ORACLE

Cloud Native Labs

Microservices at Scale

Next Steps in Kubernetes with Service Mesh

Jesse Butler Cloud Native Advocate, Oracle Cloud Infrastructure. @jlb13

#OracleCloudNative

cloudnative.oracle.com

Level Set

- Microservices
- Kubernetes
- Service Mesh

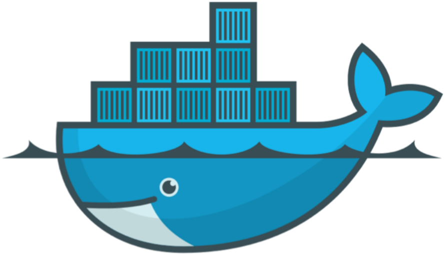
Let’s Talk About Service Mesh

A service mesh that allows us to connect, secure, control and observe services at scale, often requiring no service code modification

Though other options exist, Linkerd and Istio are the two contenders to choose from

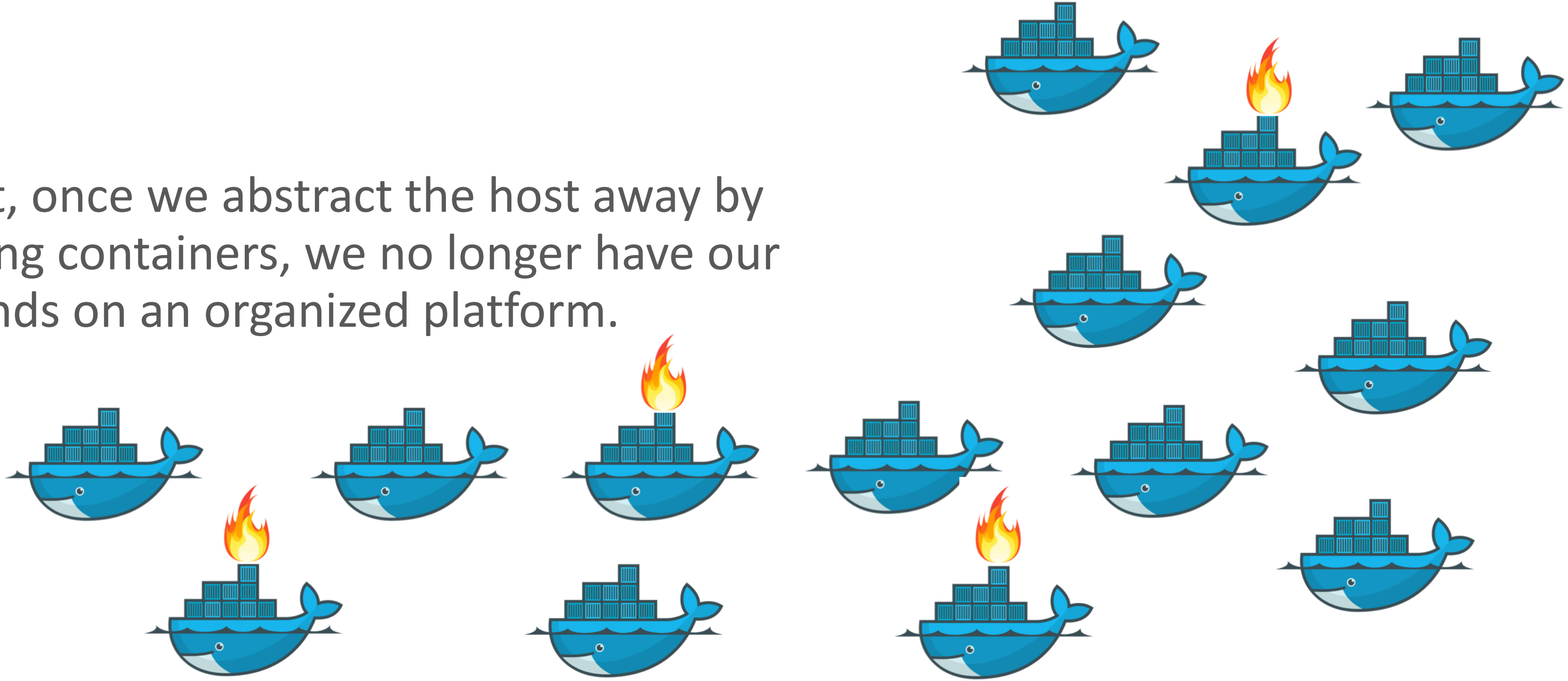
Docker

- Docker changed the way we build and ship software
