Introduction to service mesh

by Plamen Stoyanov

[K8S & CNCF Bulgaria] 25/10/2018

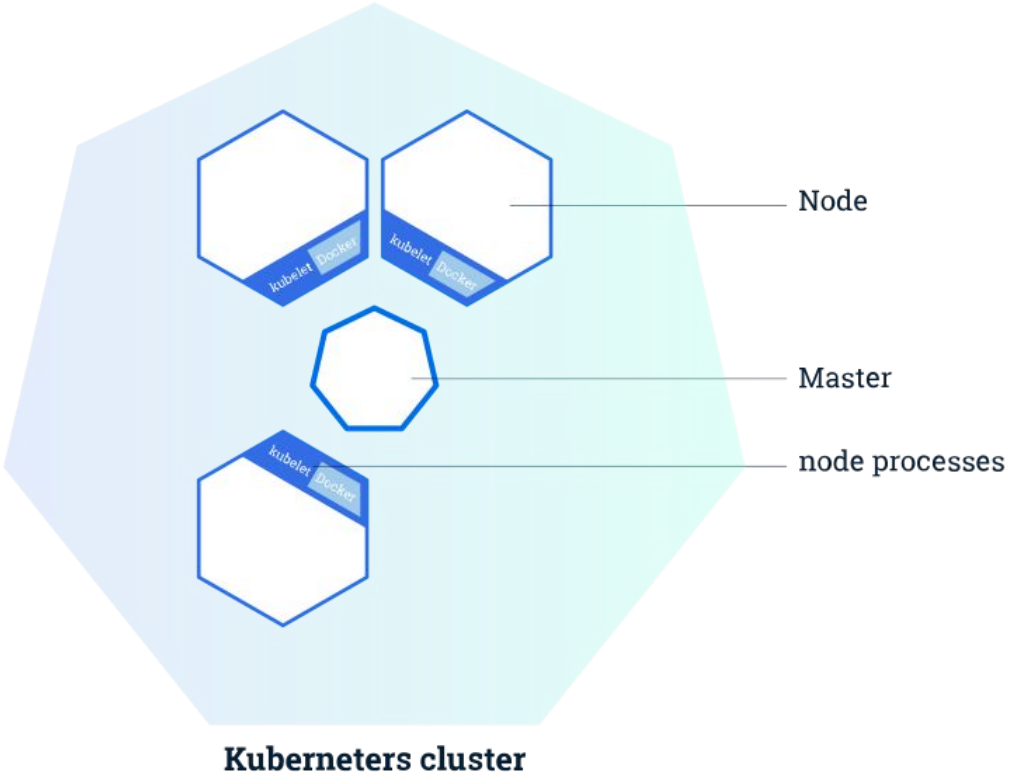
What is kubernetes good for

- Control and automate application deployments and updates.
- Make better use of hardware to maximize resources needed to run your apps.
- Scale containerized applications and their resources on the fly.
- Mount and add storage to run stateful apps.
- Declaratively manage services, which guarantees the deployed applications are always running how you deployed them.
- Health-check and self-heal your apps with autoplacement, autorestart, autoreplication, and autoscaling.

k8s cluster

Kubernetes cluster consist of

- Master
- Worker nodes

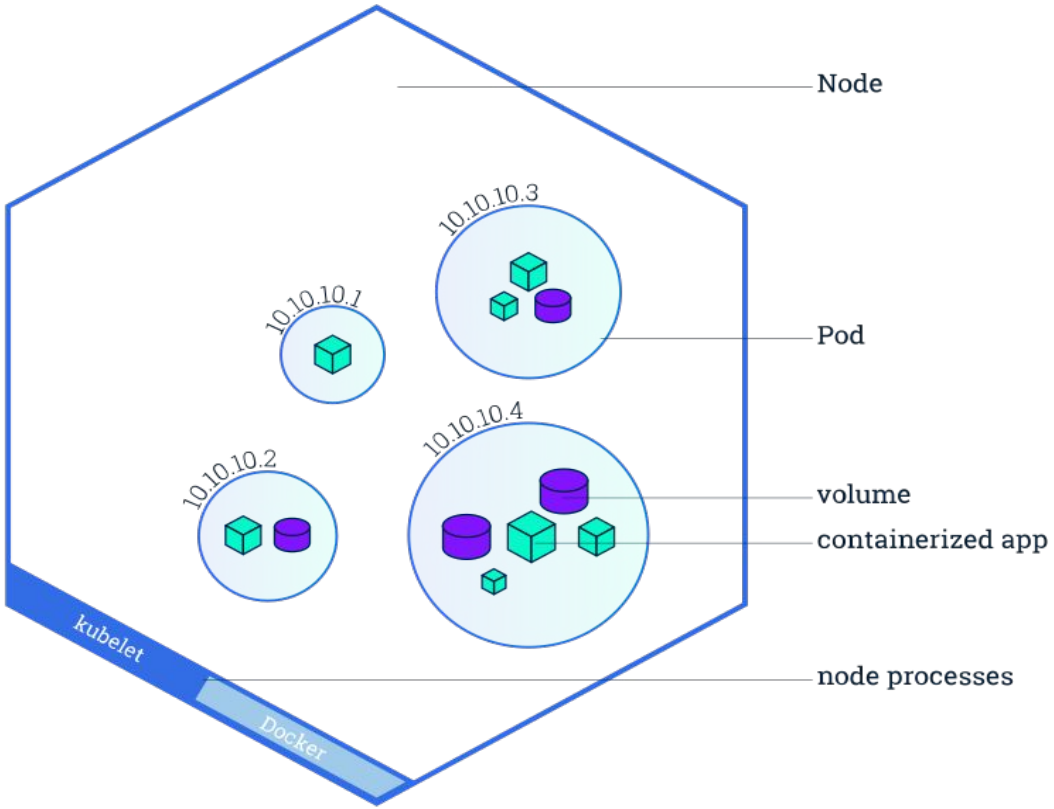


k8s node

A node is a worker machine in Kubernetes
May be a VM or physical machine
Multiple Pods can run on one Node

Every Kubernetes Node runs at least:

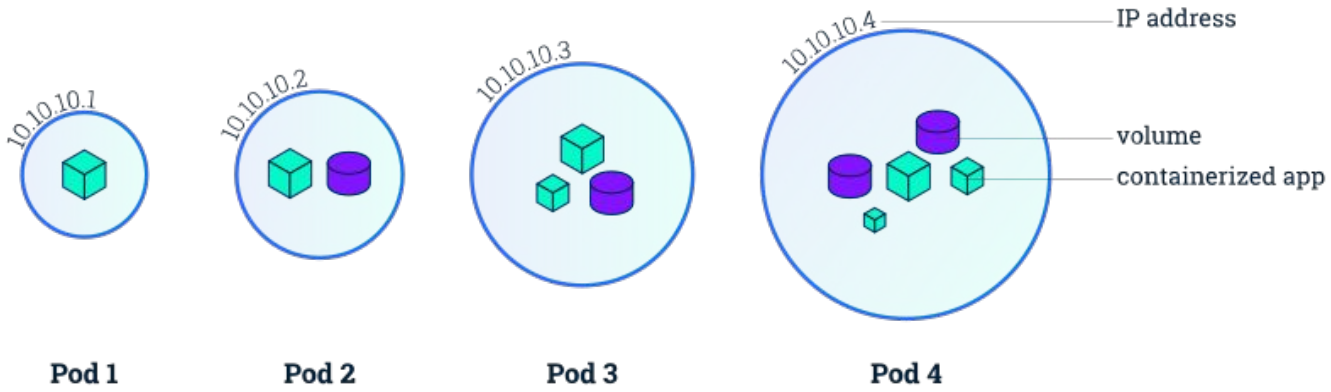
- Kubelet
- A container runtime



k8s pod

A Pod is a group of one or more application containers

- Shared context
- Shared storage
- Unique cluster IP
- Tightly coupled
- Always run on the same node

