Istio and Envoy: Enabling Sidecars for Microservices

Angela Chin
Senior Software Engineer, Pivotal
Outline

• What your microservices need
• How sidecars help
• Envoy & Istio
• Kubernetes Integration
• Cloud Foundry Integration
Outline

• What your microservices need
• How sidecars help
• Envoy & Istio
• Kubernetes Integration
• Cloud Foundry Integration
So you’re building some microservices...
Monolith
Things your microservices ought to do...

- Client-side retries
- Load balancing
- Mutual TLS
- Configurable timeouts
- Collecting metrics
Load balancing
Client-side load-balancing

Frontend

Backend Instance 1

Backend Instance 2

Backend Instance 3
Client-side load-balancing

- Frontend
- Backend Instance 1
- Backend Instance 2
- Backend Instance 3
Client-side load-balancing
Client-side load-balancing

Frontend

Backend Instance 1

Backend Instance 2

Backend Instance 3
Client-side load-balancing
Client-side load-balancing

Frontend

 Backend Instance 1

 Backend Instance 2

 Backend Instance 3
Retries
Retries

Request

Frontend

Backend

Instance 1

Backend

Instance 2

Backend

Instance 3
Retries

Request

Frontend

Backend
Instance 1

Backend
Instance 2

Backend
Instance 3
Retries

Frontend

Request

data plz?

Backend
Instance 1

Backend
Instance 2

Backend
Instance 3
Retries

- Frontend
- Backend Instance 1
- Backend Instance 2
- Backend Instance 3

Request: data plz?

Error: unavailable!
Retries

```
Retries

Frontend

Backend Instance 1

Backend Instance 2

Backend Instance 3

Request

data plz?
```
Retries

Frontend

Backend
Instance 1

Backend
Instance 2

Backend
Instance 3

Request

data plz?
data!
data!
Retries

Request

Frontend

Response

data plz?

data!
Transport Security

Frontend

Backend
Transport Security

Server Certificate
Name: Backend
Transport Security

- **Client Certificate**
  - Name: Frontend

- **Server Certificate**
  - Name: Backend
Transport Security

Frontend

Client Certificate
Name: Frontend

Expect Server
Name: backend

Backend

Server Certificate
Name: Backend

Allowed Client
ACME Corp
How do application developers usually get these features?
In-process architecture

Frontend — A Library! — Backend
Multiple languages, Multiple Libraries

Each with…

- Different features
- Different configuration
- Different quirks

*Polyglot shouldn't be painful!*
Outline

• What your microservices need
• **How sidecars help**
• Envoy & Istio
• Kubernetes Integration
• Cloud Foundry Integration
Out-of-process architecture

Your App → Separate process! → Some service
Out-of-process architecture

Your App

- retries
- load balancing
- mutual TLS
- timeouts
- metric collection
- etc.

Your business logic here
Out-of-process architecture

Client app

Your App (acting as a service)
What’s a sidecar?

- A separate process
- Runs **alongside** a microservice
- A **proxy** reachable via **localhost**
  - Layer 4 (TCP)
  - Layer 7 (HTTP)
- Can proxy **Ingress** and **Egress**
- Provides all those features we want!
When every component has a sidecar…
we get a **SERVICE MESH**
Client-side load-balancing

Connection #1!

Frontend

Sidecar

Backend 1

Sidecar

Backend 2

Sidecar

Backend 3

Sidecar
Client-side load-balancing

Frontend

Sidecar

Connection #2!

Sidecar

Backend 1

Backend 2

Backend 3
Client-side load-balancing

Frontend

Sidecar

Connection #3!!!

Backend 1

Sidecar

Backend 2

Sidecar

Backend 3

Sidecar
Client-side load-balancing

- Frontend
- Sidecar
- Backend 1
- Backend 2
- Backend 3

Load distribution:
- 80% to Backend 1
- 10% to Backend 2
- 10% to Backend 3
Outline

• What your microservices need
• How sidecars help
• **Envoy & Istio**
• Kubernetes Integration
• Cloud Foundry Integration
What is Envoy?

