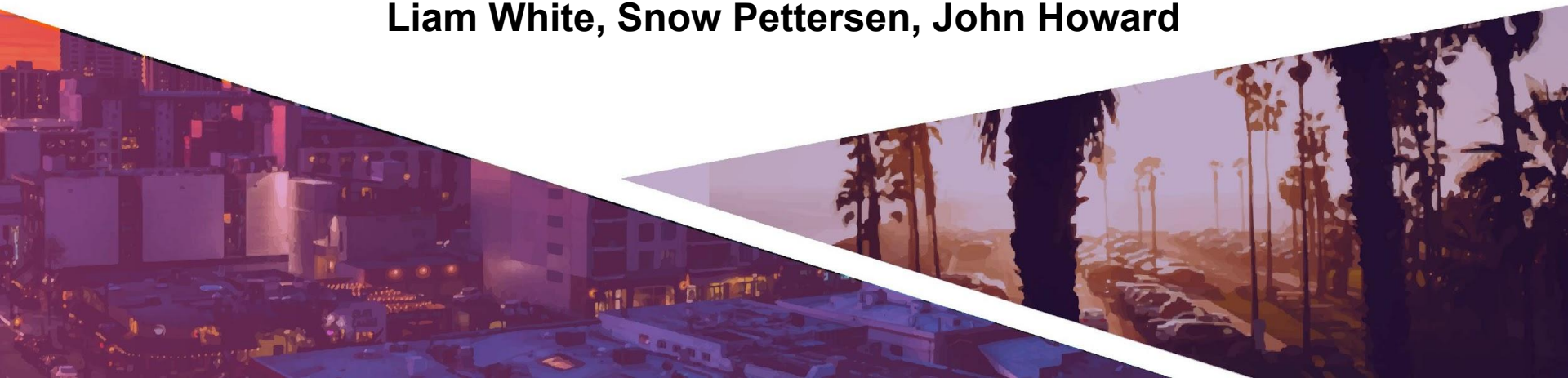
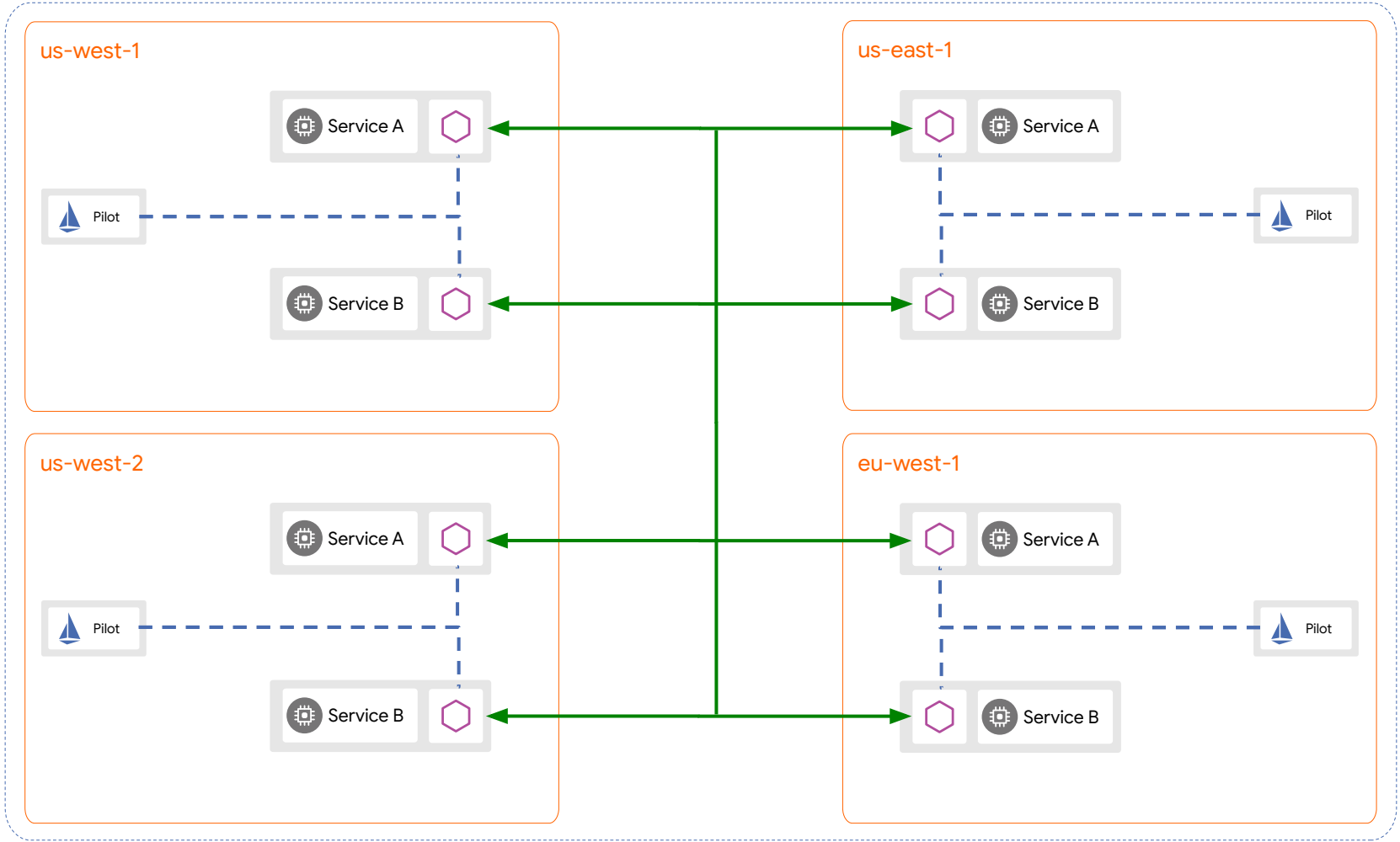
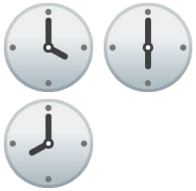
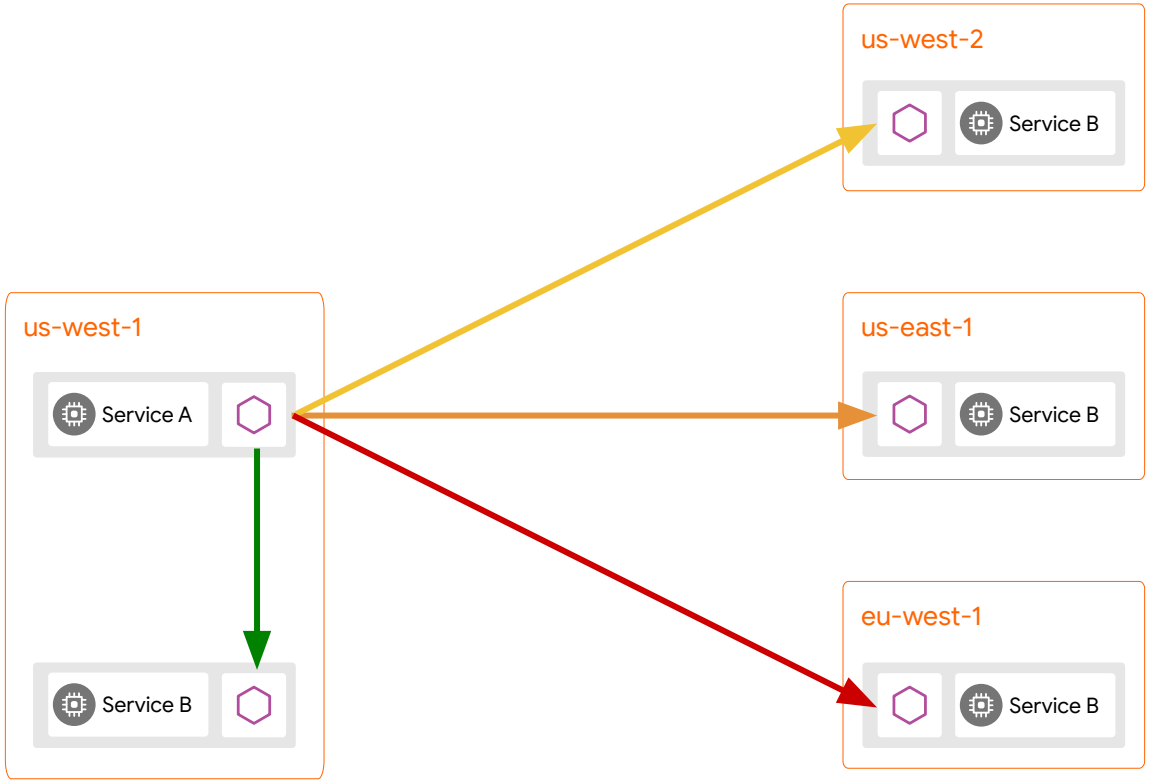
Building Low Latency Topologies with Envoy