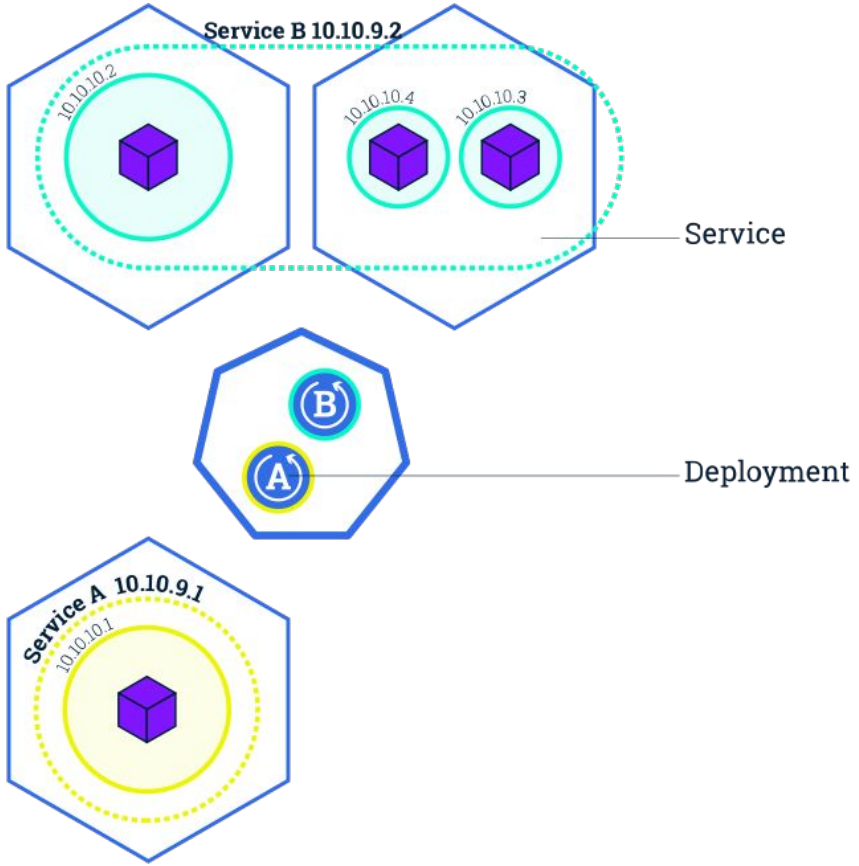


k8s Service

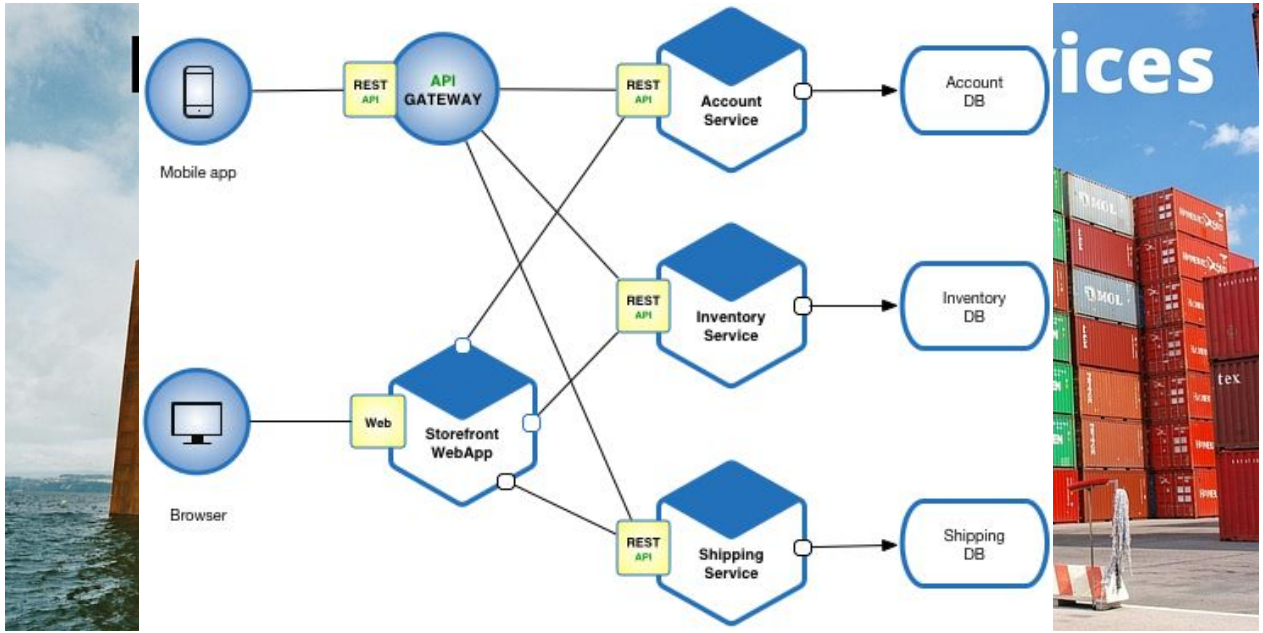
A Kubernetes **Service** is an abstraction layer which defines a logical set of Pods and enables external traffic exposure, load balancing and service discovery for those Pods.

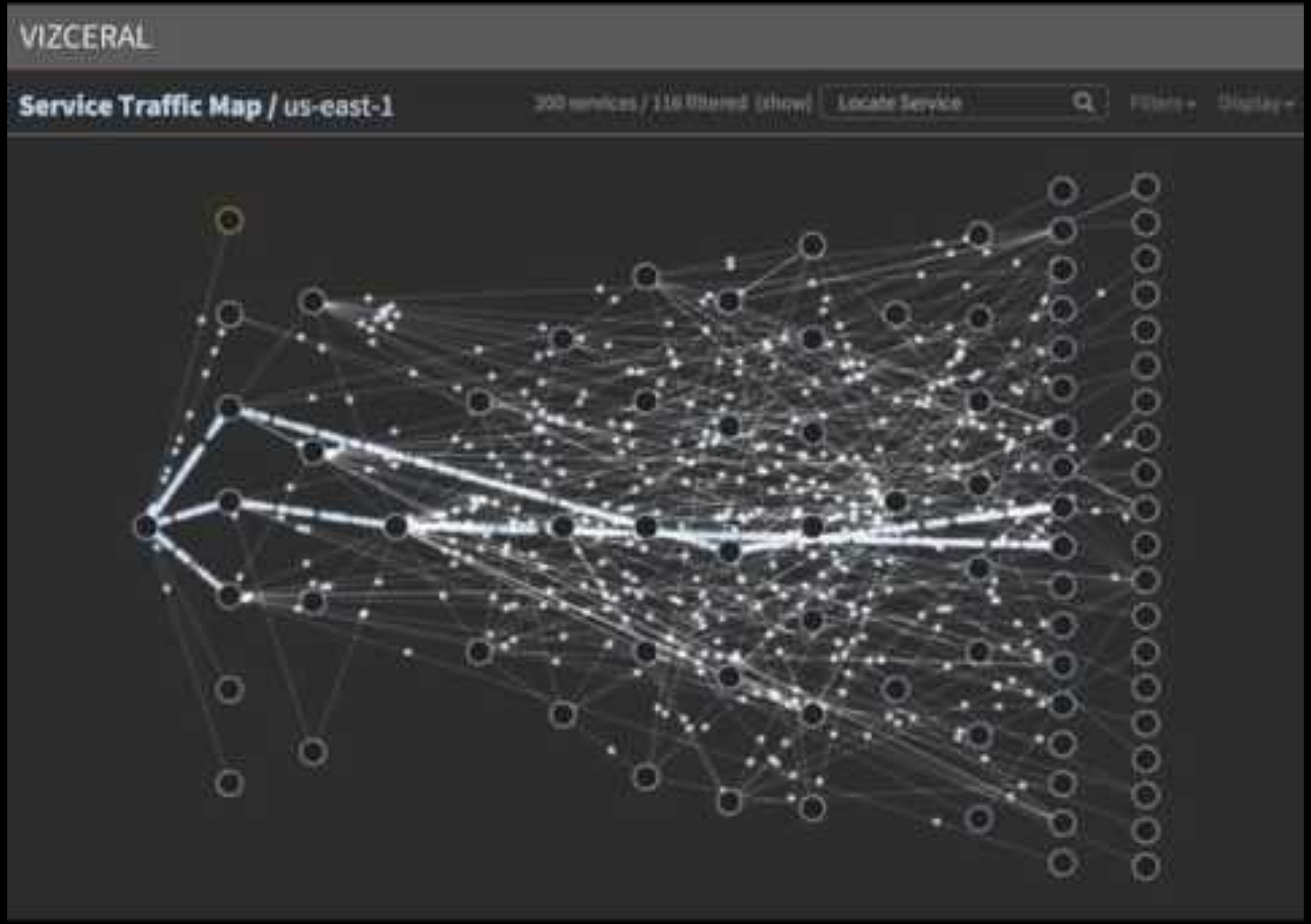
Service:

- Exposes Pods to external traffic
- Load balancing traffic across multiple Pods
- Using labels



Monolith to microservices




VIZCERAL
Service Traffic Map / us-east-1
Locate Service

Microservices challenges

- Observability
  - monitoring
  - performance
  - logging
- Traceability
- Security
  - authentication
  - encryption
  - auditing

What is a service mesh ?

