Istio

A modern service mesh

Shriram Rajagopalan
IBM
@rshriram

Louis Ryan
Google
@louiscryan

What is a ‘Service Mesh’ ?

A network for services, not bytes

- Visibility
- Resiliency & Efficiency
- Traffic Control
- Security
- Policy Enforcement

Why do you need this?

- Microservices

Why do you want this?

- Microservices
- Infrastructure Bloat **X** Polyglot

Why do you want this?

- Microservices
- Infrastructure Bloat **X** Polyglot
- Operational Velocity

Why do you want this?

- Microservices
- Infrastructure Bloat **X** Polyglot
- Operational Velocity
- Control

What is a ‘Service Mesh’ ?

A network for services, not bytes

- Visibility
- Resiliency & Efficiency
- Traffic Control
- Security
- Policy Enforcement

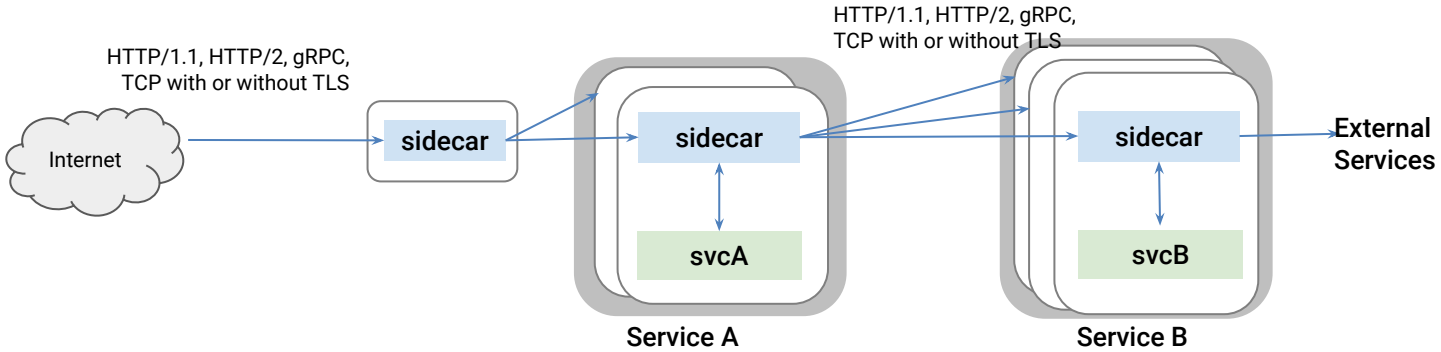
So you want to build a service mesh?

You need control over load balancing. But stop (mis)using the kernel for it!

Lightweight sidecars to manage traffic between services

Sidecars can do ***much more*** than just load balancing!

Weaving the mesh



Outbound features:

- Service authentication
- Load balancing
- Retry and circuit breaker
- Fine-grained routing
- Telemetry
- Request Tracing
- Fault Injection

Inbound features:

- Service authentication
- Authorization
- Rate limits
- Load shedding
- Telemetry
- Request Tracing
- Fault Injection

Our sidecar of choice - Envoy

- A C++ based L4/L7 proxy
- Low memory footprint
- Battle-tested @ Lyft
  - 100+ services
  - 10,000+ VMs
  - 2M req/s

Plus an awesome team willing to work with the community!

