A Microservices Murder Mystery

Thomas Rampelberg

Honest Status Page
@honest_update

Follow

We replaced our monolith with micro services so that every outage could be more like a murder mystery.

4:10 PM - 7 Oct 2015

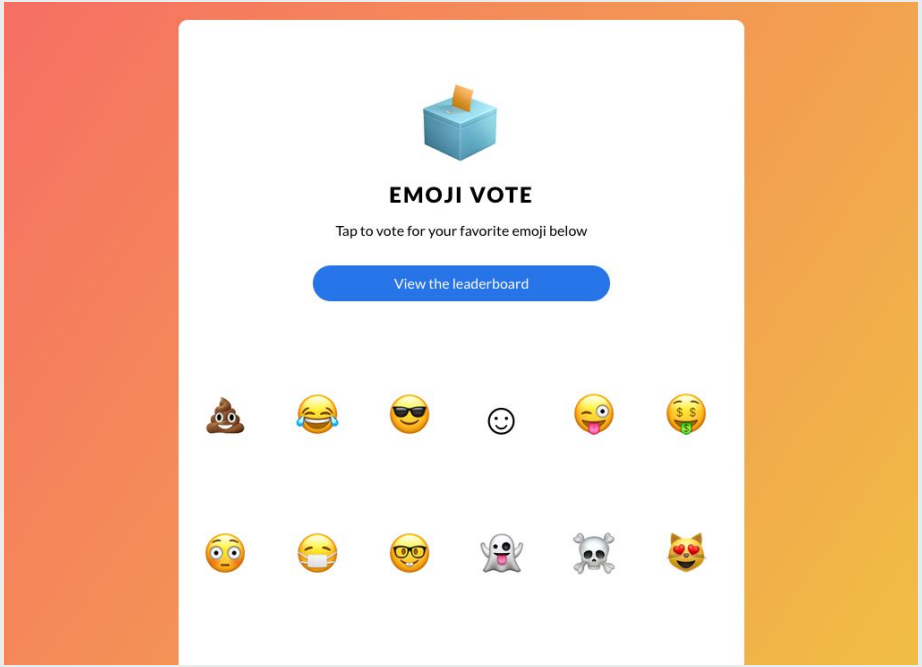
The Detective

Thomas Rampelberg

Software Engineer @ Buoyant

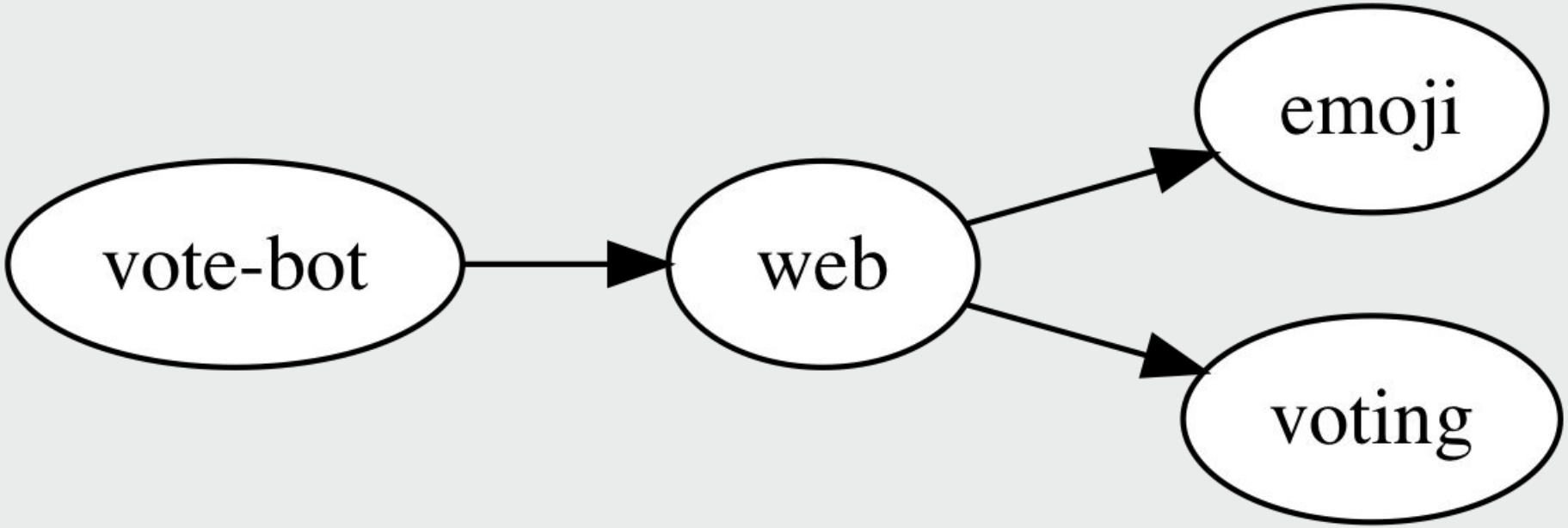