Network for services, not for bytes

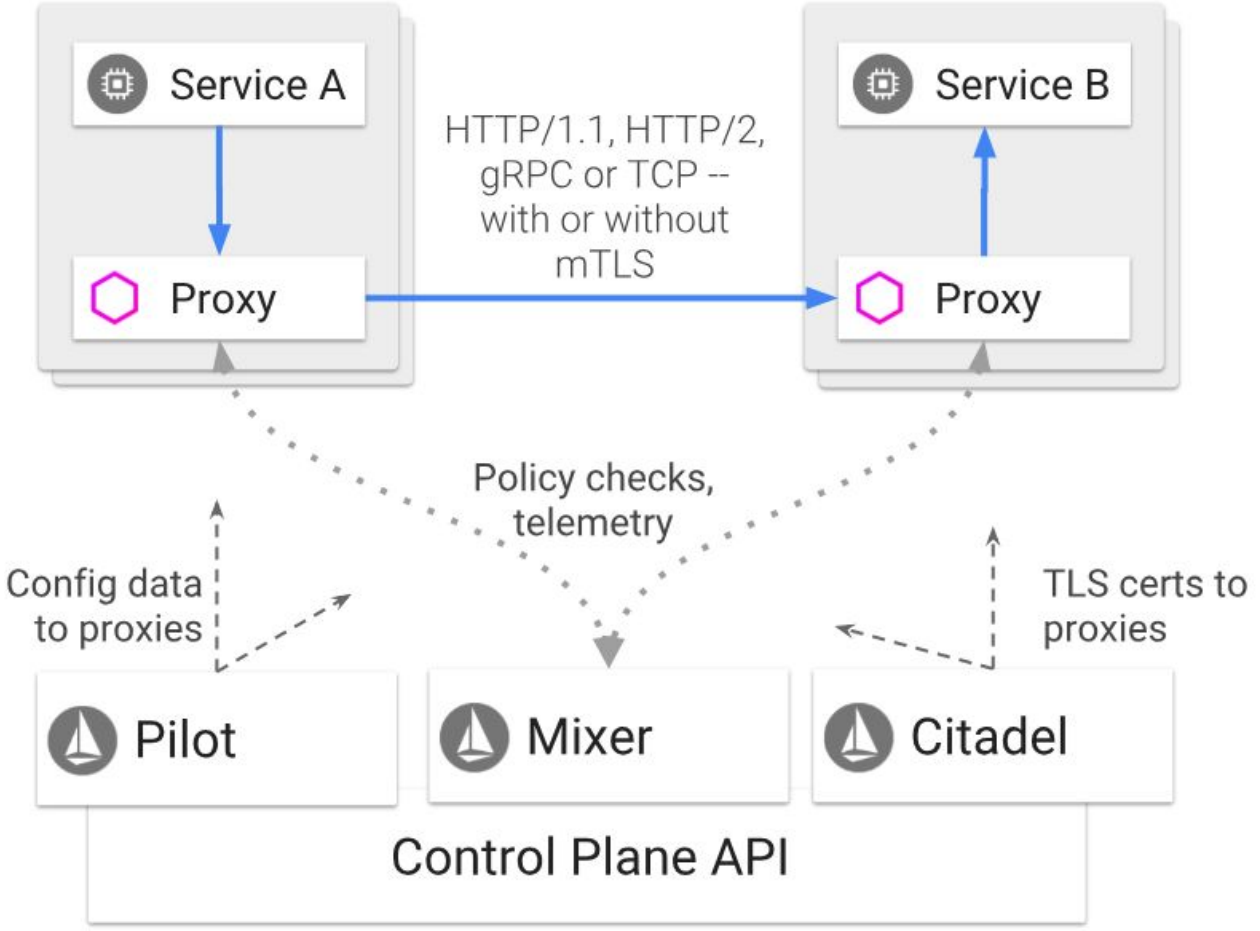
Enter  Istio

- Connect - resiliency, discovery, load balancing
- Manage - traffic control, policy enforcement
- Monitor - metrics, logging, tracing
- Secure - end-to-end authentication and authorization
- Deploy - canary, rolling upgrades, separation
- Test - manage chaos

Main Istio contributors

Istio Architecture

- **Data plane -** composed of a set of intelligent proxies (Envoy) deployed as sidecars
- **Control plane -** manages and configures the proxies to route traffic



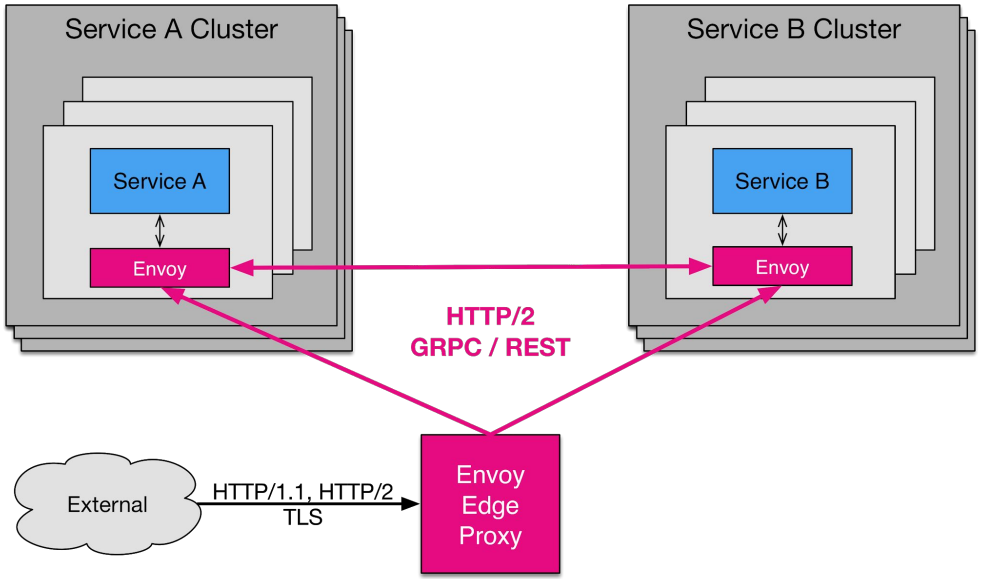
Envoy

Isto uses extended version of the Envoy proxy.

Features:

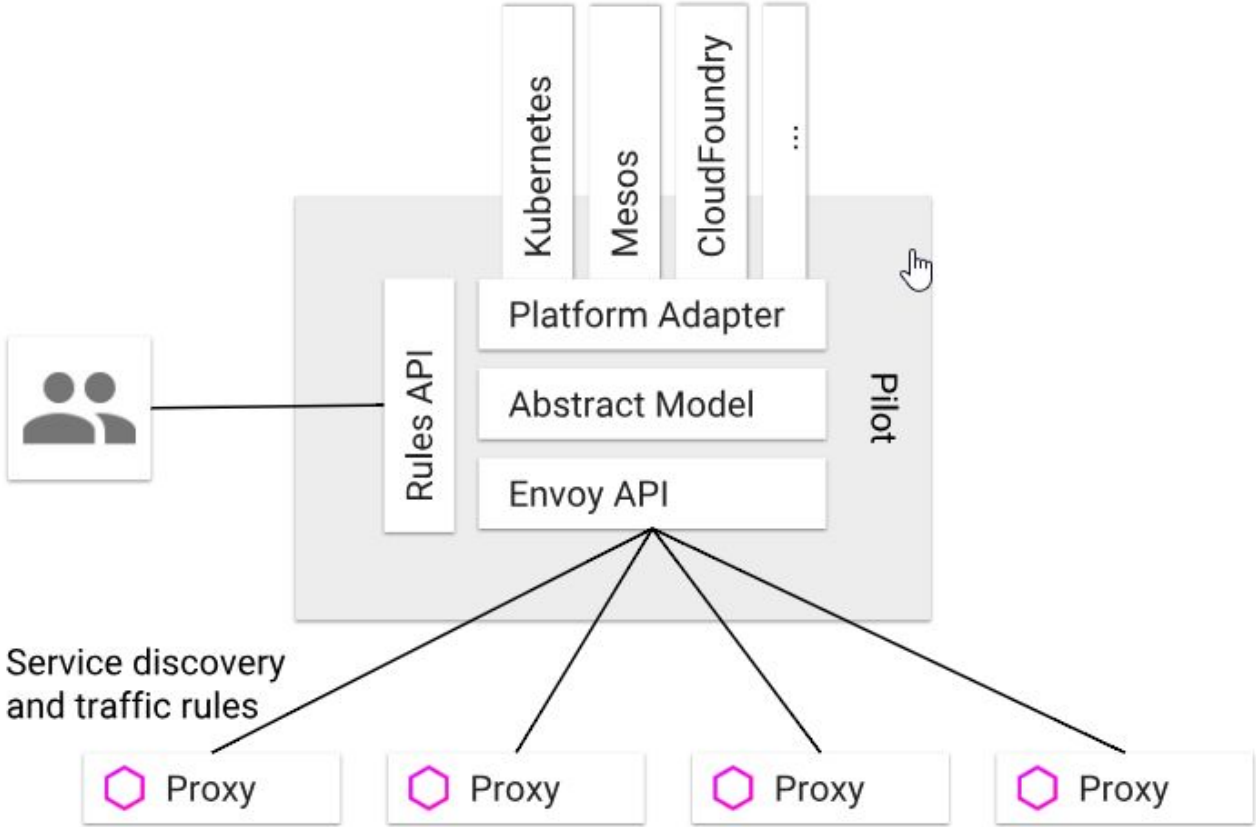
- Dynamic service discovery
- Load balancing
- TLS termination
- HTTP/2 and gRPC proxies
- Circuit breakers
- Health checks
- Staged rollouts with %-based traffic split
- Fault injection
- Rich metrics

