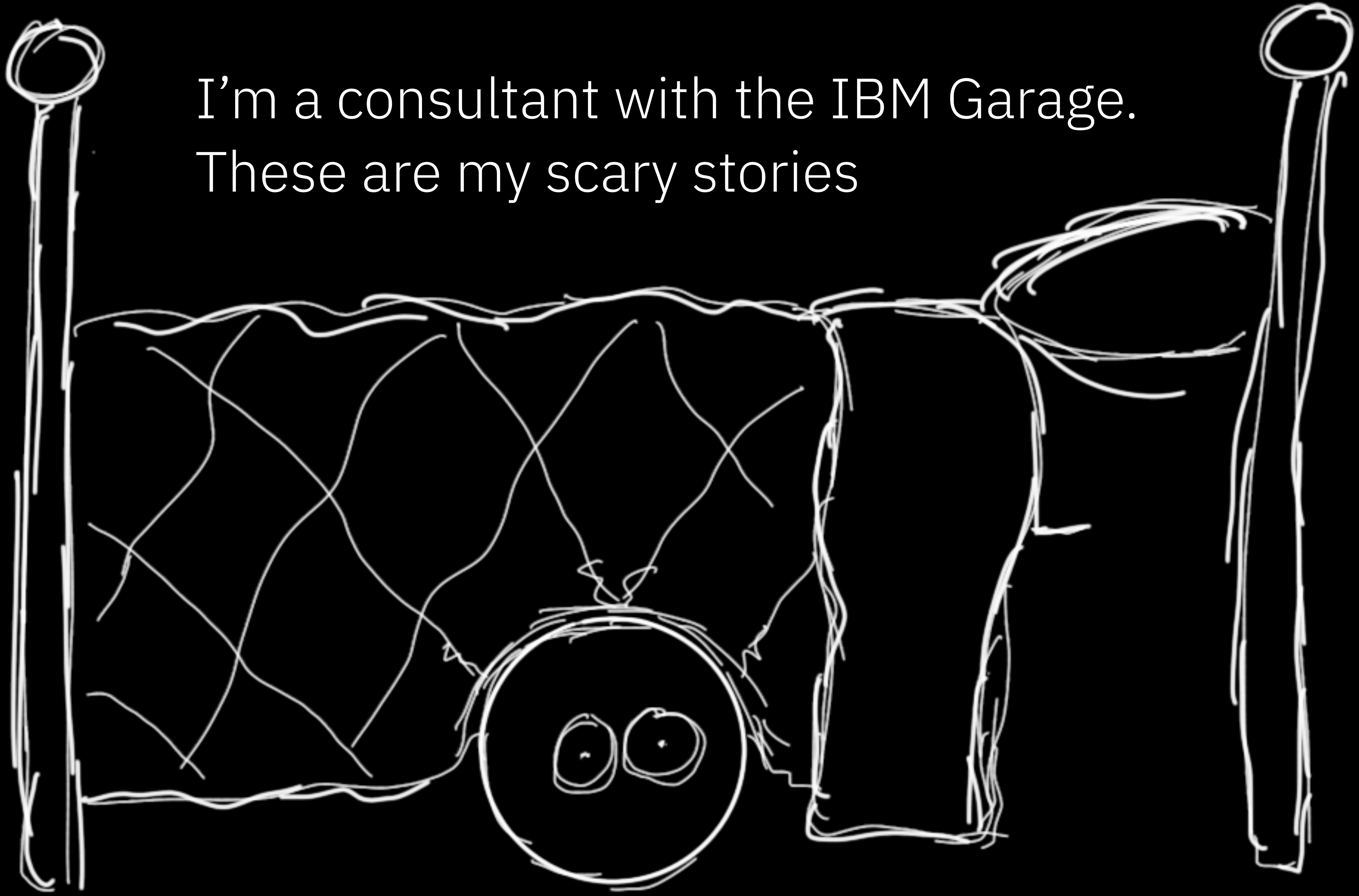
Blood-Curdling Tales of Microservices Misadventure, DevOps Dread, Grisly Governance

Holly Cummins
IBM **Garage**
@holly_cummins

I’m a consultant with the IBM Garage.
These are my scary stories



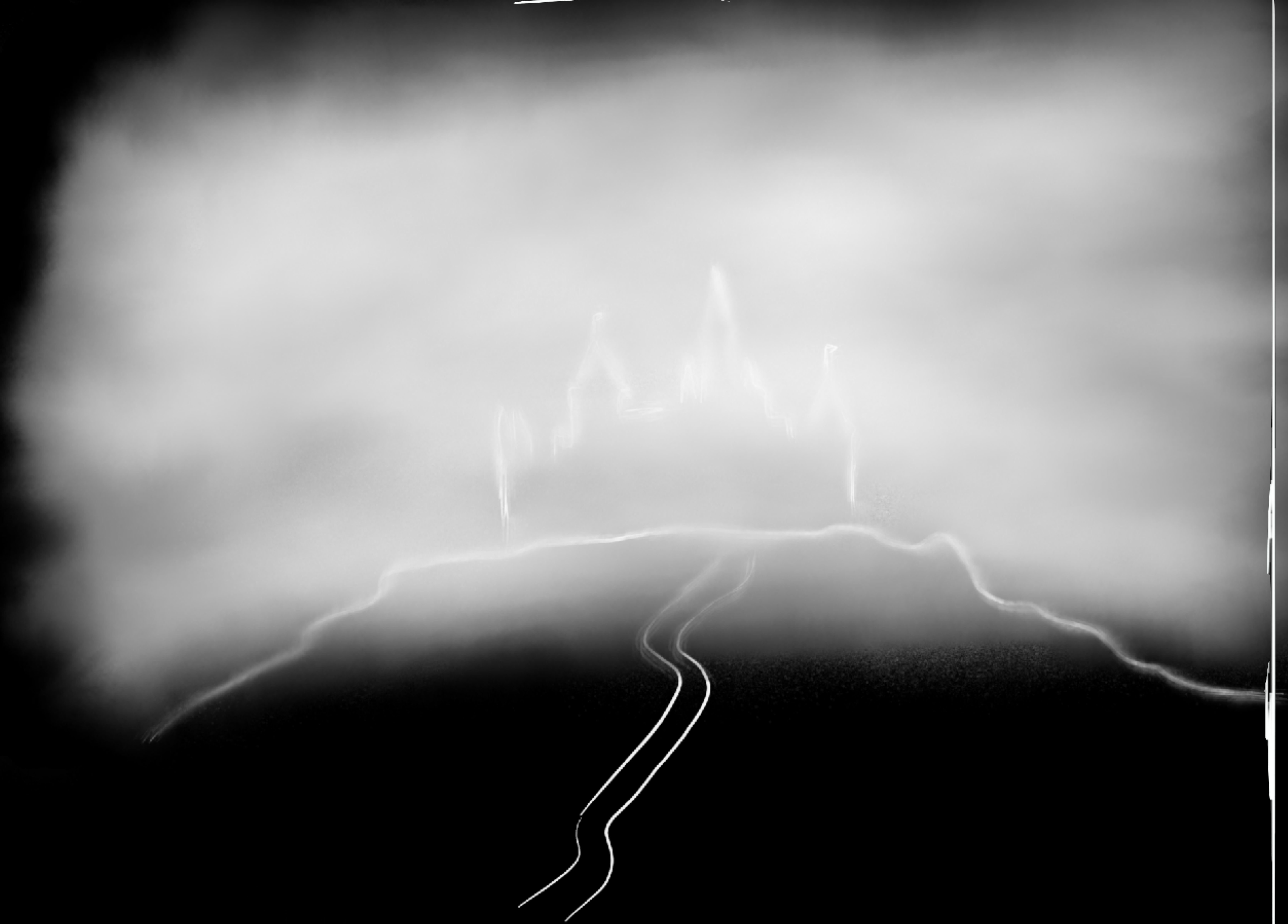
is this thing on?

http://sli.do

#L750

doom!

