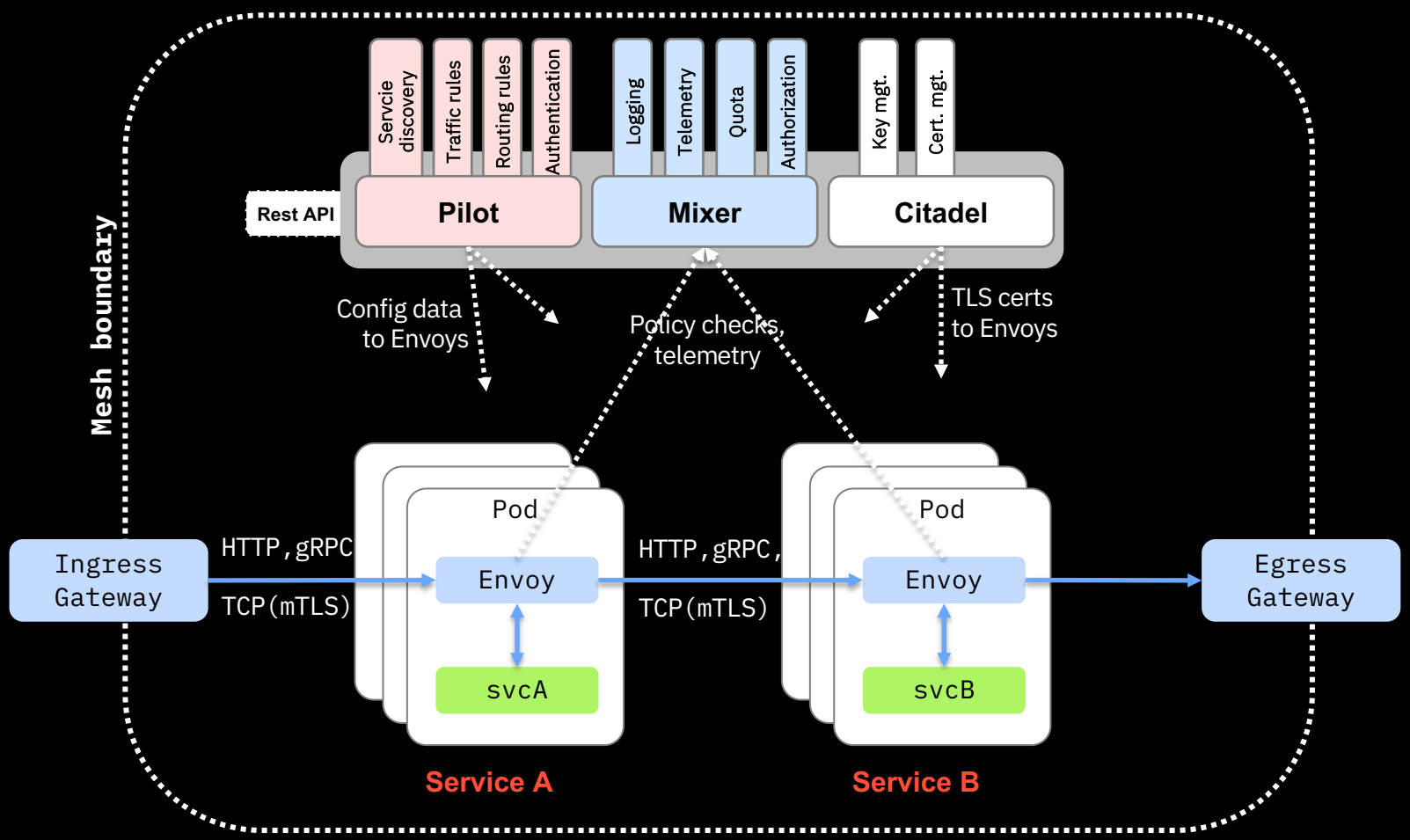
Service Mesh

Tent poles:

- Control over request routing (CI/CD release patterns)
- Resiliency (retries, timeouts, etc)
- Cascading failure prevention (circuit breaking)
- Load balancing
- Provide and manage TLS termination between endpoints
- Metrics to provide instrumentation at the service-to-service layer



ISTIO - Architecture



Istio control plane traffic. Request routing rules, resilience configuration (circuit breakers, timeouts, retries), policies (ACLs, rate limits, auth), and metrics/reports from proxies.

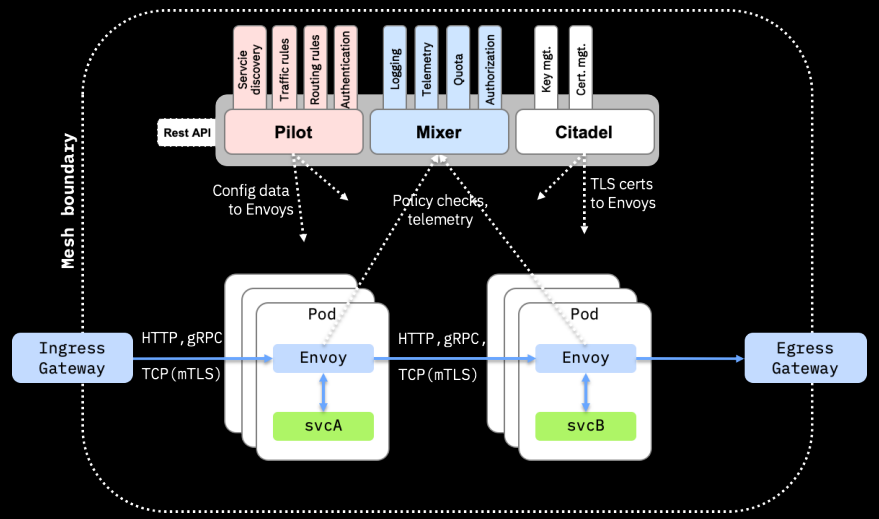
User/application traffic. HTTP/1.1, HTTP/2, gRPC, TCP with or without TLS

ISTIO - Architecture

Istio

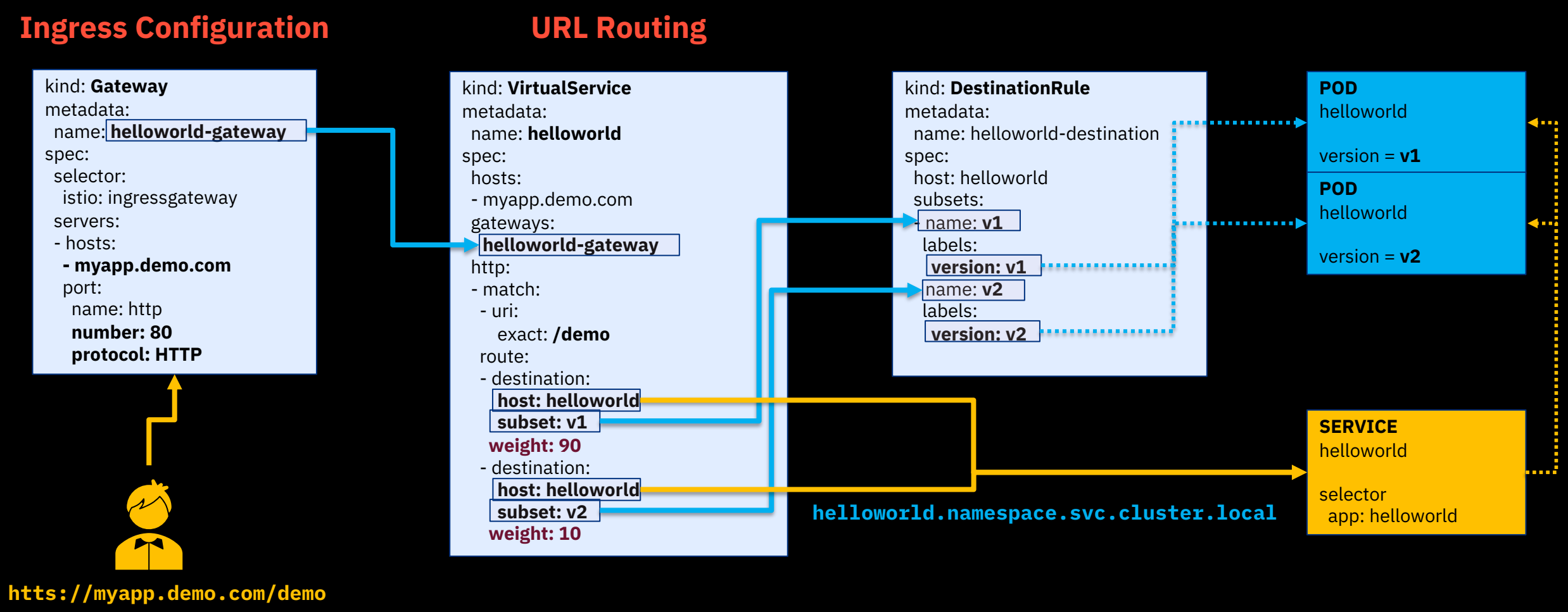
Operates at **Layer 7**

- policies can be applied based on virtual host, URL, or other HTTP headers.
- Flexibility in processing.
- Allows it to be distributed



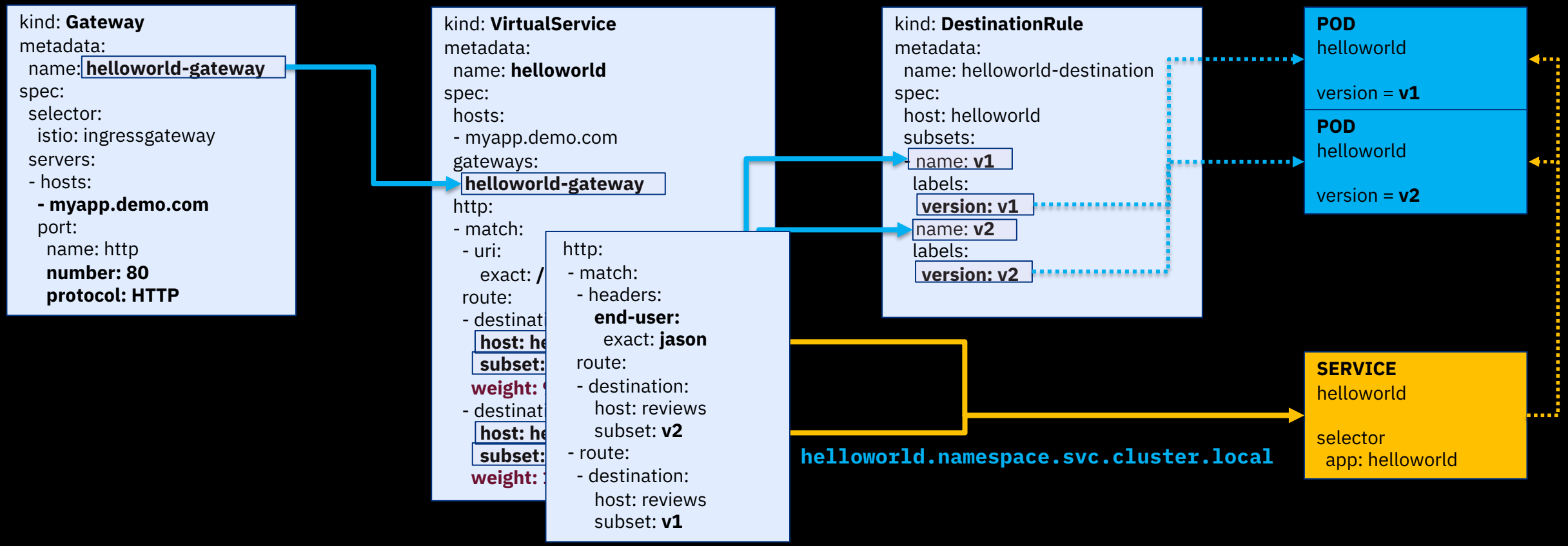
ISTIO

Custom resource definitions



ISTIO

Custom resource definitions



ISTIO
Sidecar injection

Manual Injection

```
kubectl apply -f <(istioctl kube-inject -f myapp.yaml)
```

Automatic Injection

For automatic sidecar injection, Istio relies on Mutating Admission Webhooks.
Label the namespace where you are deploying the app with `istio-injection=enabled`

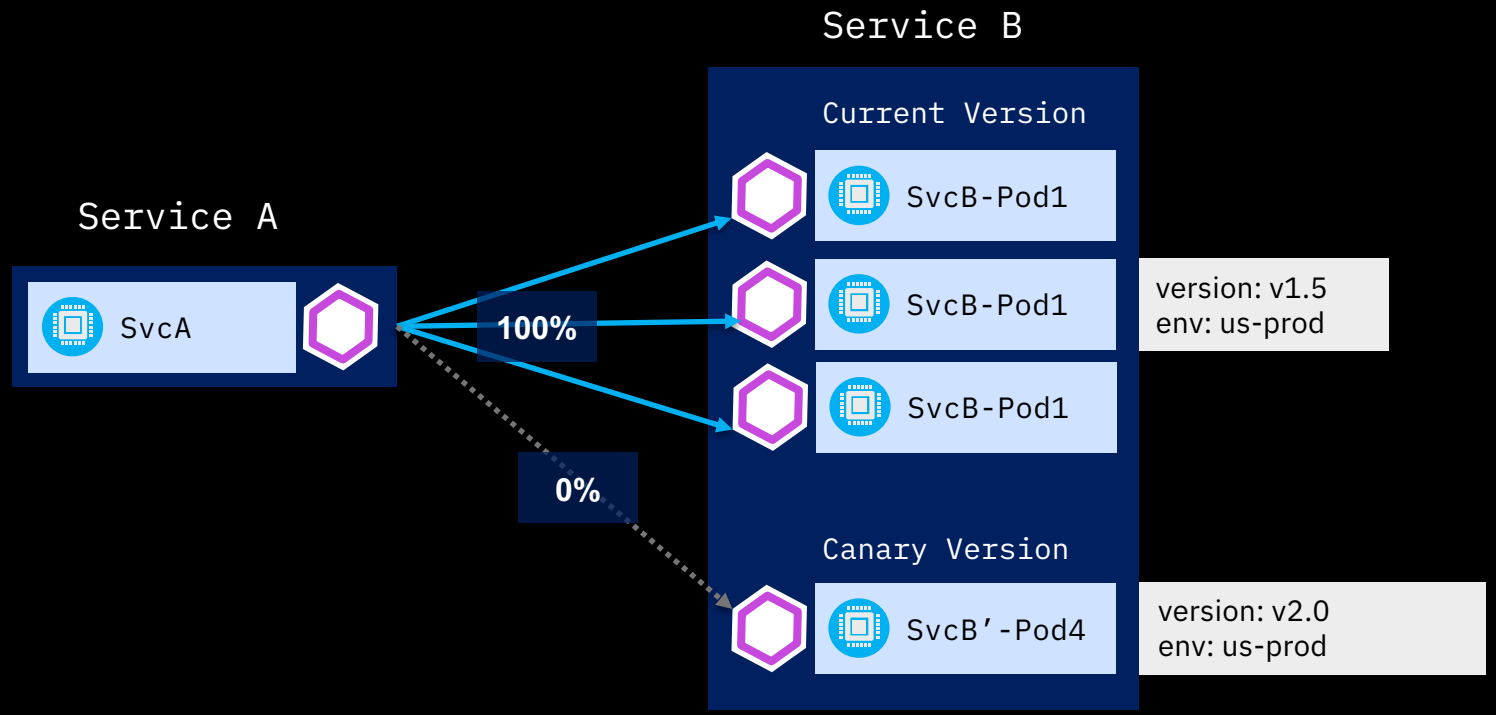
```
root@please1:~# kubectl get namespaces --show-labels
NAME                 STATUS        AGE   LABELS
default              Active        35d   istio-injection=enabled
dev-namespace        Active        35d   <none>
godemo               Terminating   16d   istio-injection=enabled
istio-system         Active        35d   icp=system
kube-public          Active        35d   <none>
kube-system          Active        35d   icp=system
platform             Active        35d   <none>
prod-namespace       Active        35d   <none>
services             Active        35d   <none>
test-namespace       Active        35d   <none>
```

```
kind: Deployment
metadata:
  name: no-injection
spec:
  template:
    metadata:
      annotations:
        sidecar.istio.io/inject: "false"
    spec:
      containers:
      - name: no-injection
        image: nginx
```

Addressing DevOps Challenges

#	CHALLENGE	ISTIO SOLUTION
CHALLENGE 1	ROLL OUT NEW VERSION WITHOUT DOWNTIME OR CHANGING CODE	TRAFFIC CONTROL
CHALLENGE 2	HOW TO DO CANARY TESTING	TRAFFIC SPLITTING
CHALLENGE 3	HOW TO DO A/B TESTING	TRAFFIC STEERING
CHALLENGE 4	THINGS DON'T ALWAYS GO CORRECTLY IN PRODUCTION…	TRAFFIC MIRRORING RESILIENCY RESILIENCY TESTING
CHALLENGE5	HOW CAN I LIMIT RATE FOR SOME OF MY SERVICES?	RATE LIMITING
CHALLENGE 6	I NEED TO VIEW AND MONITOR WHAT IS GOING ON WHEN CRISIS ARISES	TELEMETRY
CHALLENGE 7	HOW CAN I SECURE MY SERVICES?	AUTHENTICATION AUTHORIZATION CALICO