Envoy is deployed as a **sidecar** to the relevant service in the same Kubernetes pod



Pilot

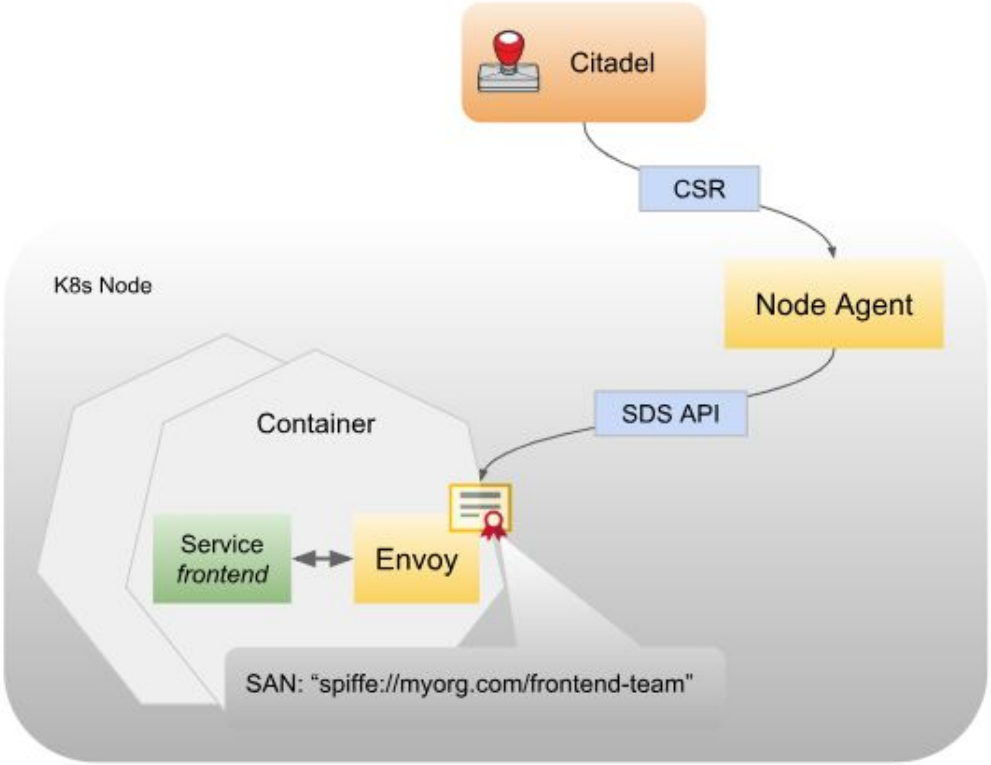
Pilot converts high level routing rules that control traffic behavior into Envoy-specific configurations, and propagates them to the sidecars at runtime



Citadel

Citadel provides strong service-to-service and end-user authentication with built-in identity and credential management.