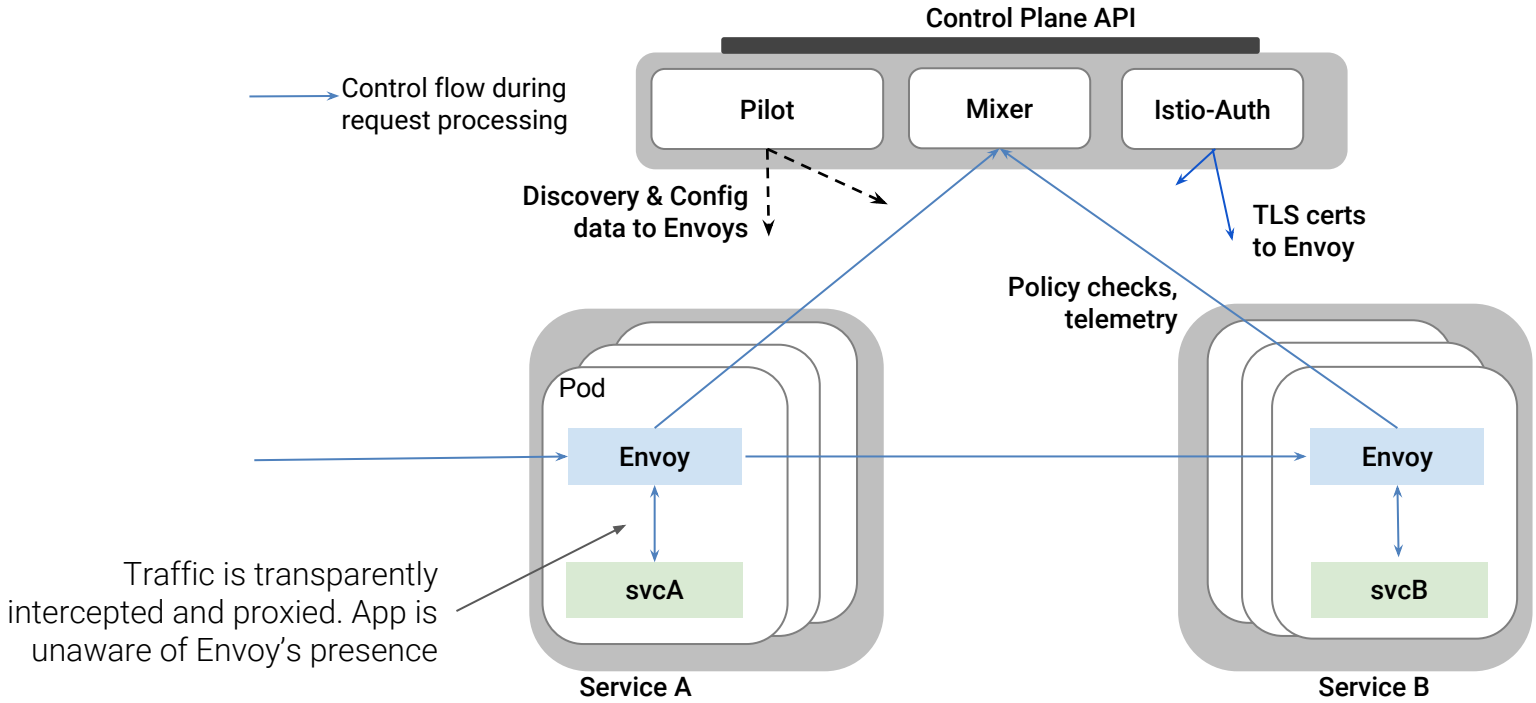
Goodies:

- HTTP/2 & gRPC
- Zone-aware load balancing w/ failover
- Health checks, circuit breakers, timeouts, retry budgets
- No hot reloads - API driven config updates

Istio's contributions:

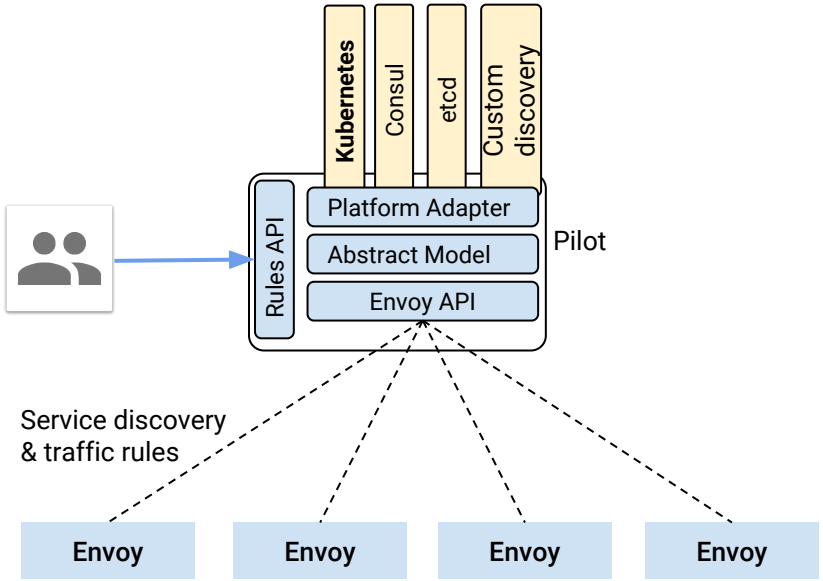
- Transparent proxying w/ SO_ORIGINAL_DST
- Traffic routing and splitting
- Request tracing using Zipkin
- Fault injection

