Advanced Kubernetes Workshop

Fairwinds

Wifi: GoogleGuest

Chrome Browser

Incognito Mode

explore.qwiklabs.com

Agenda

What you build

Lab 0: Workshop Setup

Kubernetes Primer

Lab 1: Build with Skaffold

Lab 2: Deploy with Cloud Build and Spinnaker

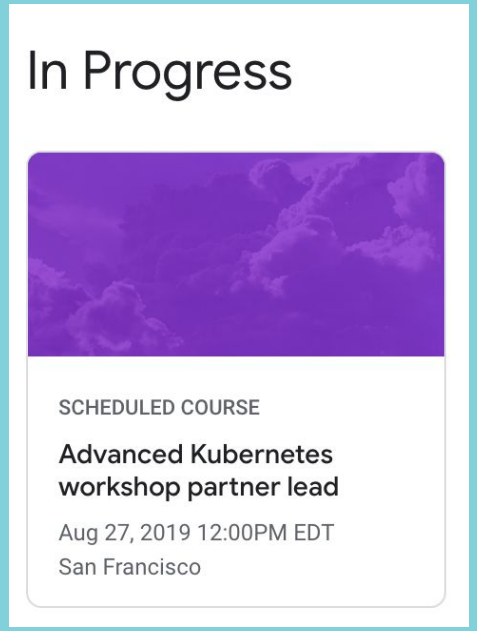
Break

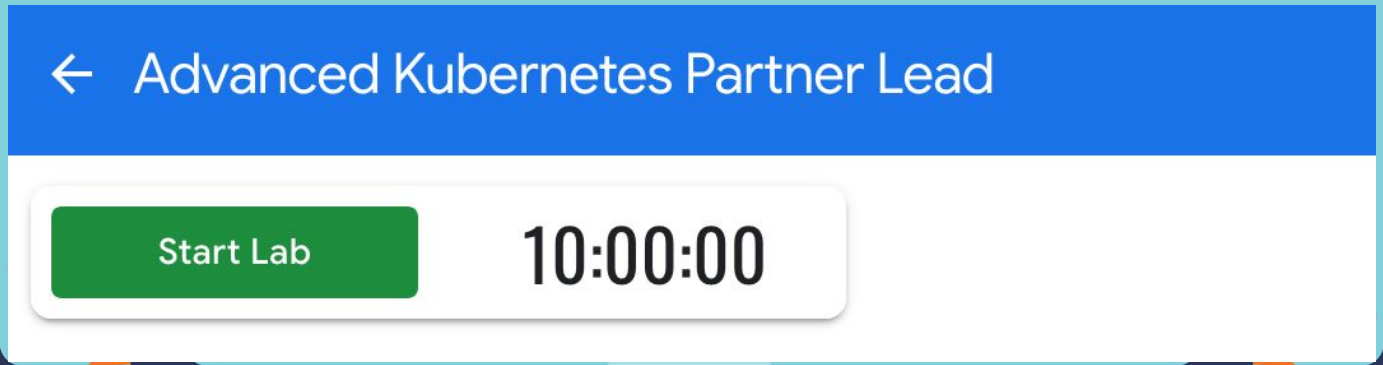
Lab 3: Control and secure with Istio

Lab 4: Monitor and troubleshoot with Istio

01

Lab 0: Setup





End Lab 09:59:47

Open Google Console

Caution: When you are in the console, do not deviate from the lab instructions. Doing so may cause your account to be blocked. Learn more.

Username

googlece41722_student(

Password

TySw92Ly4Cps

GCP Project ID

qwiklabs-gcp-9f93deb1(

Advanced Kubernetes Partner Lead

Cloning the repo

Get all files associated with this workshop by cloning the following repo.

```
git clone https://github.com/FairwindsOps/advanced-kubernetes-workshop
```

Running install scripts

Run the following command to kick off the script to install all resources for this workshop.

```
source ~/advanced-kubernetes-workshop/setup/setup.sh
```

Note: This step takes about 20 - 25 minutes to complete. Please ensure you do not close Cloud Shell during the setup.

00

What you build

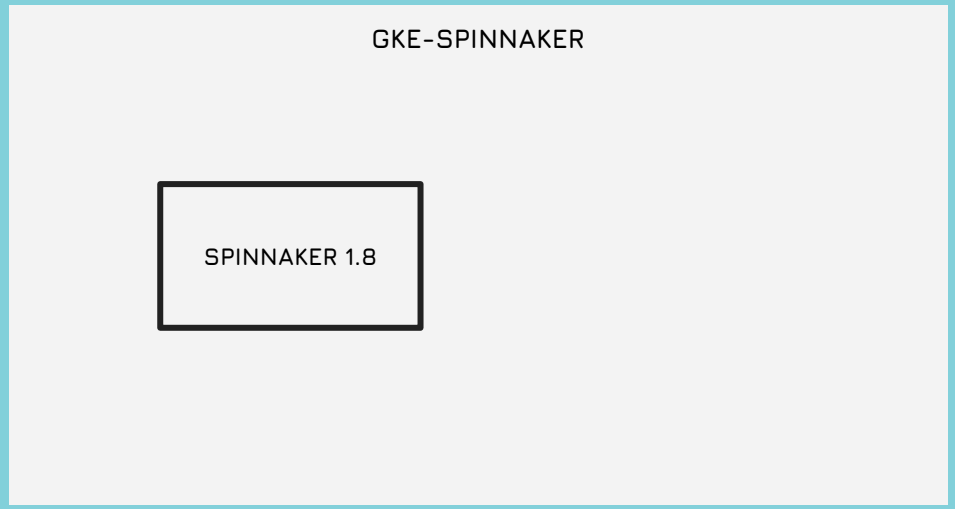
LAB 0: SETUP

GKE-SPINNAKER

GKE-CENTRAL

GKE-WEST

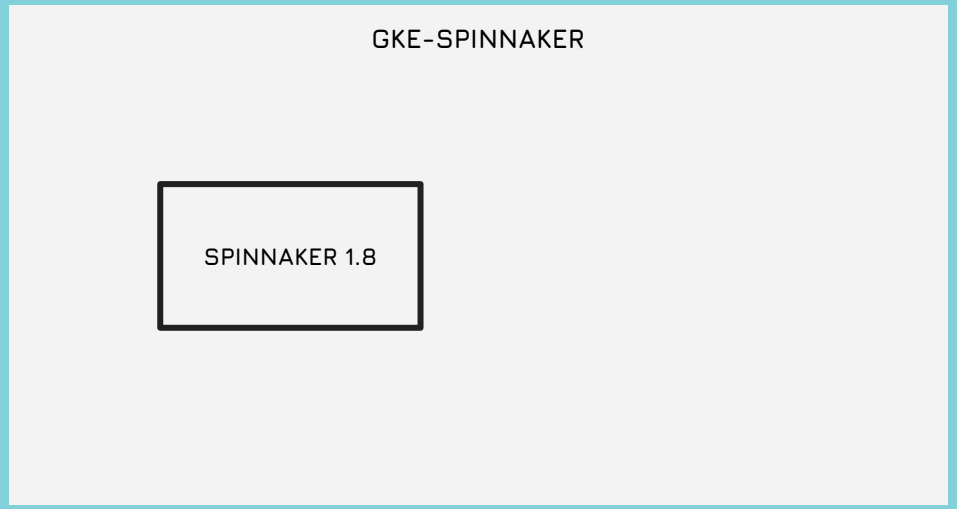
LAB 0: SETUP


GKE-SPINNAKER
SPINNAKER 1.8

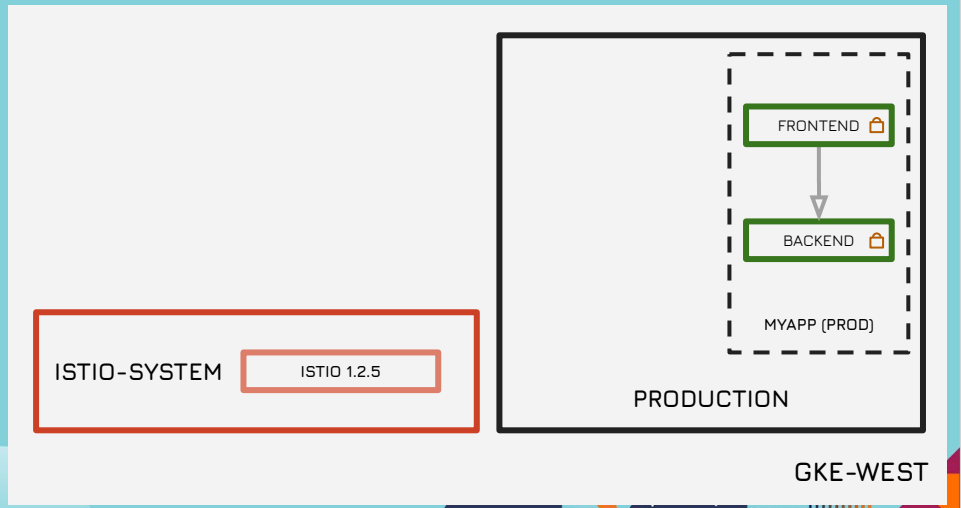

ISTIO-SYSTEM
ISTIO 1.2.5
GKE-CENTRAL


ISTIO-SYSTEM
ISTIO 1.2.5
GKE-WEST

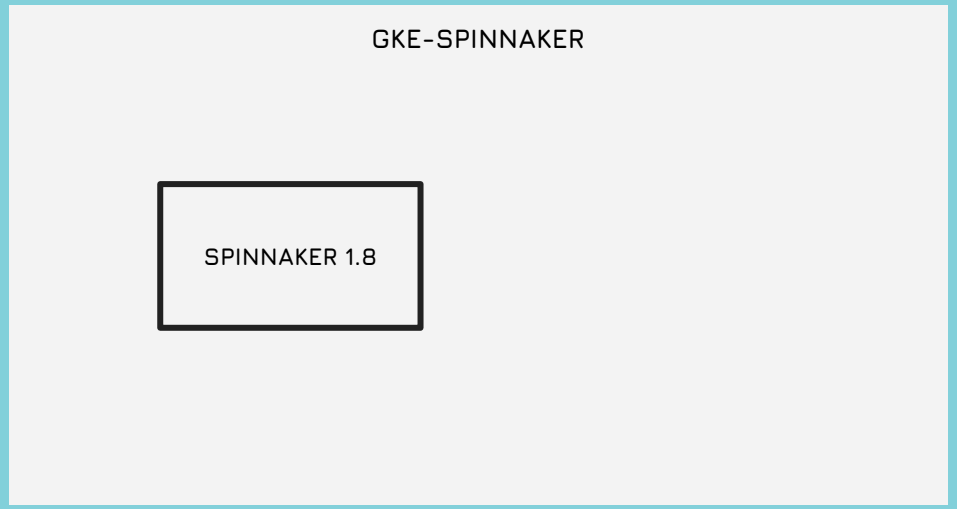
LAB 0: SETUP

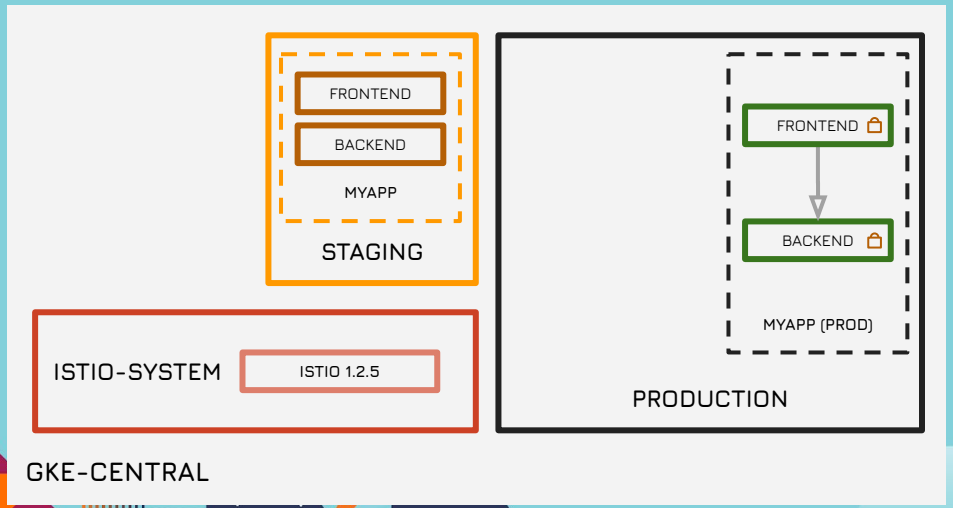

GKE-SPINNAKER
SPINNAKER 1.8

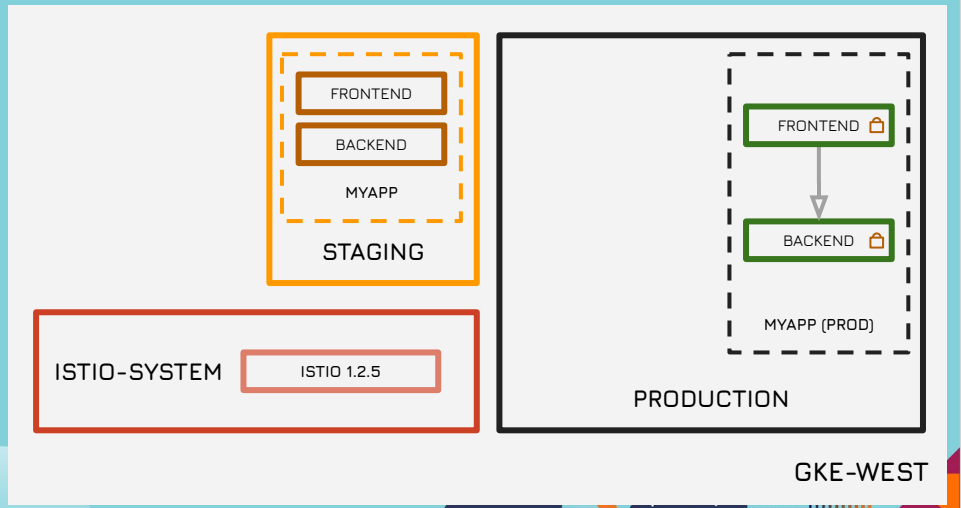

ISTIO-SYSTEM
ISTIO 1.2.5
FRONTEND
BACKEND
MYAPP (PROD)
PRODUCTION
GKE-CENTRAL


ISTIO-SYSTEM
ISTIO 1.2.5
FRONTEND
BACKEND
MYAPP (PROD)
PRODUCTION
GKE-WEST

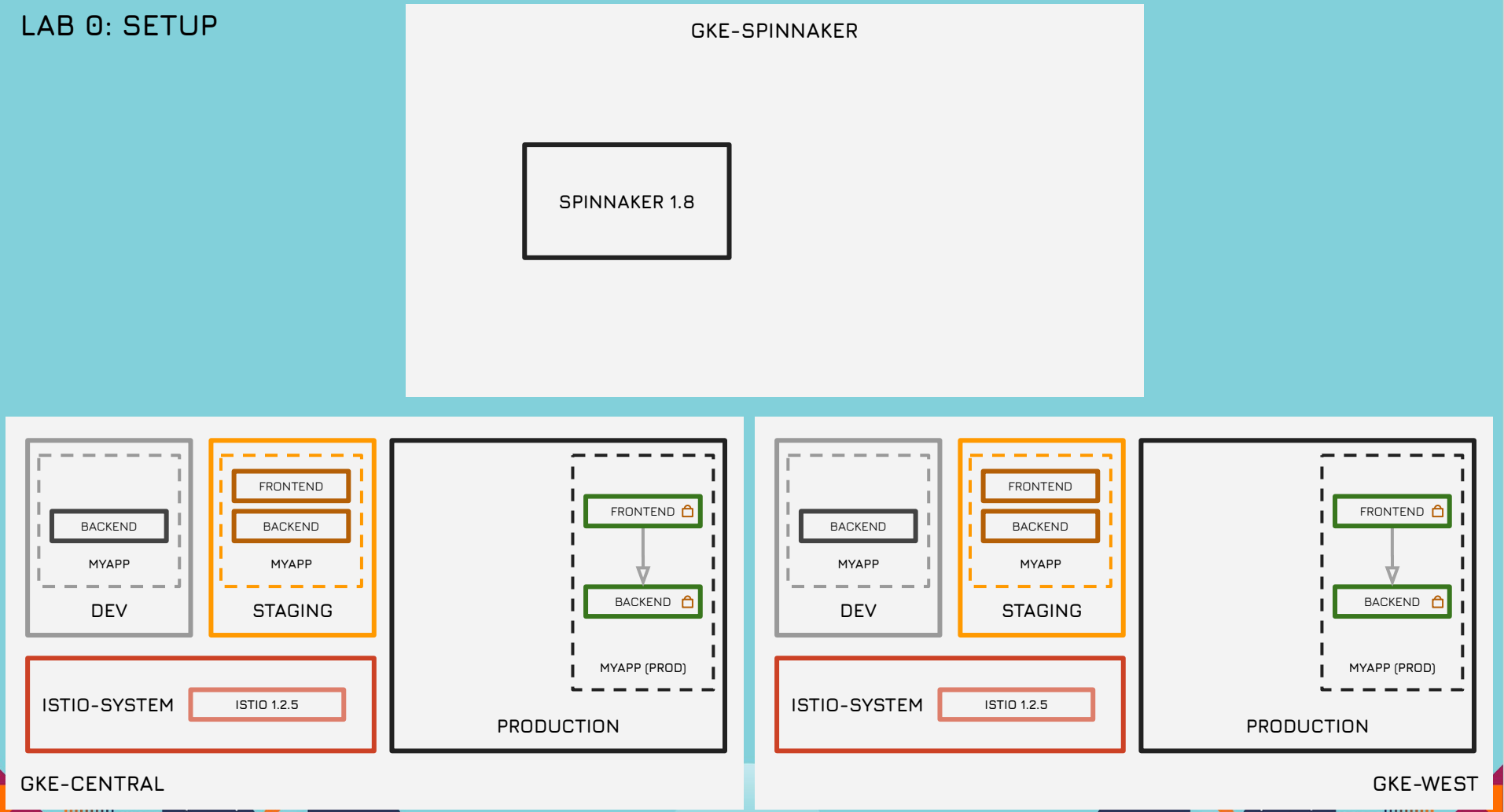
LAB 0: SETUP


GKE-SPINNAKER
SPINNAKER 1.8

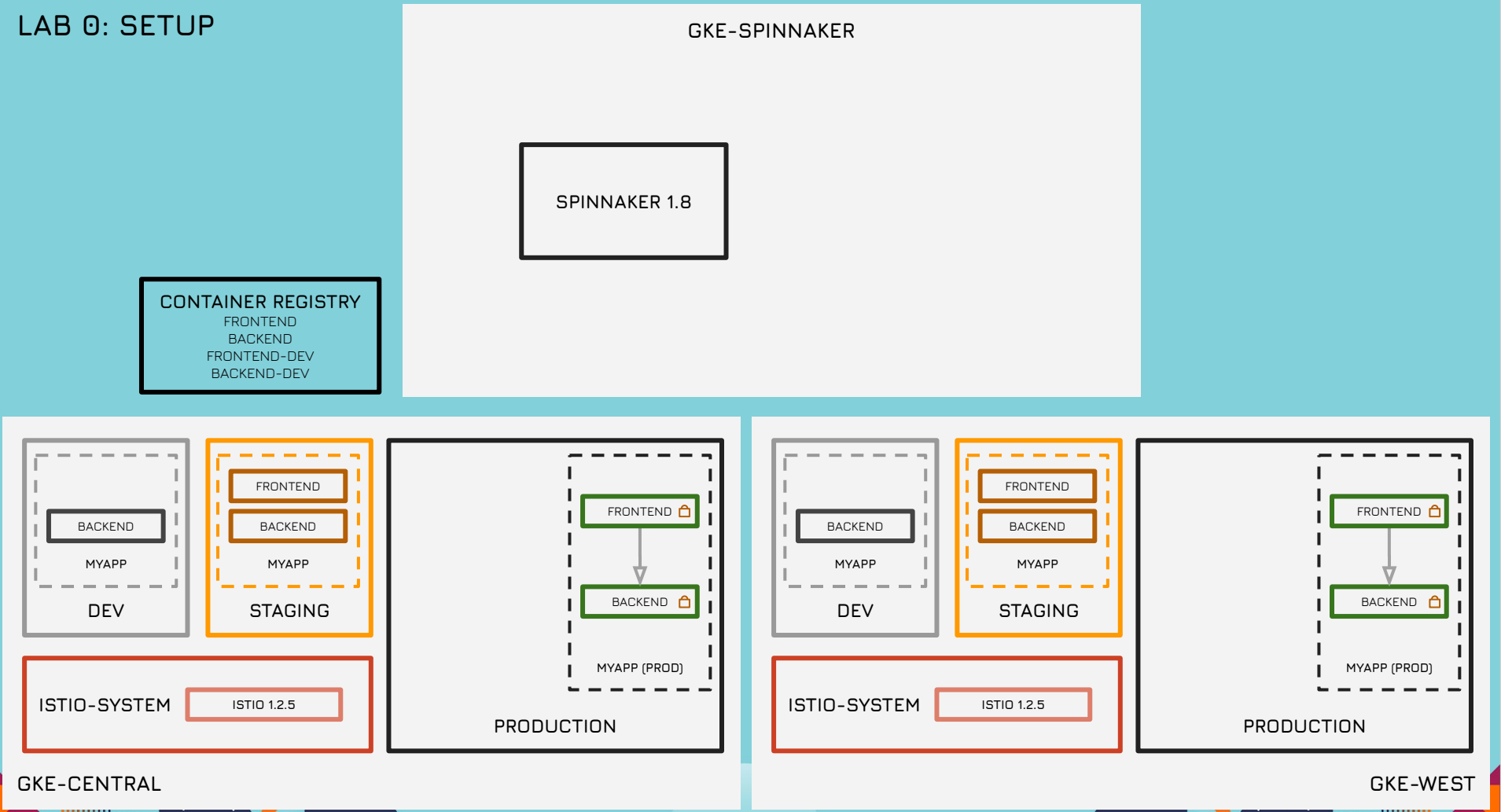

FRONTEND
BACKEND
MYAPP
STAGING
ISTIO-SYSTEM
ISTIO 1.2.5
FRONTEND
BACKEND
MYAPP (PROD)
PRODUCTION
GKE-CENTRAL


FRONTEND
BACKEND
MYAPP
STAGING
ISTIO-SYSTEM
ISTIO 1.2.5
FRONTEND
BACKEND
MYAPP (PROD)
PRODUCTION
GKE-WEST

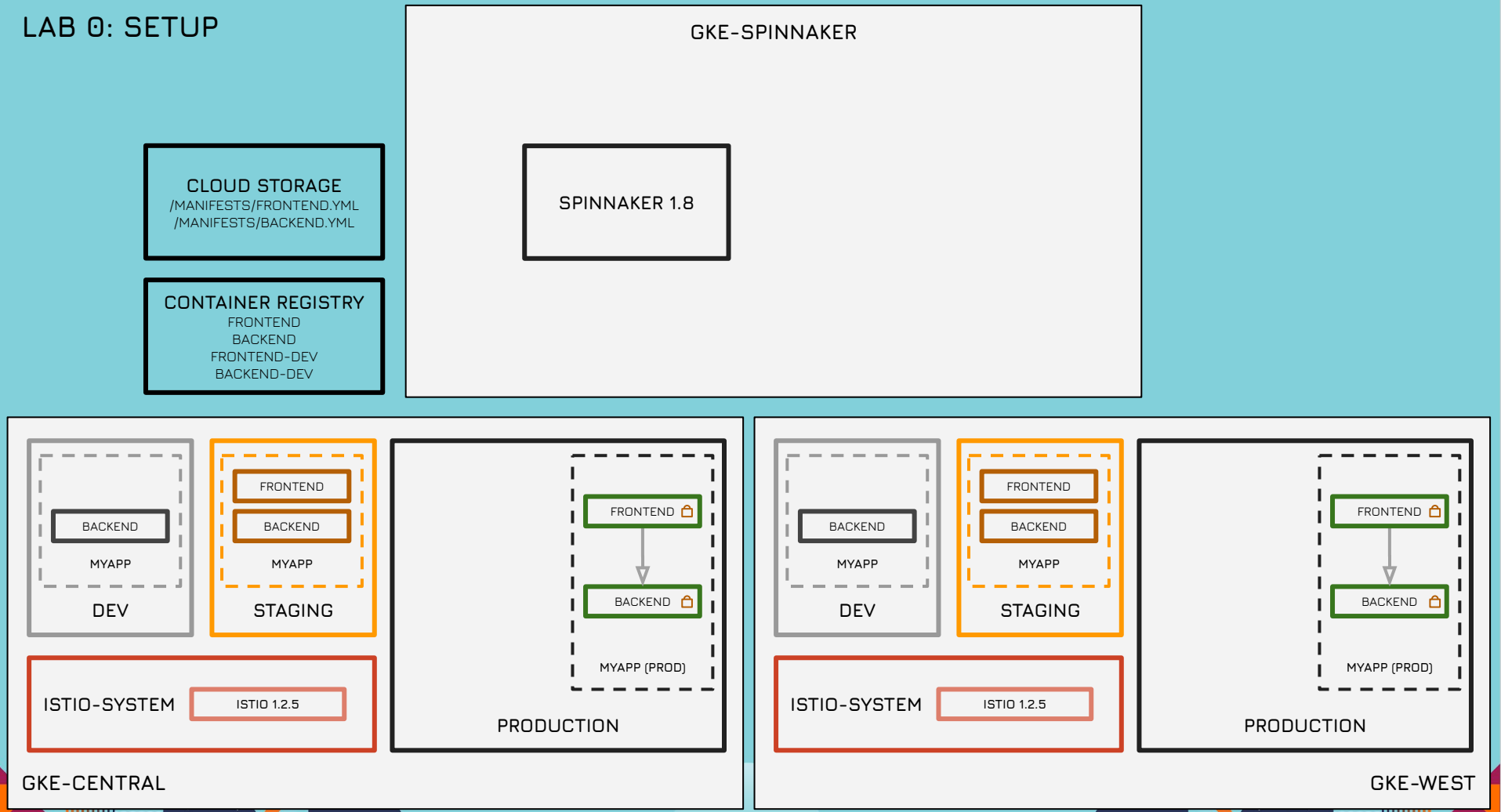
LAB 0: SETUP


GKE-SPINNAKER
SPINNAKER 1.8
BACKEND
MYAPP
DEV
FRONTEND
BACKEND
MYAPP
STAGING
ISTIO-SYSTEM
ISTIO 1.2.5
FRONTEND
BACKEND
MYAPP (PROD)
PRODUCTION
GKE-CENTRAL
BACKEND
MYAPP
DEV
FRONTEND
BACKEND
MYAPP
STAGING
ISTIO-SYSTEM
ISTIO 1.2.5
FRONTEND
BACKEND
MYAPP (PROD)
PRODUCTION
GKE-WEST

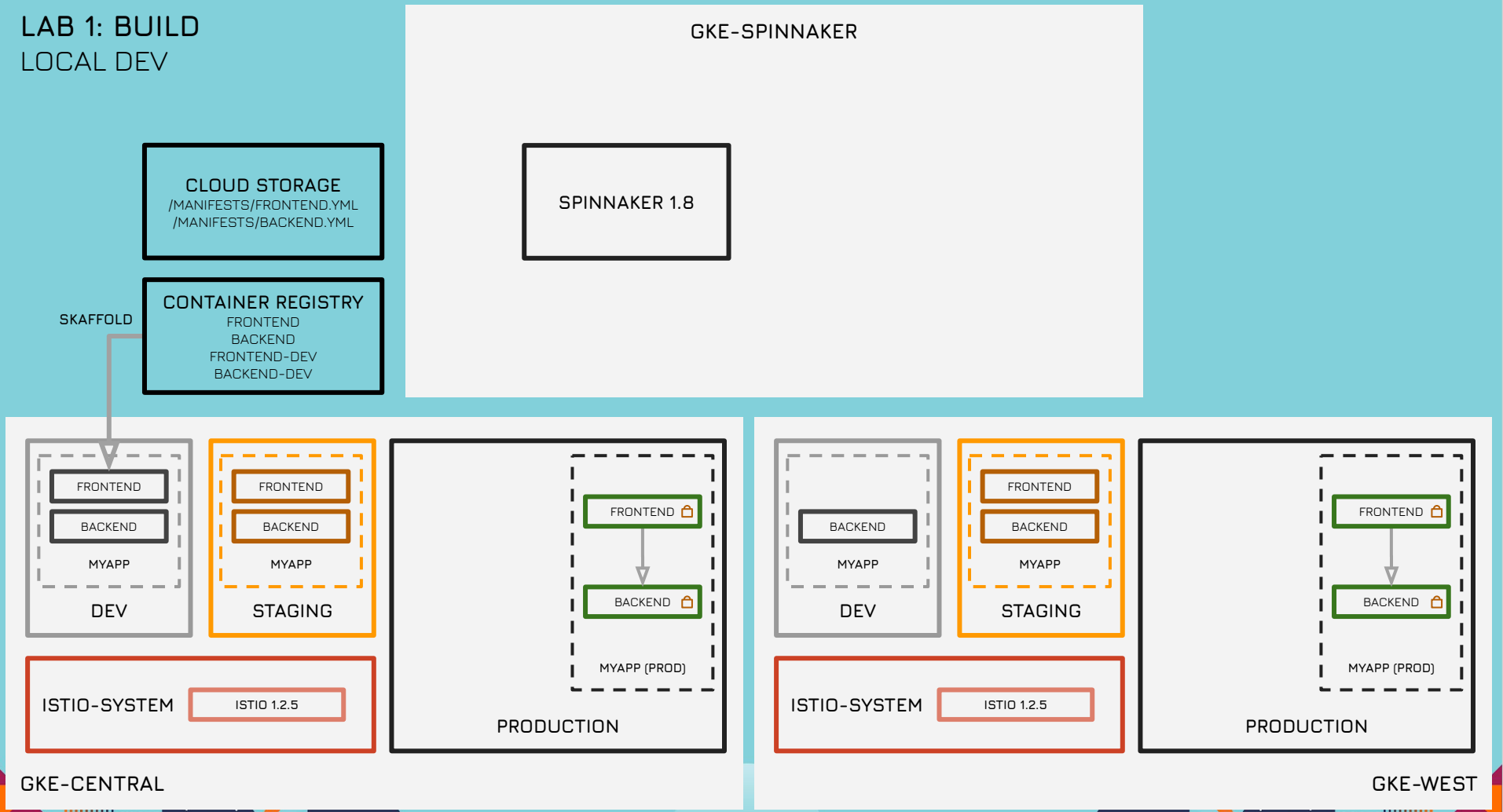
LAB 0: SETUP


GKE-SPINNAKER
SPINNAKER 1.8
CONTAINER REGISTRY
FRONTEND
BACKEND
FRONTEND-DEV
BACKEND-DEV
BACKEND
MYAPP
DEV
FRONTEND
BACKEND
MYAPP
STAGING
ISTIO-SYSTEM
ISTIO 1.2.5
FRONTEND
BACKEND
MYAPP (PROD)
PRODUCTION
GKE-CENTRAL
BACKEND
MYAPP
DEV
FRONTEND
BACKEND
MYAPP
STAGING
ISTIO-SYSTEM
ISTIO 1.2.5
FRONTEND
BACKEND
MYAPP (PROD)
PRODUCTION
GKE-WEST

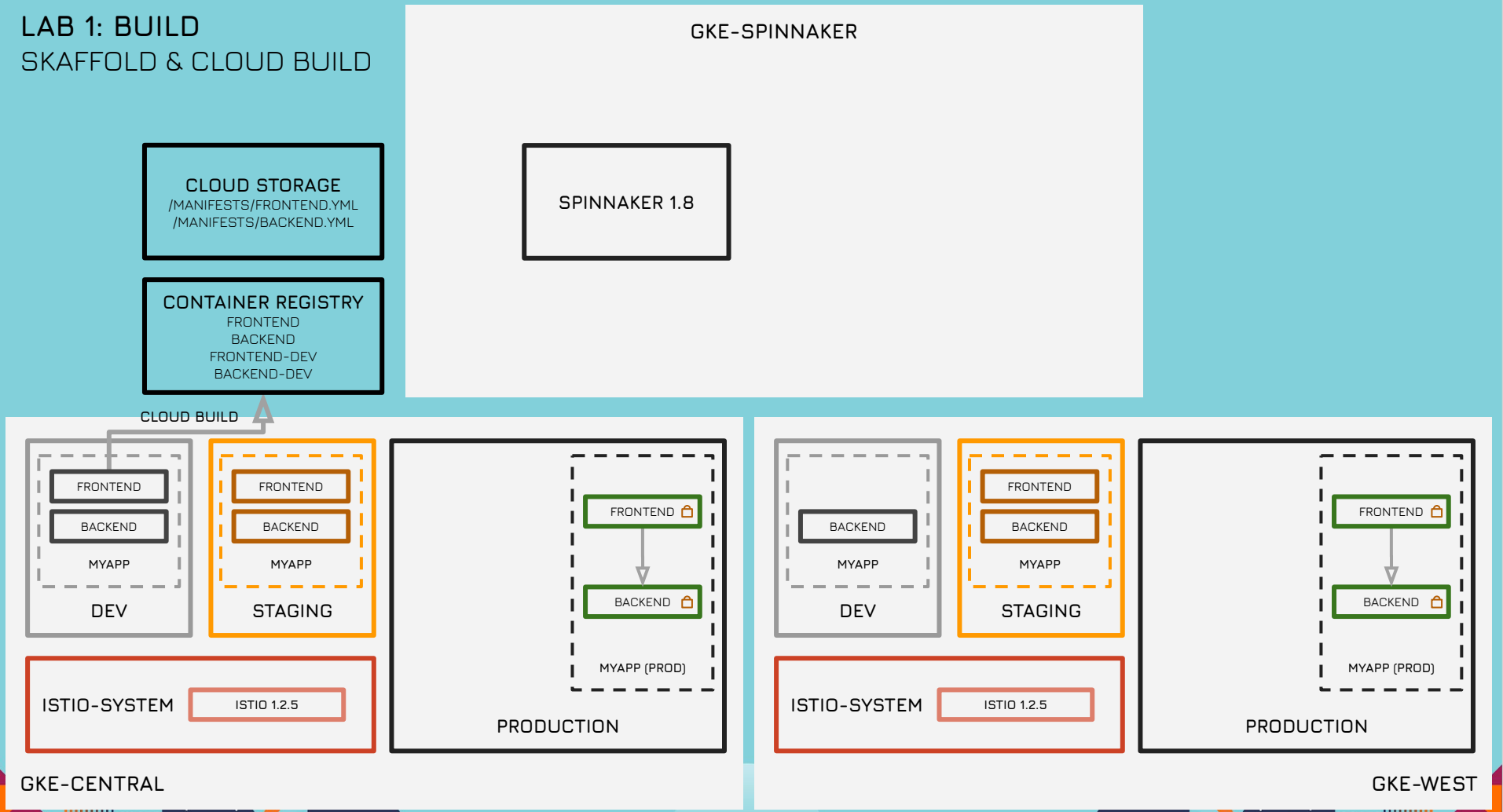
LAB 0: SETUP


GKE-SPINNAKER
SPINNAKER 1.8
CLOUD STORAGE
/MANIFESTS/FRONTEND.YML
/MANIFESTS/BACKEND.YML
CONTAINER REGISTRY
FRONTEND
BACKEND
FRONTEND-DEV
BACKEND-DEV
BACKEND
MYAPP
DEV
FRONTEND
BACKEND
MYAPP
STAGING
ISTIO-SYSTEM
ISTIO 1.2.5
FRONTEND
BACKEND
MYAPP (PROD)
PRODUCTION
GKE-CENTRAL
BACKEND
MYAPP
DEV
FRONTEND
BACKEND
MYAPP
STAGING
ISTIO-SYSTEM
ISTIO 1.2.5
FRONTEND
BACKEND
MYAPP (PROD)
PRODUCTION
GKE-WEST