- Application and host are decoupled, making application services portable
- Containers are an implementation detail, but a critical one



Docker Is a Start

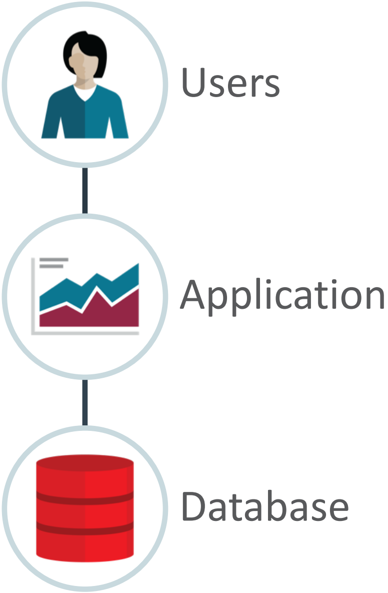
But, once we abstract the host away by using containers, we no longer have our hands on an organized platform.



Kubernetes

Kubernetes to the rescue. It provides the abstractions and organization we need for deploying containers at scale

Migration from the Old World…



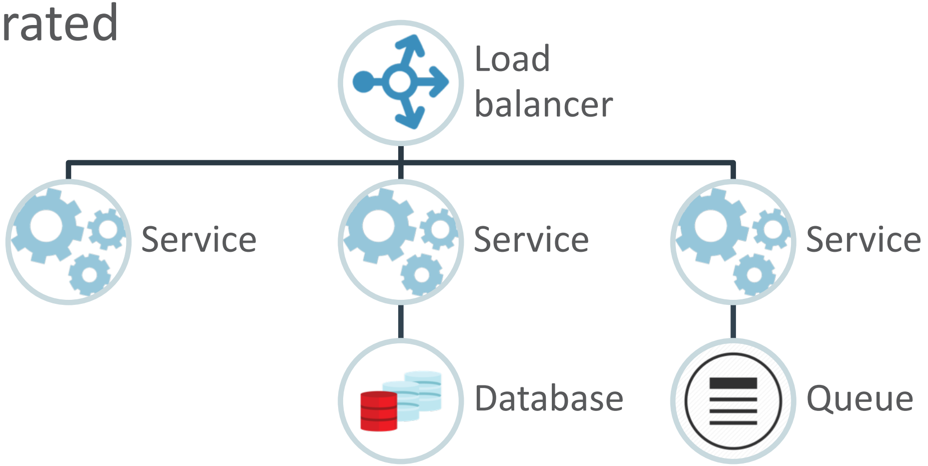
Users

Application

Database

…to Cloud Native Kubernetes Hotness

- Microservices running in orchestrated containers
- Everybody's happy
- What happens now?