Putting it all together



Modeling the Service Mesh

1. Environment-specific topology extraction
2. Topology is mapped to a platform-agnostic model.
3. Additional rules are layered onto the model. E.g. retries, traffic splits etc.
4. Configuration is pushed to Envoy and applied without restarts



What is a ‘Service Mesh’ ?

A network for services, not bytes

- **Visibility**
- Resiliency & Efficiency
- Traffic Control
- Security
- Policy Enforcement

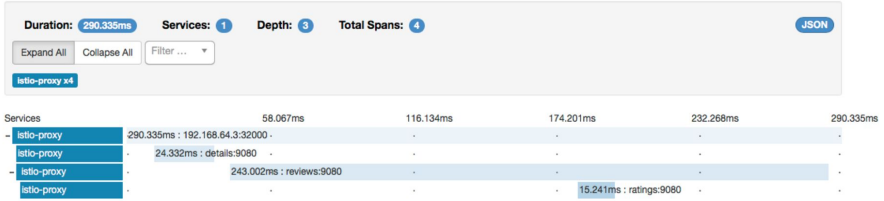
Visibility

Monitoring & tracing should not be an afterthought in the infrastructure

Goals

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

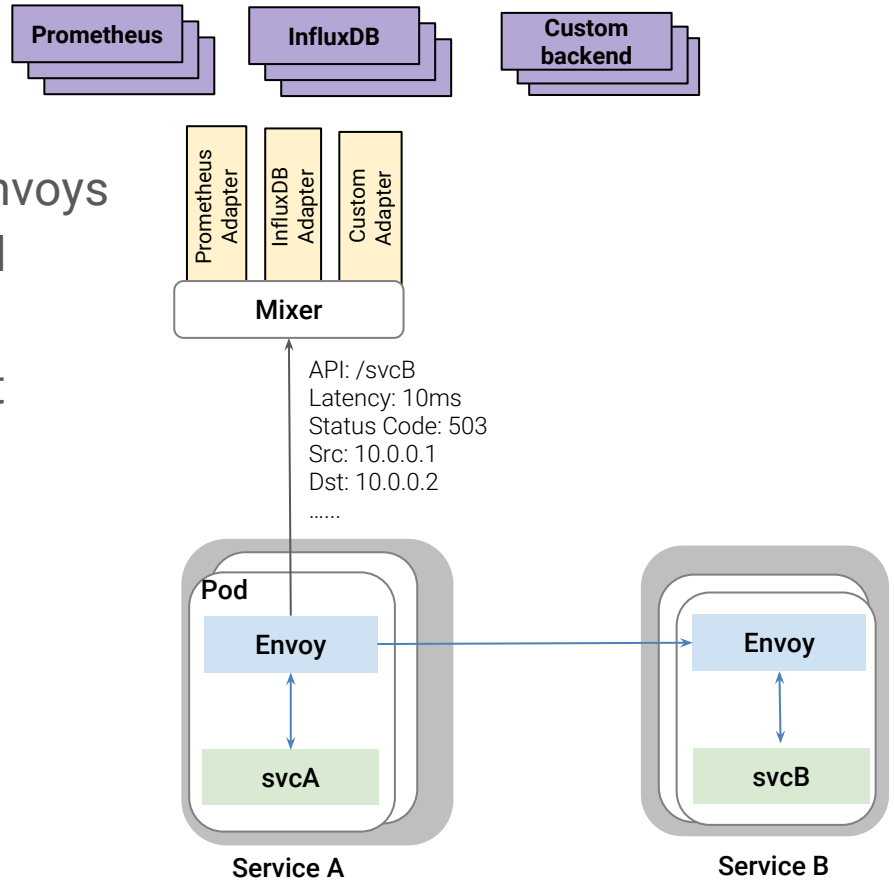
Istio - Grafana dashboard w/ Prometheus backend



