CODE MENTOR

Micro-Services Day 2 Networking

Istio

Ken Sipe

Distribute Application Engineer

Apache Mesos Contributor, Kubernetes Commiter
Apache Committer Myriad, Open DCOS
Developer: Embedded, C++, Java, Groovy, Grails, C#, GoLang
@KenSipe

Agenda

- Introduction
  - What is Istio
  - Architecture
- Core Features
  - Traffic Control
  - Service Resiliency
  - Observability
  - Security

CODE MENTOR

Labs

- Docker 20.10.6
- Minikube v1.20.0
- Kubectl
- 16 GB mem
- Strong internet (GB downloads)
- Istio 1.8 (no install needed)

https://github.com/kensipe/d2-networking

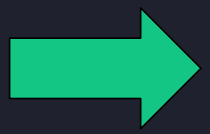
CODE MENTOR

What is Istio?

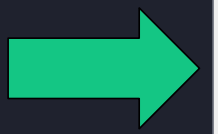
CODE MENTOR

Method Call became Network Call

Network is reliable, homogeneous and secure
- Latency is zero and bandwidth is infinite

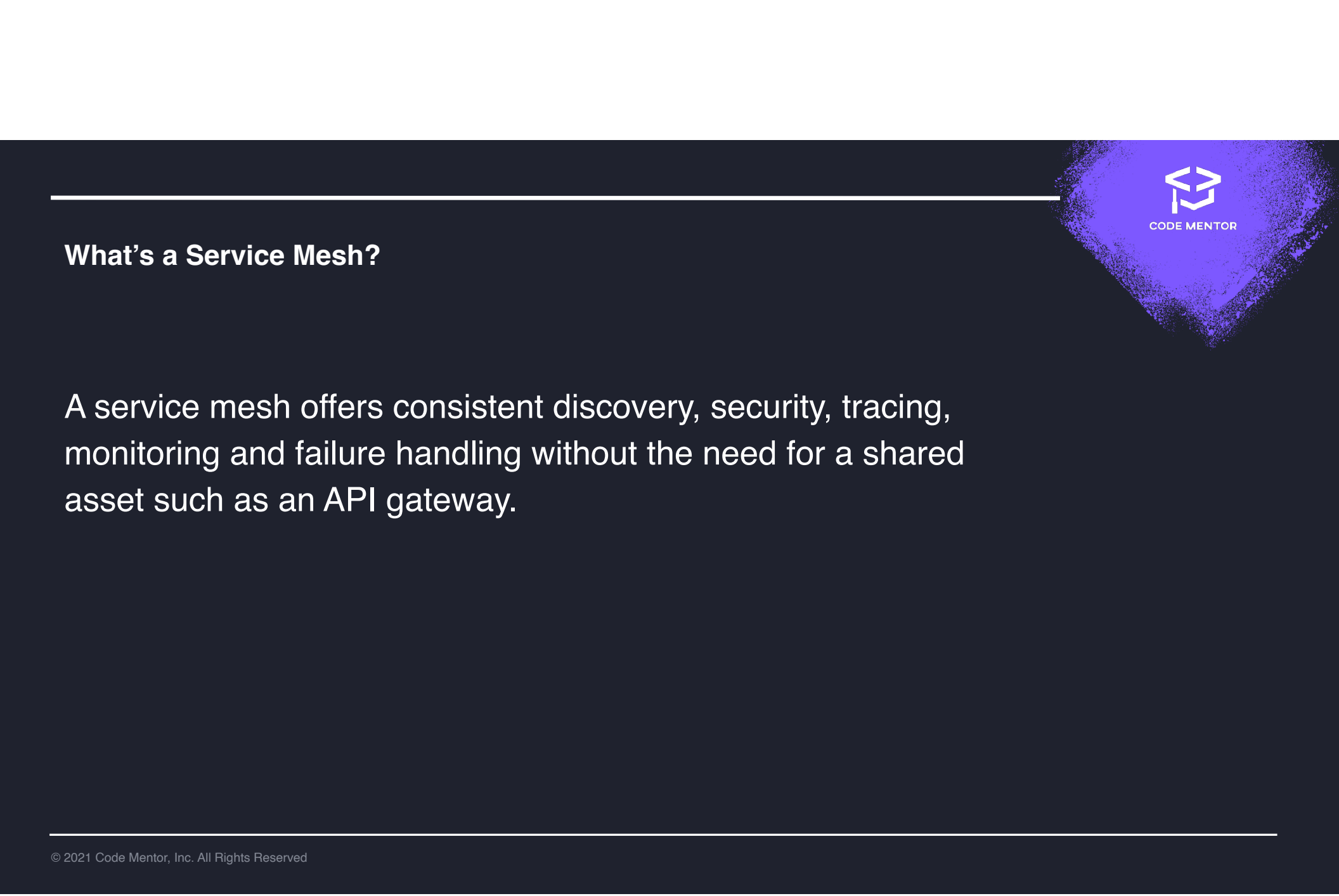


Countermeasures
- Load Balancing, Circuit Breakers, Traffic Management (Canary), Health Checks, Service Discovery
- Auth, Encryption



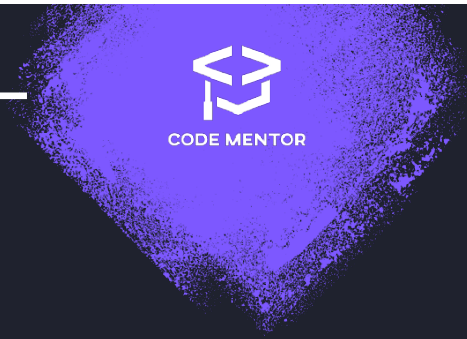
Re-Implementation is expensive
- Global rules and configurations are hard to enforce
- Originally Hystrix and Finagle, but JVM-only

https://en.wikipedia.org/wiki/Fallacies_of_distributed_computing



What's a Service Mesh?