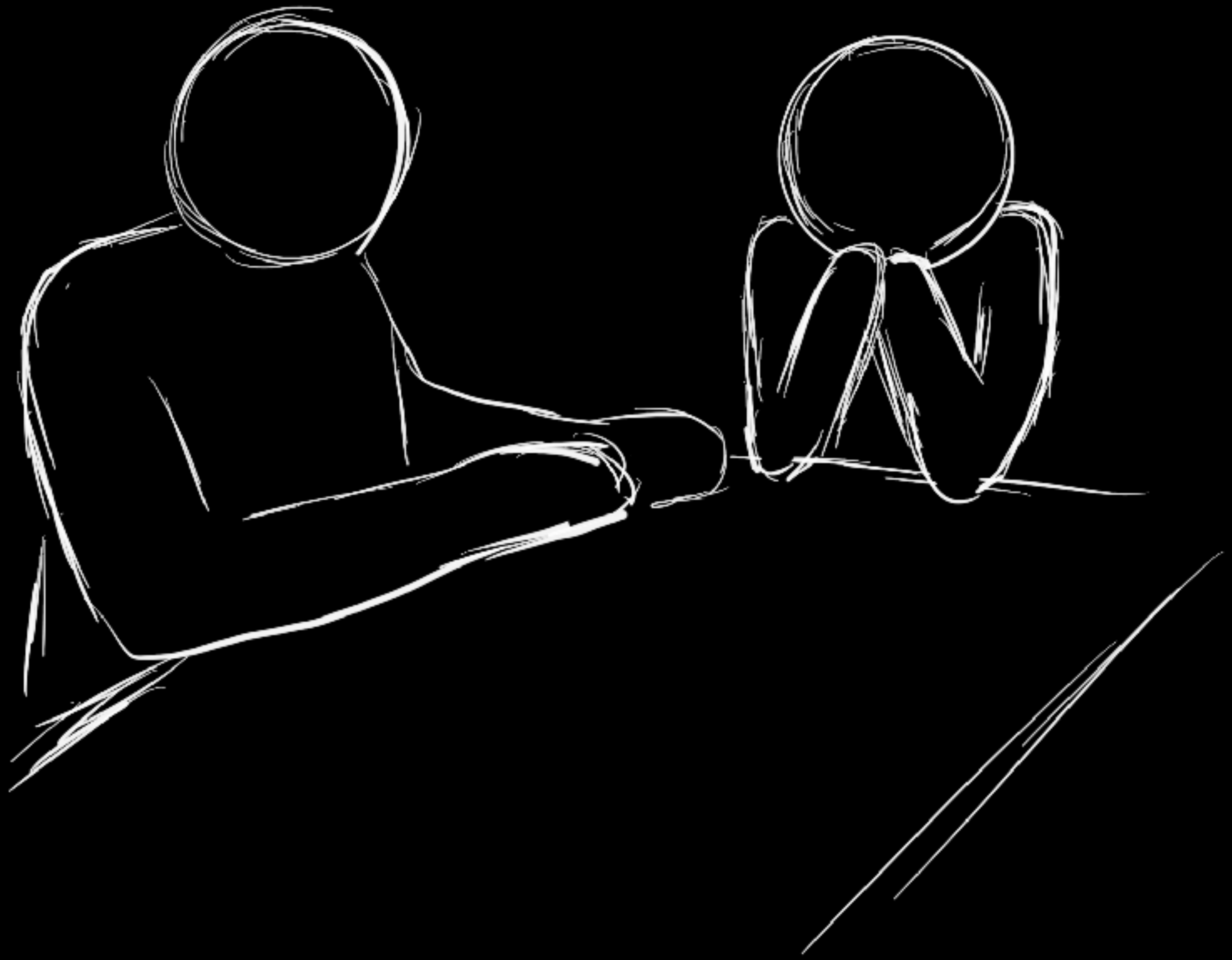
THE MURKY GOAL



what **problem** are
we trying to solve?

doom!

MICROSERVICES ENVY



we need to microservices

microservices
are not the **goal**

microservices

are not the **goal**

they are the **means**

“we’re going too slowly.

we need to get rid of COBOL and make microservices!”

“we’re going too slowly.

we need to get rid of COBOL and make microservices!”

“... but our release board only meets twice a year.”

distributed monolith

distributed monolith

but without compile-time checking

... or guaranteed function execution

reasons **not** to do microservices

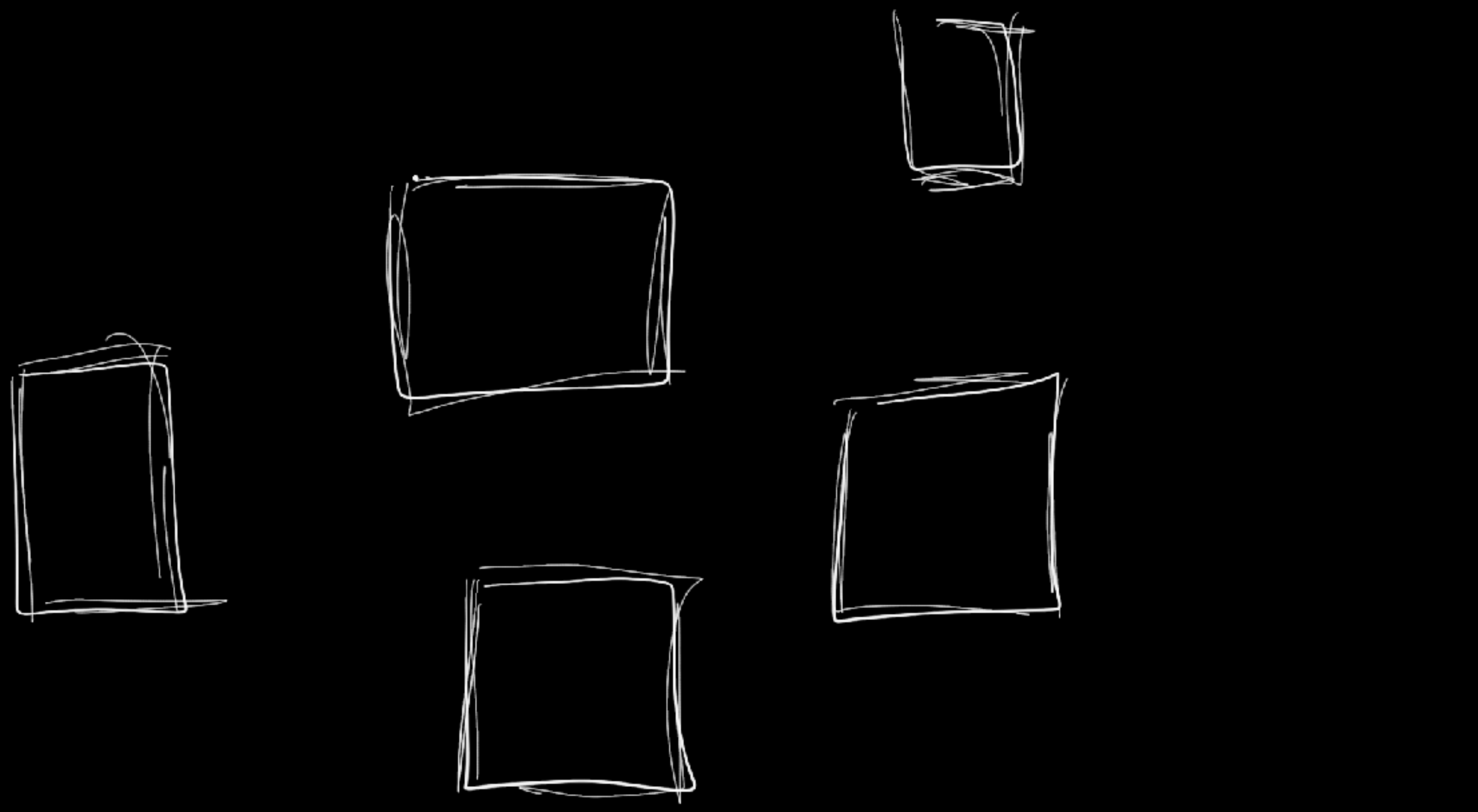
- small team
- not planning to release independently
- don’t want complexity of a service mesh - or worse yet, rolling your own
- domain model doesn’t split nicely