Istio Zipkin tracing dashboard

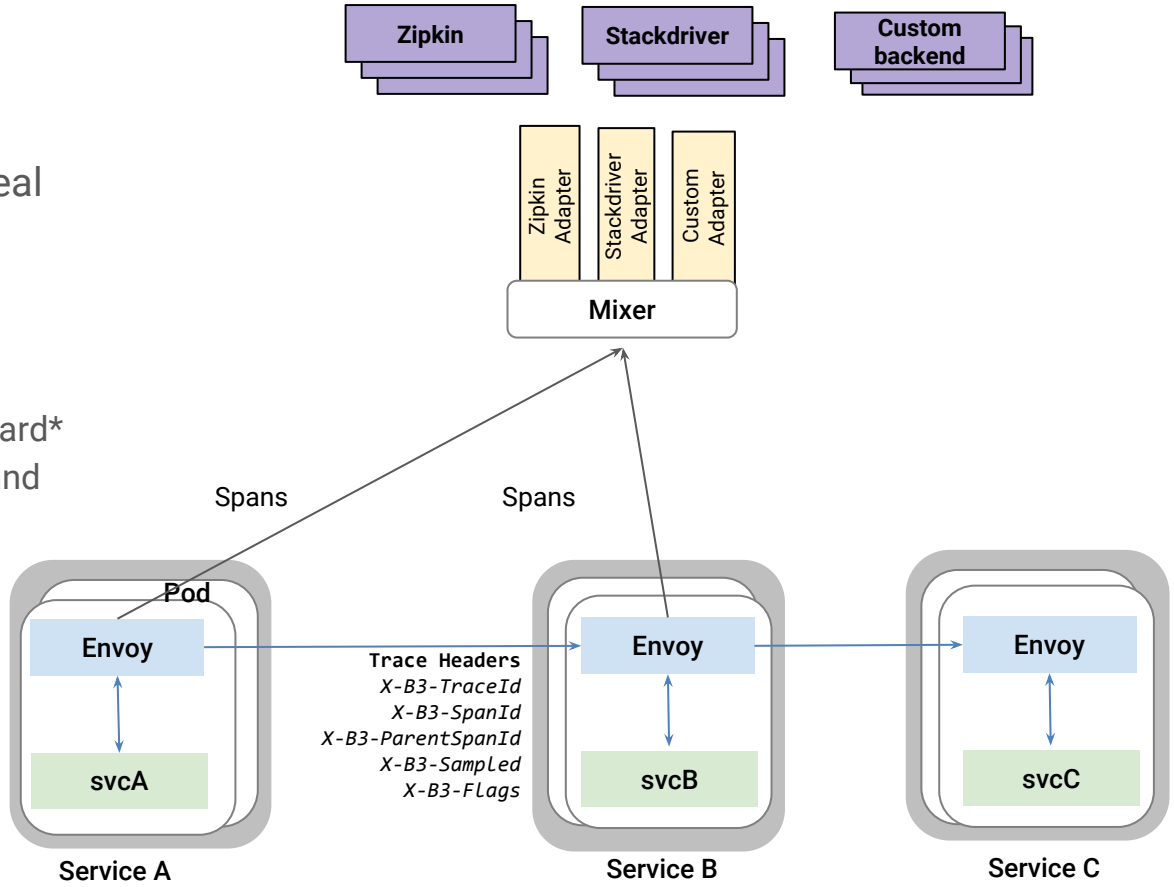
Metrics flow

- Mixer collects metrics emitted by Envoys
- Adapters in the Mixer normalize and forward to monitoring backends
- Metrics backend can be swapped at runtime



Visibility: Tracing

- Application 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



What is a ‘Service Mesh’ ?