A service mesh offers consistent discovery, security, tracing, monitoring and failure handling without the need for a shared asset such as an API gateway.



Sidecar

- Typically implemented with reverse-proxy processes deployed alongside each service process
- These proxies communicate with service registries, identity providers, log aggregators, and so on

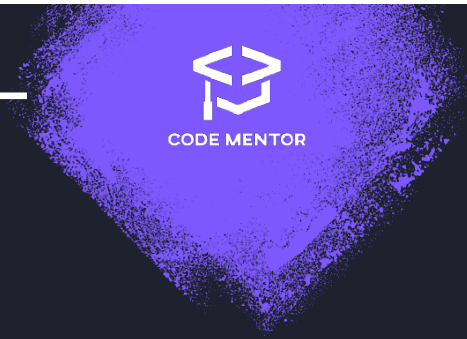
Istio

Control plane for a service mesh

- Automatic load balancing
- Fine-grained control of traffic behavior with routing rules, retries, failovers, and fault injection.
- Policy API supporting access controls, rate limits and quotas.
- Metrics, logs, and traces for all traffic within a cluster
- Identity-based authentication and authorization for service-to-service communication

CODE MENTOR

Architecture



Architecture

Data Plane

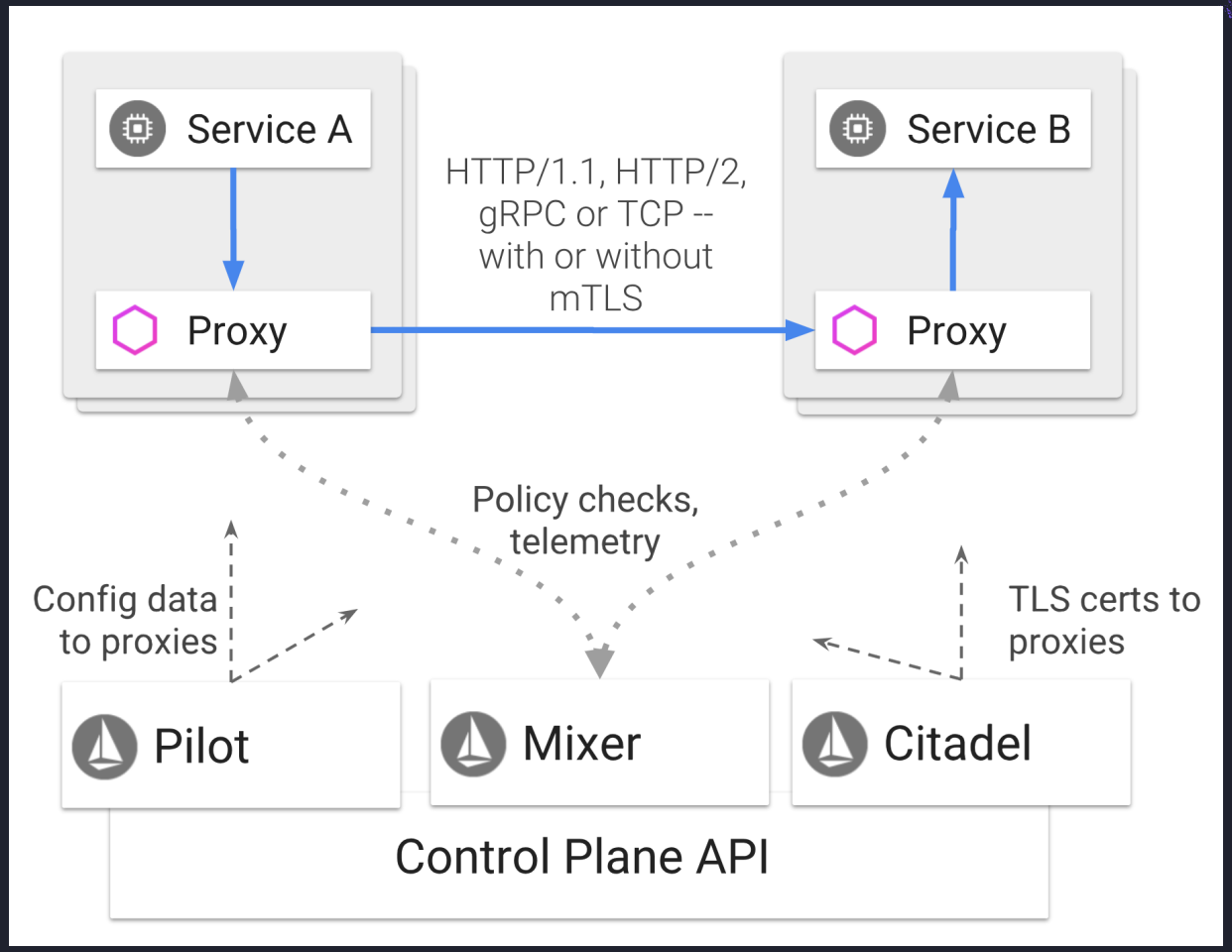
- Composed of a set of intelligent proxies (Envoy) deployed as sidecars
- These proxies control all network communication between microservices