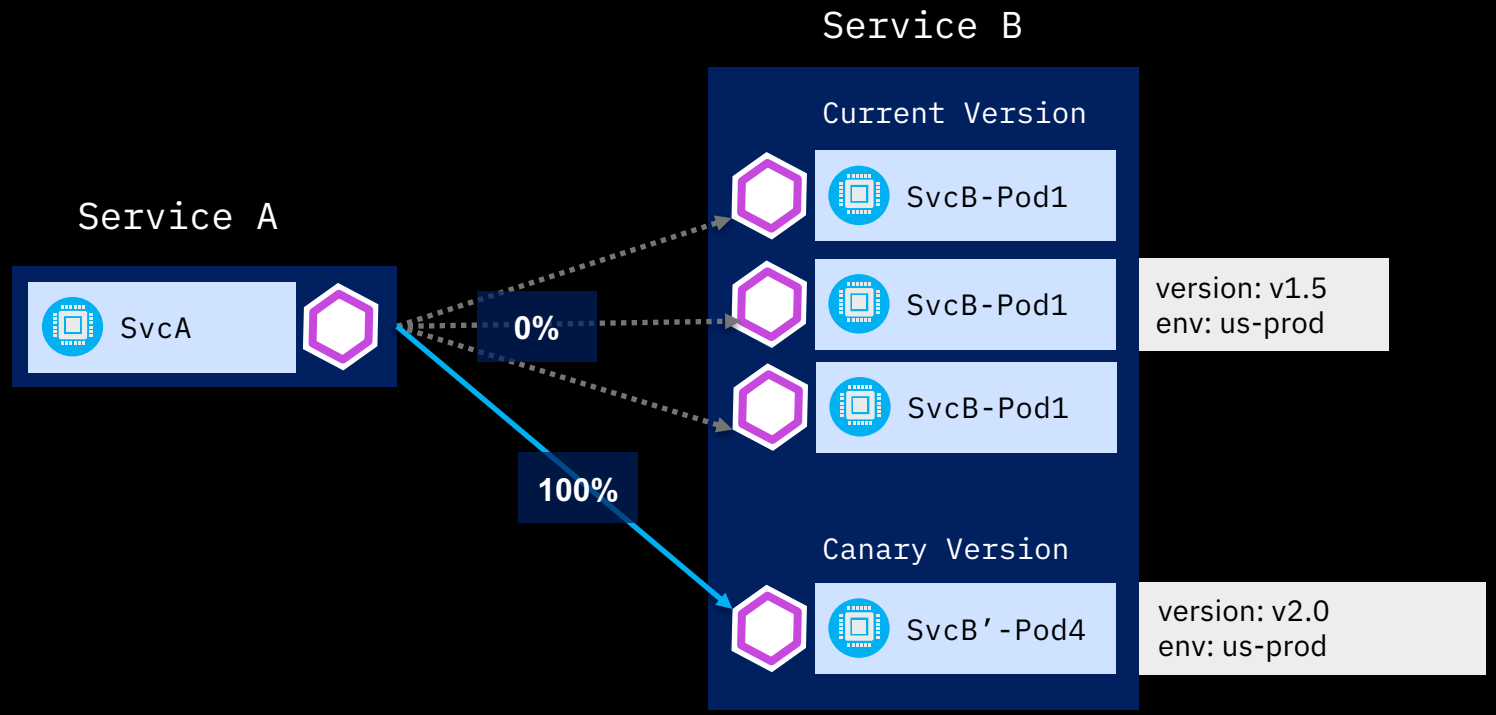
Traffic Control




```
kind: VirtualService
destination:
  serviceB.cluster.local
match:
  source:
    serviceA.cluster.local
route:
  - labels:
    version: v1.5
    env: us-prod
  weight: 100
```

CHALLENGE 1
ROLL OUT NEW VERSION WITHOUT DOWNTIME OR CHANGING CODE

Traffic Control

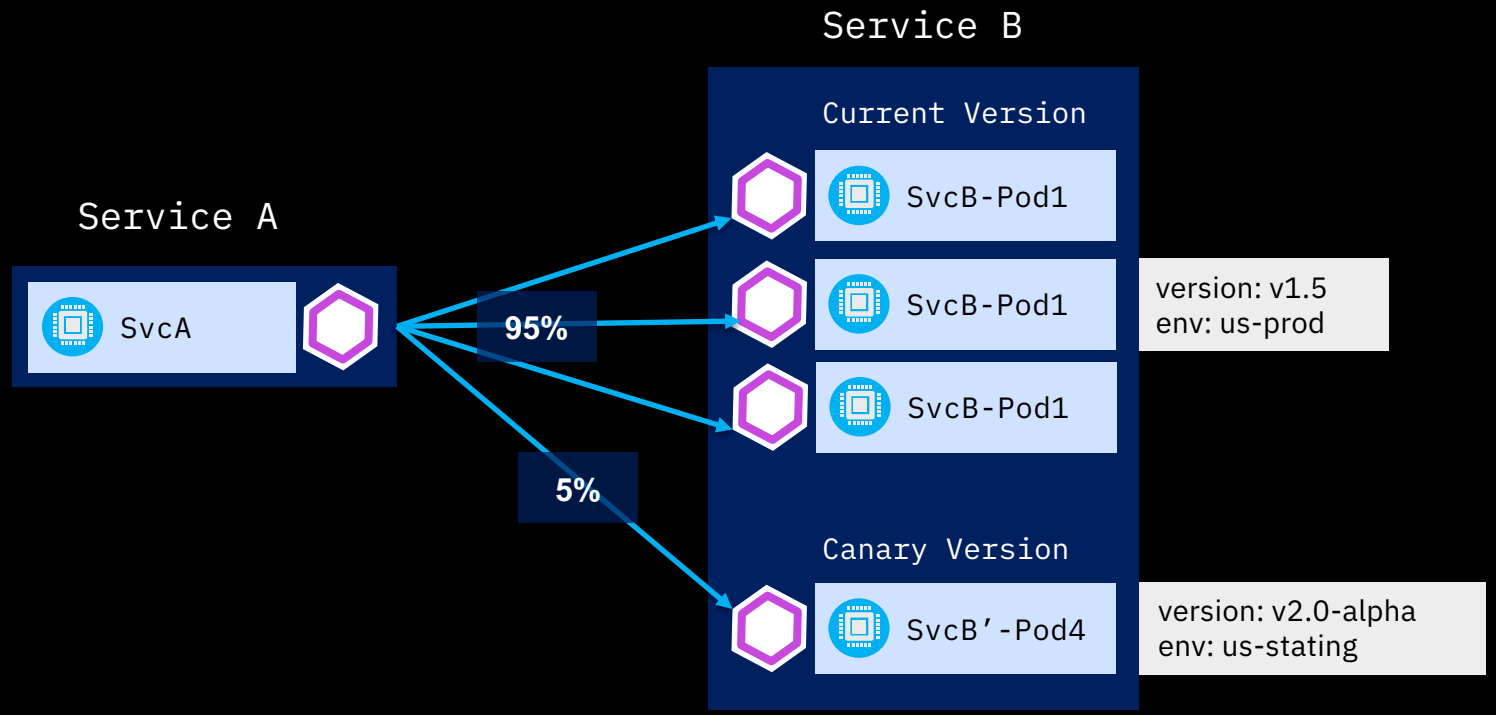



```
kind: VirtualService
destination:
  serviceB.cluster.local
match:
  source:
    serviceA.cluster.local
route:
  - labels:
    version: v2.0
    env: us-prod
  weight: 100
```

Update

CHALLENGE 1
ROLL OUT NEW VERSION WITHOUT DOWNTIME OR CHANGING CODE

Traffic Splitting

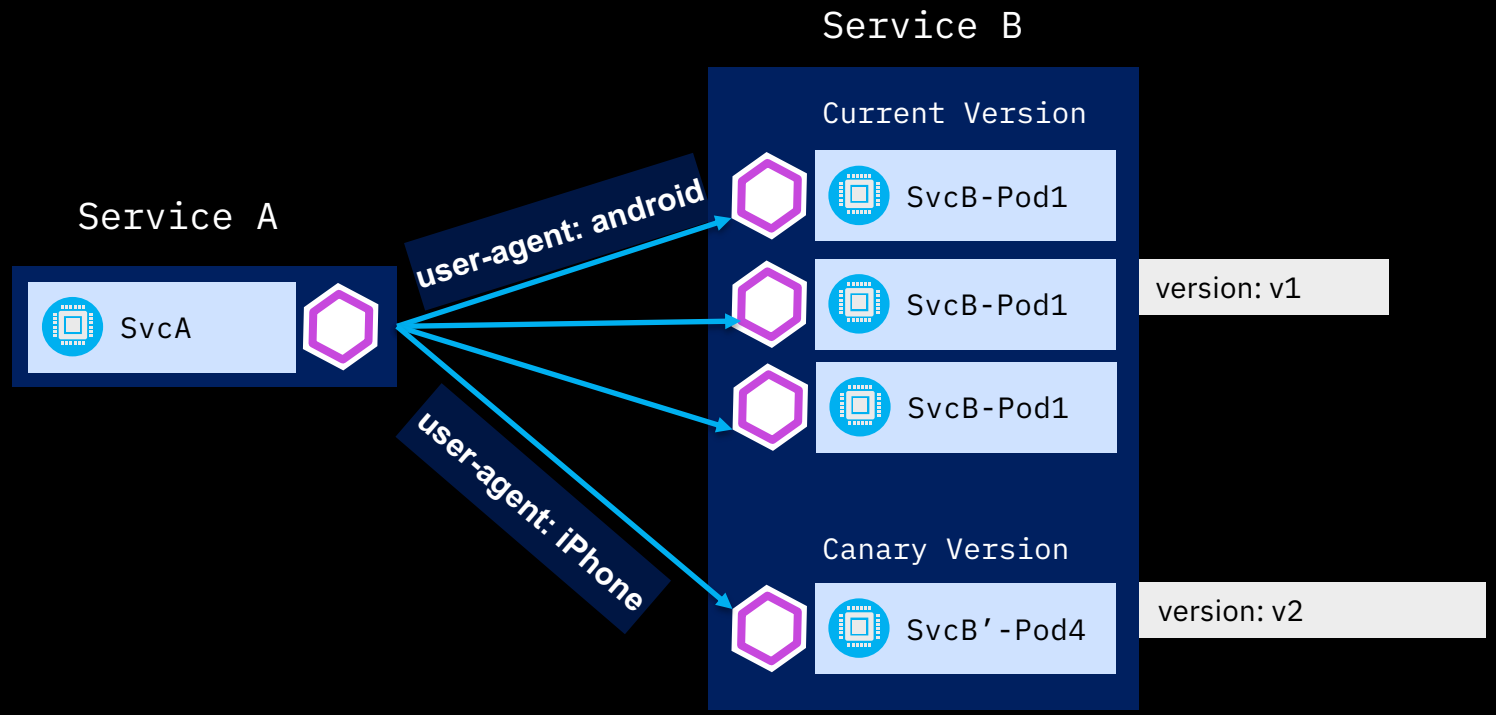



```
kind: VirtualService
destination:
  serviceB.cluster.local
match:
  source:
    serviceA.cluster.local
route:
  - labels:
    version: v1.5
    env: us-prod
  weight: 95
  - labels:
    version: v2.0-alpha
    env: us-staging
  weight: 5
```

CHALLENGE 2
HOW TO DO CANARY TESTING

Routing not based on the request content.
Staged rollouts with %-based traffic splits.

Traffic Steering

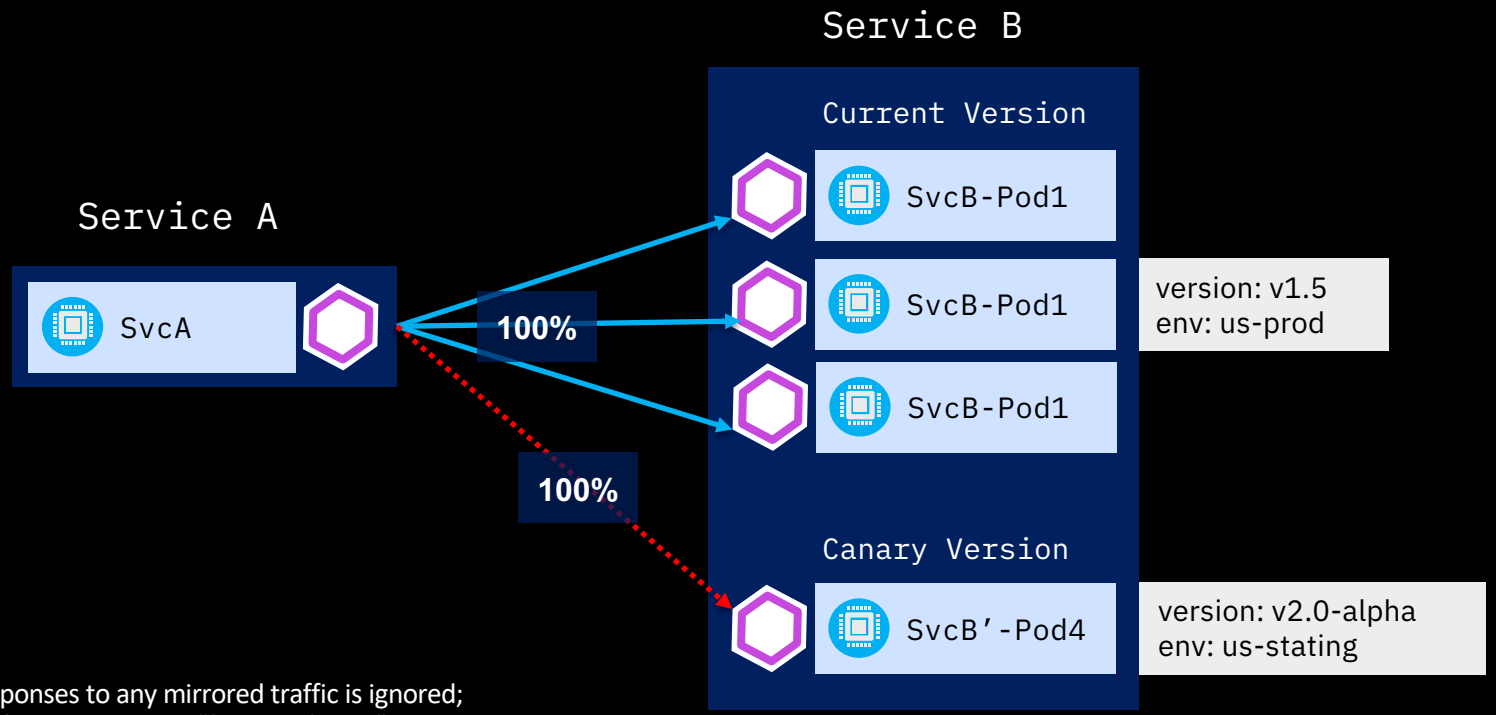



```
kind: VirtualService
destination:
 serviceB.cluster.local
match:
  httpHeaders:
    user-agent:
      regex:
        ^(.*?;)?(iPhone)(;.*)?$
 route:
- labels:
  version: v2
```

Routing based on the request content

CHALLENGE 3
HOW TO DO A/B TESTING

Traffic Mirroring



Responses to any mirrored traffic is ignored;
traffic is mirrored as "fire-and-forget"
You'll need to have the 0-weighted route to hint to
Istio to create the proper Envoy

```
kind: VirtualService
destination:
  serviceB.cluster.local
match:
  source:
    serviceA.cluster.local
route:
  - labels:
    version: v1.5
    env: us-prod
  weight: 100
  - labels:
    version: v2.0-alpha
    env: us-staging
  weight: 0
  mirror:
    name: httpbin
    labels:
      version: v2.0-alpha
      env: us-staging
```

CHALLENGE 4
THINGS DON'T ALWAYS GO CORRECTLY IN PRODUCTION...

Resiliency

Istio adds fault tolerance to your application without any changes to code

```
// Circuit breakers
destination: serviceB.example.cluster.local
policy:
- labels:
    version: v1
  circuitBreaker:
    simpleCb:
      maxConnections: 100
      httpMaxRequests: 1000
      httpMaxRequestsPerConnection: 10
      httpConsecutiveErrors: 7
      sleepWindow: 15m
      httpDetectionInterval: 5m
```

Resilience features

- Timeouts
- Retries with timeout budget
- Circuit breakers
- Health checks
- AZ-aware load balancing w/ automatic failover
- Control connection pool size and request load

CHALLENGE 4
THINGS DON'T ALWAYS GO CORRECTLY IN PRODUCTION...

Resiliency
Circuit Breakers

Istio

Connection pool

- Limits connections for reviews to invoke ratings
- **Limited to 1 concurrent connection, 1 request per connection (One concurrent request total)**
  - While requests are using all of the connections in a pool, any new requests are rejected

Outlier detection

- **If there are 3 requests in 2 seconds, reviews will be ejected for 3 minutes**
  - Request 1 will take more than 2 seconds, blocking the connection during that time
  - Request 2 won't get a connection, which will generate the first error
  - Request 3 won't get a connection, which will generate the second error, causing ejection

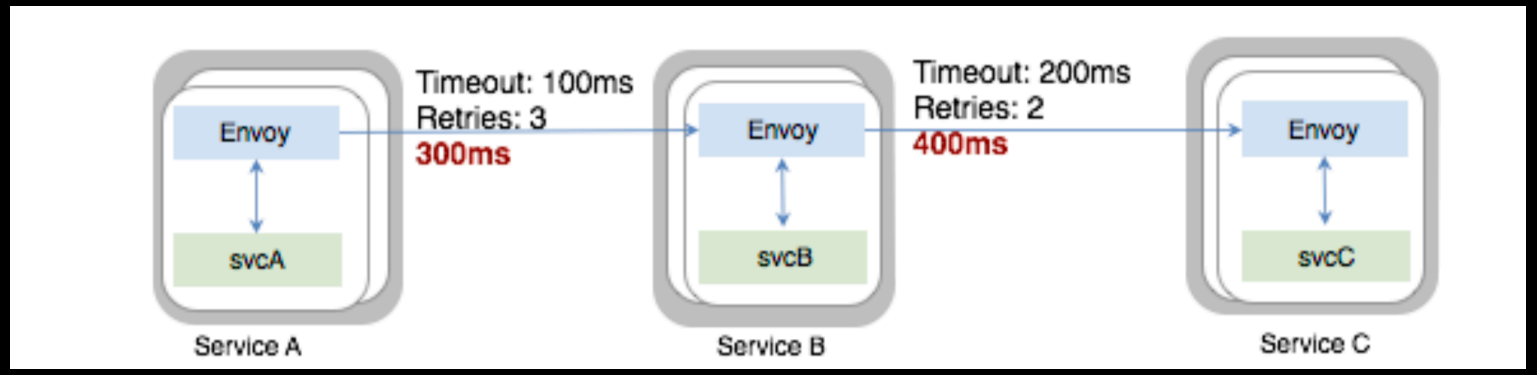
```
kind: DestinationRule
host: reviews
trafficPolicy:
  connectionPool:
    tcp:
      maxConnections: 1
    http:
      http1MaxPendingRequests: 1
      maxRequestsPerConnection: 1
  outlierDetection:
    consecutiveErrors: 2
    interval: 2s
    baseEjectionTime: 3m
    maxEjectionPercent: 100
```

CHALLENGE 4
THINGS DON'T ALWAYS GO CORRECTLY IN PRODUCTION...