Day Two

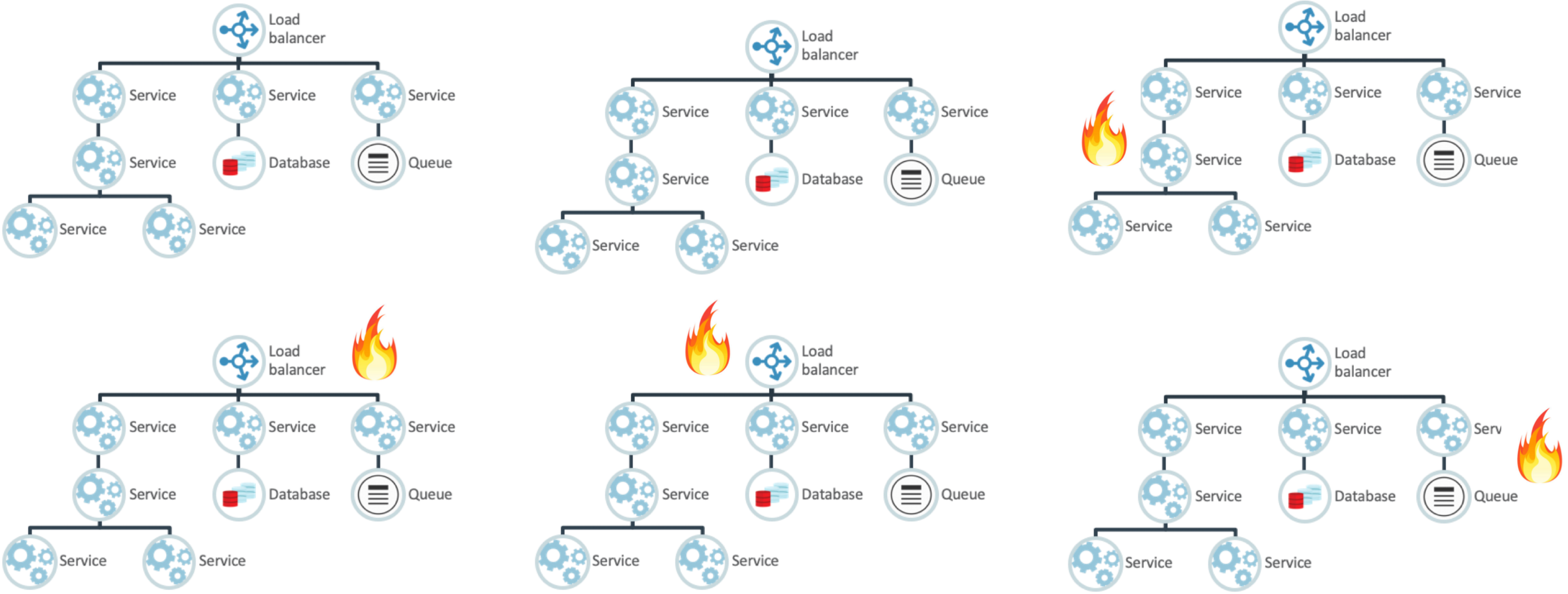

Load balancer
Service
Service
Service
Service
Database
Queue
Service
Service
Load balancer
Service
Service
Service
Service
Database
Queue
Service
Service
Load balancer
Service
Service
Service
Service
Database
Queue
Service
Service
Load balancer
Service
Service
Service
Service
Database
Queue
Service
Service
Load balancer
Service
Service
Service
Service
Database
Queue
Service
Service
Load balancer
Service
Service
Serv
Service
Database
Queue
Service
Service

Table Stakes for Services at Cloud Scale

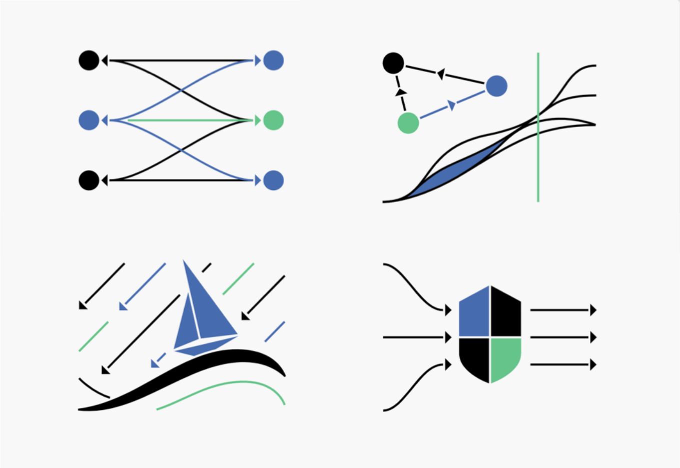
- We require a method to simply and repeatably deploy software, and simply and recoverably modify deployments
- We require telemetry, observability, and diagnosability for our software if we hope to run at cloud scale

