high reliability infrastructure migrations

stripe

JULIA EVANS
@bØrk

about me

infrastructure
engineer
@stripe

payments company
billions of dollars / year

our challenges:

~~10 million QPS~~

~~sub-millisecond latency~~

RELIABILITY

SECURITY

99.99%

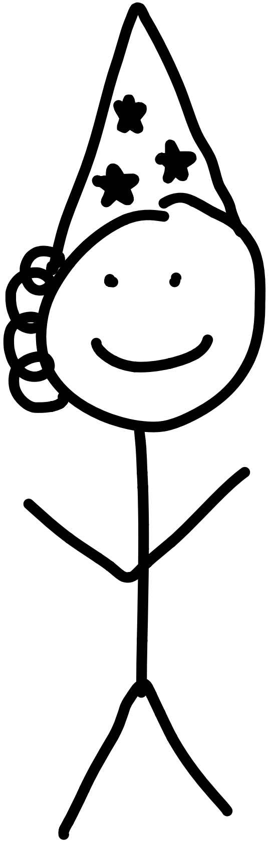
1 minute / week

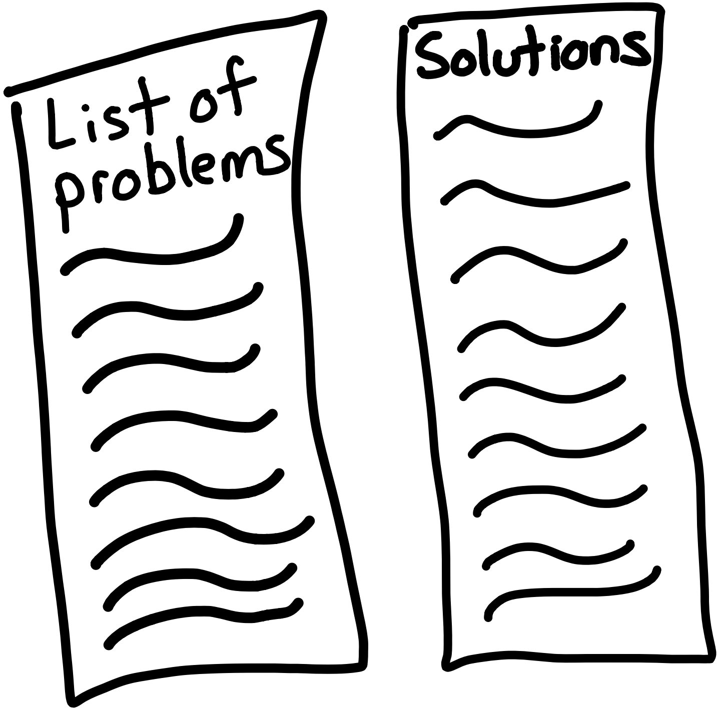
we made 2 changes

→ move some workloads to kubernetes

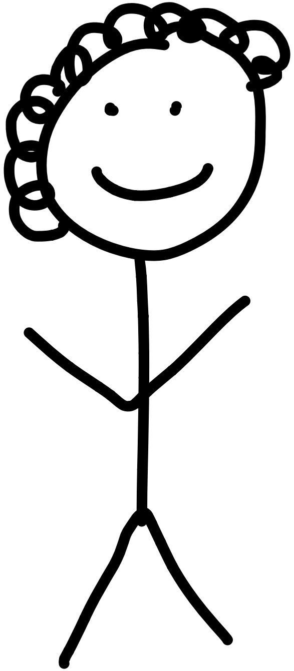
→ use Envoy for all service-to-service networking