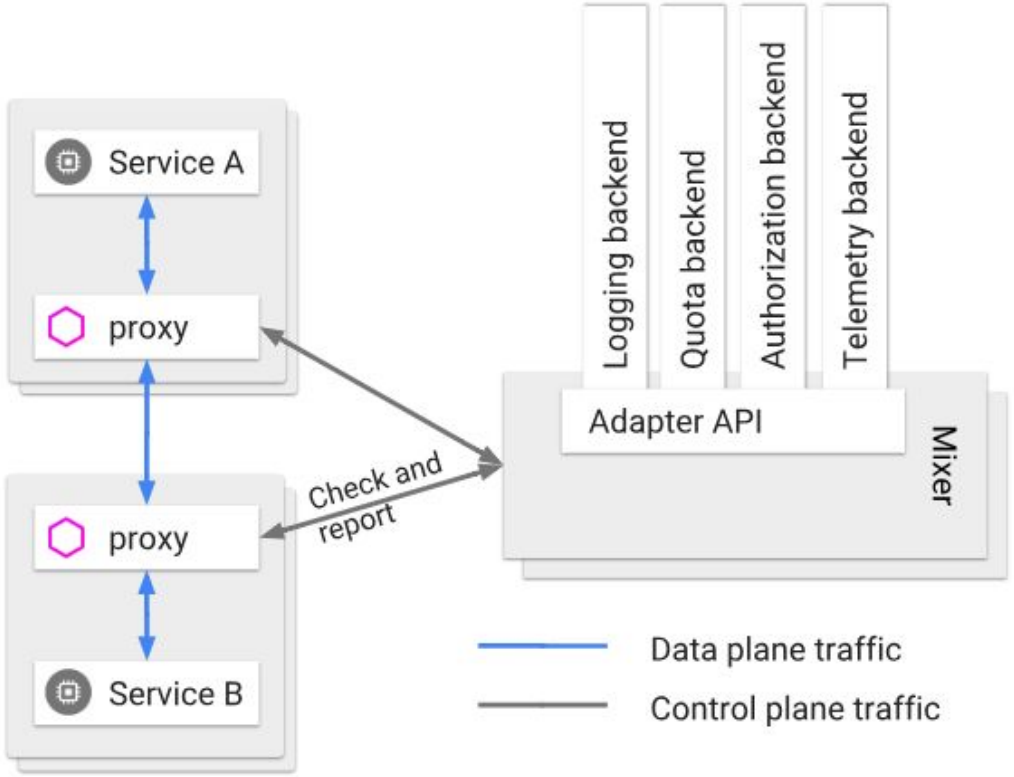
- SPIFFE
- PKI



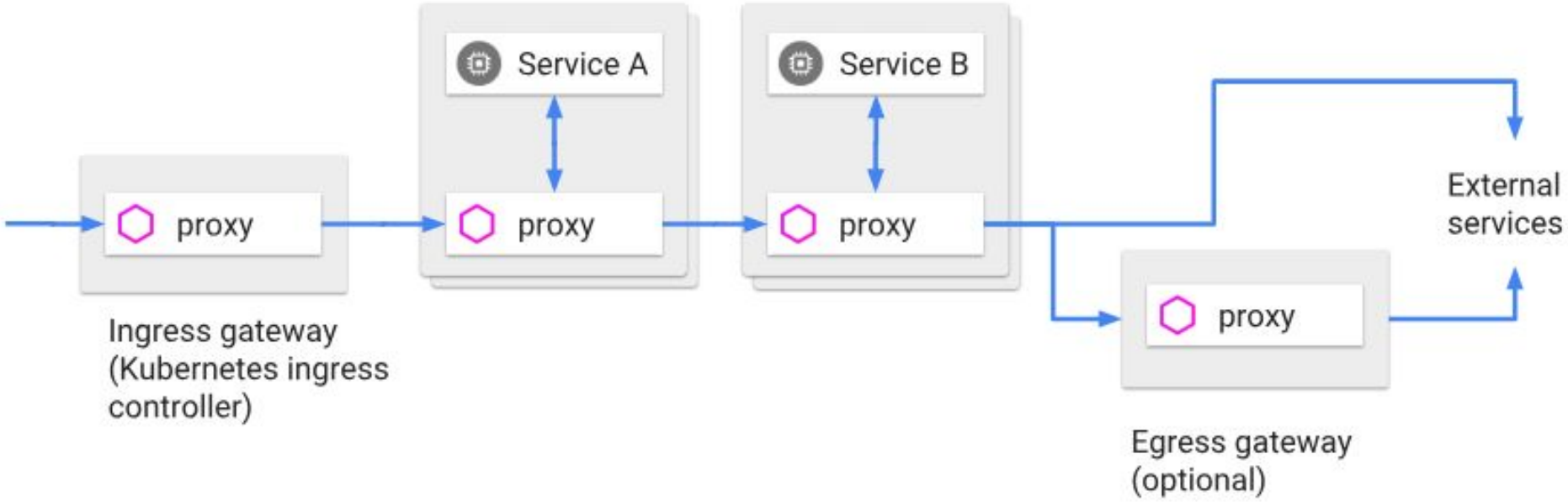
Mixer

Mixer is a platform-independent component. Mixer enforces access control and usage policies across the service mesh, and collects telemetry data from the Envoy proxy and other services

Mixer includes a flexible plugin model.

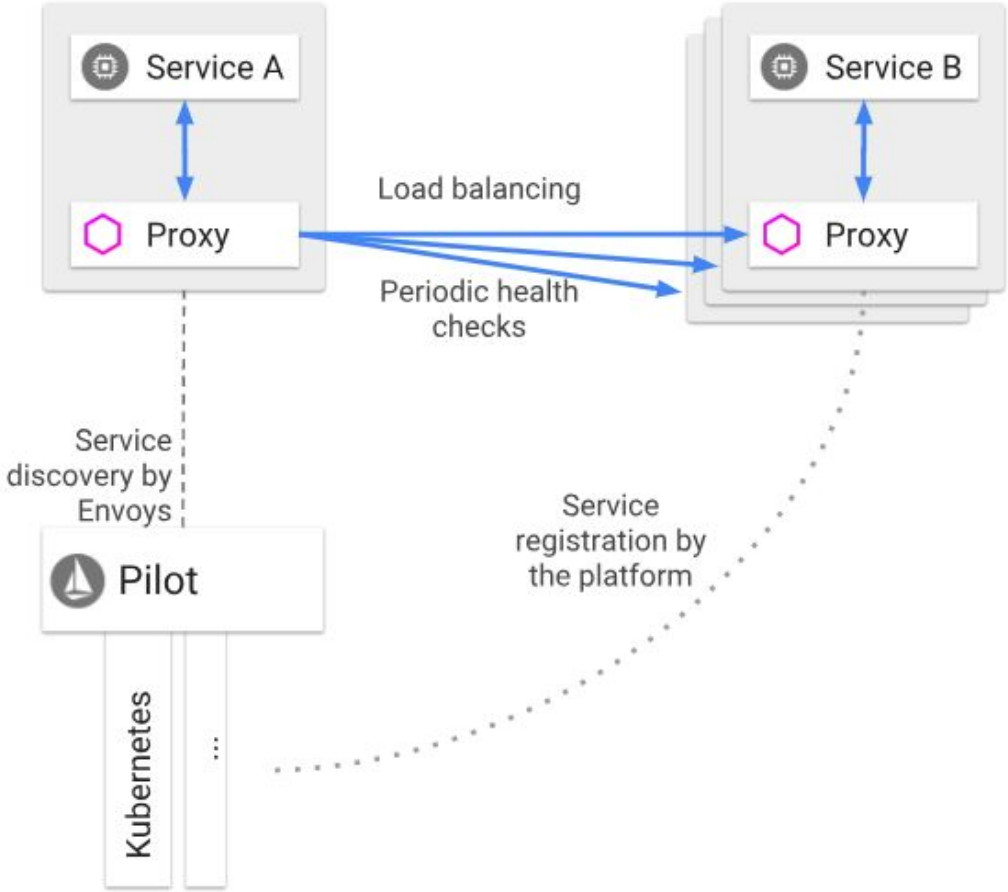


Request flow



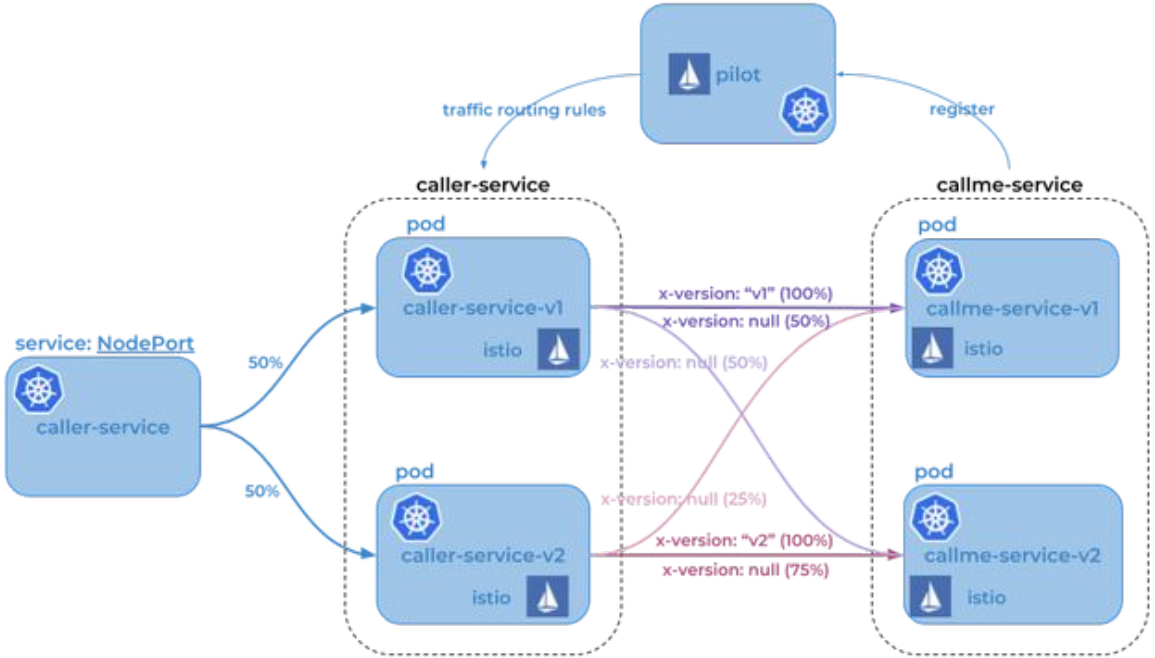
Discovery and load balancing

- K8s DNS
- Istio virtual services

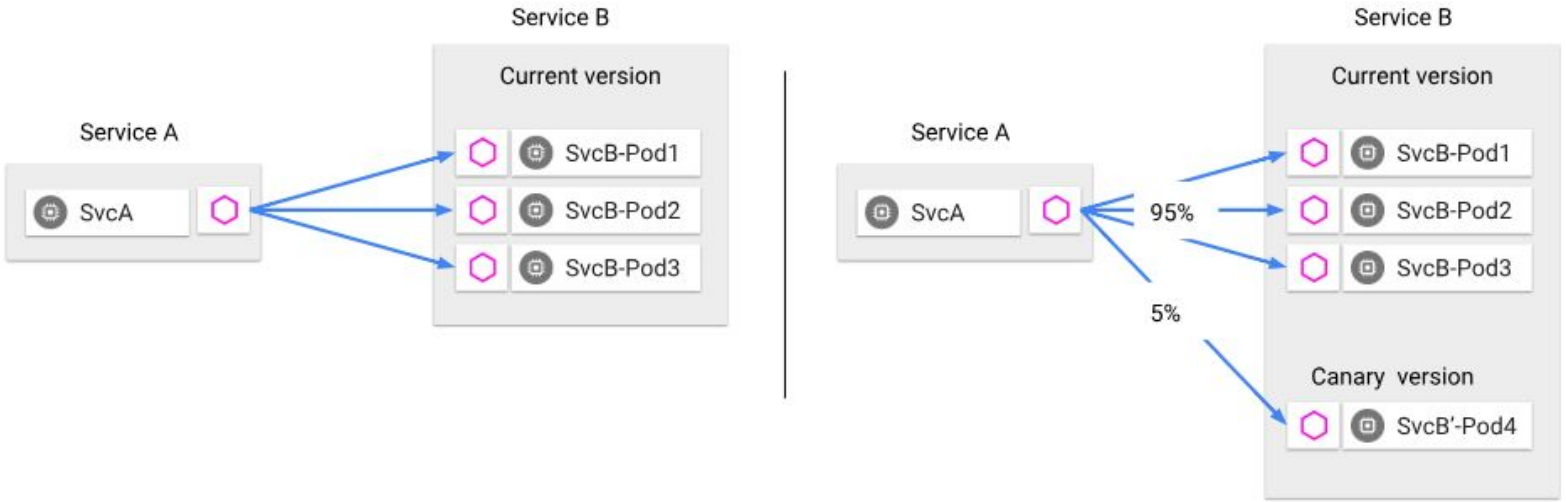


Traffic management

- Routing rules
- Traffic shifting
- Traffic mirroring
- Conditional rules
- Fault ingestion

