Sidecar All The Things!

Envoy, Istio & Cloud Foundry

Angela Chin
Software Engineer
Pivotal

Gabe Rosenhouse
Software Engineer
Pivotal
Outline

• What your microservices need
• What Cloud Foundry provides today vs what is missing
• Sidecars & how they help
• Envoy & Istio
• Work in-progress and what's next
Outline

- **What your microservices need**
- What Cloud Foundry provides today vs what is missing
- Sidecars & how they help
- Envoy & Istio
- Work in-progress and what's next
So you're building some microservices...
Things your microservices probably ought to do...

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

Request

Frontend

Backend
Instance 1

Backend
Instance 2

Backend
Instance 3
Retries

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

Request → data plz?
Retrieves

Request

Frontend

Backend

Instance 1

Instance 2

Instance 3
Retries

Request → Frontend → Backend (Instance 1)

Error: unavailable!

Request → Backend (Instance 2)

Request → Backend (Instance 3)

data plz?
Retries

Request — Frontend

Backend

Instance 1

Instance 2

Instance 3

Data plz?
Retries

Request

Frontend

Backend
Instance 1

Backend
Instance 2

Backend
Instance 3

data plz?
data!
data!
Retries

Request data plz?

Response data!

Frontend

Backend Instance 1

Backend Instance 2

Backend Instance 3
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
Transport Security
Transport Security

Frontend

Backend
Transport Security

Server Certificate
Name: Backend
Transport Security

Client Certificate
Name: Frontend

Server Certificate
Name: Backend
**Transport Security**

- **Client Certificate**
  - Name: Frontend

- **Expect Server**
  - Name: backend

- **Server Certificate**
  - Name: Backend
  - Allowed Client
    - ACME Corp
Outline

• What your microservices need
• **What Cloud Foundry provides today vs what is missing**
• Sidecars & how they help
• Envoy & Istio
• Work in-progress and what's next
How can we get these features today?
Orientation

public internet

your data center
or
virtual private cloud
Orientation

public internet

Load Balancer

CF Router

your data center
or
virtual private cloud

Internal Services
Apps
Orientation

- **Load Balancer**
  - **CF Router**
  - **Internal Services**
  - **Apps**

- **public internet**

- **your data center** or **virtual private cloud**
Orientation

- Your data center
- Public internet
- Your data center or virtual private cloud

Diagram:
- Load Balancer
- CF Router
- Internal Service
- App
- App
## CF Router

<table>
<thead>
<tr>
<th>Feature</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retries</td>
<td>✓</td>
</tr>
<tr>
<td>Load balancing</td>
<td>✓</td>
</tr>
<tr>
<td>Configurable timeouts</td>
<td>✗</td>
</tr>
<tr>
<td>Metrics collection</td>
<td>✓</td>
</tr>
<tr>
<td>Mutual TLS</td>
<td>✗</td>
</tr>
<tr>
<td>0-downtime app updates</td>
<td>Planned, but tricky</td>
</tr>
</tbody>
</table>

### Diagram

- **Load Balancer**
- **CF Router**
  - a.k.a. "gorouter"
- **Apps**
App to app communication via **CF Router**
App to app communication via CF Router

Load Balancer

CF Router
a.k.a "gorouter"

App A

App B

Anything can reach App B!
App to app communication via **Container Networking**
## Container Networking

### CF Router
- a.k.a "gorouter"

### Load Balancer

### App A

### App B

<table>
<thead>
<tr>
<th>Feature</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retries</td>
<td>✗</td>
</tr>
<tr>
<td>Load balancing</td>
<td>✗</td>
</tr>
<tr>
<td>Configurable timeouts</td>
<td>✗</td>
</tr>
<tr>
<td>Metrics collection</td>
<td>✗</td>
</tr>
<tr>
<td>Mutual TLS</td>
<td>✗</td>
</tr>
<tr>
<td>0-downtime app updates</td>
<td>✗</td>
</tr>
</tbody>
</table>
But wait...

what about a library in my app?
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
• What Cloud Foundry provides today vs what is missing
• Sidecars & how they help
• Envoy & Istio
• Work in-progress and what's next
A better approach...
Out-of-process architecture

Your App

Some service

Separate process!
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!
Client-side load-balancing

Connection #2!
Client-side load-balancing
Client-side load-balancing

Frontend

Sidecar

Backend 1

Sidecar

Backend 2

Sidecar

Backend 3
<table>
<thead>
<tr>
<th>Feature</th>
<th>Go Router (North/South)</th>
<th>Container Network (East/West)</th>
<th>... with Sidecars</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retries</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Load balancing</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Configurable timeouts</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Metrics collection</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Mutual TLS</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>0-downtime app updates</td>
<td>Planned, but tricky</td>
<td>✗</td>
<td>Still kinda tricky</td>
</tr>
</tbody>
</table>

Notes:
- The table shows features and their availability in different components.
- Go Router supports retries and load balancing.
- Container Network does not support retries or load balancing.
- With sidecars, all features are supported except for retries.