Wish

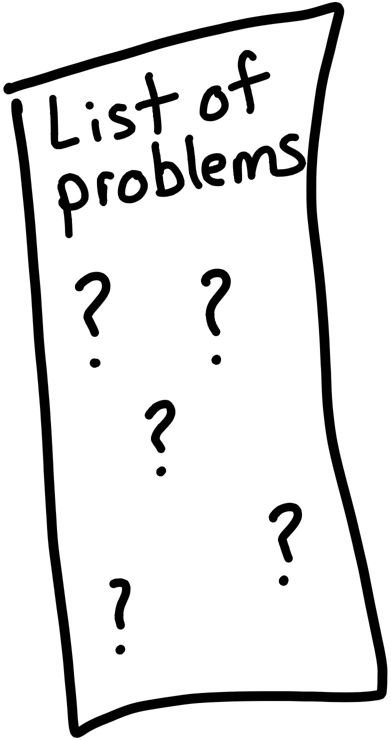


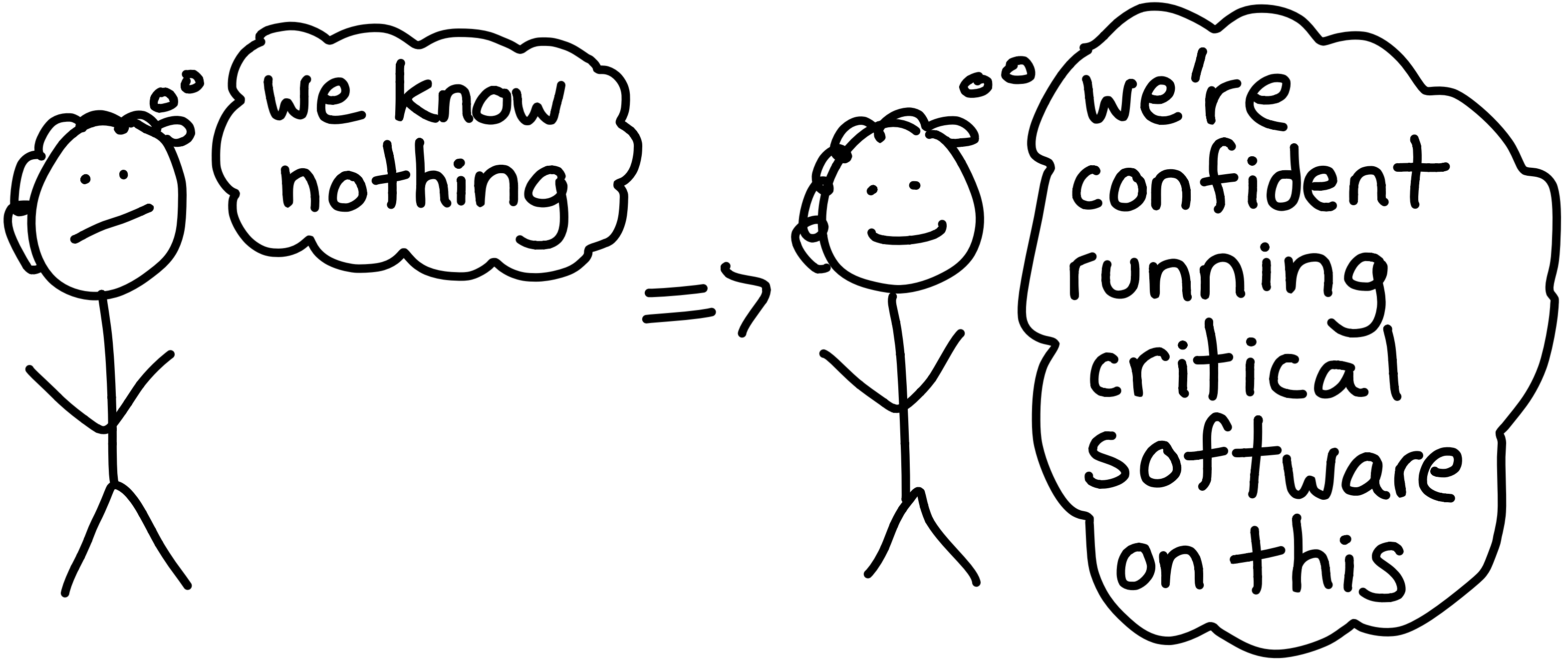


Reality

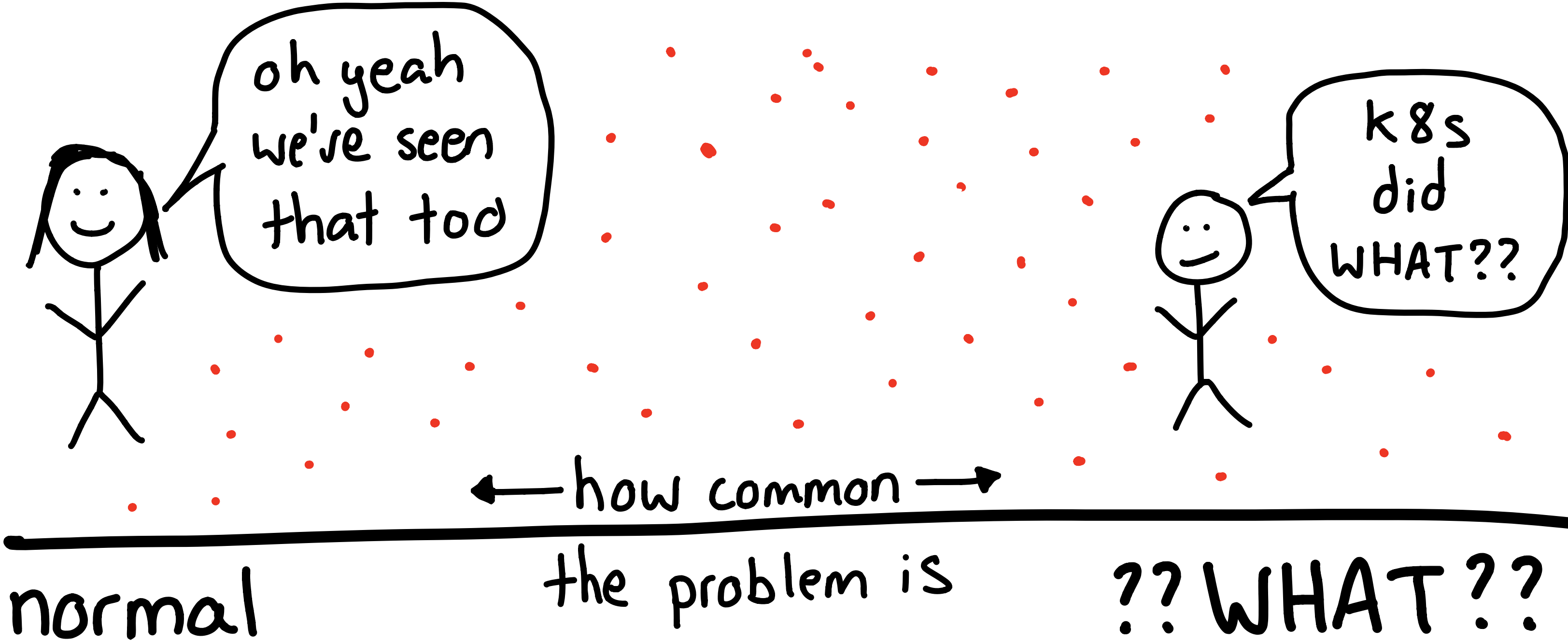


(this was me)