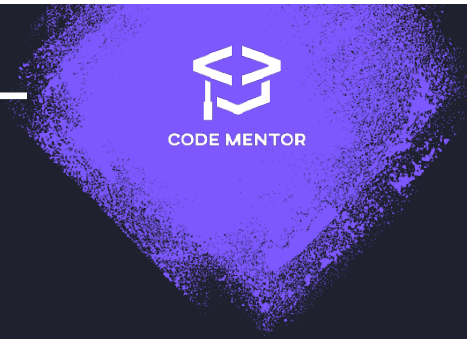
Control Plane

- Manages and configures the proxies to route traffic.
- Configures policies and collect telemetry

https://istio.io/latest/docs/ops/deployment/architecture/

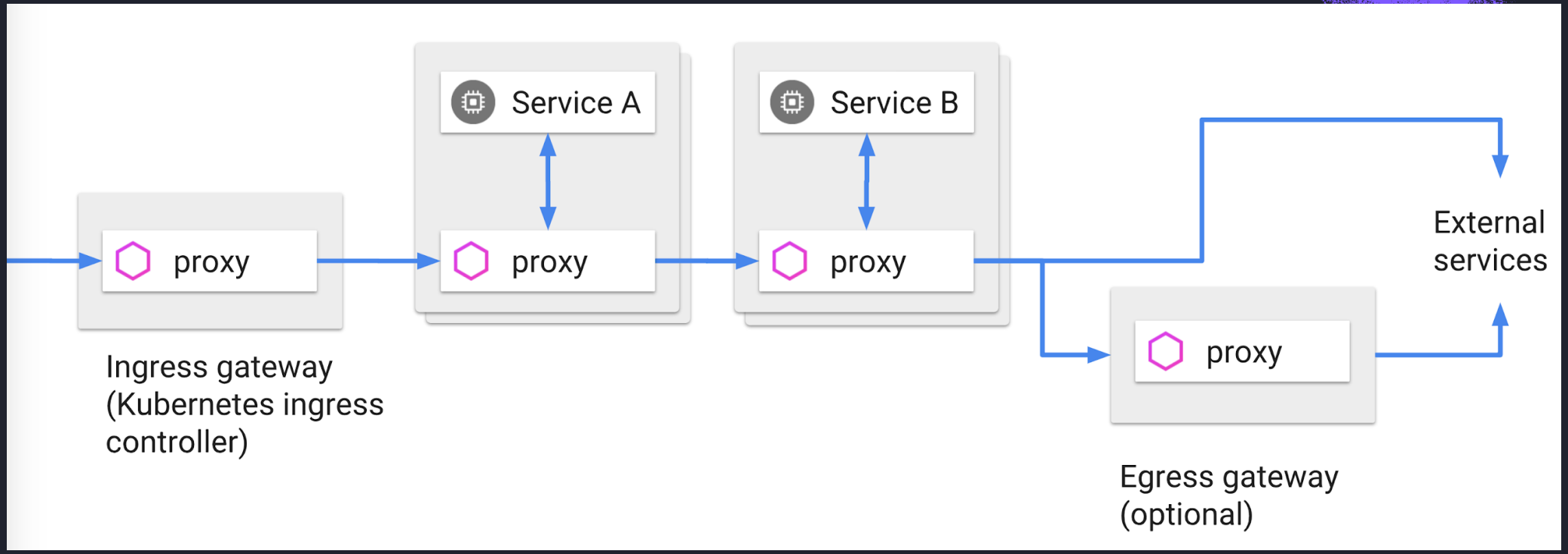
CODE MENTOR

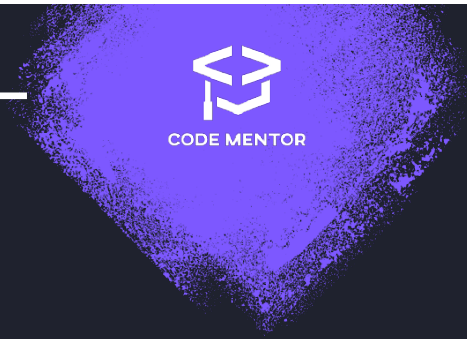




Envoy

- Service proxy written in C++
- Controls all inbound and outbound traffic
- Deployed as **sidecar**
- Istio leverages many built-in features, for example:
  - Service discovery
  - Load balancing
  - TLS termination
  - Health checks
  - Traffic Split
  - Fault injection
  - Metrics