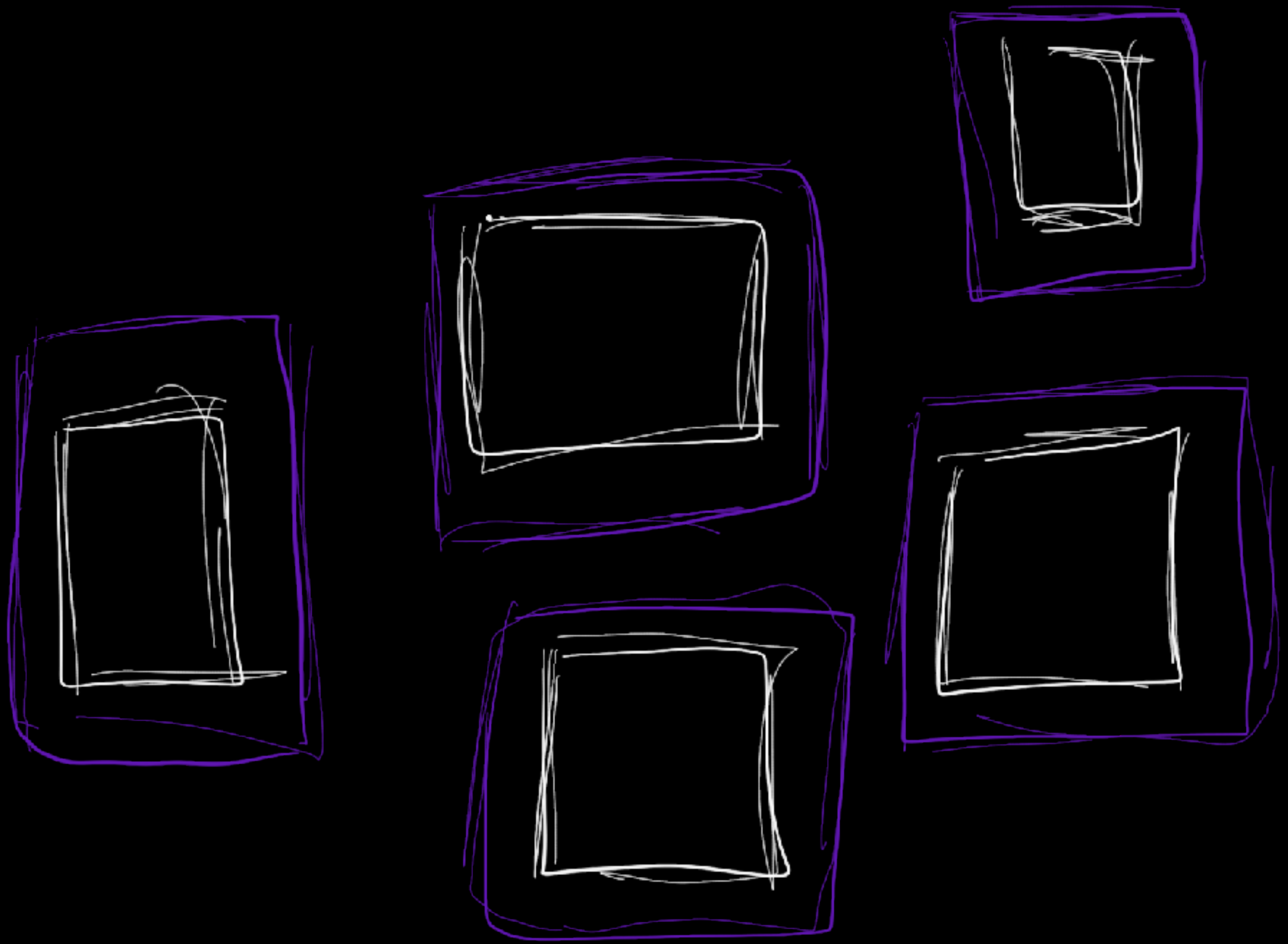
doom!

CLOUD-NATIVE SPAGHETTI

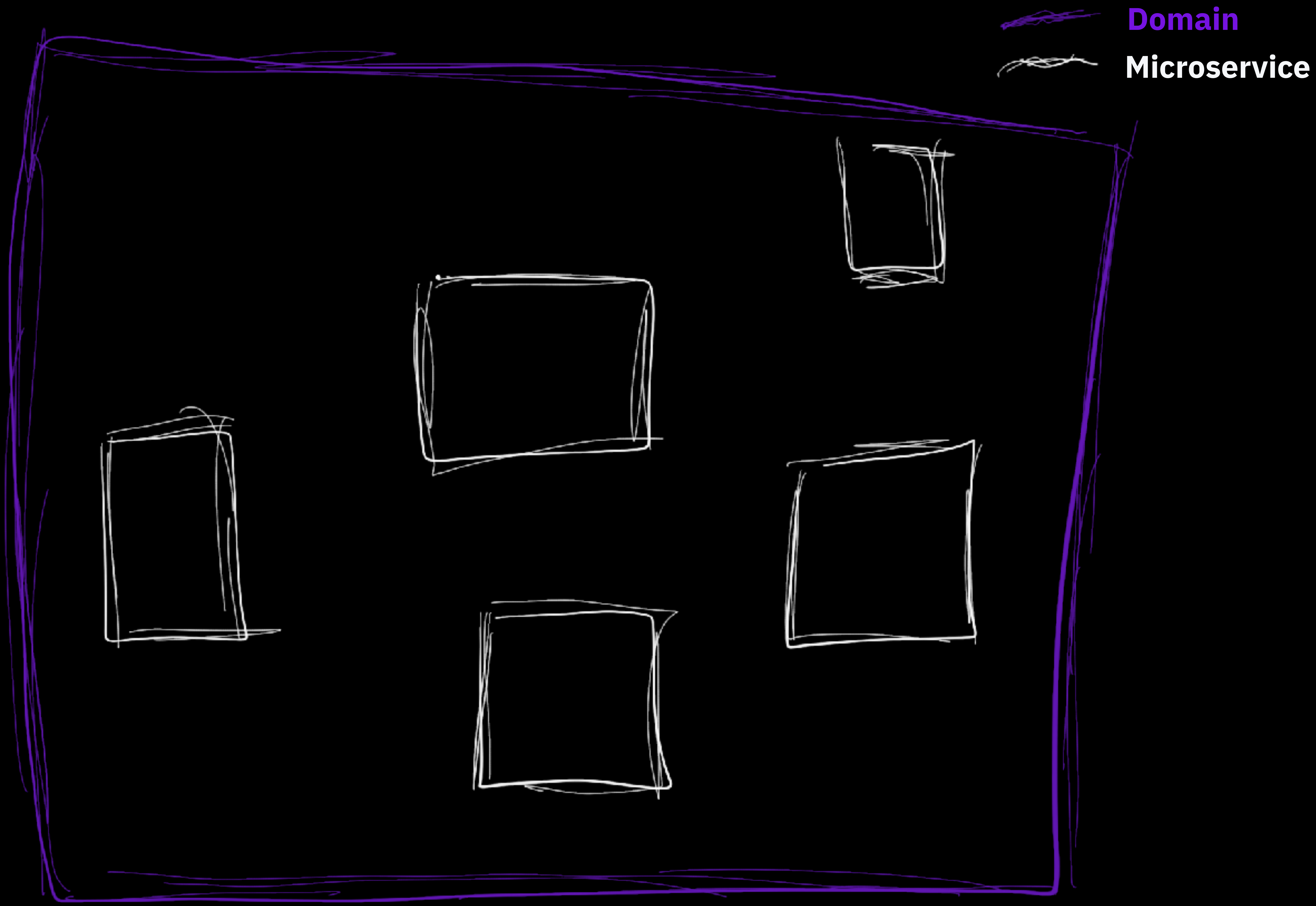
“each of our microservices has duplicated the same object model … with twenty classes and seventy fields”

“every time we touch
one microservice,
the others break”


Microservice



Domain
Microservice


Domain
Microservice

distributed != decoupled

doom!

