Dec 12, 2018

Reaching 5 Million Messaging Connections:

Our Journey with Kubernetes

Dylan O'Mahony - Cloud Architecture Manager, Bose
Dave Doyle - Software Engineering Manager, Connected

So near, yet so far

Cloud

Home

BOSE Connected

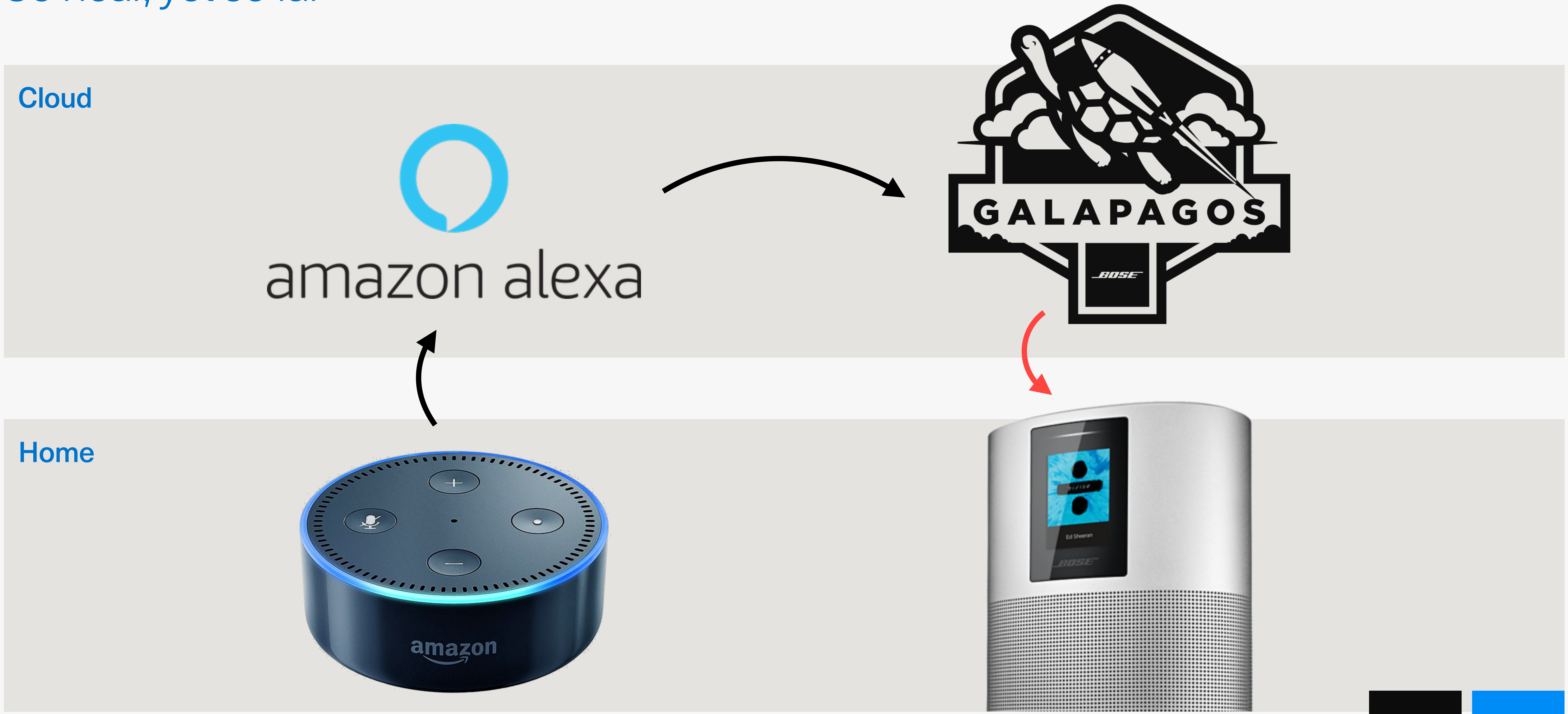
So near, yet so far

Cloud