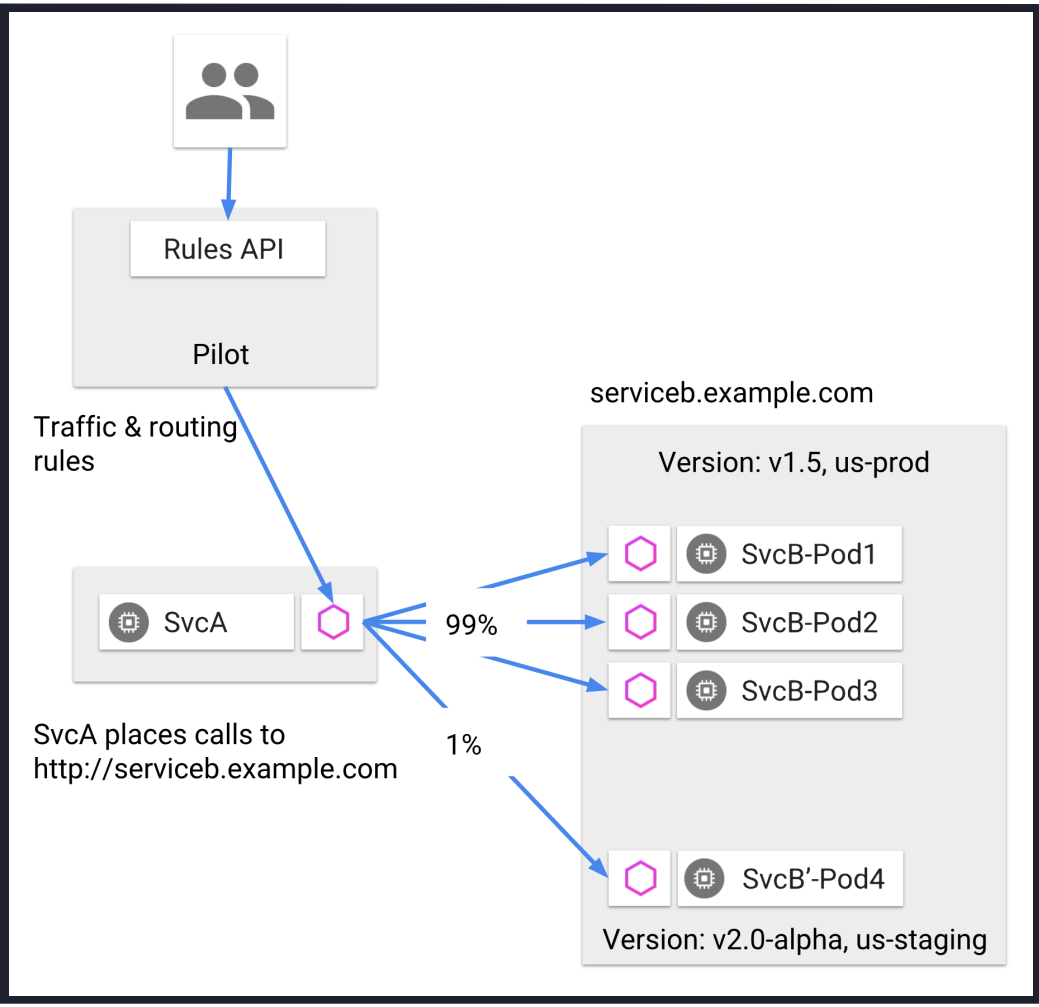


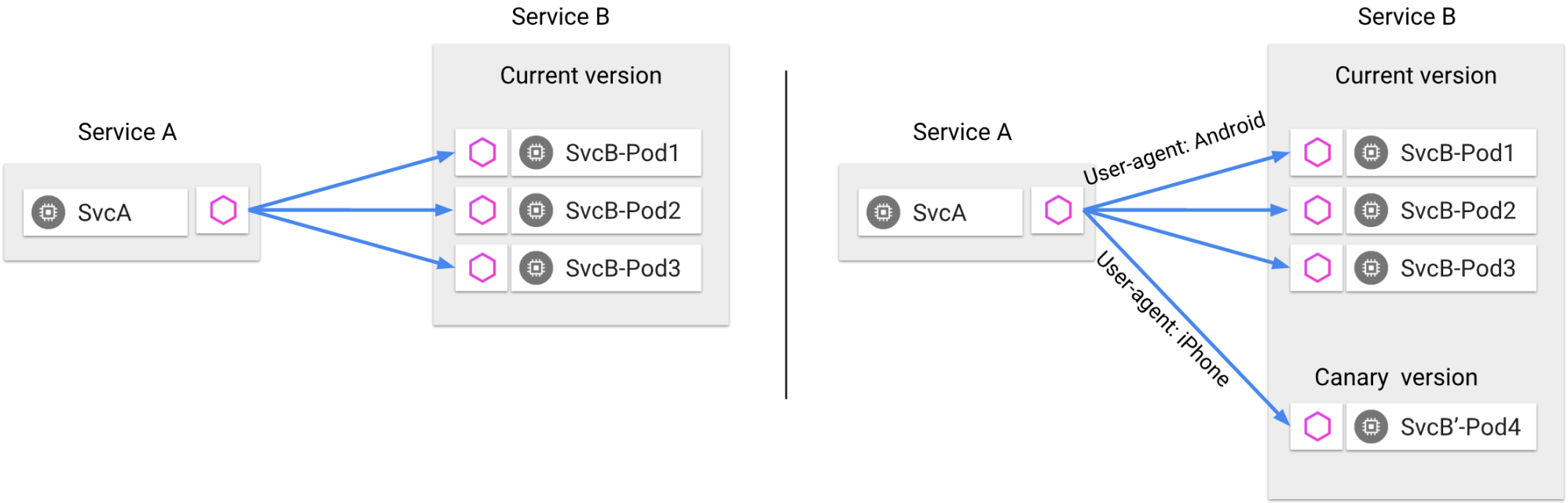
Pilot

- Traffic Control
  - Intelligent routing & policies
  - A/B tests, canary deployments
- Resiliency
  - Circuit breaker, timeouts, retries