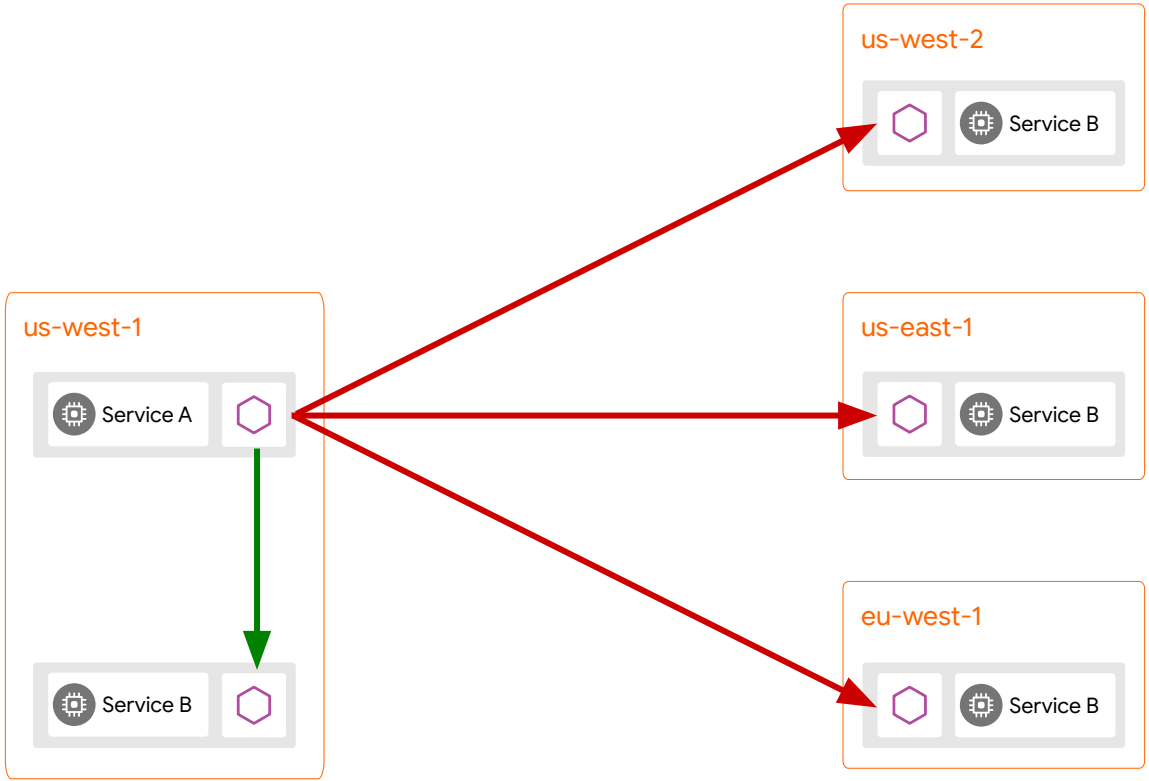
Liam White, Snow Pettersen, John Howard




us-west-1
Service A
Pilot
Service B
us-east-1
Service A
Pilot
Service B
us-west-2
Service A
Pilot
Service B
eu-west-1
Service A
Pilot
Service B




us-west-2
Service B
us-west-1
Service A
Service B
us-east-1
Service B
eu-west-1
Service B


us-west-2
Service B
us-west-1
Service A
Service B
us-east-1
Service B
eu-west-1
Service B

Corey Quinn @QuinnyPig · Sep 6

Once people get their cross-AZ data transfer bills from their microservices architectures, monoliths will come crashing back so hard you'll swear you were an actor in 2001: A Space Odyssey.

20 50 311

Matthew S. Wilson @_msw_ · Sep 6

Don't think that microservice architectures should lead to cross AZ traffic. Building well architected, resilient systems would.

4 9

Jacob at #ChaosConf @jhscott · Sep 6

I believe these bills motivated the development of zone aware load balancing in @EnvoyProxy: envoyproxy.io/docs/envoy/v1....

2 6

Matthew S. Wilson @_msw_ · Sep 6

It's not just good for bills. I think it *can* be a more resilient and reliable architecture to have services composed as microservices to keep internal requests within an AZ, while using a regional level service for state.

1 2

Jacob at #ChaosConf @jhscott · Sep 6

Ah def. bills *and latency* likely motivated.