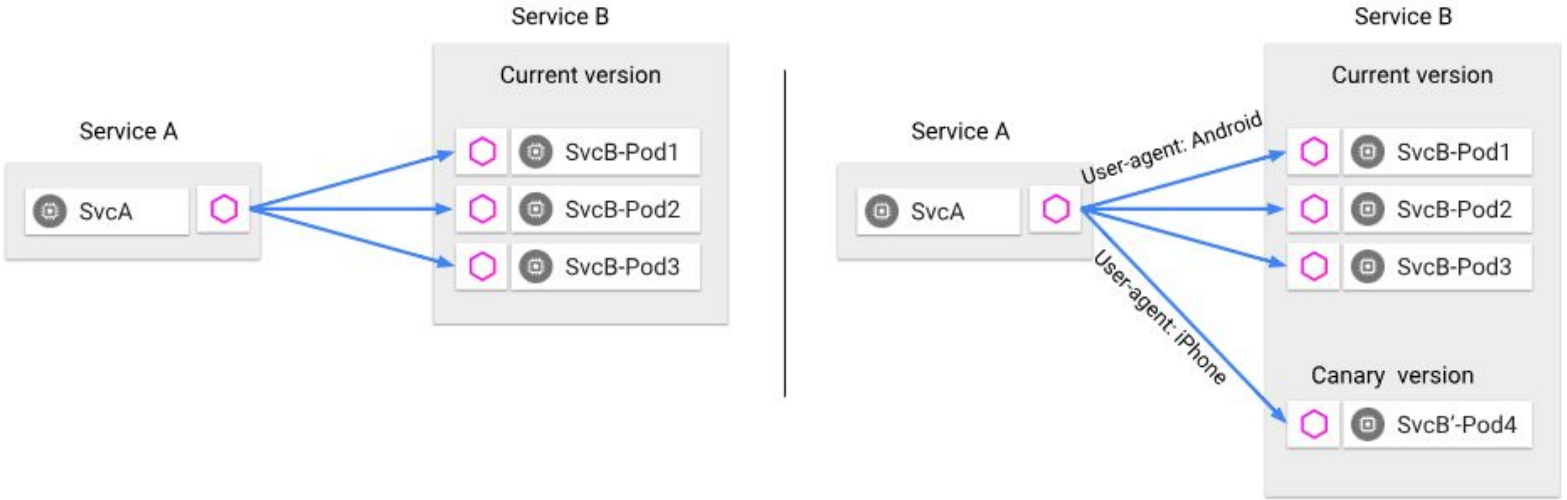


Traffic management



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

Traffic management



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

Handling failures

Envoy provides a set of out-of-the-box opt-in failure recovery features that can be taken advantage of by the services in an application.

Features include:

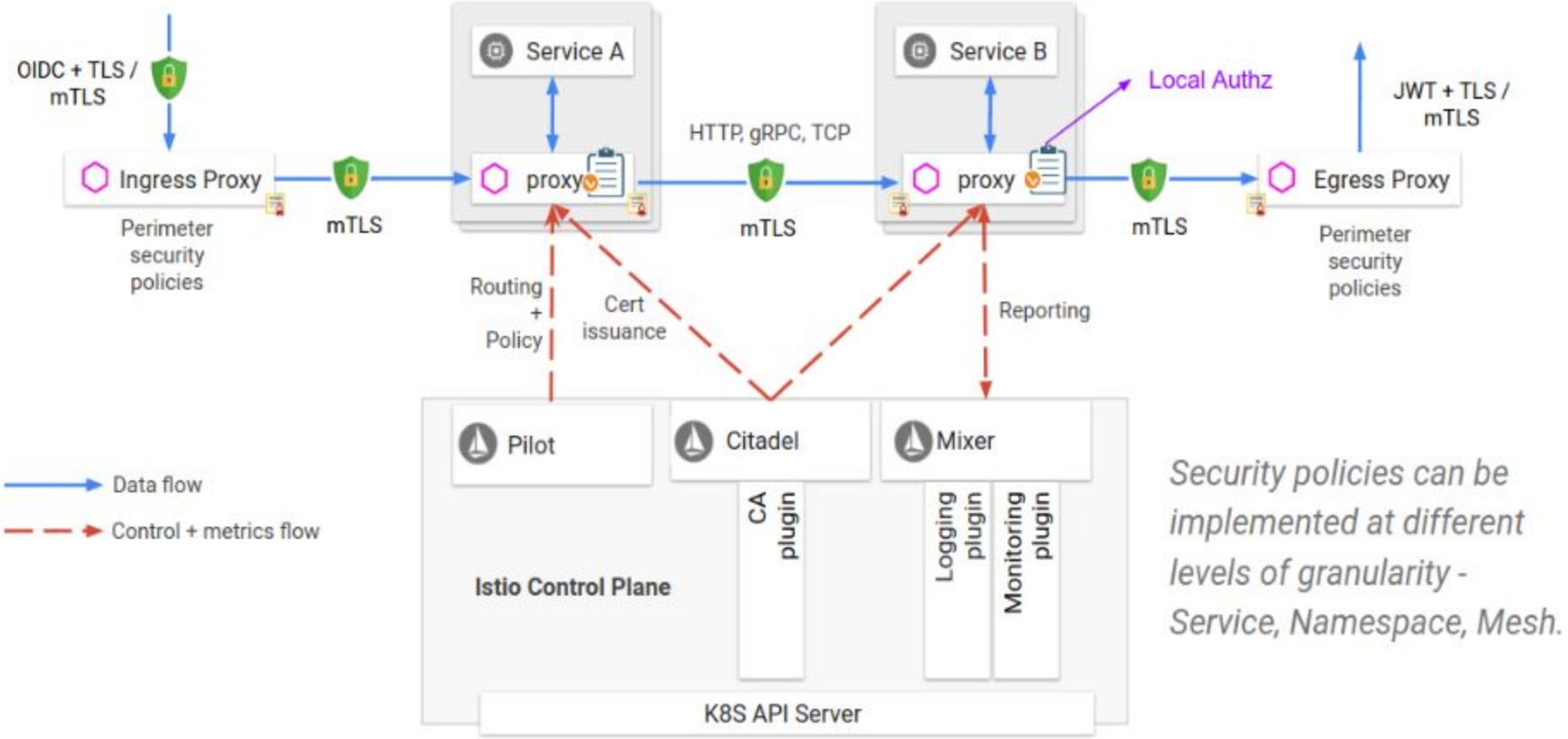
- Timeouts
- Bounded retries with timeout budgets and variable jitter between retries
- Limits on number of concurrent connections and requests to upstream services
- Active (periodic) health checks on each member of the load balancing pool
- Fine-grained circuit breakers (passive health checks) – applied per instance in the load balancing pool