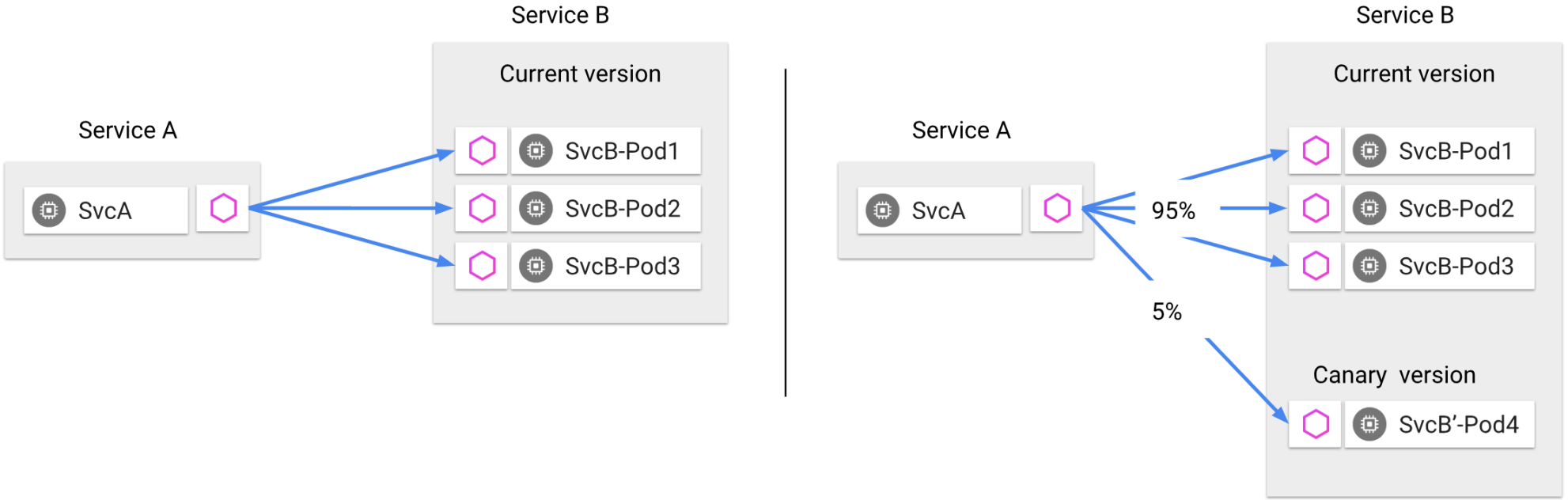
CODE MENTOR

Traffic Control





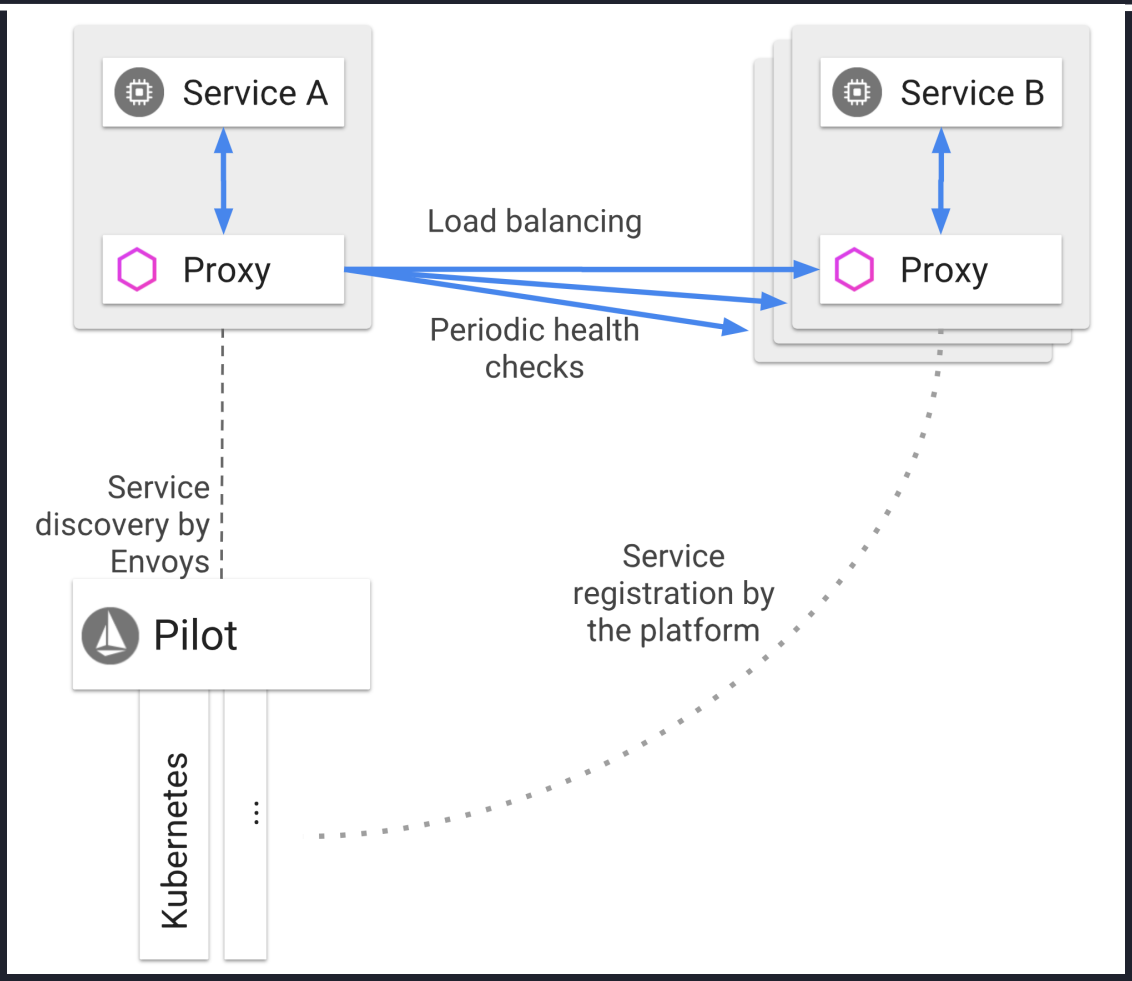
Content-based traffic steering - The content of a request can be used to determine the destination of a request



Traffic splitting decoupled from infrastructure scaling - proportion of traffic routed to a version is independent of number of instances supporting the version