1

Matt Klein
@mattklein123

Replying to @jhscott @_msw_ and 2 others

Mostly bills. 😉

9:08 PM · Sep 6, 2019 · Twitter Web App

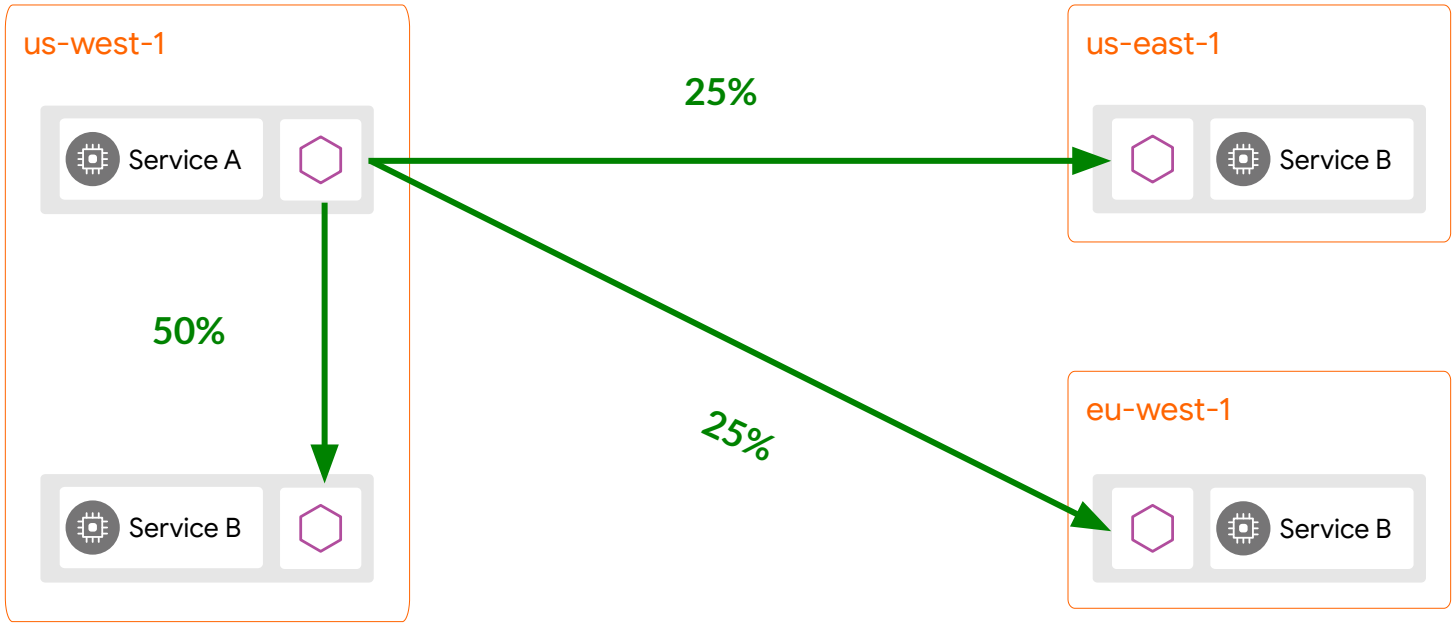
Maximize Availability
Minimize Latency & Costs

Zone Aware Load Balancing?

- Envoy cluster name must be static and known at bootstrap
- Originally implemented for a different set of constraints

Locality Weighted Load Balancing?