A network for services, not bytes

- Visibility
- **Resiliency & Efficiency**
- Traffic Control
- Security
- Control

Resiliency

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

```
// Circuit breakers

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

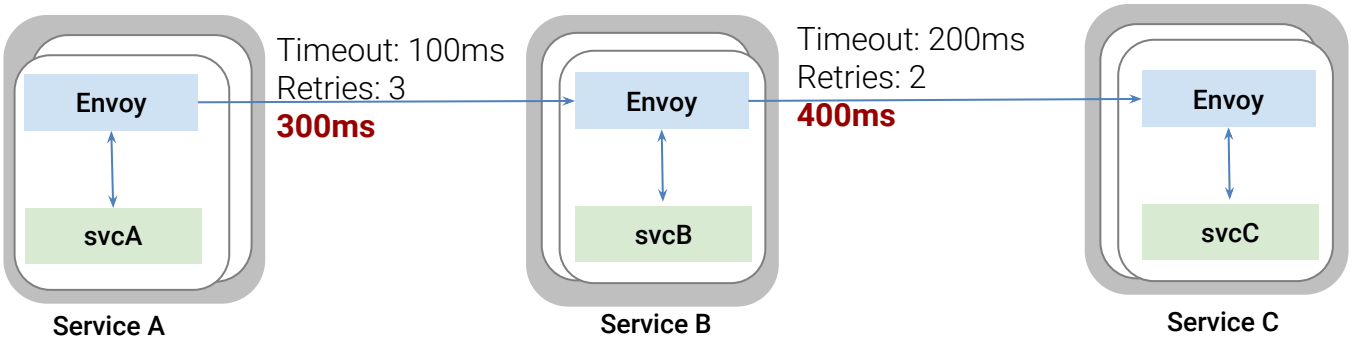
Resilience features

- Timeouts
- Retries with timeout budget
- Circuit breakers
- Health checks
- AZ-aware load balancing w/ automatic failover
- Control connection pool size and request load
- 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
- Inter-service communication happens over HTTP/2
  - HTTP/1.1 connections are transparently upgraded
  - QUIC on the roadmap
- TLS offload
  - No more JSSE or stale SSL versions.
- HTTP/2 and gRPC proxying

What is a ‘Service Mesh’ ?

A network for services, not bytes

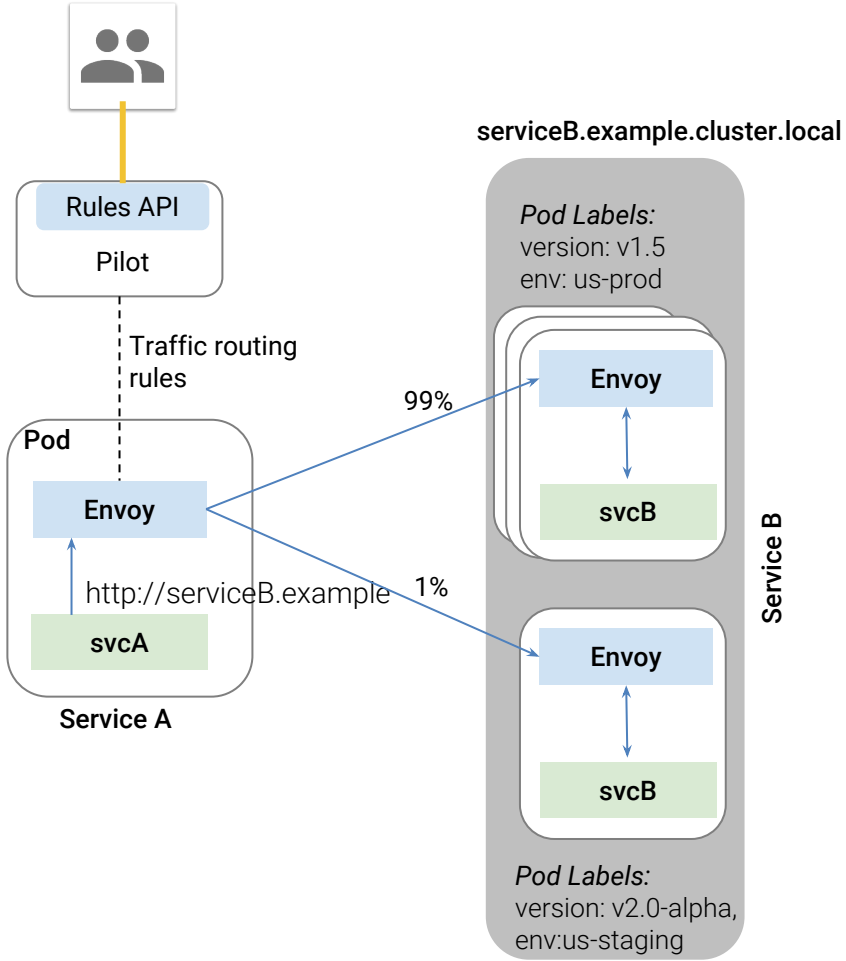
- Visibility
- Resiliency & Efficiency
- **Traffic Control**
- Security
- Policy Enforcement

Traffic Splitting