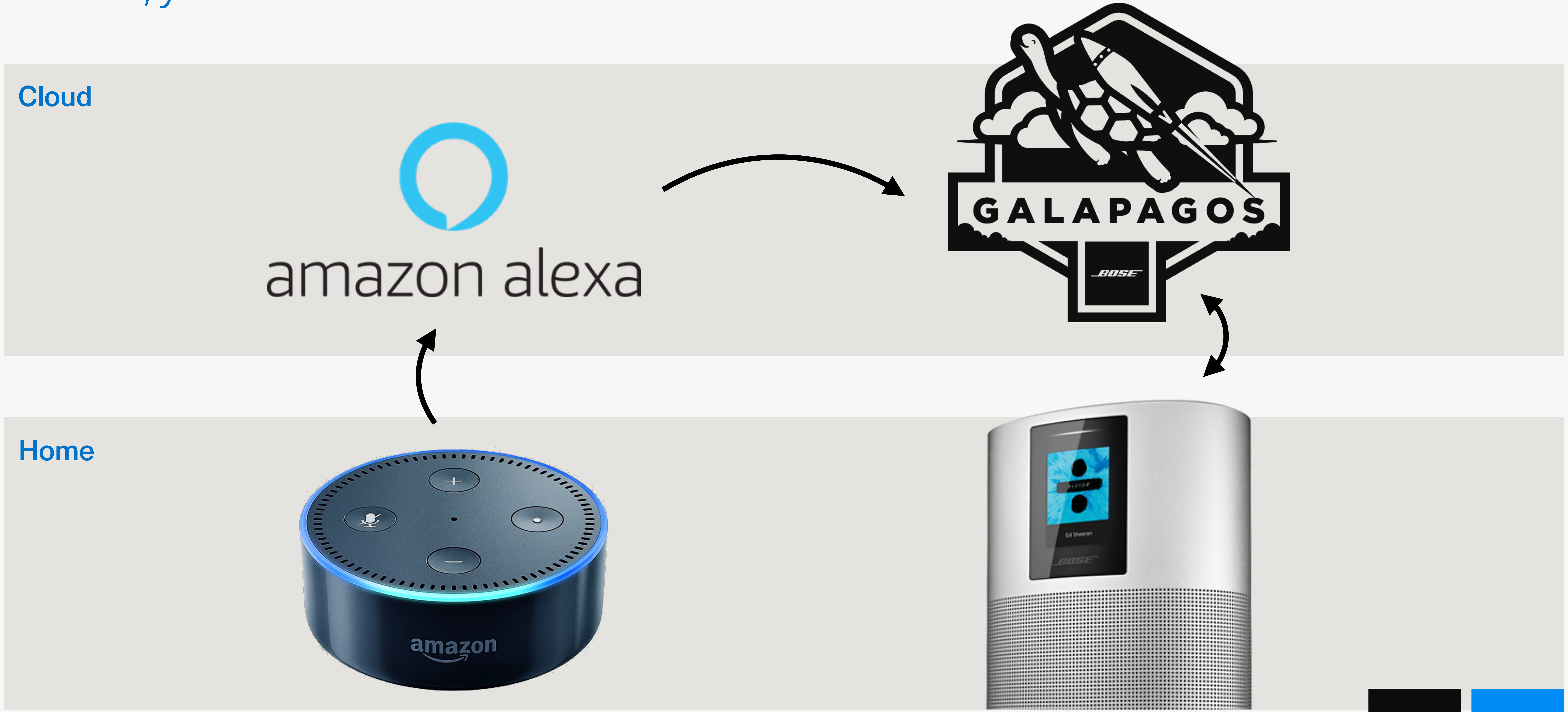
amazon alexa

Home

So near, yet so far



So near, yet so far



Where I’m coming from.

The Team

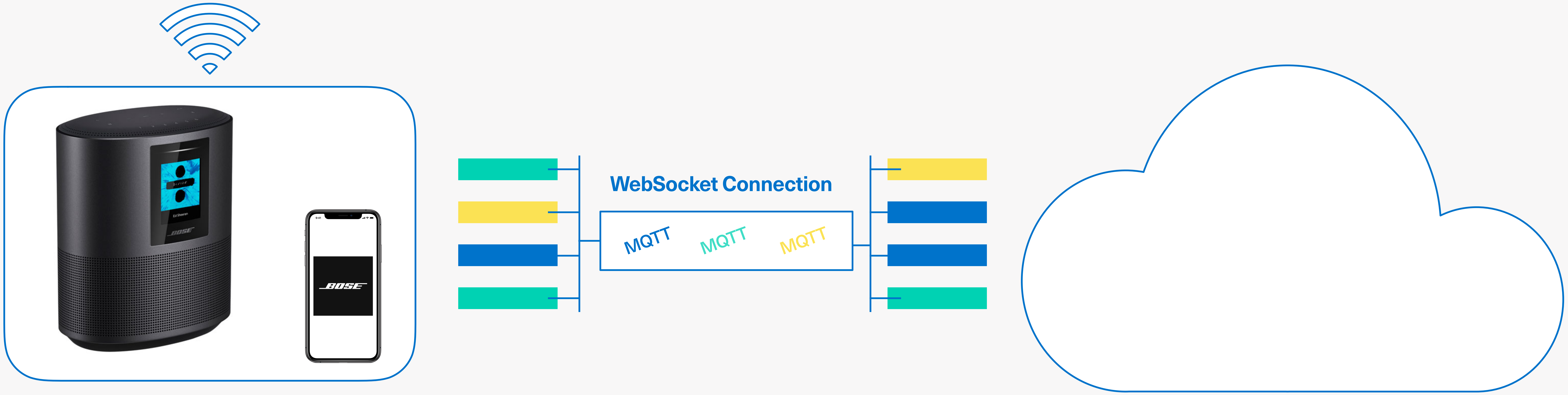
Four people.
Two teams.
Makers and breakers.

The Stack

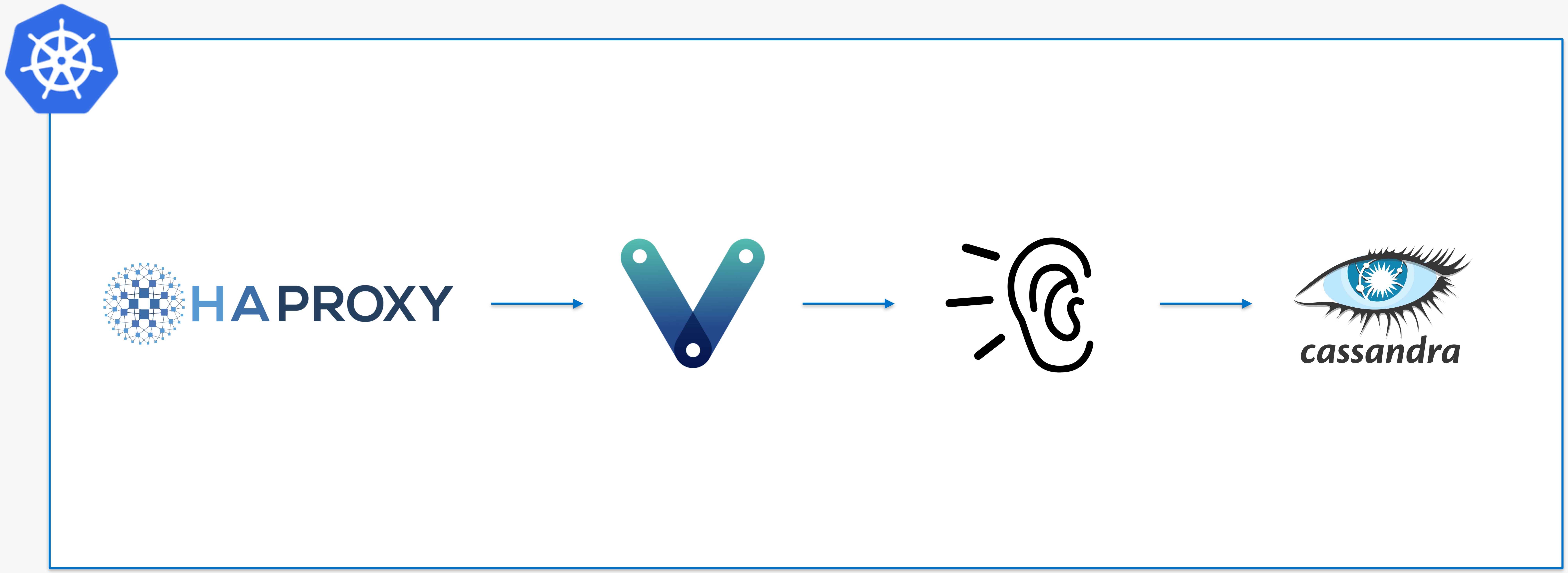
Infrastructure: “Galapagos”