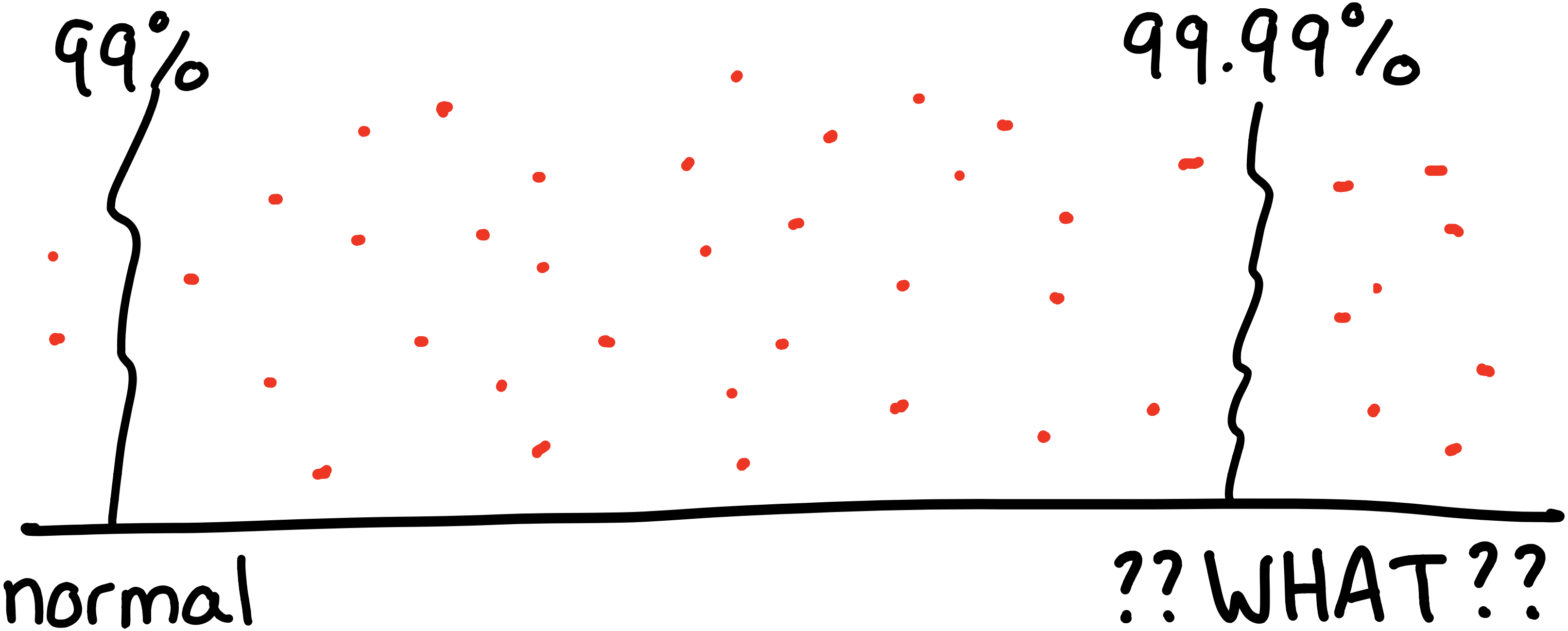



we know nothing
=>
we're confident running critical software on this

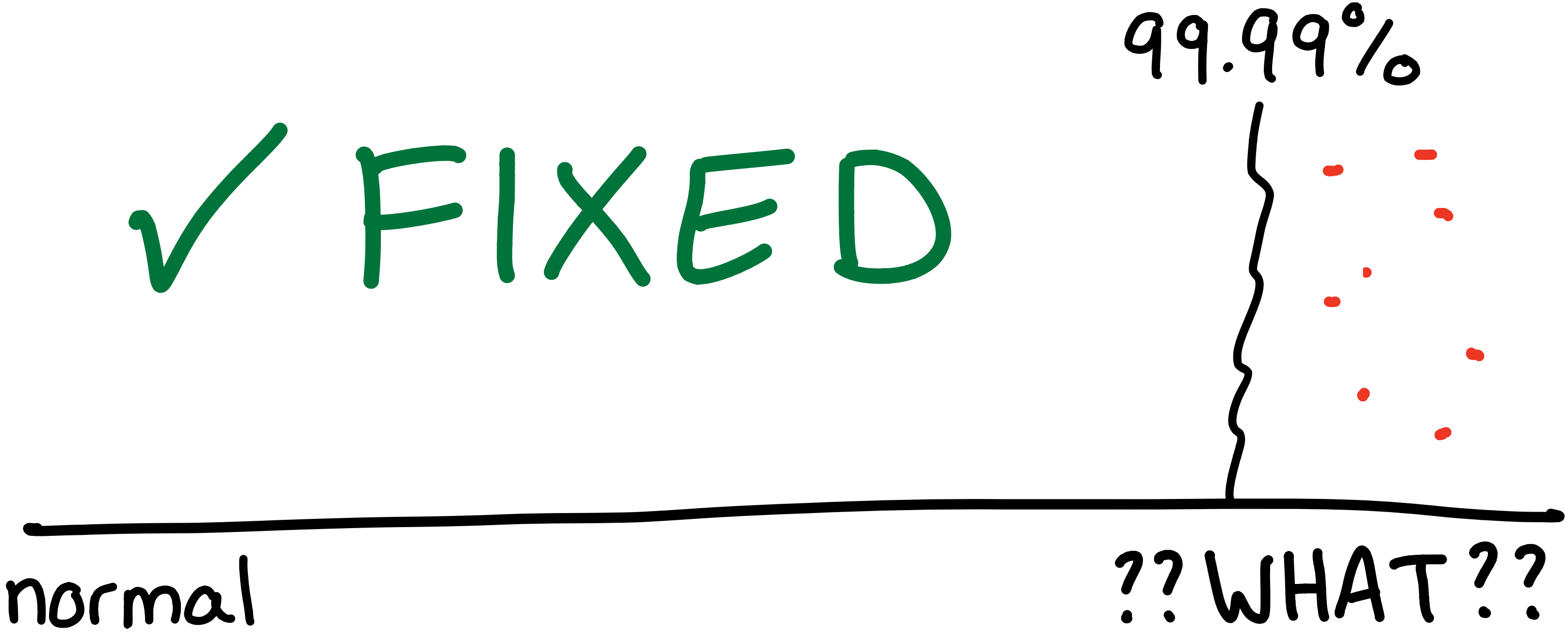
what could go wrong



what could go wrong



the goal

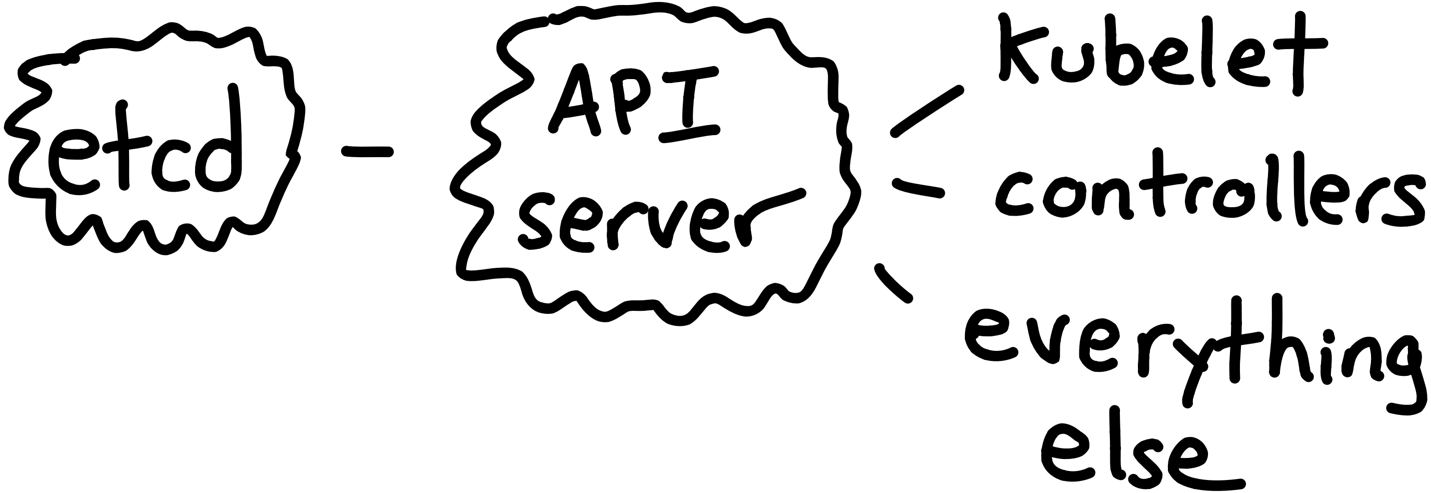


how to get there

- understand the design
- run gamedays
- classify your failures
- have incidents only once
- make incremental changes
- always have a rollback plan

Understand kubernetes' design

k8s design



understand the design

ignore (most) new software

Kubernetes Envoy

<u>that's it</u>

theory isn't enough

how can kubernetes break

learn how your system breaks

cause problems on purpose

culture of learning & rigor

"gamedays"

"DiRT"

weekly cadence

Run gamedays

gamedays:

→ test how your system behaves under known failures

→ let you learn without duress

→ share knowledge

Run gamedays

- terminate an etcd instance
- push invalid configuration
- destroy all apiserver instances (or just 1)
- container registry outage
- take down Envoy control plane

Run these in QA, but also in production!