**Plan:**
- The future planning includes support for retries (Planned) with sidecars.

**Complexity:**
- Mutual TLS is supported in all configurations.
- 0-downtime app updates are currently planned but still tricky.
How sidecars help Cloud Foundry
Example: "Route Integrity"

*If the control plane suffers an outage, how do we avoid mis-routing requests?*
Example: "Route Integrity"

How does the routing control plane work today?
GoRouter

HTTP Proxy

Route Table

Message Bus
("NATS")

Port 1

Port 2

Diego Cell 1

App A

Container

App B

Container

Diego Cell 2

App B

Container

App C

Container

cell1:port1 = app-a.example.com
cell1:port2 = app-b.example.com
cell2:port1 = app-b.example.com
cell2:port2 = app-c.example.com

Route Emitter

Route Emitter

Port 1

Port 2

Message Bus
("NATS")

Port 1

Port 2

GoRouter

HTTP Proxy

Route Table

Diego Cell 1

App A

Container

App B

Container

Diego Cell 2

App B

Container

App C

Container

cell1:port1 = app-a.example.com
cell1:port2 = app-b.example.com
cell2:port1 = app-b.example.com
cell2:port2 = app-c.example.com
app-a.example.com: [cell1:port1]
app-b.example.com: [cell1:port2, cell2:port1]
app-c.example.com: [cell2:port2]
GoRouter

HTTP Proxy

Route Table

Message Bus ("NATS")

Diego Cell 1

Route Emitter

App A Container

Port 1

App B Container

Port 2

Diego Cell 2

Route Emitter

App B Container

Port 1

App C Container

Port 2

app-a.example.com: [cell1:port1]
app-b.example.com: [cell1:port2, cell2:port1]
app-c.example.com: [cell2:port2]

App B plz!
Example: Route Integrity

But sometimes the routing control plane becomes unavailable...
GoRouter

HTTP Proxy

Route Table

Message Bus ("NATS")

Diego Cell 1

Route Emitter

Container

App A

Container

App B

Port 1

Port 2

Diego Cell 2

Route Emitter

Container

App B

Container

App C

Port 1

Port 2

App B plz!
GoRouter

HTTP Proxy

Route Table

Message Bus ("NATS")

Diego Cell 1

 Route Emitter

Container
App A

App B

Port 1
Port 2

Diego Cell 2

 Route Emitter

Container
App C

App B

Port 1
Port 2

app-a.example.com: [cell1:port1]
app-b.example.com: [cell1:port2, cell2:port1]
app-c.example.com: [cell2:port2]

App B plz!
GoRouter

HTTP Proxy

Route Table

Message Bus ("NATS")

GoRouter

app-a.example.com: [cell1:port1]
app-b.example.com: [cell1:port2, cell2:port1]
app-c.example.com: [cell2:port2]

Diego Cell 1

GoRouter

Port 1

App A

Container

GoRouter

Port 2

App B

Container

Diego Cell 2

GoRouter

Port 1

App C

Container

GoRouter

Port 2

App B

Container

Diego Cell 1

App B plz!
Example: Route Integrity

How can sidecars help?
GoRouter

HTTP Proxy

Route Table

Message Bus ("NATS")

Diego Cell 1

Route Emitter

App A

App B

Container

Diego Cell 2

Route Emitter

App C

App B

Container

Port 1

Port 2

sidecar proxy

App B plz!
GoRouter

HTTP Proxy

Route Table

Message Bus ("NATS")

Diego Cell 1

Port 1

App A

Container

sidecar proxy

App B

Container

Route Emitter

Diego Cell 2

Port 1

Port 2

App C

Container

sidecar proxy

App B

Container

sidecar proxy

Route Emitter

App B plz!
GoRouter

Route Table

HTTP Proxy

Message Bus ("NATS")

Diego Cell 1

Route Emitter

Container

App A

Port 1

sidecar proxy

Container

App B

Port 2

Diego Cell 2

Route Emitter

Container

App C

Container

App B

Port 1

sidecar proxy

Container

App B

Port 2

Expect Certificate
Server = App B
Container = Foo

Actual Certificate
App ID = App B
Container ID = Foo

App B plz!
GoRouter

HTTP Proxy

Route Table

Message Bus ("NATS")

Diego Cell 1

Route Emitter

Container

App A

Container

App B

Port 1

Port 2

Diego Cell 2

Route Emitter

Container

App C

Container

App B

Port 1

Port 2

App B plz!