Resiliency
Fault injection

Fault injection can be used for testing
- Faults are caused on a percentage of requests
- Faults can cause a request delay or failure

In this example, ratings is being invoked
- All of the requests from bar have a 2 second timeout
- 40% of the requests from bar also have a 5 second delay



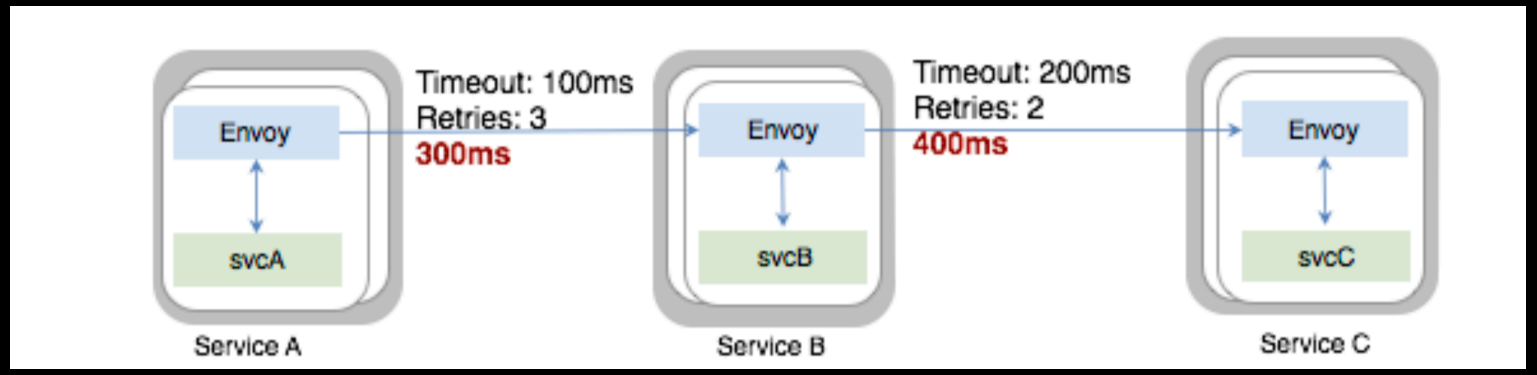

```
kind: VirtualService
hosts:
  - ratings
http:
- match:
  - headers:
      end-user:
        exact: bar
  fault:
    delay:
      percent: 40
      fixedDelay: 5s
  timeout: 2s
  route:
  - destination:
      host: ratings
```

CHALLENGE 4
HOW DO I INJECT FAULT TO MY MICROSERVICES TO PREPARE MYSELF?

Resiliency
Fault injection

Fault injection can be used for testing
- Faults are caused on a percentage of requests
- Faults can cause a request delay or failure

In this example, ratings is being invoked
- 20% of the requests from foo get an HTTP 500 error
- All other requests (not foo or bar) have a 4 second timeout




```
kind: VirtualService
hosts:
  - ratings
http:
- match:
  - headers:
      end-user:
        exact: foo
  fault:
    abort:
      percent: 20
      httpStatus: 500
  route:
  - destination:
      host: ratings
- route:
  - destination:
      host: ratings
  timeout: 4s
```

Timeout is measured after the host is invoked, it is calculated after delay period has passed.

The 40% of requests from bar will time out after 7 seconds (5 sec delay + 2 sec timeout)

CHALLENGE 4
HOW DO I INJECT FAULT TO MY MICROSERVICES TO PREPARE MYSELF?

Rate limiting

Istio protects your application from rogue actors by imposing ratelimits

```
Quotas:
- name: requestcount.quota.istio-system
  maxAmount: 5000
  validDuration: 1s
  overrides:
  - dimensions:
      destination: ratings
      source: reviews
      sourceVersion: v3
    maxAmount: 1
    validDuration: 1s
  - dimensions:
      destination: ratings
    maxAmount: 100
    validDuration: 1s
```

Rate limit

- ❖ Configurable limits with overrides
- ❖ Multiple rate limiting backends
- ❖ Conditional rate limiting

CHALLENGE 5
HOW CAN I LIMIT RATE FOR SOME OF MY SERVICES?

Rate limiting

Every distinct rate limit configuration represents a counter.
If the number of requests in the last validDuration duration exceed maxAmount, Mixer returns a RESOURCE_EXHAUSTED message to the proxy.

Global rate limit of 500 calls per second.

If “reviews” is called, it’s limited to one call every 5 seconds.

If “reviews” is called from 10.28.11.20, it’s limited to 99 calls per seconds.

```
kind: handler
quotas:
- name: requestcountquota.instance.istio-system
  maxAmount: 500
  validDuration: 1s

  overrides:

  - dimensions:
      destination: reviews
    maxAmount: 1
    validDuration: 5s

  - dimensions:
      destination: productpage
      source: "10.28.11.20"
    maxAmount: 99
    validDuration: 1s
```

CHALLENGE 5
HOW CAN I LIMIT RATE FOR SOME OF MY SERVICES?

Telemetry

Monitoring & tracing should not be an afterthought in the infrastructure

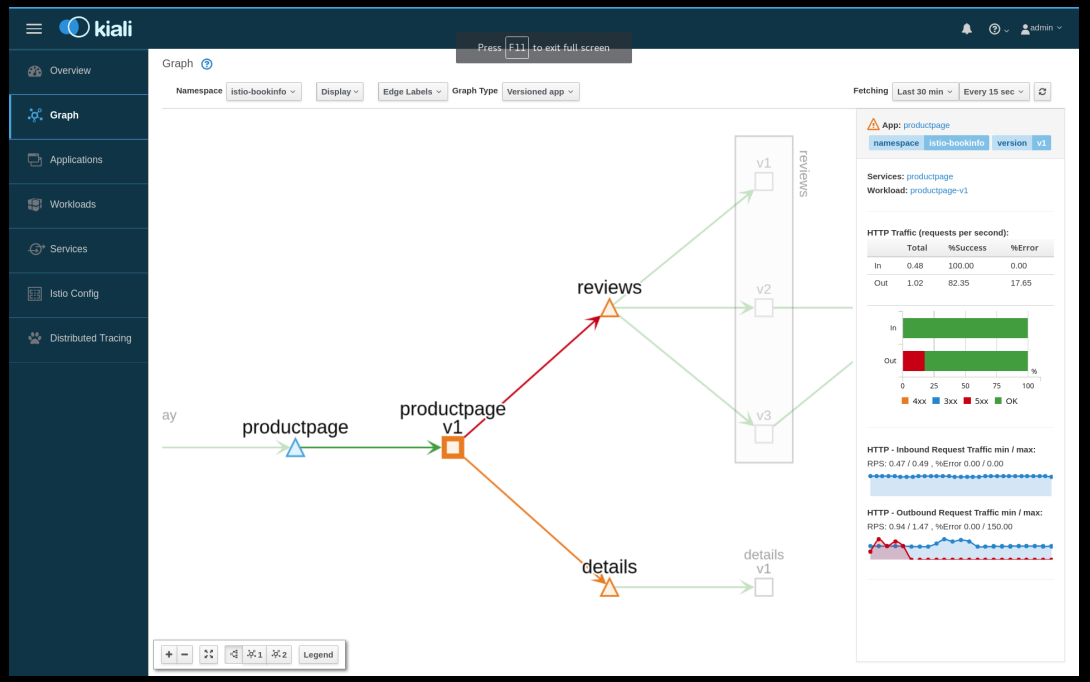
Goals

- Metrics without instrumenting apps
- Consistent metrics across fleet
- Trace flow of requests across services
- Portable across metric backend providers

CHALLENGE 6
I NEED TO VIEW WHAT IS GOING ON WHEN CRISIS ARISES

Kiali

Kiali (greek κιάλι)
monocular or *spyglass*
Visualise the service mesh topology, features like circuit breakers or request rates

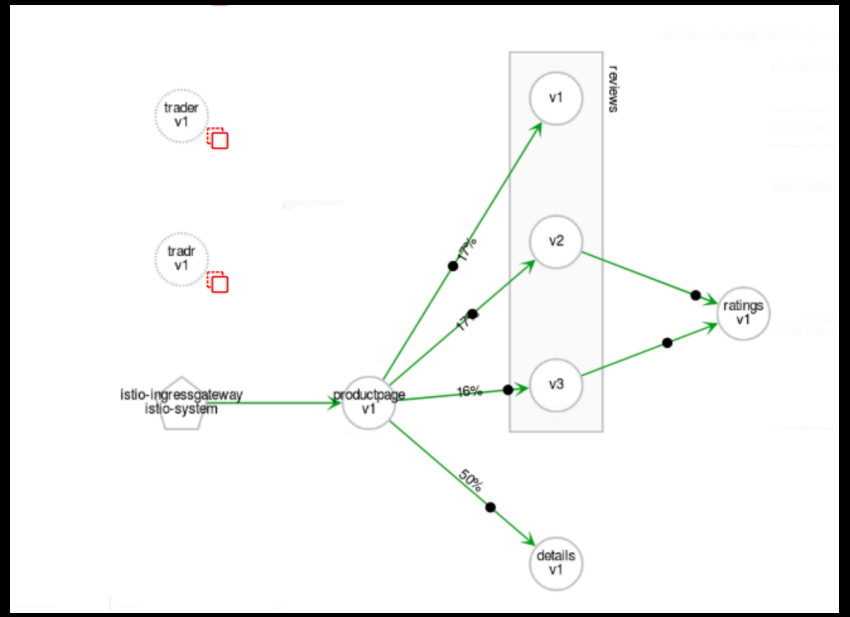


Features

Graph
- Health
- Types
- Side Panel
- Traffic Animation

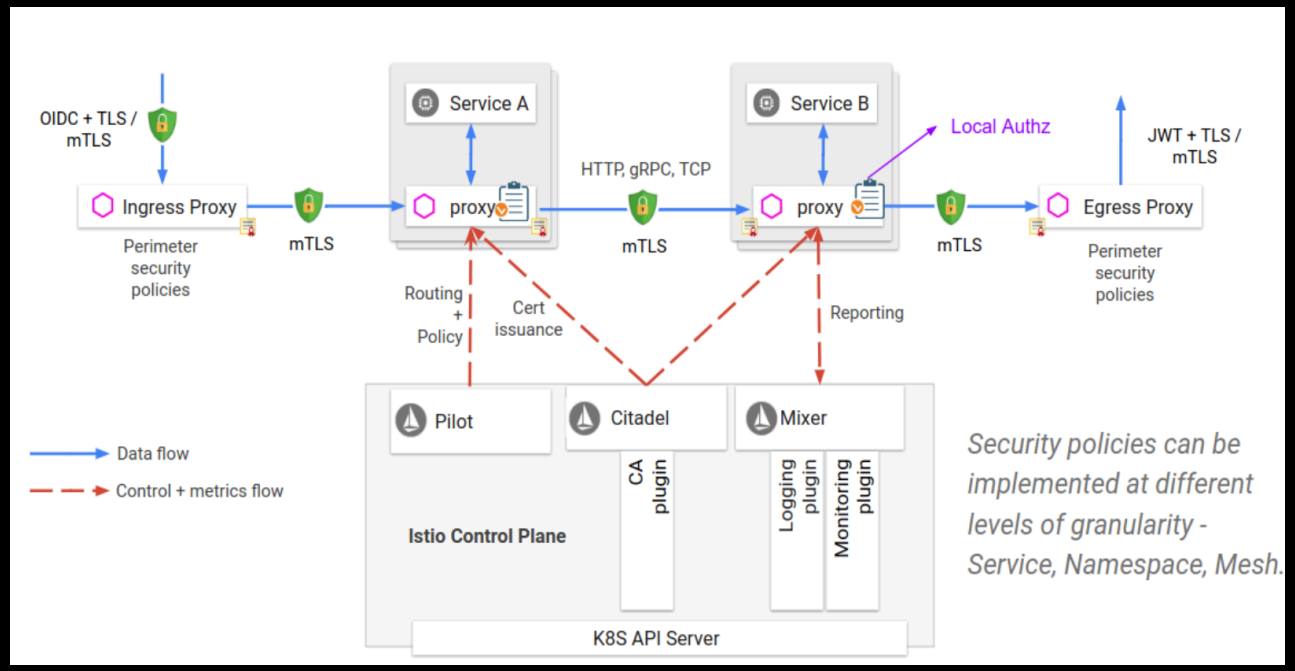
Applications, Workloads and Services
- Detailed Metrics

Traffic Routing
Istio compliance
Istio Configuration



CHALLENGE 6
I NEED TO VIEW WHAT IS GOING ON WHEN CRISIS ARISES

Security



Authentication

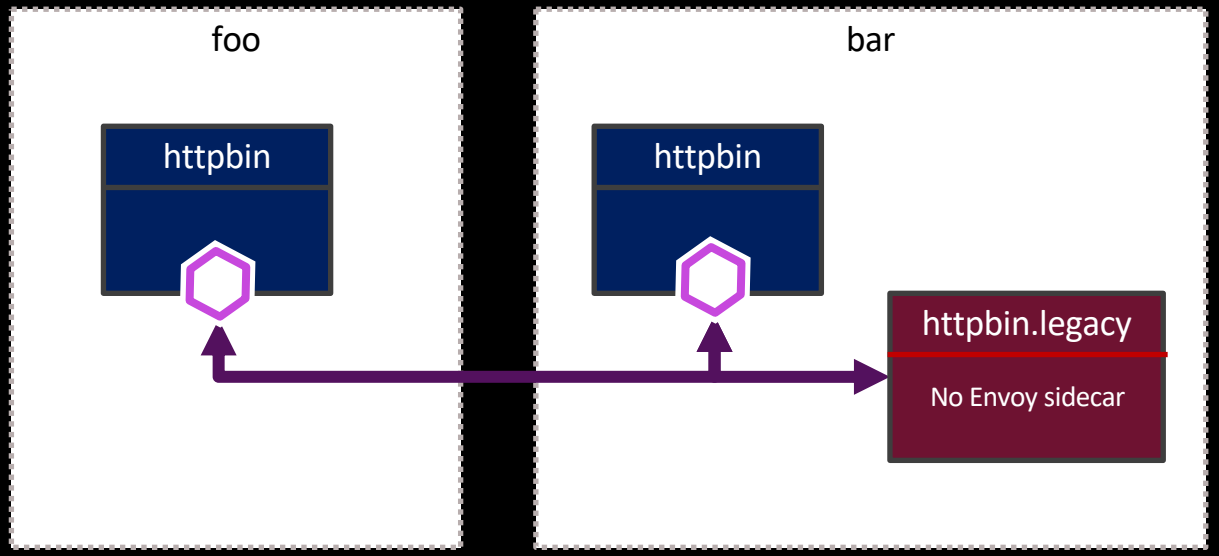
- Transport authentication, also known as service-to-service authentication
- Origin authentication, also known as end-user authentication

Authorization

- Based on RBAC
- Namespace-level, service-level and method-level access control for services

CHALLENGE 7
HOW CAN I SECURE MY SERVICES?

Security - Authentication

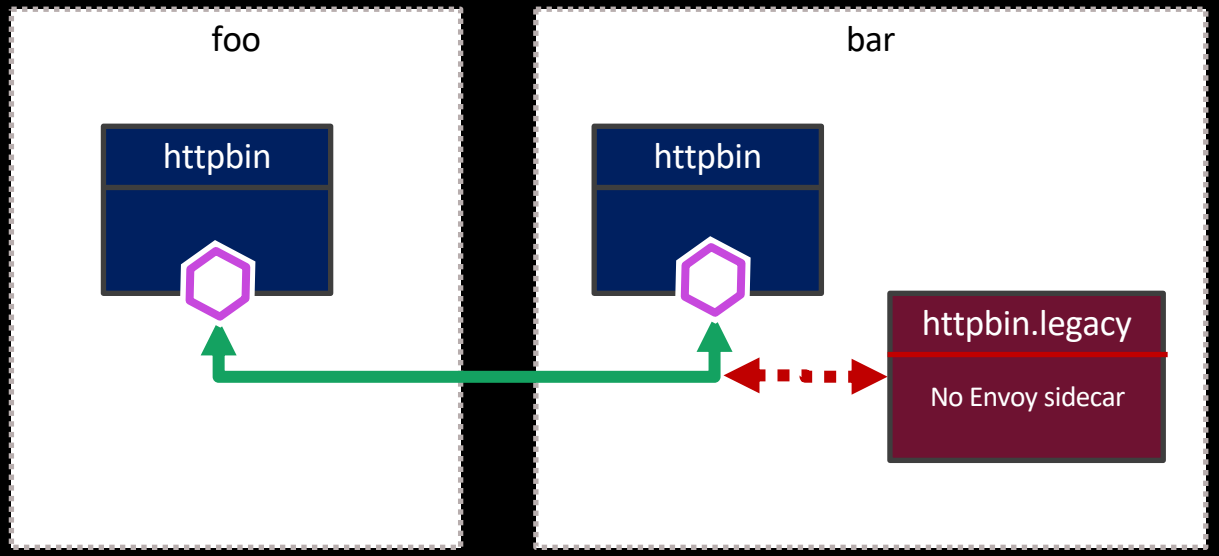


EXAMPLE

CHALLENGE 7
HOW CAN I SECURE MY SERVICES?