- Kubernetes on AWS (not EKS)
- Each team member had a full rollout of the stack

Solution Model



Solution Components



Ingress & Load Balancing: HAProxy

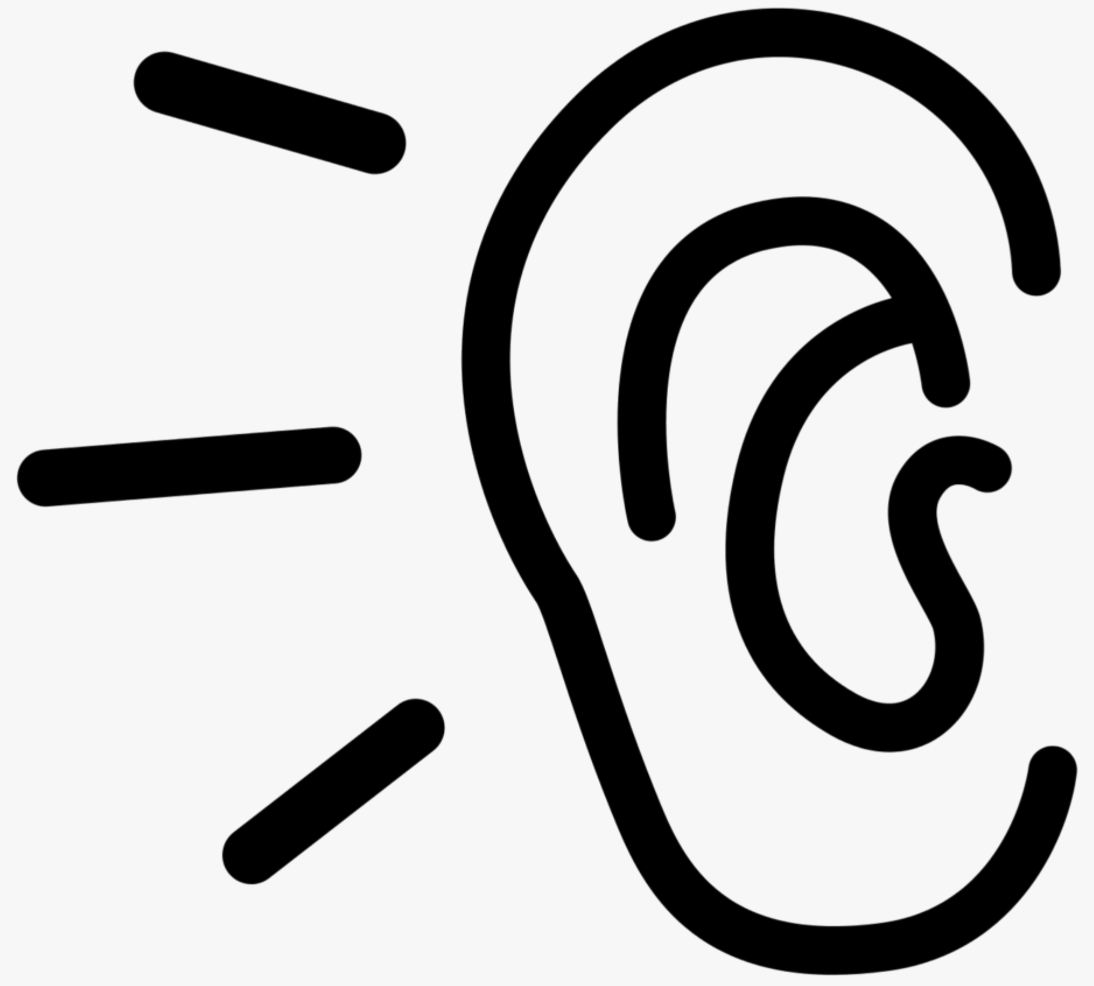
- De facto standard for proxy and load balancing
- TCP for WebSockets
- Less confusing than most ingress options

Message Broker: VerneMQ

- Clustering
- Bridging (future considerations)
- MQTTv5 shared subscriptions
- Fault tolerance
- Well-defined netsplit behaviour
- Time order integrity

The Glue: Listening Service

- Written in Golang
- Subscribes to VerneMQ with a shared subscription
- Writes shadow states to Cassandra
- Lightweight and performant