Day 2 Solutions

- Ingress and Traffic Management
- Tracing and Observability
- Metrics and Analytics
- Identity and Security

Abstract Requirements

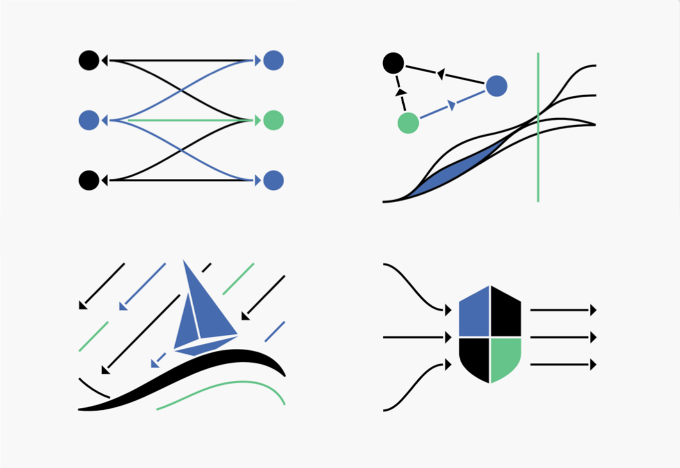
- Traffic Management
- Observability
- Security
- Policy



Hard Things are Hard

These are Hard Problems™, and some software may address one of them well.

Service mesh attempts to address them all.



What Is a Service Mesh?

- Infrastructure layer for controlling and monitoring service-to-service traffic
- Greatly simplifies service implementation via service discovery, automated retries, timeouts and more
- A data plane deployed alongside application services, and a control plane used to manage the mesh

Service Mesh is Not an API Gateway

API Gateways deal with north-south traffic, inbound to your cluster

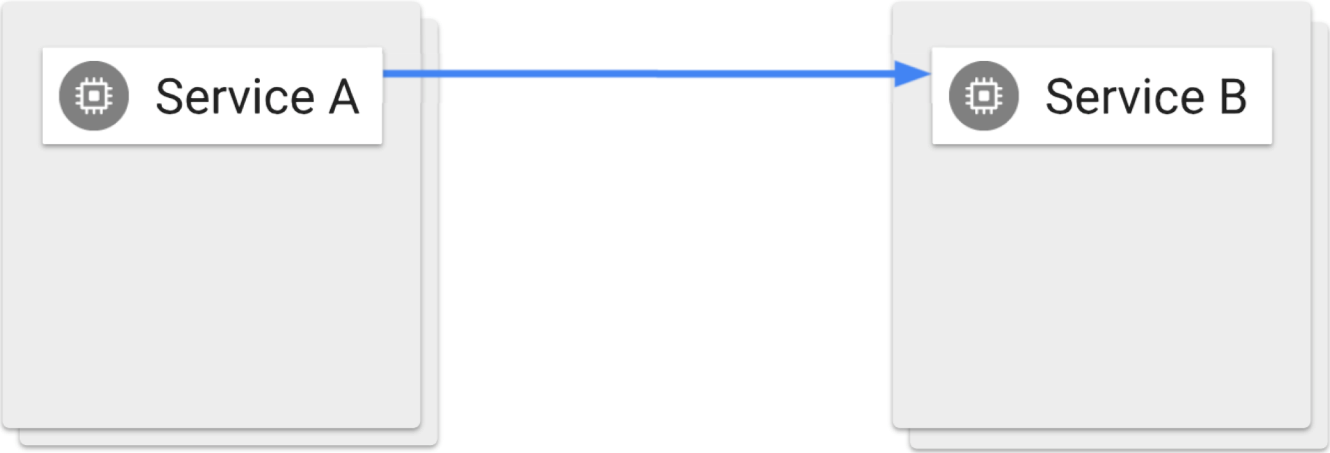
Service Mesh is concerned with east-west traffic, between your services within your cluster