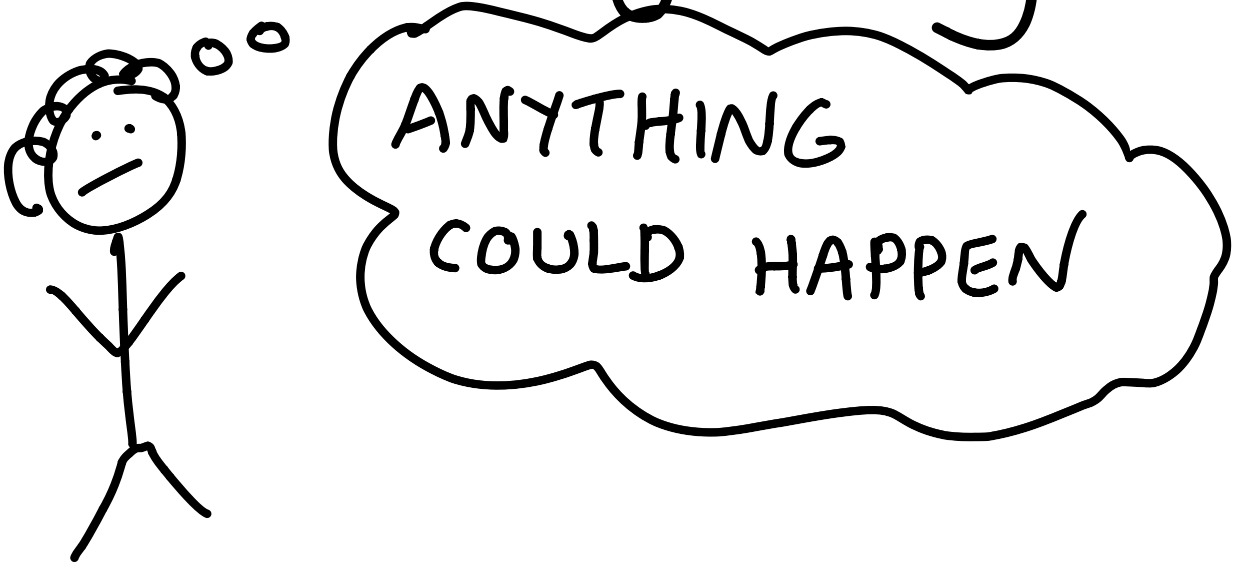
Run gamedays

Kubernetes terminated every running pod in the cluster "pod eviction"

We fixed & then tested the fix

classify your
failure modes

at the beginning



~all our failure modes

→ containers don't start

→ permissions errors

→ networking issues

learn your failure modes

Reasons pods don't start:

- IAM rate limiting
- scheduler bug
- etcd is down
- lots more

so many reasons

classification

=> monitoring

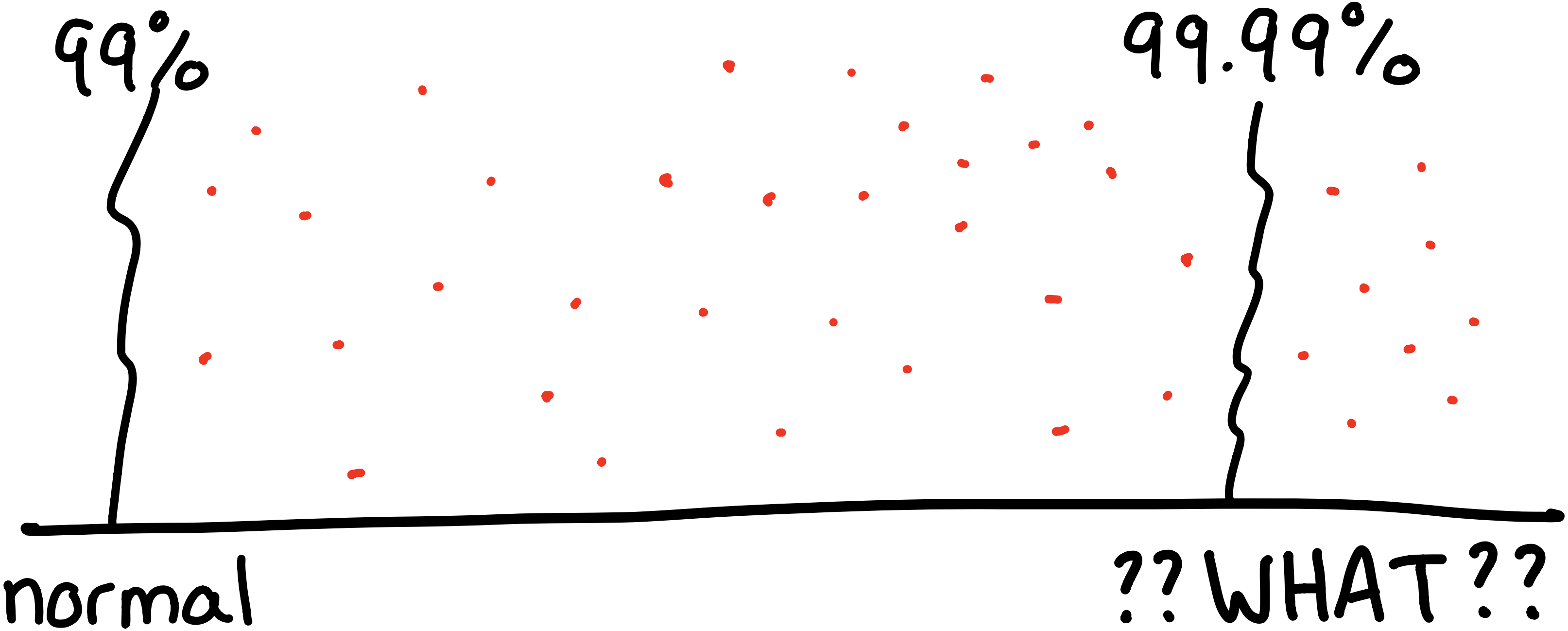
♥ heartbeat job ♥

Have every incident ★ only once ★

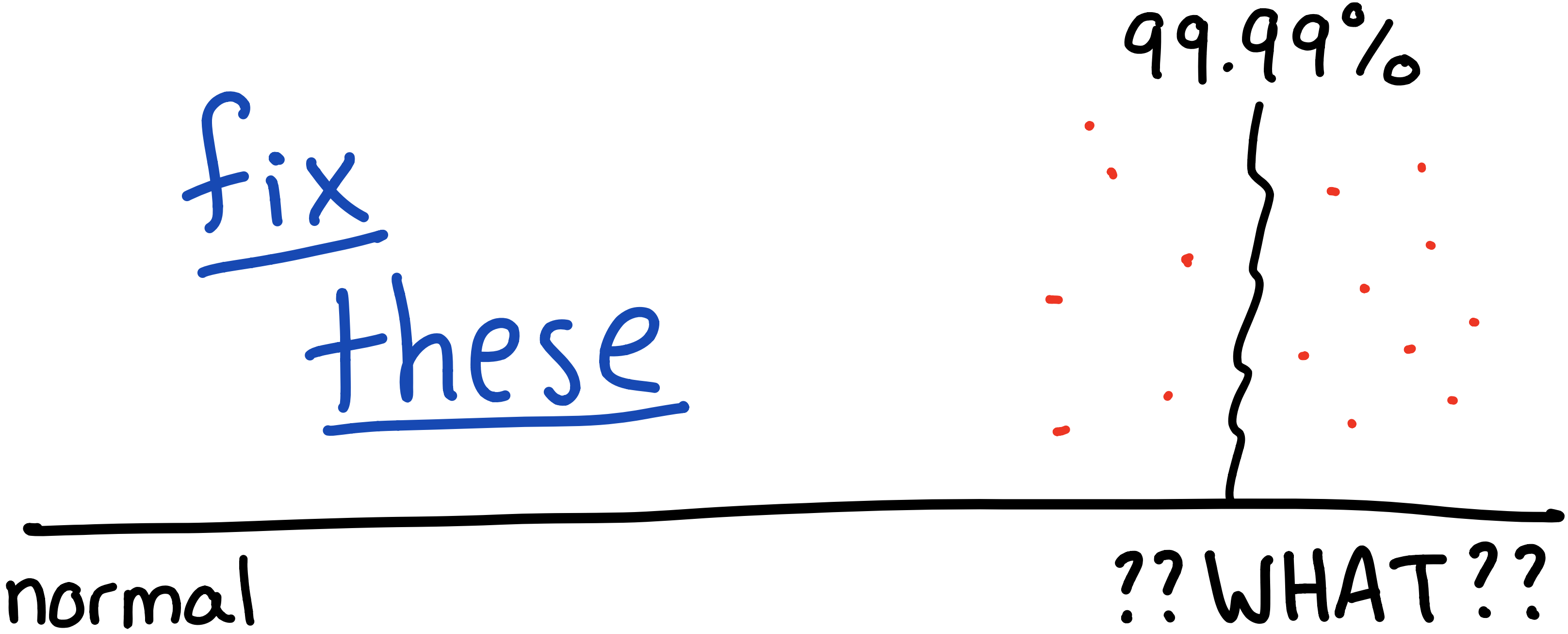
Have incidents only once

If you don't understand your incidents, they will happen again

Your problem space



Your problem space



Have incidents only once

1. Find a problem
2. Find causes
3. Implement remediations
4. Problem never comes back*

* usually

Fix <u>categories</u>

of incidents

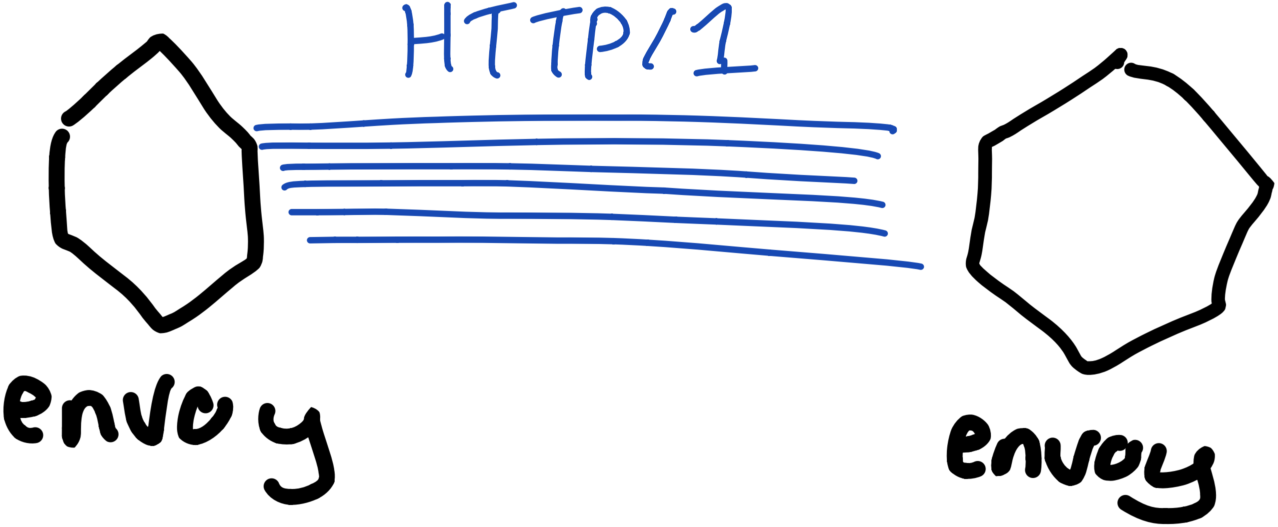
some Envoy issues

request timeouts

connection timeouts

slow request

thundering herd

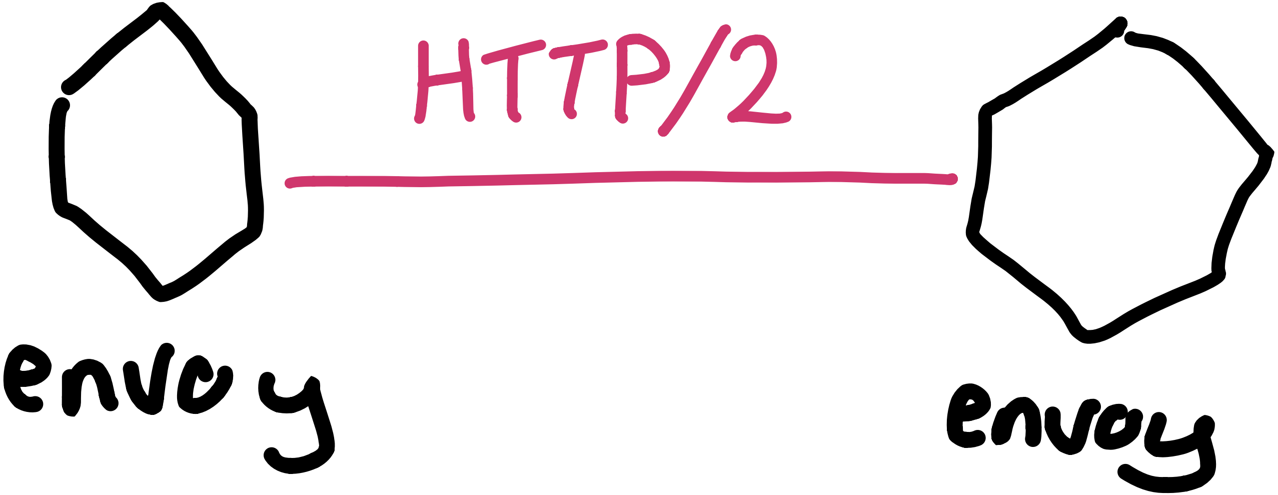


all HTTP/1 conn pool issues

Solution:

Use HTTP/2

(Envoy is designed for HTTP/2)