MICROSERVICES OPS MAYHEM

doom!

MICROSERVICES OPS MAYHEM

do you know how to operate these things?

(there are quite a few of them)

do you know how to operate these things?

(there are quite a few of them)

observability

observability

SRE

SRE

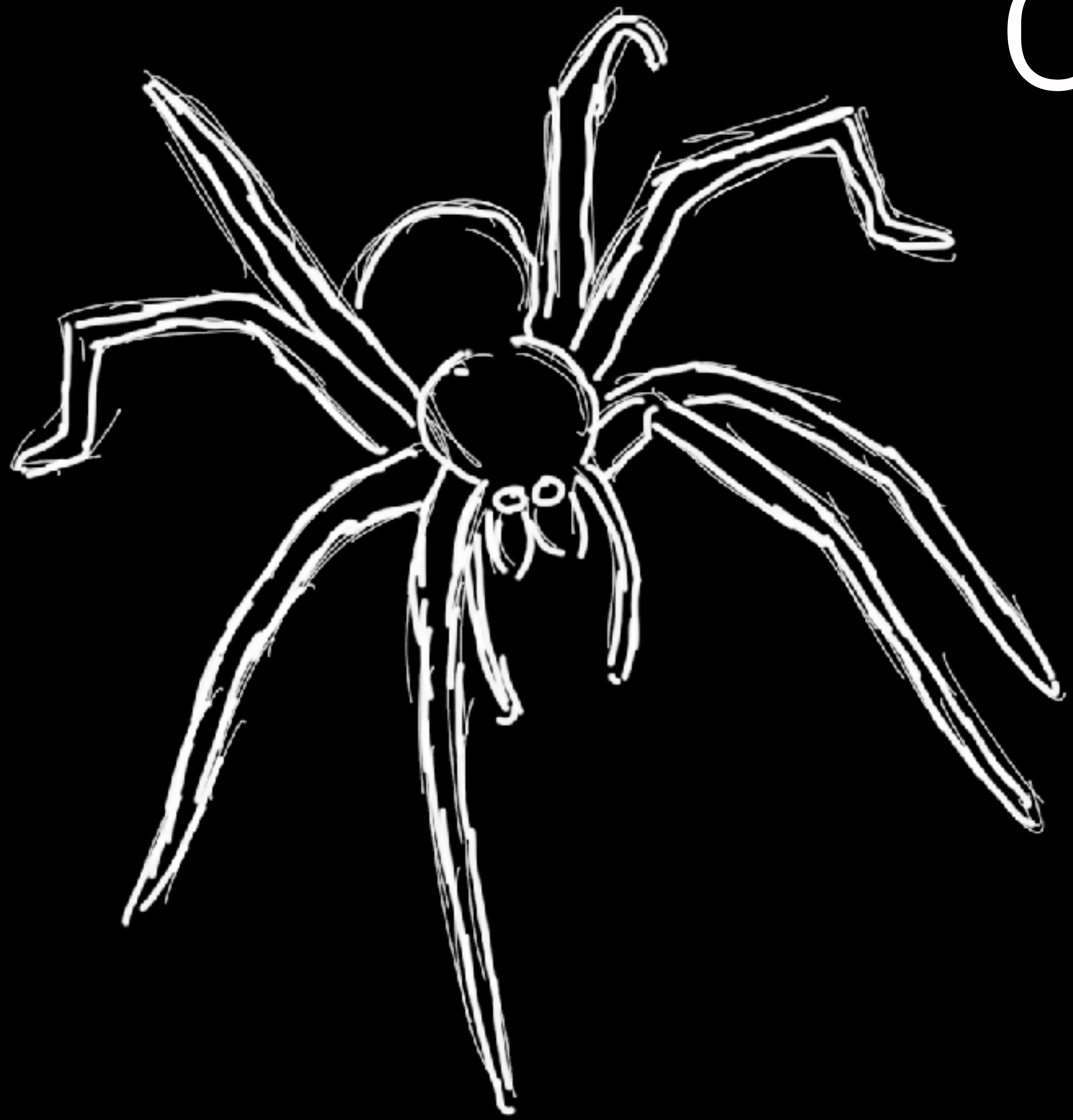
doom!

THE 'SOMEDAY' AUTOMATION

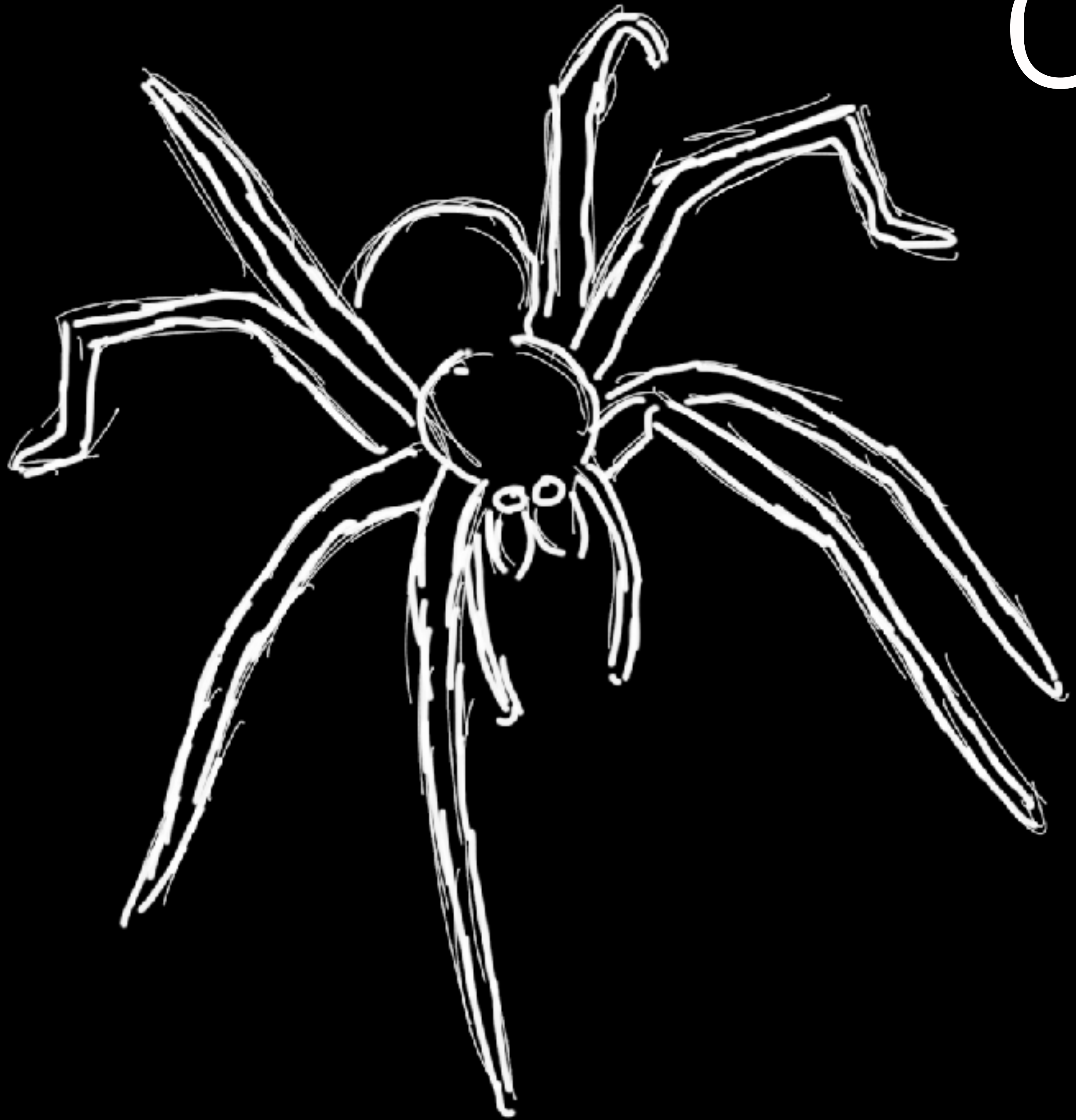
“our ops isn’t automated”