Though Istio does have an ingress gateway…

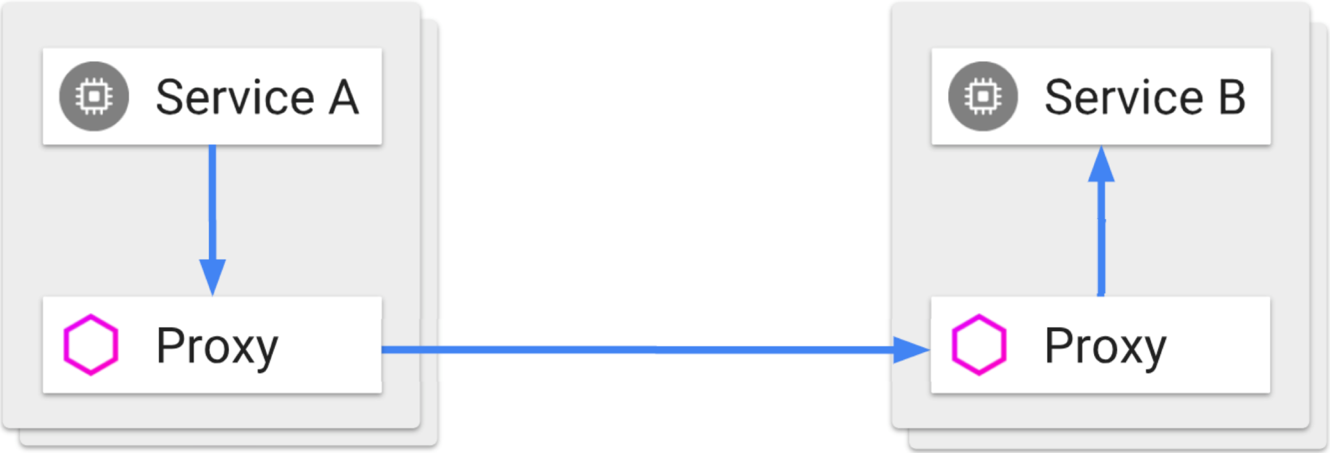
Service Mesh Architecture

- Both Istio and Linkerd use a proxy on the data plane to create the mesh
- Each service you add to the mesh has a proxy injected into its pod
- This vantagepoint is what gives a service mesh its power – it sees and knows all