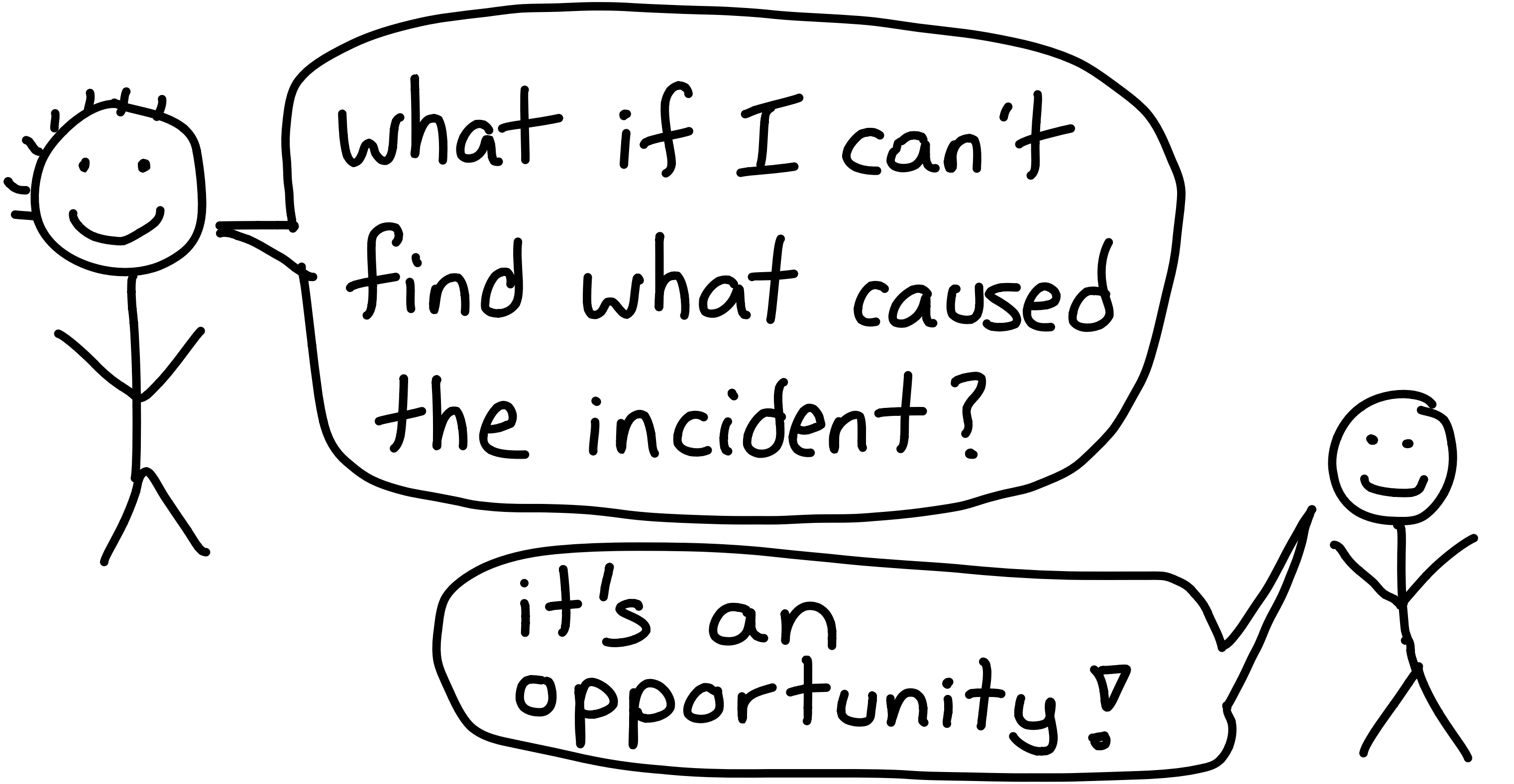

HTTP/2
envoy
envoy

tell your coworkers what you learned

♥ incident reports ♥

example: etcd EBS issue
throttling => leader elections

Have incidents only once



incidents teach you how to build a reliable system

make incremental changes

make incremental changes

- 5% of traffic
- 1 host
- a non-critical service

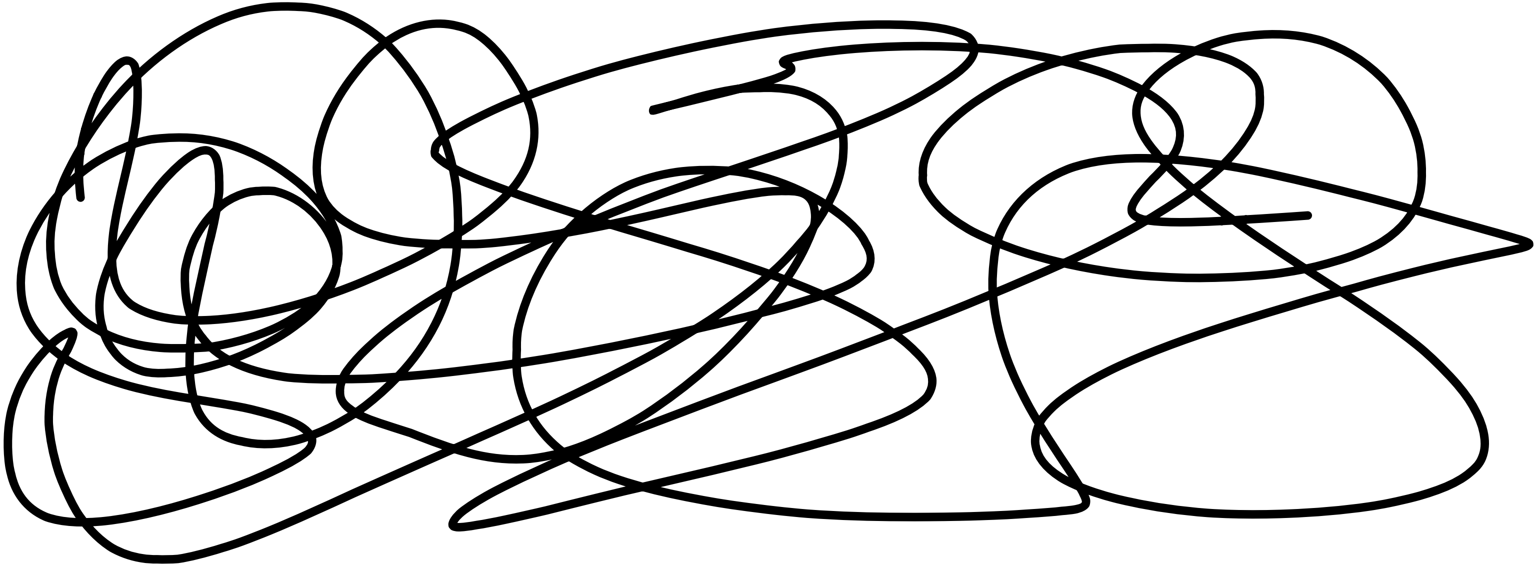
make incremental changes

establish an interface boundary

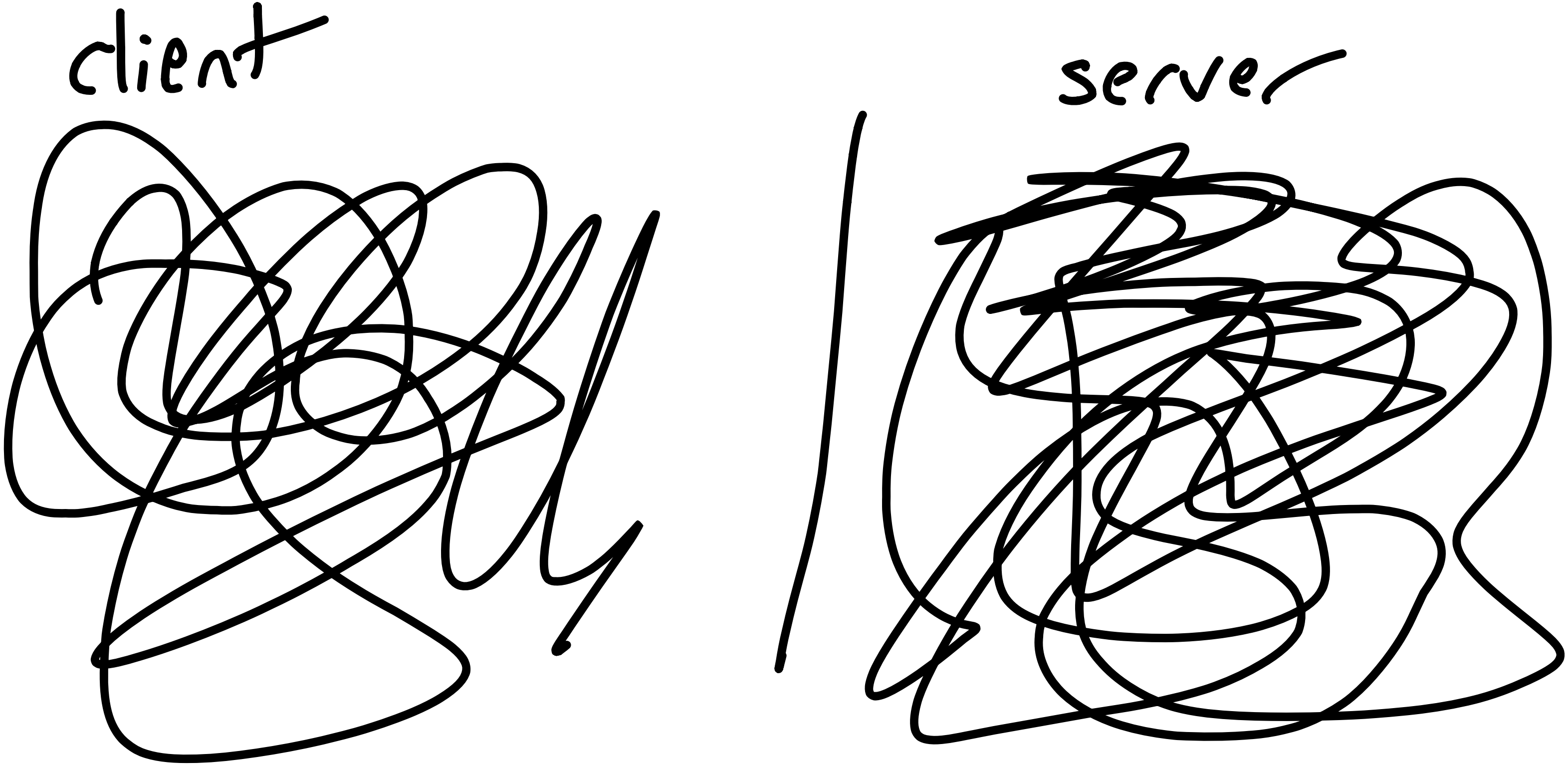
make incremental changes

our deploy system

circa 2017



make incremental changes



make incremental changes

new client

make incremental changes

new client

new server

“no haunted forests”

make incremental changes

don't expose kubernetes to developers

① reduce cognitive load

② reduce support burden

escape from YAML:

★ skycfg ★

skycfg.fun

YAML

```
name: missing-review-finder
owner: risk
schedule: 30 0 * * *
disabled: false
command:
- ruby
- scripts/cron/risk-missing-review-finder
```

- what other attributes are supported?
- what k8s config does it generate?

♥ code ♥

```
return stripe_service(
    image = default_image,
    command = einhorn(henson_service = "home-srv",
        script = "home/srv.rb",
        workers = 8,
        port = 9768,
    ),
    iam_role = "homesrv.kube.%s.%s" % (
        ctx.vars["stripe.cluster"],
        ctx.vars["stripe.environment"],
    ),
    replicas = 3,
    cpu = kube.cores(4),
    mem = kube.gigabytes(16),
    block_egress = False,
)
```