envoycon



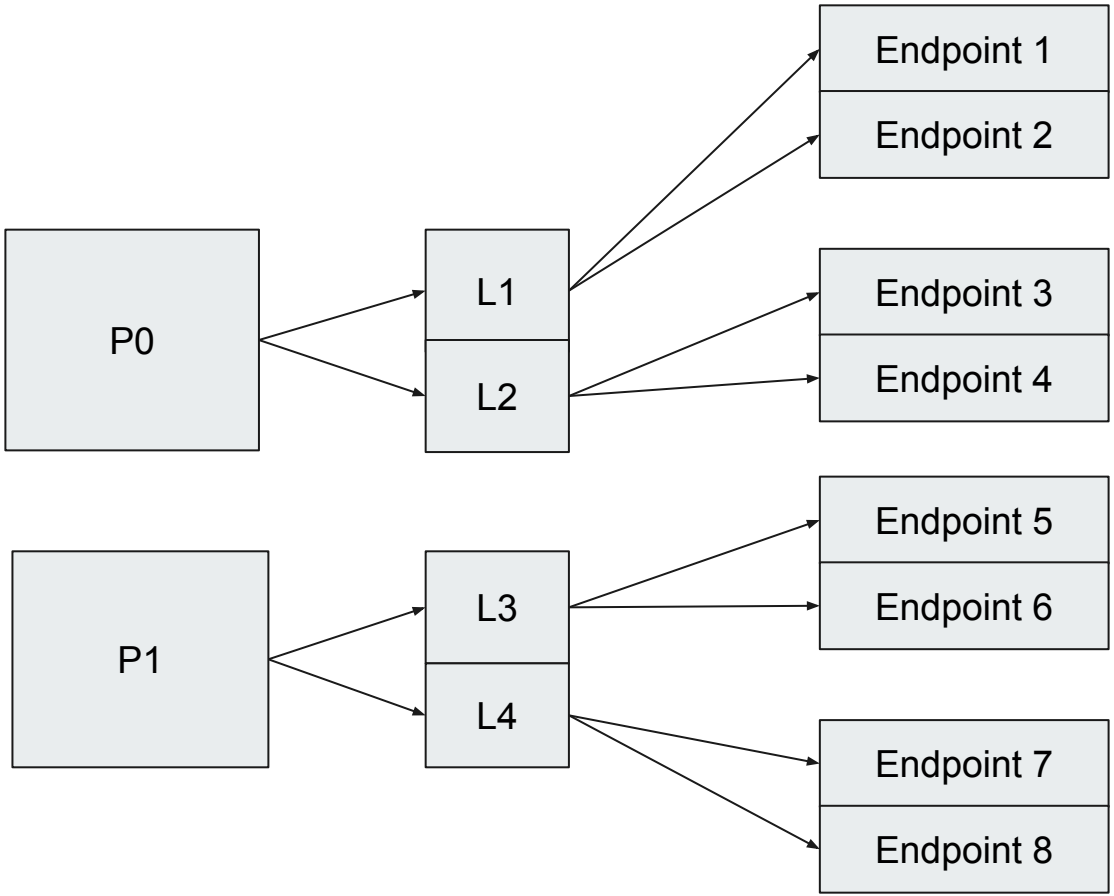
Load Balancing Components

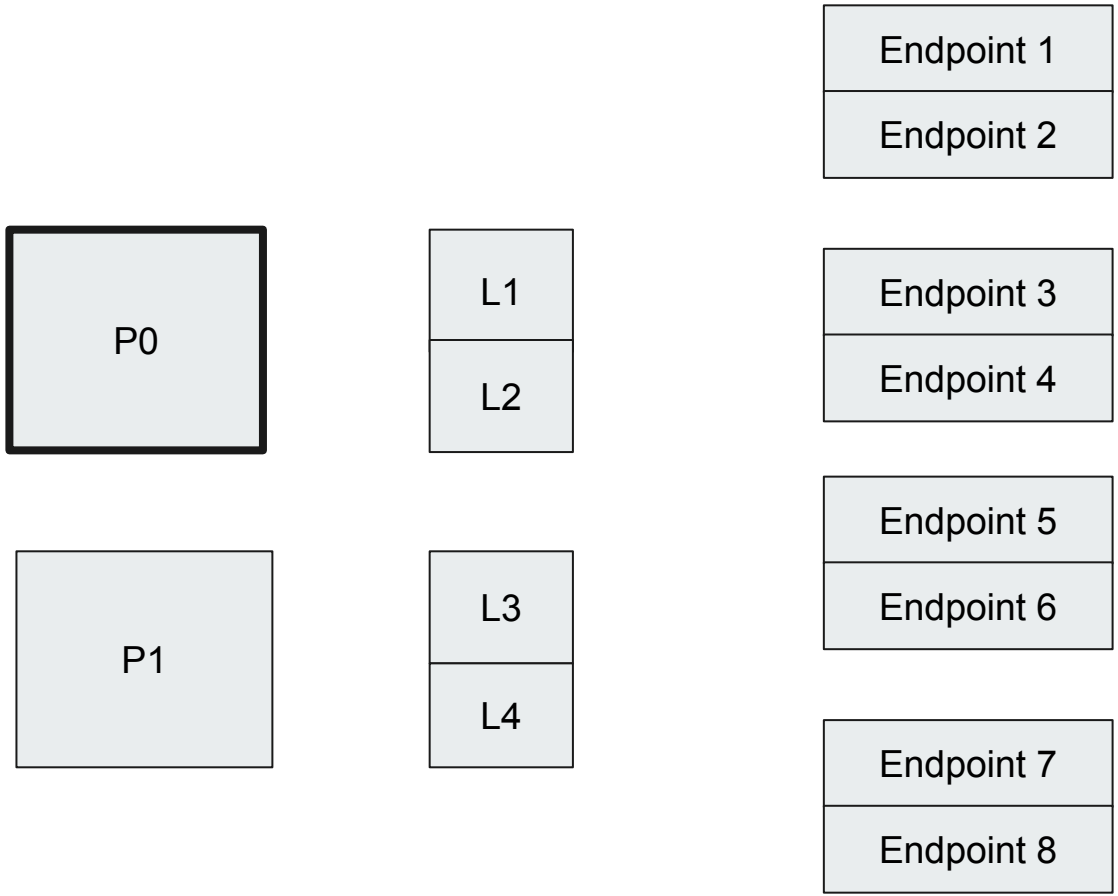
Priority

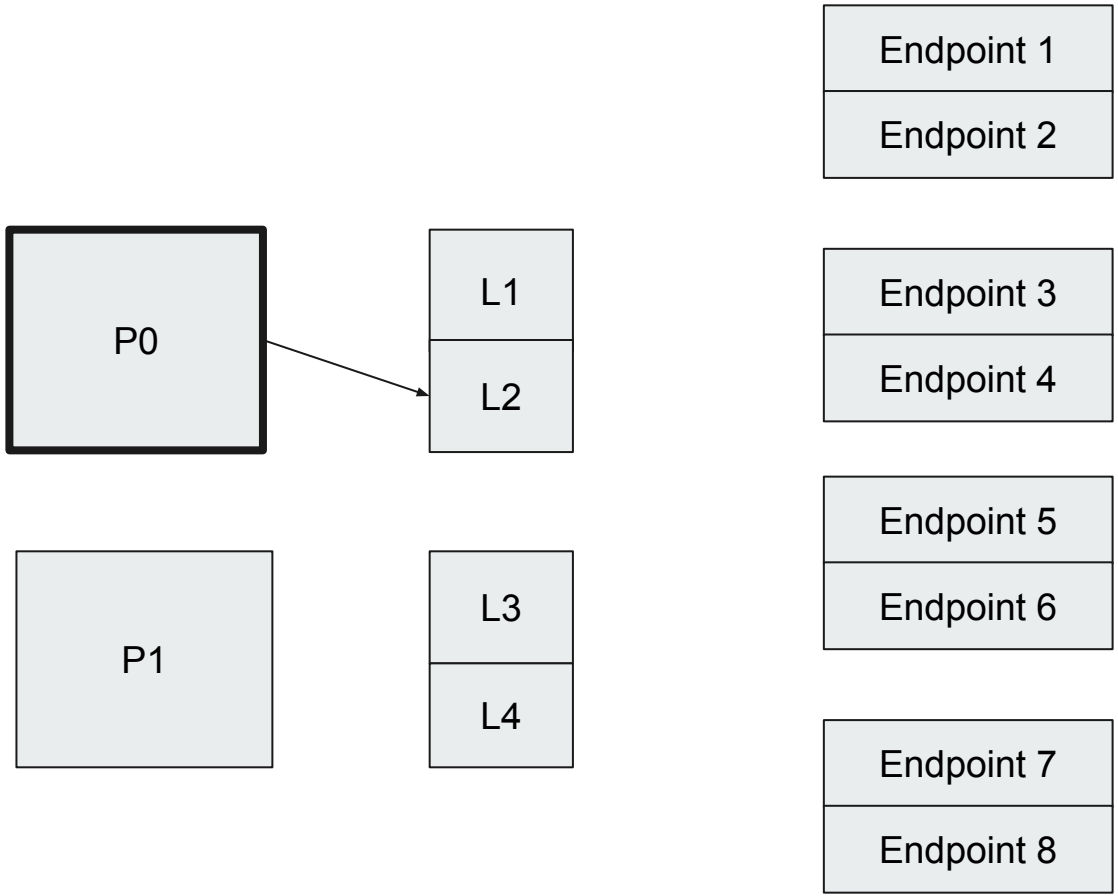
- Endpoints are given a numeric priority, starting at 0
- Endpoints are selected from priorities in order based on host health
- As P(N) becomes unavailable, traffic spills over to P(N+1)