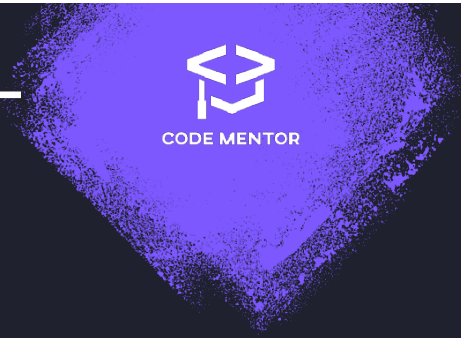
CODE MENTOR

Service Resiliency



CODE MENTOR

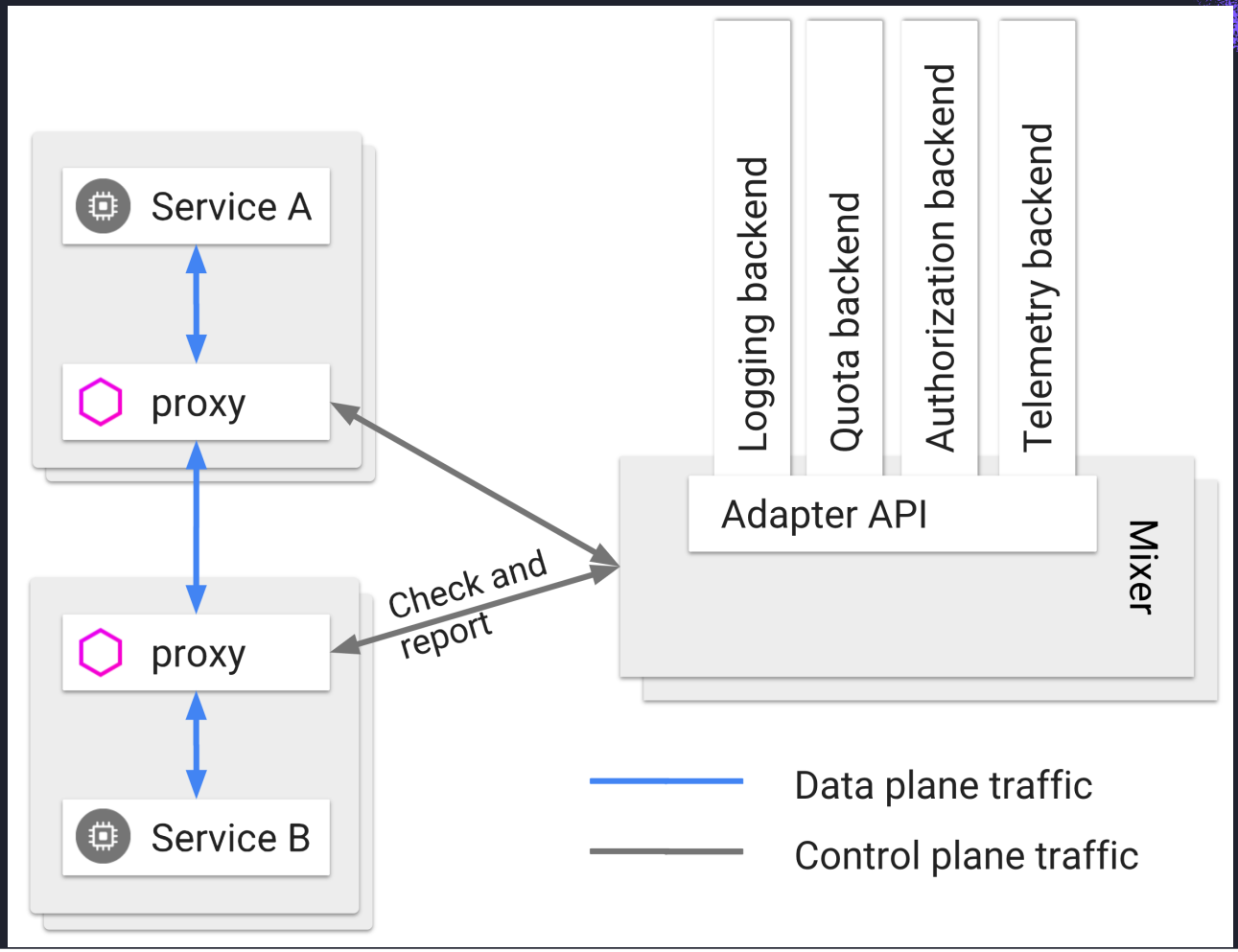
Observability



Mixer

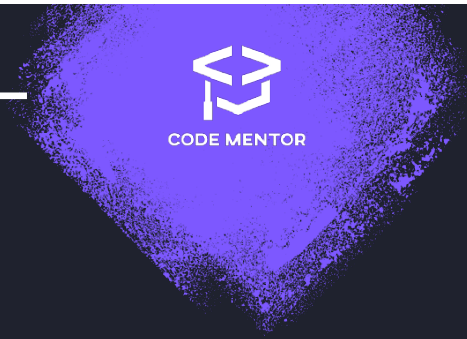
- Access control and usage policies
- Quota/API Management
- Collects Telemetry (Prometheus, Grafana)
- Tracing (Jaeger)