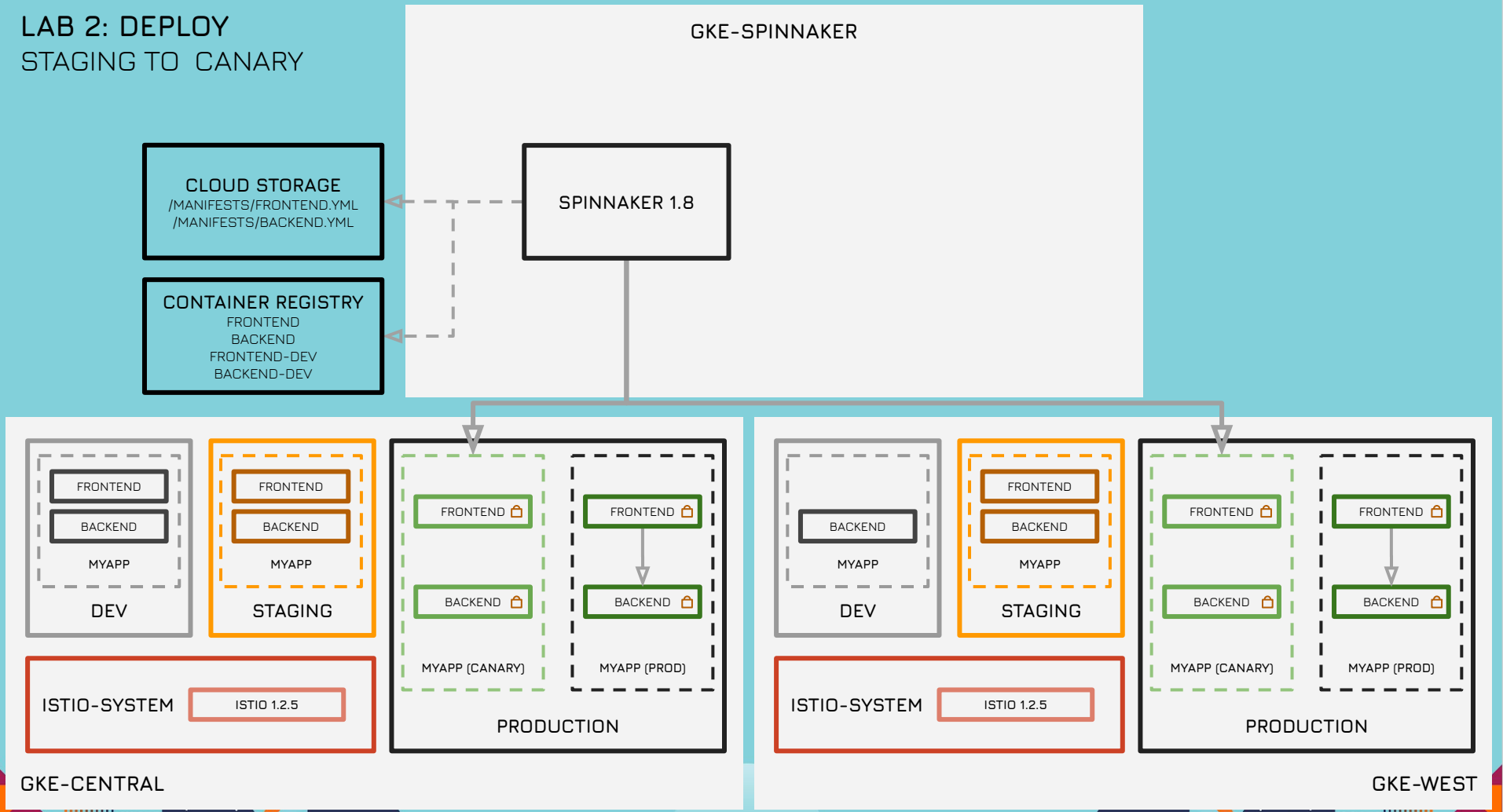
LAB 1: BUILD
LOCAL DEV



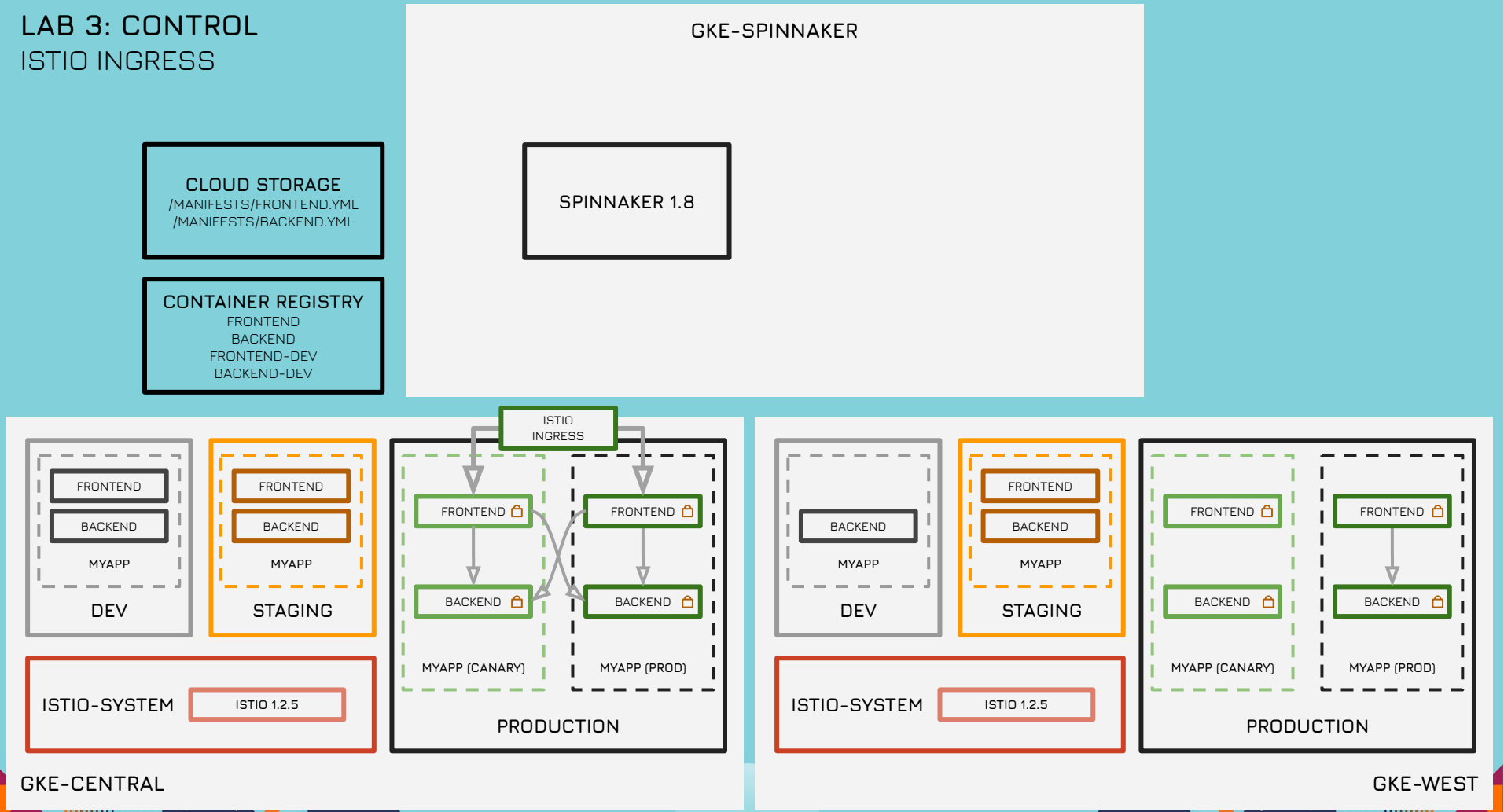
LAB 1: BUILD
SKAFFOLD & CLOUD BUILD



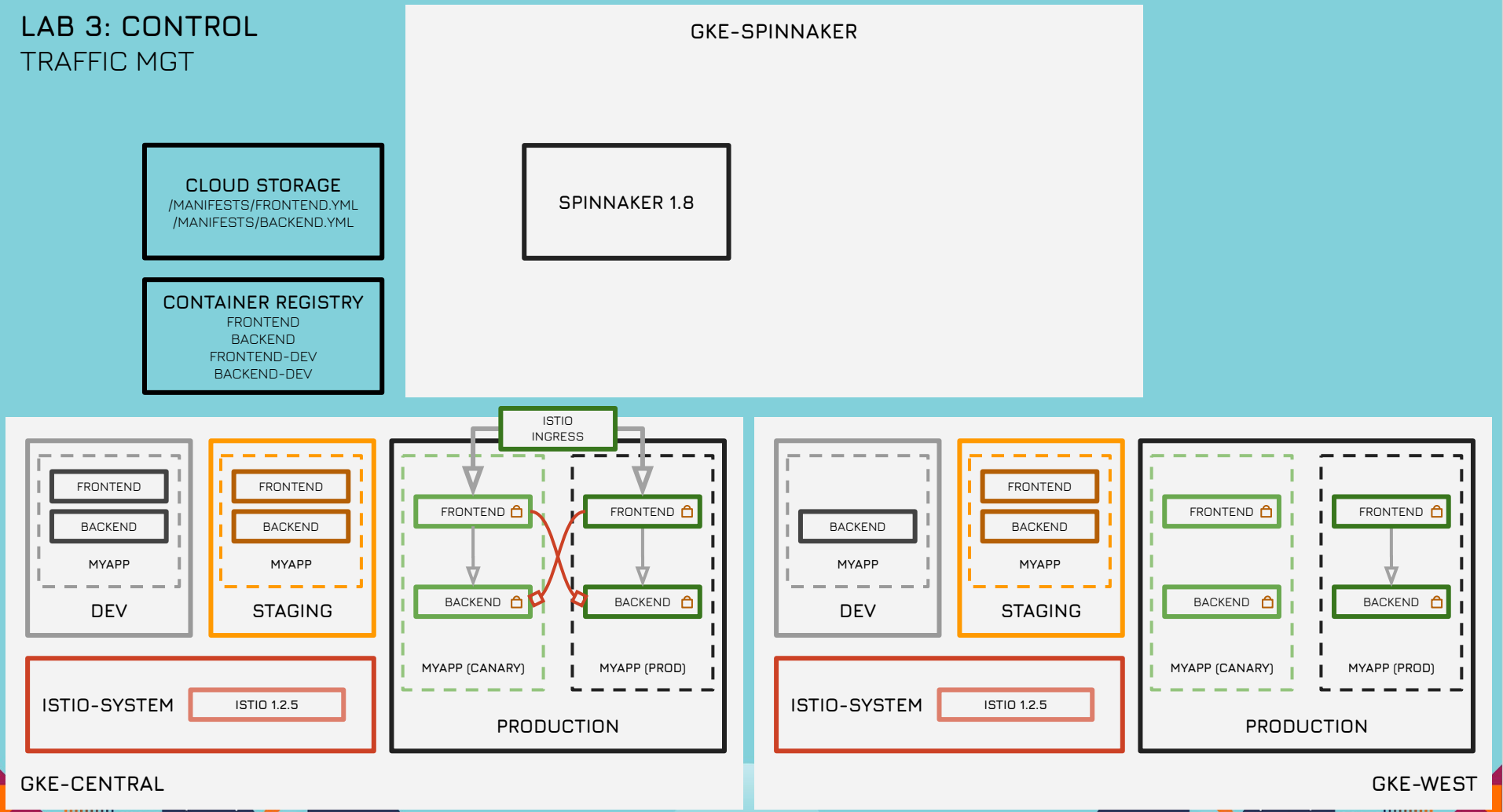
LAB 2: DEPLOY
STAGING TO CANARY



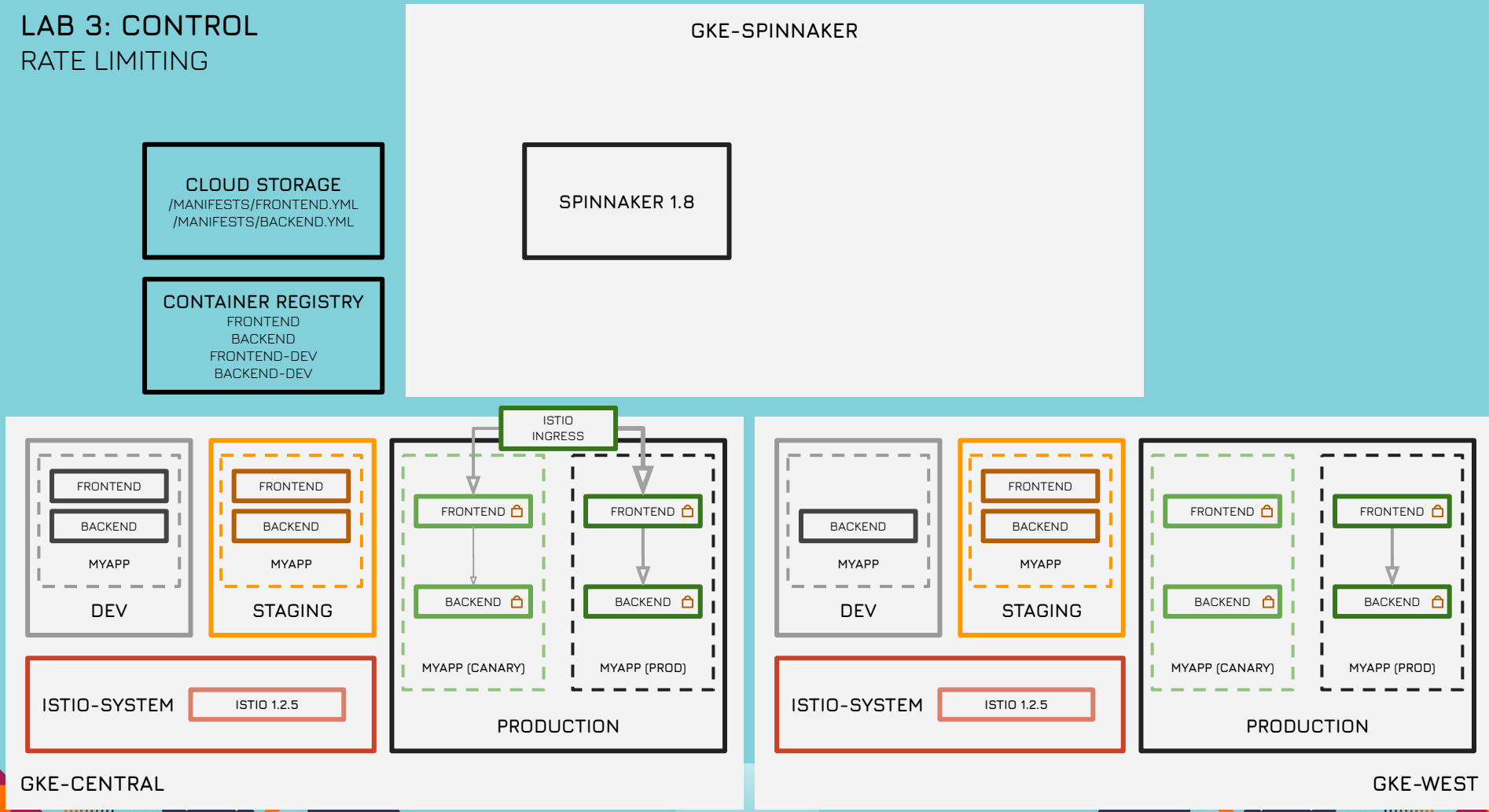
LAB 3: CONTROL
ISTIO INGRESS



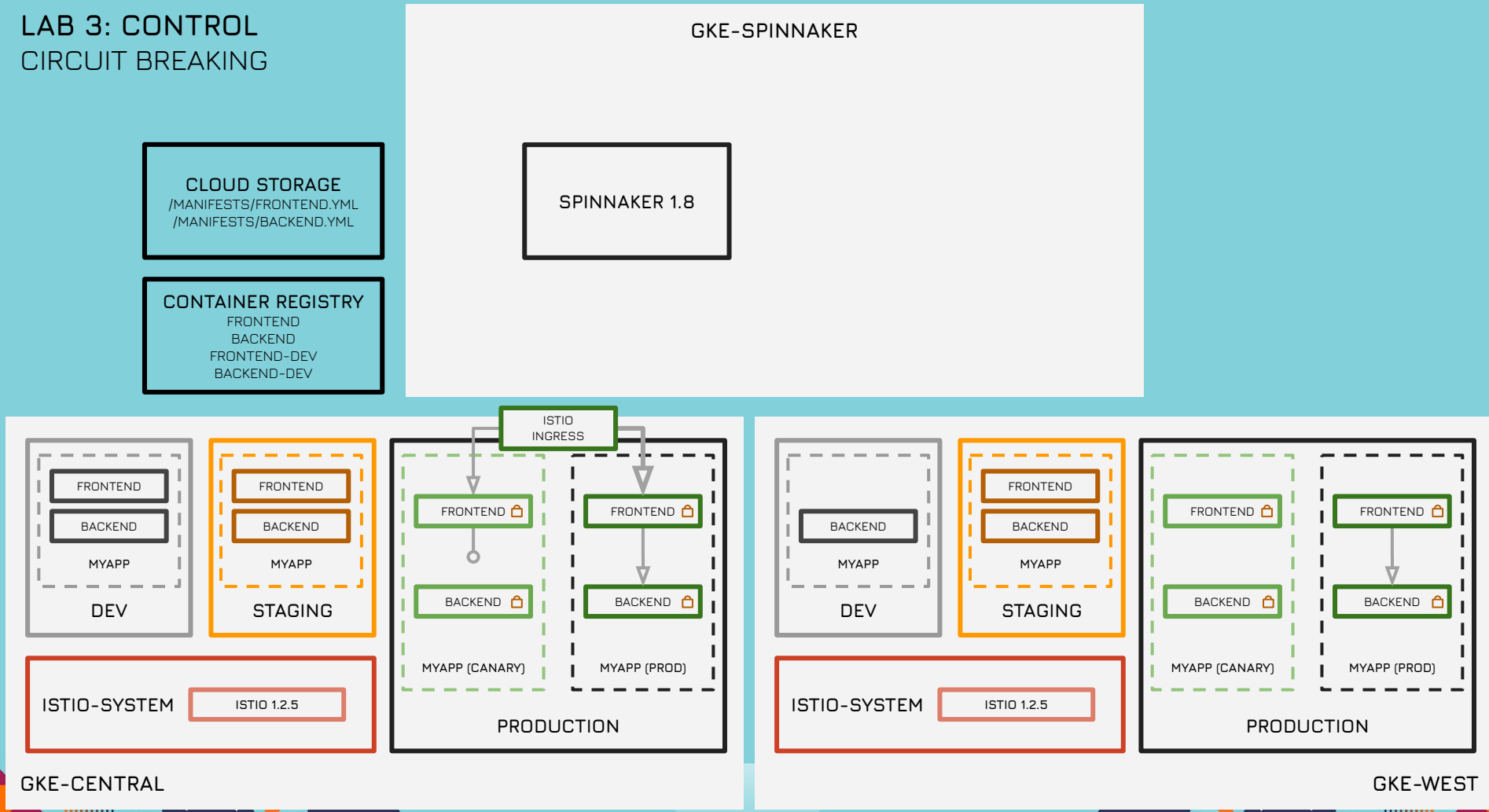
LAB 3: CONTROL
TRAFFIC MGT



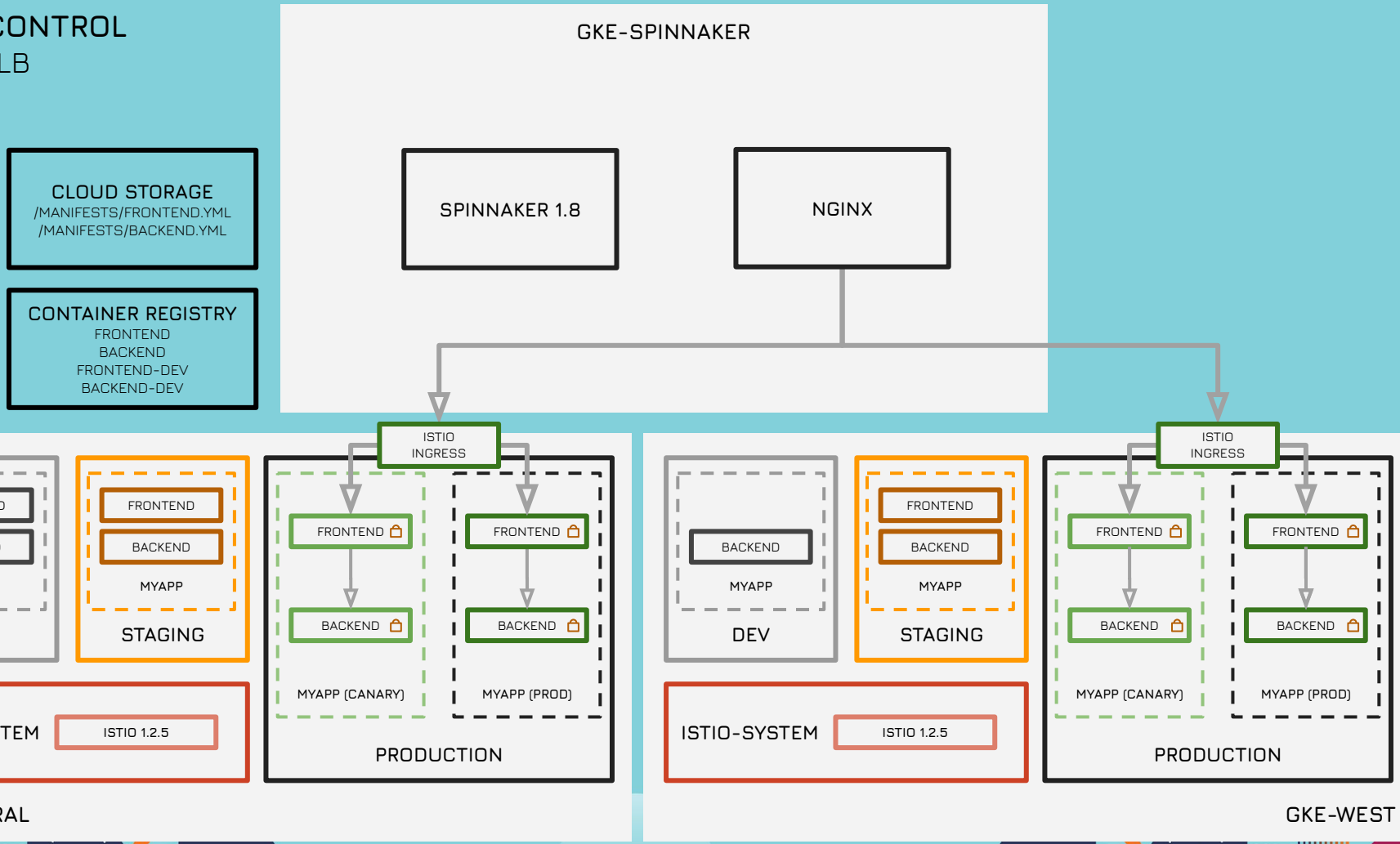
LAB 3: CONTROL
RATE LIMITING



LAB 3: CONTROL
CIRCUIT BREAKING



LAB 3: CONTROL
GLOBAL LB



LAB 4: MONITOR

METRICS

PROMETHEUS + GRAFANA

DISTRIBUTED TRACING

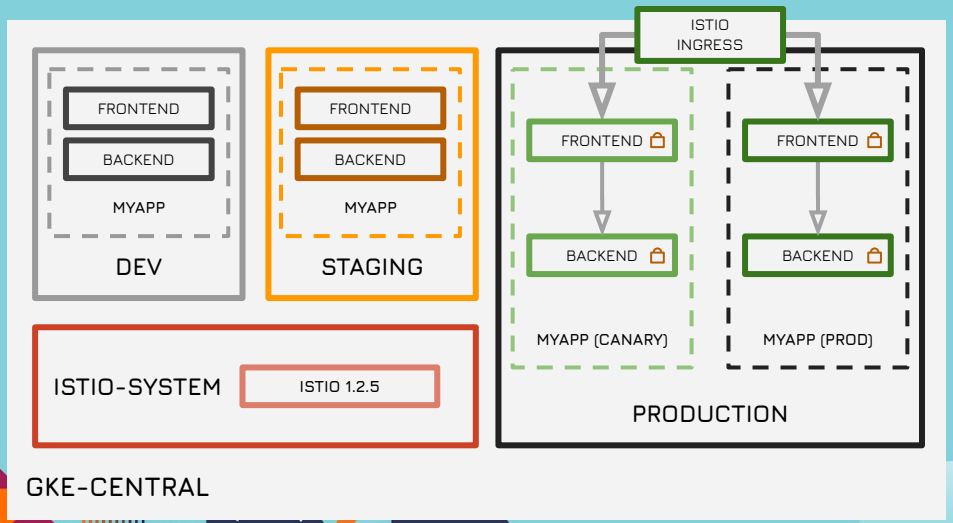
JAEGER

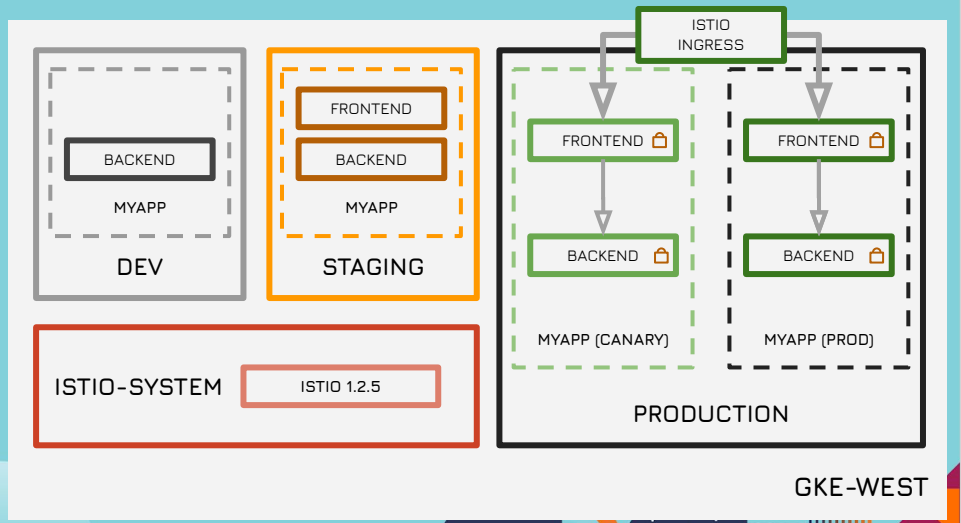
MESH VISUALIZATION

KIALI

LOGGING / MONITORING

STACKDRIVER

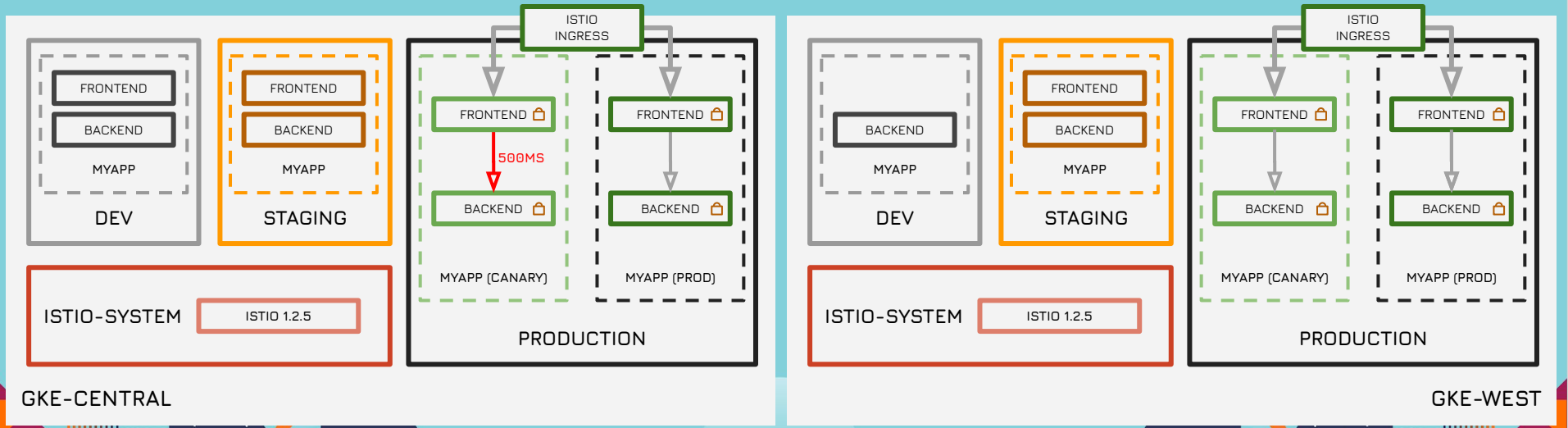




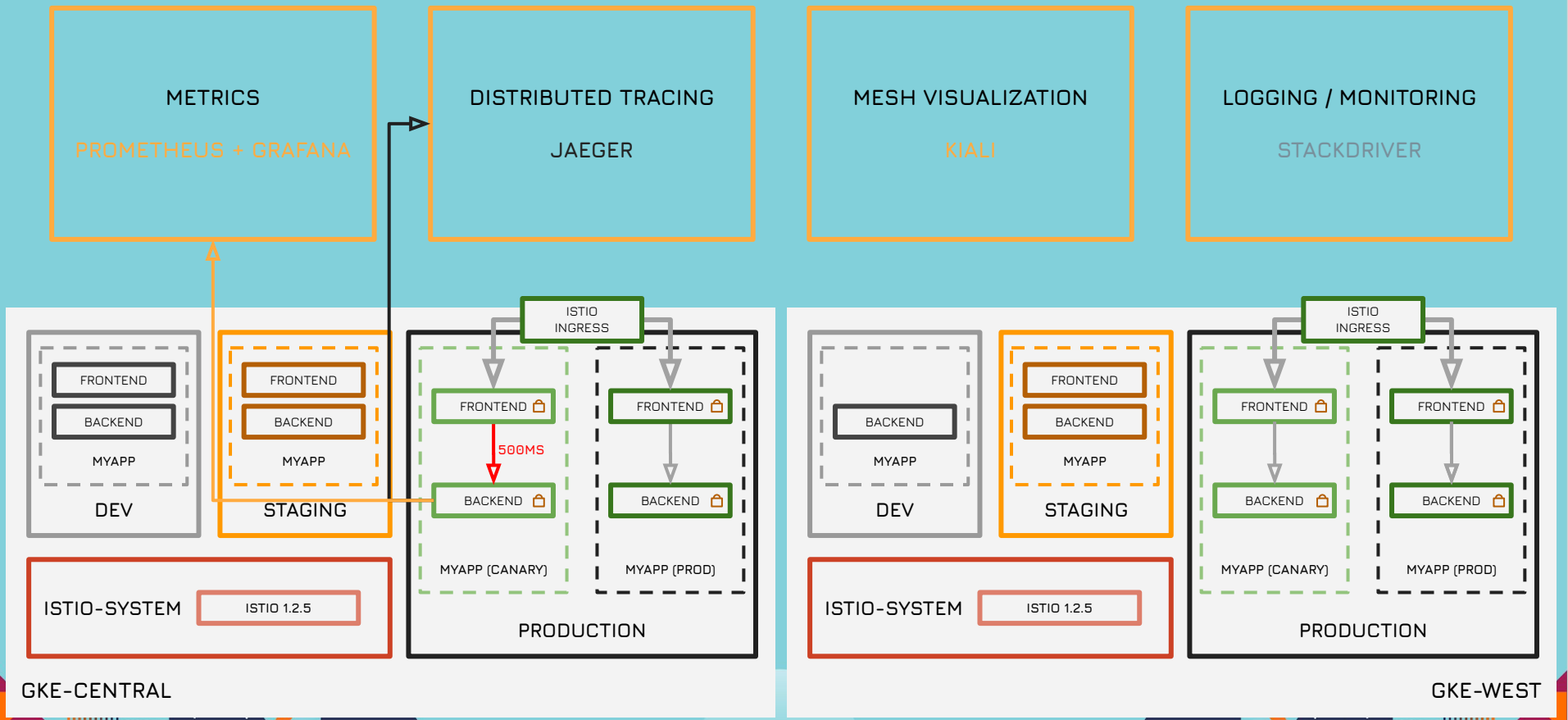
LAB 4: MONITOR

METRICS, TRACING, VISUALIZATION, LOGGING

METRICS	DISTRIBUTED TRACING	MESH VISUALIZATION	LOGGING / MONITORING
PROMETHEUS + GRAFANA	JAEGER	KIALI	STACKDRIVER

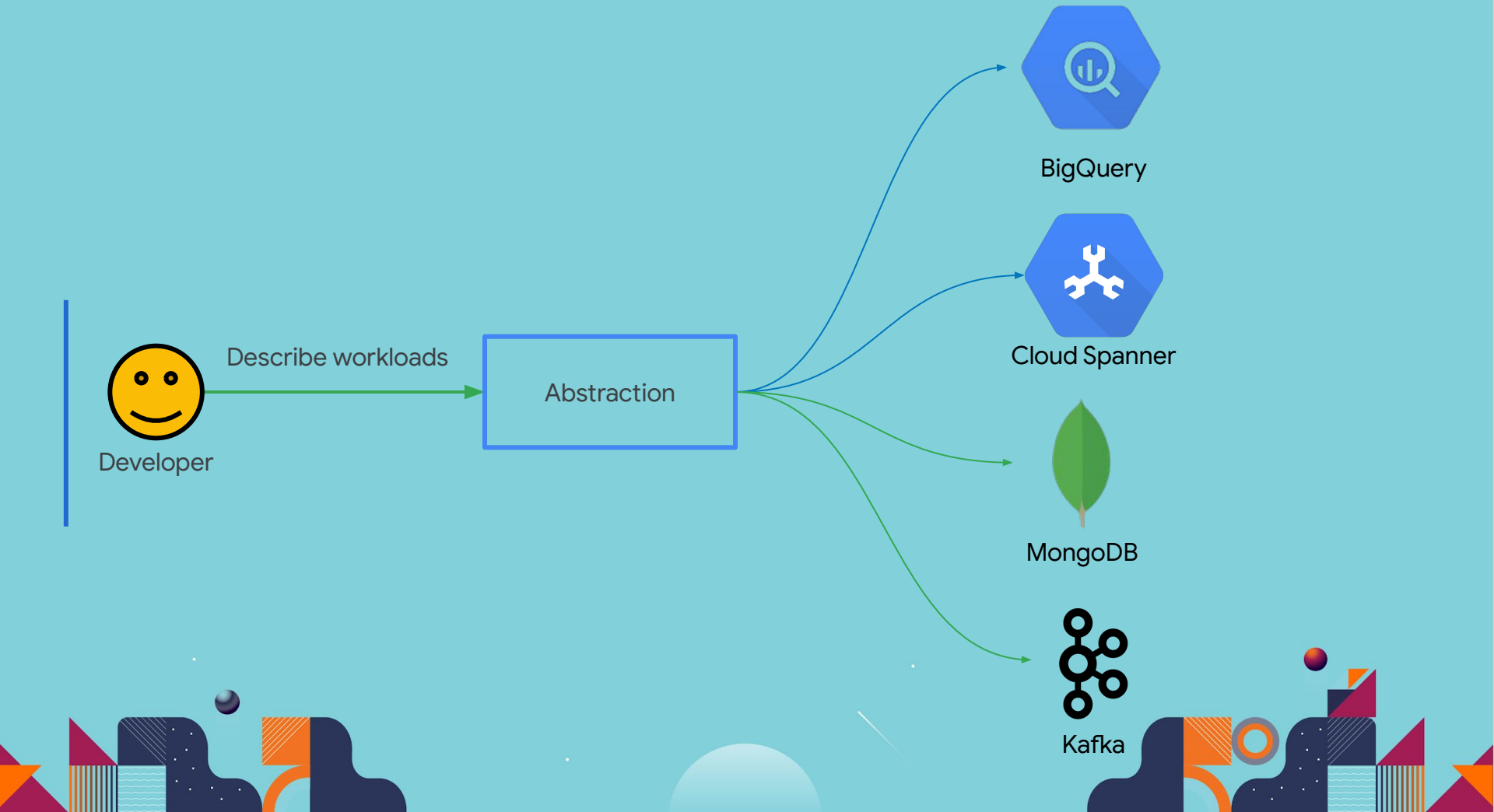


LAB 4: MONITOR
TROUBLESHOOTING



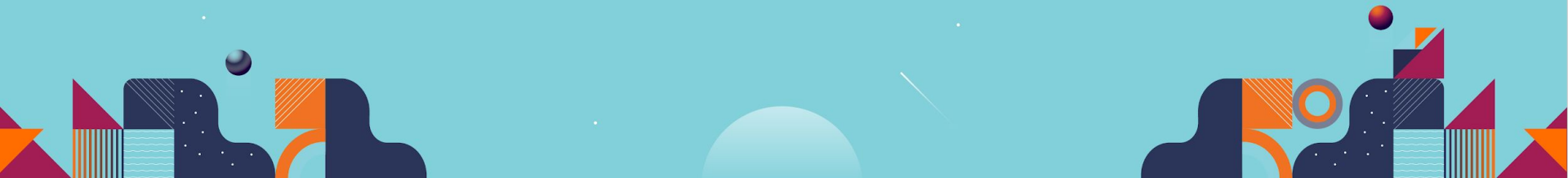
02

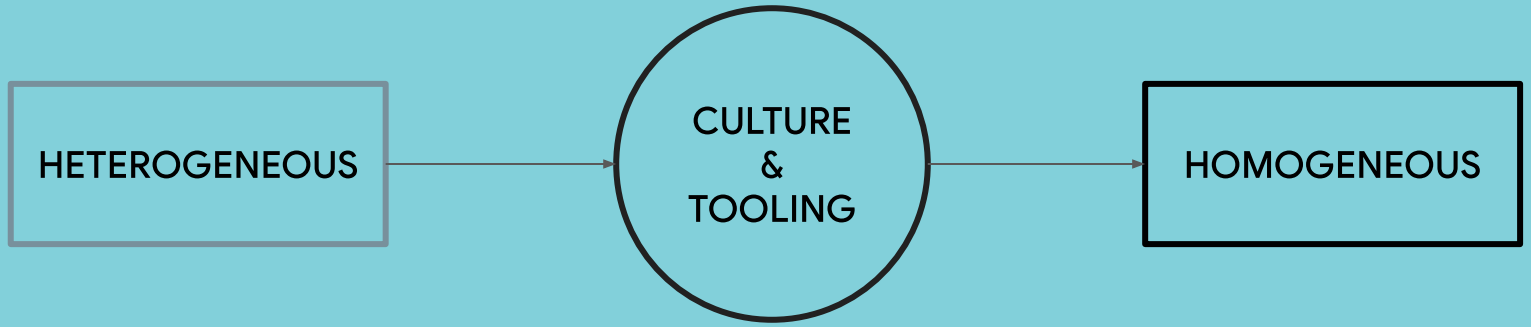
Distributed Applications in Multi-cloud

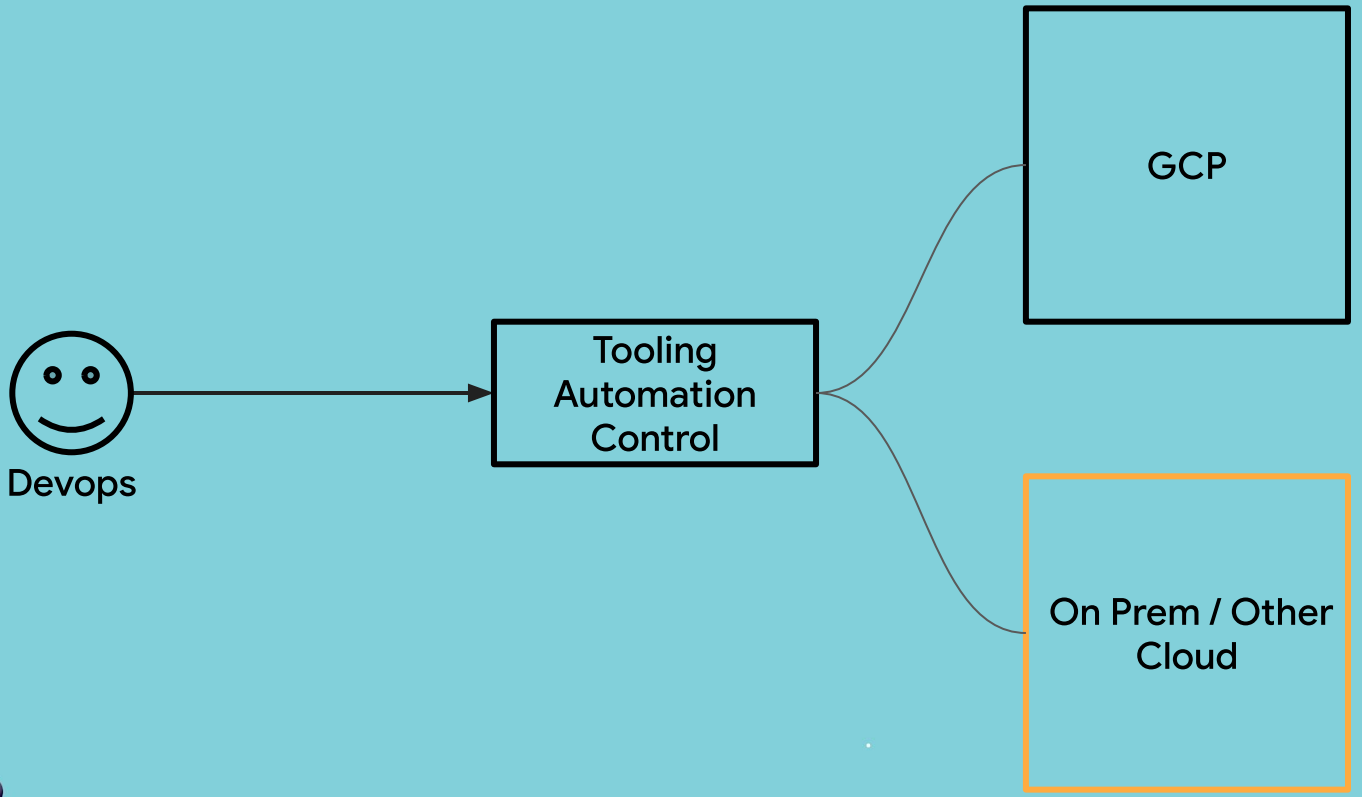

Describe workloads
Abstraction
Developer
BigQuery
Cloud Spanner
MongoDB
Kafka

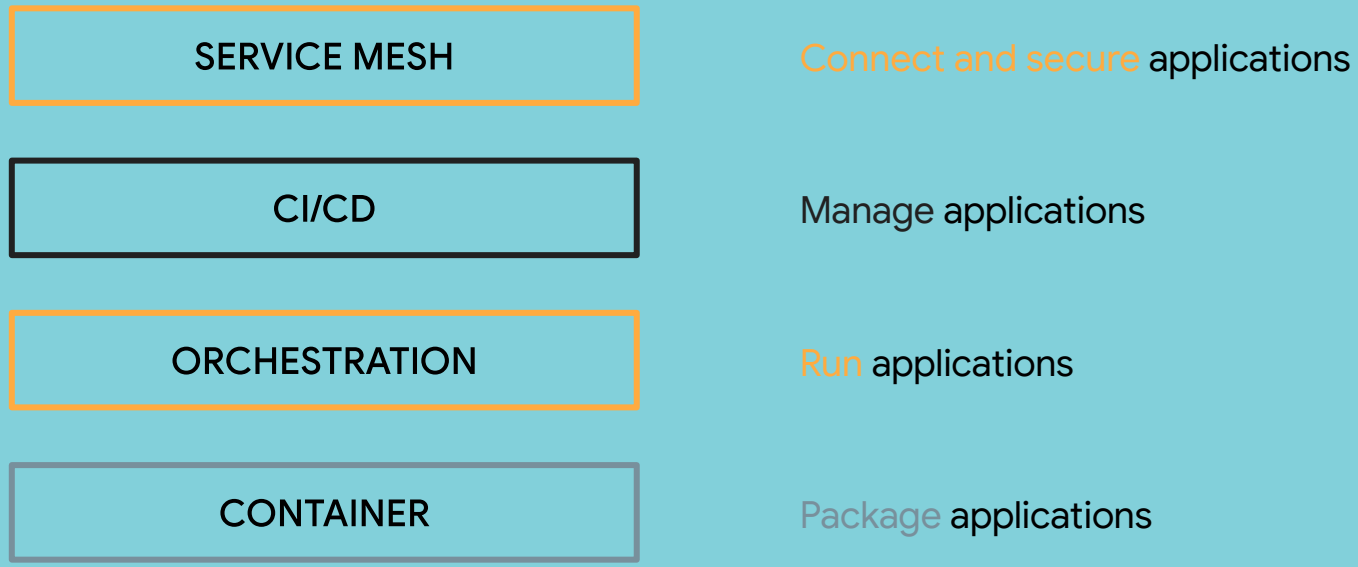
- Zero vendor lock-in
- Cost & Control
- Resiliency & Availability
- Services & Innovation
- Technology / Business Partnerships

Opinionated Systems vs Freedom of Choice



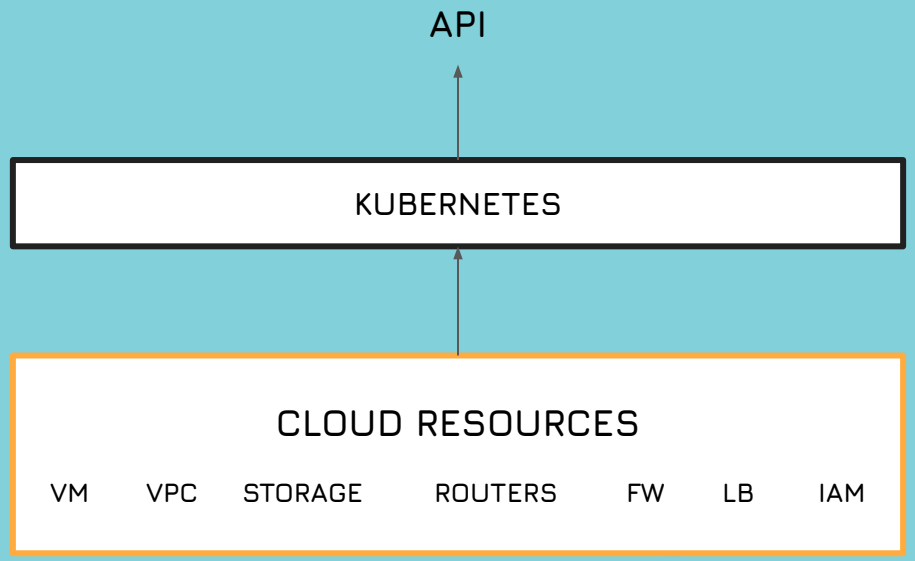

HETEROGENEOUS
CULTURE
&
TOOLING
HOMOGENEOUS