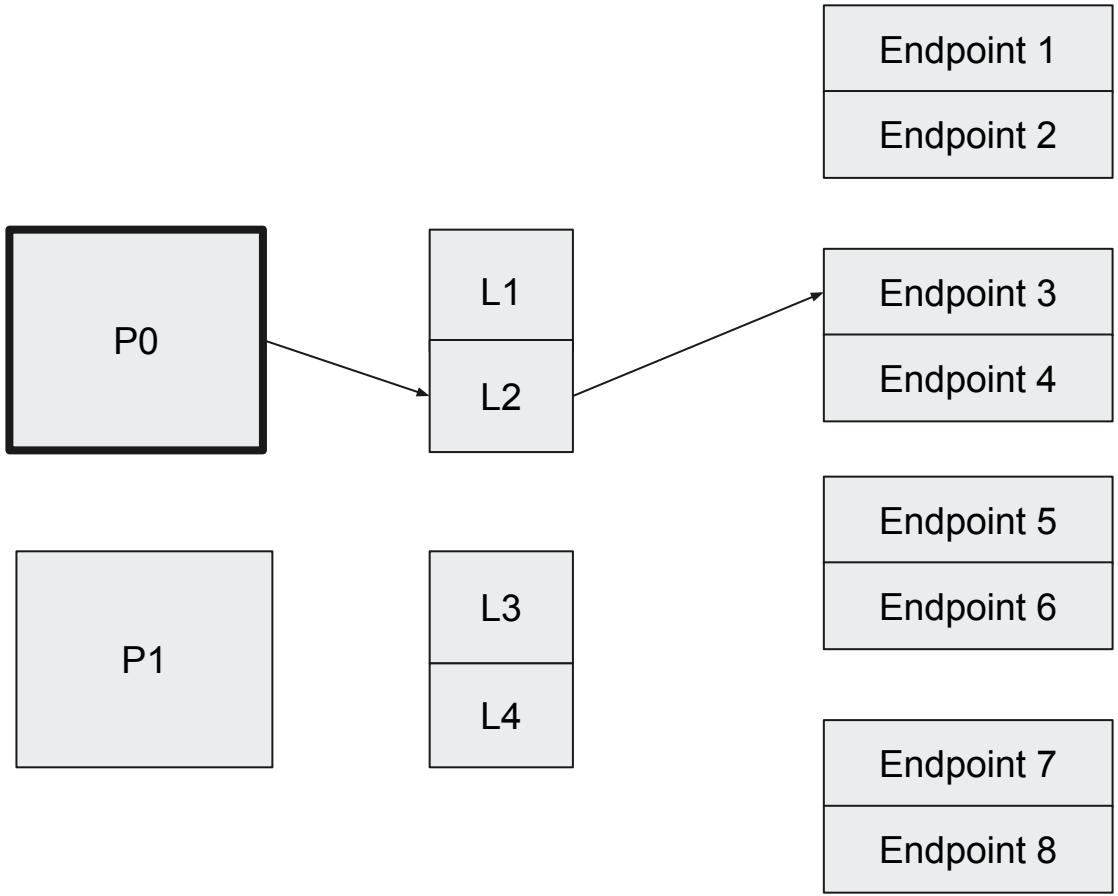
Localities

- Within a priority, endpoints can be grouped into localities
- A locality is selected using weighted RR based on locality weight
- Locality weights are scaled according to host availability
- Host is selected from within locality using specified LB algorithm


P0
P1
L1
L2
L3
L4
Endpoint 1
Endpoint 2
Endpoint 3
Endpoint 4
Endpoint 5
Endpoint 6
Endpoint 7
Endpoint 8

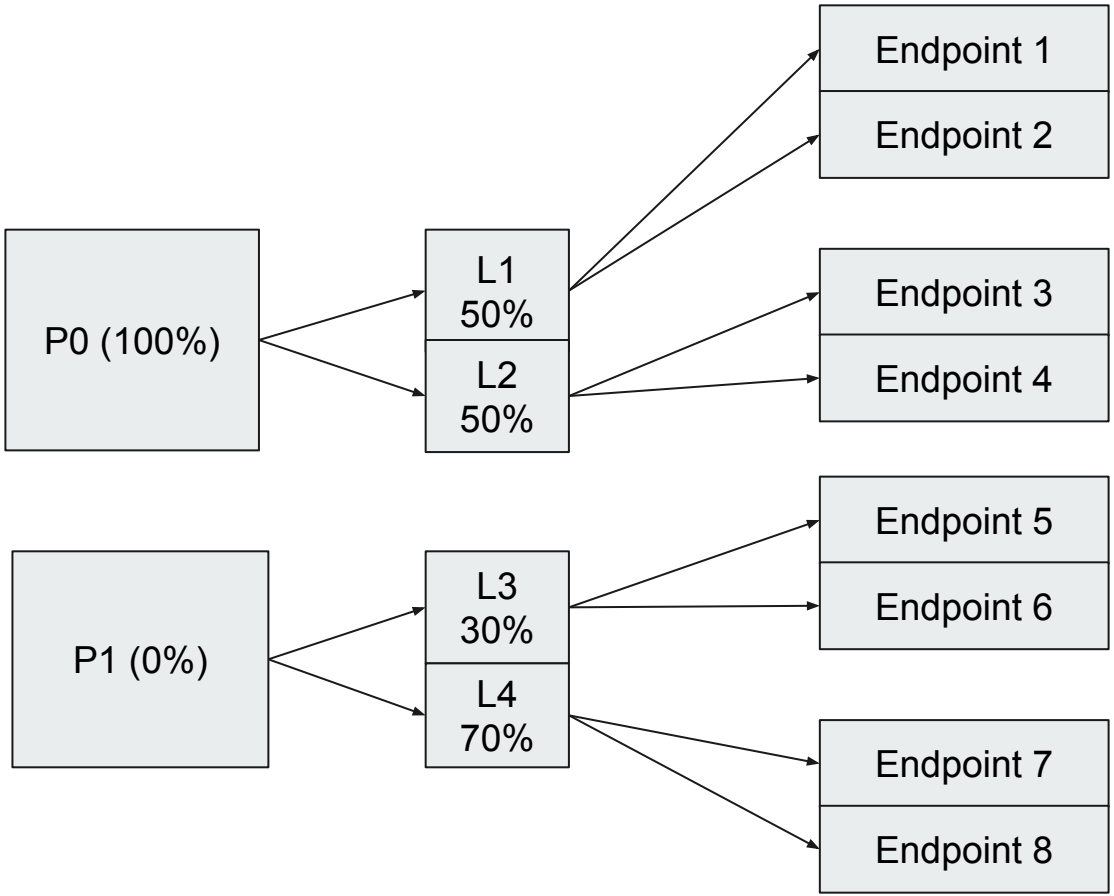

P0
P1
L1
L2
L3
L4
Endpoint 1
Endpoint 2
Endpoint 3
Endpoint 4
Endpoint 5
Endpoint 6
Endpoint 7
Endpoint 8