Istio Security

- Secure by default
- Zero-trust network
- Defense in depth

- Identity
- Policy
- Encryption

- Communication
- Endpoints
- Data
- Platform

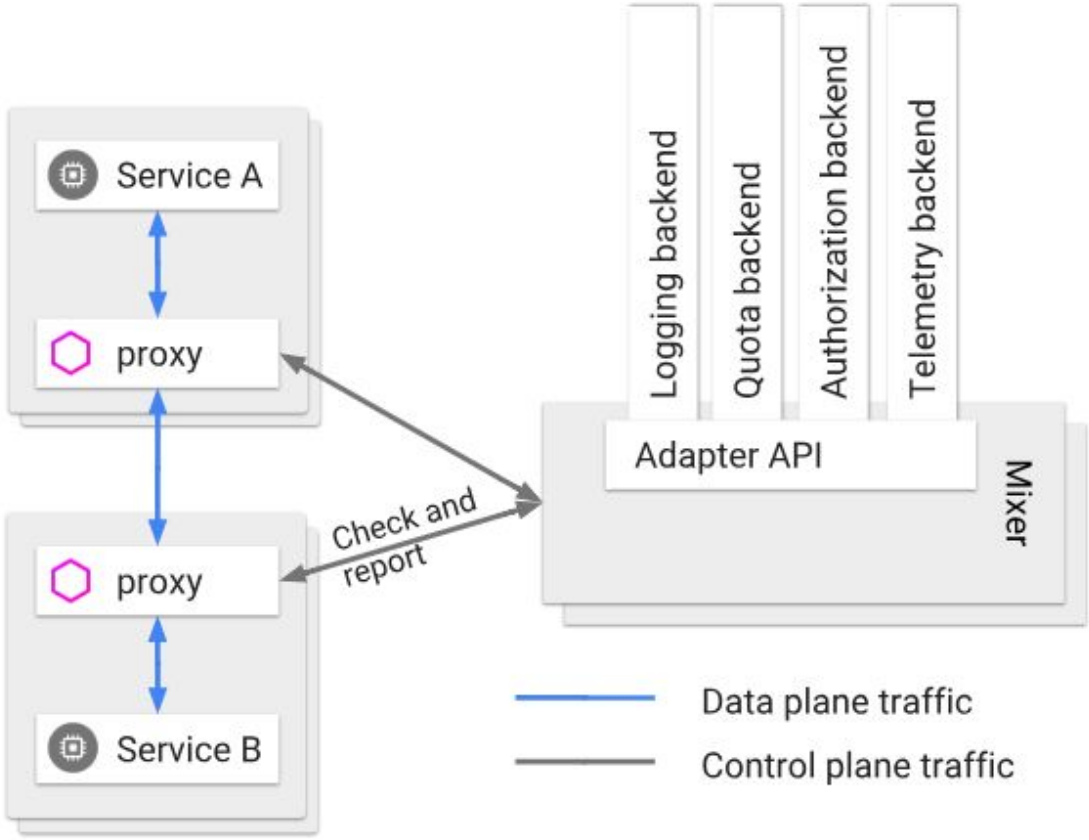


Security policies can be implemented at different levels of granularity - Service, Namespace, Mesh.

