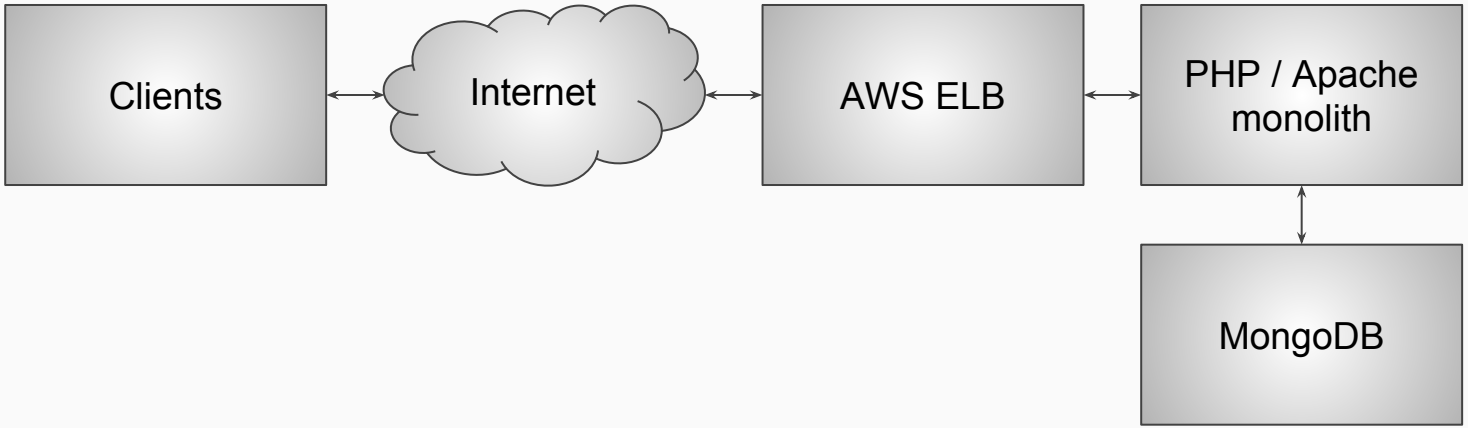
Lyft's Envoy:
Embracing a Service Mesh

Matt Klein / @mattklein123, Software Engineer @Lyft

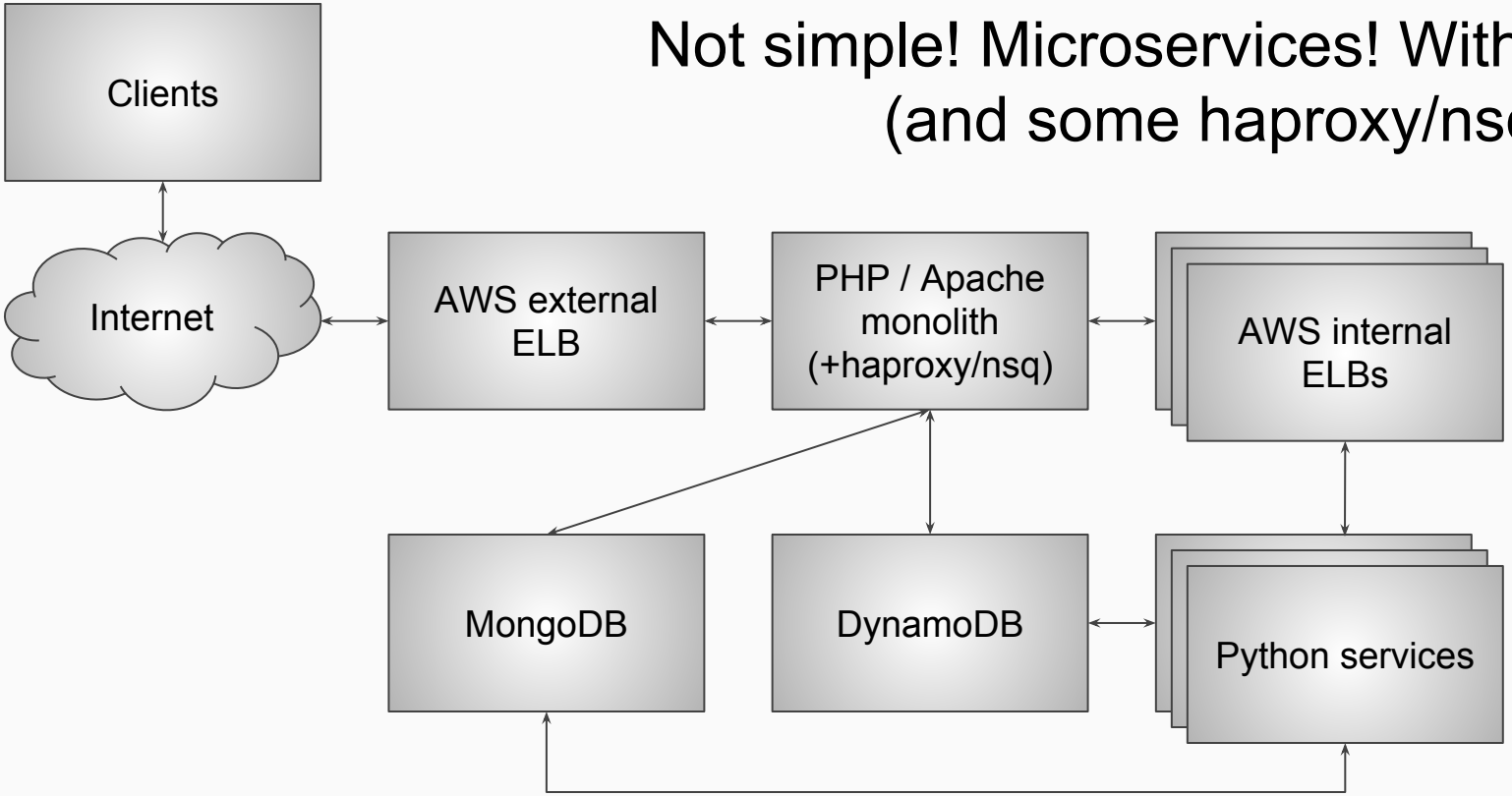
Lyft ~5 years ago



Simple! No microservices! (*but still not that simple*)

Lyft ~3 years ago

Not simple! Microservices! With monolith!
(and some haproxy/nsq)



Lyft’s microservice architecture problems 3 years ago

- Multiple **Languages** and frameworks.
- Many **Protocols** (HTTP/1, HTTP/2, gRPC, databases, caching, etc.).
- Black box **load balancers** (AWS ELB).