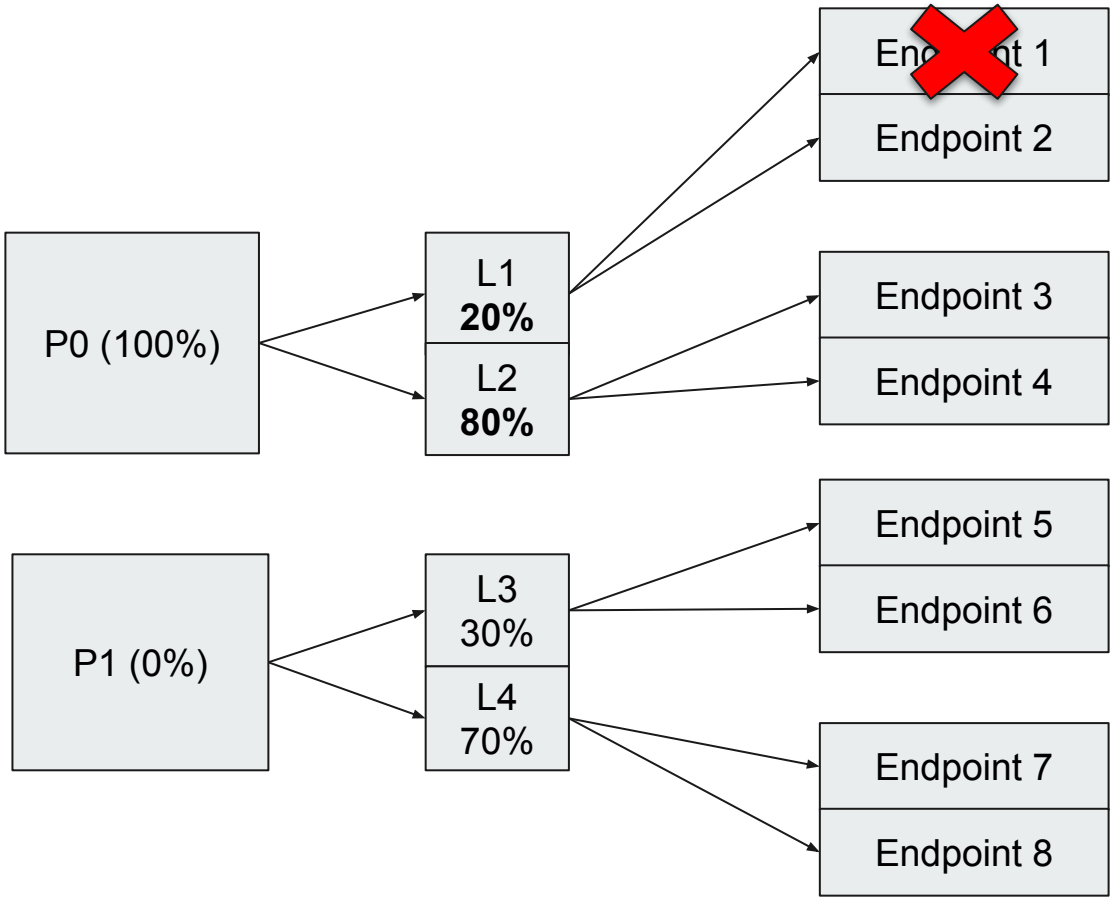

P0
P1
L1
L2
L3
L4
Endpoint 1
Endpoint 2
Endpoint 3
Endpoint 4
Endpoint 5
Endpoint 6
Endpoint 7
Endpoint 8

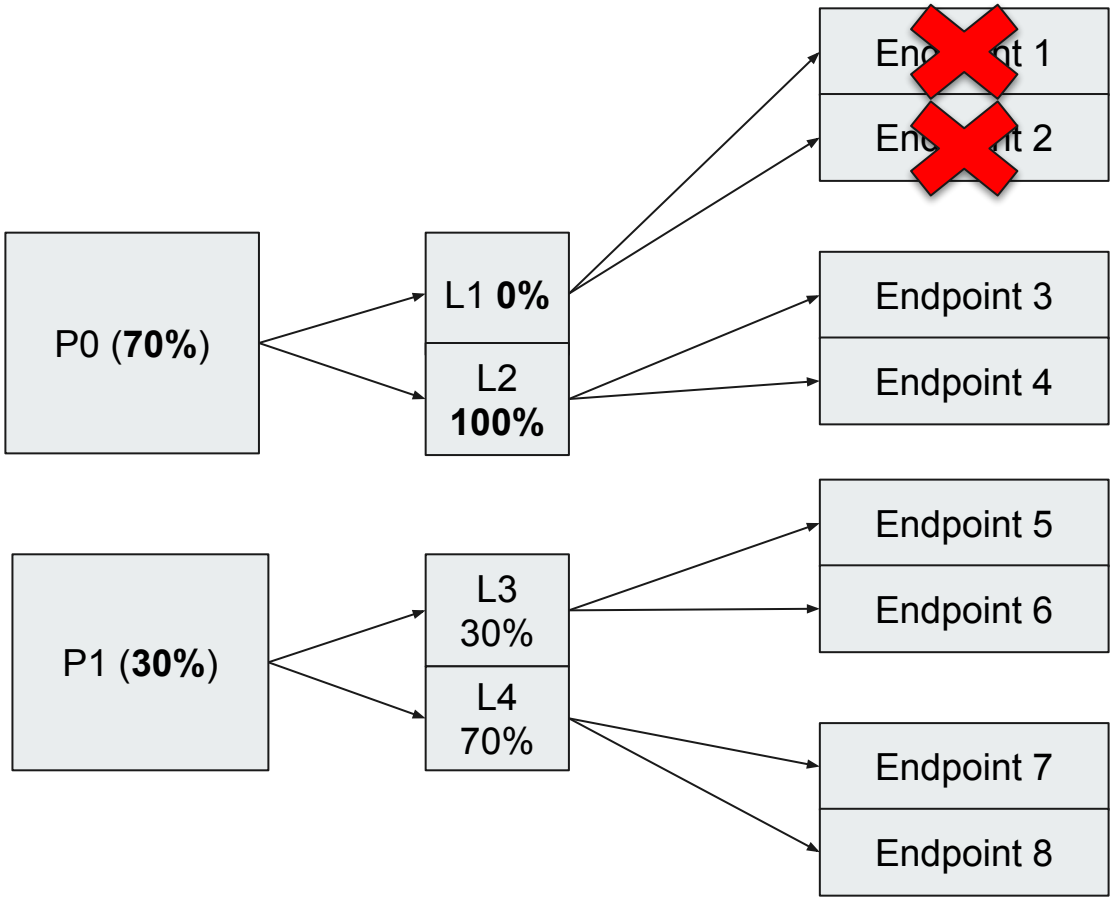

Endpoint 1
Endpoint 2
P0
L1
L2
Endpoint 3
Endpoint 4
Endpoint 5
Endpoint 6
P1
L3
L4
Endpoint 7
Endpoint 8

Spillover

- Envoy considers each priority and locality to be 40% overprovisioned by default
- This is known as the overprovisioning factor
- Provides a buffer before traffic spills over to another priority
- Current Availability = (Available/Total) * Overprovisioning Factor


P0 (100%)
P1 (0%)
L1
50%
L2
50%
L3
30%
L4
70%
Endpoint 1
Endpoint 2
Endpoint 3
Endpoint 4
Endpoint 5
Endpoint 6
Endpoint 7
Endpoint 8


P0 (100%)
L1
20%
L2
80%
Endpoint 1
Endpoint 2
Endpoint 3
Endpoint 4
P1 (0%)
L3
30%
L4
70%
Endpoint 5
Endpoint 6
Endpoint 7
Endpoint 8

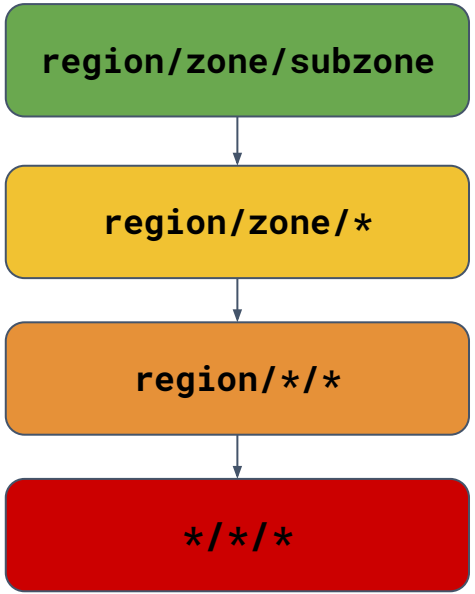

P0 (70%)
P1 (30%)
L1 0%
L2 100%
L3 30%
L4 70%
Endpoint 1
Endpoint 2
Endpoint 3
Endpoint 4
Endpoint 5
Endpoint 6
Endpoint 7
Endpoint 8