@grampelberg

The Scene of the Crime

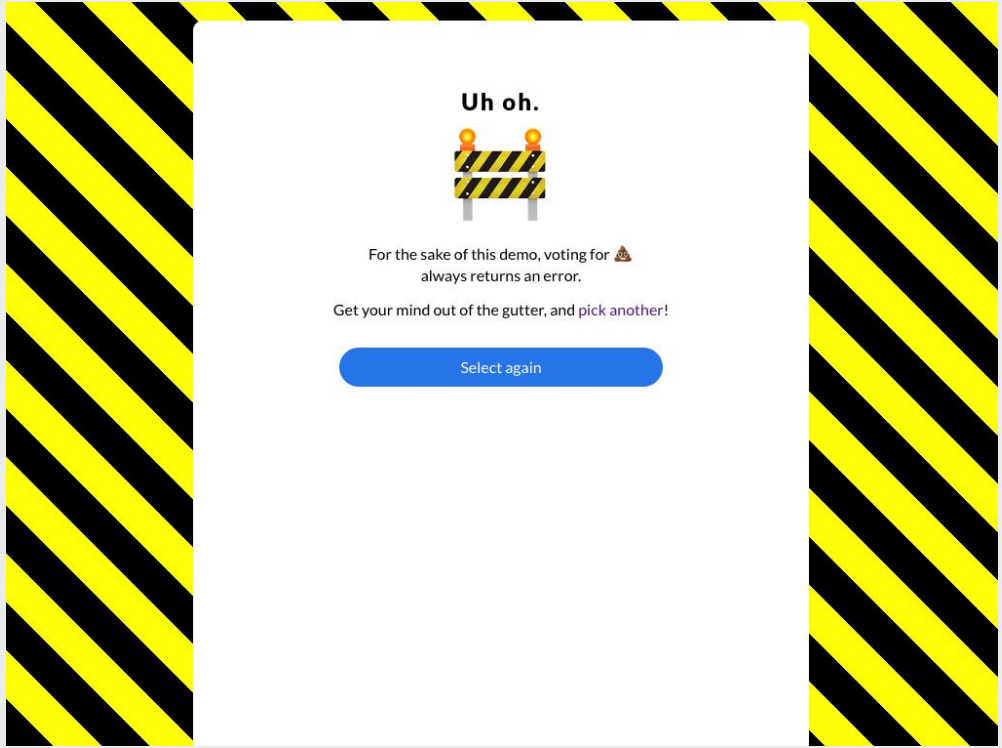


https://tidesf.l5d.io

The Suspects



The Murder

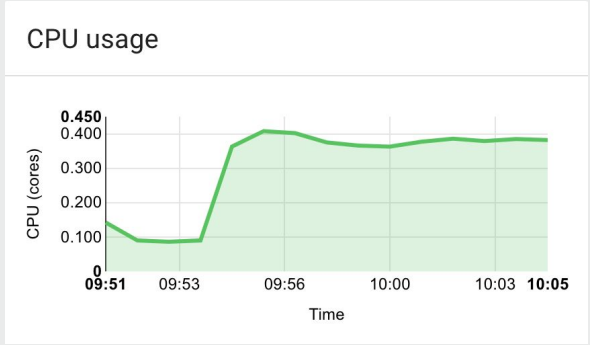

Uh oh.
For the sake of this demo, voting for 💩
always returns an error.
Get your mind out of the gutter, and pick another!
Select again

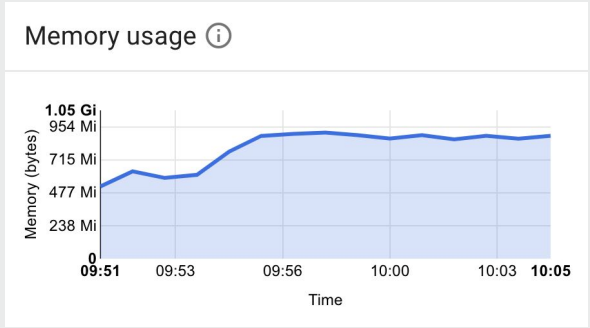
🙋 Question the Suspects!