- L4 / L7 Proxy
- C++
- Low memory footprint
- Designed for service mesh
- Dynamic configuration
What is Envoy?

- L4 / L7 Proxy
- C++
- Low memory footprint
- Designed for service mesh
- Dynamic configuration

- Built by Lyft
- Open source
- Part of the Cloud Native Computing Foundation
Envoy data path
Envoy control plane

Configuration Server

incoming requests

listeners

filter chains

cluster definitions

outgoing requests
Envoy control plane

Configuration Server

Envoy API

listeners

filter chains

cluster definitions
We need something to serve the dynamic configuration
What is Istio

- Envoy control plane
- Go
- Runs on Kubernetes today…
  - community wants it to be cross-platform
What is Istio

- Envoy control plane
- Go
- Runs on Kubernetes today…
  - community wants it to be cross-platform

- Built by Google w/ IBM
- Open source
Using Istio

- Provide yaml files to Istio to set up routing rules to services
- Domain-specific language (DSL) to control API and inter-service connectivity
- Rules are qualified by destination service name or source identifiers
mTLS

Frontend

Envoy

Istio

Backend 1

Envoy

Backend 2

Envoy

Backend 3

Envoy
mTLS

Frontend

Envoy

Istio

Backend 1

Envoy

Backend 2

Envoy

Backend 3

Envoy

mTLS?
mTLS

Frontend

Envoy

Istio

mTLS!

Backend 1

Envoy

Backend 2

Envoy

Backend 3

Envoy
mTLS

Frontend

Envoy

Backend 1

Envoy

Backend 2

Envoy

Istio
mTLS

Frontend

Envoy

mTLS?

Backend 1

Envoy

Backend 2

Envoy

Istio
Retries

subscribe!
Retries

Frontend

Envoy

Istio

Backends!

Backend 1

Envoy

Backend 2

Envoy

Backend 3

Envoy
Retries
Retries

Frontend

Envoy

Istio

Envoy

Backend 1

Envoy

Backend 2

Envoy

Backend 3

Envoy
Retries

Frontend

Envoy

Istio

Envoy

Backend 1

Envoy

Backend 2

Envoy

Backend 3
Istio Terminology

- **Service**: unit of (destination) application.
- **Service Versions**: variants of (destination) application binary.
- **Service Registry**: keeps track of pods/VMs of a service
- **Route Rule/Virtual Service**: rule defining what service to send a request to
- **Destination Rule**: rule defining what to do after destination service is identified
Example Route Rule Config

```yaml
apiVersion: config.istio.io/v1alpha2
kind: RouteRule
metadata:
  name: reviews-default
spec:
  destination:
    name: reviews
  route:
    - labels:
        version: v1
        weight: 100
```
Weighted Routing

apiVersion: config.istio.io/v1alpha2
kind: RouteRule
metadata:
  name: reviews-v2-rollout
spec:
  destination:
    name: reviews
  route:
    - labels:
      version: v2
      weight: 25
    - labels:
      version: v1
      weight: 75
What does this translate to in Envoy?

```
"virtual_host": [{
    "domain": ["serviceB.example.cluster.local"],
    "route": [{
        "prefix": "/",
        "weighted_clusters": [
            {"name": "cluster_v1", "weight": 75},
            {"name": "cluster_v2", "weight": 25}
        ]
    }]
}]
```
Example Destination Rule Config

```yaml
apiVersion: config.istio.io/v1alpha2
kind: DestinationPolicy
metadata:
  name: reviews-v1-cb
spec:
  destination:
    name: reviews
  labels:
    version: v1
circuitBreaker:
  simpleCb:
    maxConnections: 100
```
Partial Rollout

- Frontend
- Envoy

Istio

subscribe!