```
// A simple traffic splitting rule

destination: serviceB.example.cluster.local
match:
  source: serviceA.example.cluster.local
route:
- tags:
    version: v1.5
    env: us-prod
  weight: 99
- tags:
    version: v2.0-alpha
    env: us-staging
  weight: 1
```

Traffic control is decoupled from infrastructure scaling

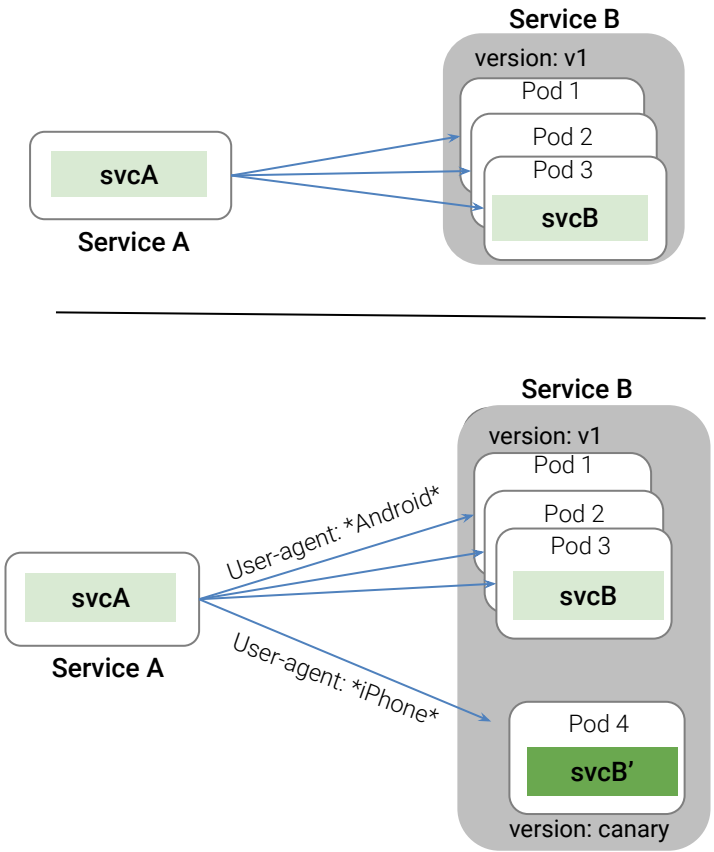


Traffic Steering