- subset of Python
- typechecked
- sandboxed

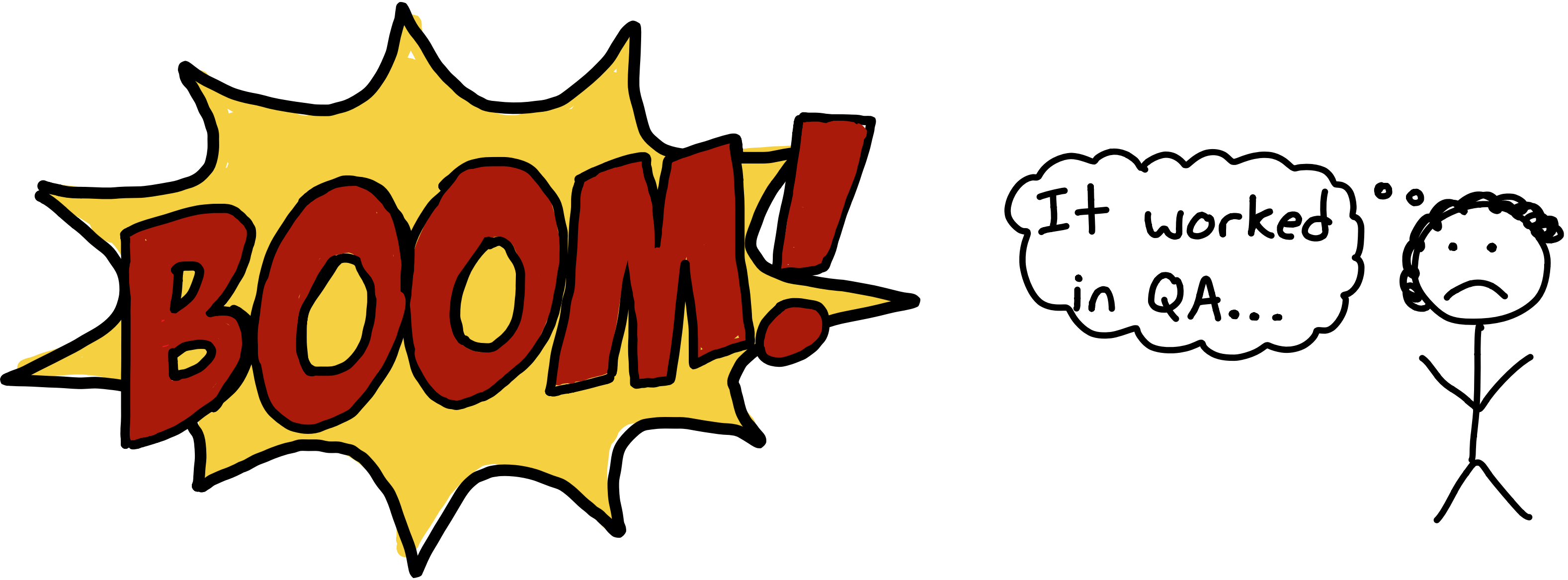
github.com/stripe/skycfg

skycfg.fun

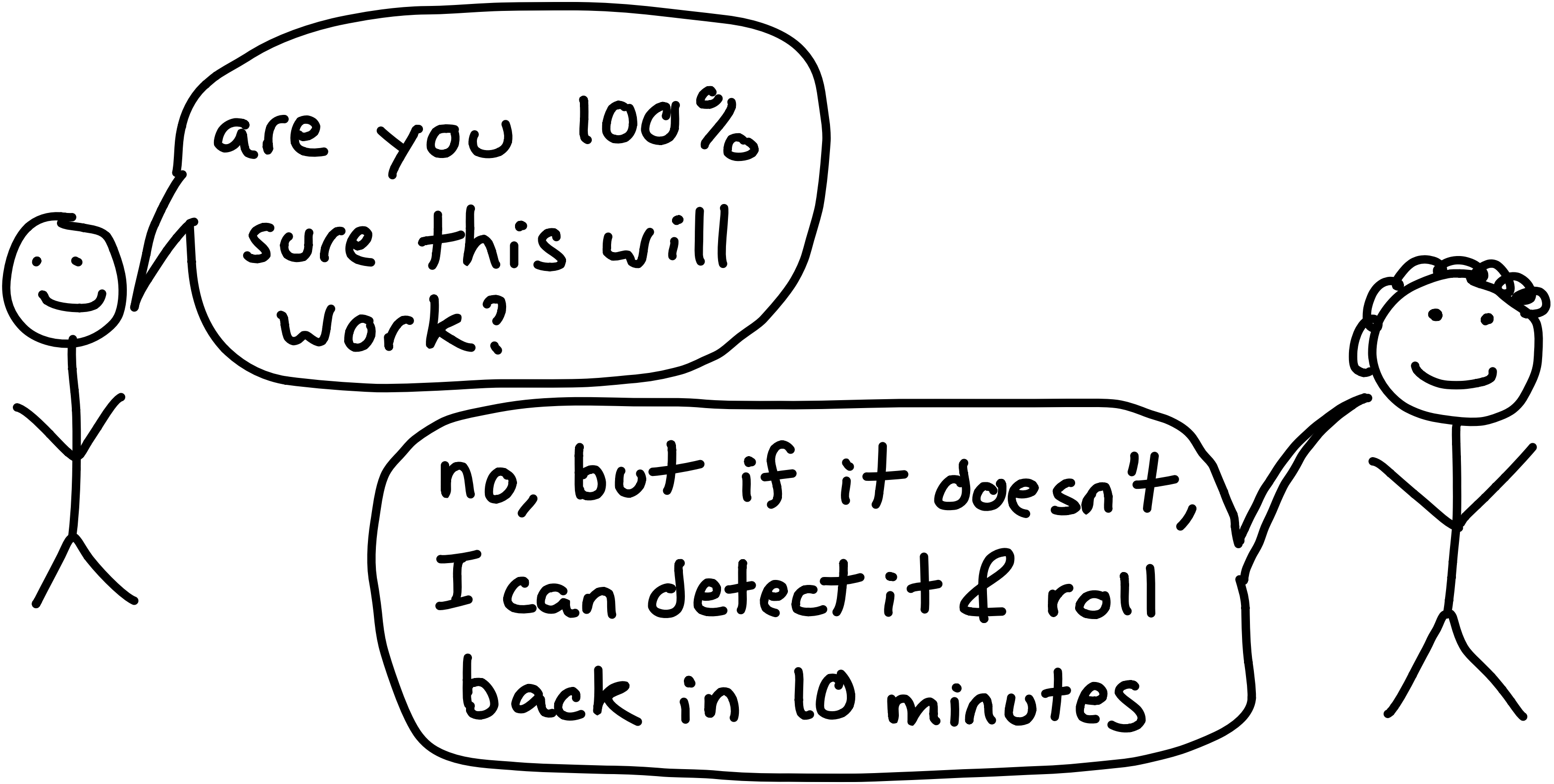
always have a

♥ rollback plan ♥

have a rollback plan



have a rollback plan


are you 100% sure this will work?
no, but if it doesn't, I can detect it & roll back in 10 minutes

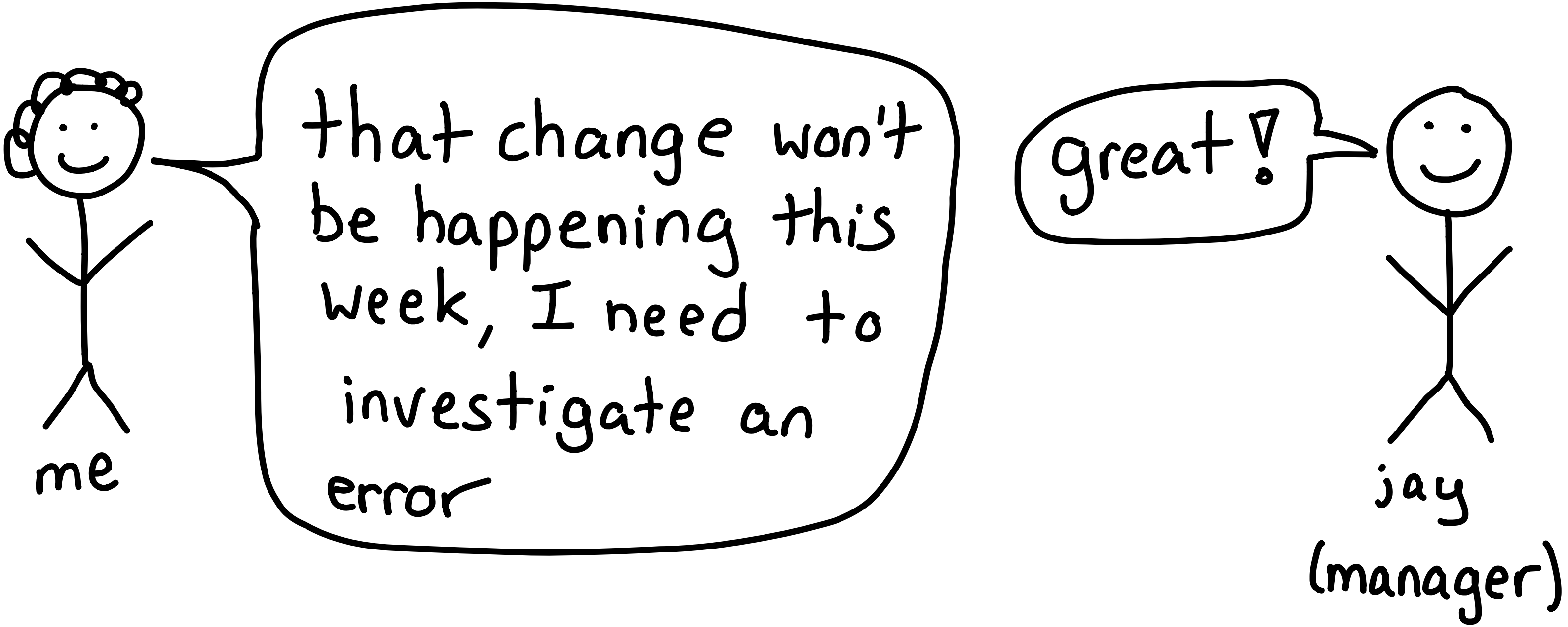
★ playbook ★

- understand the design
- run gamedays
- classify your failures
- have incidents only once
- make incremental changes
- always have a rollback plan

culture & leadership

- it's ok to start out not being an expert but you need to become one
- build an engine of learning
- building that expertise takes time

managers: make space for your team to learn


that change won't be happening this week, I need to investigate an error
me
great!
jay (manager)

thanks

a lot!

ps: we're hiring in Seattle!

stripe.com/jobs

intro	0-4
genedays	4-7
modes	7-9
only once	9-14:30
incremental	14:30-18
rollback	18-19
end	19-22