- Backend 1
- Envoy

- Backend 2
- Envoy

v1

v2
Partial Rollout

Istio

Frontend

Envoy

Envoy

Backend 1

Backend 2

v1

v2
Partial Rollout

- Frontend
- Envoy
- Envoy
- Backend 1
- Backend 2
- Istio

75% to Envoy
25% to Envoy
How does Istio actually work?

- **Pilot**
- **Mixer**
- **Auth**

Envoy config

Policy checks

TLS certs

Envoy

App A

Envoy

App B
Istio Pilot Architecture

Cloud Platform
- Platform Adapter
- Abstract Model
- Envoy API

Pilot

Envoy
- App A

Envoy
- App B
Outline

• What your microservices need
• How sidecars help
• Envoy & Istio
• Kubernetes Integration
• Cloud Foundry Integration
Kubernetes Integration

- Envoy per pod
- By default, all HTTP traffic proxied through Envoy via iptables (for now!)
- Platform adapter transforms information from Kubernetes API server
- Rules are qualified by destination service name or source identifiers
Outline

• What your microservices need
• How sidecars help
• Envoy & Istio
• Kubernetes Integration
• Cloud Foundry Integration
Cloud Foundry Integration

- Envoy per container
- Only specific ingress traffic routed through Envoy (for now!)
- Platform adapter transforms information from Cloud Controller and Diego components
TLS b/w Router and Application

Load Balancer

GoRouter

Envoy

App A

Envoy

App B
Envoy for Gateway

Load Balancer

Envoy

TLS

App A

Envoy

TLS

App B

Pilot

Platform Adapter

Abstract Model

Envoy API
The Future
Partial Rollout

Istio

subscribe!

Service

Backend 1

Envoy

Frontend

Envoy
Partial Rollout

Istio

subscribe!

Frontend

Envoy

100%

Envoy

Backend 1

Service
Partial Rollout

Istio

Frontend

Envoy

100%

backends!

Service

Backend 1

Envoy

Backend 2

Envoy
Partial Rollout

Istio

Frontend

Envoy

100%

Backend 1

Envoy

Backend 2

Service

v1

backends!
Partial Rollout

Frontend

Envoy

100%

Istio

Service

Backend 1

Envoy

Backend 2

Envoy

v1

v2

backends!
Partial Rollout

- Frontend
- Envoy
- Envoy
- Backend 1
- Backend 2
- Envoy
- Envoy
- Istio
- Service

75% to Backend 1

25% to Backend 2
Partial Rollout

Frontend
Envoy

25%

Envoy
Backend 1

Envoy
Backend 2

75%

Istio
Service

v1

v2
Partial Rollout
Partial Rollout

Istio

Service

Frontend

Envoy

25%

Backend 1

Envoy

Backend 2

Envoy

v1

v2

healthy?
Partial Rollout

Istio

Frontend

Envoy

Envoy

Backend 1

Backend 2

Service

v1

v2

50%

50%
Partial Rollout

Frontend

Envoy

50%

Backend 1

Envoy

Backend 2

Envoy

Istio

Service

50%

healthy?
Partial Rollout

Frontend

Envoy

50%

Backend 1

Envoy

Backend 2

Envoy

50%

Istio

Service

healthy?
Partial Rollout

- Frontend
- Envoy

25% to Backend 1

75% to Backend 2

Istio

Service
Partial Rollout

Frontend

Envoy

Envoy

Backend 1

Backend 2

Envoy

Istio

Service

100%
Discovery across platforms
Conclusion

- Envoy is a L4/L7 proxy designed to run as a sidecar
- Istio is a sidecar control-plane
- Sidecars & Service Meshes make developers and operators happy!
Further reading

• Envoy: https://www.envoyproxy.io/docs/envoy/latest/
• Istio: https://istio.io/docs/concepts/
• Istio Example App: https://istio.io/docs/guides/bookinfo.html
Thank you!

achin@pivotal.io

We’re hiring: https://pivotal.io/careers
Service Mesh versus SDN

- Programmability
- Configuration
- Multi-tenancy