“our tests aren’t automated”

“we don’t know if our code works”



“we don’t know if our code works”

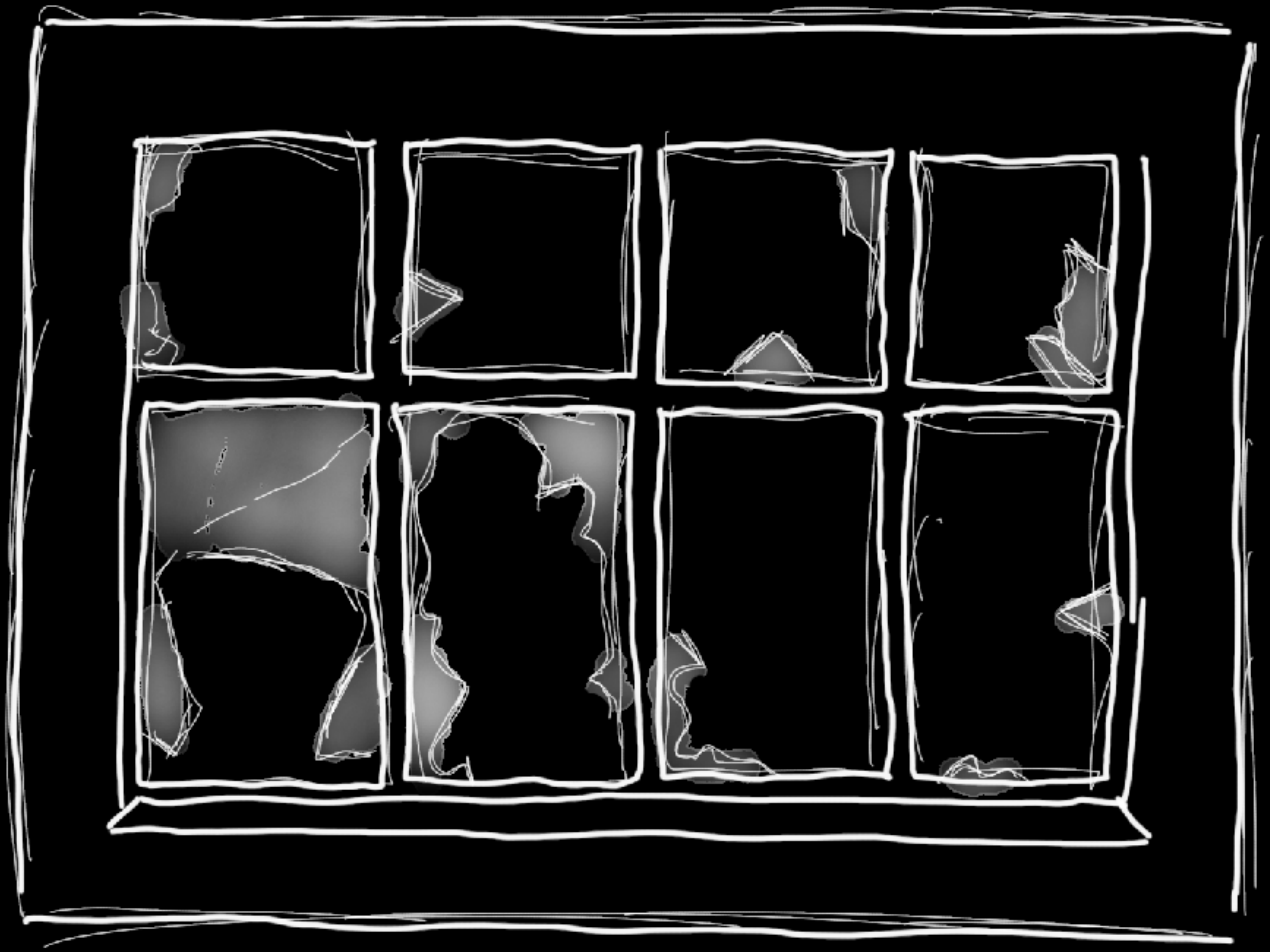


@holly_cummins

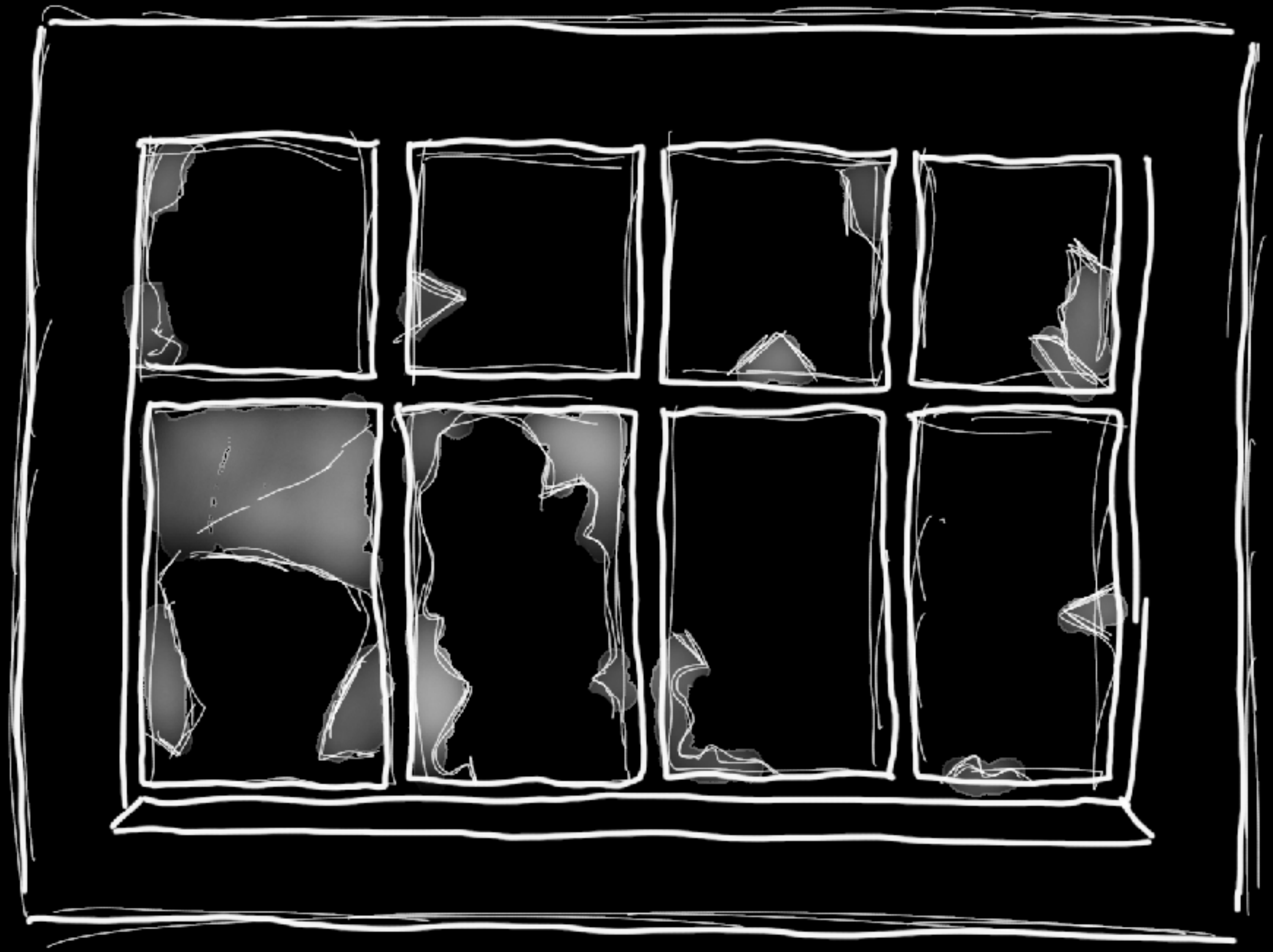
microservices **need**
automated integration tests

microservices **need** automated contract tests

the rotting automation



@holly_cummins



“oh yes, that build has been broken for a few weeks…”