- Lack of consistent **Observability** (stats, tracing, and logging).
- Partial or no implementations of **retry**, **circuit breaking**, **rate limiting**, **timeouts**, and other distributed systems best practices.
- Minimal **Authentication** and **Authorization**.
- Per language **libraries** for service calls.
- Extremely difficult to **debug** latency and failures.
- Developers did not **trust** the microservice architecture.

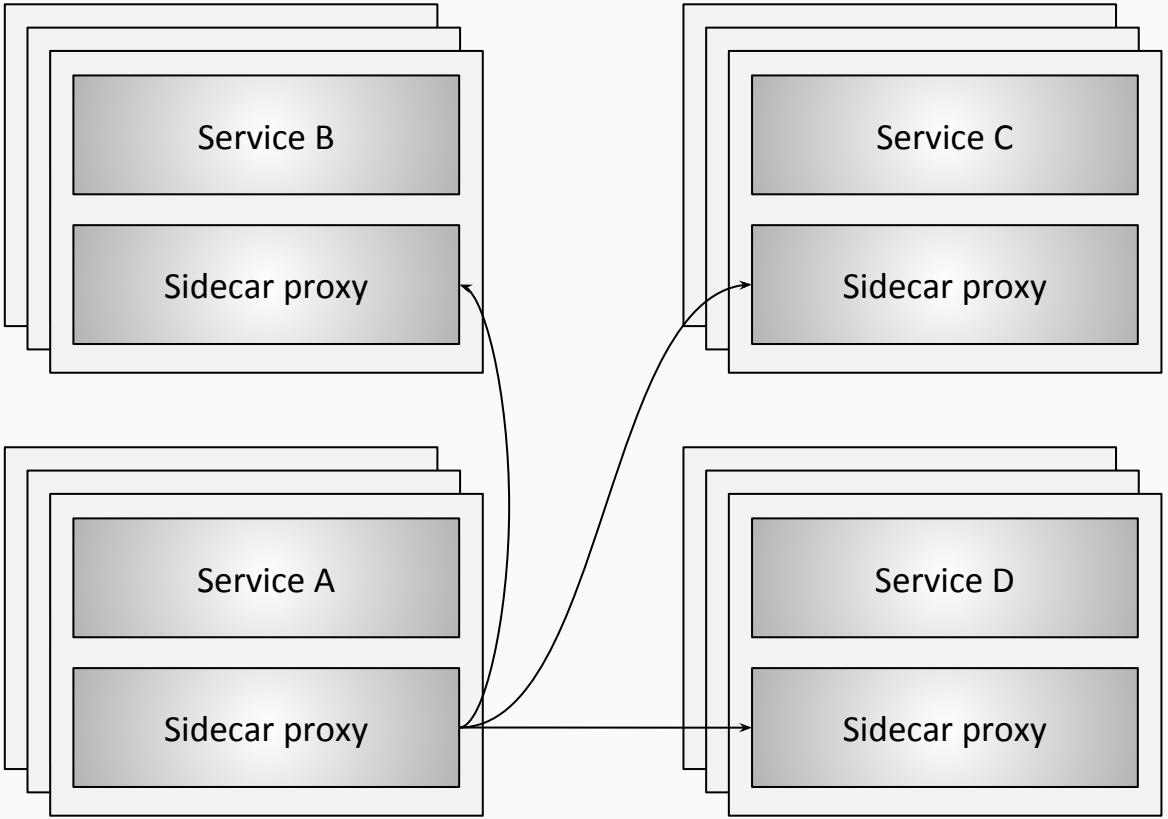
Lyft’s architecture problems 3 years ago

A really big and confusing mess...