Not encrypted
Encrypted (TLS)
No communication

Security - Authentication

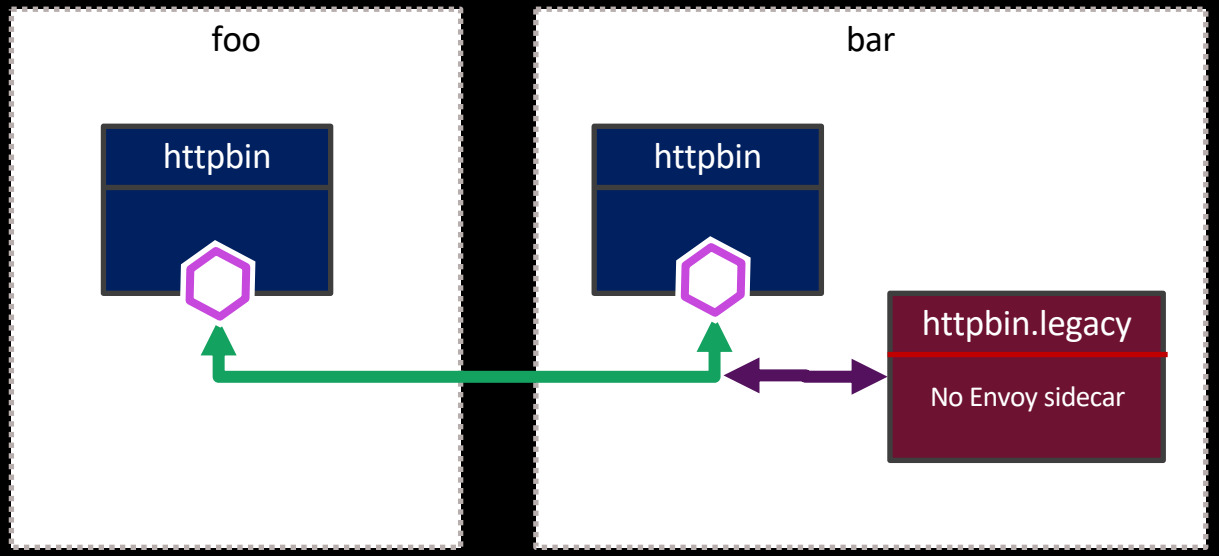



```
kind: DestinationRule
metadata:
  name: "default"
spec:
  host: "*.local"
  trafficPolicy:
    tls:
      mode: ISTIO_MUTUAL
```

CHALLENGE 7
HOW CAN I SECURE MY SERVICES?

Not encrypted
Encrypted (TLS)
No communication

Security - Authentication




```
kind: DestinationRule
metadata:
 name: "httpbin-legacy"
spec:
 host: "httpbin.legacy.svc.cluster.local"
 trafficPolicy:
   tls:
     mode: DISABLE
```

CHALLENGE 7
HOW CAN I SECURE MY SERVICES?

Not encrypted
Encrypted (TLS)
No communication

Security - Authorization

Istio Role Based Access

- **OFF**: Istio authorization is disabled.
- **ON**: enabled for all services in the mesh.
- **ON_WITH_INCLUSION**: enabled for all services specified in the inclusion field.
- **ON_WITH_EXCLUSION**: enabled for all services except the ones in the exclusion field.

```
kind: RbacConfig
metadata:
  name: my-user
spec:
  mode: 'ON_WITH_INCLUSION'
  inclusion:
    services:
    - "webapp.default.svc.cluster.local"
    - "frontend.default.svc.cluster.local"
    - "feedback.default.svc.cluster.local"
```

CHALLENGE 7
HOW CAN I SECURE MY SERVICES?

Security - Authorization

Istio Role Based Access

- **services**: A list of service names.
- **methods**: A list of HTTP method names (GET, POST,...)
- **paths**: HTTP paths (in the form of /packageName.serviceName/methodName)
- **constraints**: additional conditions for your rules.

```
kind: ServiceRole
metadata:
  name: service-viewer
spec:
  rules:
  - services:
    - "webapp.default.svc.cluster.local"
    - "frontend.default.svc.cluster.local"
    methods: ["GET", "HEAD"]
    paths: ["*"]
    constraints:
    - key: request.headers[version]
      values: ["v1", "v2"]
```

CHALLENGE 7
HOW CAN I SECURE MY SERVICES?

Security - Authorization

Access for:

- Service in **Namespace “default” only accessible by authenticated users** and services
- User: "*" assigns the ServiceRole to all (both authenticated and unauthenticated)

```
kind: ServiceRoleBinding
metadata:
  name: binding-products-all-authenticated-users
spec:
  subjects:
  - properties:
      source.principal: "*"
  - properties:
      source.namespace: "default”
  roleRef:
    kind: ServiceRole
    name: ”service-viewer"
```

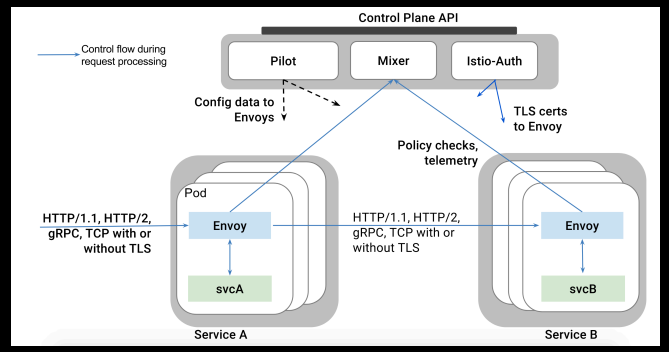
CHALLENGE 7
HOW CAN I SECURE MY SERVICES?

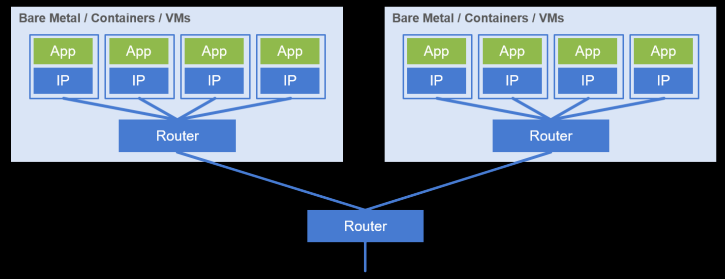
Security
Using Istio in concert with Calico

Istio

PROJECT CALICO

“RPC” — L7	Layer	“Network” — L3-4
Userspace	Implementation	Kernel
Pod	Enforcement Point	Node





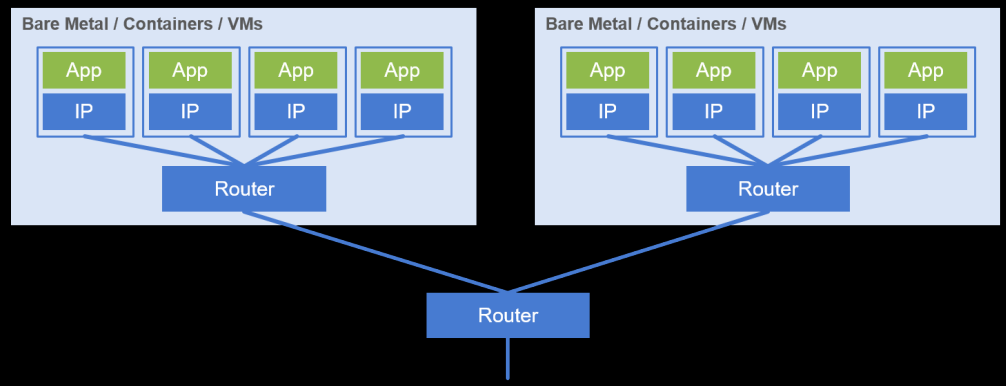
Security
Using Istio in concert with Calico

“RPC” — L7	Layer	“Network” — L3-4
Userspace	Implementation	Kernel
Pod	Enforcement Point	Node
Ideal for applying policy in support of operational goals, like service routing, retries, circuit-breaking, etc	Strengths	Universal, highly efficient, and isolated from the pods, making it ideal for applying policy in support of security goals

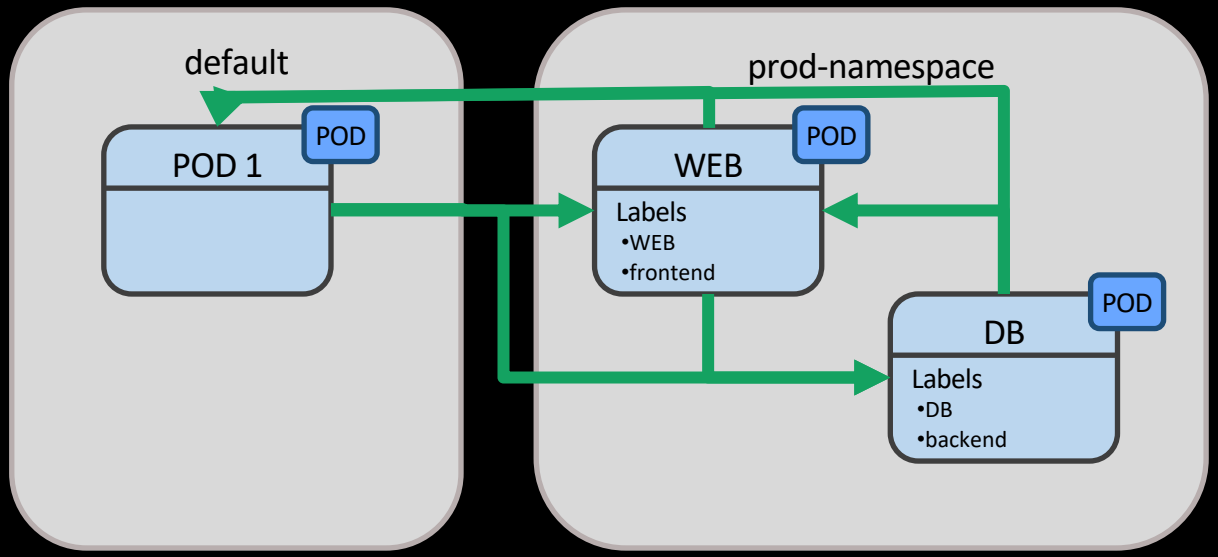
Security
Using Istio in concert with Calico

Operates at **Layer 3**, which is the network layer

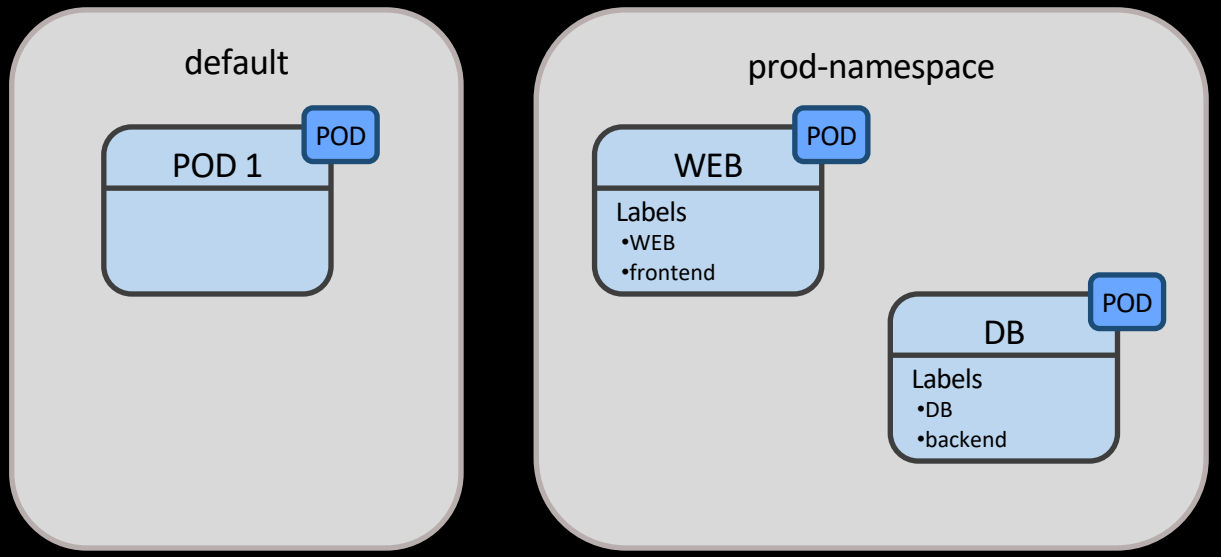
- Has the advantage of being **universal** (DNS, SQL, real-time streaming, ...)
- Can extend beyond the service mesh (including to **bare metal or VM** endpoints not under the control of Kubernetes).
- Calico's policy is enforced at the host node, outside the network namespace of the guest pods.
- Based on **iptables**, which are packet filters implemented in the standard Linux kernel, it is extremely fast.



Security
Using Istio in concert with Calico

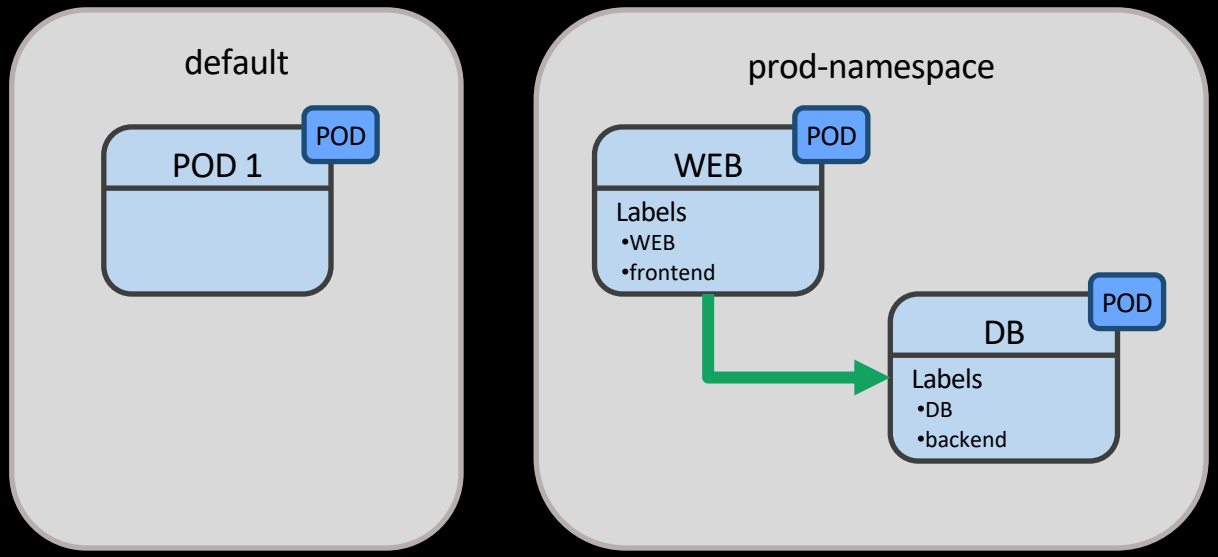


Security
Using Istio in concert with Calico




```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: default-deny-all
spec:
  podSelector:
    matchLabels: {}
```

Security
Using Istio in concert with Calico




```
kind: NetworkPolicy
metadata:
  name: access-frontend-backend
  namespace: prod-namespace
spec:
  podSelector:
    matchLabels:
      run: DB
  ingress:
    - from:
      - podSelector:
          matchLabels:
            run: WEB
```

Service Mesh - Bad Idea ?

A Service Mesh is not always the right solution...

- **Service Meshes are Opinionated**
  They are a *platform* solution. “Work their way”
- **Service Meshes are Complex**
  Adds considerable complexity with sidecars and control plane
- **Service Meshes can be Slow**
  Routing traffic through a series of proxies can get painfully slow (about 700 nodes → reflector)
- **Service Meshes are for Developers**
  Focused primarily on Developer view.

https://glasnostic.com/blog/comparing-service-meshes-linkerd-vs-istio

Getting started

- Go to https://istio.io/

  Download ISTIO Release

With Kubectl

```
$ kubectl apply -f install/kubernetes/helm/istio/templates/crds.yaml

$ kubectl apply -f install/kubernetes/istio-demo.yaml
```

With HELM Templating

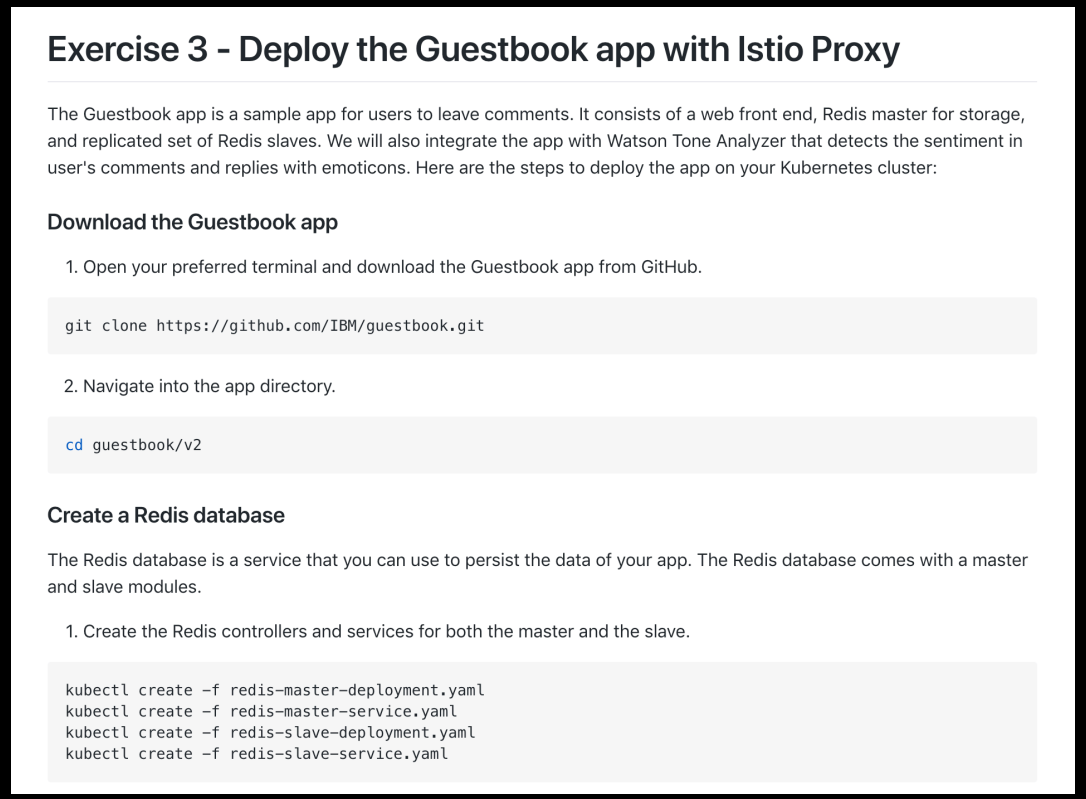
```
$ kubectl create namespace istio-system

$ helm template --name istio
      --namespace istio-system
      --set grafana.enabled=true
      --set servicegraph.enabled=true
      --set kiali.enabled=true > istio.yaml

$ kubectl apply -f istio.yaml
```

Getting started

- IBMs ISTIO 101 Hands On
- Go to https://github.com/IBM/istio101



Exercise 3 - Deploy the Guestbook app with Istio Proxy

The Guestbook app is a sample app for users to leave comments. It consists of a web front end, Redis master for storage, and replicated set of Redis slaves. We will also integrate the app with Watson Tone Analyzer that detects the sentiment in user's comments and replies with emoticons. Here are the steps to deploy the app on your Kubernetes cluster:

Download the Guestbook app

1. Open your preferred terminal and download the Guestbook app from GitHub.

```
git clone https://github.com/IBM/guestbook.git
```

2. Navigate into the app directory.

```
cd guestbook/v2
```

Create a Redis database

The Redis database is a service that you can use to persist the data of your app. The Redis database comes with a master and slave modules.