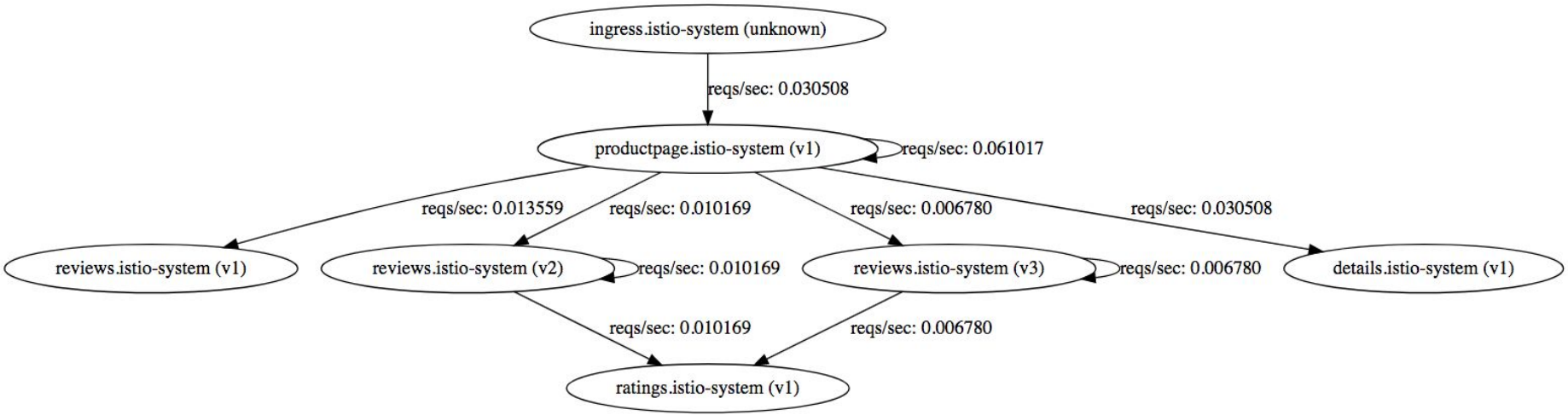
Istio Security Architecture

Istio Observability

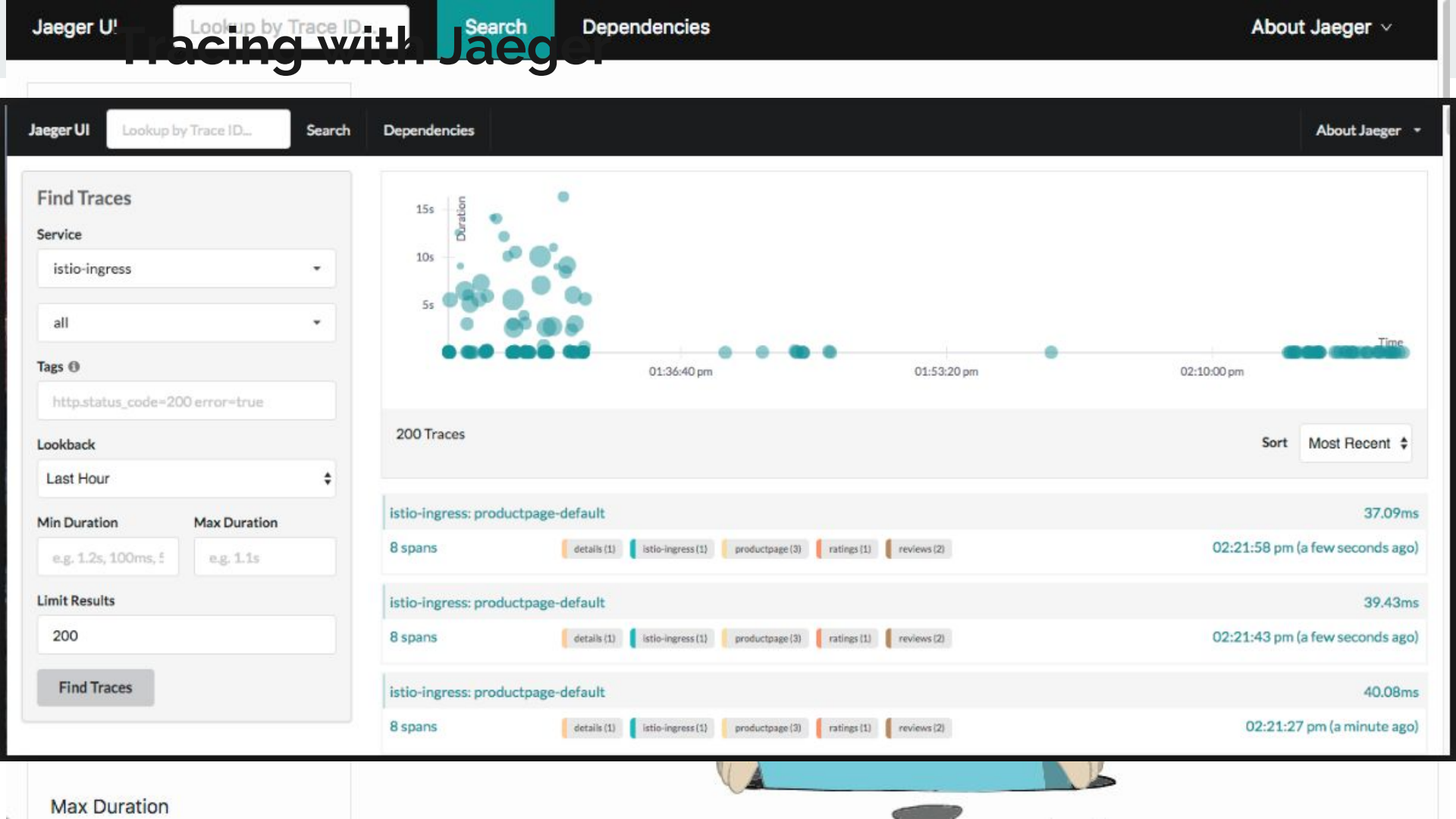
- Telemetry
- Logging
- Auditing



Istio service graph



Tracing with Jaeger



Max Duration

Istio Mesh Da

Prometheus

istio_request_count

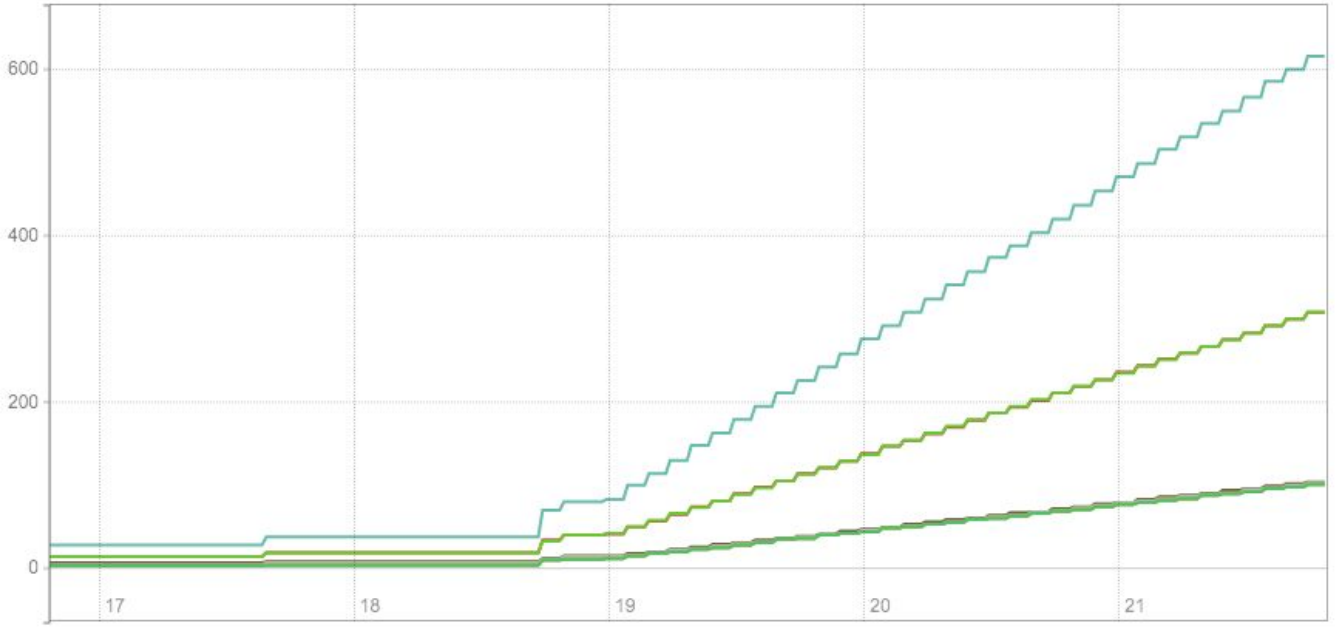
Load time: 794ms
Resolution: 1s
Total time series: 10

Execute

- insert metric at cursor -

Graph Console

5m Until Res. (s) stacked



istio_request_count{destination_service="reviews.istio-system.svc.cluster.local",destination_version="v3",instance="istio-mixer.istio-system:42422",job="istio-
istio_request_count{destination_service="reviews.istio-system.svc.cluster.local",destination_version="v3",instance="istio-mixer.istio-system:42422",job="istio-
istio_request_count{destination_service="reviews.istio-system.svc.cluster.local",destination_version="v2",instance="istio-mixer.istio-system:42422",job="istio-

Load time: 149ms
Resolution: 14s
Total time series: 10

Execute

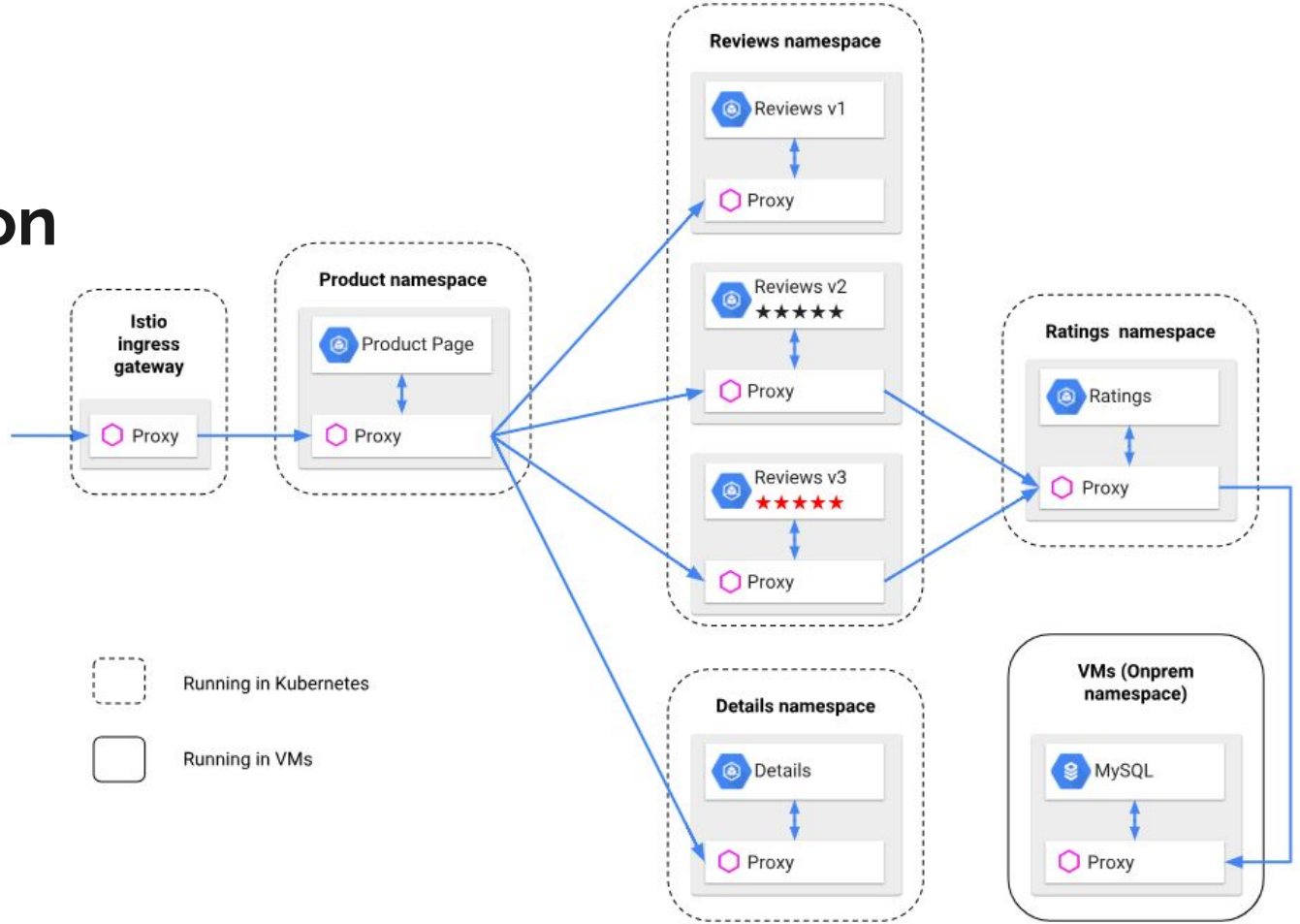
Graph Con

Element	Value
istio_request_cou mesh",response_	154
istio_request_cou mesh",response_	154
istio_request_cou mesh",response_	308
istio_request_cou mesh",response_	51
istio_request_cou	50

reviews

VM expansion

- Exposing Pilot, Mixer and Citadel outside of k8s cluster
- Envoy on VM
- Iptables rules



Recap

Istio

Connect, secure, control, and observe services.

- Open Source
- Production-Grade Service mesh
- Scales your control, not your ops team
- Runs anywhere (on-prem, any public cloud)
- Istio Features
    - Service discovery and load balancing
    - Black belt routing capabilities
    - Failure handling
    - Failure ingestion
    - Service and user Identities + RBAC
    - mTLS
    - Tracing
    - Telemetry
    - Central logging

Links

Istio.io (official site)
Microservices in the Cloud with Kubernetes and Istio (Google I/O '18) (presentation)
Lightning-Talk Style Demo of Istio and OpenCensus (demo project on github)

That’s all folks!

Q&A time

Feel free to reach out at
linkedin.com/in/plstoyanov