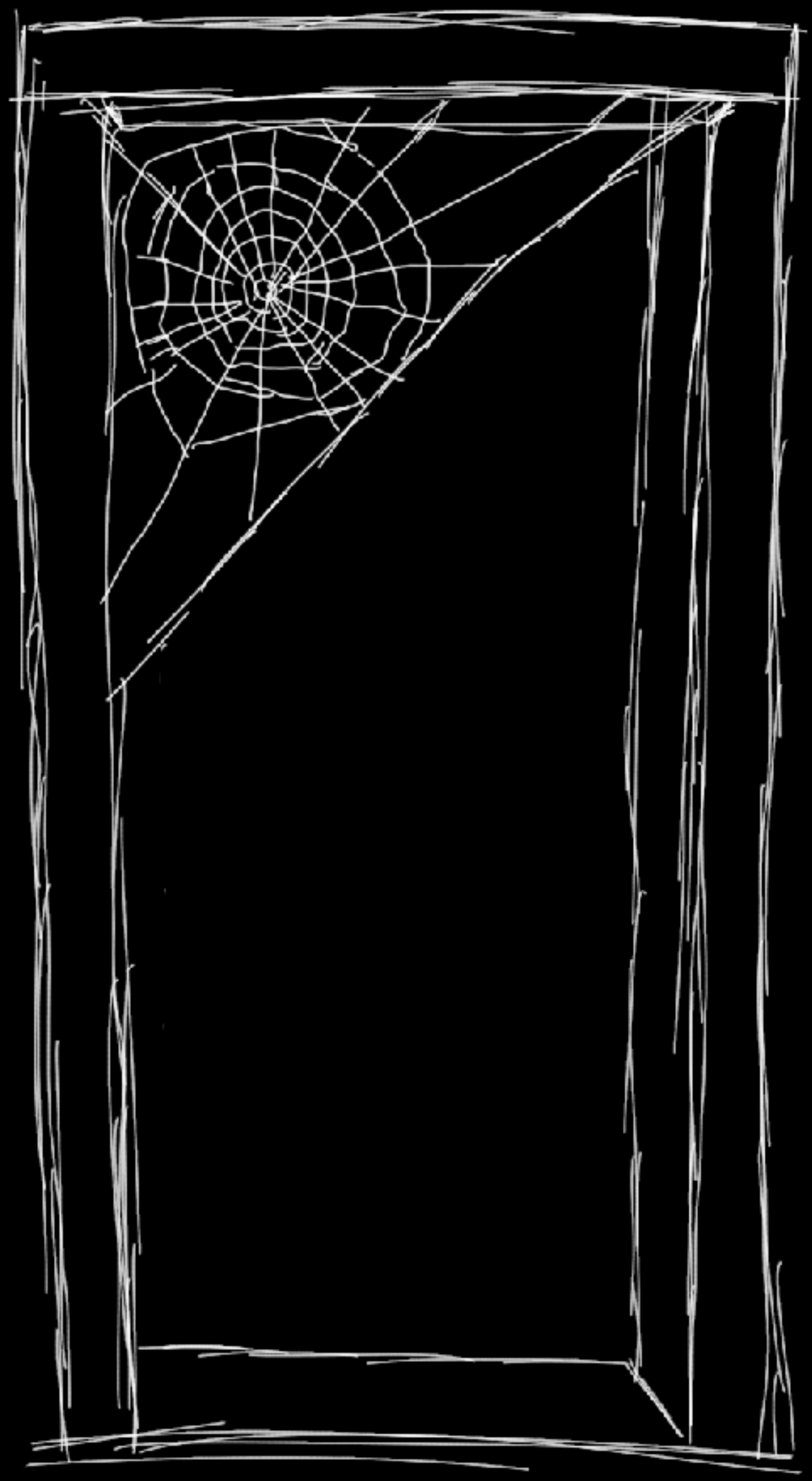
“we don’t know when the build is broken” let’s talk about your build

http://sli.do

#L750

doom!

THE NOT-ACTUALLY-CONTINUOUS CONTINUOUS INTEGRATION AND CONTINUOUS DEPLOYMENT



“we have a CI/CD”

“we have a CI/CD”

Here lies
code
on a laptop
FOR SOMETIME

CI/CD is something you **do**
not a tool you buy

“i’ll merge my branch into our CI next week”

"CI/CD … CI/CD … CI/CD …

we release every six months …

CI/CD …. "

what **is** CD?

http://sli.do

#L750

continuous.

I don’t think that word means what you think it means.

how do **you** do continuous?

http://sli.do

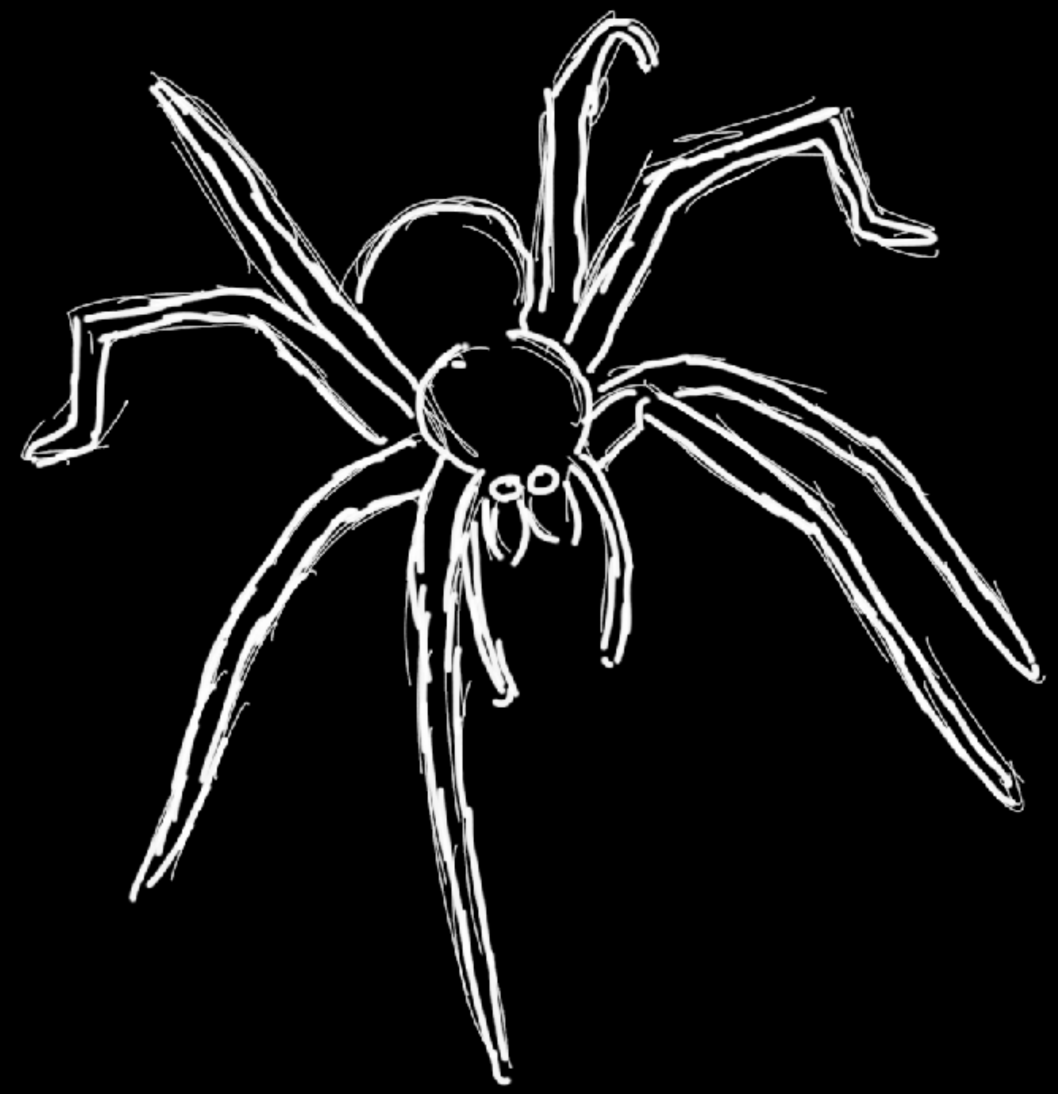
#L750

how do **you** CD?