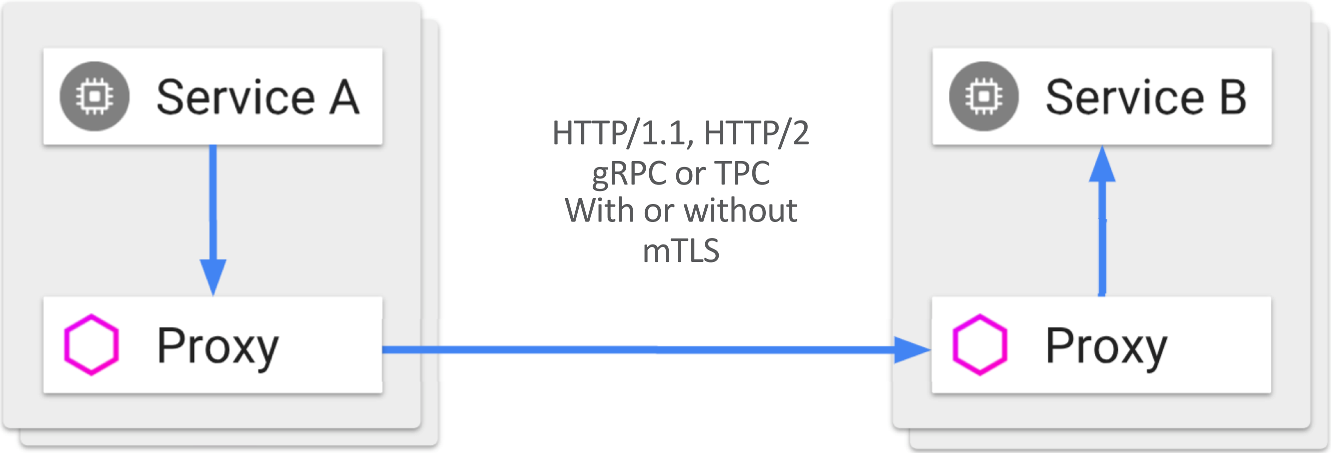
Sidecar Proxy



Sidecar Proxy



Sidecar Proxy



Let’s Look at Linkerd

Linkerd is an ultralight service mesh for Kubernetes and other orchestration platforms

Linkerd2 has a wholly reimplemented proxy and is built for low latency and massive scaling

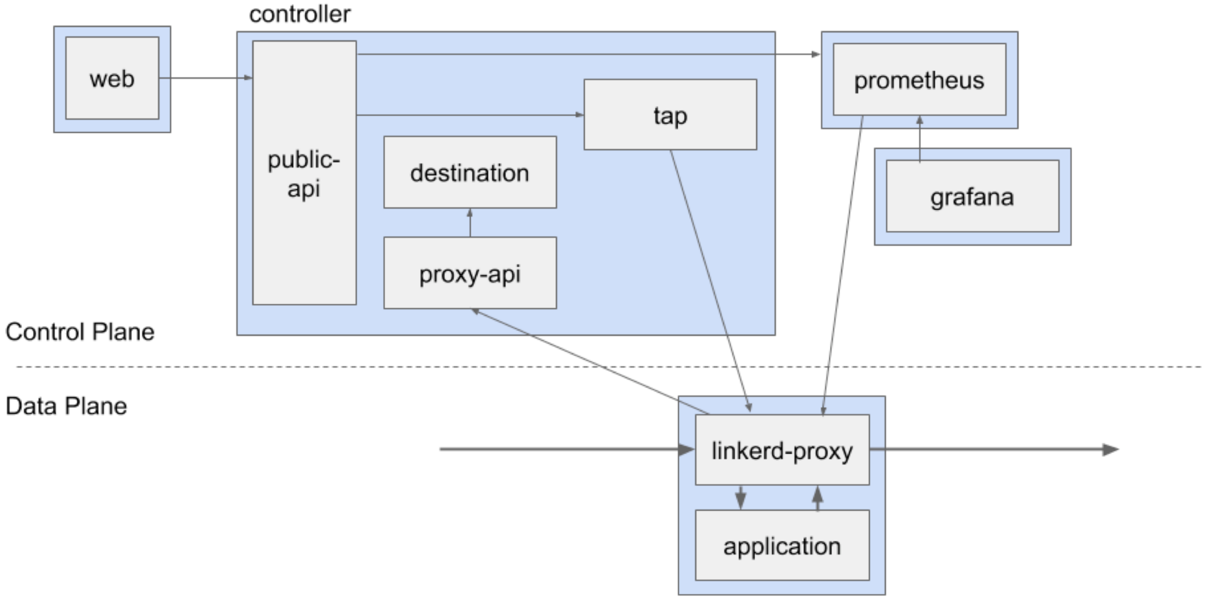
Linkerd Features

- Deep runtime diagnostics
  - Comprehensive suite of diagnostic tools, including automatic service dependency maps and live traffic samples
- Actionable service metrics
  - Allows you to monitor *golden metrics*—success rate, request volume, and latency—for every service and define response

Linkerd Features

- Simple, minimalist design
  - No complex APIs or configuration. For most applications, Linkerd will "just work" out of the box
- Ultralight and ultra fast
  - Built in Rust, Linkerd's data plane proxies are incredibly small (<10 mb) and blazing fast (p99 < 1ms)

Linkerd Components



Using Linkerd

- Linkerd CLI utilities
  - Routes, stats, tap, profiles
- Unified dashboard
- Configure services with typical Kubernetes workflows - CRDs
- Automated sidecar injection is possible

Let’s Look at Istio

Istio a service mesh for Kubernetes that allows us to connect, secure, control and observe services at scale, often requiring no service code modification.