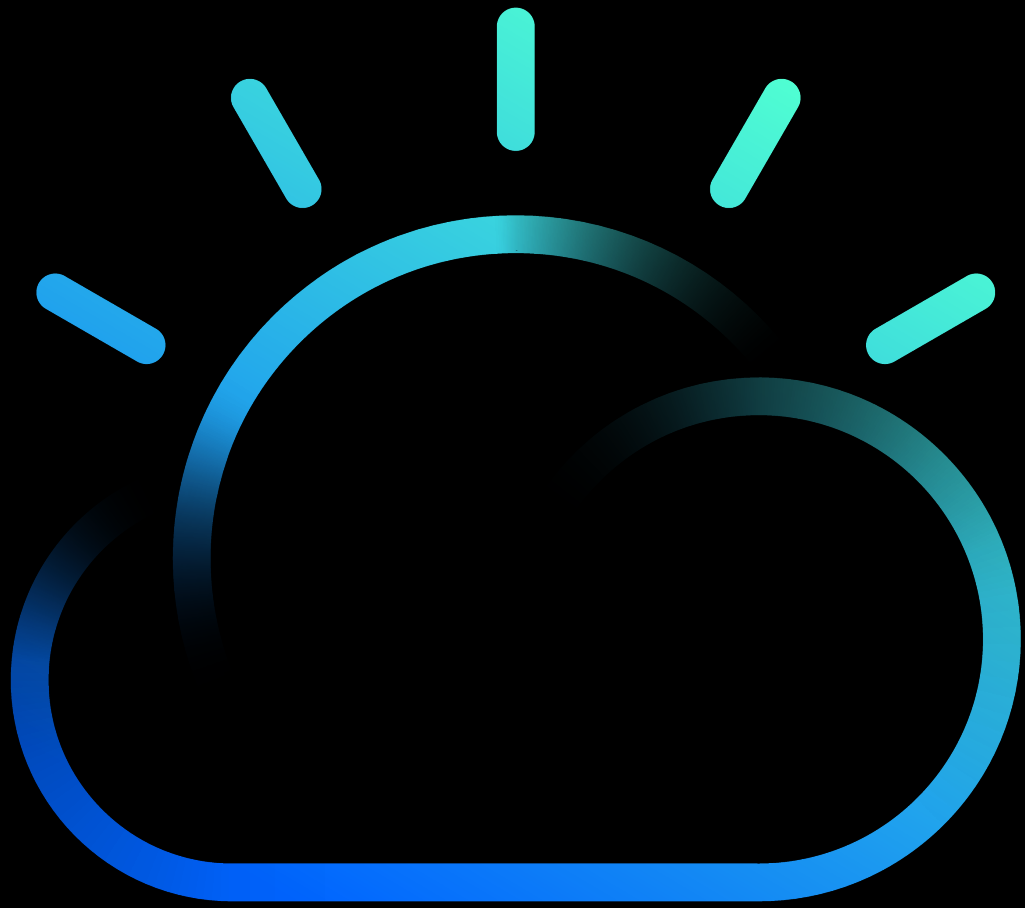
1. Create the Redis controllers and services for both the master and the slave.

```
kubectl create -f redis-master-deployment.yaml
kubectl create -f redis-master-service.yaml
kubectl create -f redis-slave-deployment.yaml
kubectl create -f redis-slave-service.yaml
```

Useful links

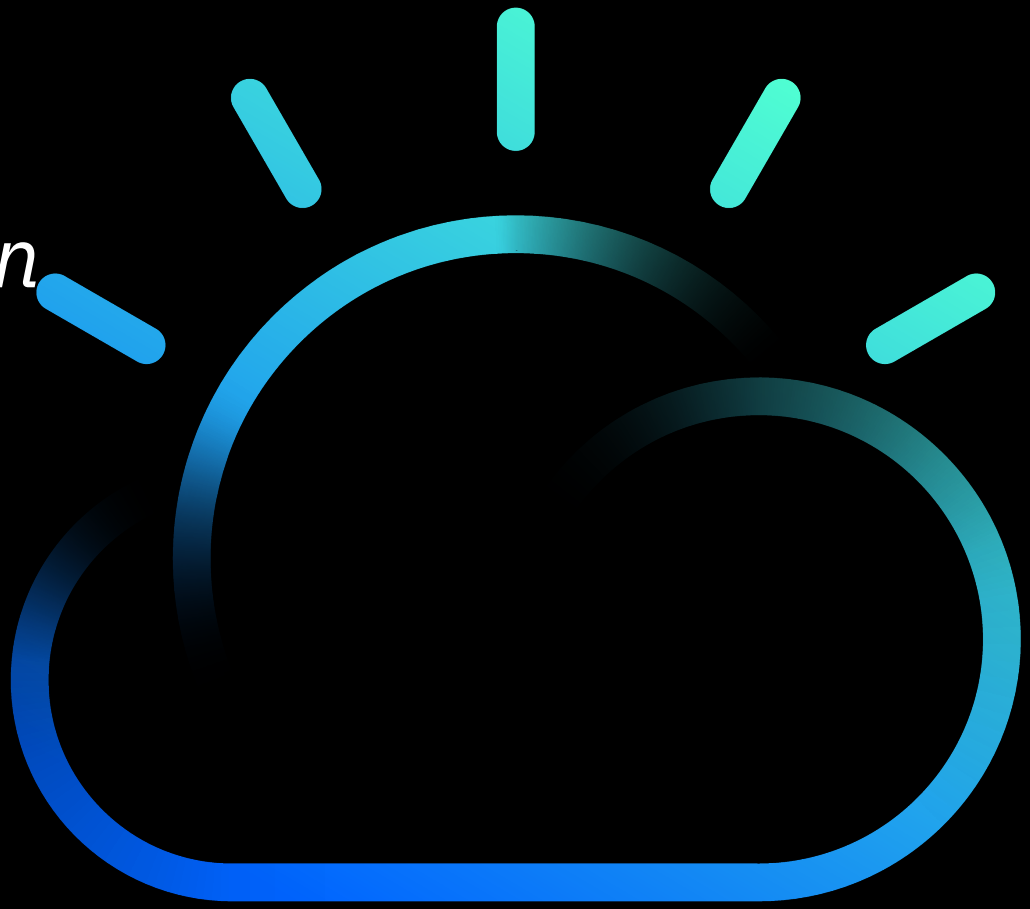
- Web istio.io
- Twitter: @Istiomesh
- Istio 101: https://github.com/IBM/istio101
- Traffic management using Istio: https://ibm.co/2F7xSnf
- Resiliency and fault-tolerance using Istio: https://bit.ly/2qStF2B
- Reliable application roll out and operations using Istio: https://bit.ly/2K9IRQX

QUESTIONS?



IBM **Cloud**

The path to Cloud
Mesh Networking - *Hands On*



06

IBM **Cloud**

Remember your Team Color

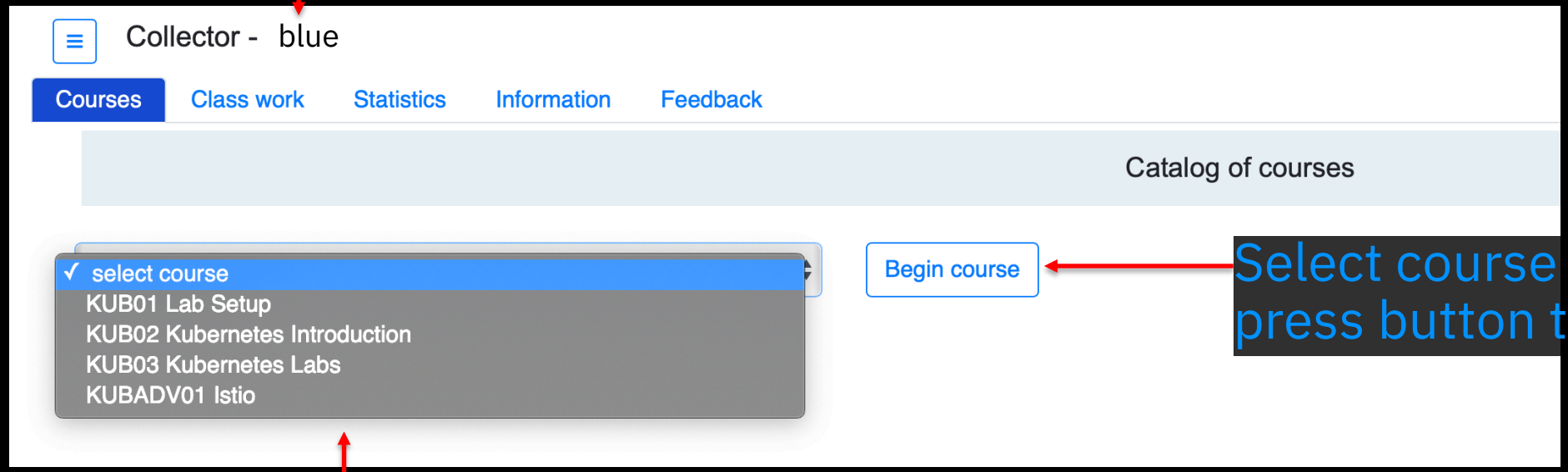
black 31701	olive 31711	peru 31715
white 31702	brown 31712	chocolate 31716
red 31703	lightblue 31713	orchid 31717
blue 31704	orange 31708	gold 31718
yellow 31705	purple 31709	pink 31719
lime 31706	maroon 31710	violet 31720
cyan 31707	firebrick 31714	

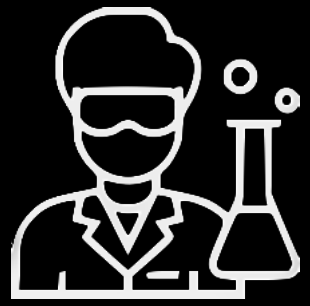
Collector - Accessing team web site

http://158.177.137.195:{port#}

blue 31704

Team name / color will be shown





KUBADV01 Istio

Lab 0 : Introduction

Lab 1 - Make sure minikube is running

Lab 2 - Installing Istio

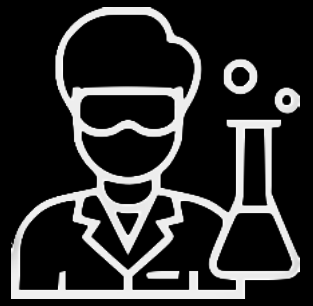
Lab 3 - Deploy the Bookinfo App

Lab 4 - Monitoring with Kiali

Lab 5 - Traffic flow management

Lab 6 - Access policy enforcement

Lab 7 - Telemetry data aggregation

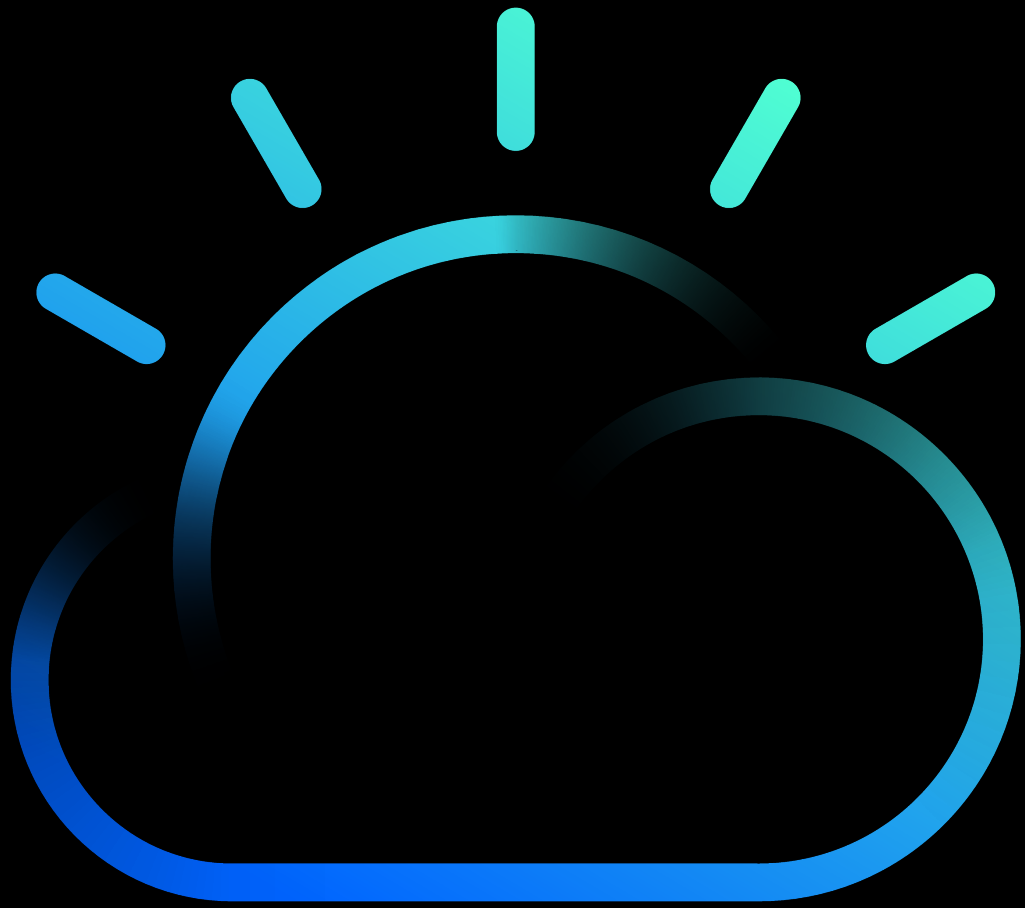


READY SET GO!!!!

http://158.177.137.195:{port#}

Duration: 60 mins

QUESTIONS?



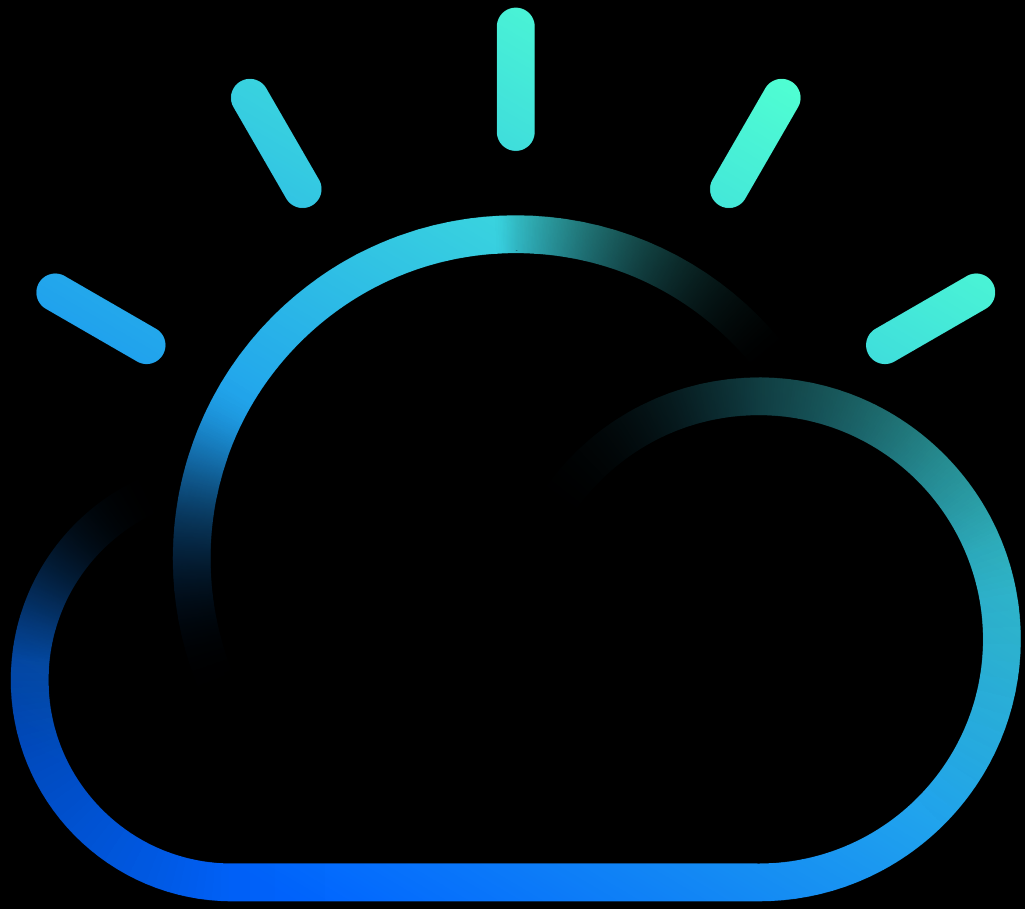
IBM **Cloud**

IBM

The path to Cloud

Serverless with Knative

07



IBM **Cloud**

Serverless paradigm

Source-to-image

Simply provide code to the platform and the platform manages all of the hosting aspects (e.g., building, hosting, scaling, etc.) for them.

Auto-scaling/scale-to-zero

Scales the application based on the load it is experiencing. Including scaling the application down to zero instances when it is not in use.

Short-lived functions

Splitting up the microservices into even smaller “functions” allows for a more fine-grained hosting model, meaning better resource utilization.

Event-driven

Optimized scaling by responding to events rather than simply always running and waiting for something to happen. This allows for a much more loosely-coupled architecture.

Knative

A platform for developers

to

build and run

serverless applications

atop Kubernetes

Open source project being developed by some of the key cloud innovators, including IBM, RedHat, Google, Pivotal, and SAP

Knative - building blocks

Knative solves these concerns through its three main components:

Build: Integrate building of container images into the specification of the application configuration. This allows for the **source-to-image model** where developers can **encapsulate the configuration** of their application alongside the specification of how to build their application images all at once.