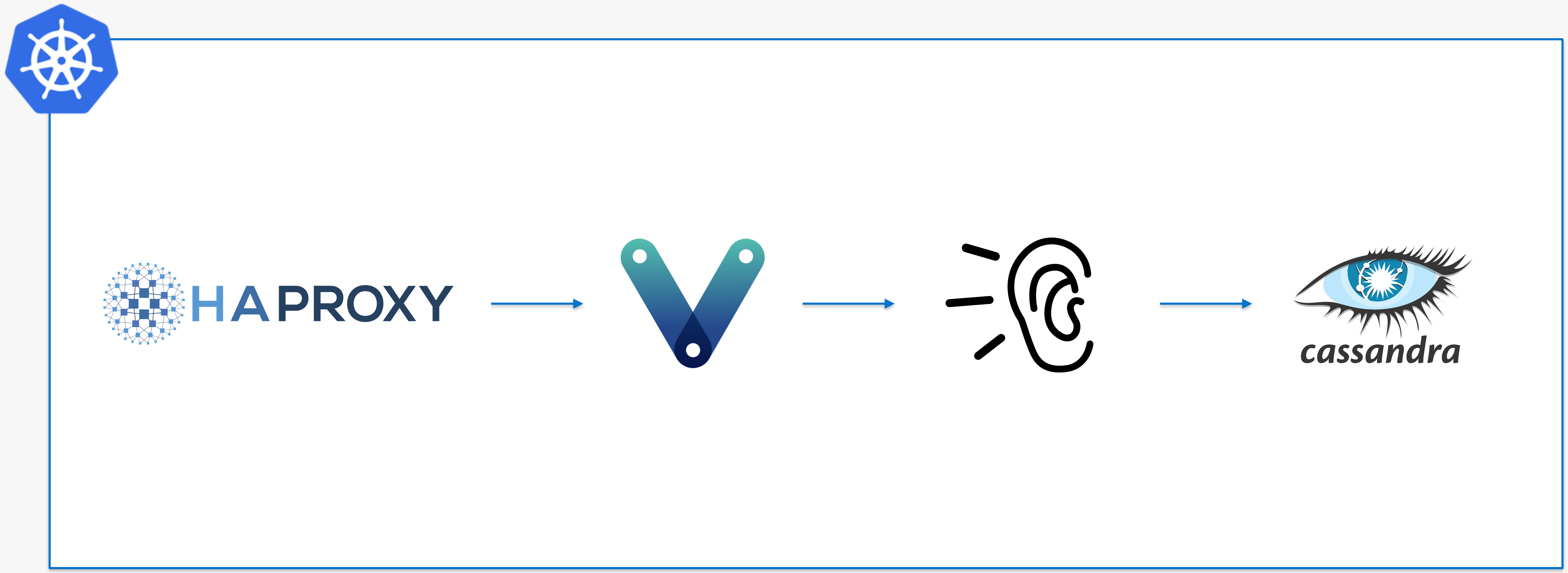
Shadow Store: Cassandra

- Performant
- Fault tolerant
- Massively scalable
- Stable

Setup: Kubernetes

- All images built on Alpine
- **StatefulSet:** VerneMQ, Cassandra
- **DaemonSet:** HAProxy (ingress nodes)
- **Deployment:** Listening Service, Prometheus, Grafana

Solution Components

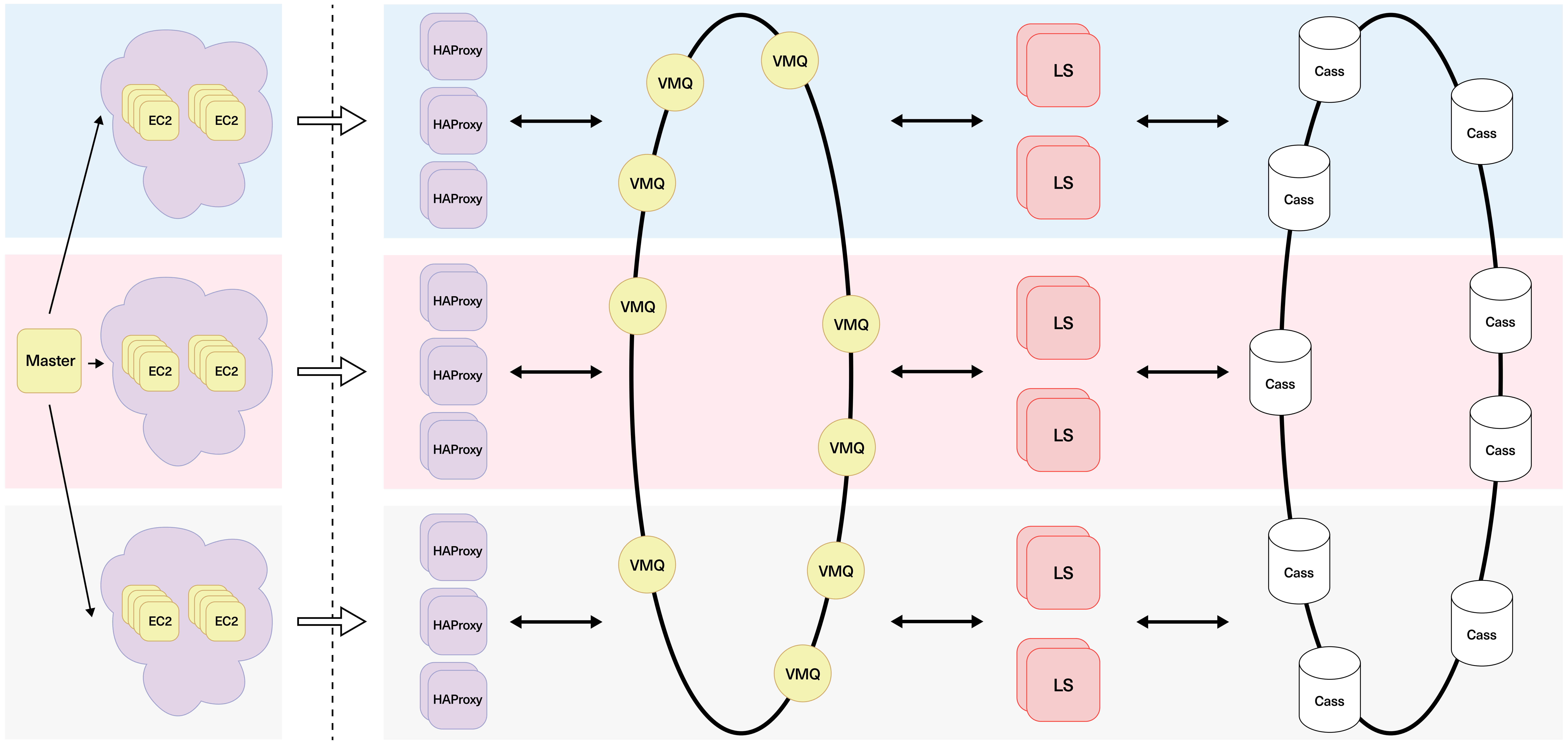


Test Rig: Locust

- Explored MZBench and JMeter
- Wanted more flexibility
- Decided on Locust

LOCUST

High-Level Architecture



Testing

Target

5,000,000 Persistent Concurrent Connections

Result

340 Connections

Blocker: Python File Descriptor limits

- Paho MQTT client
- Python and *select()*
- Python has max 1024 file descriptors open when using *select()*