- ☐ Service running?
- ☐ Health checks passing?
- ☐ Logs look okay?

Nothing out of the ordinary, constable

- [x] Service running!
- [ ] Health checks passing?
- [ ] Logs look okay?





Everything is fine

- [x] Service running!
- [x] Health checks passing!
- [ ] Logs look okay?

Deployments

Name	Labels	Pods
vote-bot	app: vote-bot	1 / 1
web	app: web-svc	1 / 1
emoji	app: emoji-svc	1 / 1
voting	app: voting-svc	1 / 1

🤷 Okay, *something* is fishy

- ✅ Service running!
- ✅ Health checks passing!
- ⁉️ Logs look … something?

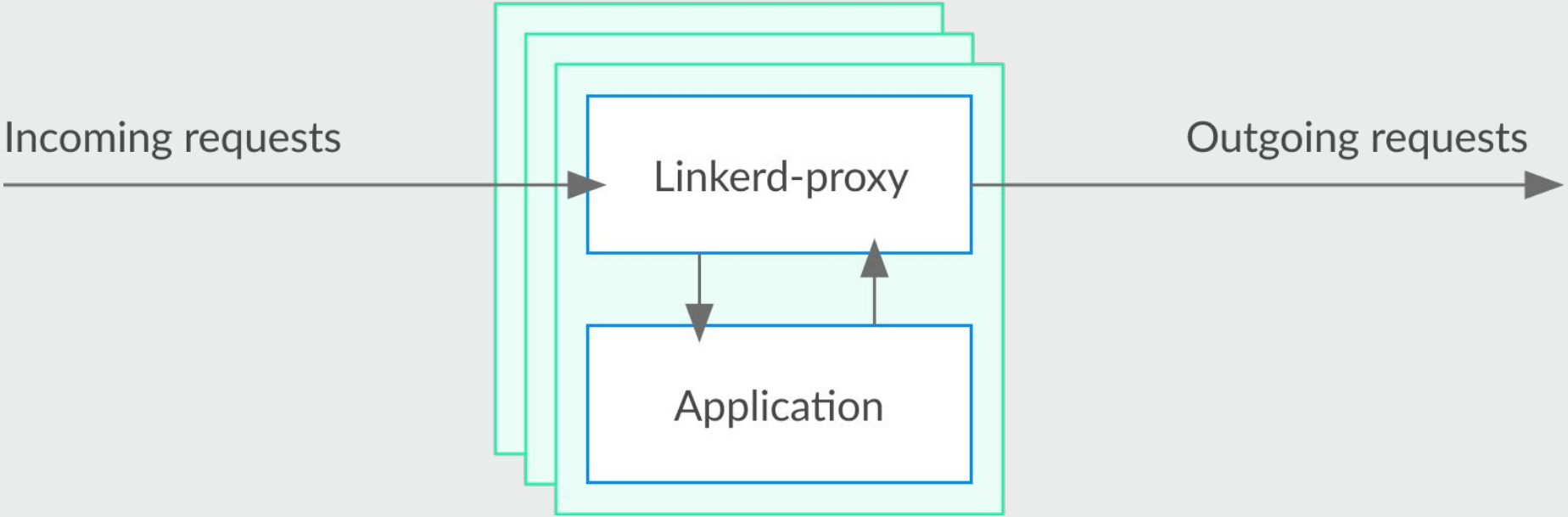
```
2018/11/13 22:26:58 Error serving request [&{GET /api/vote?choice=:poop: HTTP/1.1 1 1 map[User-Agent:[Go
-http-client/1.1] Accept-Encoding:[gzip]] {} <nil> 0 [] false web-svc.emojivoto:80 map[choice:[:poop:]]
map[] <nil> map[] 10.4.0.31:34560 /api/vote?choice=:poop: <nil> <nil> <nil> 0xc420481b80}]: rpc error: c
ode = Unknown desc = ERROR
2018/11/13 22:27:01 Error serving request [&{GET /api/vote?choice=:poop: HTTP/1.1 1 1 map[Accept-Encodin
g:[gzip] User-Agent:[Go-http-client/1.1]] {} <nil> 0 [] false web-svc.emojivoto:80 map[choice:[:poop:]]
map[] <nil> map[] 10.4.0.31:34590 /api/vote?choice=:poop: <nil> <nil> <nil> 0xc420481f00}]: rpc error: c
ode = Unknown desc = ERROR
2018/11/13 22:27:15 Error serving request [&{GET /api/vote?choice=:poop: HTTP/1.1 1 1 map[Accept-Encodin
g:[gzip] User-Agent:[Go-http-client/1.1]] {} <nil> 0 [] false web-svc.emojivoto:80 map[choice:[:poop:]]
map[] <nil> map[] 10.4.0.31:34618 /api/vote?choice=:poop: <nil> <nil> <nil> 0xc420480f80}]: rpc error: c
ode = Unknown desc = ERROR
2018/11/13 22:27:18 Error serving request [&{GET /api/vote?choice=:poop: HTTP/1.1 1 1 map[User-Agent:[Go
-http-client/1.1] Accept-Encoding:[gzip]] {} <nil> 0 [] false web-svc.emojivoto:80 map[choice:[:poop:]]
map[] <nil> map[] 10.4.0.31:34730 /api/vote?choice=:poop: <nil> <nil> <nil> 0xc420481380}]: rpc error: c
ode = Unknown desc = ERROR
2018/11/13 22:27:19 Error serving request [&{GET /api/vote?choice=:poop: HTTP/1.1 1 1 map[User-Agent:[Go
-http-client/1.1] Accept-Encoding:[gzip]] {} <nil> 0 [] false web-svc.emojivoto:80 map[choice:[:poop:]]
map[] <nil> map[] 10.4.0.31:34754 /api/vote?choice=:poop: <nil> <nil> <nil> 0xc420061280}]: rpc error: c
ode = Unknown desc = ERROR
2018/11/13 22:27:24 Error serving request [&{GET /api/vote?choice=:poop: HTTP/1.1 1 1 map[User-Agent:[Go
-http-client/1.1] Accept-Encoding:[gzip]] {} <nil> 0 [] false web-svc.emojivoto:80 map[choice:[:poop:]]
map[] <nil> map[] 10.4.0.31:34766 /api/vote?choice=:poop: <nil> <nil> <nil> 0xc420061980}]: rpc error: c
ode = Unknown desc = ERROR
2018/11/13 22:27:33 Error serving request [&{GET /api/vote?choice=:poop: HTTP/1.1 1 1 map[User-Agent:[Go
-http-client/1.1] Accept-Encoding:[gzip]] {} <nil> 0 [] false web-svc.emojivoto:80 map[choice:[:poop:]]
map[] <nil> map[] 10.4.0.31:34798 /api/vote?choice=:poop: <nil> <nil> <nil> 0xc420060040}]: rpc error: c
ode = Unknown desc = ERROR
2018/11/13 22:27:38 Error serving request [&{GET /api/vote?choice=:poop: HTTP/1.1 1 1 map[User-Agent:[Go
-http-client/1.1] Accept-Encoding:[gzip]] {} <nil> 0 [] false web-svc.emojivoto:80 map[choice:[:poop:]]
map[] <nil> map[] 10.4.0.31:34864 /api/vote?choice=:poop: <nil> <nil> <nil> 0xc420060880}]: rpc error: c
ode = Unknown desc = ERROR
```