Fitting it all Together

Istio Implementation

envoycon

- Determine priority by closest region/zone/subzone matching
- Locality information typically set automatically from Kubernetes node information, or manual configuration



Istio Implementation

envoycon

- Simple config by default, fine tuning available
- Automatically enabled if (passive) health checks are defined
- Still growing - Istio 1.5 adds per-service config and cross region retries

```
localityLbSetting:
  failover:
  - from: us-west
    to: us-east
  - from: us-east
    to: us-west
```

Square Implementation

- Uses a similar heuristic to Istio
- Additional feature requirements:
  - Be able to impact failover rules at runtime
  - Have retries try another region
  - Finer routing priorities between regions

Controlling Failover with Subset LB

- By tagging endpoints with metadata, the subset LB can restrict load balancing to only consider endpoints that match
- Routing priority is preserved for endpoints that match the subset criteria
- Configured by setting specific headers on HTTP requests
- Allows limiting failover to only consider some regions when the latency cost of failover is too great

Cross Region Retries

- Allows active-active services to utilize resources in multiple regions without failing requests
- Attempted priorities can be excluded when selecting the host for the retry
- Respects the subset match criteria, allowing this behavior to be configured at runtime

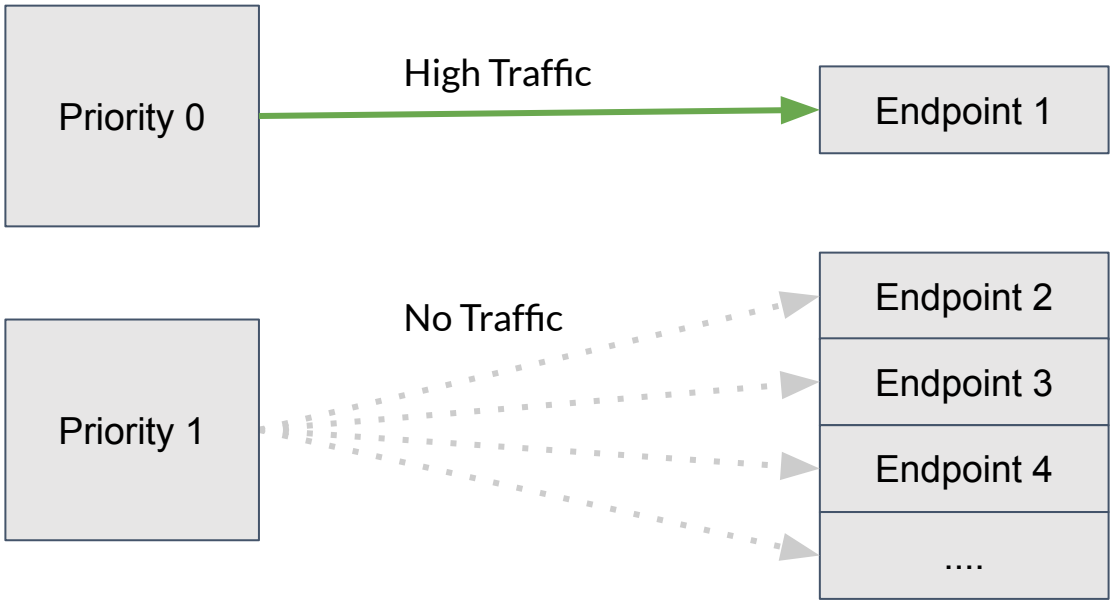
Better Region Failover

envoycon

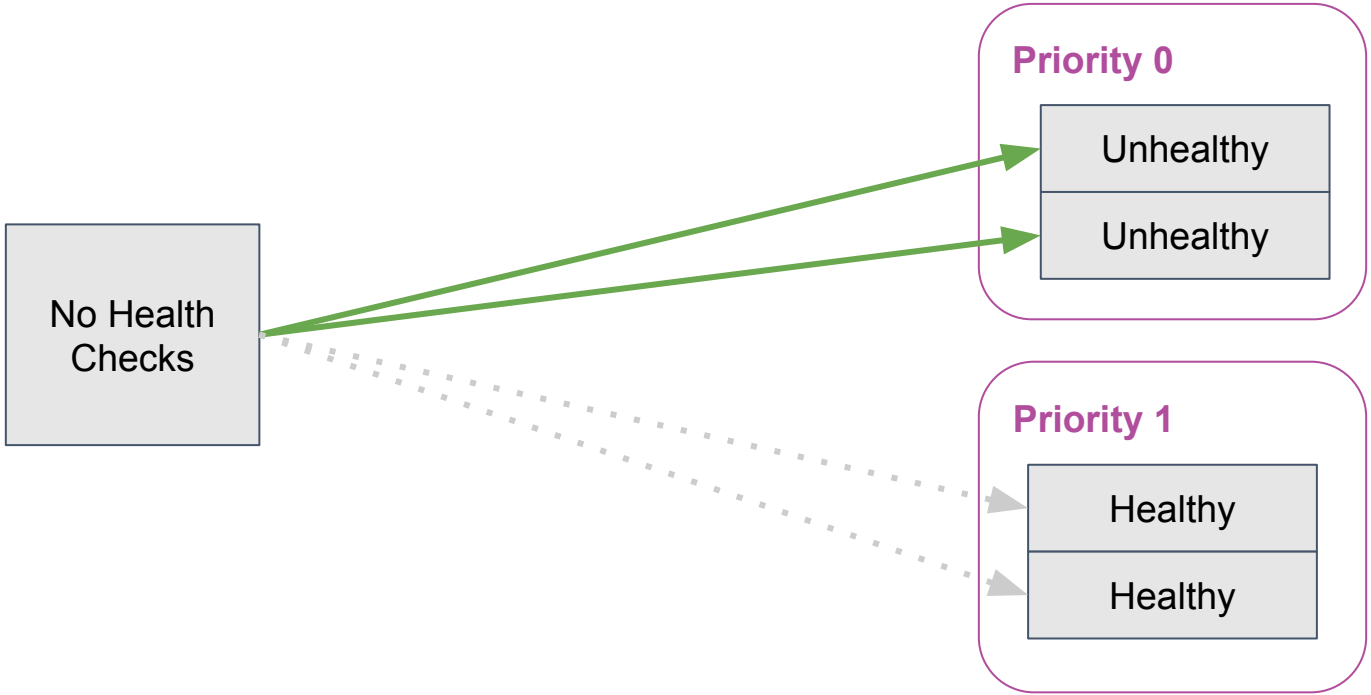
- The list of regions Square operates in is well known
- Allows us to order the region failovers based on origin region in the control plane
- ap-east -> [us-west, us-east]
- us-west -> [us-east, ap-east]

Gotchas?