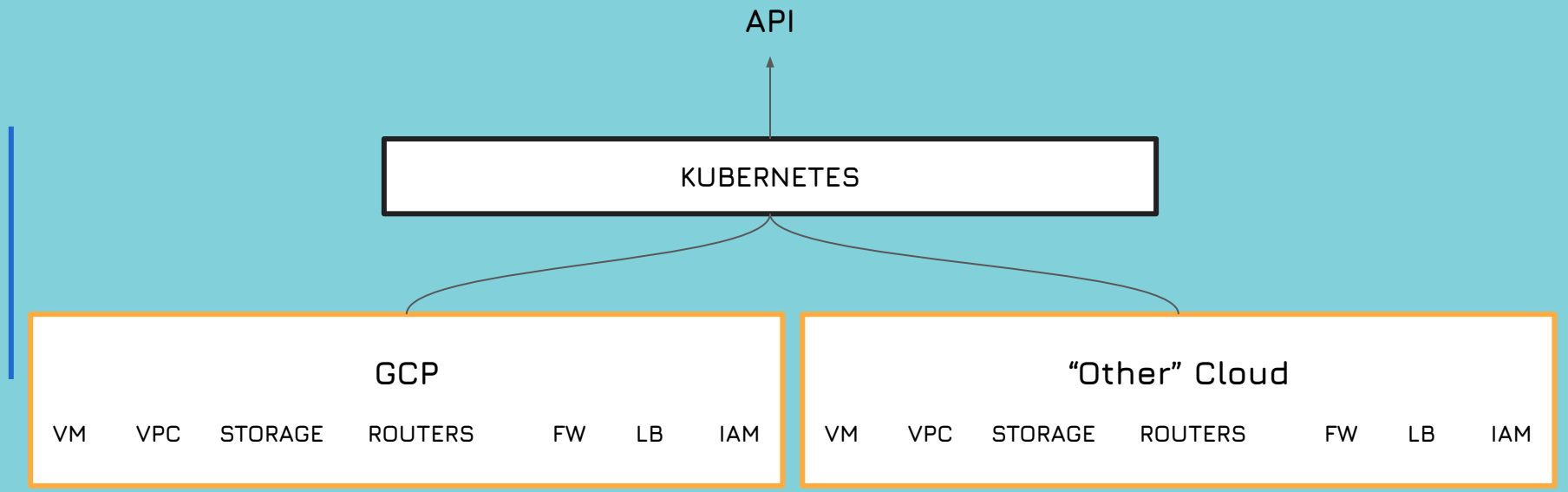

GCP
Tooling
Automation
Control
Devops
On Prem / Other
Cloud


SERVICE MESH
Connect and secure applications
CI/CD
Manage applications
ORCHESTRATION
Run applications
CONTAINER
Package applications

ISTIO	Connect and secure applications
SPINNAKER	Manage applications
KUBERNETES	Run applications
DOCKER	Package applications

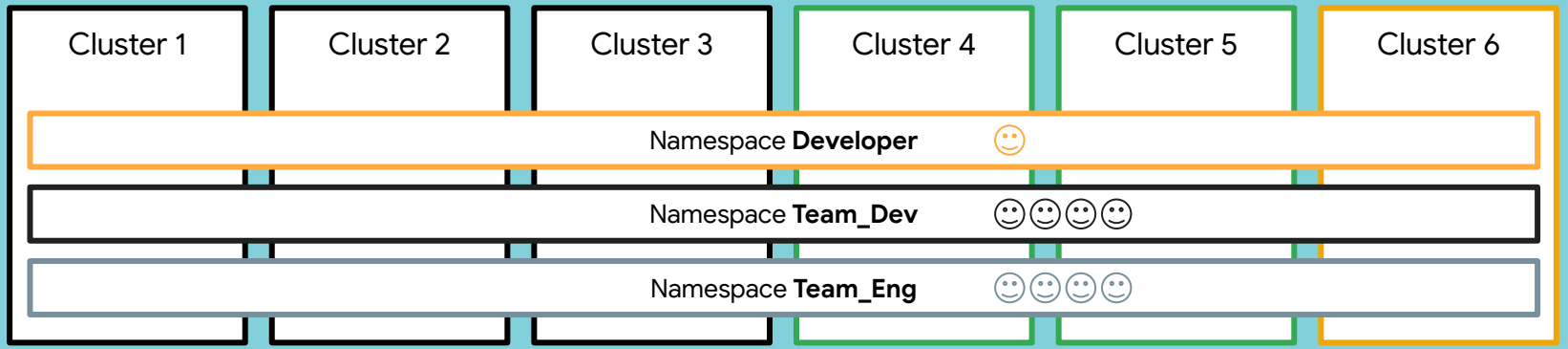

Kubernetes is a **declarative** way to **describe** your applications

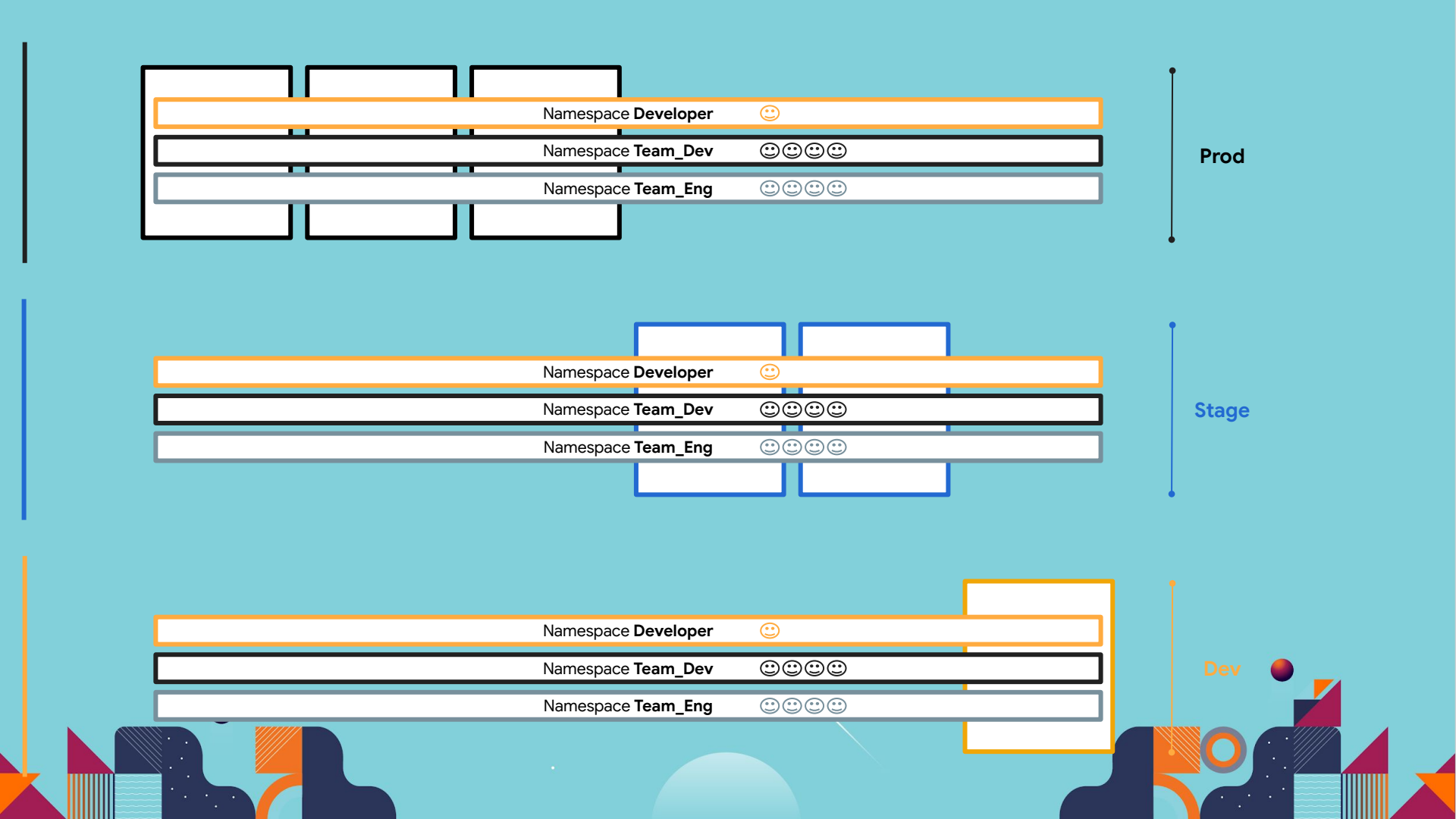



API
KUBERNETES
GCP
VM VPC STORAGE ROUTERS FW LB IAM
“Other” Cloud
VM VPC STORAGE ROUTERS FW LB IAM

Cluster Architecture & Management

Cluster Consistency



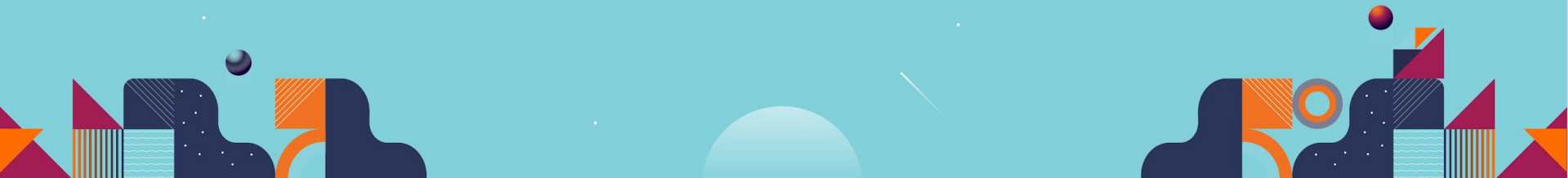

Namespace Developer
Namespace Team_Dev
Namespace Team_Eng
Prod
Namespace Developer
Namespace Team_Dev
Namespace Team_Eng
Stage
Namespace Developer
Namespace Team_Dev
Namespace Team_Eng
Dev

Cluster Management

Application Management

Service Discovery

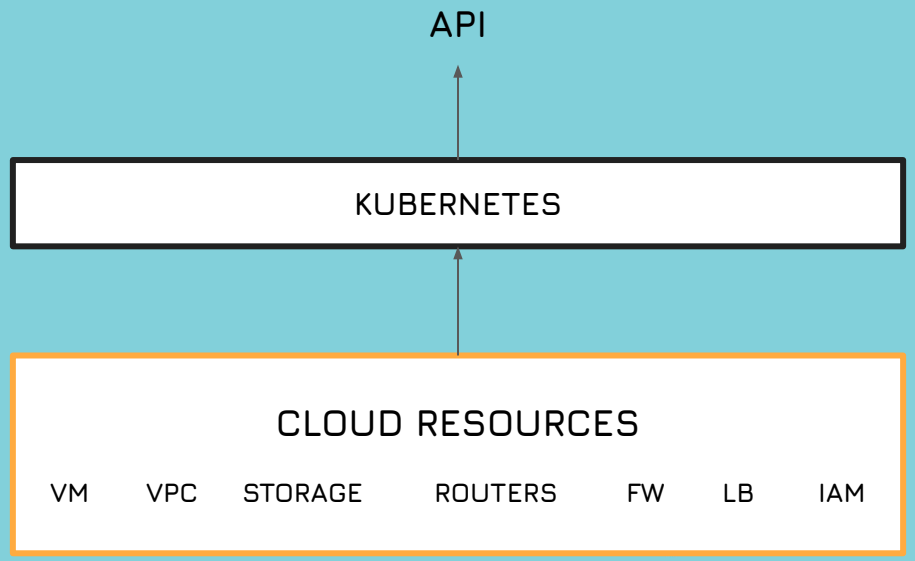

Federation vs Cluster Independence

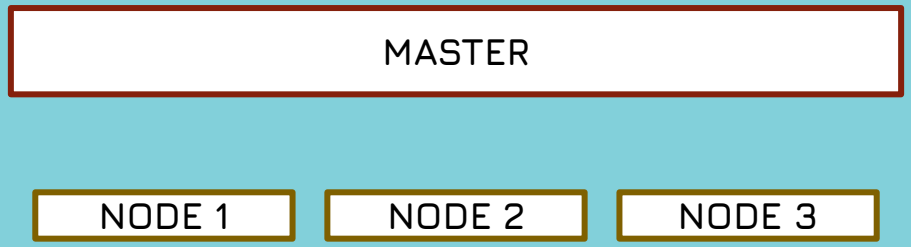


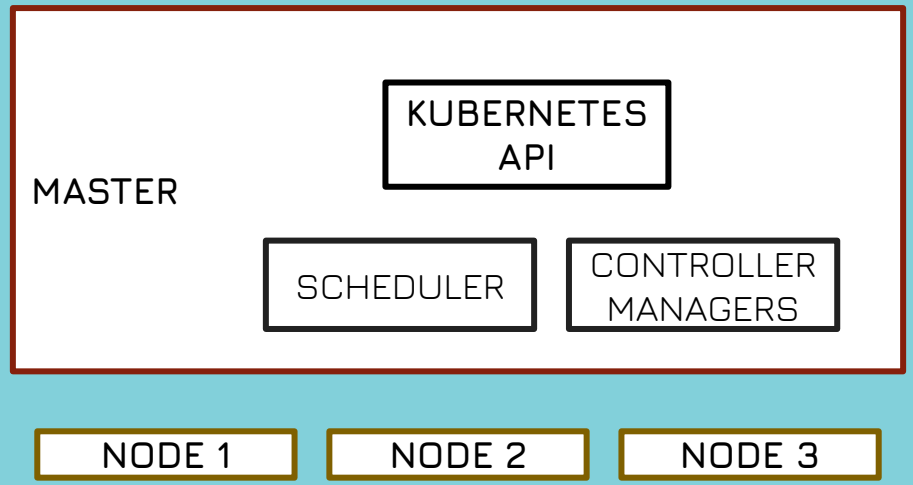
03

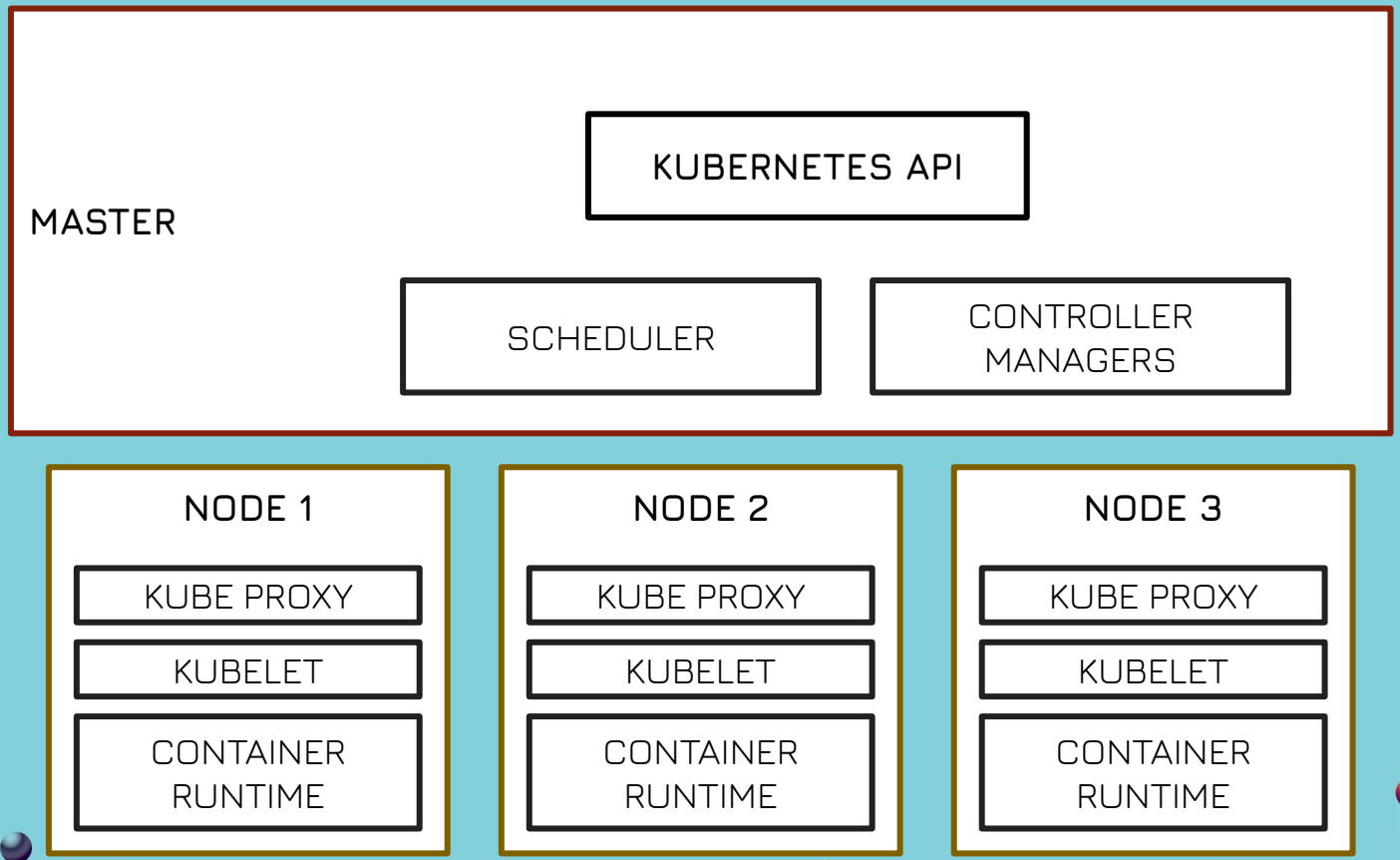
Kubernetes Primer

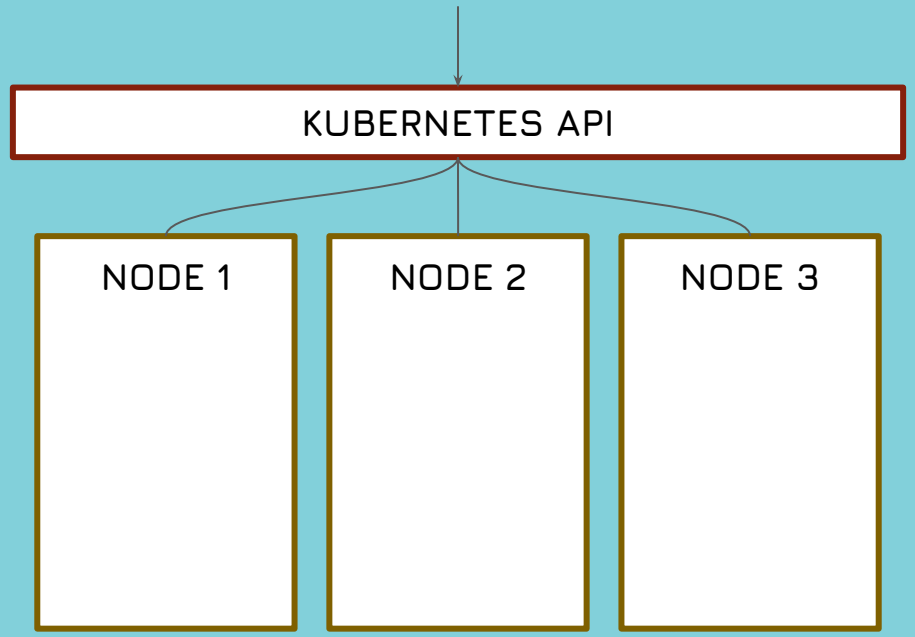
Kubernetes is a **declarative** way to **describe** your applications

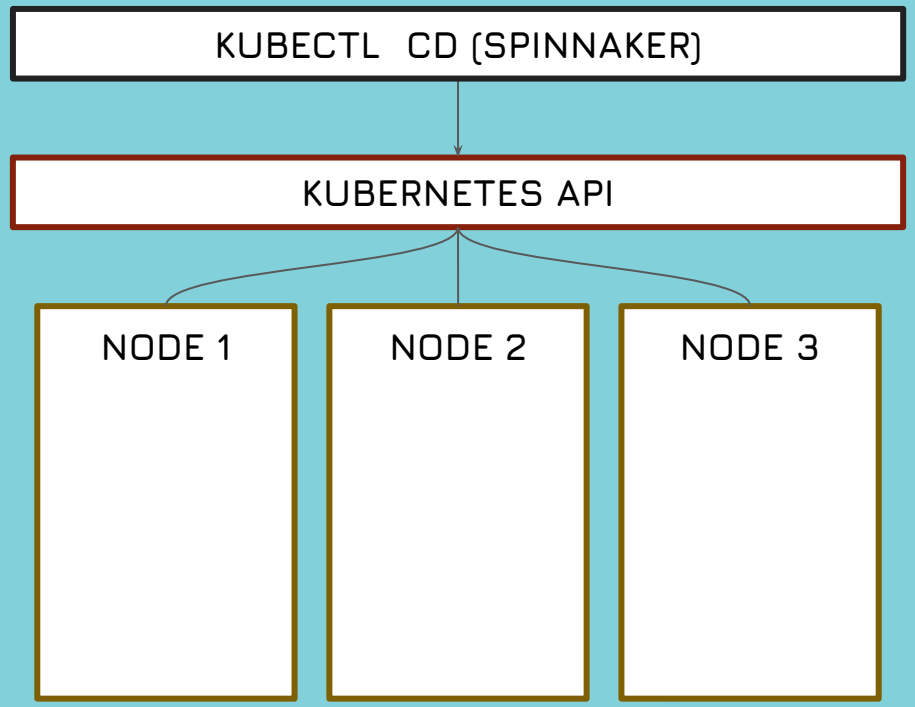



MASTER
NODE 1
NODE 2
NODE 3


MASTER
KUBERNETES API
SCHEDULER
CONTROLLER MANAGERS
NODE 1
NODE 2
NODE 3


MASTER
KUBERNETES API
SCHEDULER
CONTROLLER MANAGERS
NODE 1
KUBE PROXY
KUBELET
CONTAINER RUNTIME
NODE 2
KUBE PROXY
KUBELET
CONTAINER RUNTIME
NODE 3
KUBE PROXY
KUBELET
CONTAINER RUNTIME