Forensics

1. Stand up a monitoring solution
2. Pick a dashboard solution
3. Build some dashboards
4. Instrument the services
5. ....
6. Profit?

Linkerd: Microservice Forensics

What is Linkerd?



Introducing linkerd 🎈

```
linkerd is a dynamic linker for distributed applications (aka
"microservices").  In the same way that `ld(1)` binds software
components (libraries), linkerd binds services by mediating
inter-service communication (RPC).

linkerd builds upon finagle & netty--Twitter's JVM networking
stack--and it exposes many of the advanced operational features
developed by Twitter, Soundcloud, and other large internet
applications.

linkerd provides a minimal configuration structure that describes
general rules for linking RPC requests to a service discovery system
(i.e. _namers_).  This configuration system, built on Jackson, exposes
a module system so that additional functionality may be instrumented
at package- or deploy-time.

This repository currently includes only _libraries_ for building
linkerd.  The `linkerd-daemon` package provides a Main that is capable
of initializing linkerd; however, because it does not depend on any of
the protocol or namer modules, it is not suitable to execute this main
without additional build configuration (TBD).

Current protocol support:
- http
- thrift [experimental]
- mux [experimental]

Current namer support:
- file-system based discovery
- kubernetes master [experimental]

linkerd is under active development, and so current APIs should not be
considered stable until the project reaches version 1.0.0.
```

master release-0.0.11 ··· 0.1.0

olix0r committed on Jan 13, 2016

24+ months in production

2k+ Slack channel members

7,000+ GitHub stars

20m+ DockerHub pulls

80+ contributors

400b+ production requests/mo

Forensics

- [ ] Stand up a monitoring solution
- [ ] Pick a dashboard solution
- [ ] Build some dashboards
- [ ] Instrument the services
- [ ] Track down the root cause