Serving: **Event-driven** hosting scheme to ensure that the **applications are scaled** based on actual need, including **scaling down to zero** when appropriate. Also automatically **manages the rolling-out** of newer versions of the code and allow for advanced traffic routing (such as A/B testing), relying on Istio.

Eventing: **Core eventing primitives** to allow for the specification of interest in events from event sources (both internal and external to the cluster), as well as simple orchestration.

Knative – Serving

The Knative Serving project provides middleware primitives that enable:

- Automatically deploy containers and configure routing
- Automatically scale up and down, including scale-to-zero
- Point-in-time snapshots of deployments allows multiple versions of applications at once
- Easy rollbacks, blue-green deployments, partial load testing, etc.

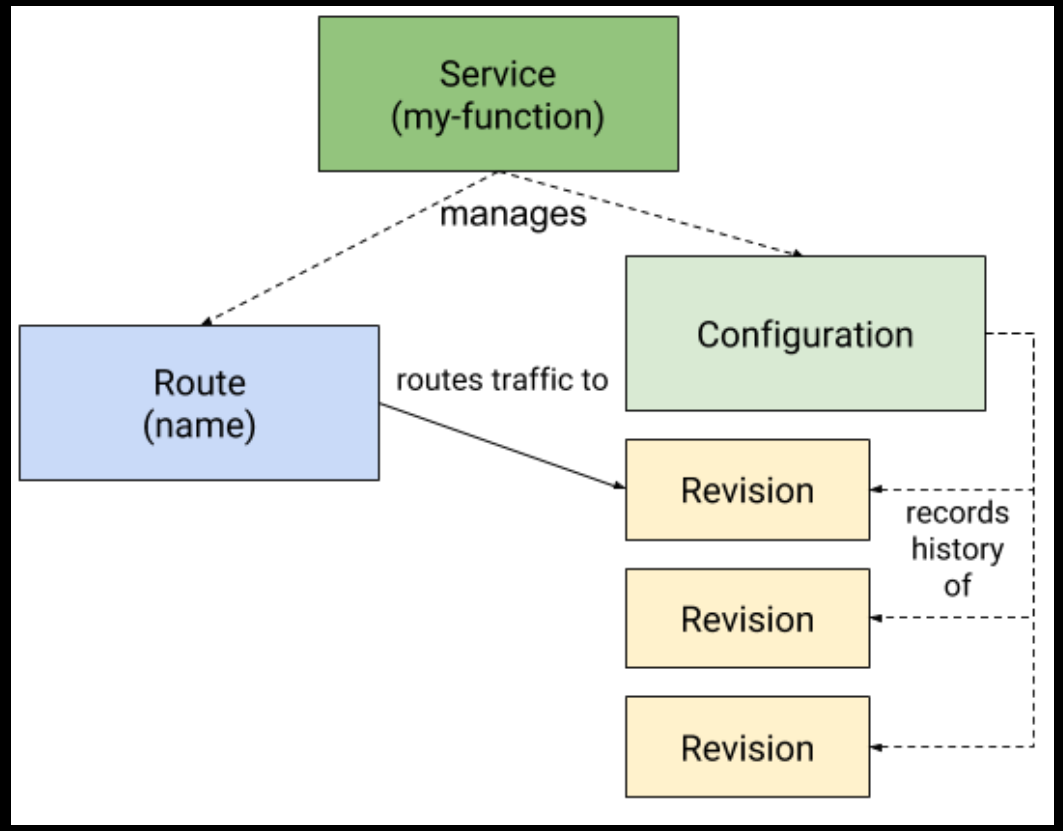
Knative – Serving

Revision: Single instance of your application. Contains a specific image and a specific set of configuration options - immutable.

Configuration: Responsible for defining the application image and its configuration, similar to a revision, except these values are mutable. Whenever these values are changed a new instance of the application (Revision) is created.

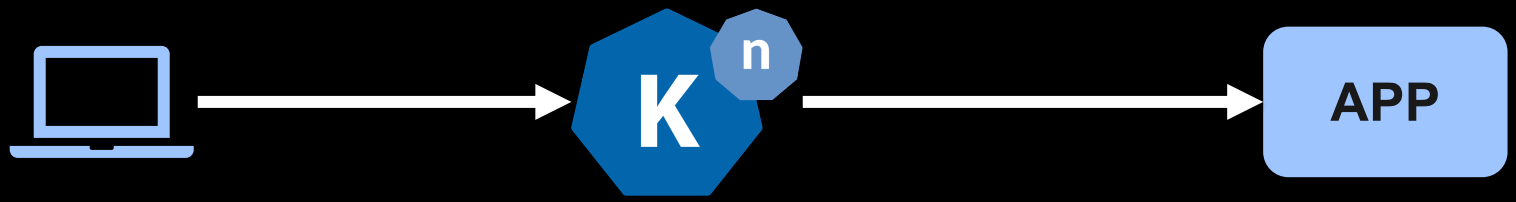
Route: Directing of traffic to a specific revision. By default sends all traffic to the latest revision. Used to specify traffic splits

(Knative) Service: Highest-level resource that ties together a complete serverless application. Only resource that users need to interact with to deploy their application.



Knative – Serving

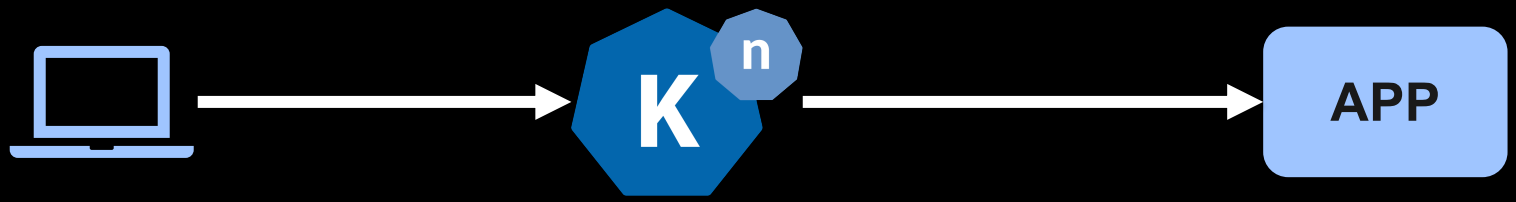
Deployment




```
apiVersion: serving.knative.dev/v1alpha1
kind: Service
metadata:
  name: knative-helloworld
  namespace: default
spec:
  runLatest:
    configuration:
      revisionTemplate:
        spec:
          container:
            image: docker.io/gswk/knative-helloworld:latest
```

Knative – Serving

Deployment



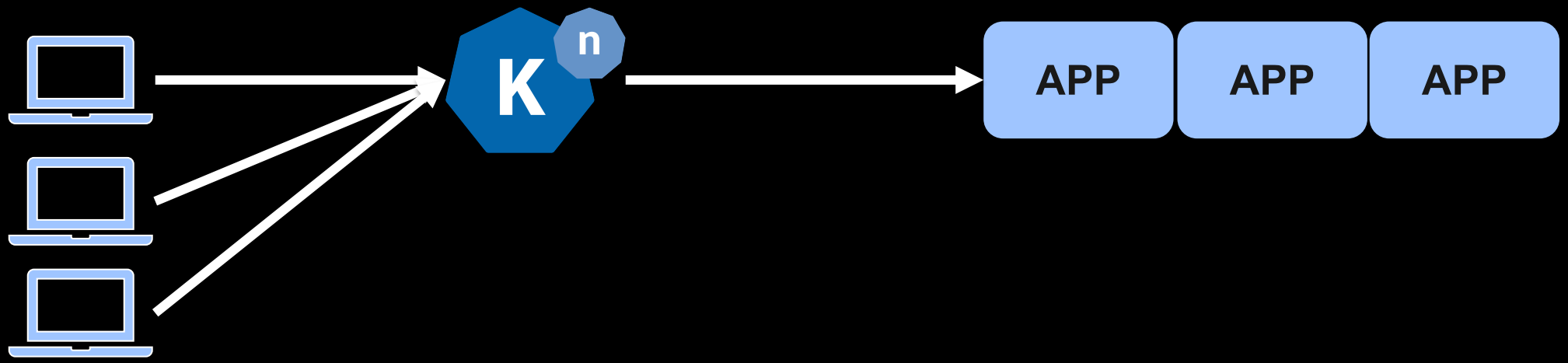
App reachable at:

{app-name}.{namespace}.{custom-domain}

knative-helloworld.default.example.com

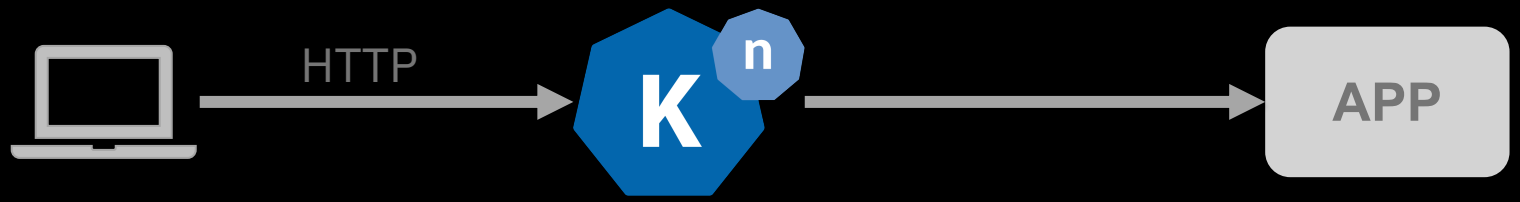
Knative – Serving

Scale up



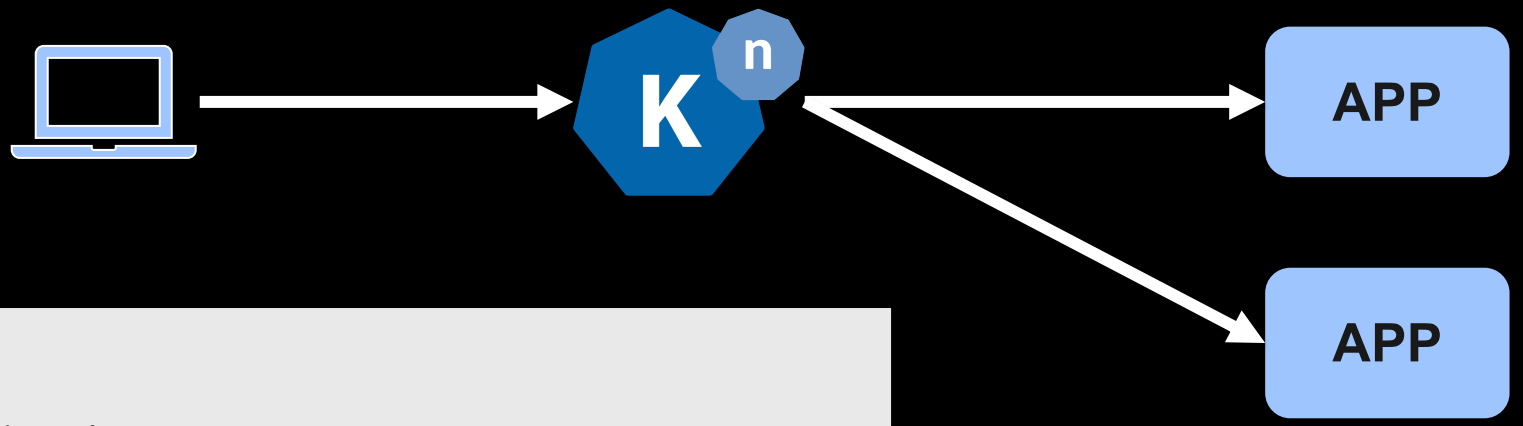
Knative – Serving

Scale down



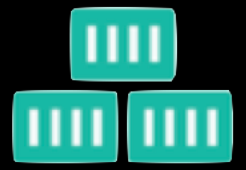
Knative – Serving

Routing (Istio)




```
kind: Route
metadata:
  name: knative-routing-demo
  namespace: default
spec:
  traffic:
  - revisionName: knative-routing-demo-00001
    name: v1
    percent: 10
  - revisionName: knative-routing-demo-00002
    name: v2
    percent: 90
```

Knative – Build

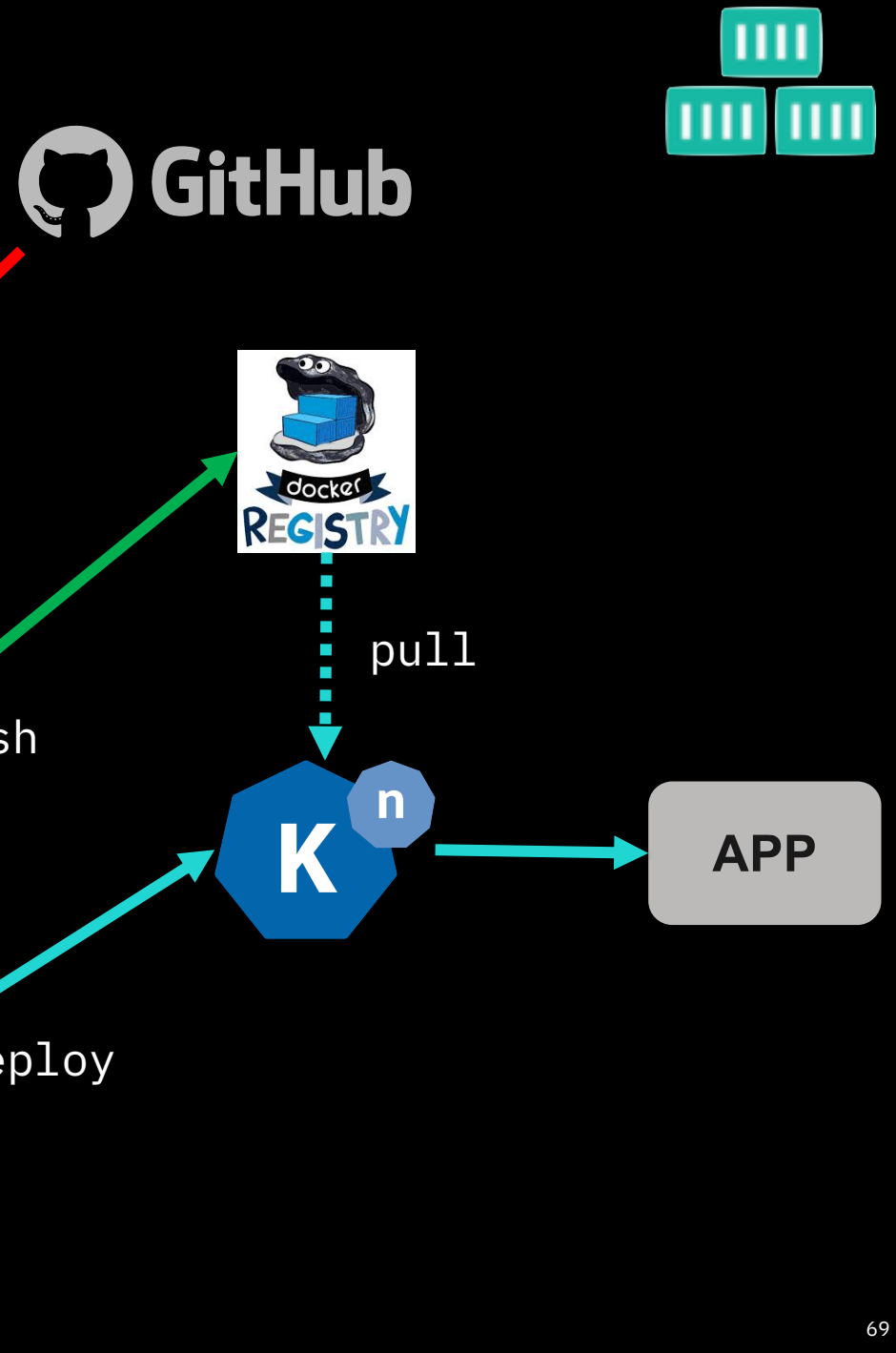


Build and package code on-cluster.

- Builds are ran completely within Kubernetes
- Code is pulled from git at build time
- Packaged as container images and pushed to a registry of your choice
- Build Templates - prepackaged descriptions of different ways to build code

Knative – Build

```
apiVersion: serving.knative.dev/v1alpha1
kind: Service
metadata:
  name: demo-from-source
spec:
  runLatest:
    configuration:
      build:
        kind: Build
        spec:
          source:
            git:
              url: https://github.com/niklaushirt/simple-app.git
              revision: master
          template:
            name: kaniko
            arguments:
              - name: IMAGE
                value: docker.io/niklaushirt/demo-from-source:latest
      revisionTemplate:
        spec:
          container:
            image: docker.io/niklaushirt/demo-from-source:latest
```

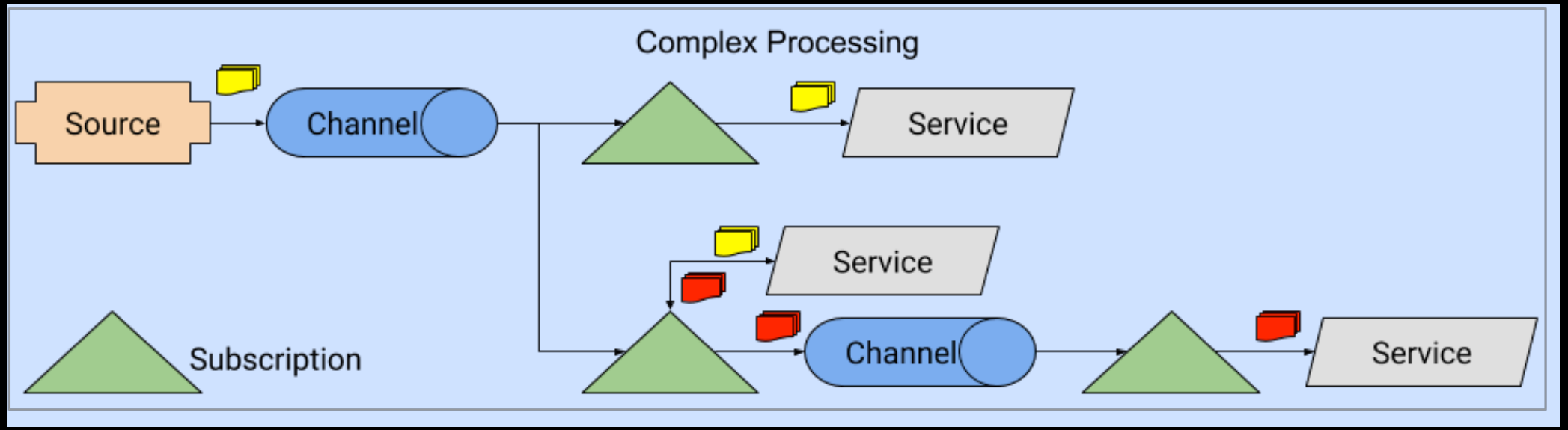


Knative – Eventing

Designed to address a common need for cloud native development, providing composable primitives to enable late-binding event sources and event consumers.