Workaround

- Replaced *select()* call
- Tried async_io library
- Did not work

```
% make python
```

Result

700k Connections

Blocker: Configuration defaults and NAT

- HAProxy port exhaustion
- VerneMQ default connection config limits
- Service abstraction NAT

Workaround

Reconfigure Everything

- VerneMQ: fix max connection setting, add 3 more listeners
- Bypass Kubernetes *Service*
- HAProxy
  - round-robin VerneMQ nodes
  - increase source ports
  - vertically scale ingress nodes for more iops/bandwidth
- Created app to query Kubernetes API, returned templated config

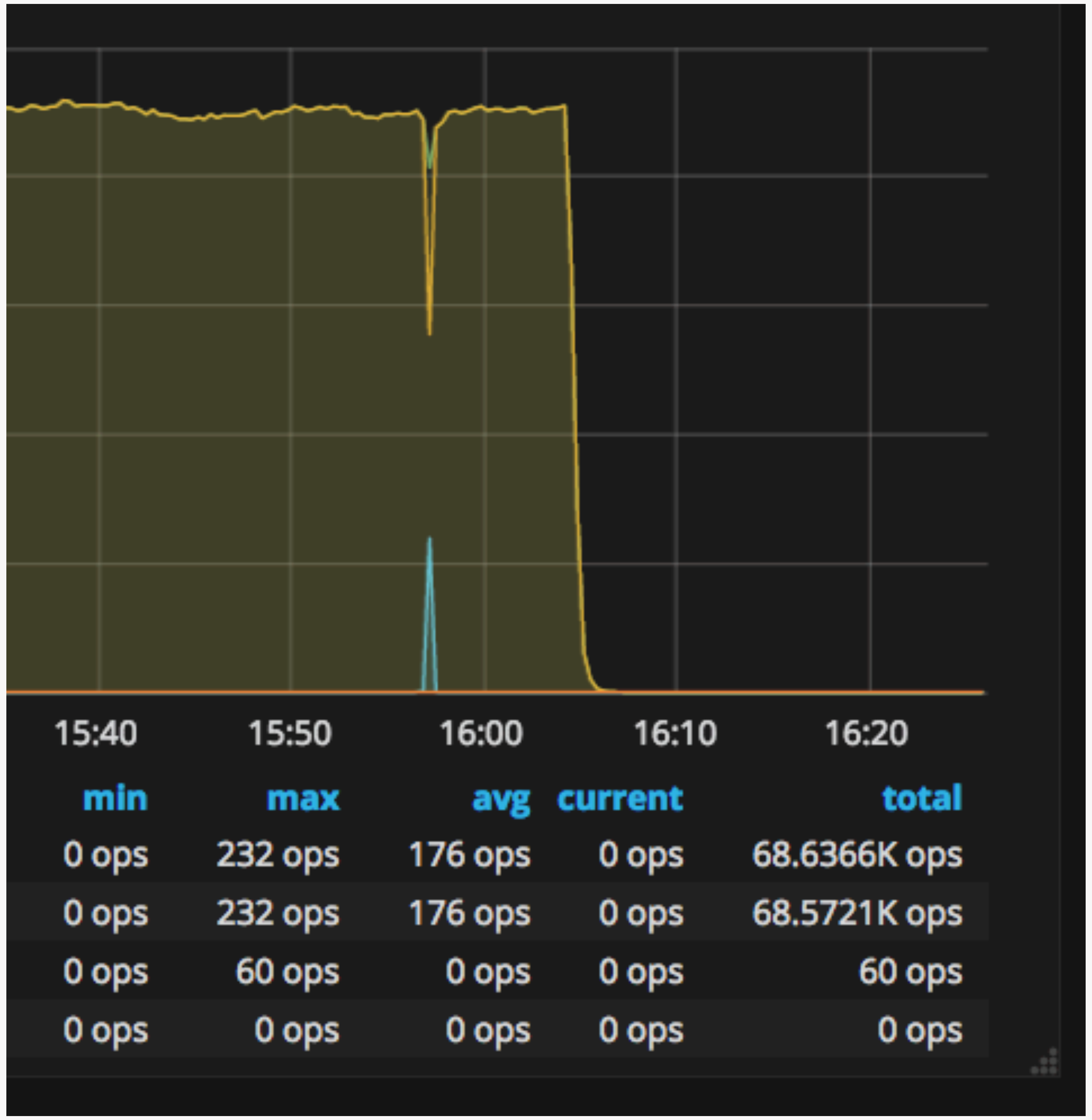
Service Mesh?