Forensics

- [x] Stand up a monitoring solution
- [ ] Pick a dashboard solution
- [ ] Build some dashboards
- [ ] Instrument the services
- [ ] Track down the root cause

Forensics

- [x] Stand up a monitoring solution
- [x] Pick a dashboard solution
- [ ] Build some dashboards
- [ ] Instrument the services
- [ ] Track down the root cause

Forensics

- [x] Stand up a monitoring solution
- [x] Pick a dashboard solution
- [x] Build some dashboards
- [ ] Instrument the services
- [ ] Track down the root cause

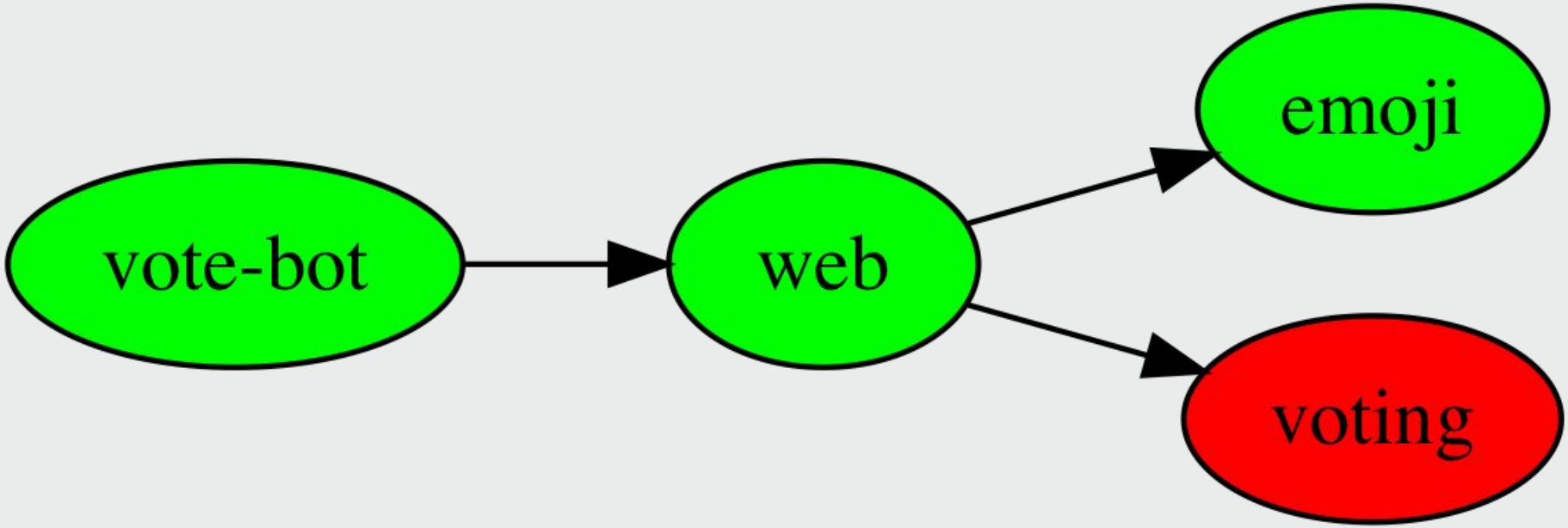
Forensics

- [x] Stand up a monitoring solution
- [x] Pick a dashboard solution
- [x] Build some dashboards
- [x] Instrument the services
- [ ] Track down the root cause

Forensics

- [x] Stand up a monitoring solution
- [x] Pick a dashboard solution
- [x] Build some dashboards
- [x] Instrument the services
- [x] Track down the root cause

The Culprit



What is possible?

- Alerting
- Anomaly Detection
- Autoscaling
- Capacity Planning
- Chargeback
- Intelligent Deployments
- SLA Compliance

LINKERD

Slides	http://bit.ly/linkerd-tidesf
Tutorial	http://bit.ly/tidesf-tutorial
Get Started!	https://bit.ly/linkerd-get-started
Kubecon	http://bit.ly/linkerd-kubecon