http://sli.do

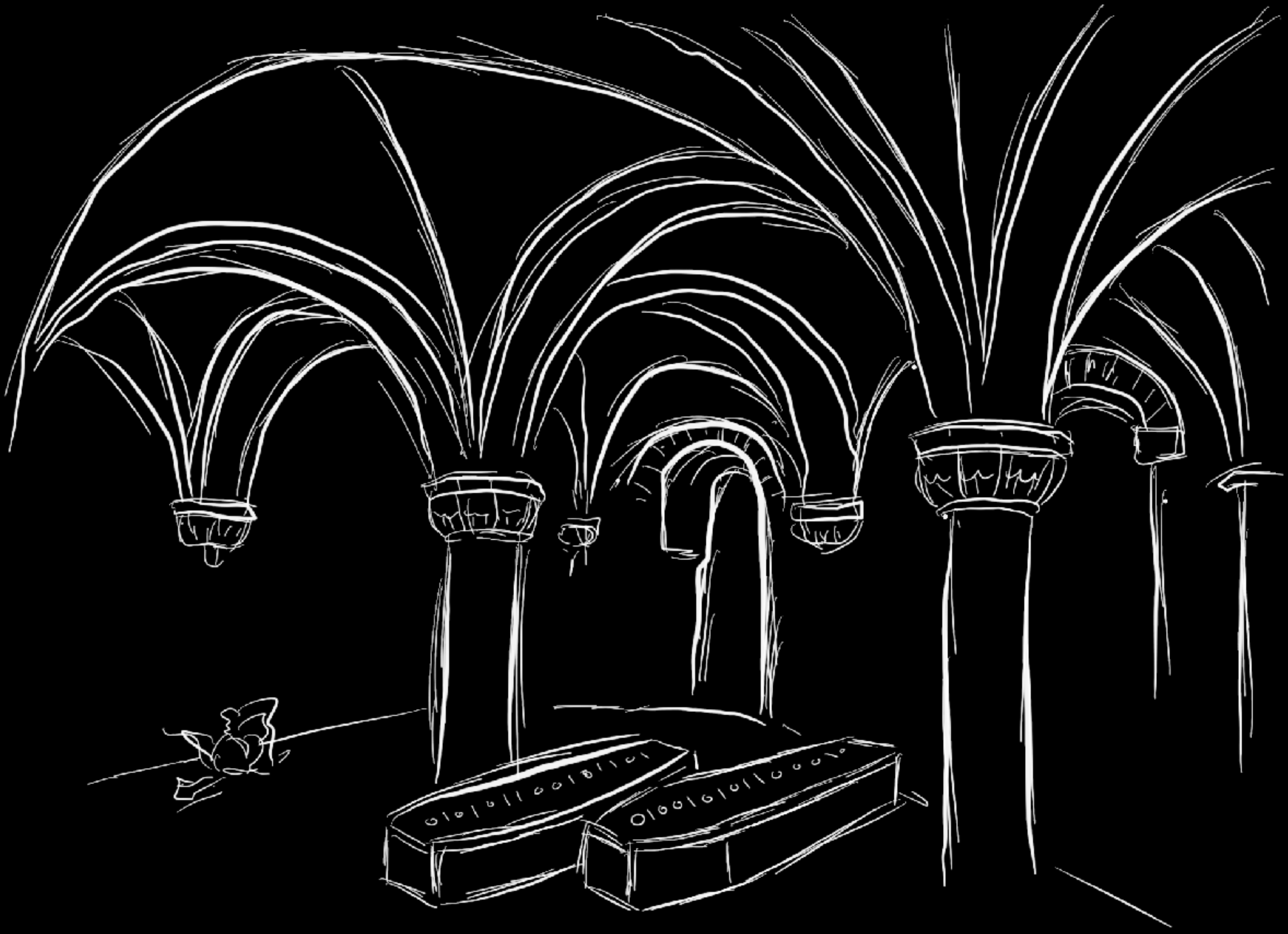
#L750

“we can’t ship until we have more confidence in the quality”

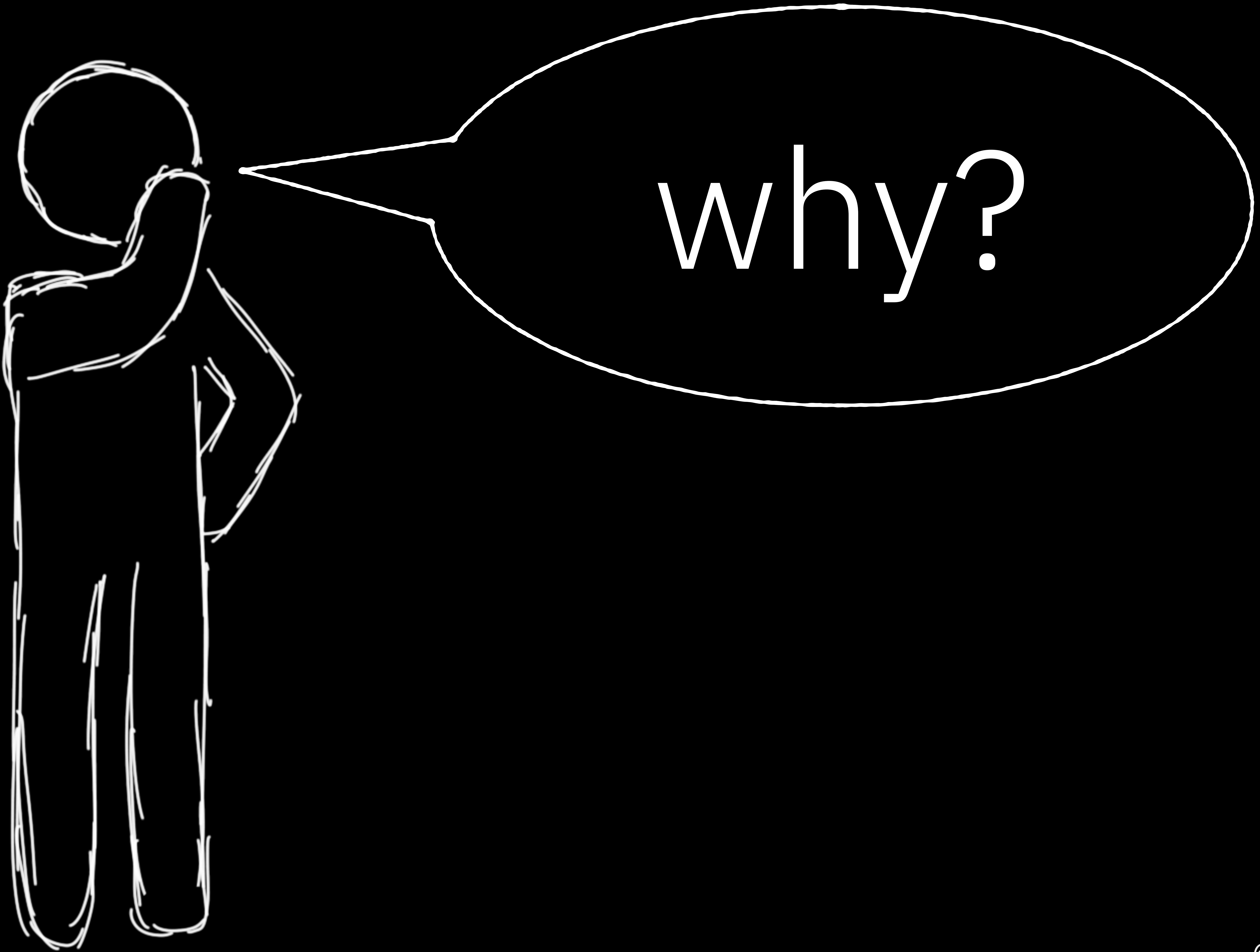


doom!