KUBERNETES API
NODE 1
NODE 2
NODE 3

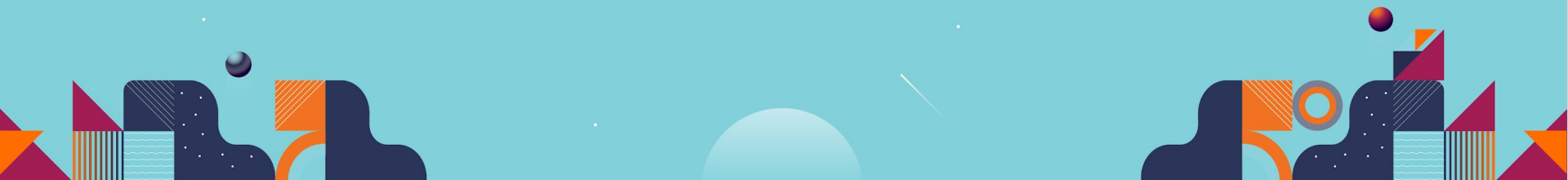

KUBECTL CD (SPINNAKER)
KUBERNETES API
NODE 1
NODE 2
NODE 3

Kube C.T.L

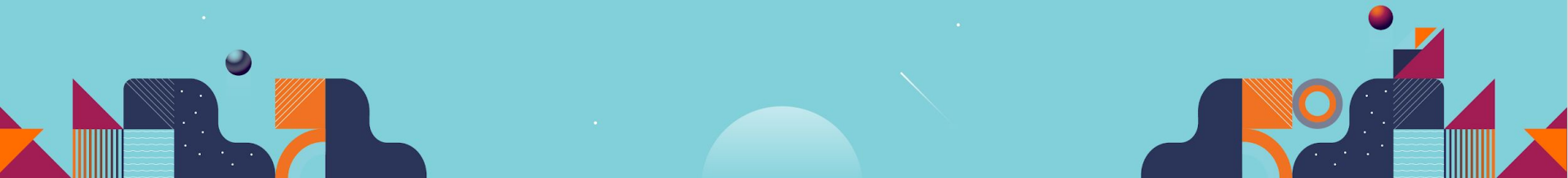
Kubecuddle

q-bec-til

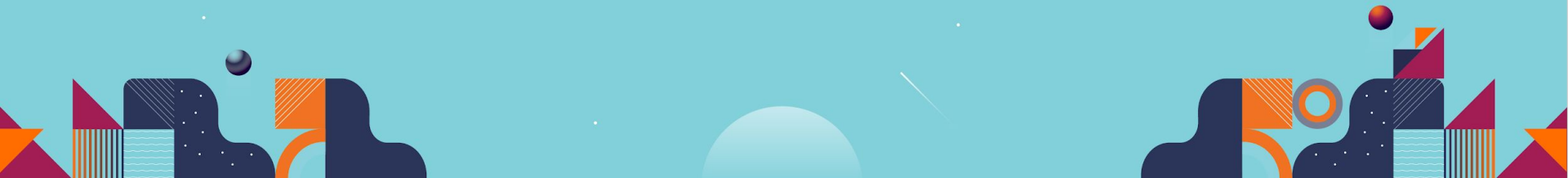
kubectl
(pronounced kube control)



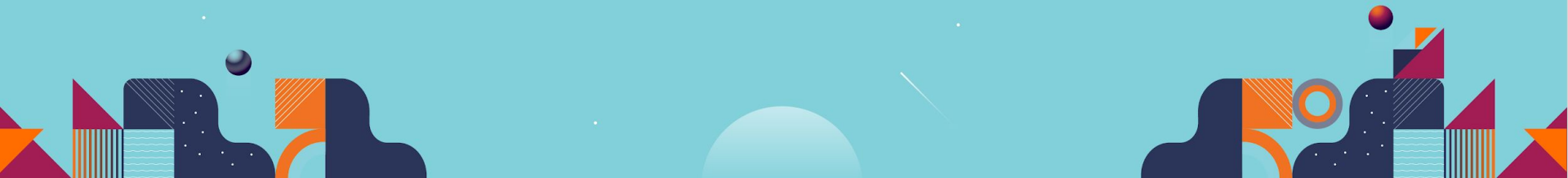
Everything in Kubernetes is a Resource



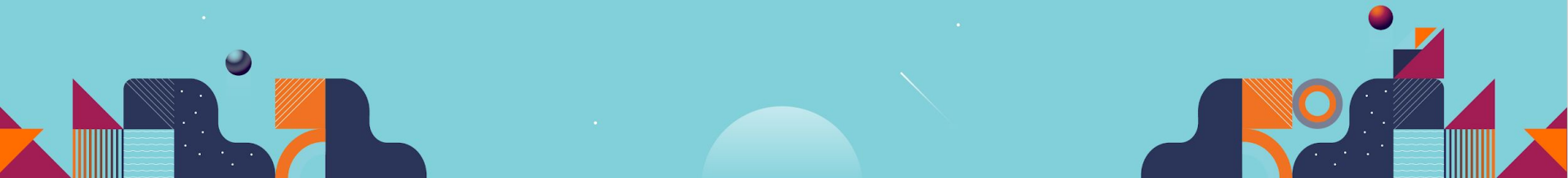
Resources are endpoints in Kubernetes API

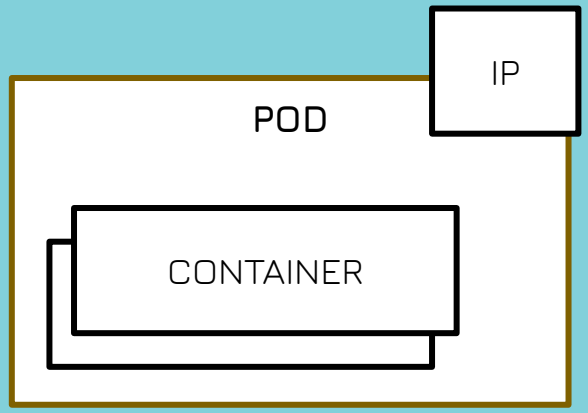


Labels to keep track of Resources



Pod Deployment Service Ingress Secret Configmap

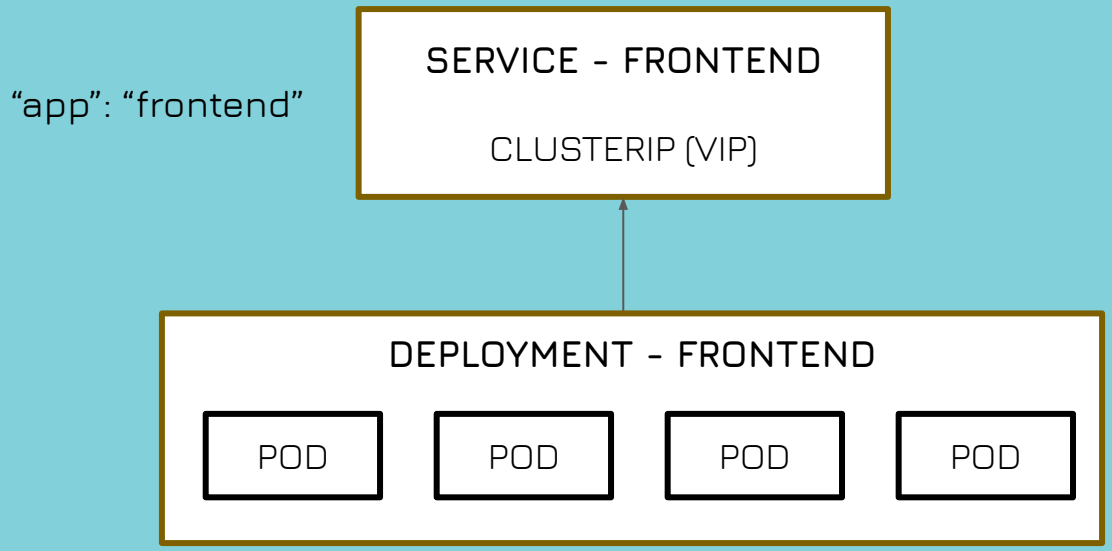


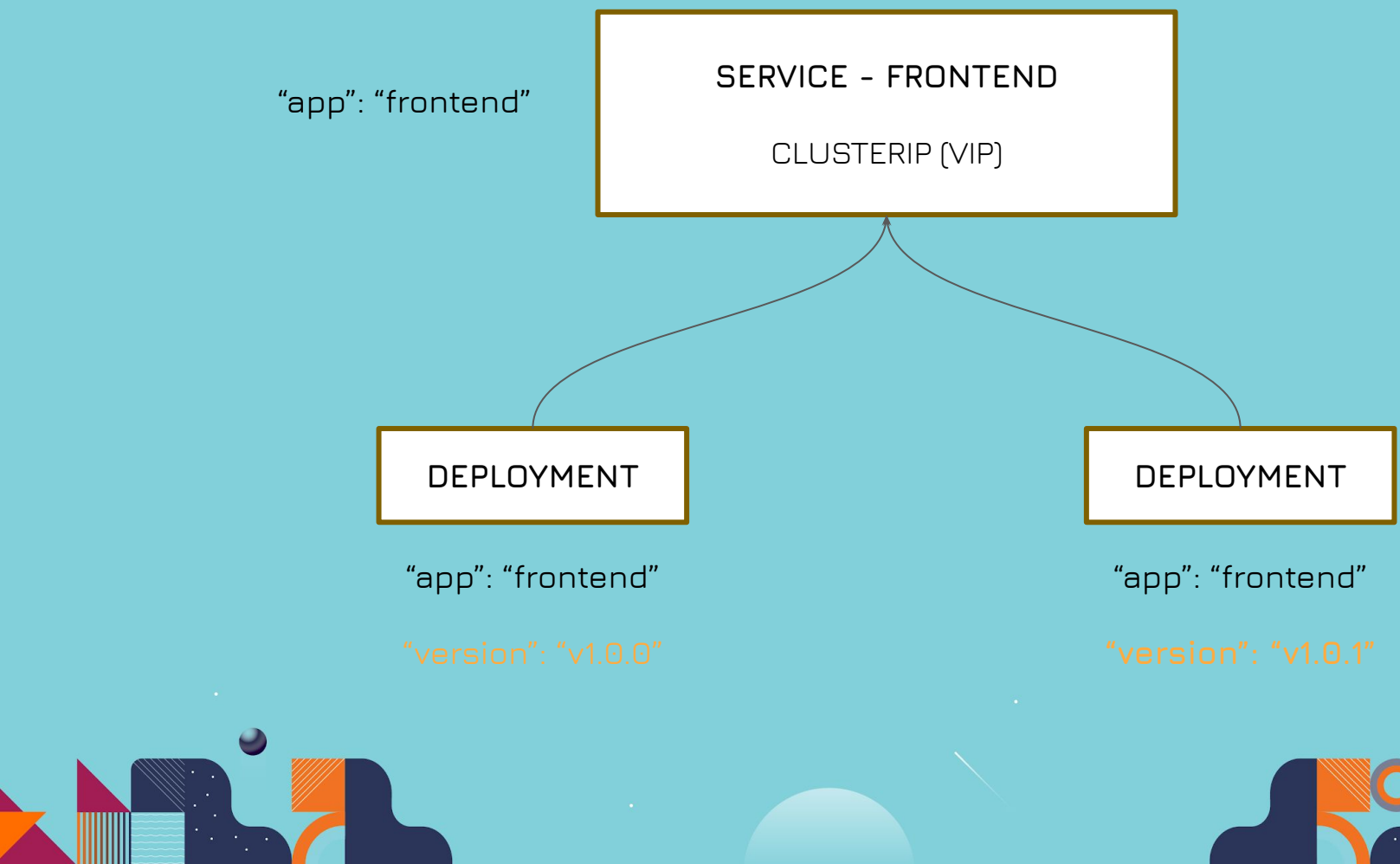


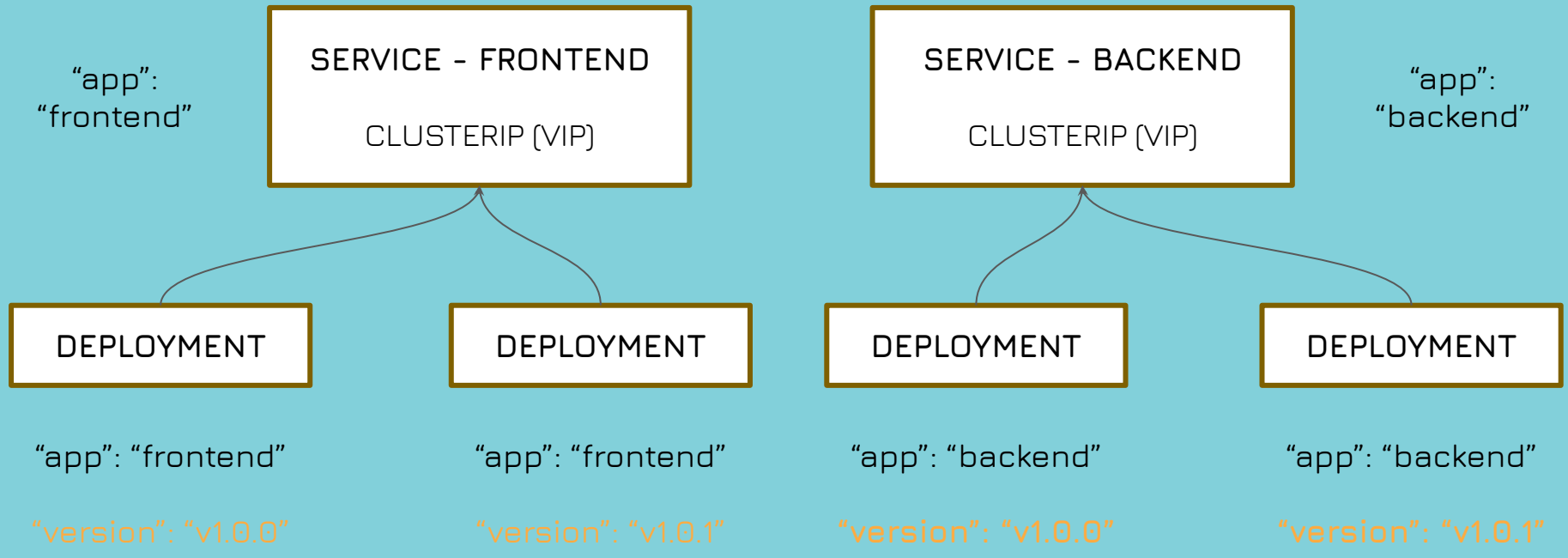
“app”: “frontend”

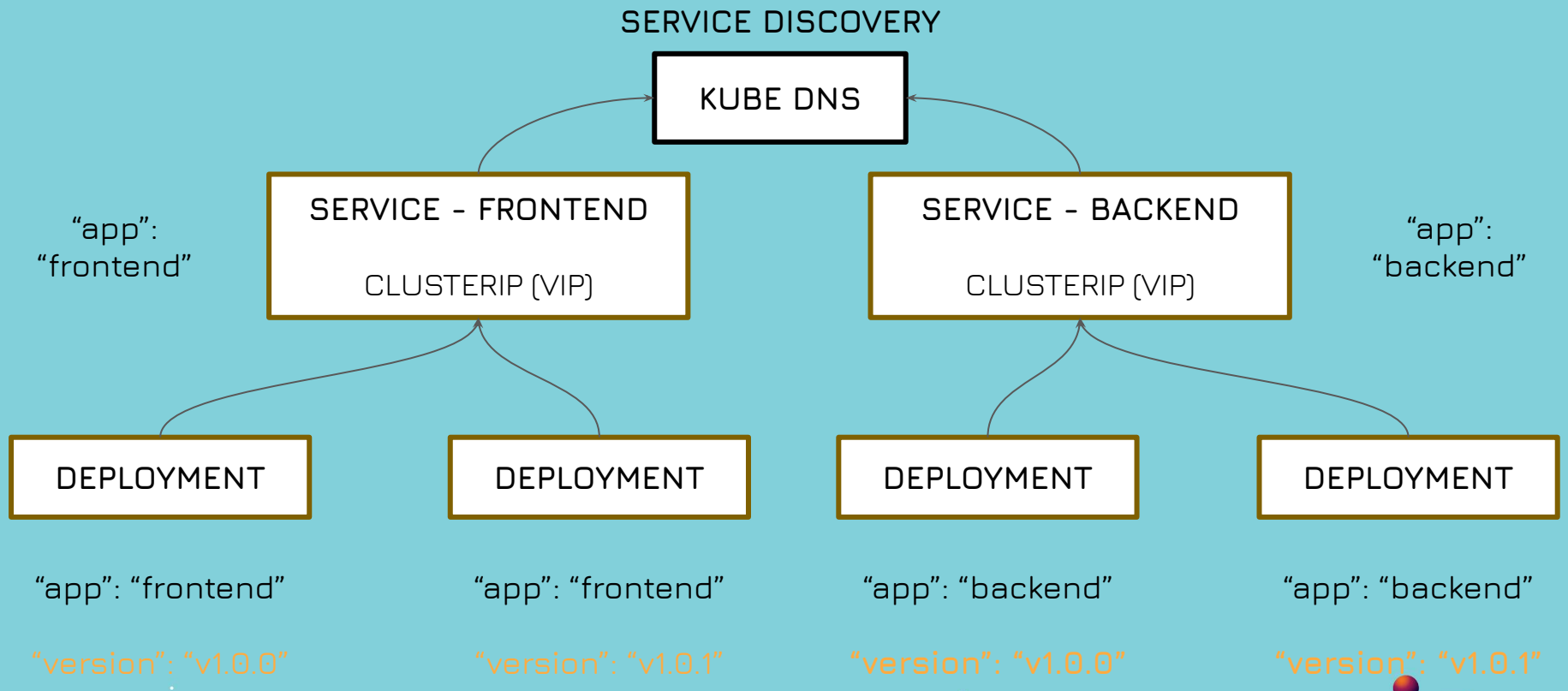
“app”: “frontend”

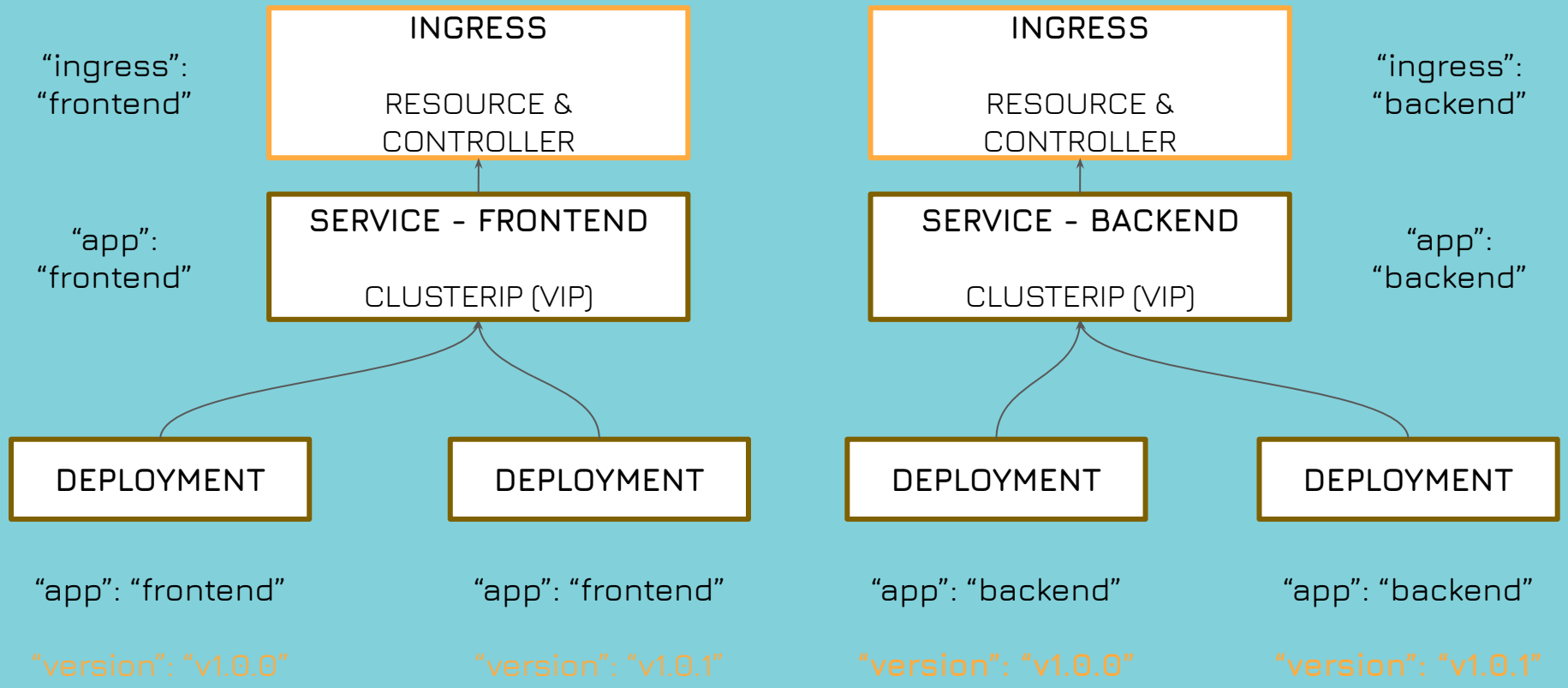
“version”: “v1.0.0”

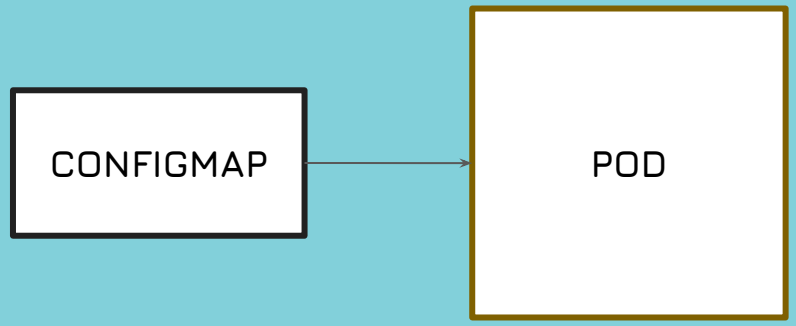

"app": "frontend"
SERVICE - FRONTEND
CLUSTERIP (VIP)
DEPLOYMENT - FRONTEND
POD
POD
POD
POD
"app": "frontend"
"version": "v1.0.0"

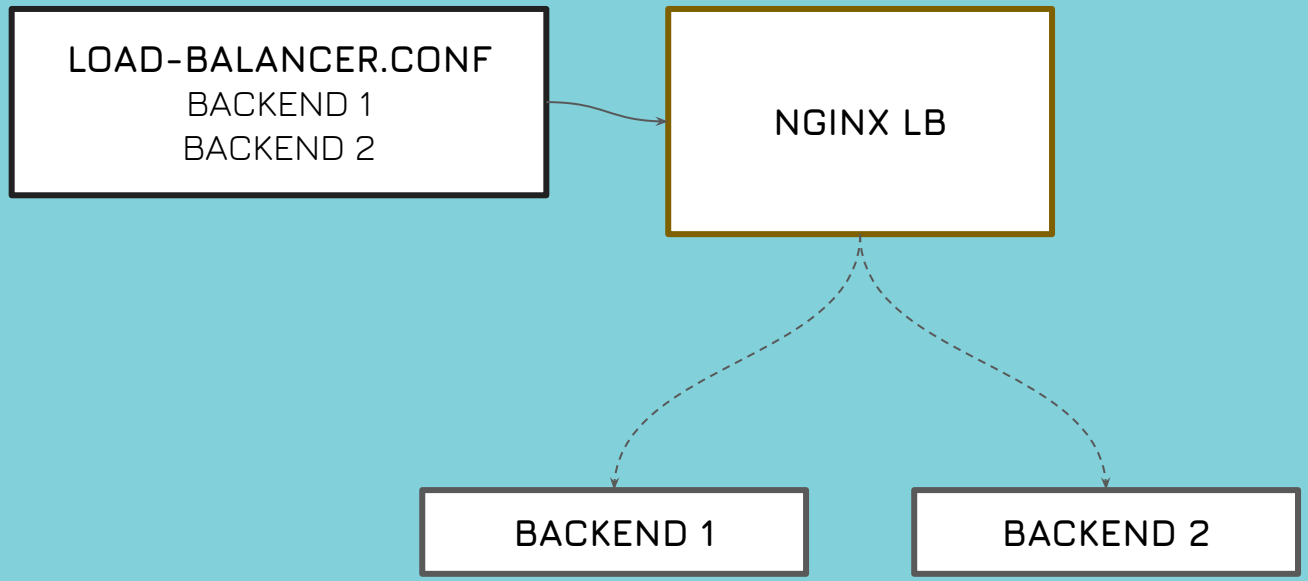

"app": "frontend"
SERVICE - FRONTEND
CLUSTERIP (VIP)
DEPLOYMENT
"app": "frontend"
"version": "V1.0.0"
DEPLOYMENT
"app": "frontend"
"version": "V1.0.1"

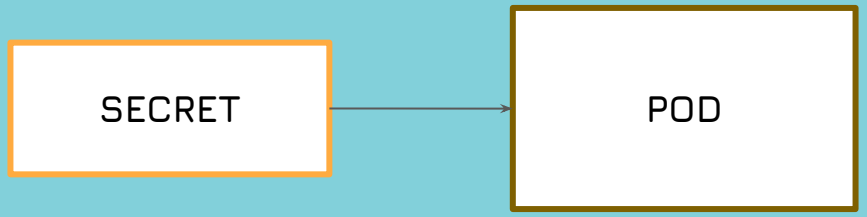

"app": "frontend"
SERVICE - FRONTEND
CLUSTERIP (VIP)
SERVICE - BACKEND
CLUSTERIP (VIP)
"app": "backend"
DEPLOYMENT
DEPLOYMENT
DEPLOYMENT
DEPLOYMENT
"app": "frontend"
"app": "frontend"
"app": "backend"
"app": "backend"
"version": "v1.0.0"
"version": "v1.0.1"
"version": "v1.0.0"
"version": "v1.0.1"

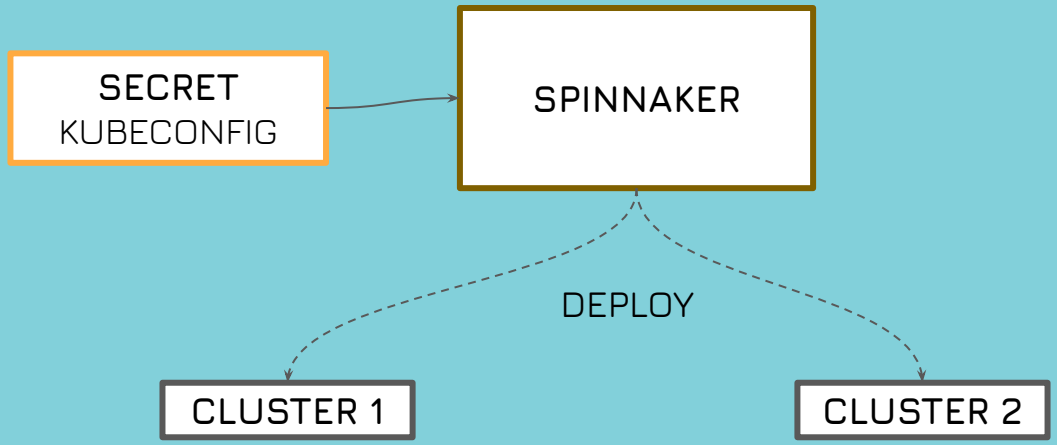

SERVICE DISCOVERY
KUBE DNS
SERVICE - FRONTEND
CLUSTERIP (VIP)
SERVICE - BACKEND
CLUSTERIP (VIP)
“app”: “frontend”
“app”: “backend”
DEPLOYMENT
DEPLOYMENT
DEPLOYMENT
DEPLOYMENT
“app”: “frontend”
“app”: “frontend”
“app”: “backend”
“app”: “backend”
“version”: “v1.0.0”
“version”: “v1.0.1”
“version”: “v1.0.0”
“version”: “v1.0.1”


INGRESS
RESOURCE & CONTROLLER
"ingress": "frontend"
SERVICE - FRONTEND
CLUSTERIP (VIP)
"app": "frontend"
DEPLOYMENT
DEPLOYMENT
"app": "frontend"
"app": "frontend"
"version": "v1.0.0"
"version": "v1.0.1"
INGRESS
RESOURCE & CONTROLLER
"ingress": "backend"
SERVICE - BACKEND
CLUSTERIP (VIP)
"app": "backend"
DEPLOYMENT
DEPLOYMENT
"app": "backend"
"app": "backend"
"version": "v1.0.0"
"version": "v1.0.1"


CONFIGMAP
POD


LOAD-BALANCER.CONF
BACKEND 1
BACKEND 2
NGINX LB
BACKEND 1
BACKEND 2


SECRET
POD


SECRET
KUBECONFIG
SPINNAKER
DEPLOY
CLUSTER 1
CLUSTER 2

Custom Resources

ReactiveOps RBAC Manager

README.md

RBAC Manager

go report A+ PASSED

RBAC Manager was designed to simplify authorization in Kubernetes. This is an operator that supports declarative configuration for RBAC with new custom resources. Instead of managing role bindings or service accounts directly, you can specify a desired state and RBAC Manager will make the necessary changes to achieve that state.

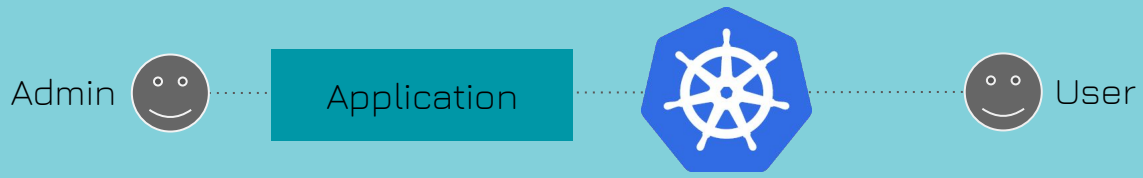
This project has three main goals:

1. Provide a declarative approach to RBAC that is more approachable and scalable.
2. Reduce the amount of configuration required for great auth.
3. Enable automation of RBAC configuration updates with CI/CD.