What is Envoy and the service mesh?

The network should be transparent to applications. When network and application problems do occur it should be easy to determine the source of the problem.

Service mesh refresher



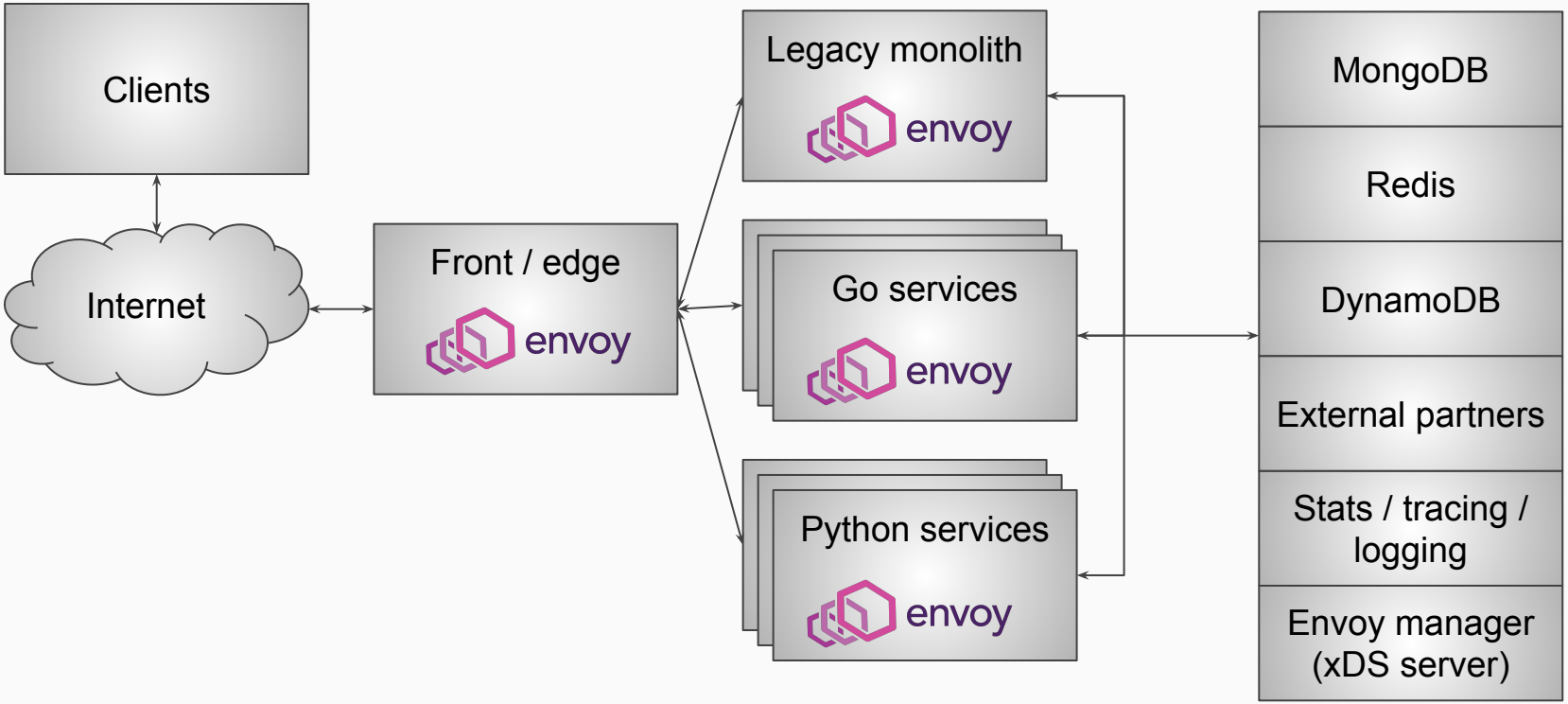
Envoy

- **Out of process architecture**
- **High performance / low latency code base**
- **L3/L4 filter architecture**
- **HTTP L7 filter architecture**
- **HTTP/2** first
- **Service discovery** and **active/passive health checking**
- **Advanced load balancing**
- Best in class **observability** (stats, logging, and tracing)
- **Authentication** and **authorization**
- **Edge proxy**