- Services are loosely coupled and can be developed and deployed independently on, and across a variety of platforms (for example Kubernetes, VMs, SaaS or FaaS).
- Event producers and event sources are independent.
- Other services can be connected to the Eventing system.
- Ensure cross-service interoperability.

Knative – Eventing



Some Event Sources

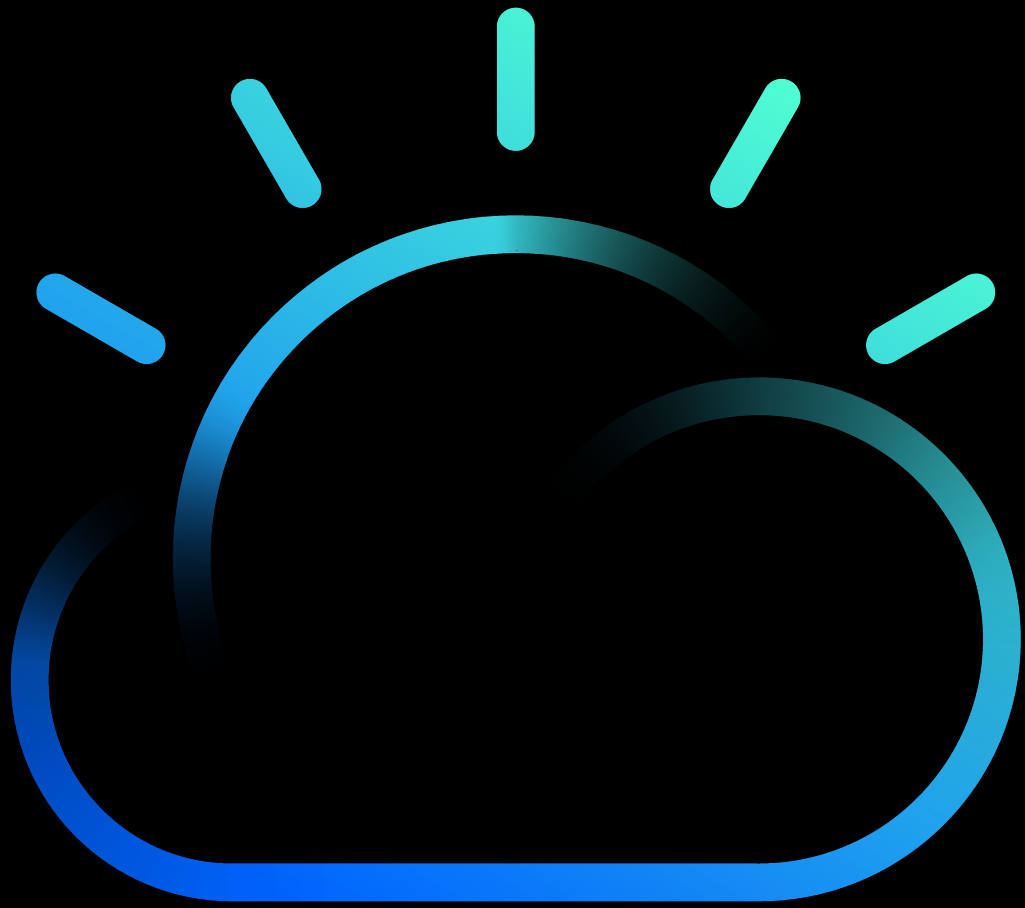
Kubernetes	Brings Kubernetes cluster events into Knative.
GitHub	Registers for events of the specified types on the specified GitHub organization/repository
Cron Job	Uses an in-memory timer to produce events on the specified Cron schedule.
....	

https://github.com/knative/docs/blob/master/eventing/sources/README.md

Getting started

- Go to https://github.com/knative/
- Istio and Knative: Extending Kubernetes for a New Developer Experience ibm.biz/Bd2Vet
- Knative: What is it and why should you care? ibm.biz/Bd2V8A
- Install Knative and Deploy an App on IBM Cloud ibm.biz/Bd2V8C
- IBMs Knative 101 Hands On https://github.com/IBM/knative101

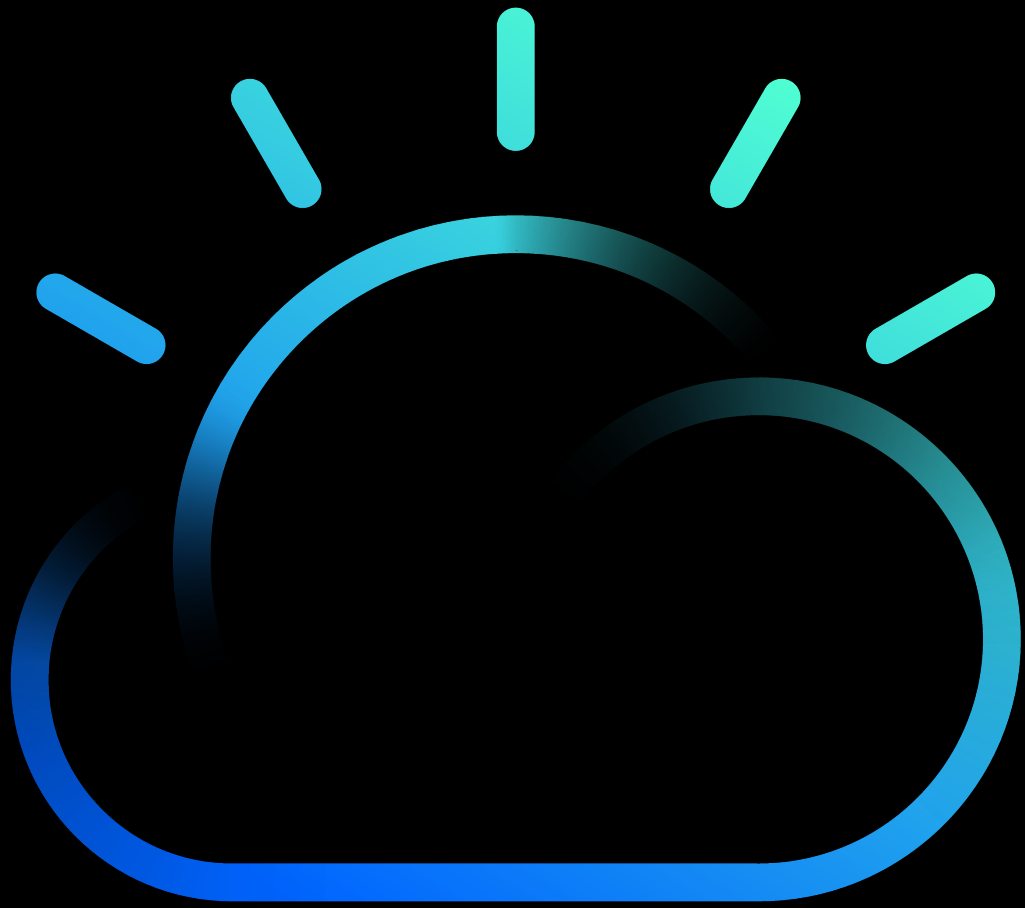
QUESTIONS?



IBM **Cloud**

The path to Cloud
Serverless with Knative
Hands On

08



IBM **Cloud**

Remember your Team Color

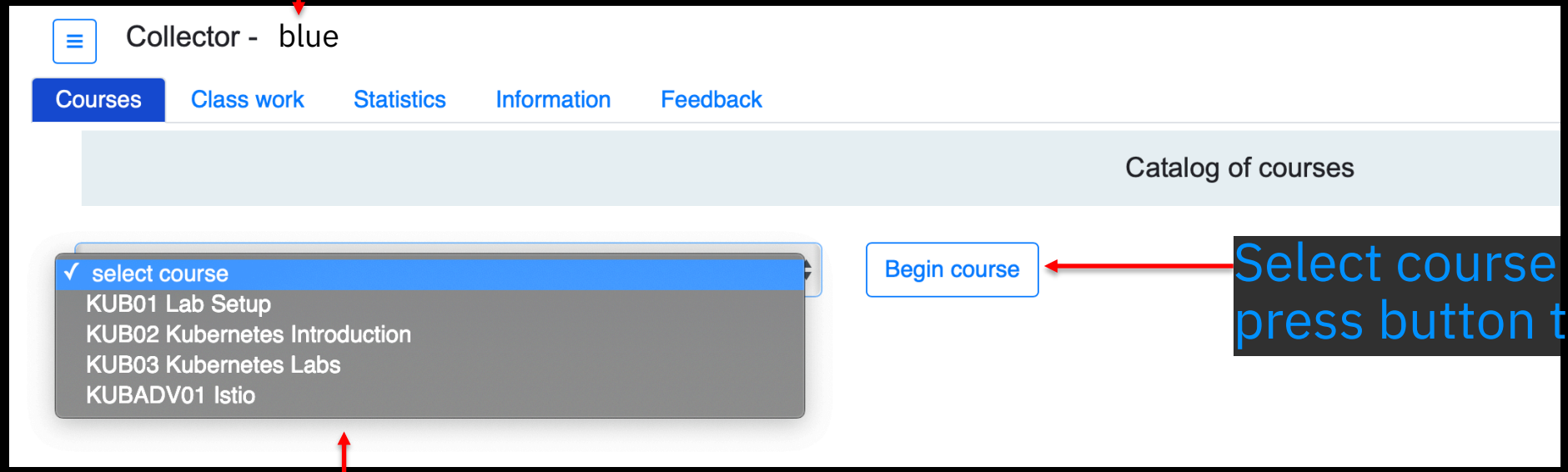
black 31701	olive 31711	peru 31715
white 31702	brown 31712	chocolate 31716
red 31703	lightblue 31713	orchid 31717
blue 31704	orange 31708	gold 31718
yellow 31705	purple 31709	pink 31719
lime 31706	maroon 31710	violet 31720
cyan 31707	firebrick 31714	

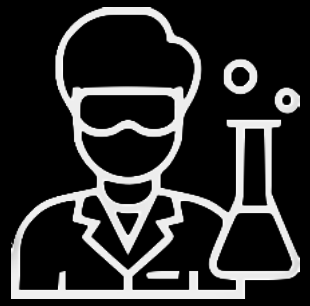
Collector - Accessing team web site

http://158.177.137.195:{port#}

blue 31704

Team name / color will be shown





KUBADV02 Knative

Lab 1 - Introduction

Lab 2 - Clone the Application Repo and Provide Container Registry Credentials

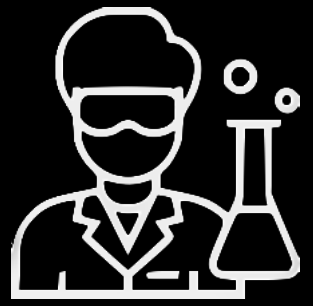
Lab 3 - Install Istio and Knative

Lab 4 - Deploy Our First Knative Application

Lab 5 - Build and Deploy our Knative Application

Lab 6 - Deploy vnext Version Using knctl

Lab 7 - A/B Testing with knctl

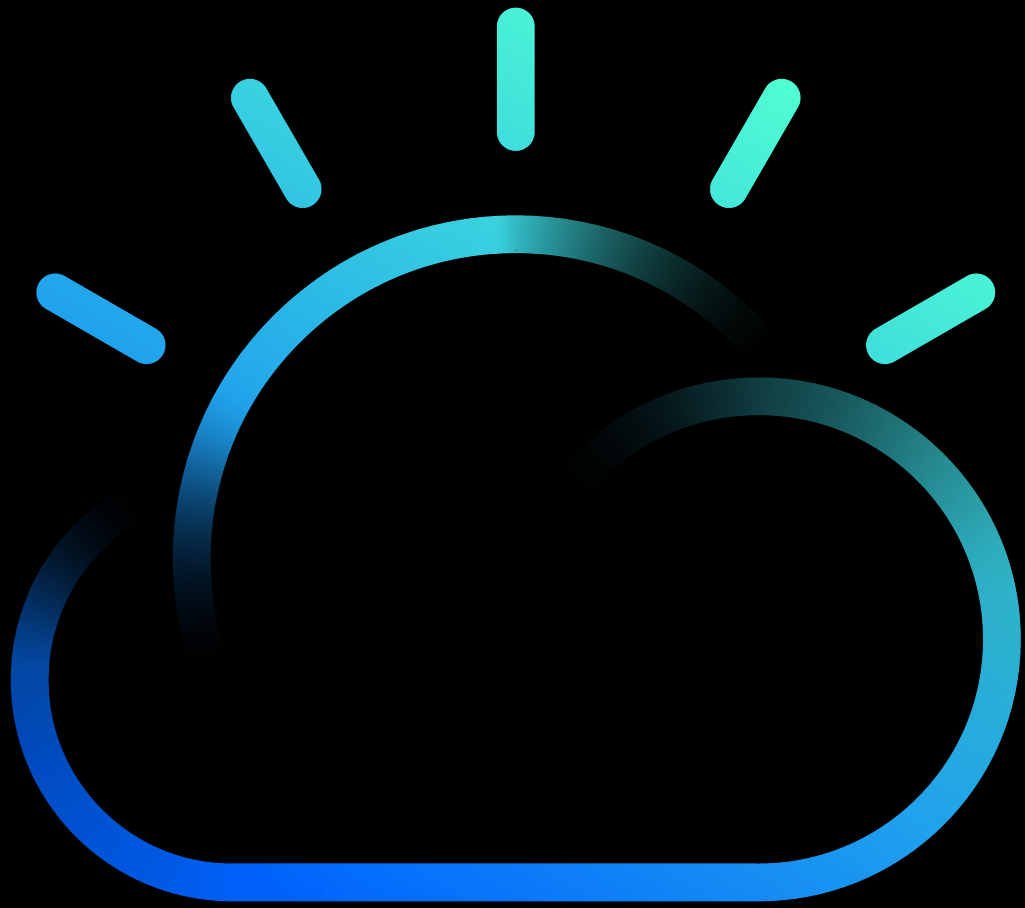


READY SET GO!!!!

http://158.177.137.195:{port#}

Duration: 60 mins

QUESTIONS?

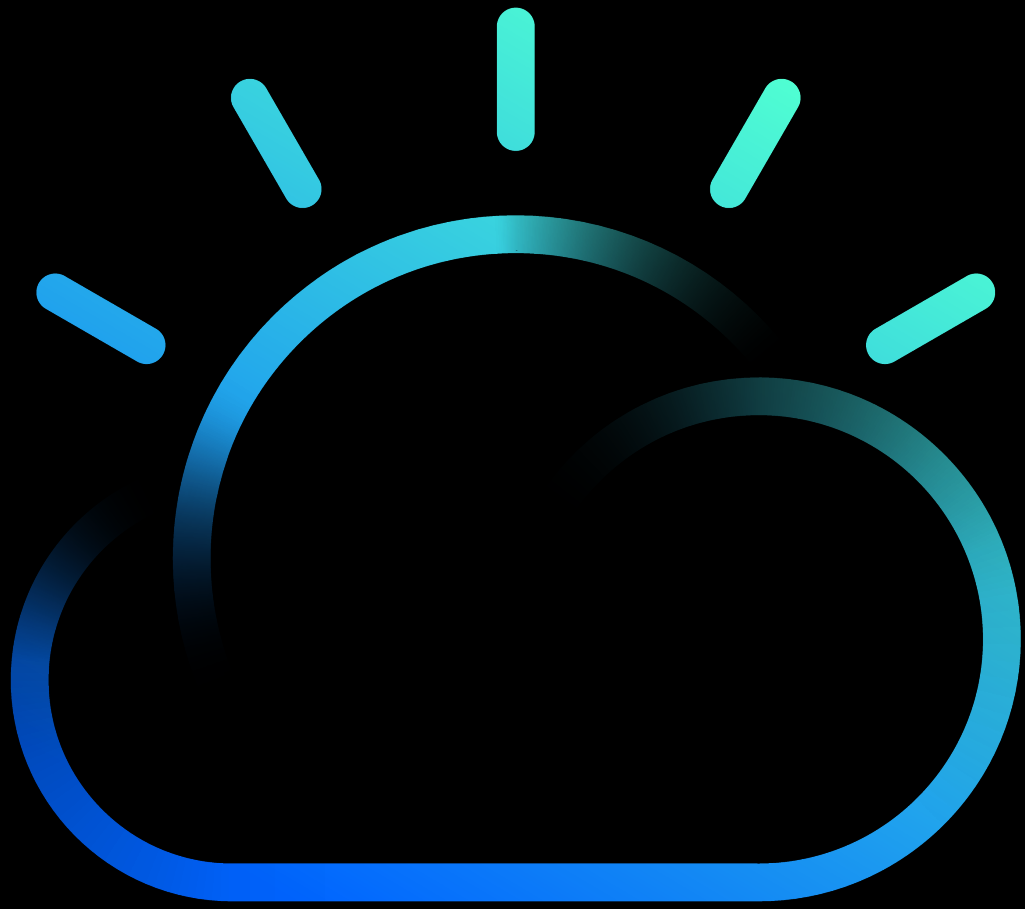


IBM **Cloud**

The path to Cloud

GitOps with ArgoCD

09



IBM Cloud

ArgoCD

A tool for developers

for

declarative, GitOps continuous delivery

atop Kubernetes

GitOps

At its core, GitOps refers to
a set of practices and tooling
that put
Git at the center of the DevOps toolchain
and as the
source of truth
for what should be deployed on the cluster.

With GitOps, developers and operators use familiar Git workflows to define, review, approve, and audit changes to their infrastructure and applications, whereas automated tools take care of synchronizing the live state of their cluster with the desired state described in Git.