```
// Content-based traffic steering rule

destination: serviceB.example.cluster.local
match:
  httpHeaders:
    user-agent:
      regex: ^(.*?;)?(iPhone)(;.*)?$
precedence: 2
route:
- tags:
    version: canary
```

Content-based traffic steering



What is a ‘Service Mesh’ ?

A network for services, not bytes

- Visibility
- Resiliency & Efficiency
- Traffic Control
- **Security**
- Policy Enforcement

Securing Microservices

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

Problem: Strong Service Security at Scale

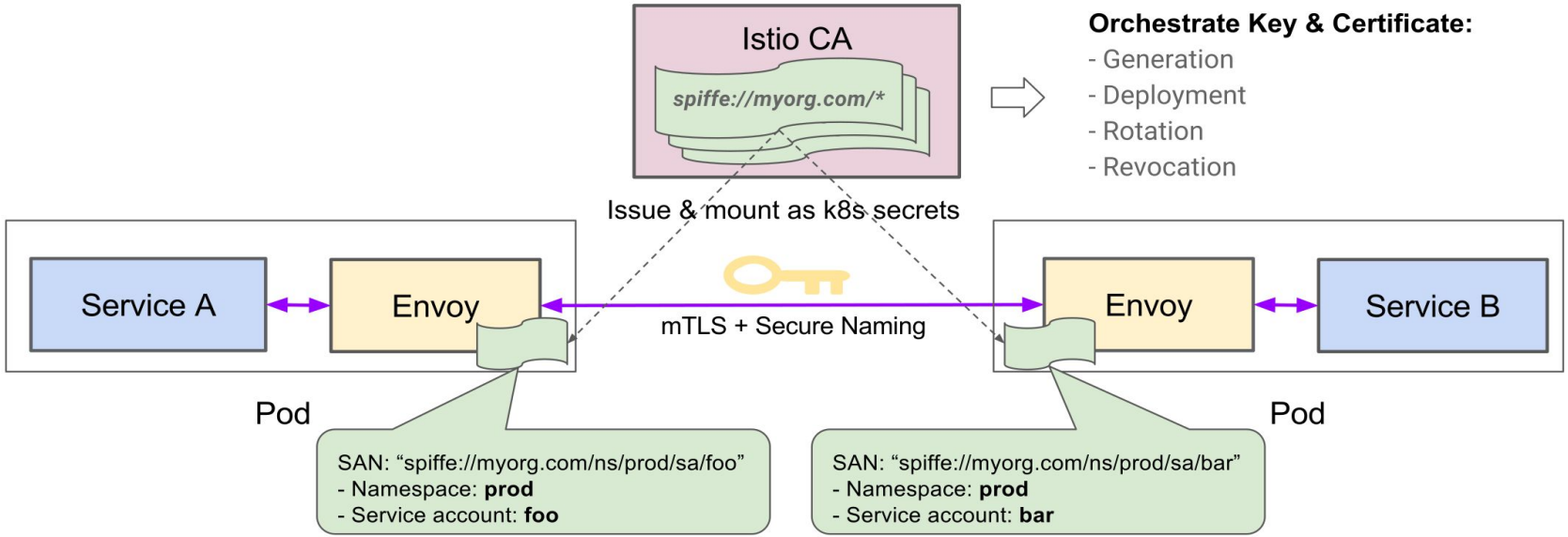
Concerns

- Concerned about insider access risks
- Adopting a (micro-)services architecture
- Audit & Compliance

Issues

- Modern architectures are based on dynamically placed workloads and remotely accessed shared (micro-)services.
- Existing network based security paradigms either enable broad access within a network or are brittle / hard to manage.
- Customers want a way to limit sensitive data access to only limited services (or identities) and enforce strong authentication at scale.

Istio - Security at Scale



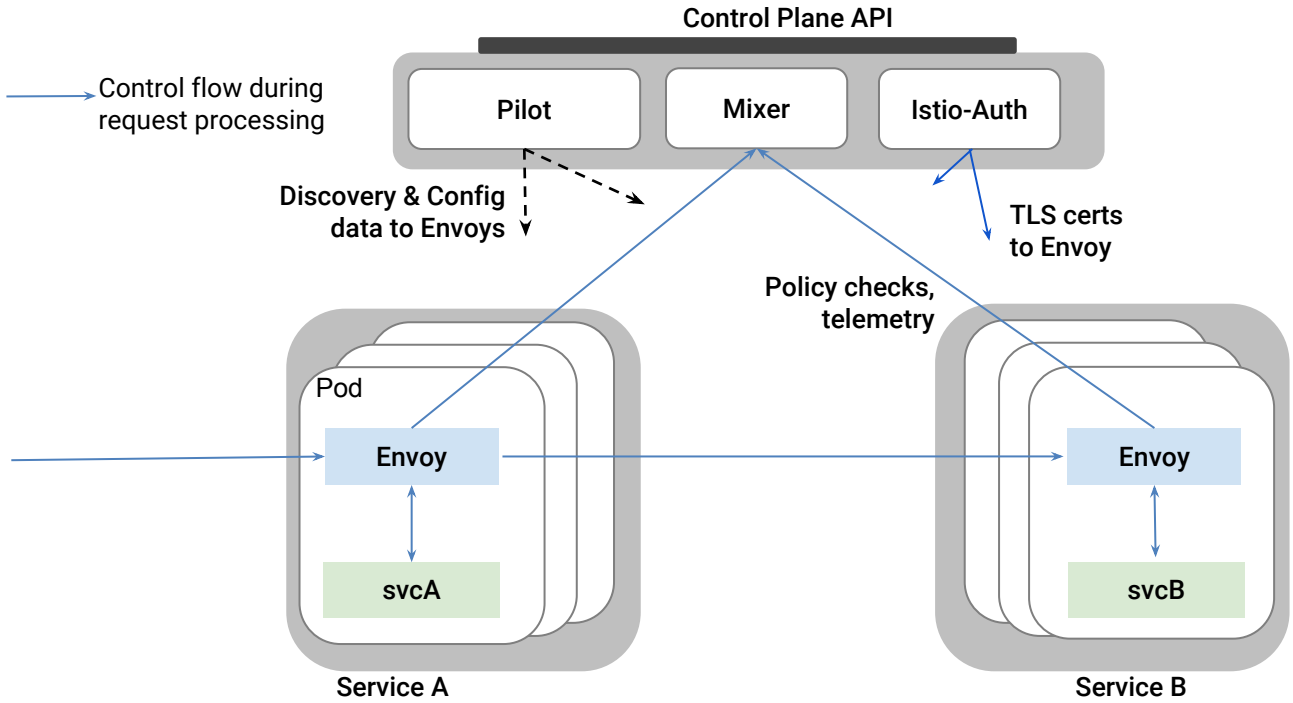
spiffe.io

What is a ‘Service Mesh’ ?

A network for services, not bytes

- Visibility
- Resiliency & Efficiency
- Traffic Control
- Security
- **Policy Enforcement**

Putting it all together

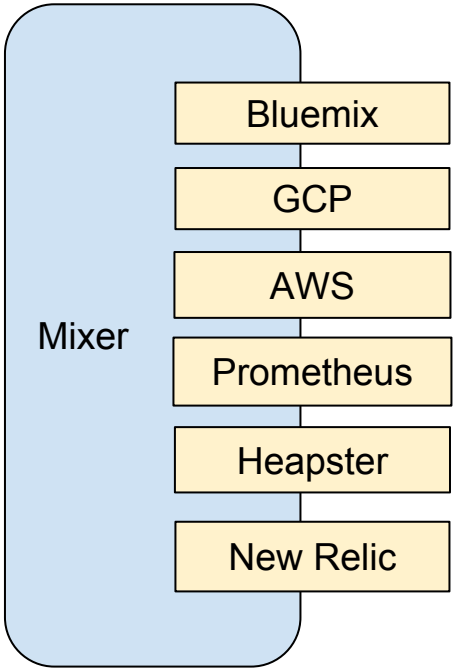


What’s Mixer For?

- Nexus for policy evaluation and telemetry reporting
  - Precondition checking