Observability

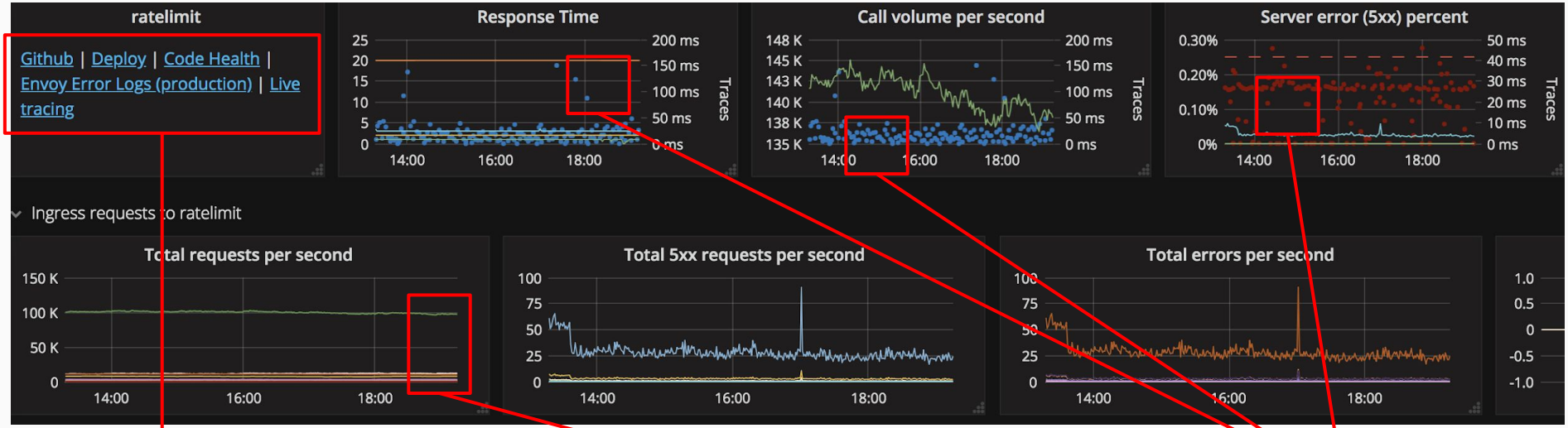
- **Observability** is by far the most important thing that Envoy and the service mesh provides.
- Having all traffic transit through Envoy provides a single place to:
  - Produce consistent **statistics** for every hop.
  - Create and propagate a stable **request ID / tracing context**.
  - Consistent **logging**.
  - Distributed **tracing**.

Lyft today



Obs, obs, obs, obs, obs, obs...