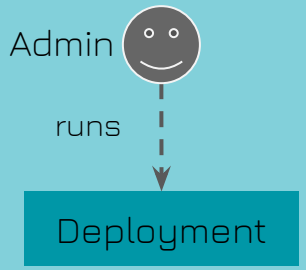
Istio

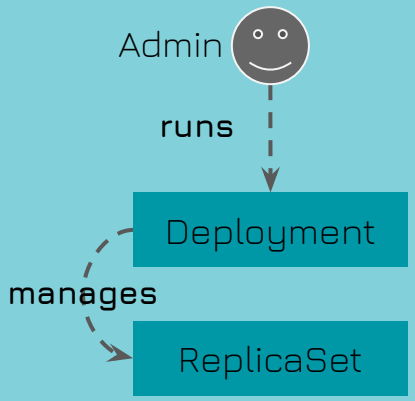
Bringing it all together

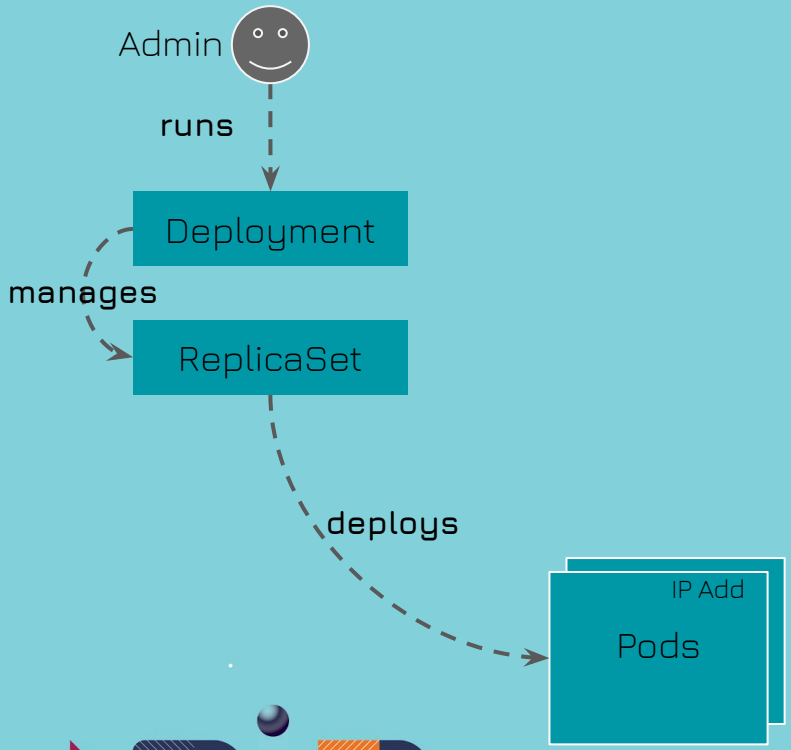
An Application Deployment to User Story

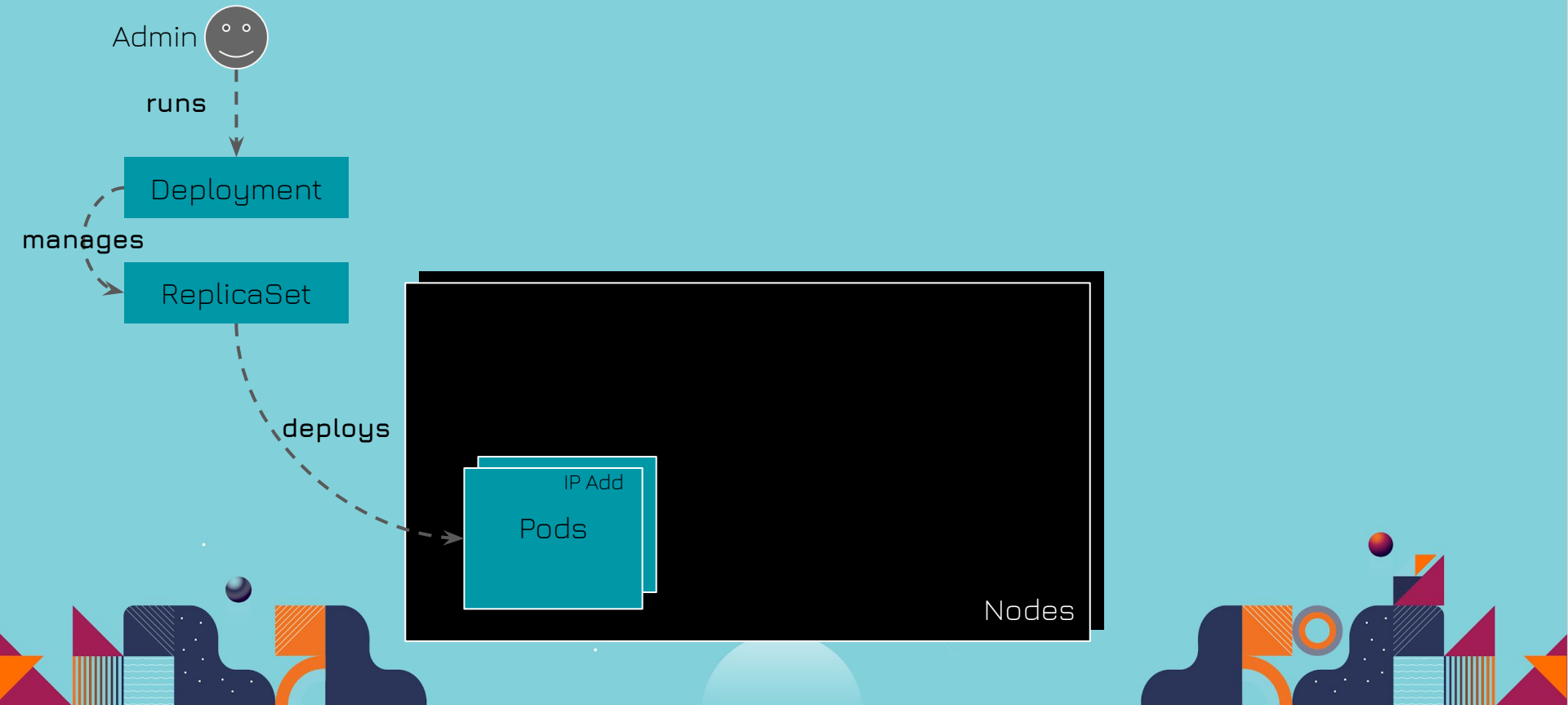


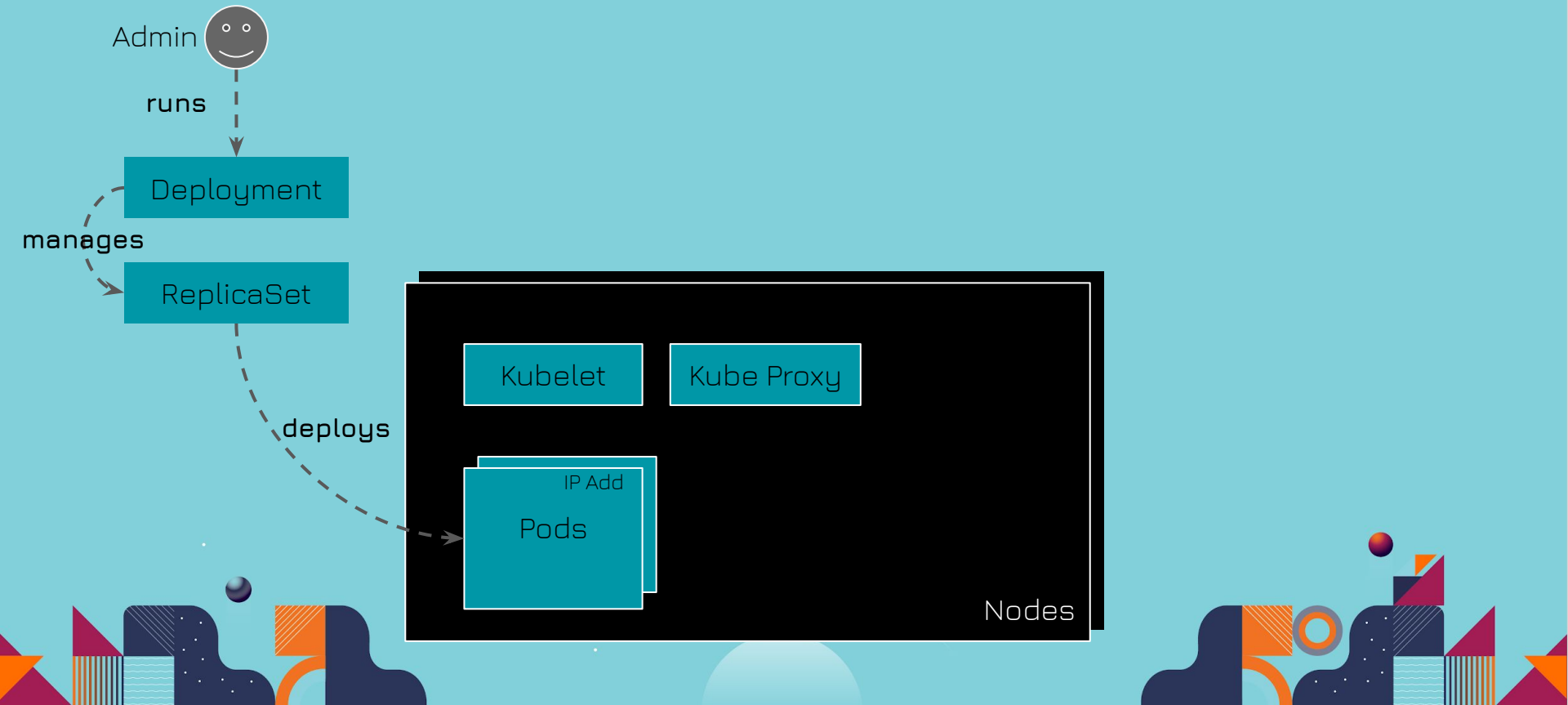
Admin

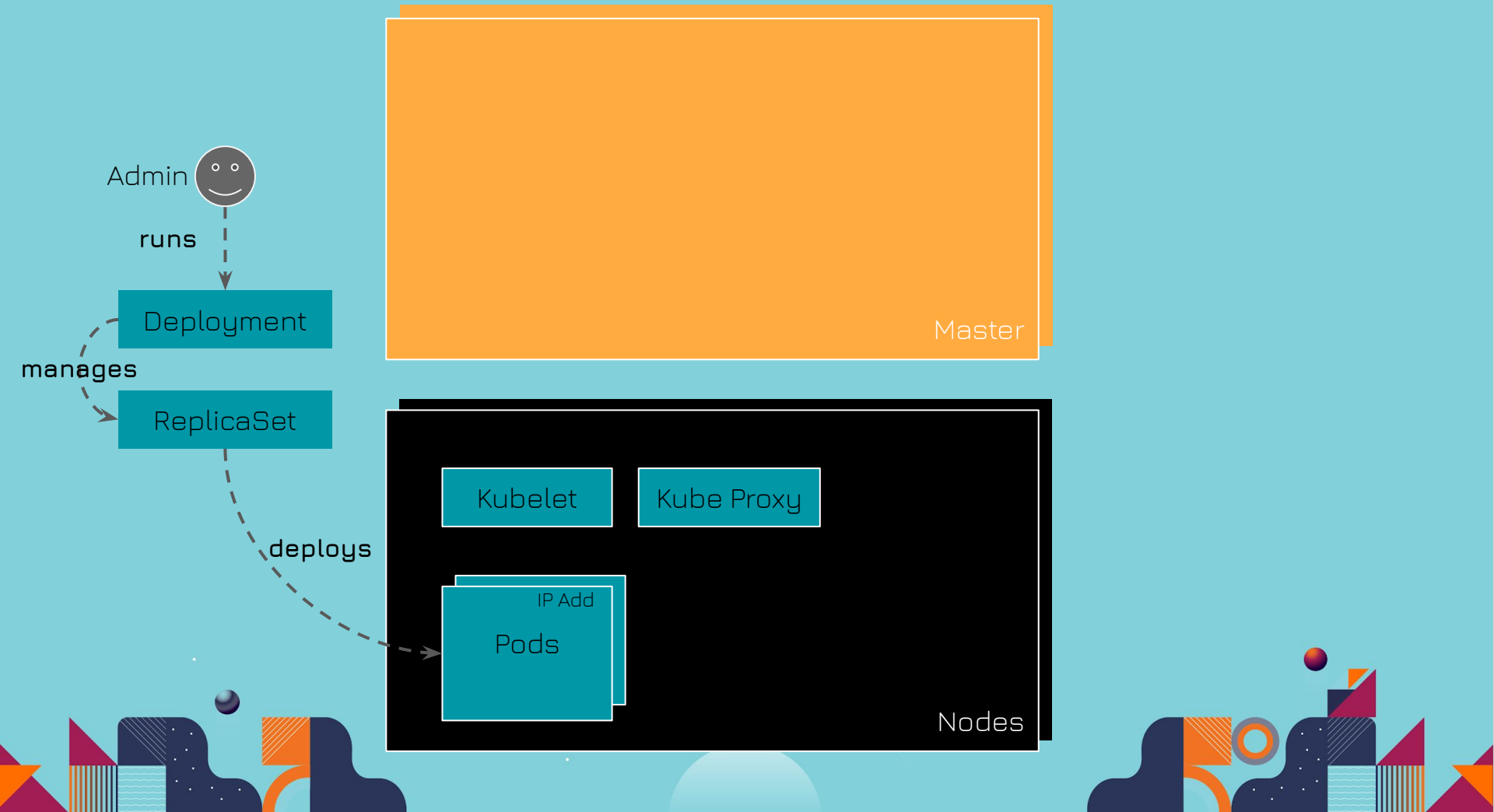

Admin
runs
Deployment

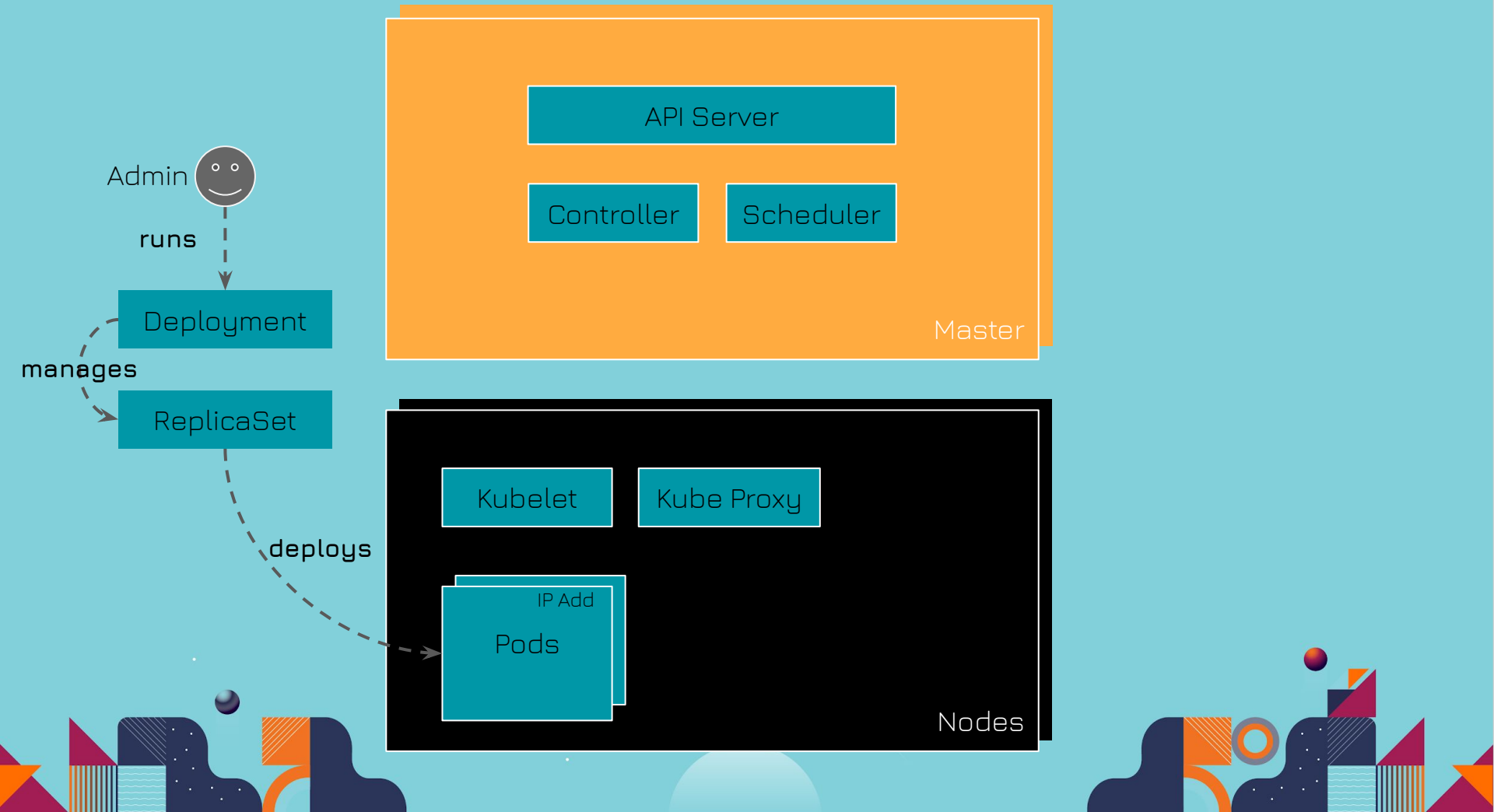

Admin
runs
Deployment
manages
ReplicaSet

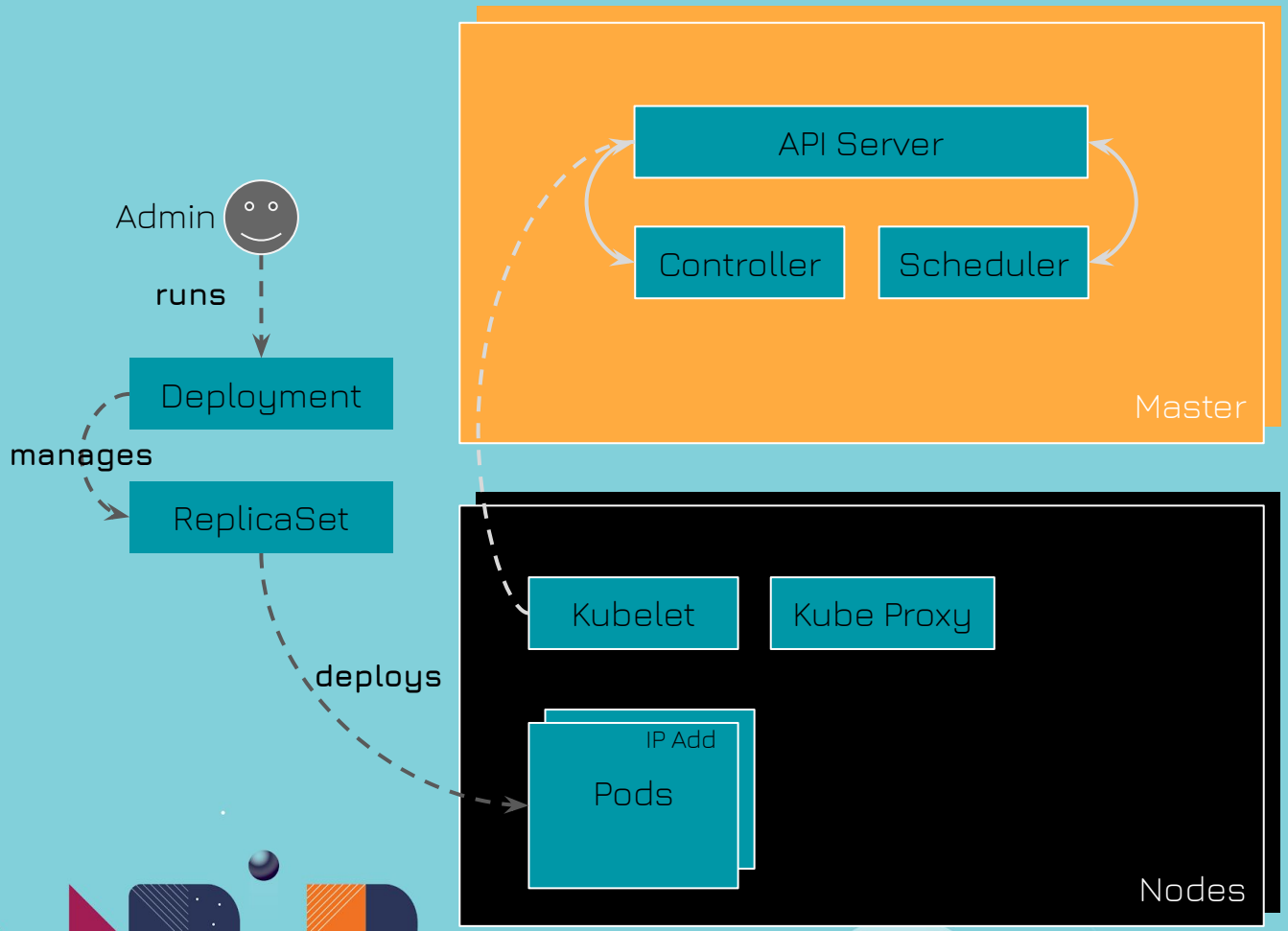

Admin
runs
Deployment
manages
ReplicaSet
deploys
IP Add
Pods

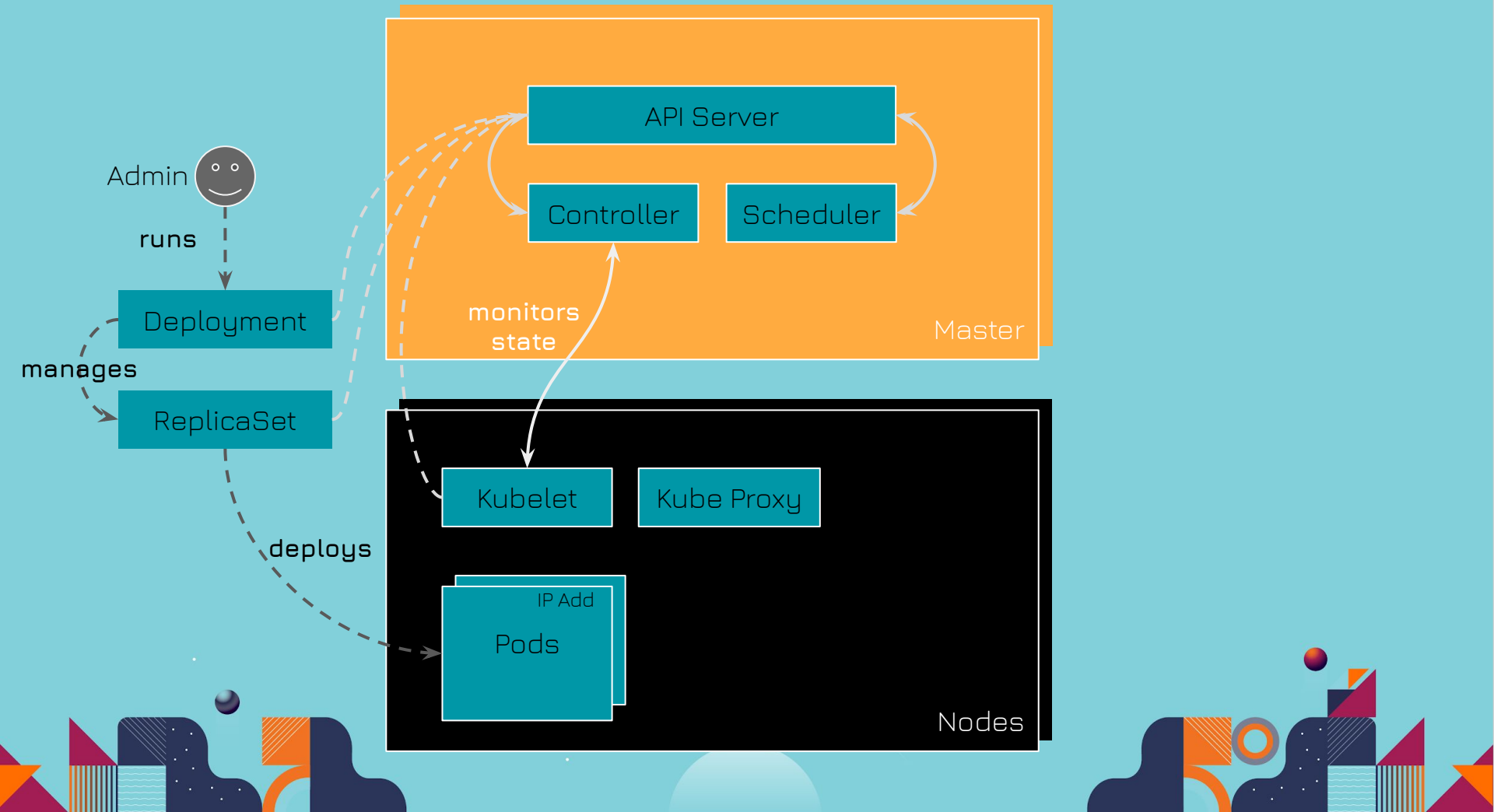

Admin
runs
Deployment
manages
ReplicaSet
deploys
IP Add
Pods
Nodes

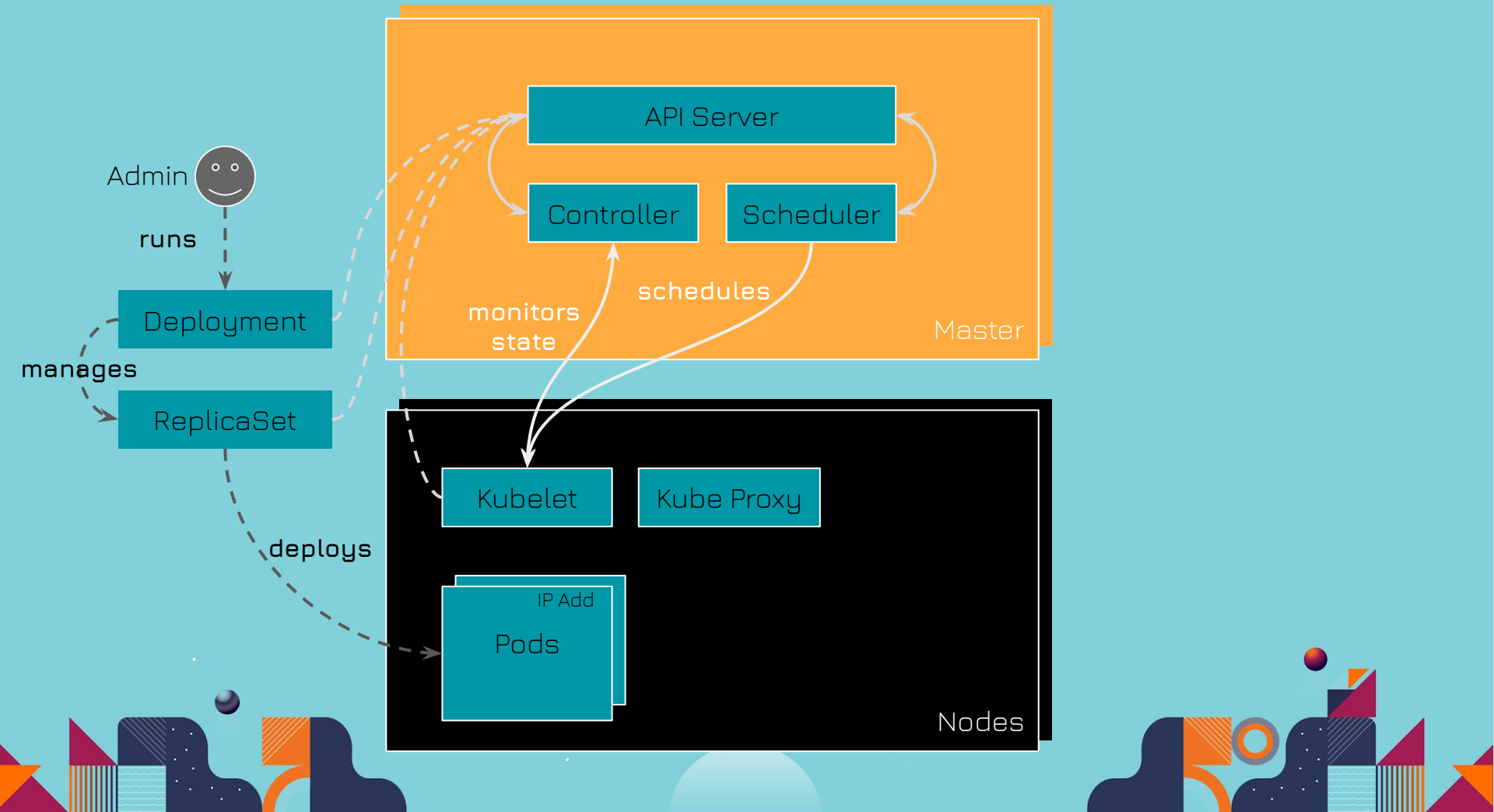

Admin
runs
Deployment
manages
ReplicaSet
deploys
Kubelet
Kube Proxy
IP Add
Pods
Nodes

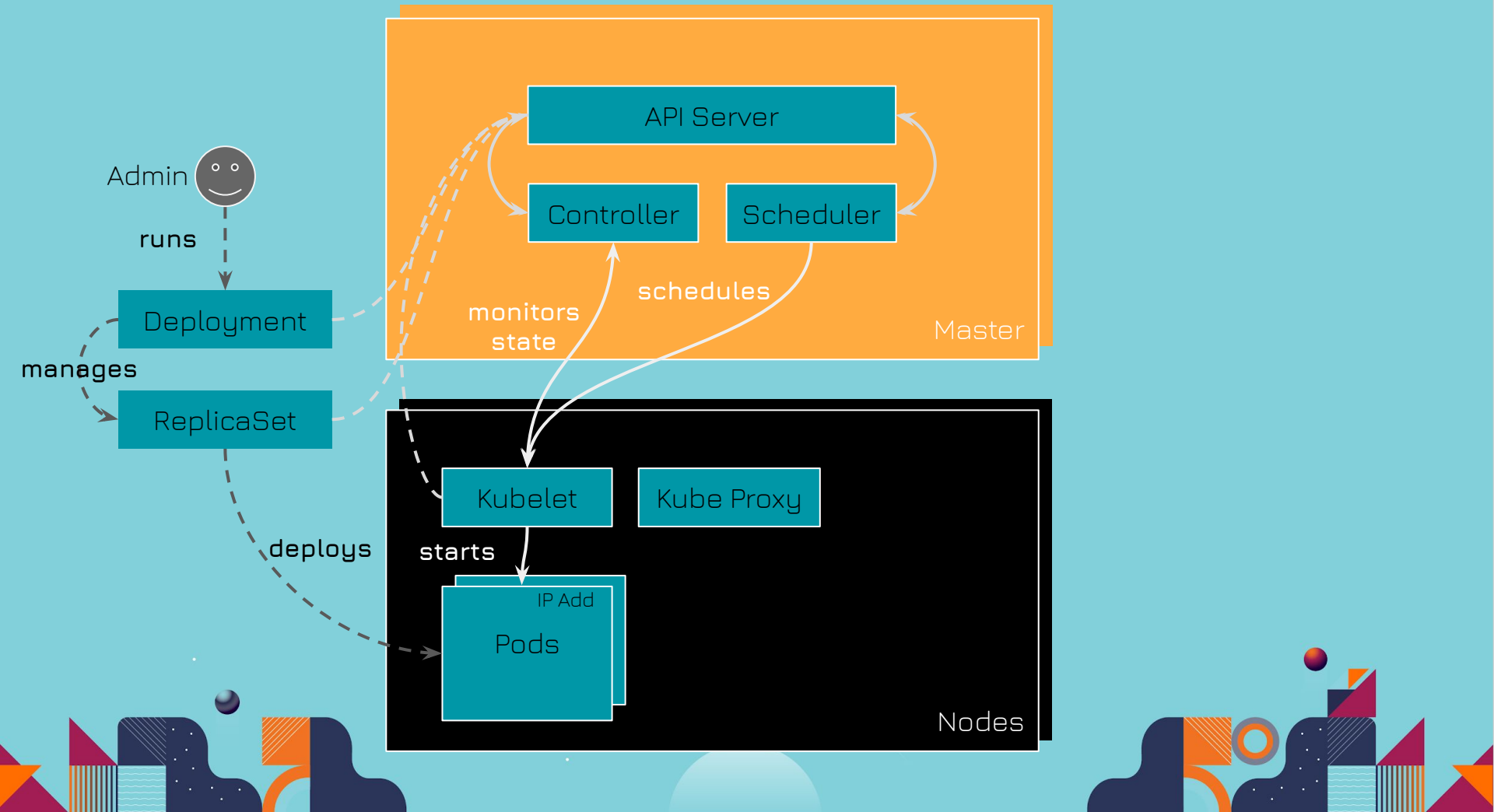

Admin
runs
Deployment
manages
ReplicaSet
deploys
Master
Kubelet
Kube Proxy
IP Add
Pods
Nodes

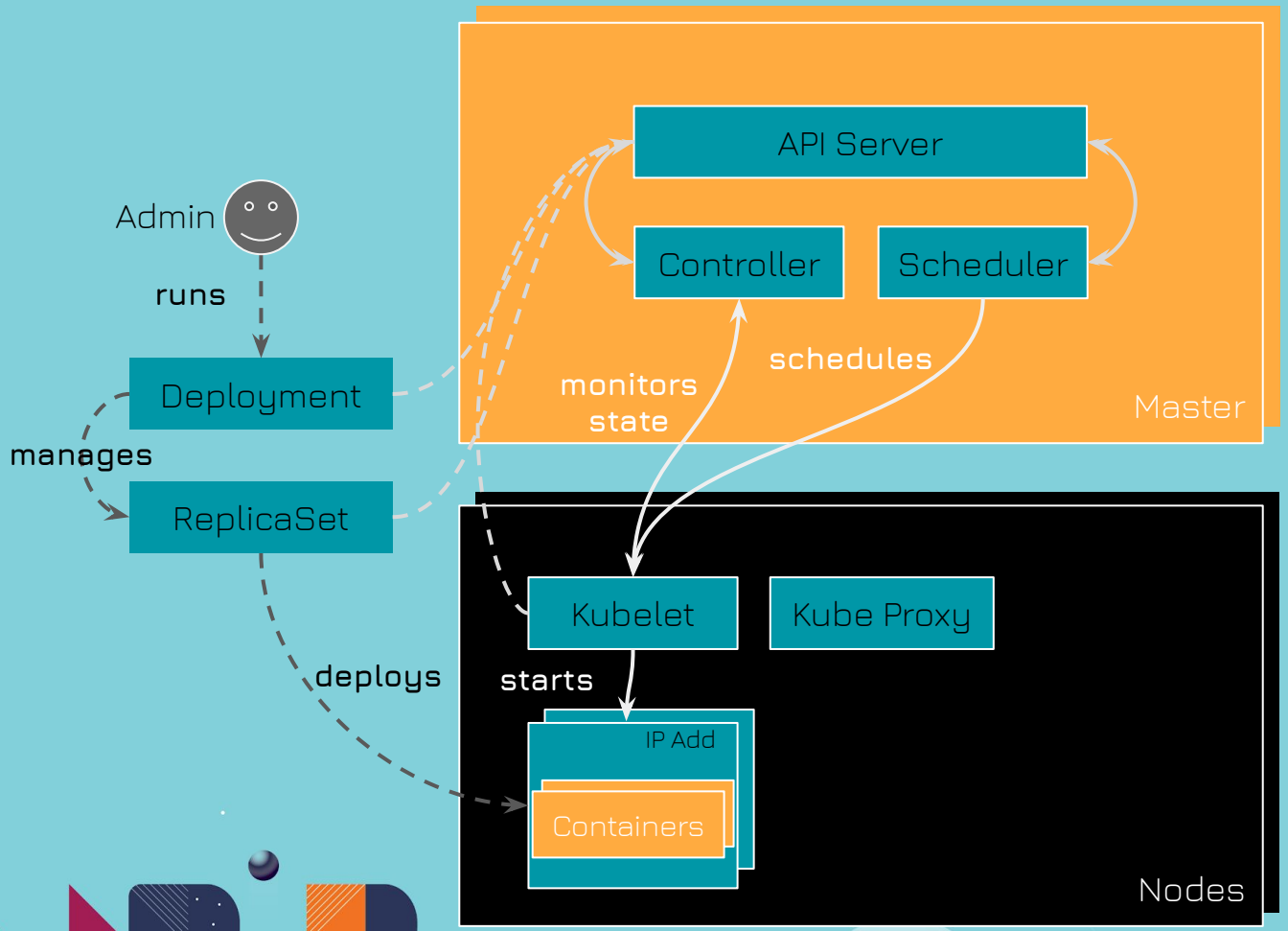

API Server
Controller
Scheduler
Master
Admin
runs
Deployment
manages
ReplicaSet
deploys
Kubelet
Kube Proxy
IP Add
Pods
Nodes

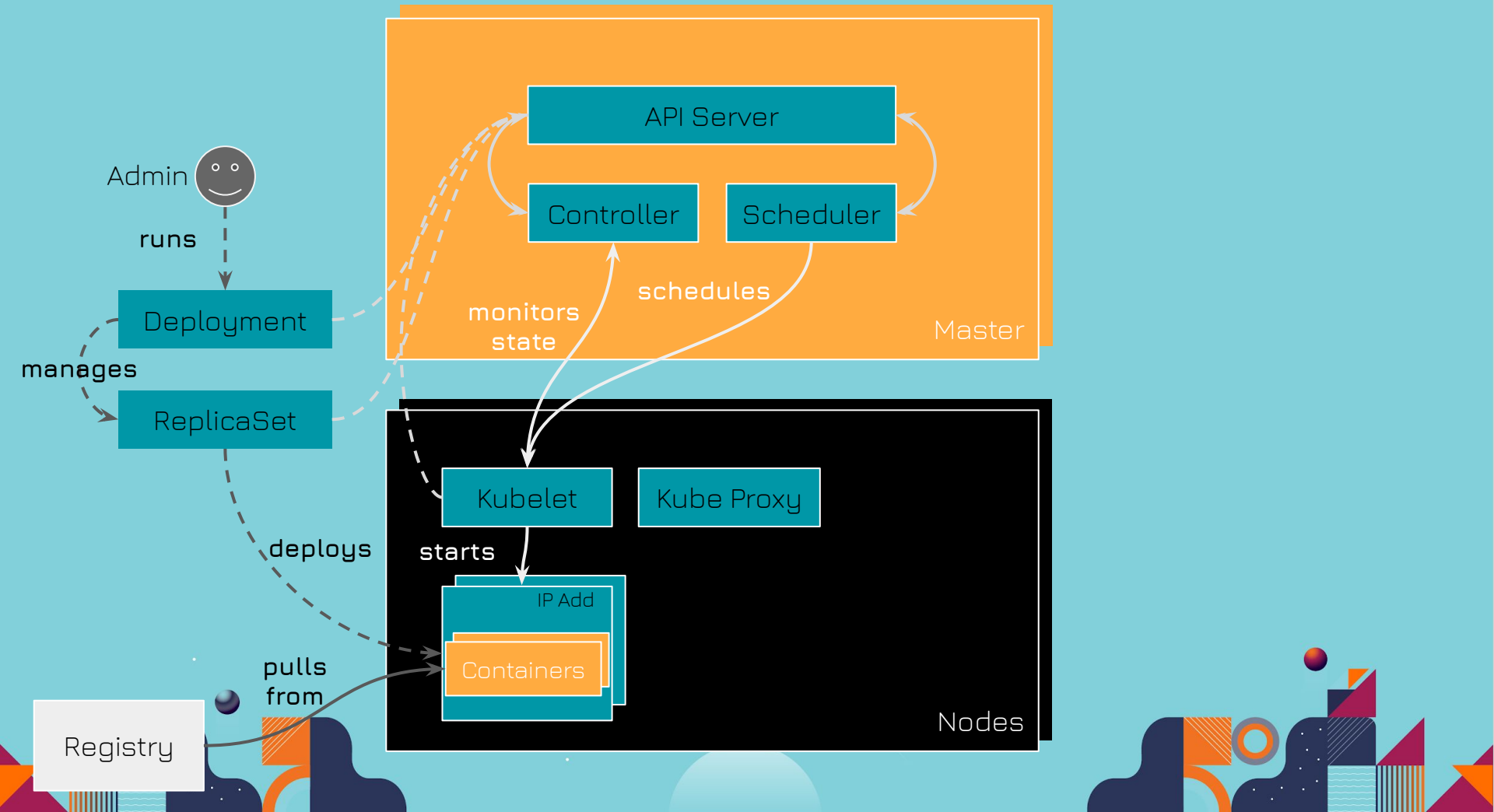

API Server
Controller
Scheduler
Master
Admin
runs
Deployment
manages
ReplicaSet
deploys
Kubelet
Kube Proxy
IP Add
Pods
Nodes