Result

1.1 Million Connections

- Subscriptions were failing
- VerneMQ nodes were being terminated
- Kubernetes brought them back up

Blocker: ?



min	max	avg	current	total
0 ops	232 ops	176 ops	0 ops	68.6366K ops
0 ops	232 ops	176 ops	0 ops	68.5721K ops
0 ops	60 ops	0 ops	0 ops	60 ops
0 ops	0 ops	0 ops	0 ops	0 ops

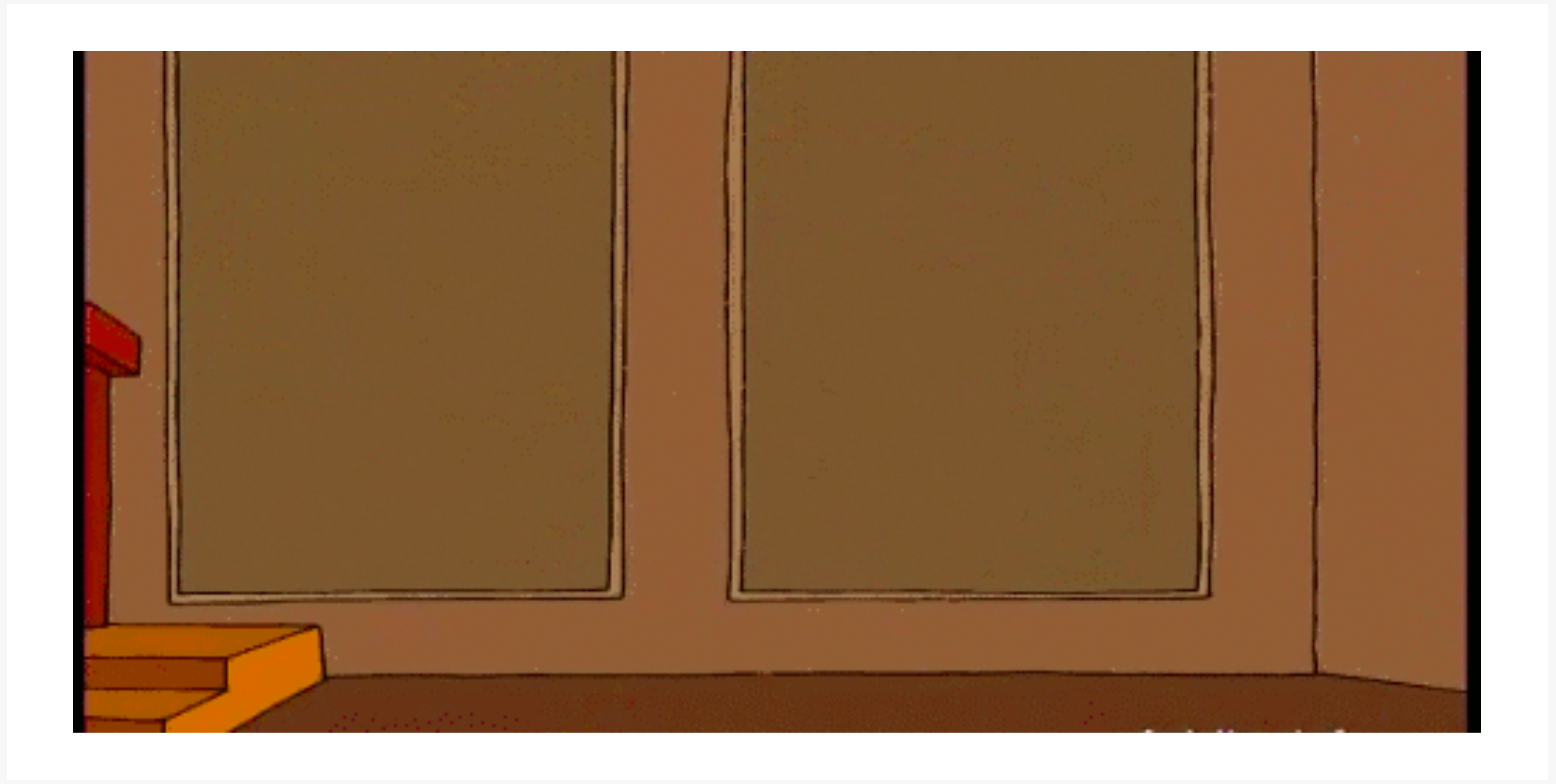
Diagnosing the Problem

- Scaled VerneMQ incrementally from 10 to 80 nodes
- Conclusion: resize/reallocation issue

Workaround

Exponential Backoff

- Modified clients to add custom behavior
- Delayed subscriptions to begin at decaying rate
- VerneMQ recovered