Istio Features

- Traffic Management
  - Fine-grained control with rich routing rules, retries, failovers, and fault injection
- Observability
  - Automatic metrics, logs, and traces for all traffic within a cluster, including cluster ingress and egress

Istio Features

- Security
  - Strong identity-based AuthN and AuthZ layer, secure by default for ingress, egress and service-to-service traffic
- Policy
  - Extensible policy engine supporting access controls, rate limits and quotas

Using Istio

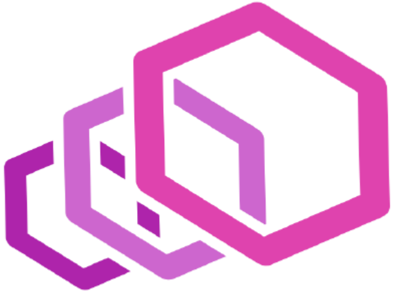
- istioctl, cli for mesh admin
- Kiali – dashboard BUI
- Configure services with typical Kubernetes workflows - CRDs
- Sidecare auto-injection is optional on a per-namespace basis

Istio Components

- Envoy
  - Sidecar proxy
- Pilot
  - Propagates rules to sidecars
- Mixer
  - Enforces access control, collects telemetry data
- Citadel
  - Service-to-service and end-user AuthN and AuthZ

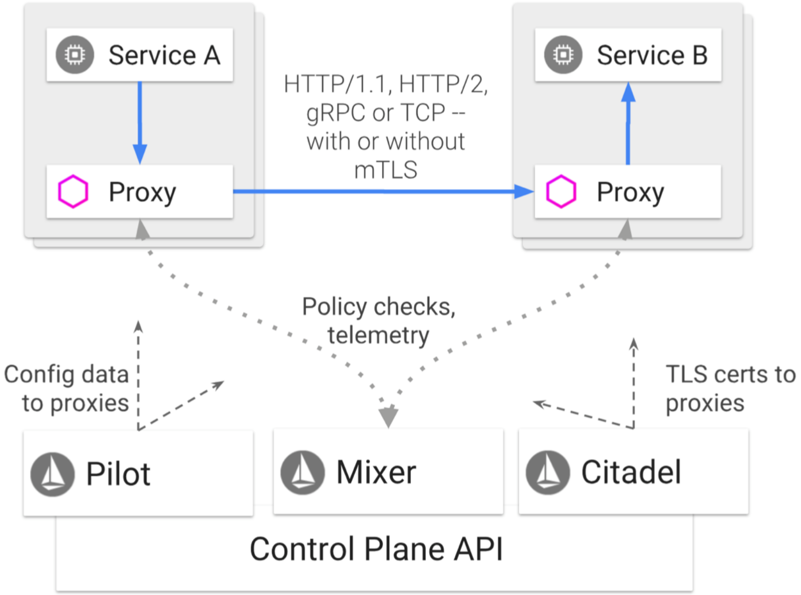
Envoy

High performance proxy which mediates inbound and outbound traffic.



- Dynamic service discovery
- Load balancing
- TLS termination
- HTTP/2 and gRPC proxies
- Circuit breakers
- Health checks
- Split traffic
- Fault injection
- Rich metrics

Istio Architecture



Linkerd or Istio

- Superficially speaking…
  - Istio for depth and features
  - Linkerd for simplicity and ease-of-use
- Your mileage may vary ☺

ORACLE

Cloud Native Labs

Thanks!

Twitter: @jlb13

cloudnative.oracle.com

$500 OCI trial: bit.ly/ChiK8sOC