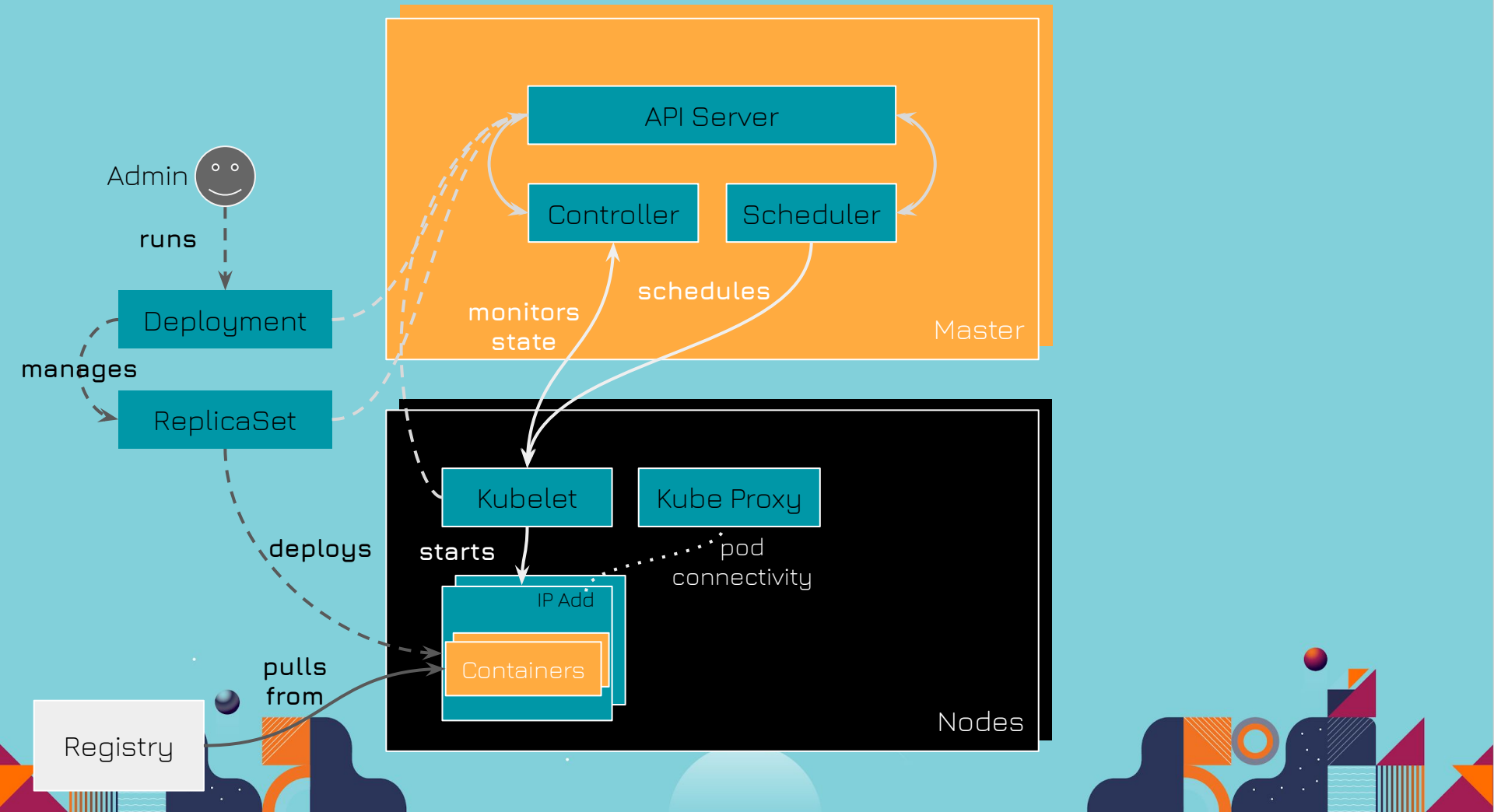

API Server
Controller
Scheduler
Master
Admin
runs
Deployment
manages
ReplicaSet
monitors
state
Kubelet
Kube Proxy
deploys
IP Add
Pods
Nodes

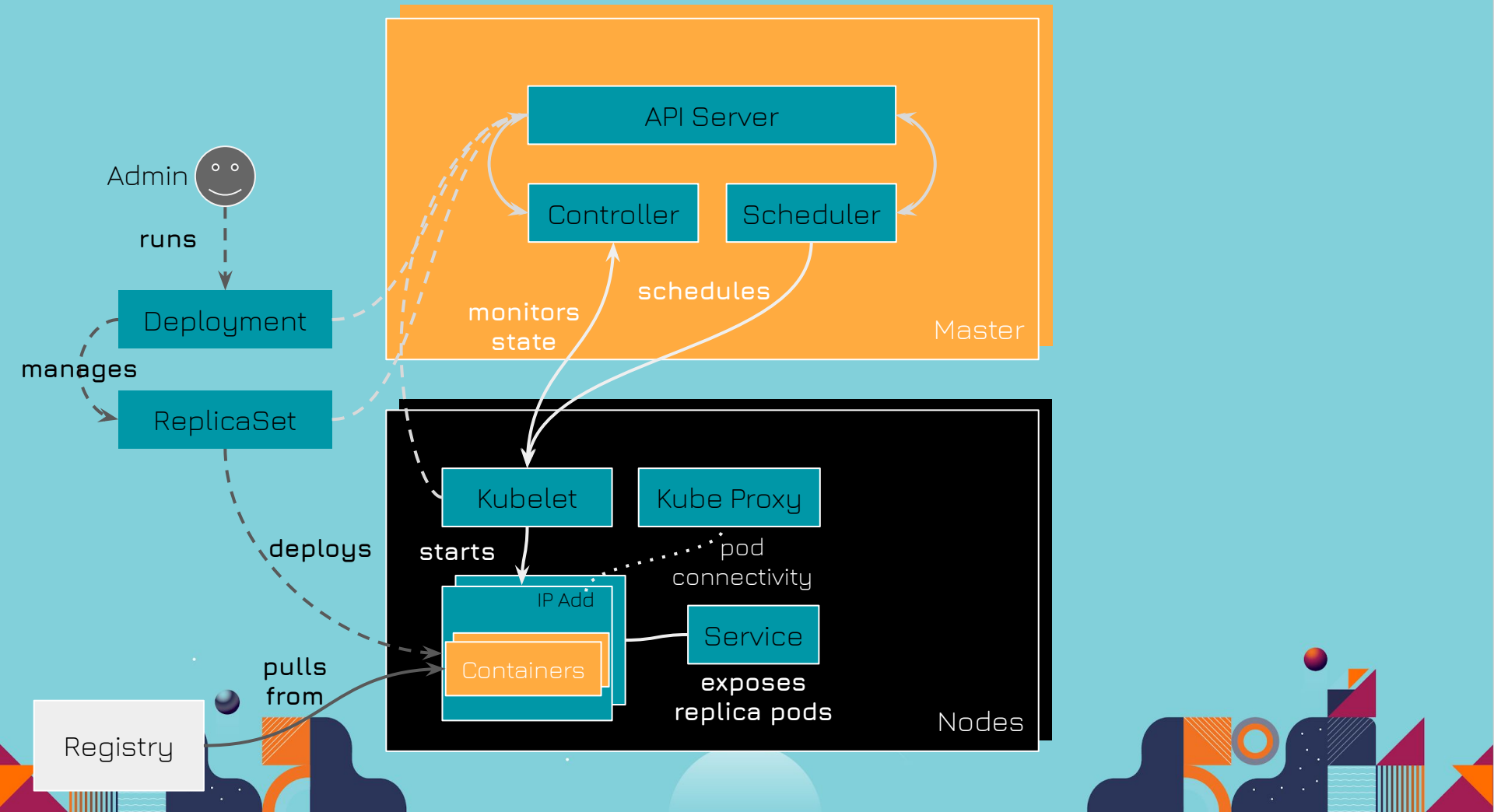

API Server
Controller
Scheduler
Master
Admin
runs
Deployment
manages
ReplicaSet
deploys
monitors
state
schedules
Kubelet
Kube Proxy
IP Add
Pods
Nodes

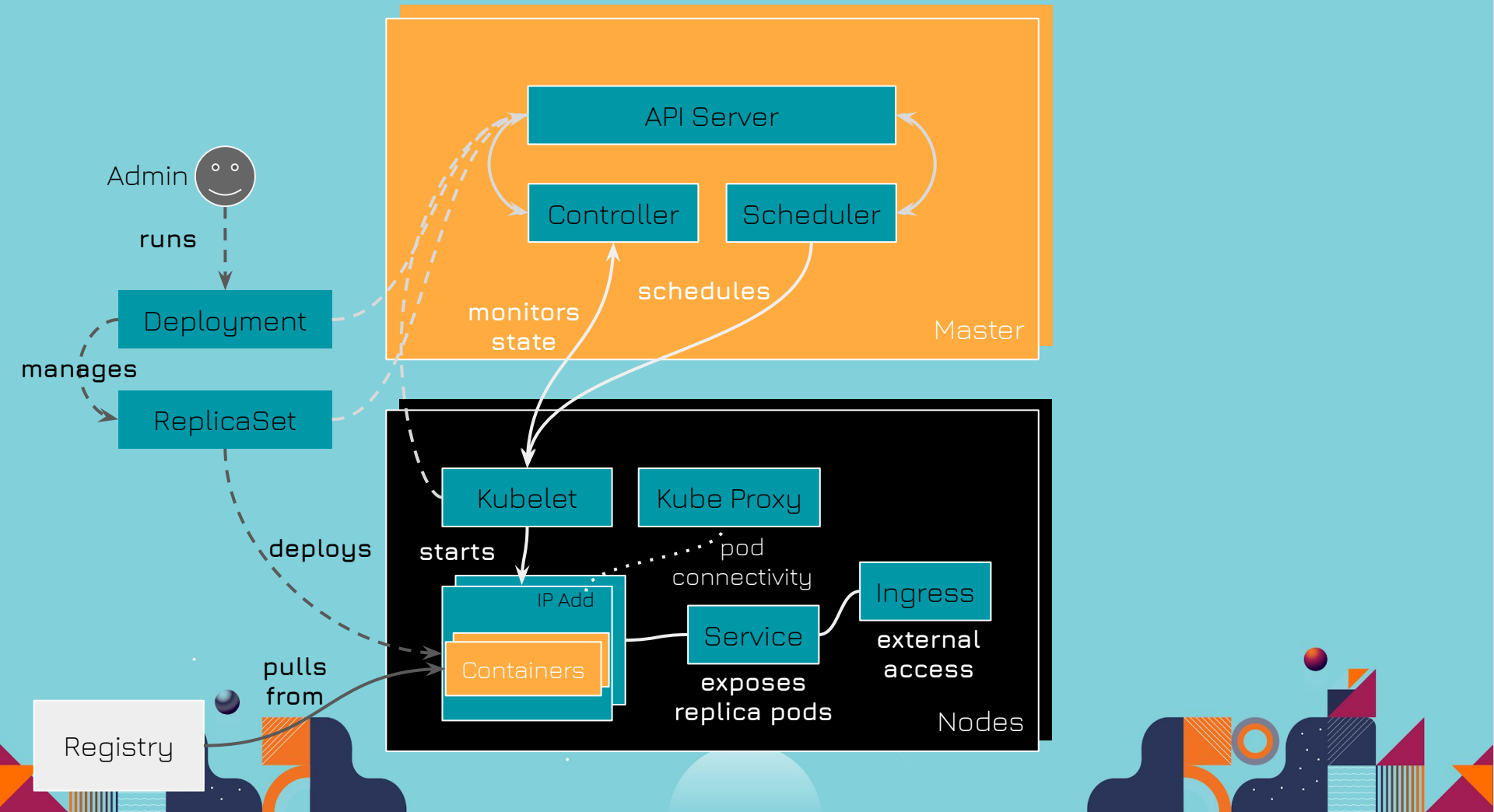

API Server
Controller
Scheduler
Master
Admin
runs
Deployment
manages
ReplicaSet
deploys
monitors
state
schedules
Kubelet
Kube Proxy
starts
IP Add
Pods
Nodes

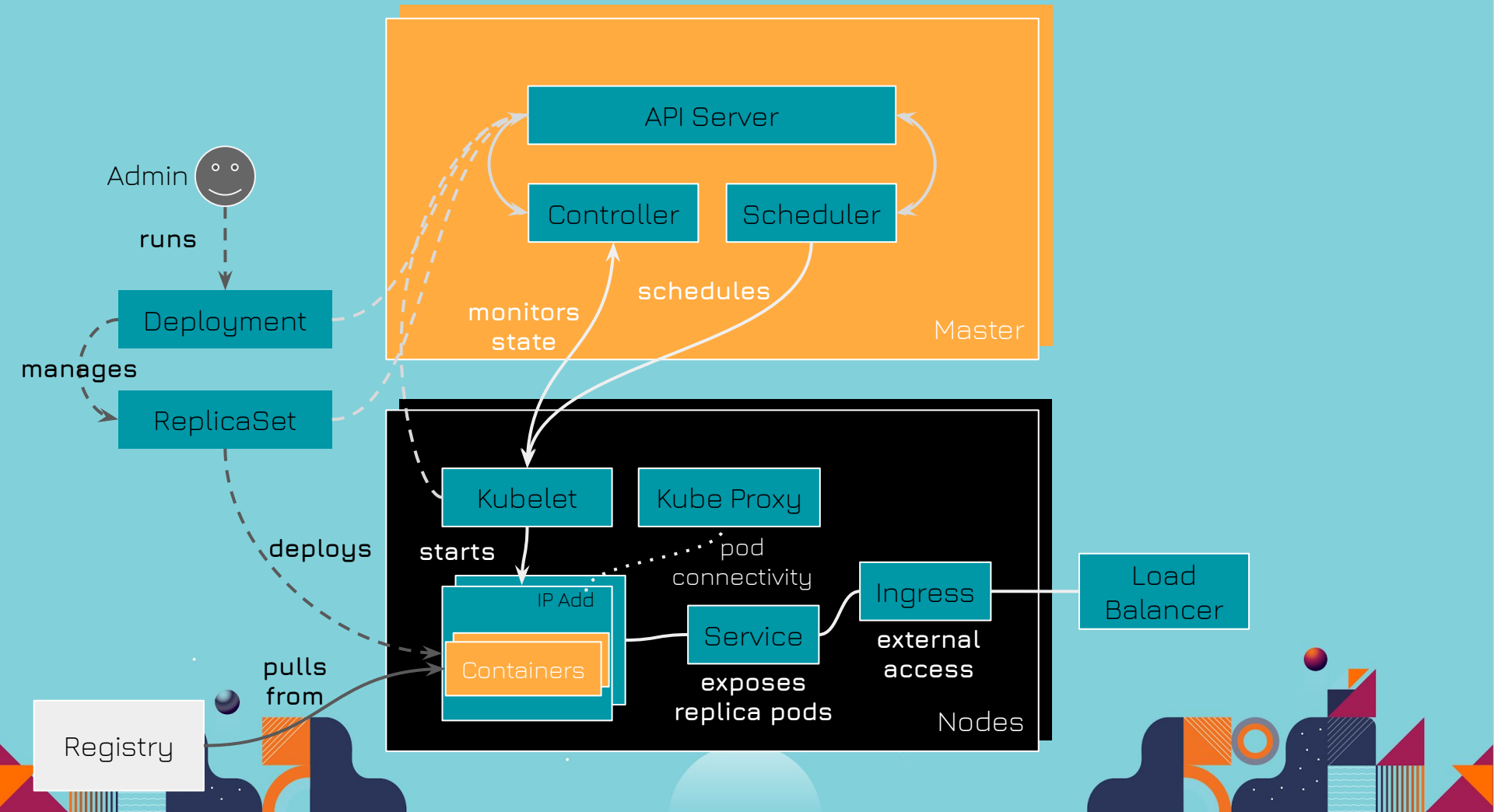

API Server
Controller
Scheduler
Master
Admin
runs
Deployment
manages
ReplicaSet
deploys
monitors
state
schedules
Kubelet
Kube Proxy
starts
IP Add
Containers
Nodes

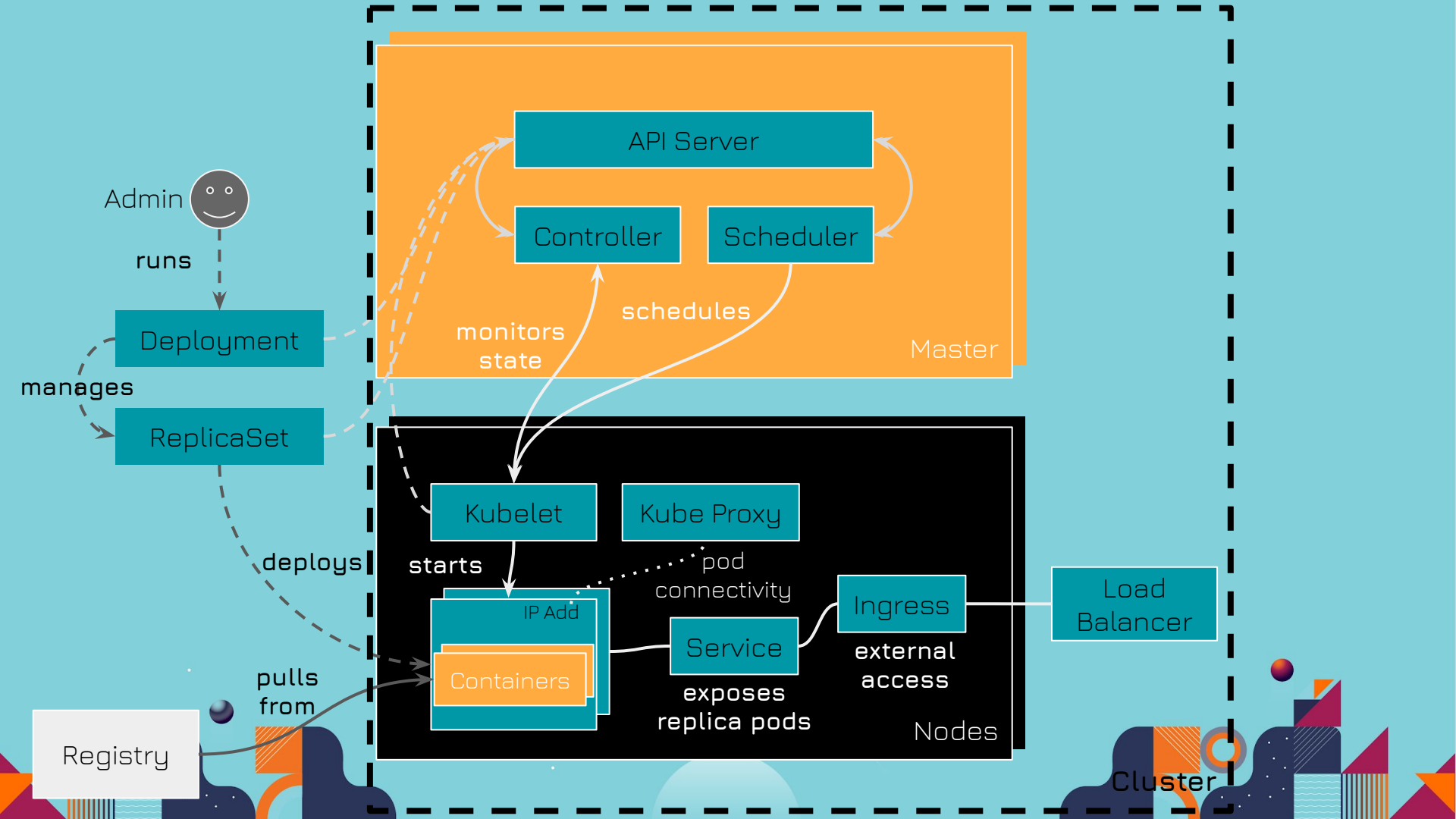

API Server
Controller
Scheduler
Master
Admin
runs
Deployment
manages
ReplicaSet
monitors
state
schedules
Kubelet
Kube Proxy
starts
IP Add
Containers
deploys
pulls
from
Registry
Nodes

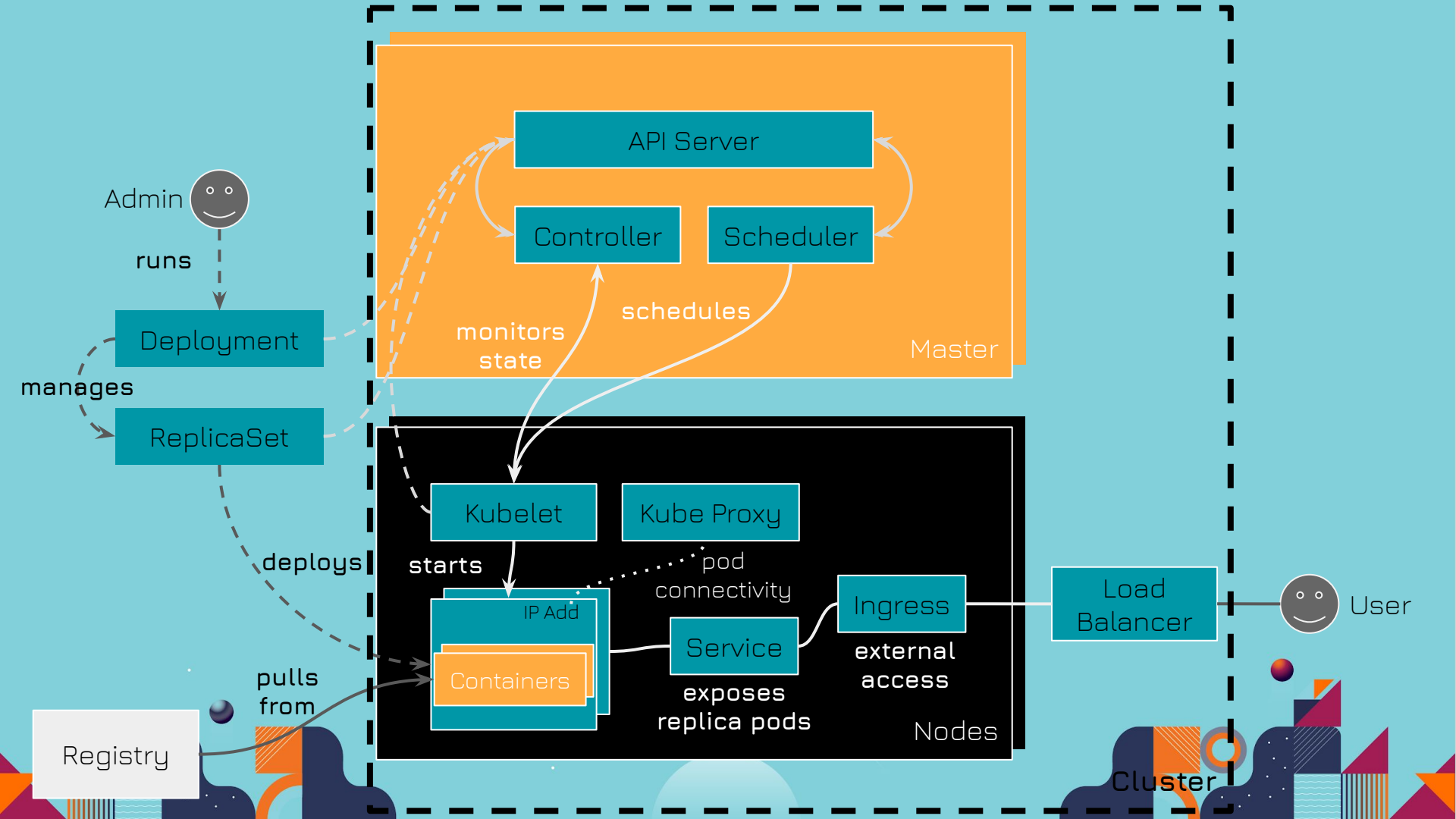

API Server
Controller
Scheduler
Master
Admin
runs
Deployment
manages
ReplicaSet
monitors
state
schedules
Kubelet
Kube Proxy
starts
pod
connectivity
deploys
IP Add
Containers
Nodes
pulls
from
Registry


API Server
Controller
Scheduler
Master
Admin
runs
Deployment
manages
ReplicaSet
monitors
state
schedules
deploys
Kubelet
Kube Proxy
starts
pod
connectivity
IP Add
Containers
Service
exposes
replica pods
Nodes
pulls
from
Registry


API Server
Controller
Scheduler
Master
Admin
runs
Deployment
manages
ReplicaSet
monitors
state
schedules
deploys
Kubelet
Kube Proxy
starts
pod
connectivity
IP Add
Containers
Service
Ingress
external
access
exposes
replica pods
Nodes
pulls
from
Registry


API Server
Controller
Scheduler
Master
Admin
runs
Deployment
manages
ReplicaSet
monitors
state
schedules
Kubelet
Kube Proxy
starts
pod
connectivity
deploys
IP Add
Containers
Service
exposes
replica pods
Ingress
external
access
Load
Balancer
Nodes
pulls
from
Registry

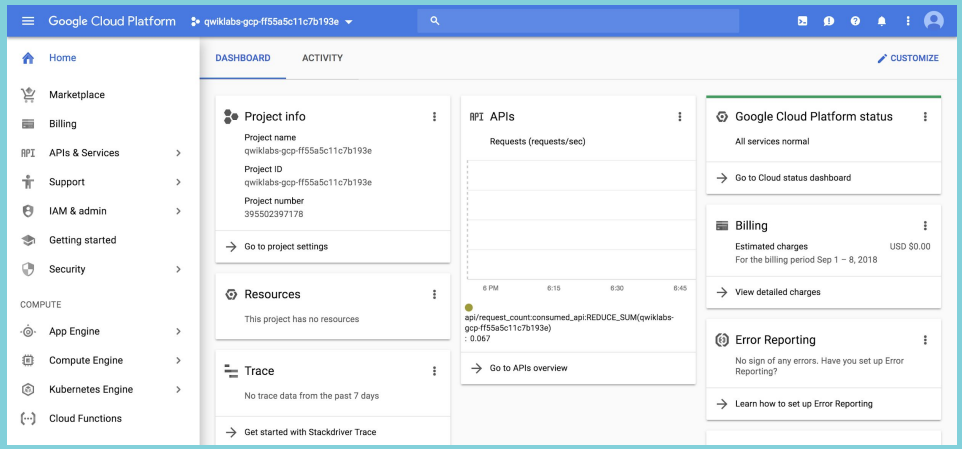

API Server
Controller
Scheduler
Master
Admin
runs
Deployment
manages
ReplicaSet
monitors
state
schedules
Kubelet
Kube Proxy
starts
pod
connectivity
deploys
IP Add
Containers
Service
exposes
replica pods
Ingress
external
access
Load
Balancer
Nodes
pulls
from
Registry
Cluster


API Server
Controller
Scheduler
Master
Admin
runs
Deployment
manages
ReplicaSet
monitors
state
schedules
Kubelet
Kube Proxy
starts
pod
connectivity
deploys
IP Add
Containers
Service
exposes
replica pods
Ingress
external
access
Load
Balancer
User
Nodes
pulls
from
Registry
Cluster

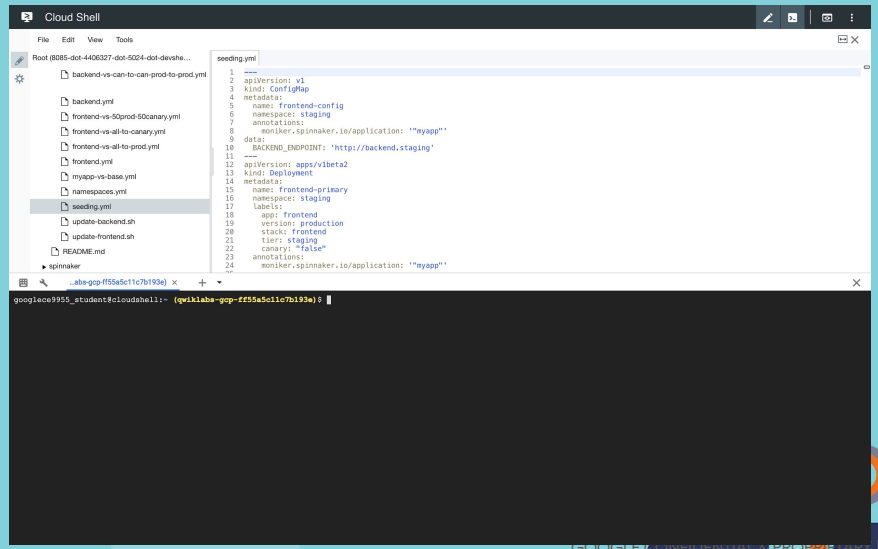
Check on progress of setup script

Know before you start the labs

Cloud Console



Cloud Shell Code Editor (BETA)



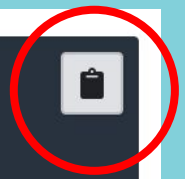
Don’t hit END LAB unless you’re done/sure!

Always know your context!

```
~ (gke-spinnaker) $ kubectx
gke-central
gke-spinnaker
gke-west
~ (gke-spinnaker) $
```

Use copy/paste as much as possible

```
cd ~
curl -L https://git.io/getLatestIstio | sh -
cd istio-0.8.0
```



Variables

```
~ (gke-spinnaker) $ source ~/advanced-kubernetes-workshop/setup/env.sh
Your active configuration is: [cloudshell-11479]
~ (gke-spinnaker) $
```

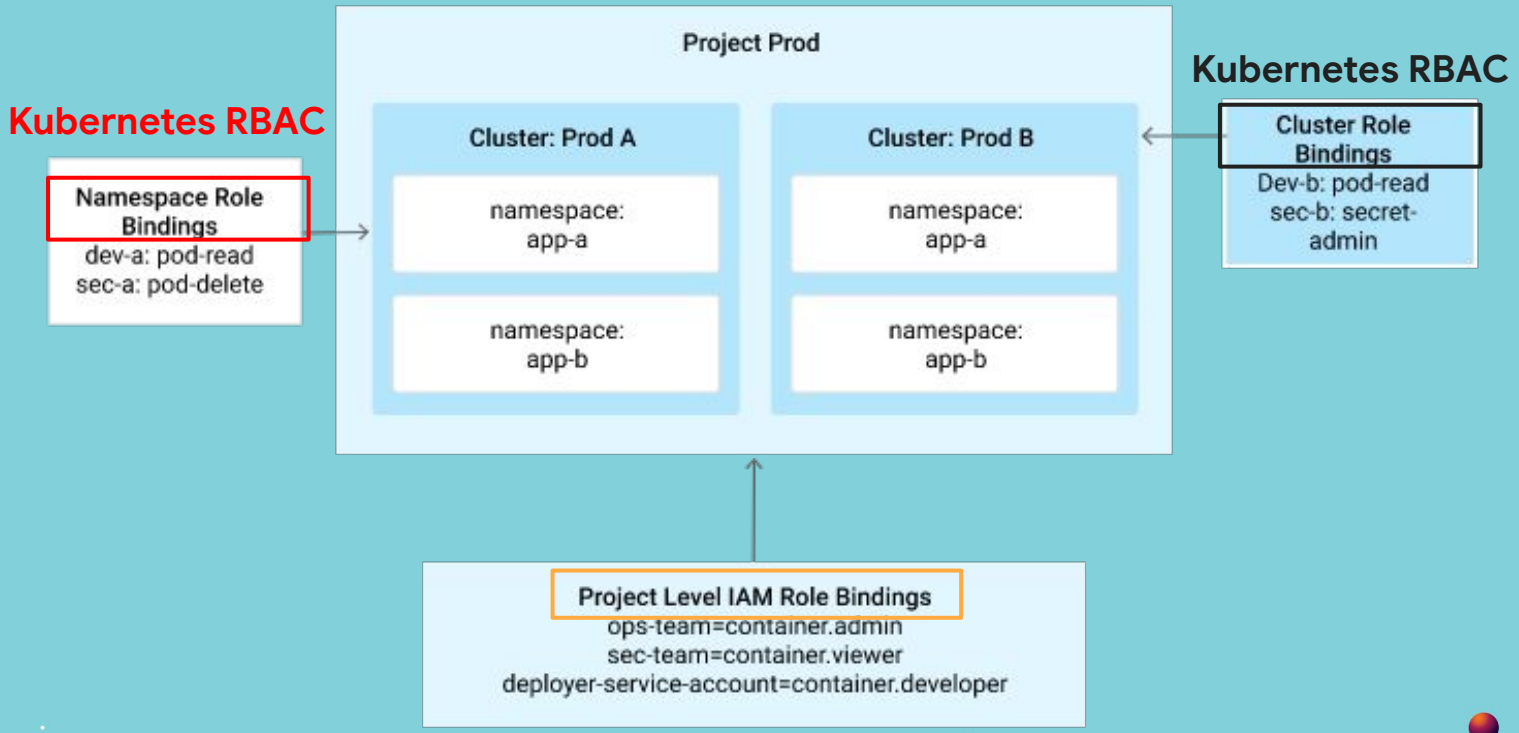
Workshop Helper Commands

```
~ (gke-spinnaker) $ workshop_
workshop_connect            workshop_fortio            workshop_hal-config
workshop_create-pipelines   workshop_get-ingress       workshop_terraform-apply
~ (gke-spinnaker) $ workshop_
```

Helm

Package manager for Kubernetes

Client & Tiller


Kubernetes RBAC
Namespace Role Bindings
dev-a: pod-read
sec-a: pod-delete
Project Prod
Cluster: Prod A
namespace: app-a
namespace: app-b
Cluster: Prod B
namespace: app-a
namespace: app-b
Kubernetes RBAC
Cluster Role Bindings
Dev-b: pod-read
sec-b: secret-admin
Project Level IAM Role Bindings
ops-team=container.admin
sec-team=container.viewer
deployer-service-account=container.developer
GCP IAM

Lab 1: Skaffold

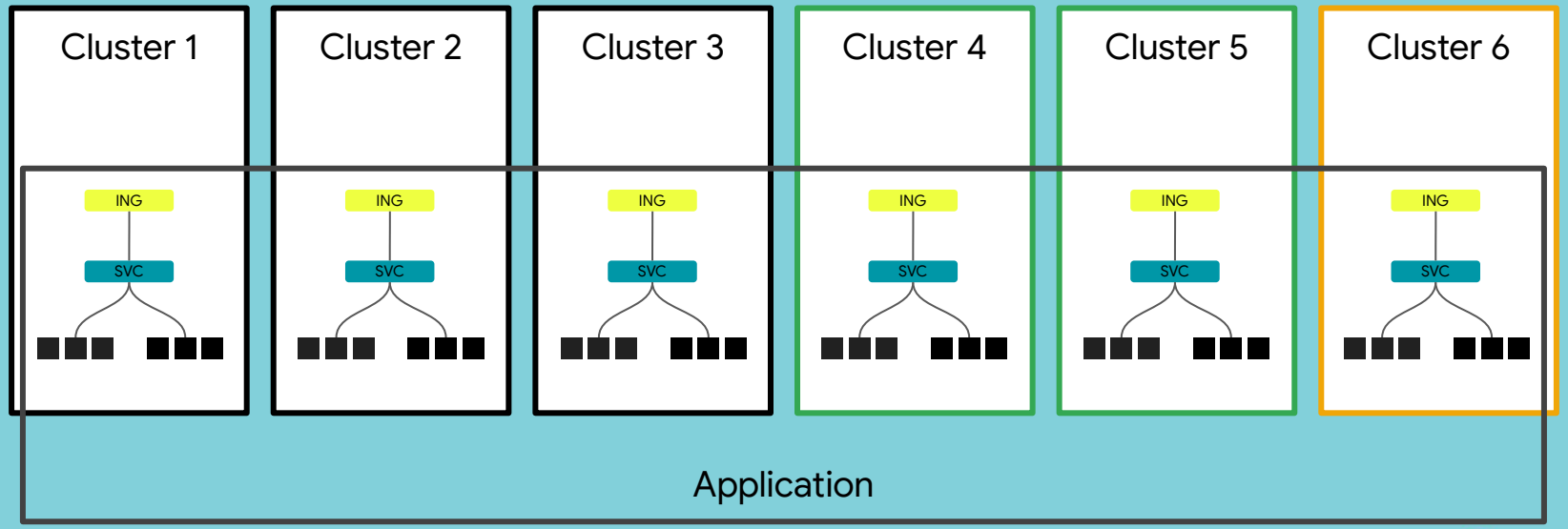
After lab 1 stop and wait for further instructions