Expect Certificate
Server = App B
Container = Foo

Actual Certificate
App ID = App C
Container ID = Bar
GoRouter
HTTP Proxy
Route Table

Message Bus
("NATS")

Diego Cell 1
Route Emitter
Container
App A
Container
App B

Diego Cell 2
Route Emitter
Container
App C
Container
App B

Port 1
1, 2
Port 2

sidecar proxy

App B plz!
Example: Route Integrity

- Add a sidecar proxy to every backend app container
- TLS between Router and sidecar proxy
- Router checks identity of backend
- Prevents mis-routing even when control plane is unavailable
This work is already in-flight!

**Experimental features...**

- Latest Routing Release
  Router does mutual TLS to backends

- Latest Diego Release
  Inject **Envoy** sidecar proxy into app containers
  hosting per-instance identity credentials
Outline

- What your microservices need
- What Cloud Foundry provides today vs what is missing
- Sidecars & how they help
- Envoy & Istio
- Work in-progress and what's next
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
- Recent addition to 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

- listeners
- filter chains
- cluster definitions

Envoy API
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
Istio Use Case

Frontend

Envoy

Istio

Envoy

Backend 1

Envoy

Backend 2

Envoy

Backend 3

Envoy

backends?
Istio Use Case

Frontend

Envoy

backends!

Istio

Backend 1

Envoy

Backend 2

Envoy

Backend 3

Envoy
Istio Use Case

Frontend
Envoy

Backend 1
Envoy

Backend 2
Envoy

Backend 3
Envoy

Istio

mTLS!
Istio Use Case

Frontend

Envoy

Istio

Backend 1

Envoy

Backend 2

Envoy

Backend 3

Envoy

mTLS?
Istio Use Case
Istio Use Case

Frontend

Envoy

Backend 1

Envoy

Backend 2

Envoy

Istio
Istio Use Case

Frontend

Envoy

backends!

Istio

Backend 1

Envoy

Backend 2

Envoy
Istio Use Case

Frontend

Envoy

mTLS?

Backend 1

Envoy

Backend 2

Envoy

Istio
Istio Use Case

Frontend

Envoy

Envoy

Backend 1

Envoy

Backend 2

mTLS!
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
Istio + CF

Diagram:

- CF Routing
  - CF Adapter
  - Abstract Model
  - Envoy API

Pilot

- Envoy
  - App A
- Envoy
  - App B
Outline

• What your microservices need
• What Cloud Foundry provides today vs what is missing
• Sidecars & how they help
• Envoy & Istio
• Work in-progress and what's next
Work in-flight

- Diego
  - [done] Every container gets unique cert & key encoding CF App ID
  - [experimental] Every container gets an Envoy
- Routing
  - [experimental] mutual TLS to backends
- Container Networking
  - [in-flight] DNS for apps on the container network
What we (might) do next

• Garden: Pods!
• Diego: Envoy in the Pod
• Container Networking: Client-side load-balancing w/ Envoy
• Routing: Envoy as edge router?
Possibilities for the future

• Envoy co-located with BOSH-deployed Service VMs
• Envoy as Egress proxy (NAT box)
• Expose Istio functionality to CF Operators or Developers?
How would Istio work on CF?

Routing tier

Istio

Tenant 1

Tenant 2
How would Istio work on CF?

Routing tier

Tenant 1

Istio

Tenant 2

Istio
Takeaways

Sidecars make developers and operators happy!

Envoy is a L4/L7 proxy designed to run as a sidecar
   We're integrating Envoy into CF application instances.

Istio is a sidecar control-plane
   We are exploring how to integrate Istio into CF.
Acknowledgements

CF Community
- Diego
- Networking
- Routing

Envoy Community

Istio Community
We want feedback!

- What's your use case?
- What integrations do you want to see?

Cloud Foundry Slack: #sidecars