Result

1.5 Million Connections

Blocker: Resources - Erlang/OTP Scheduler

- Erlang schedulers went to 100% utilization
- Increasing resources didn’t help

Workaround

Reconfigure due to *cgroups*

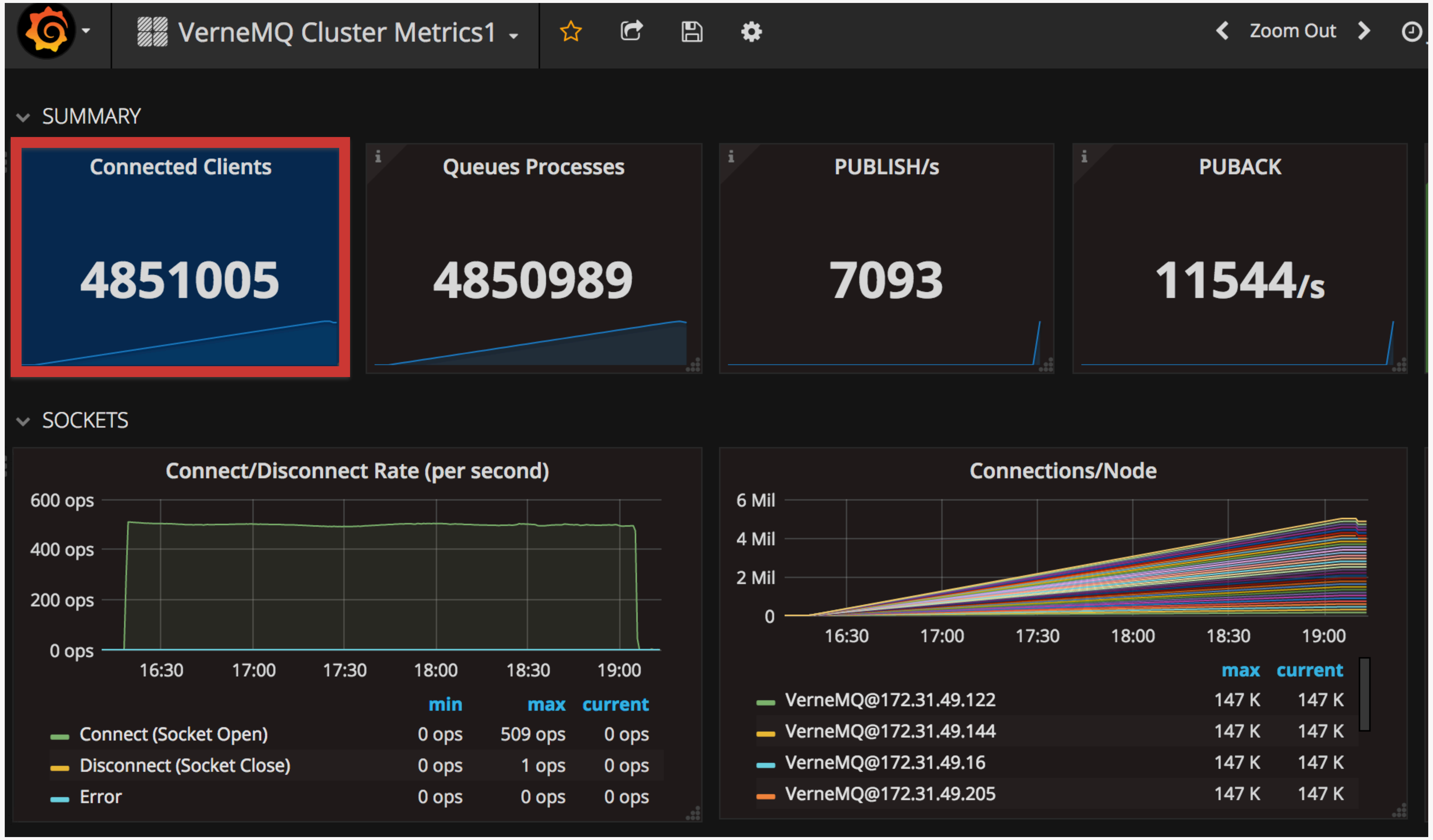
- Erlang/OTP is not *cgroup*-aware
- Directly configure vCPUs in Erlang for the scheduler

Result

4.85 Million Connections

Blocker:

Resources
Resources
Resources



	min	max	current
🟩 Connect (Socket Open)	0 ops	509 ops	0 ops
🟨 Disconnect (Socket Close)	0 ops	1 ops	0 ops
🟦 Error	0 ops	0 ops	0 ops

	max	current
🟩 VerneMQ@172.31.49.122	147 K	147 K
🟨 VerneMQ@172.31.49.144	147 K	147 K
🟦 VerneMQ@172.31.49.16	147 K	147 K
🟧 VerneMQ@172.31.49.205	147 K	147 K

5,000,001

Active WebSocket Connections

69 ms

Average latency for published message to reach subscriber

9,779

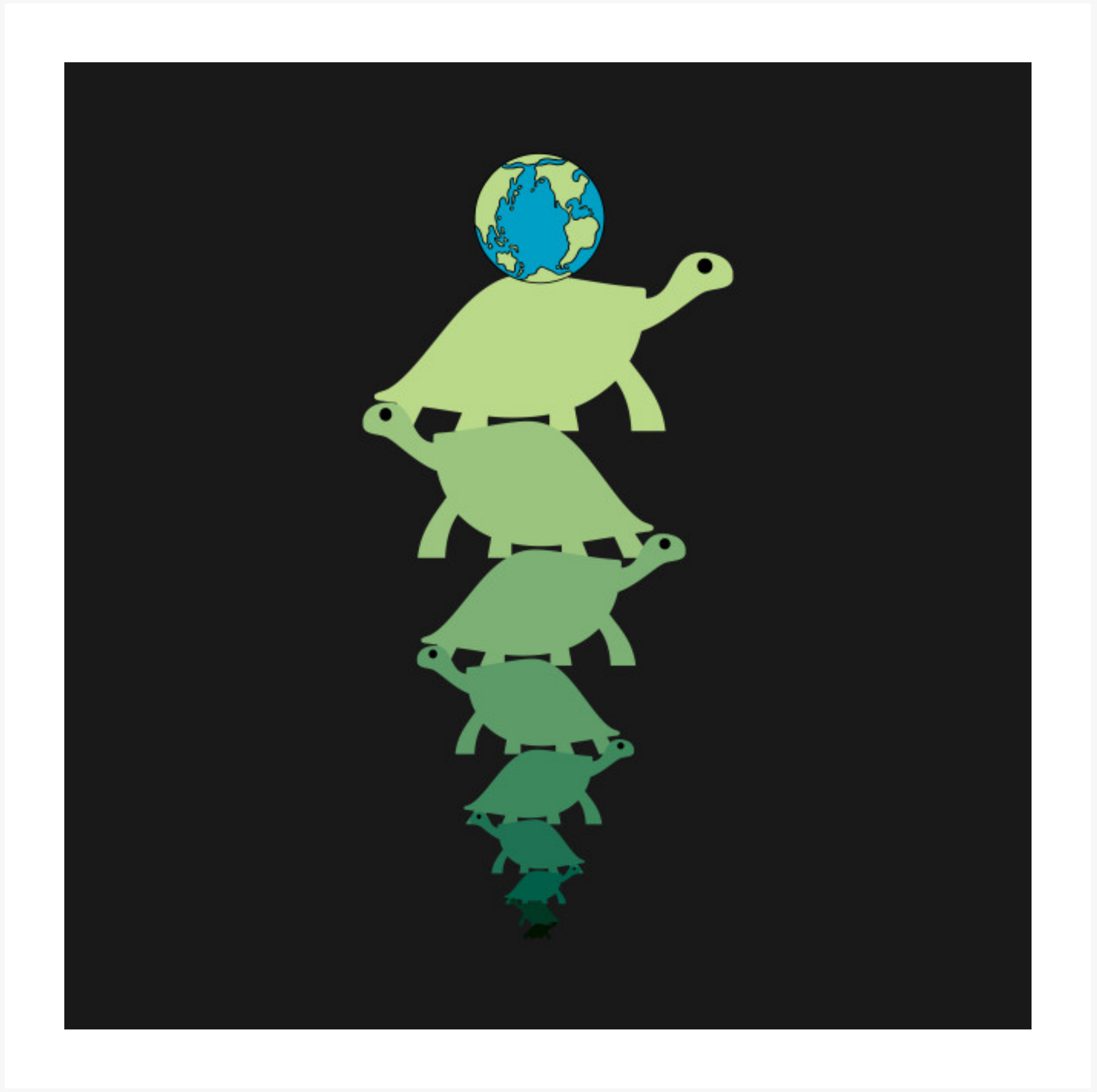
Average throughput of publishes per second