04

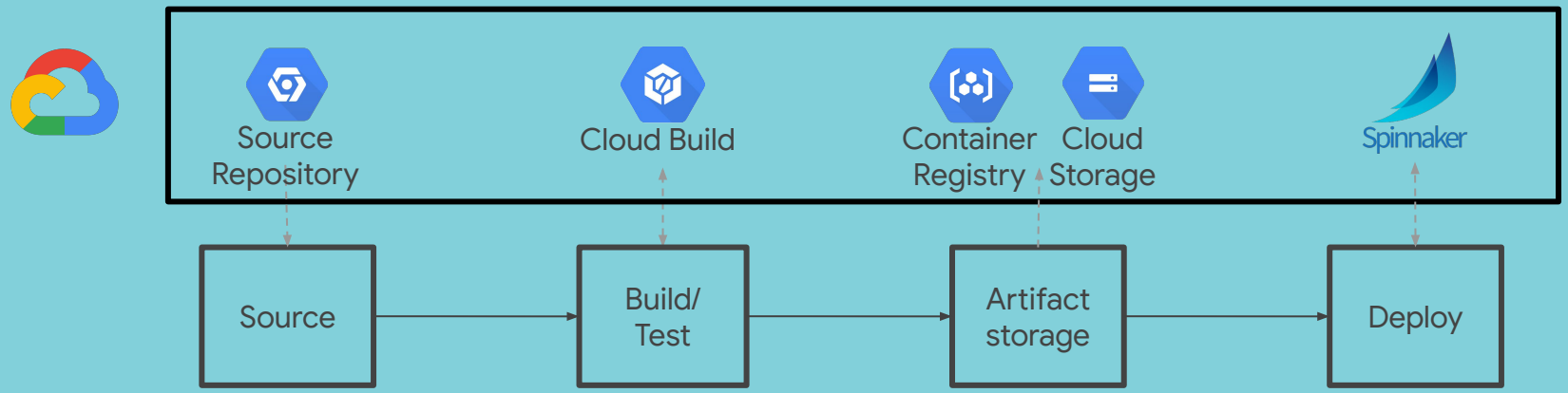
Continuous Delivery with Spinnaker

An Application in Kubernetes

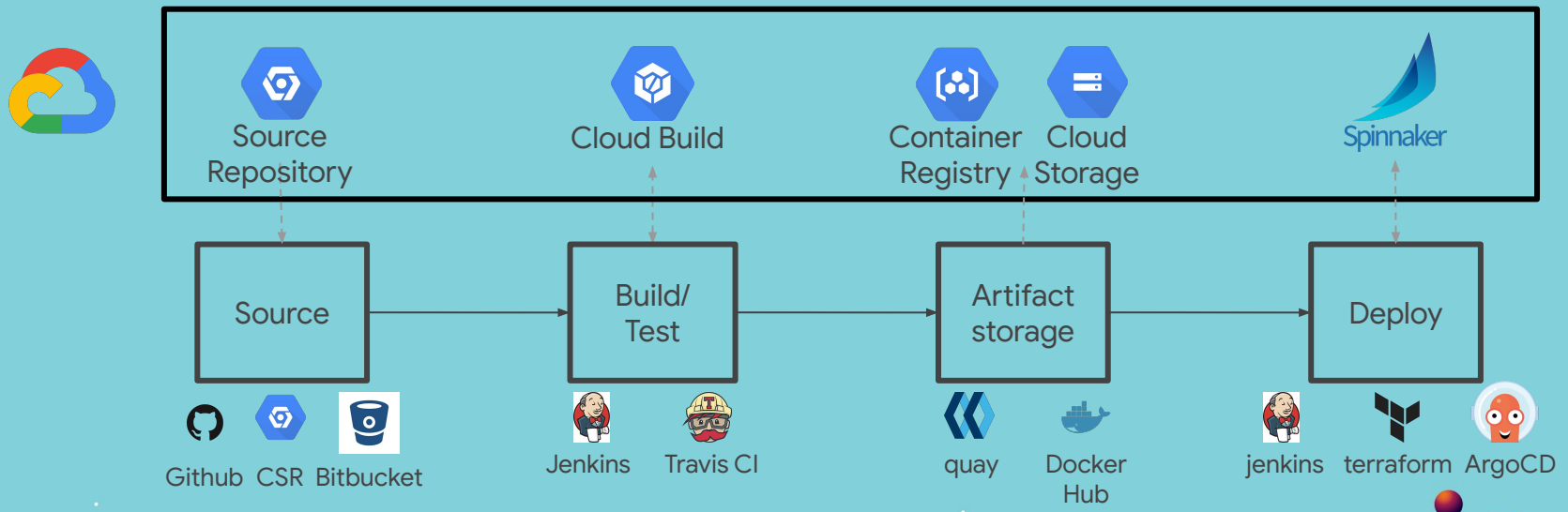
An Application in Kubernetes



CI/CD on Google Cloud



CI/CD on Google Cloud



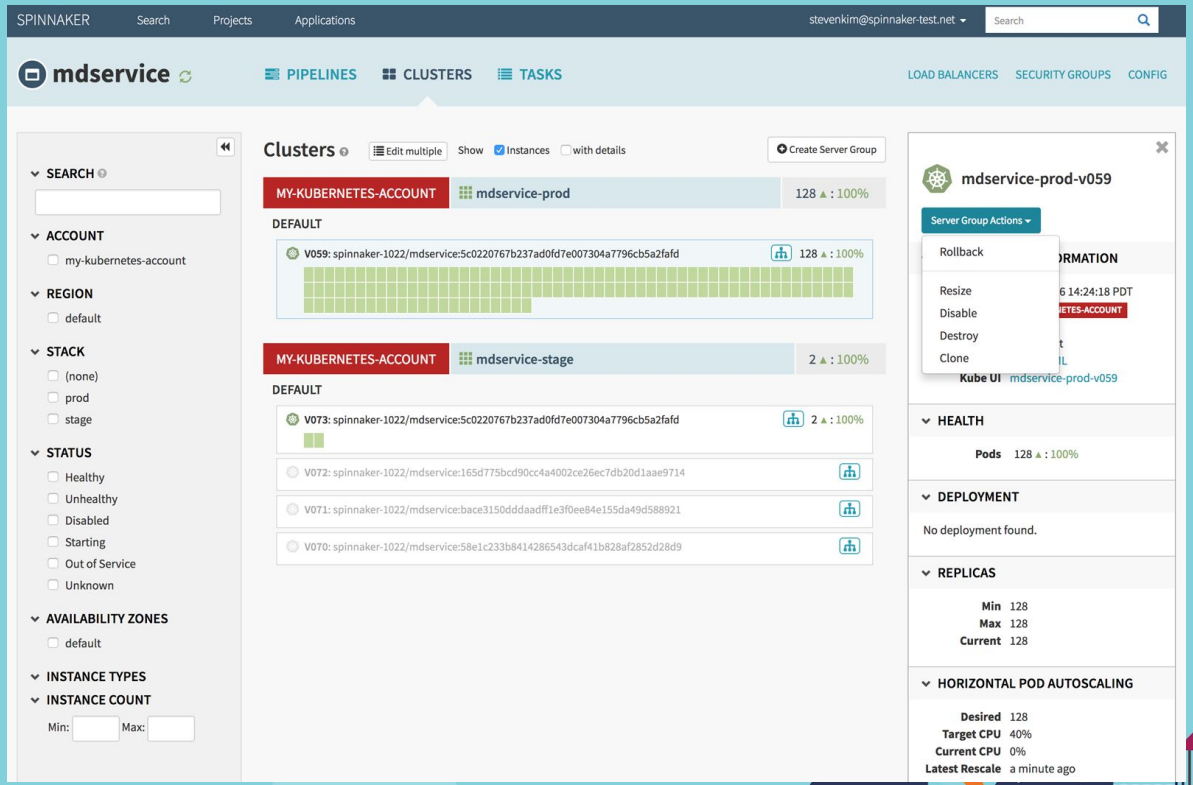
Spinnaker

Spinnaker is an open-source, multi-cloud, ***continuous delivery*** platform

Application management
Workload management

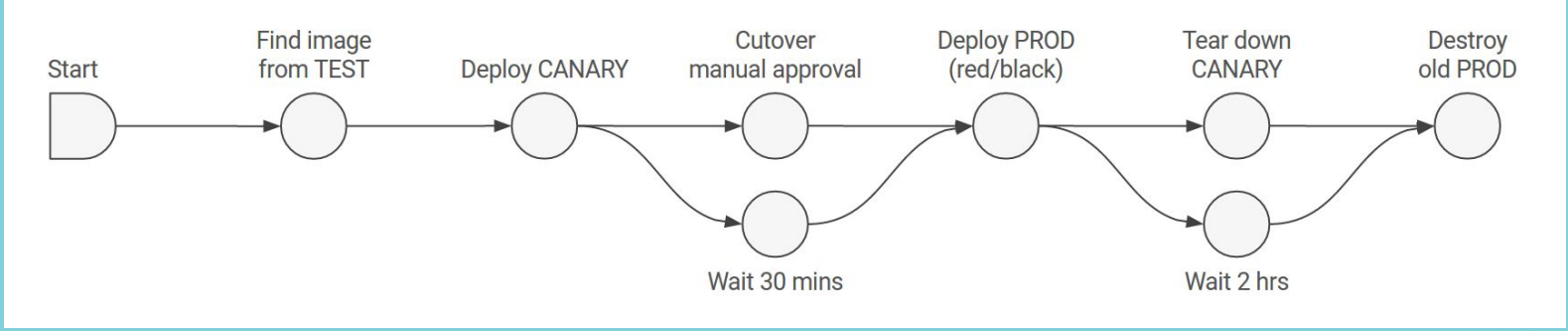
Application Centric Management

- Clusters
- Namespaces
- Services
- Load Balancers



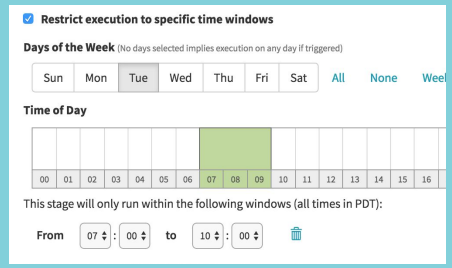
Deployment Management

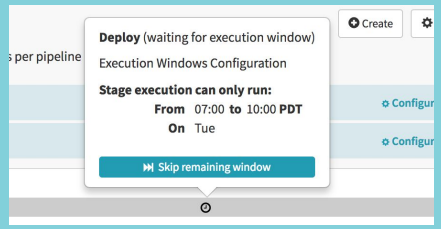
- Pipelines
- Stages



Safe Deployments

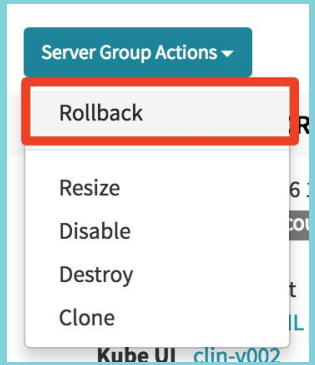
Execution Windows





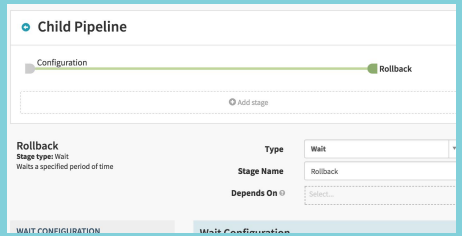
Manual Judgements

Manual Rollbacks



Automated Rollbacks

Trigger a pipeline that does a rollback on a failed deployment



Lab 2: Deploy with Spinnaker
Stop and wait when complete

05

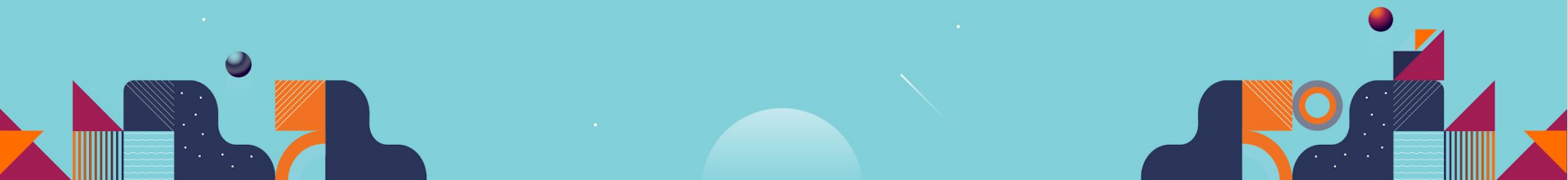
Istio & Service Mesh

Everybody got all fired up about kubernetes and microservices and then were like ‘Oh, s--t, what’s going on?’ Istio gives us a view of our entire system and lets us find trouble spots.

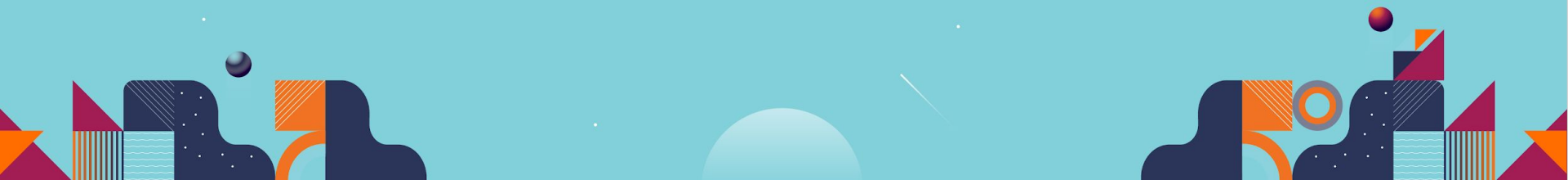
– An early adopter who will remain nameless

Service Mesh

Transparently automate **application network functions**.



Separating applications from network functions



Secure, Monitor, Manage

Intelligent Routing

- Dynamic route configuration
- A/B tests
- Canaries
- Gradually upgrade versions

Resilience

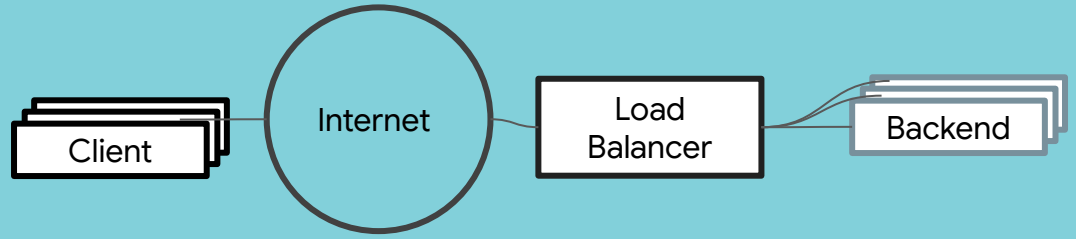
- Timeouts
- Retries
- Health checks
- Circuit breakers

Security & Policy

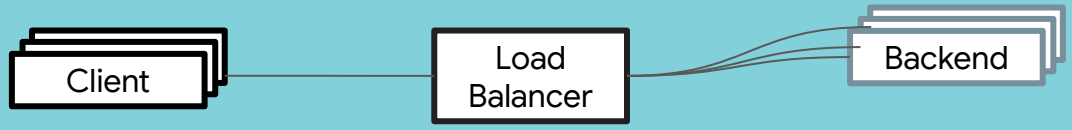
- Mutual TLS
- Organizational policy
- Access policies
- Rate Limiting

Telemetry

- Service Dependencies
- Traffic Flow
- Distributed Tracing

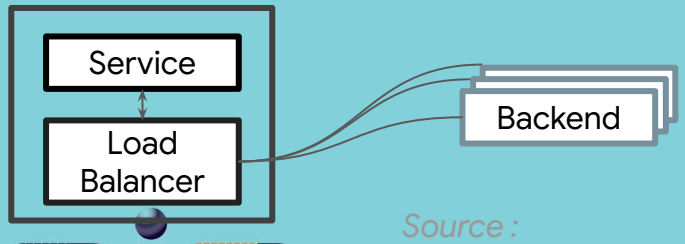


Edge Proxy



Middle Proxy

Embedded Client Library

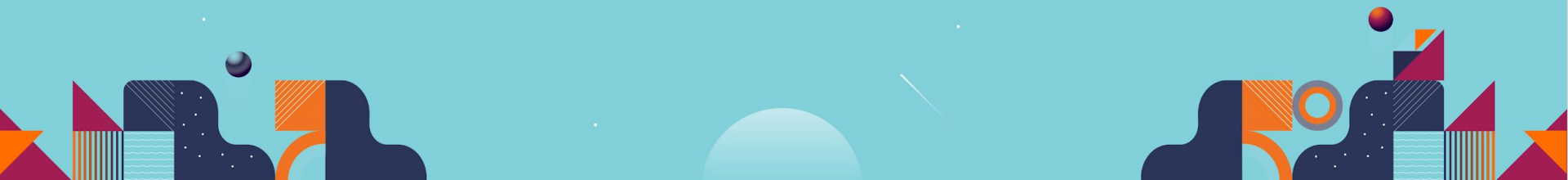


Sidecar Proxy

Source : https://blog.envoyproxy.io/introduction-to-modern-network-load-balancing-and-proxying-a57f6ff80236

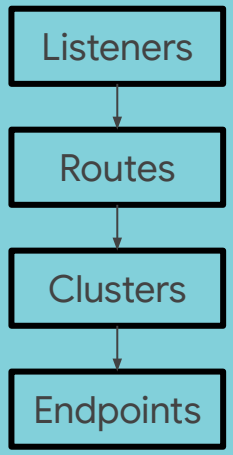
Envoy

- Lightweight L4/L7 Proxy
- Resilient and Scalable
- HTTP/2 and gRPC
- Health checks, circuit breakers, timeouts, retry budgets
- No hot reloads - API driven config updates



Envoy Configuration

Configured with **Aggregated Discovery Services, ADS**



What Envoy listens for (LDS)

Where traffic can be sent (RDS)

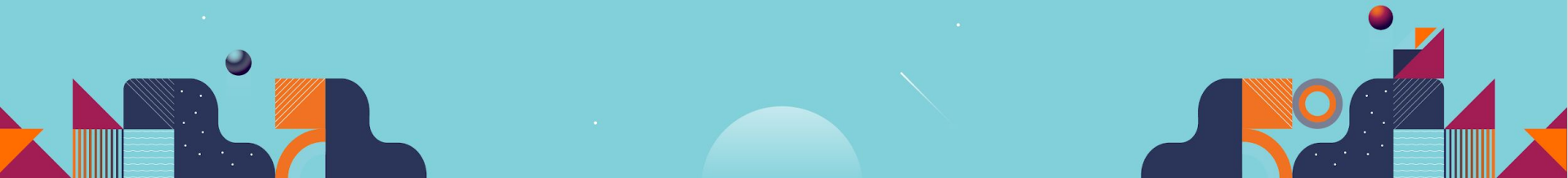
How to send traffic (CDS)

Hosts able to receive traffic (EDS)

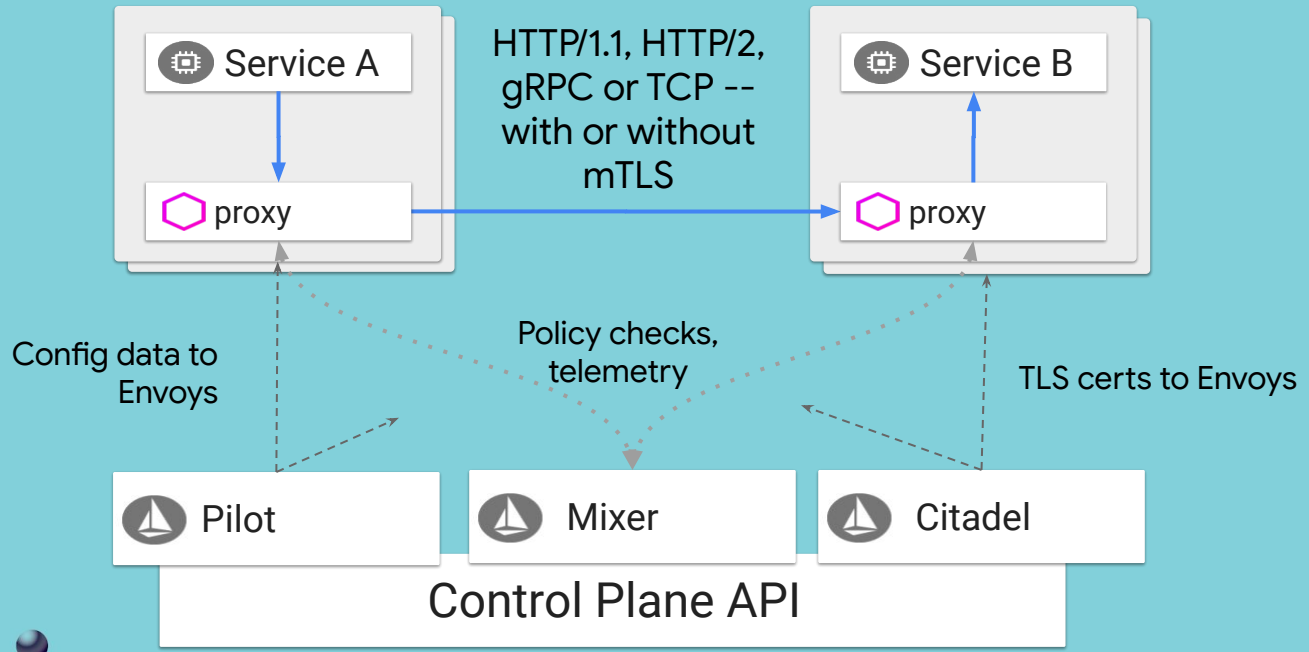

"You now work on 'Istio"'

Istio

- Security
- Observability
- Traffic Management

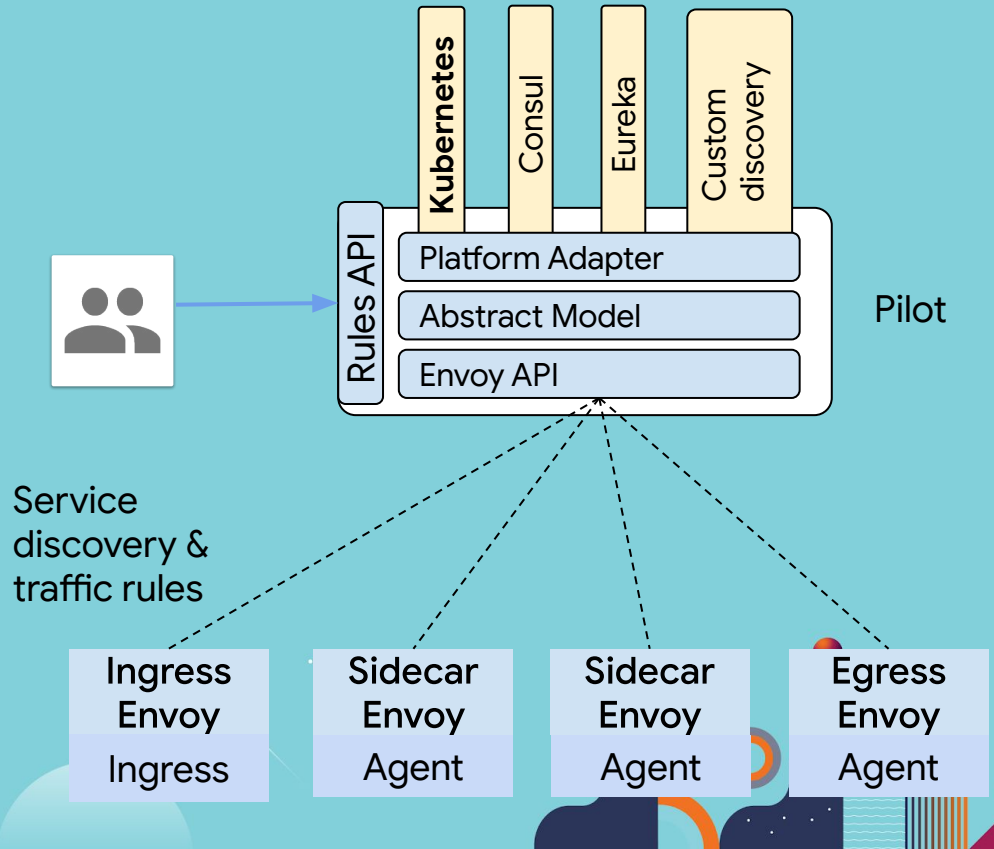


Istio Architecture



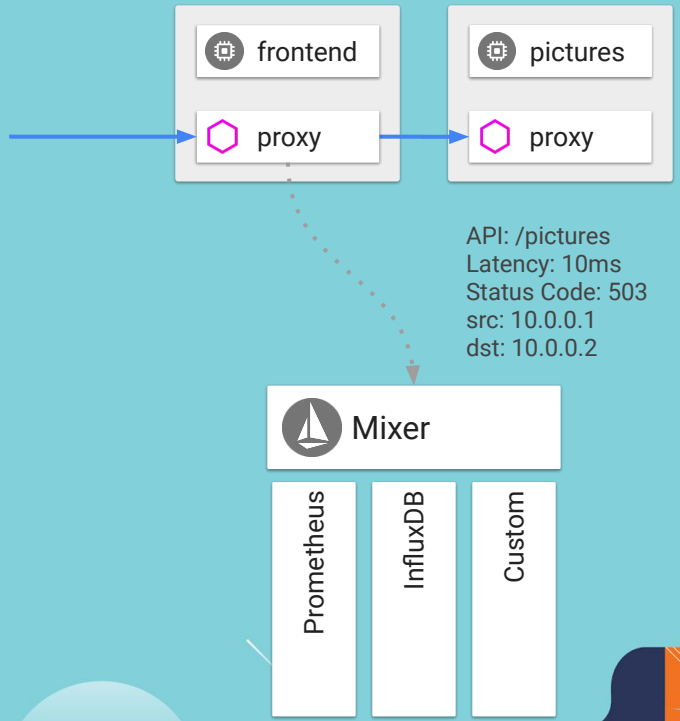
Pilot configures the data plane

- Service Topology
- Routing Rules
- Sidecar / Envoy Configuration

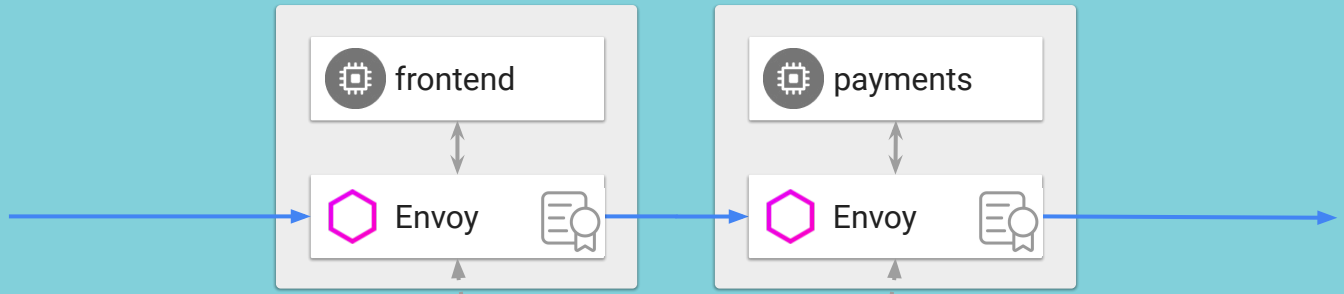


Mixer provides policy and telemetry

- Policy / Quota Enforcement
- Telemetry
- Extensible via Adapters



Citadel provides identity and encryption

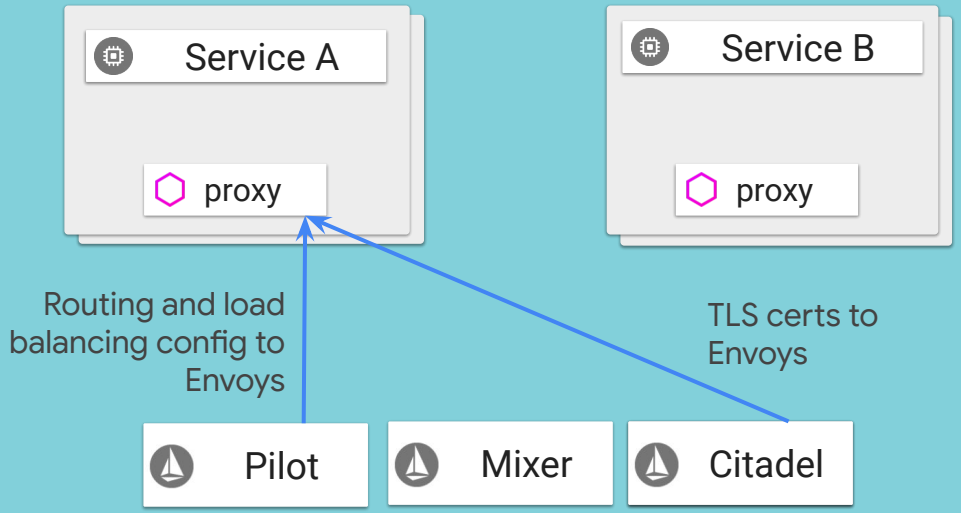


SAN:
"spiffe://example.local/ns/prod/sa/foo"
Namespace: **prod**
Service account: **foo**

SAN:
"spiffe://example.local/ns/prod/sa/bar"
Namespace: **prod**
Service account: **bar**

Citadel

Life of a request in the mesh

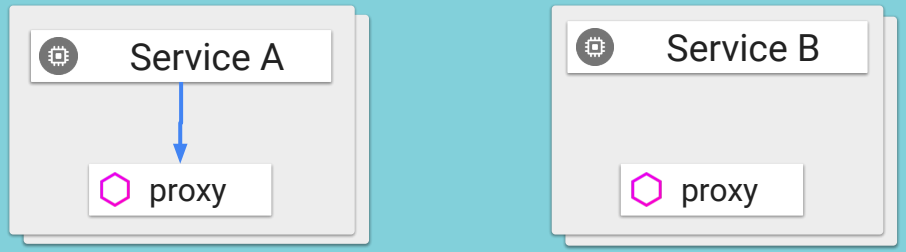


Service A comes up.

Envoy is deployed with it and fetches service information, routing and configuration policy from Pilot.

If Istio Auth is being used, TLS certs are securely distributed as well.

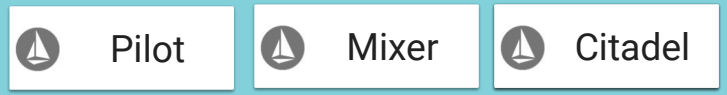
Life of a request in the mesh



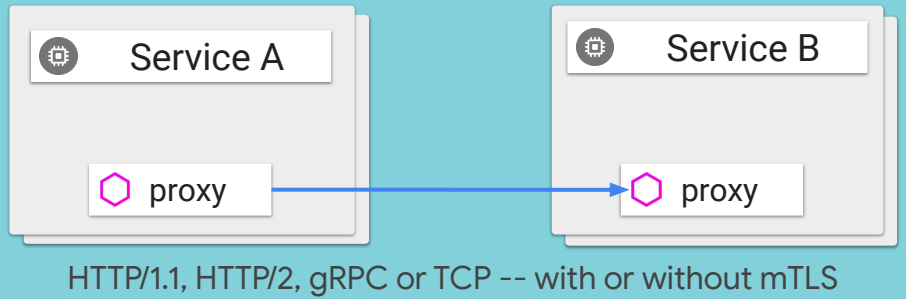
Service A places a call to service B.

Client-side Envoy intercepts the call.

Envoy consults config to know how/where to route call to service B.

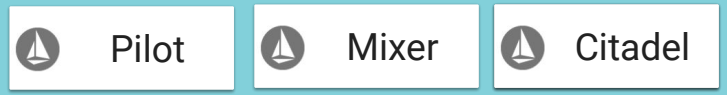


Life of a request in the mesh

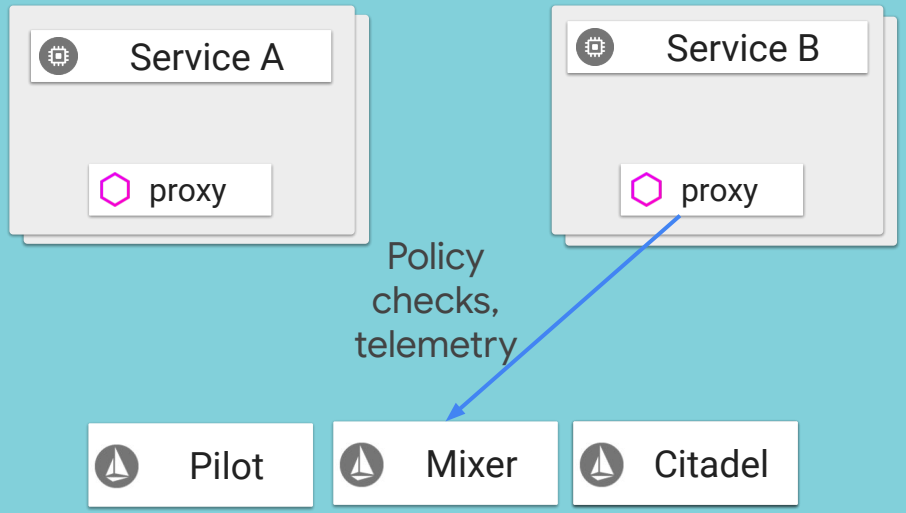


Envoy forwards request to appropriate instance of service B.

There, the Envoy proxy deployed with the service intercepts the call.

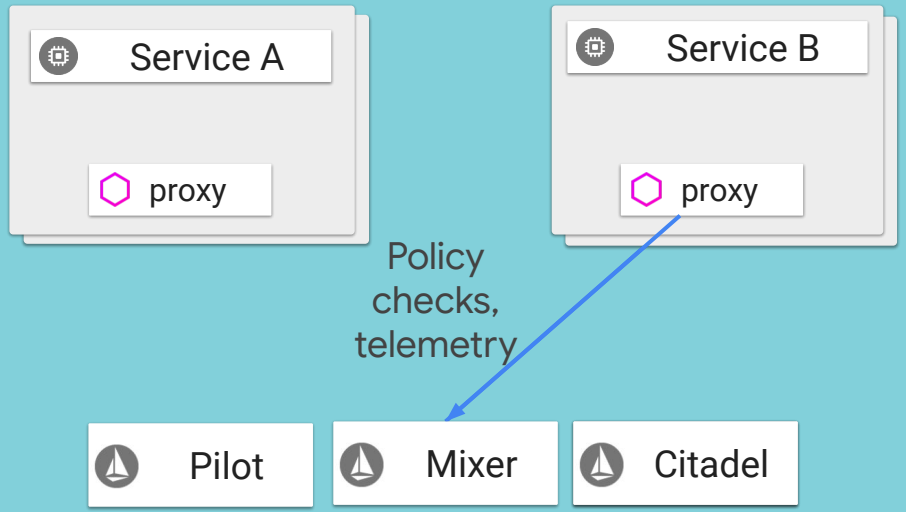


Life of a request in the mesh



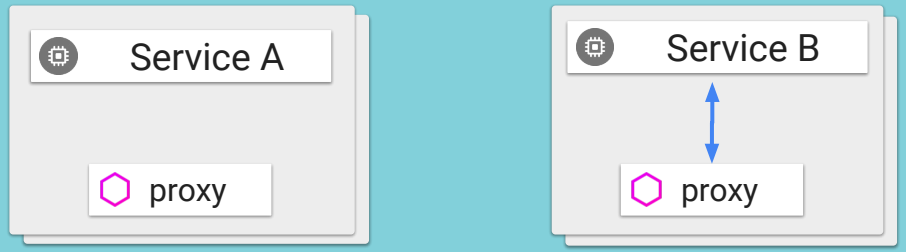
Server-side Envoy checks with Mixer to validate that call should be allowed (ACL check, quota check, etc).