ArgoCD

Argo CD is a declarative, GitOps continuous delivery tool for Kubernetes.

Application definitions, configurations, and environments should be declarative and version controlled. Application deployment and lifecycle management should be automated, auditable, and easy to understand.

- kustomize applications
- helm charts
- ksonnet applications
- jsonnet files
- Plain directory of YAML/json manifests

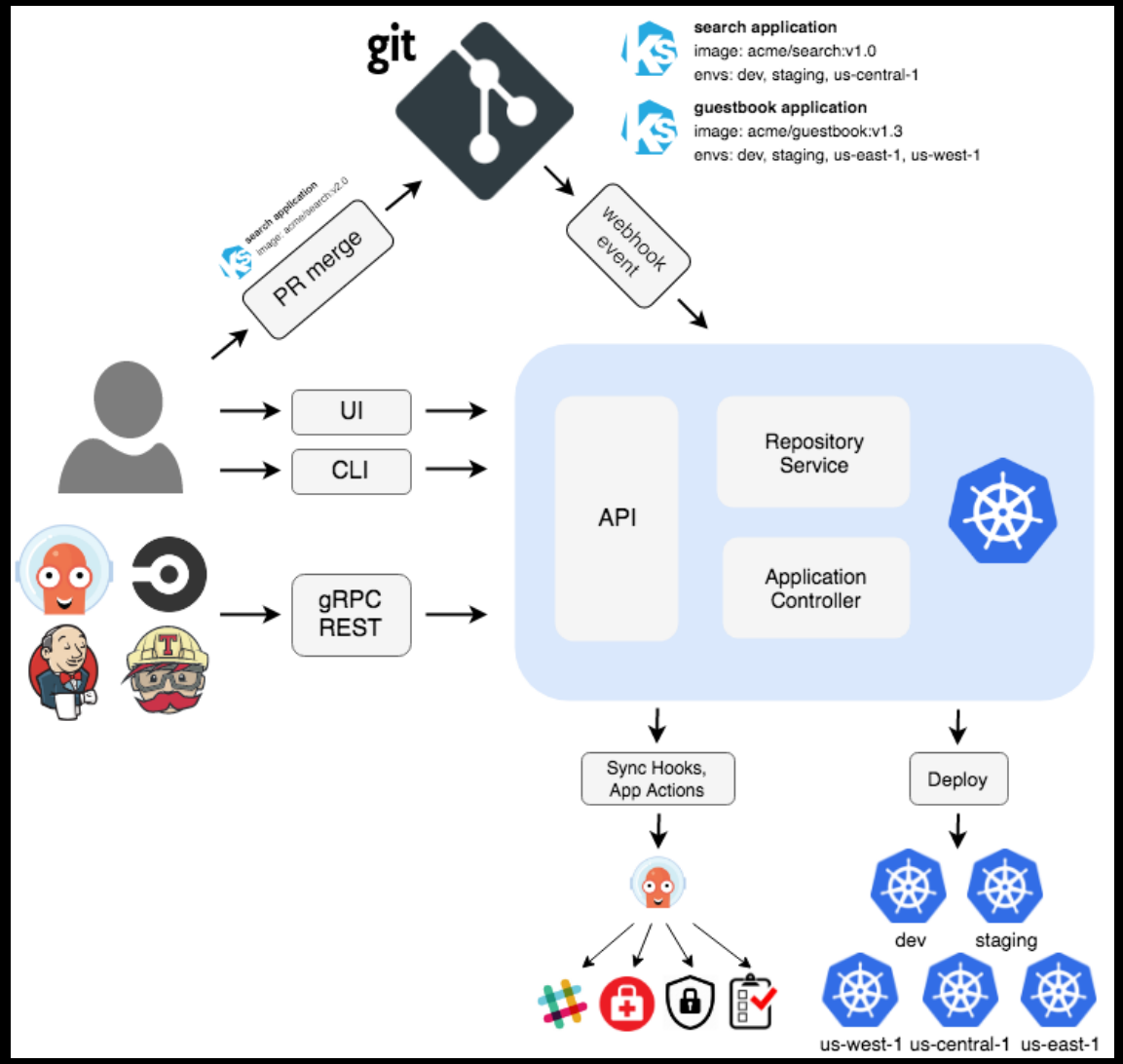
ArgoCD

Some notable features of Argo CD:

- Multiple Git repositories and cluster destinations
- Automated or manual synchronization
- Pruning of deleted resources
- Rollback/roll-anywhere
- Multiple manifests template formats (Helm, ~~Ksonnet~~, Kustomize) or plain YAML manifests
- Role-based access control
- Webhook integration (GitHub, GitLab, BitBucket)
- Git branch tracking or tag/commit pinning
- Declarative management of its own configuration
- Continuous monitoring of deployed applications
- Audit trail and history for application events and API calls
- Health assessment statuses on all components of the application
- Web console, CLI, and gRPC/REST API
- SSO Integration (OIDC, LDAP, SAML 2.0, and others)
- PreSync, Sync, PostSync hooks for complex application rollouts (e.g., canary upgrades, blue/green)

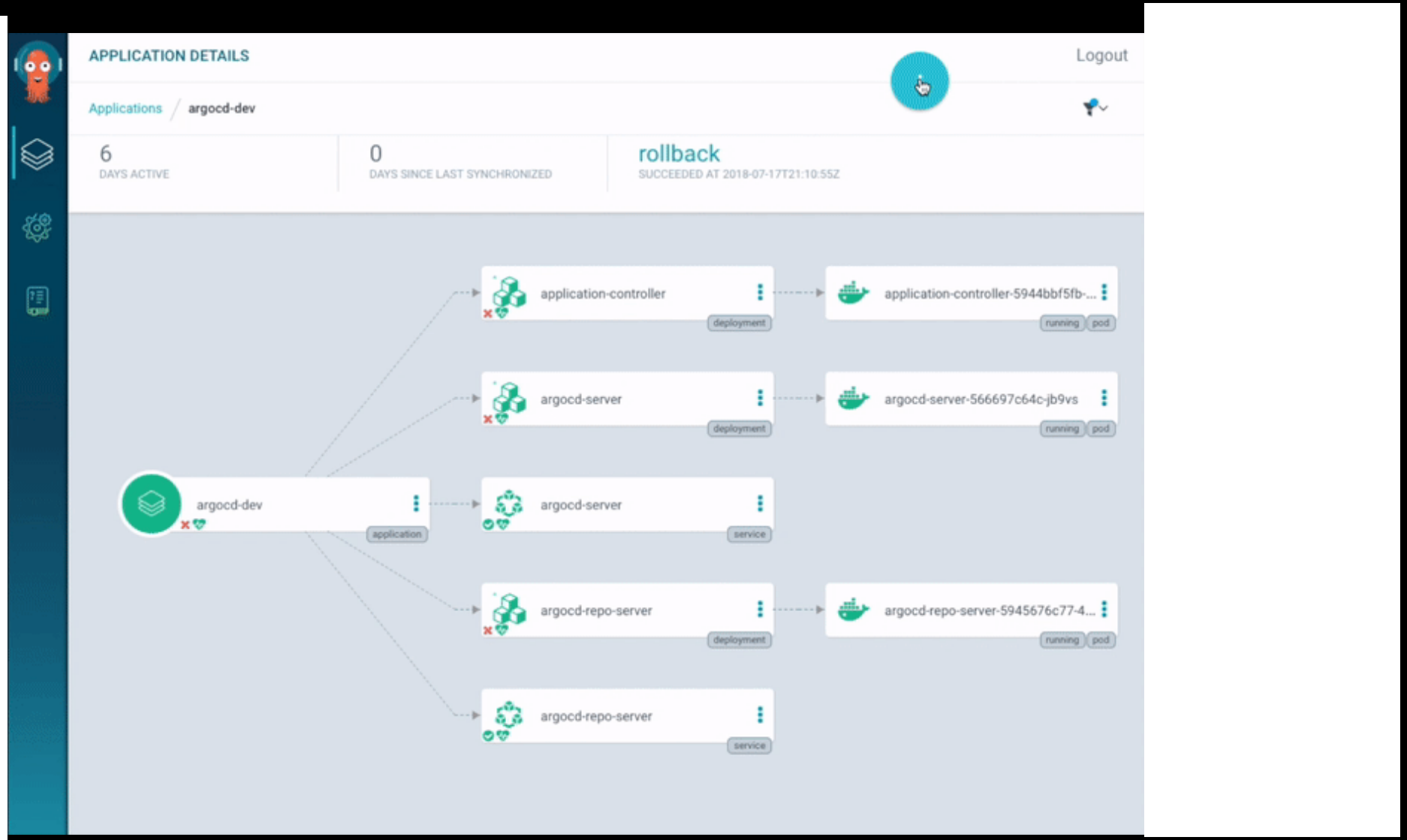
ArgoCD

Argo CD is a declarative, GitOps continuous delivery tool for Kubernetes.

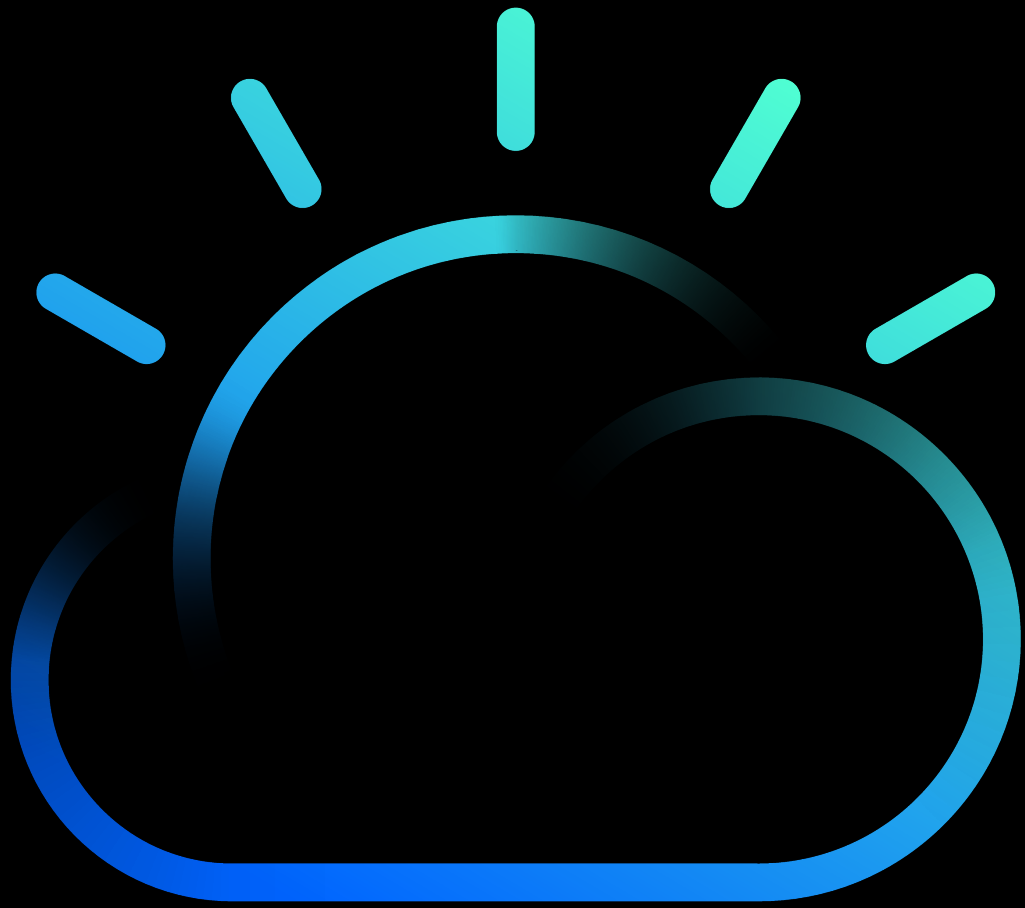


ArgoCD

Argo CD is a declarative, GitOps continuous delivery tool for Kubernetes.



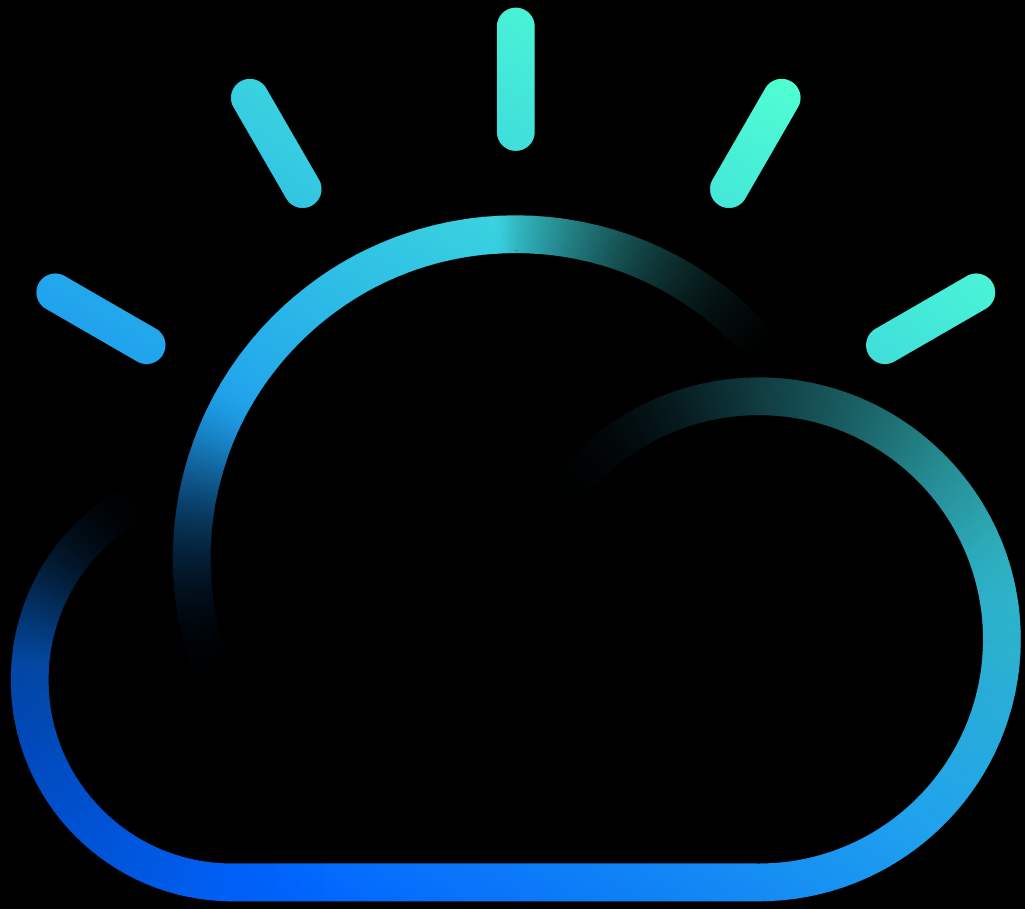
QUESTIONS?



IBM **Cloud**

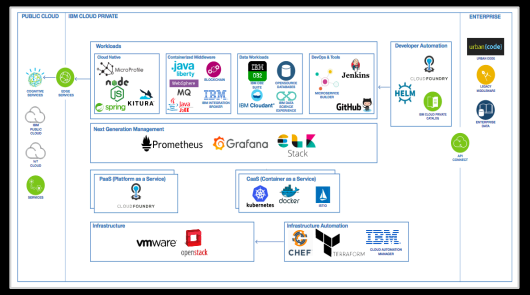
The path to Cloud
Wrap Up

99



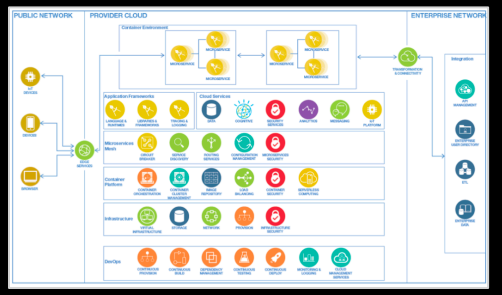
IBM **Cloud**

IBM Reference Architectures



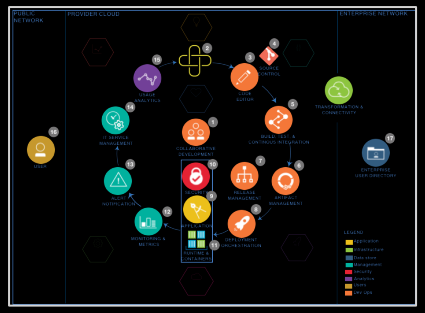
Private cloud reference architecture
http://ibm.biz/BdYF9S

Microservices reference architecture
http://ibm.biz/BdYF9e

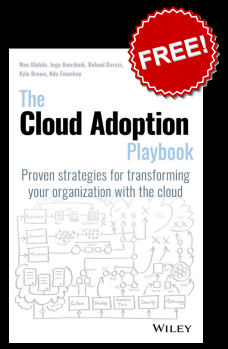


IBM Cloud Garage Method
http://ibm.biz/BdYF9u

DevOps reference architecture
http://ibm.biz/BdYF9T

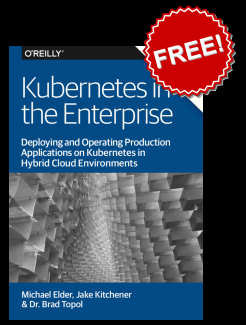


Cloud Adoption - Further reads



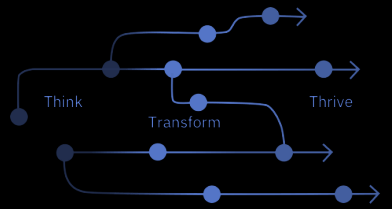
The de facto guide to improving your enterprise with the cloud, created by distinguished members of our Solution Engineering team
http://ibm.biz/playbook

IBM Cloud Private for Dummies
http://ibm.biz/BdZedY



Deploying and Operating Production Applications on Kubernetes in Hybrid Cloud Environments
http://ibm.biz/k8sintheenterprise

Cloud Adoption and Transformation Consultancy
http://ibm.biz/BdYFCx



IBM Cloud Private Application Developer's Guide
http://www.redbooks.ibm.com/abstracts/sg248441.html

THANK YOU!!!!

Niklaus Hirt

nikh@ch.ibm.com

@nhirt