  - Quotas & Rate Limiting
- Primary point of extensibility
- Enabler for platform mobility
- Operator-focused configuration model

Plugin Model for Extensibility

- Mixer uses pluggable *adapters* to extend its functionality
  - Adapters are modules that interface to infrastructure backends
  - They expose specialized interfaces (logging, metrics, quotas, etc)
  - Multi-interface adapters are possible (e.g., a Stackdriver adapter exposing logging & monitoring)
- Adapters run within the Mixer process



Attributes - The behavioral vocabulary

```
target.service = “playlist.svc.cluster.local”
request.size   = 345
request.time   = 2017-04-12T12:34:56Z
source.ip      = 192.168.10.1
source.name    = “music-fe.serving.cluster.local”
source.user    = “admin@musicstore.cluster.local”
api.operation  = “GetPlaylist”
```

Attributes

- Typed name-value tuples that describe behaviors within the mesh
  - Base vocabulary
  - Extensible
- Envoy and Services produce attributes, Mixer consumes them
- Attributes are fundamental to how operators experience Istio

Roadmap

- More networking features - UDP, Payload transforms, Websocket, Global LB
- VMs and other environments
- Hybrid cloud & federation
- Value-add integrations - ACLs, Telemetry, Audit, Policy, ...
- Security - vTPM/HSM & Cert stores, Federation, Cloud Platforms, ...
- Stability

Community Partners

- RedHat
- Pivotal
- WeaveWorks
- Tigera
- Datawire
- Scytale (SPIFFE)

… and you!

Thanks! Phew