Life of a request in the mesh

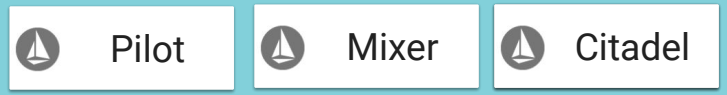


Mixer checks with appropriate adaptors (policy engine, quota adaptor) to verify that the call can proceed and returns True/False to Envoy

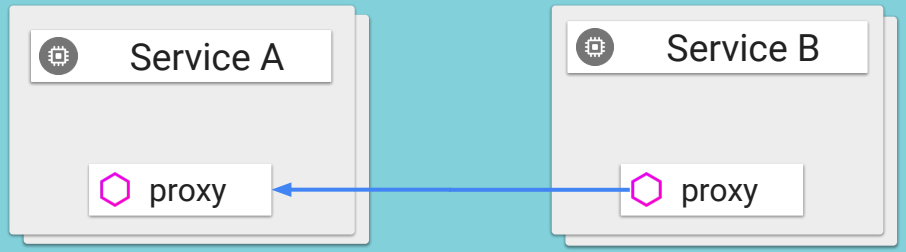
Life of a request in the mesh



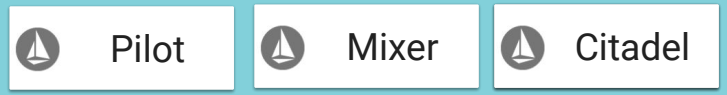
Server-side Envoy forwards request to service B, which process request and returns response



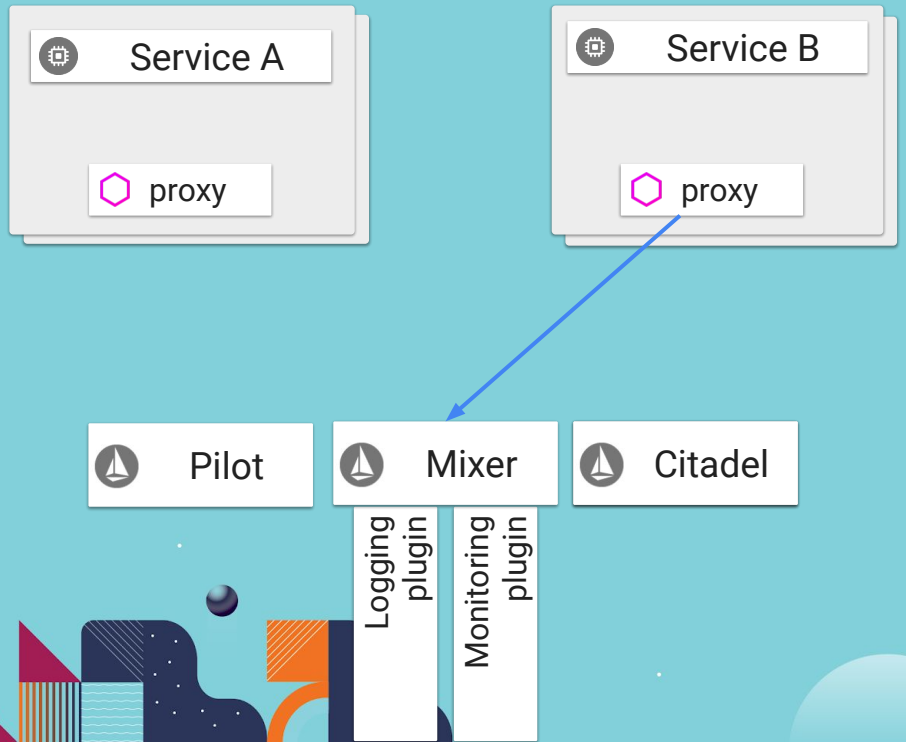
Life of a request in the mesh



Envoy forwards response to the original caller, where response is intercepted by Envoy on the caller side.

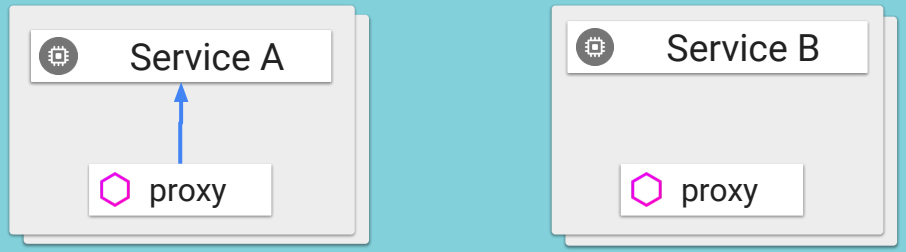


Life of a request in the mesh

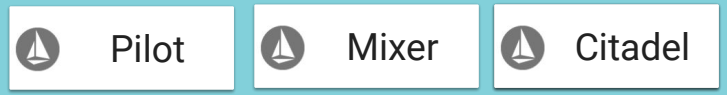


Envoy reports telemetry to Mixer, which in turn notifies appropriate plugins

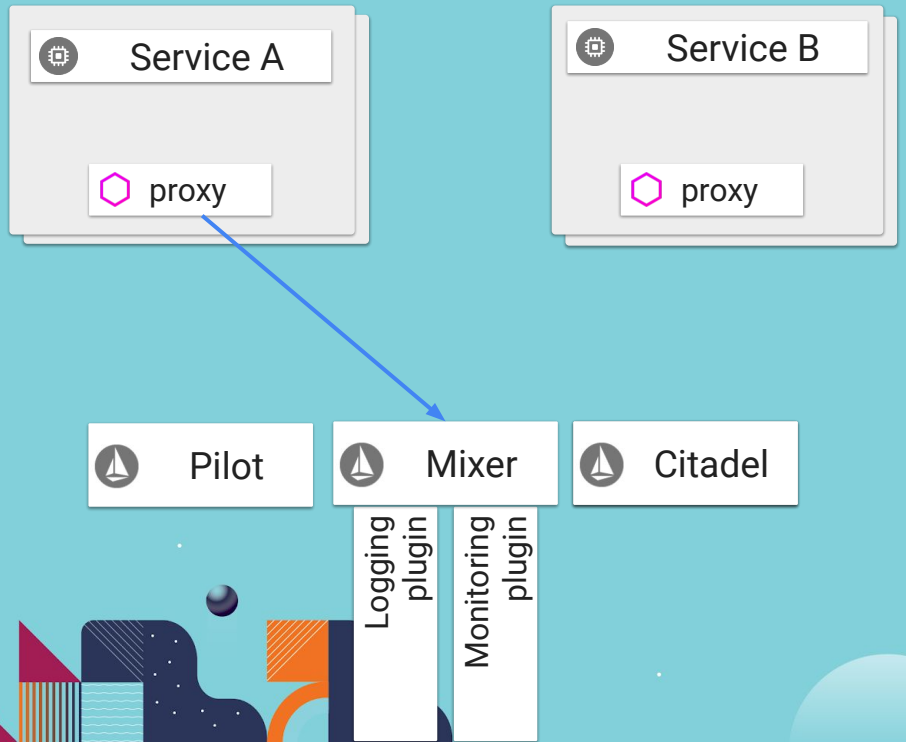
Life of a request in the mesh



Client-side Envoy forwards response to original caller.

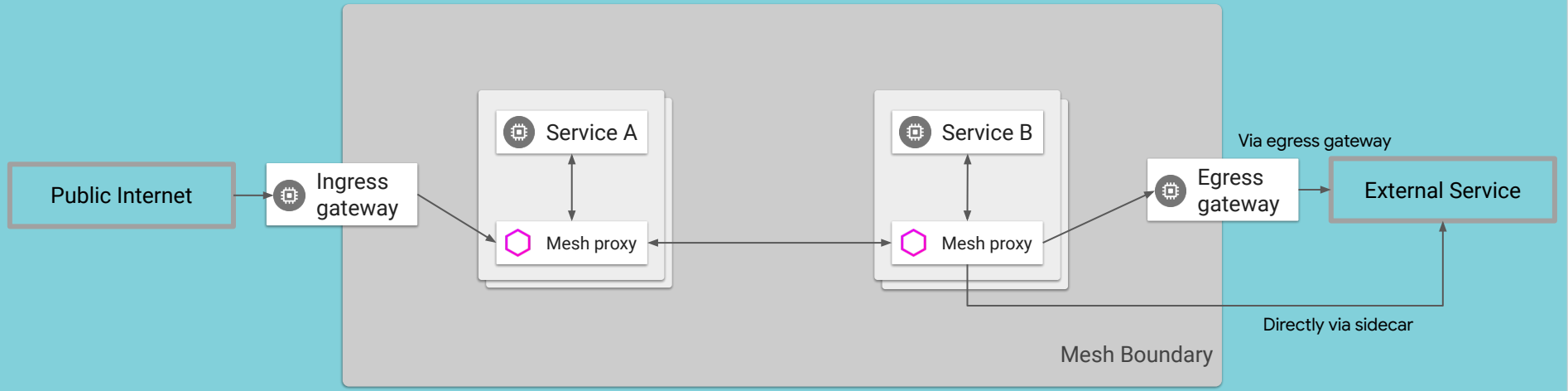


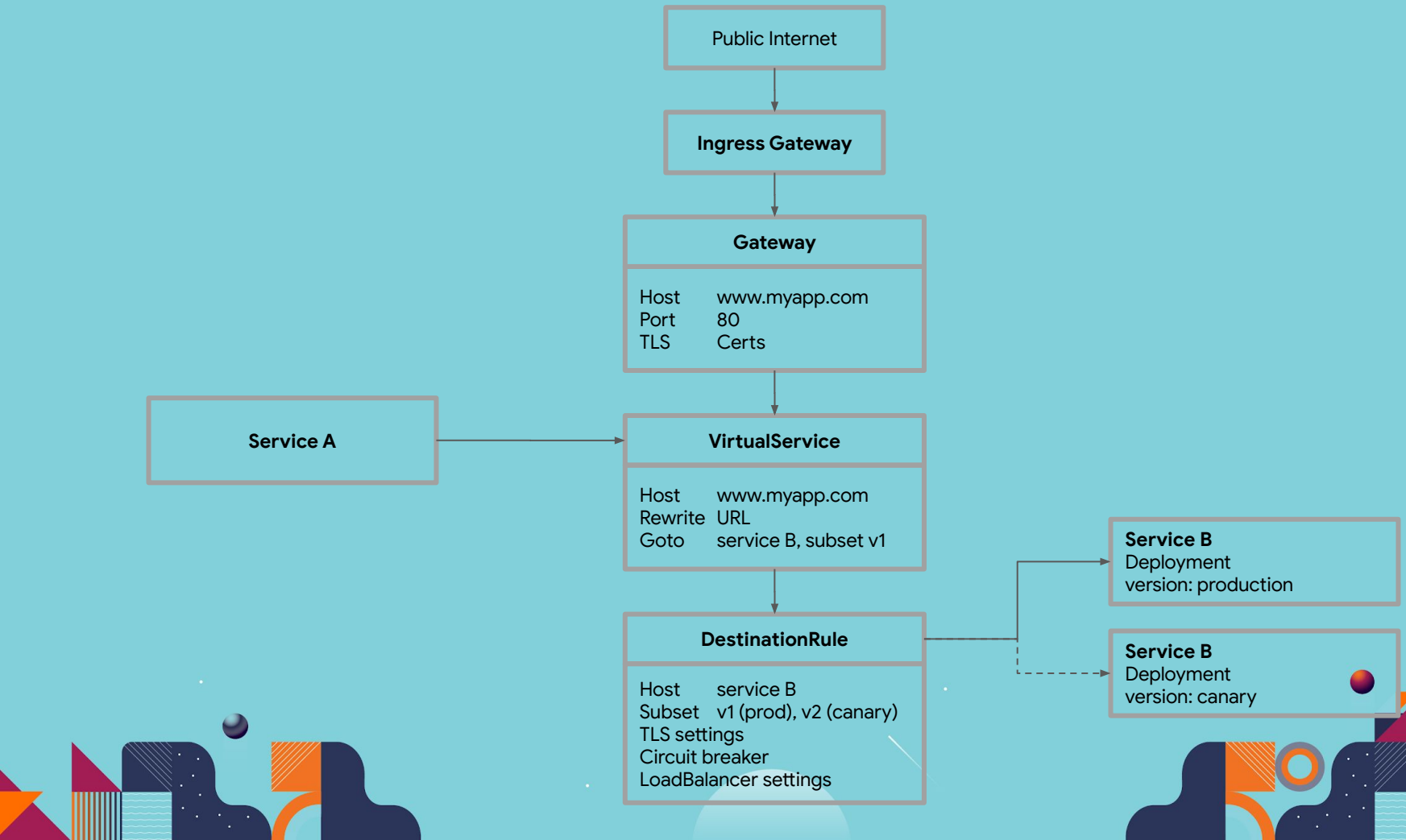
Life of a request in the mesh



Client-side Envoy reports telemetry to Mixer (including client-perceived latency), which in turn notifies appropriate plugins

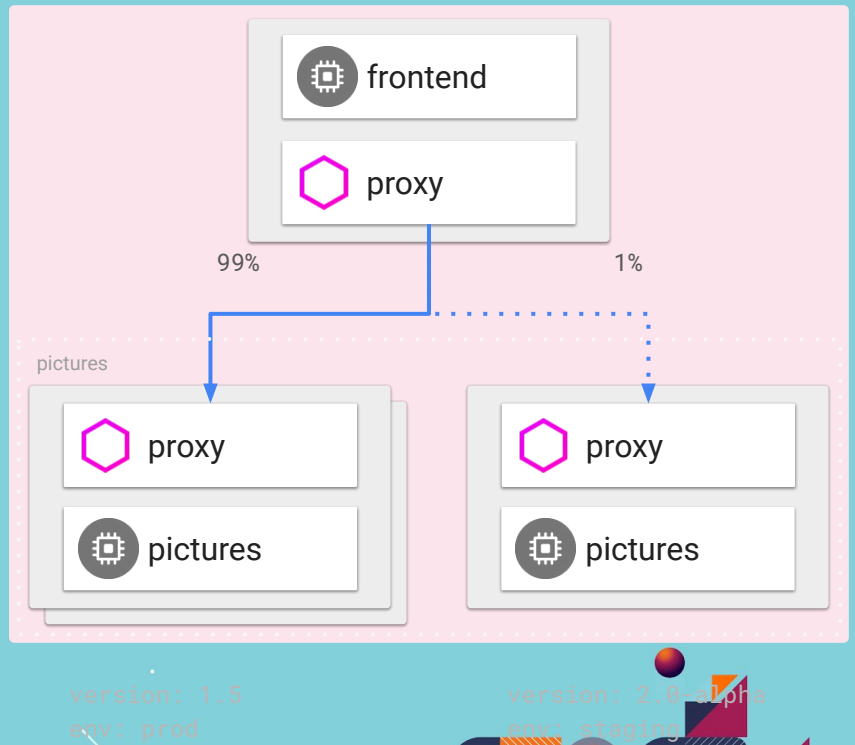
Traffic Management


Public Internet
Ingress gateway
Service A
Mesh proxy
Service B
Mesh proxy
Via egress gateway
Egress gateway
External Service
Directly via sidecar
Mesh Boundary


Public Internet
Ingress Gateway
Gateway
Host www.myapp.com
Port 80
TLS Certs
Service A
VirtualService
Host www.myapp.com
Rewrite URL
Goto service B, subset v1
DestinationRule
Host service B
Subset v1 (prod), v2 (canary)
TLS settings
Circuit breaker
LoadBalancer settings
Service B
Deployment
version: production
Service B
Deployment
version: canary

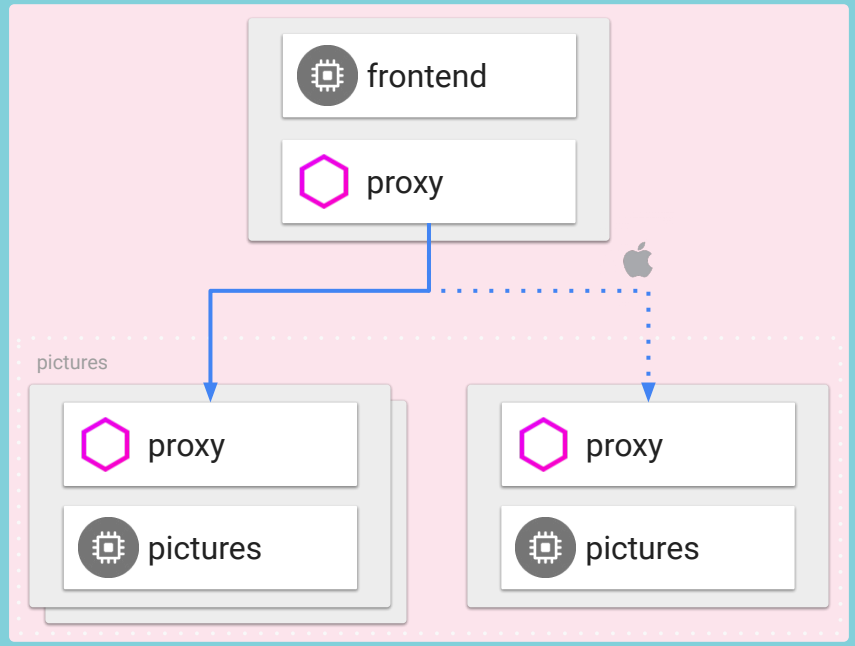
Application rollout

```
// A simple traffic splitting rule
destination: pictures.example.local
match:
  source: frontend.example.local
route:
- tags:
    version: v1.5
    env: prod
  weight: 99
- tags:
    version: v2.0-alpha
    env: staging
  weight: 1
```



Traffic steering

```
// Content-based traffic steering rule
destination: pictures.example.local
match:
  httpHeaders:
    user-agent:
      regex: ^(.*?;)?(iPhone)(;.*)?$
precedence: 2
route:
- tags:
    version: 2.0-alpha
    env: staging
```



version: 1.5
env: prod

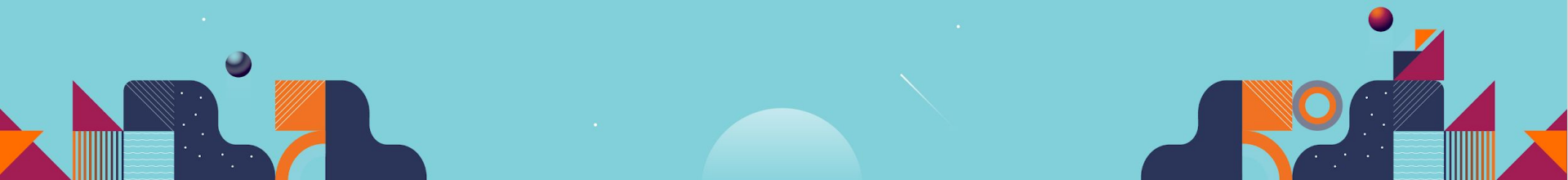
version:
2.0-alpha env:
staging

Resiliency

```
// Circuit breakers
destination: auth.cluster.local
policy:
- tags:
    version: v1
  circuitBreaker:
    simpleCb:
      httpConsecutiveErrors: 7
      sleepWindow: 5m
      httpDetectionInterval: 1m
```

Resiliency features

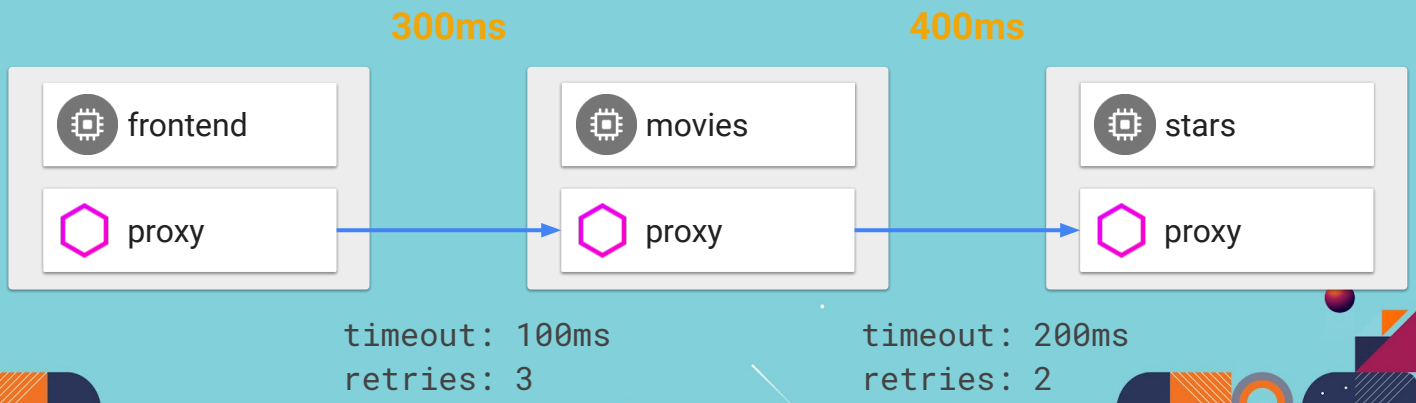
- Timeouts
- Retries with timeout budget
- Circuit breakers
- Health checks
- AZ-aware load balancing w/ automatic failover
- Control connection pool size and request load
- Systematic fault injection



Resiliency testing

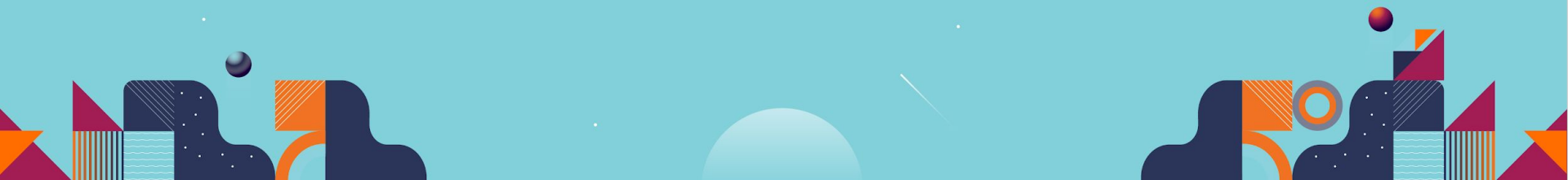
Systematic fault injection to identify weaknesses in failure recovery policies

- HTTP/gRPC error codes
- Delay injection



Efficiency

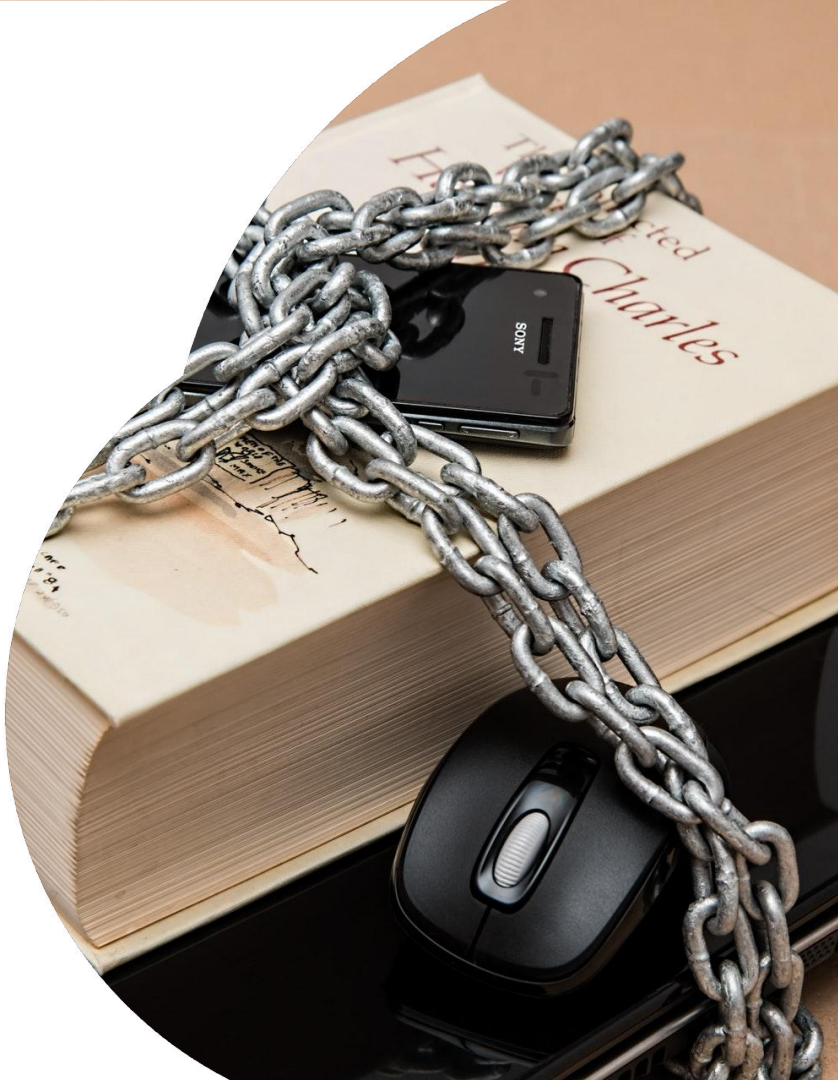
- L7 load balancing
  - Passive/Active health checks, circuit breaks
  - Backend subsets
  - Affinity
- TLS offload
  - No more JSSE or stale SSL versions
- HTTP/2 and gRPC proxying



Security

Security

- Verifiable identity
- Secure naming / addressing
- Traffic encryption
- Revocation



Lab 3: Control and secure with Istio
After lab 3 stop and wait for further instructions


Observability

Observability

Monitoring & tracing should not be an afterthought in the infrastructure

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

Istio Dashboard

Zoom Out | Last 5 minutes | Refresh every 5s

Source: All | Source Version: All | HTTP Destination: All | TCP Destination: All | Destination Version: All | Istio.io

Istio

Istio is an open platform that provides a uniform way to connect, manage, and secure microservices. Need help? Join the Istio community.

Global Request Volume

35 ops

Global Success Rate (non-5xx responses)

84.9%

4xxs

N/A

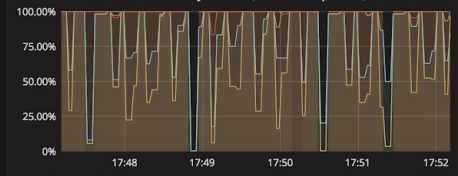
5xxs

5.2 ops

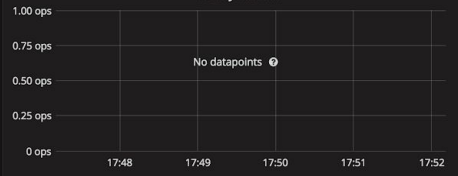
Service Mesh

Request Volume

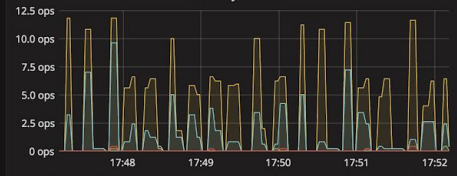
Success Rate by Service (non-5xx responses)



4xxs by Service



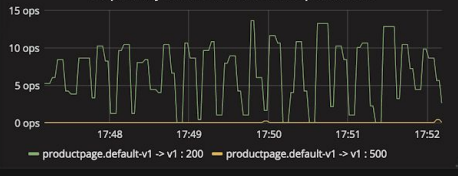
5xxs by Service



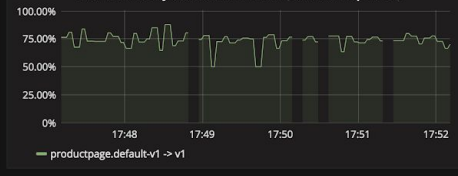
HTTP Services

details.default.svc.cluster.local

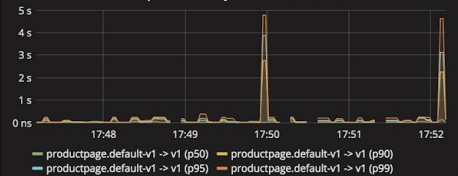
Requests by Source, Version, and Response Code



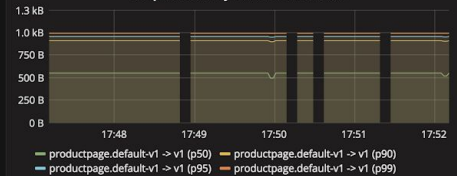
Success Rate by Source and Version (non-5xx responses)



Response Time by Source and Version

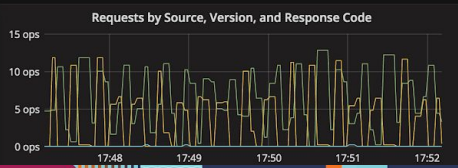


Response Size by Source and Version

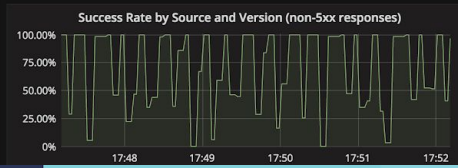


ingress.istio-system.svc.cluster.local

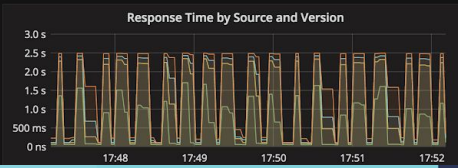
Requests by Source, Version, and Response Code



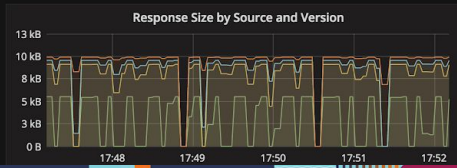
Success Rate by Source and Version (non-5xx responses)



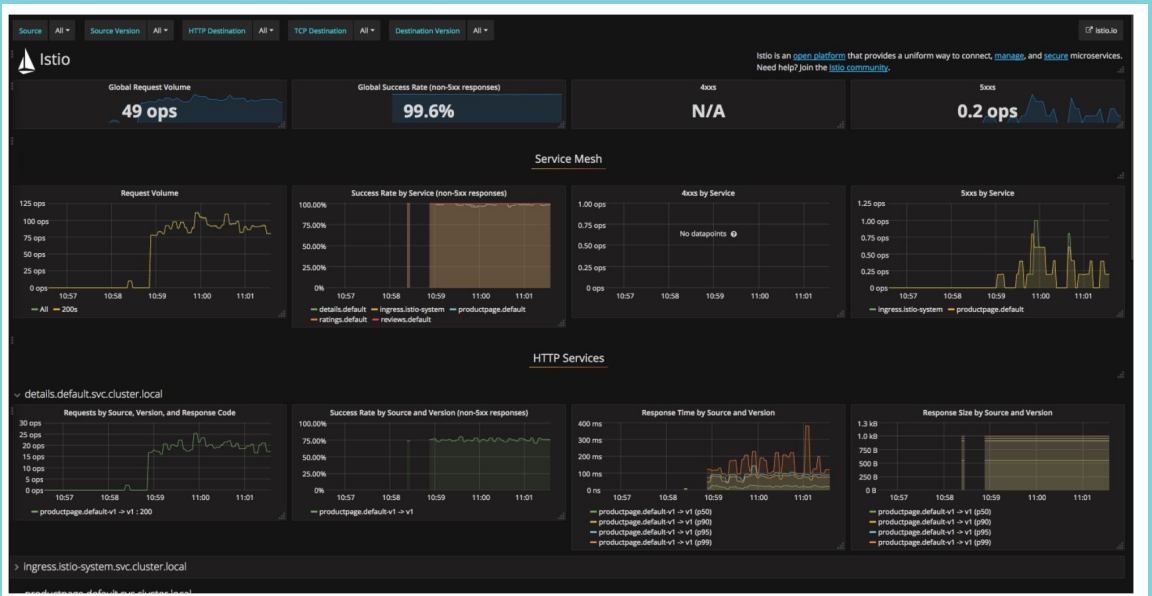
Response Time by Source and Version



Response Size by Source and Version



Traffic management and operational agility



Istio Dashboard With Traffic

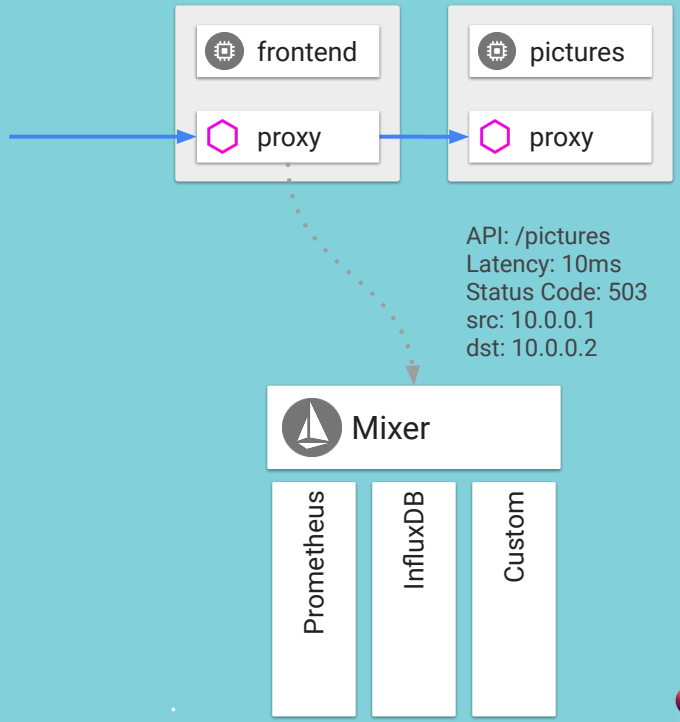
1. Direct traffic away from starving instances
2. **Scale** by directing traffic to multiple versions
3. **Roll out new versions** without worrying about ops challenges
4. Apply access control, rate limiting policies to **protect services** from bad behavior

Metrics flow

Mixer collects metrics emitted by Envoys

Adapters in the Mixer normalize and forward to monitoring backends

Metrics backend can be swapped at runtime



Tracing

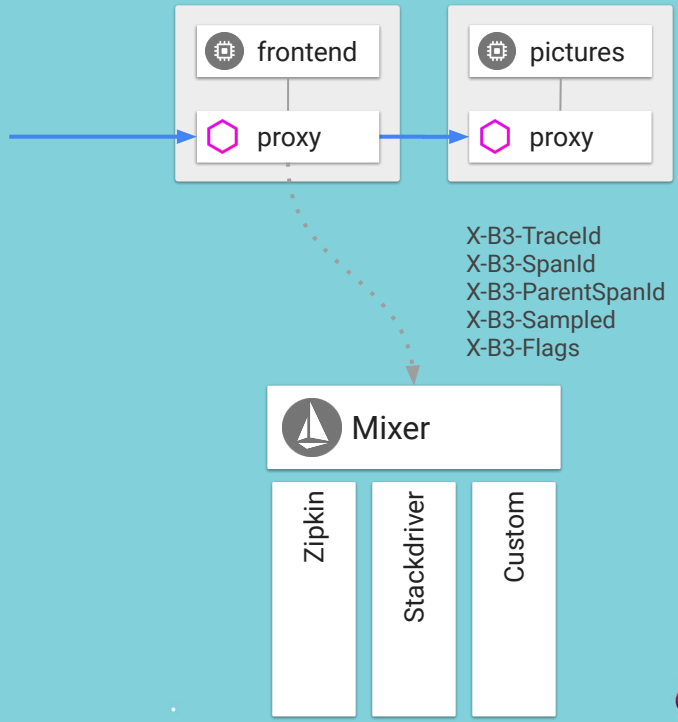
Applications do not have to deal with generating spans or correlating causality

Envoys generate spans

Applications need to forward context headers on outbound calls

Envoys send traces to Mixer

Adapters at Mixer send traces to respective backends



Lab 4: Monitor and troubleshoot with Istio
After lab 4 stop and wait for further instructions


Thank you. Please take our survey

https://hubs.ly/H0kC0320

Fairwinds