https://istio.io/latest/docs/ops/integrations/



CODE MENTOR

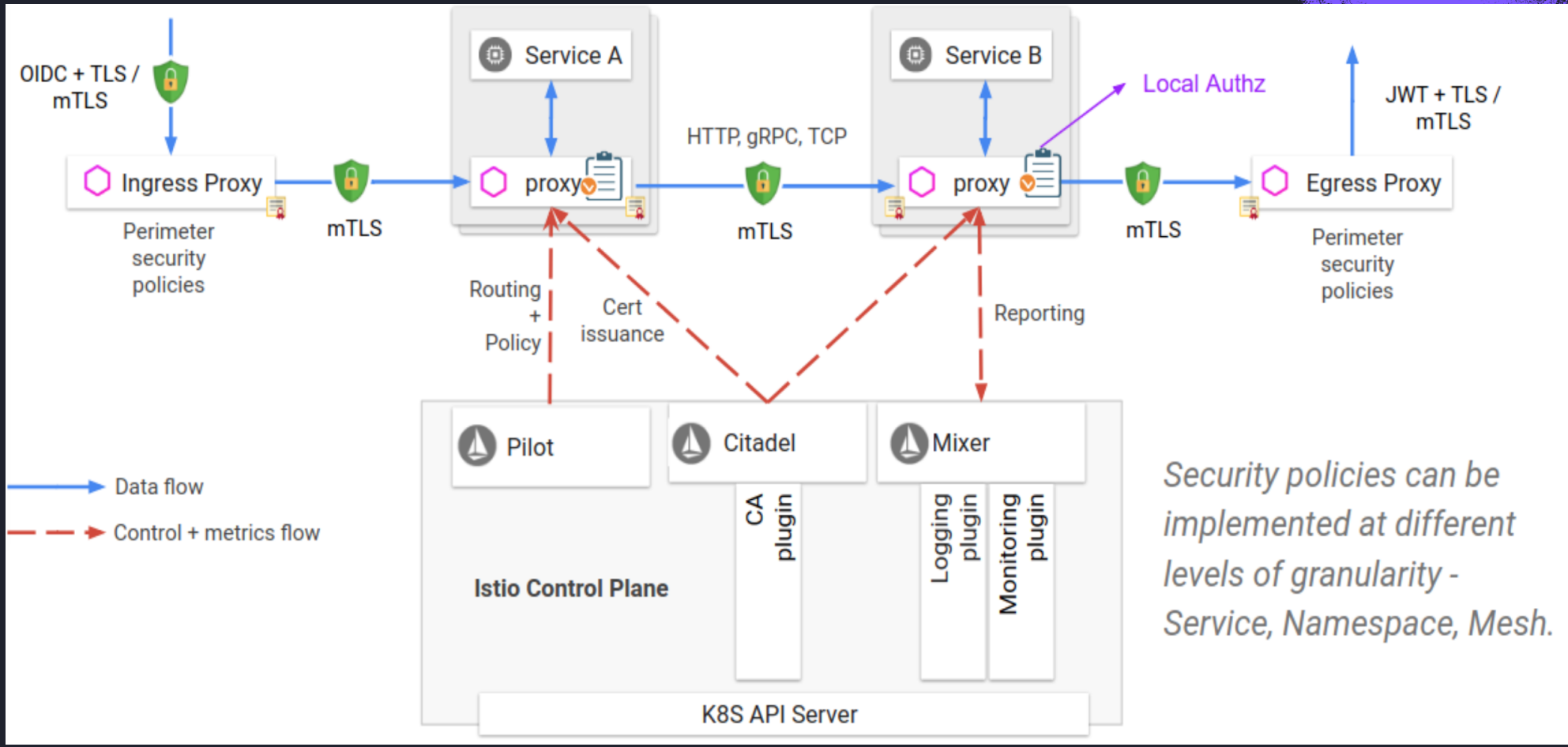
Security



Citadel

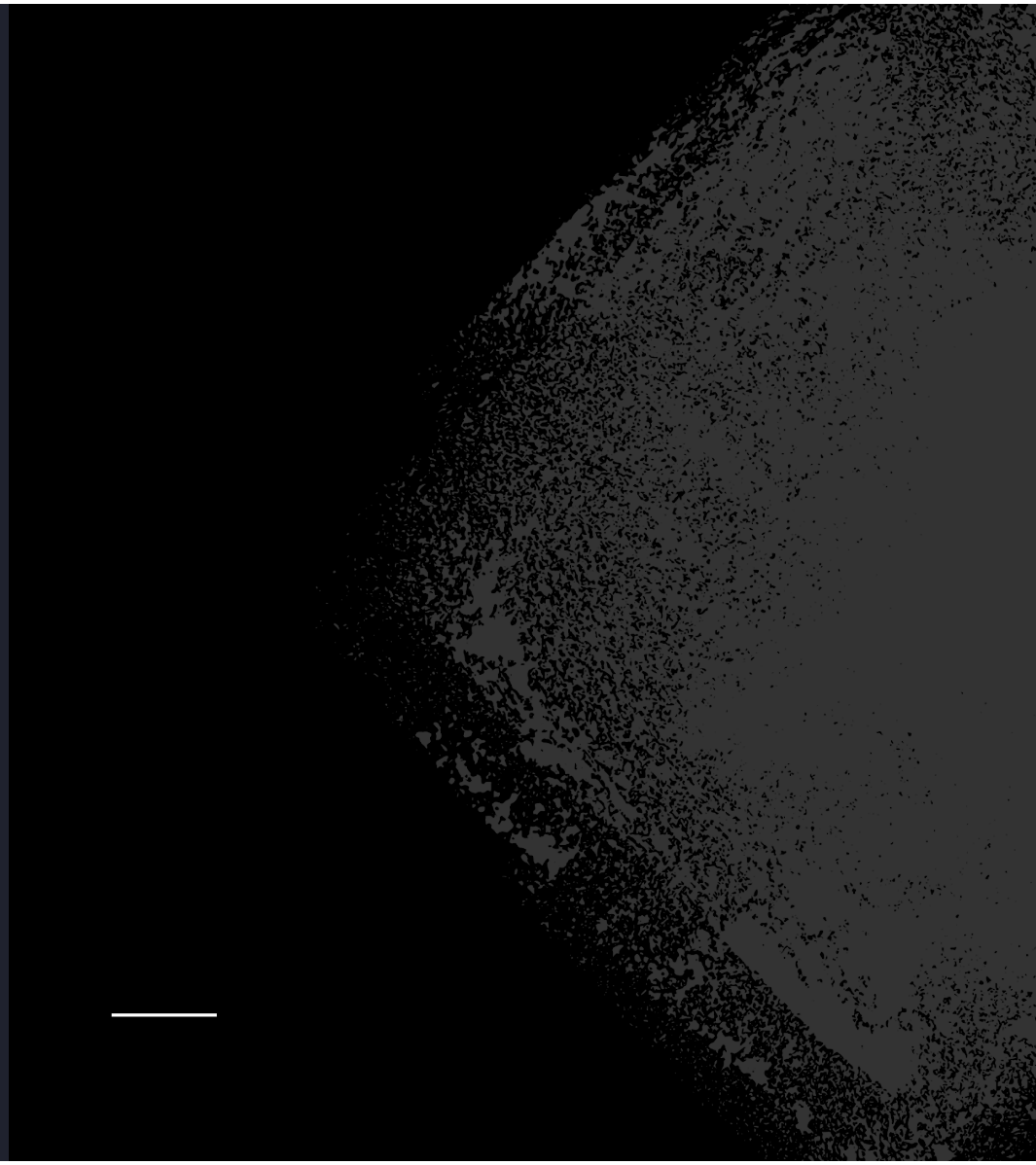
- Enforce mTLS between services
- Authentication and authorization

CODE MENTOR



Security policies can be implemented at different levels of granularity - Service, Namespace, Mesh.

Question?



CODE MENTOR