Per service auto-generated panel

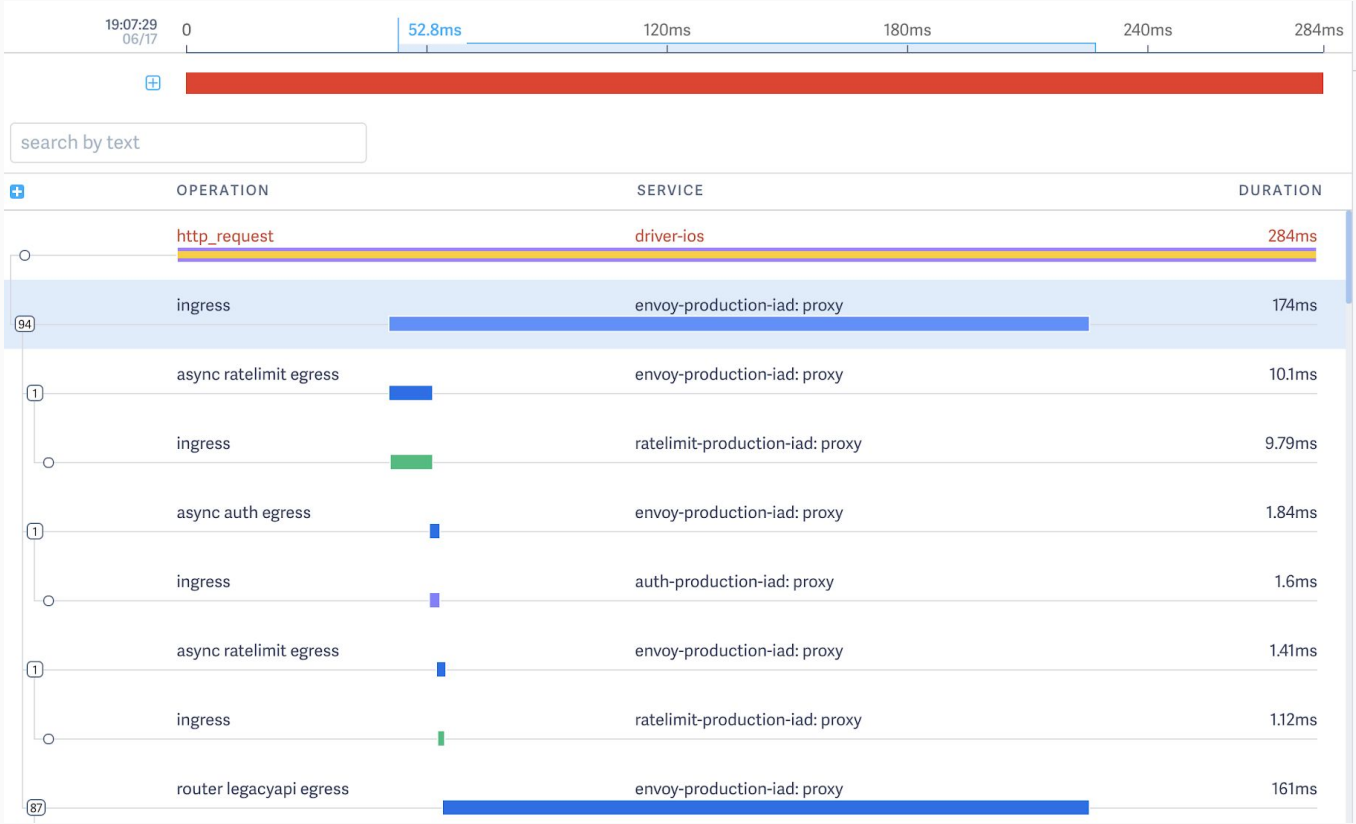


Links to interesting data

Per-caller information

Clickable traces from top-level panel

Distributed tracing

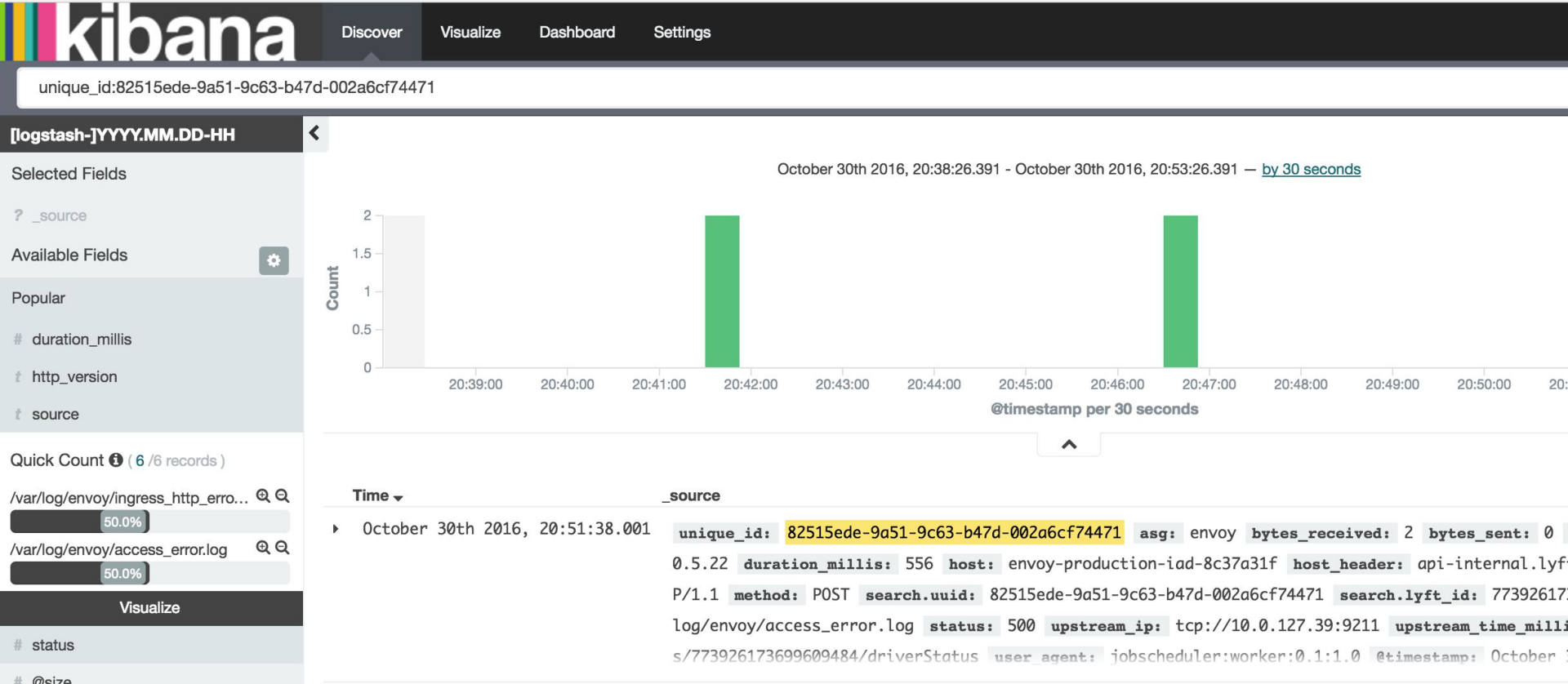


Service
envoy-production-iad: proxy

Tags

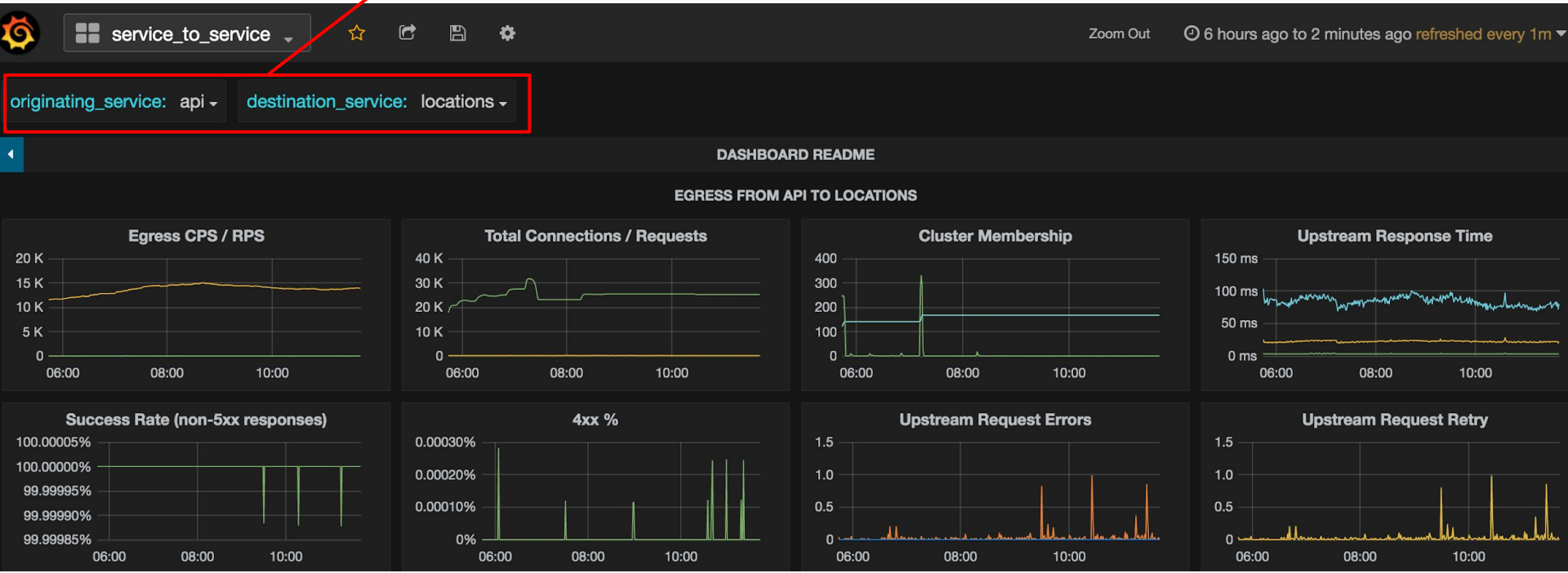
component	proxy
downstream_clu...	envoy-production-iad
guid:x-client-trac...	30663F8A-A6CD-4AAD-9268-B420FC7003C8
guid:x-request-id	d50a4678-40f4-b9f5-ac93-57a64878eb25
http.method	PUT
http.protocol	HTTP/2
http.status_code	200
http.url	https://api.lyft.com/users/1021136220529481O/location
node_id	019ff77b184a47006
request_size	270
response_flags	-
response_size	3966
upstream_cluster	legacyapi
user_agent	com.lyft.ios.driver:iOS:11.2.5:1001.57.3390899
x-lyft-user-id	1021136229305294810
zone	us-east-1d