Success!

Key Learnings

1. Mind your dependencies

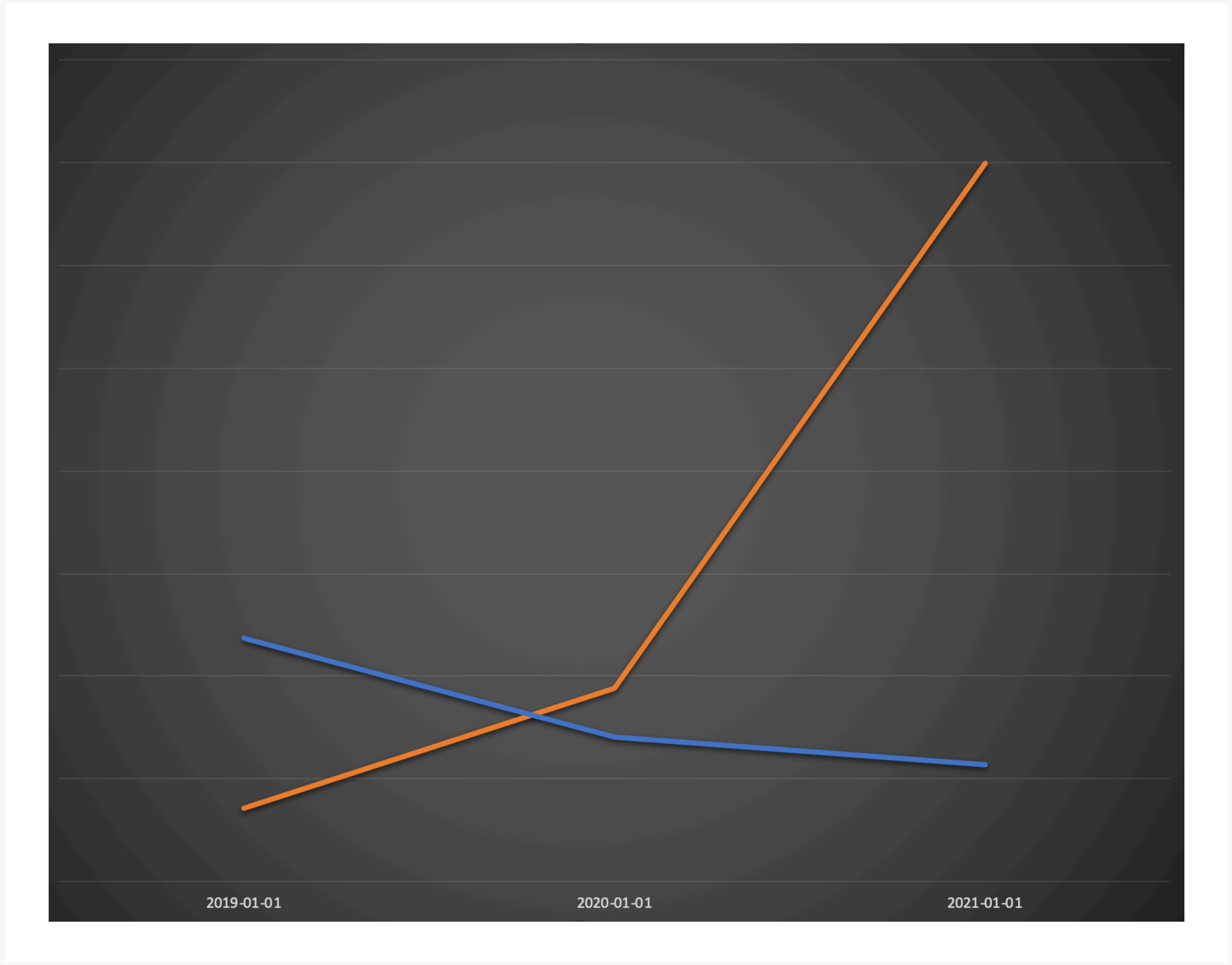
2. Experiment with resource limits



3. Layers complicate troubleshooting

4. Starting at scale is different than organic growth

Cost per device per annum



5. Our solution was *a lot* cheaper

Conclusion

HAPROXY

verneMQ

cassandra

Dec 12, 2018

Thank you.

@bose
@connectedio

Credits

Peter Chow-Wah
Software Engineer, Connected

Scott Wallace
Software Engineer, Connected

Eric Ko
Software Engineer, Connected

Thomas Aston
Lead Project Manager, Connected

Cameron Rowshanbin
Software Engineer, Connected

Kitty Chio
Project Manager, Connected

Special Thanks

Josh West
Principal Cloud Engineer
& Team Lead, Bose

Myles Steinhauser
Senior Cloud Engineer, Bose

Yiwei Chen
Cloud Engineer, Bose

Kevin Bralten
Solutions Engineer, Connected

@bose @connectedio