Uneven Load Distribution

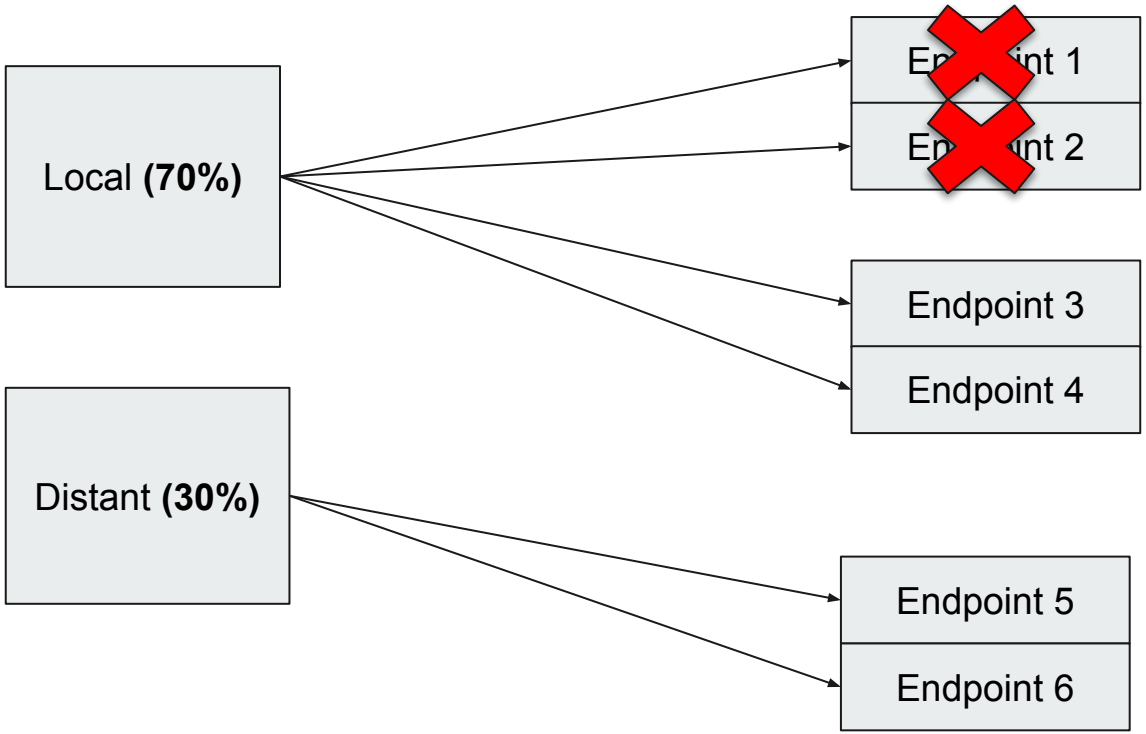


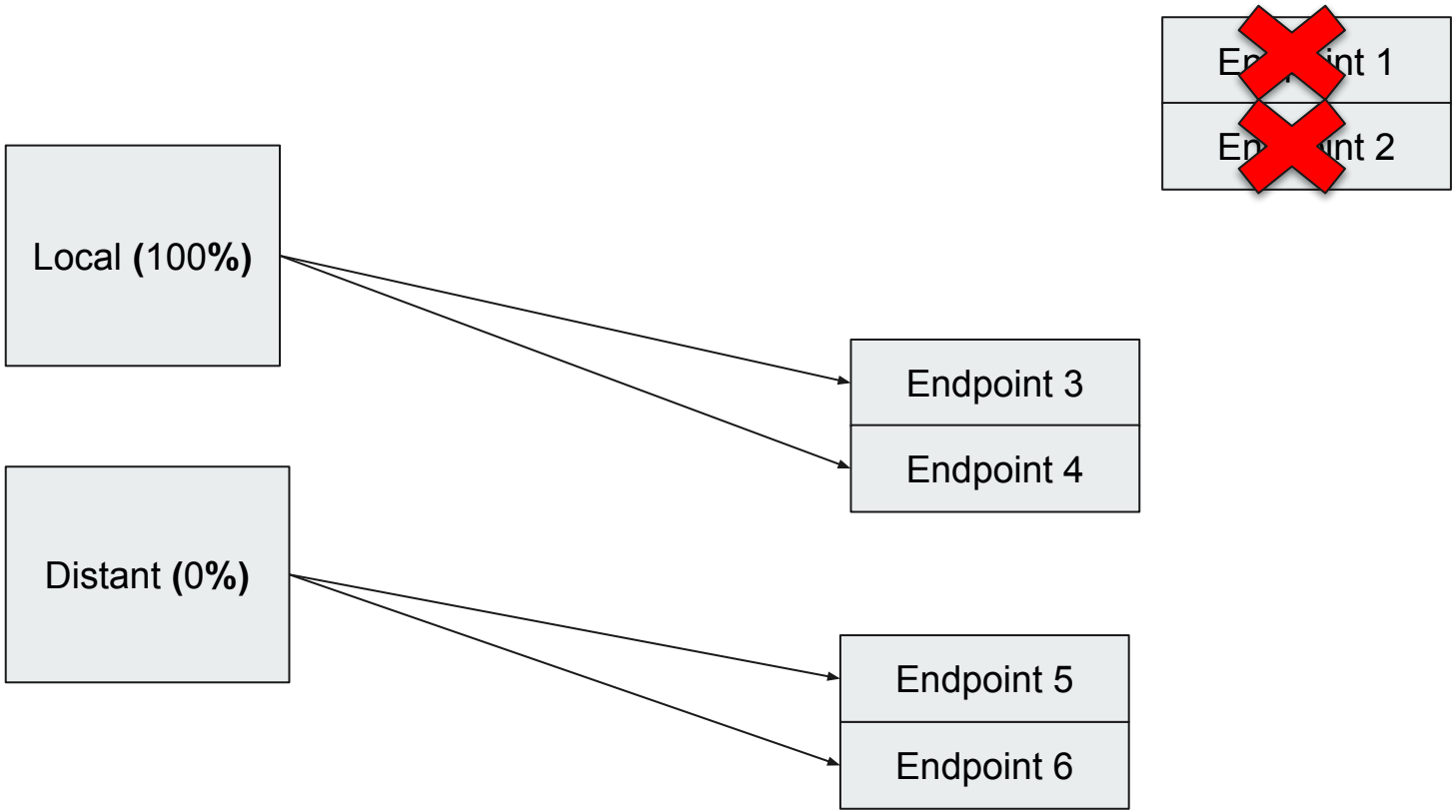
Health Checks



Conflicting Health Checks

- An unhealthy endpoint is not the same as a missing endpoint
- The control plane removing endpoints can impact spillover
- Hard to reason about mixing health checks (such as Kubernetes) with Envoy health checks


Local (70%)
Distant (30%)
Endpoint 3
Endpoint 4
Endpoint 5
Endpoint 6


Endpoint 1
Endpoint 2
Local (100%)
Endpoint 3
Endpoint 4
Distant (0%)
Endpoint 5
Endpoint 6

Connection Setup

- Envoy won’t create connections until it needs them
- Service owners sometimes react to very slow requests showing up as outliers
- Usually turns out to be that failover triggered, causing TLS setup to happen to another region

envoycon