Logging

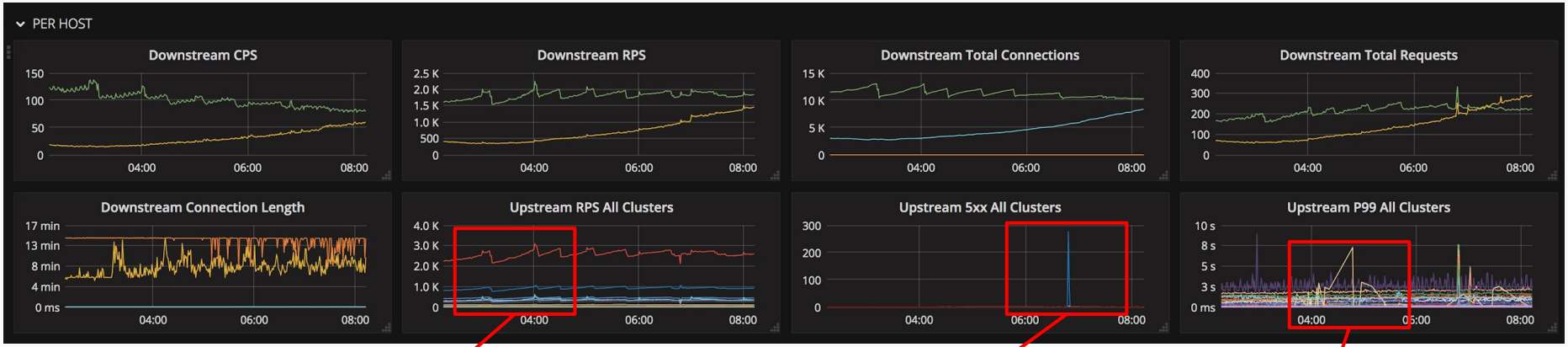

kibana
Discover
Visualize
Dashboard
Settings
unique_id:82515ede-9a51-9c63-b47d-002a6cf74471
[logstash-]YYYY.MM.DD-HH
Selected Fields
? _source
Available Fields
Popular
duration_millis
t http_version
t source
Quick Count (6 /6 records)
/var/log/envoy/ingress_http_erro...
50.0%
/var/log/envoy/access_error.log
50.0%
Visualize
status
October 30th 2016, 20:38:26.391 - October 30th 2016, 20:53:26.391 — by 30 seconds
Count
2
1.5
1
0.5
0
20:39:00
20:40:00
20:41:00
20:42:00
20:43:00
20:44:00
20:45:00
20:46:00
20:47:00
20:48:00
20:49:00
20:50:00
@timestamp per 30 seconds
Time
_source
October 30th 2016, 20:51:38.001
unique_id: 82515ede-9a51-9c63-b47d-002a6cf74471 asg: envoy bytes_received: 2 bytes_sent: 0
0.5.22 duration_millis: 556 host: envoy-production-iad-8c37a31f host_header: api-internal.lyf
P/1.1 method: POST search.uuid: 82515ede-9a51-9c63-b47d-002a6cf74471 search.lyft_id: 7739261
log/envoy/access_error.log status: 500 upstream_ip: tcp://10.0.127.39:9211 upstream_time_milli
s/7739261736996094484/driverStatus user_agent: jobscheduler:worker:0.1:1.0 @timestamp: October

Service to service template dashboard

Template with drop down for every service



Edge proxy

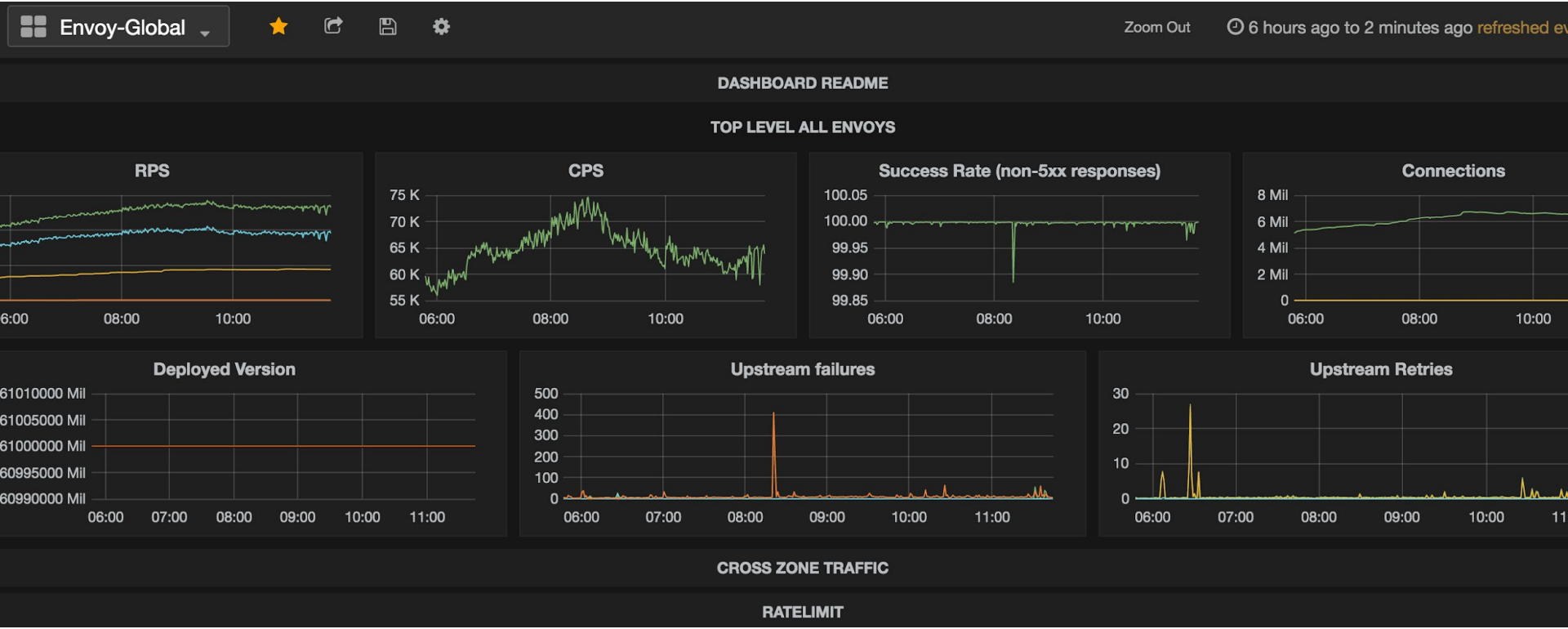


Per-upstream cluster RPS

Per-upstream cluster 5xx

Per-upstream cluster timings

Global health dashboard



Envoy thin clients @Lyft

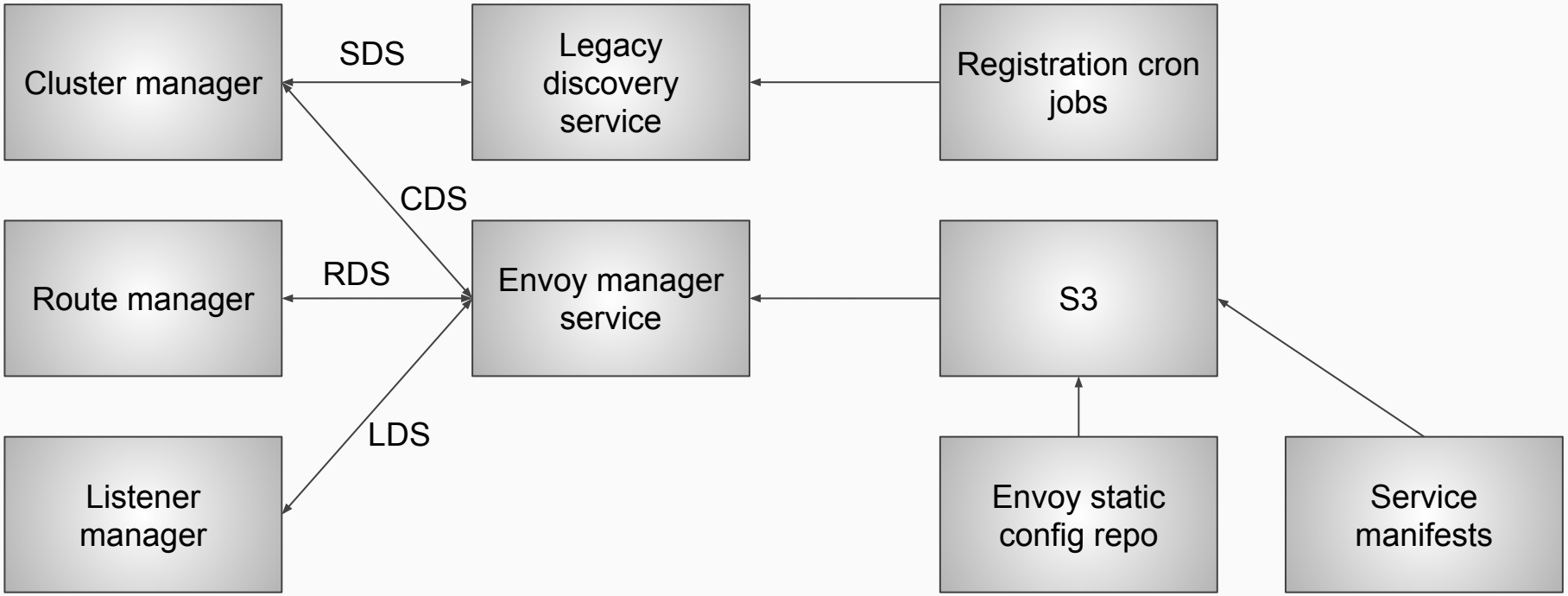
```
from lyft.api_client import EnvoyClient
switchboard_client = EnvoyClient(
    service='switchboard'
)
msg = {'template': 'breaksignout'}
headers = {'x-lyft-user-id': 12345647363394}
switchboard_client.post("/v2/messages", data=msg, headers=headers)
```

- Abstract away egress port
- Request ID/tracing propagation
- Guide devs into good timeout, retry, etc. policies
- Similar thin clients for Go and PHP

Envoy config management via xDS APIs

- Envoy is a **universal data plane**
- xDS == * Discovery Service (various configuration APIs). E.g.,:
  - LDS == Listener Discovery Service
  - CDS == Cluster Discovery Service
- Both gRPC streaming and JSON/YAML REST via proto3!
- Central management system can control a fleet of Envoys **avoiding per-proxy config file hell**
- **Global bootstrap config** for every Envoy, rest taken careof by the management server
- Envoys + xDS + management system == **fleet wide traffic management distributed system**

Envoy config management via xDS APIs @lyft



Only need a very tiny bootstrap config for each envoy...

Lyft’s Envoy deployment

- 100s of services
- 10Ks of hosts
- 5-10M mesh RPS
- Majority h2
- All edge, StS, and vast majority of external partners
- MongoDB, DynamoDB, Spanner, Redis
- Evolving configuration management system as we move to K8s

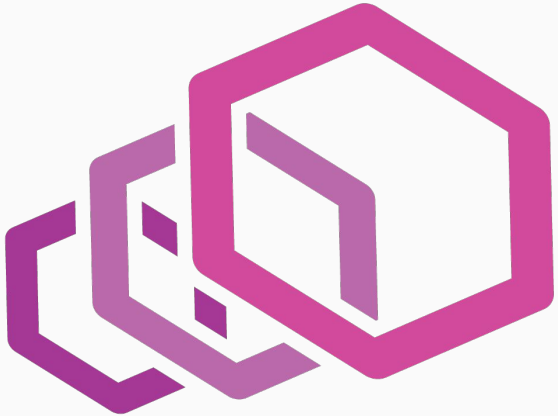
Envoy adoption

And lots more not listed...

Why Envoy + Q&A

- Quality + velocity
- Extensibility
- Eventually consistent configuration API
- No “open core” / paid premium version. It’s all there
- **Community, community, community**

Critical mass has nearly been achieved. Becoming too costly to **not** use?



envoy