THE SOFTWARE CRYPT



“we can’t actually **release** this.”


why?

“we’ve scheduled the architecture board review for a month after the project is ready to ship” “we can’t release this microservice…

we deploy all our microservices at the same time.”

why?

oh yes, we don’t know if they work

“we can’t ship until every feature is complete” what’s the point of architecture that can go faster, if you don’t go faster?

feedback is good engineering

deferred wiring

feature flags

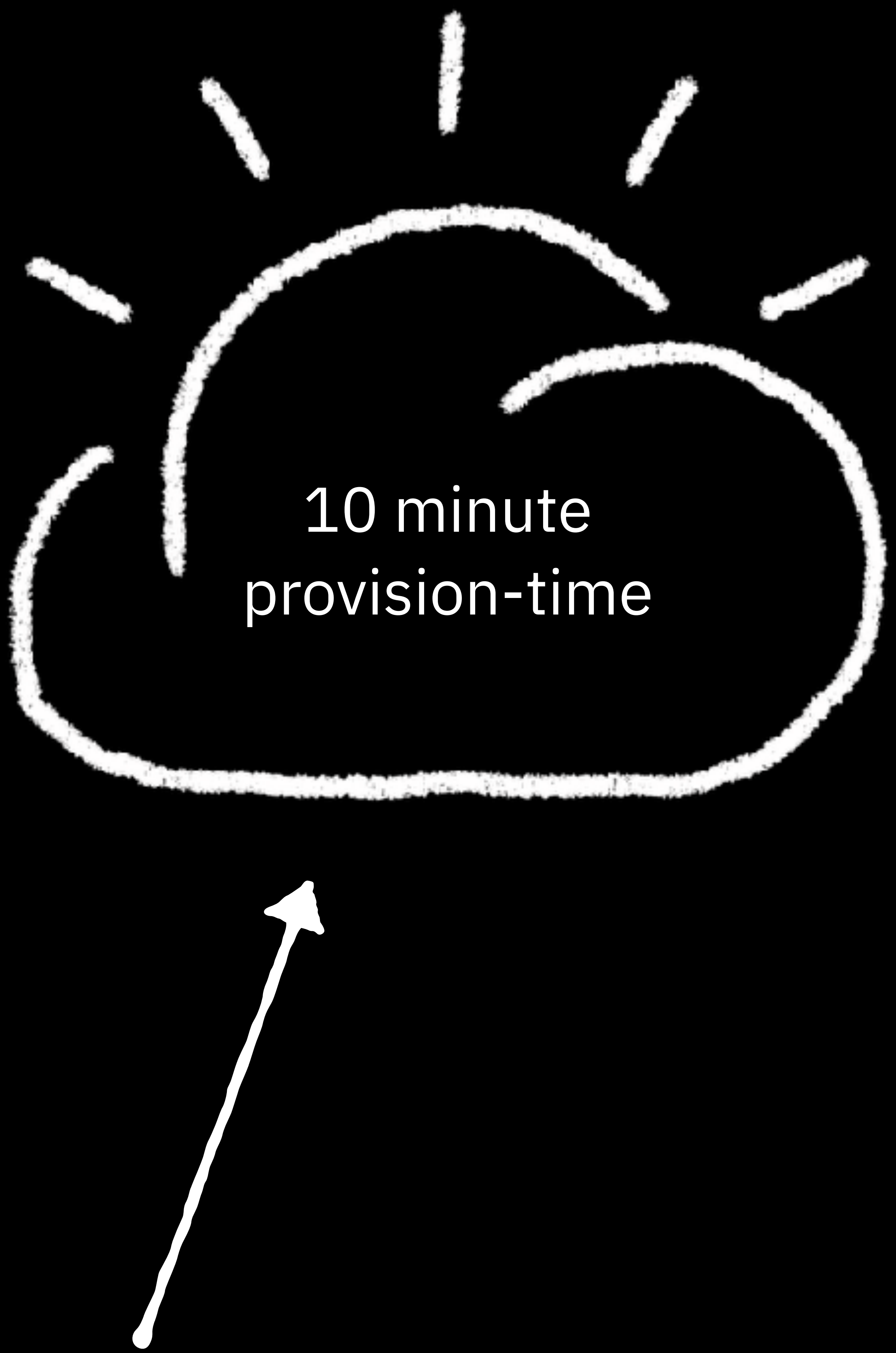
a/b testing

canary deploys

doom!

THE LOCKED-DOWN TOTALLY RIGID INFLEXIBLE UN-CLOUDY CLOUD

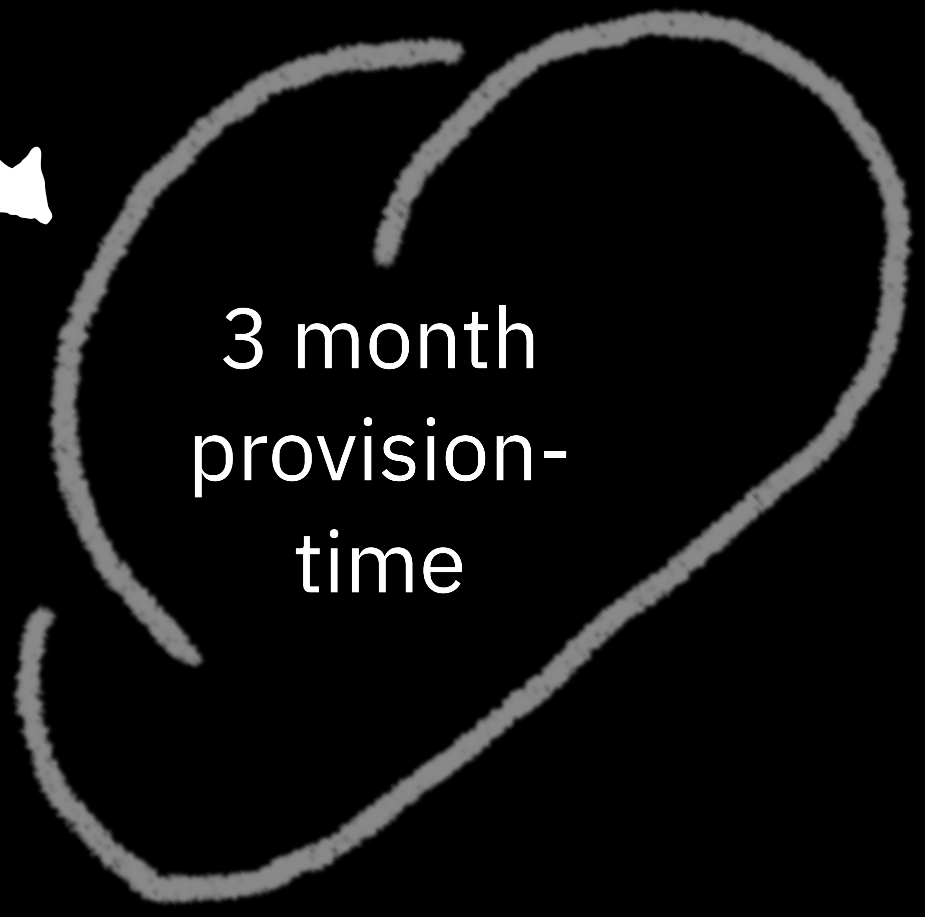
“this provisioning software is broken”

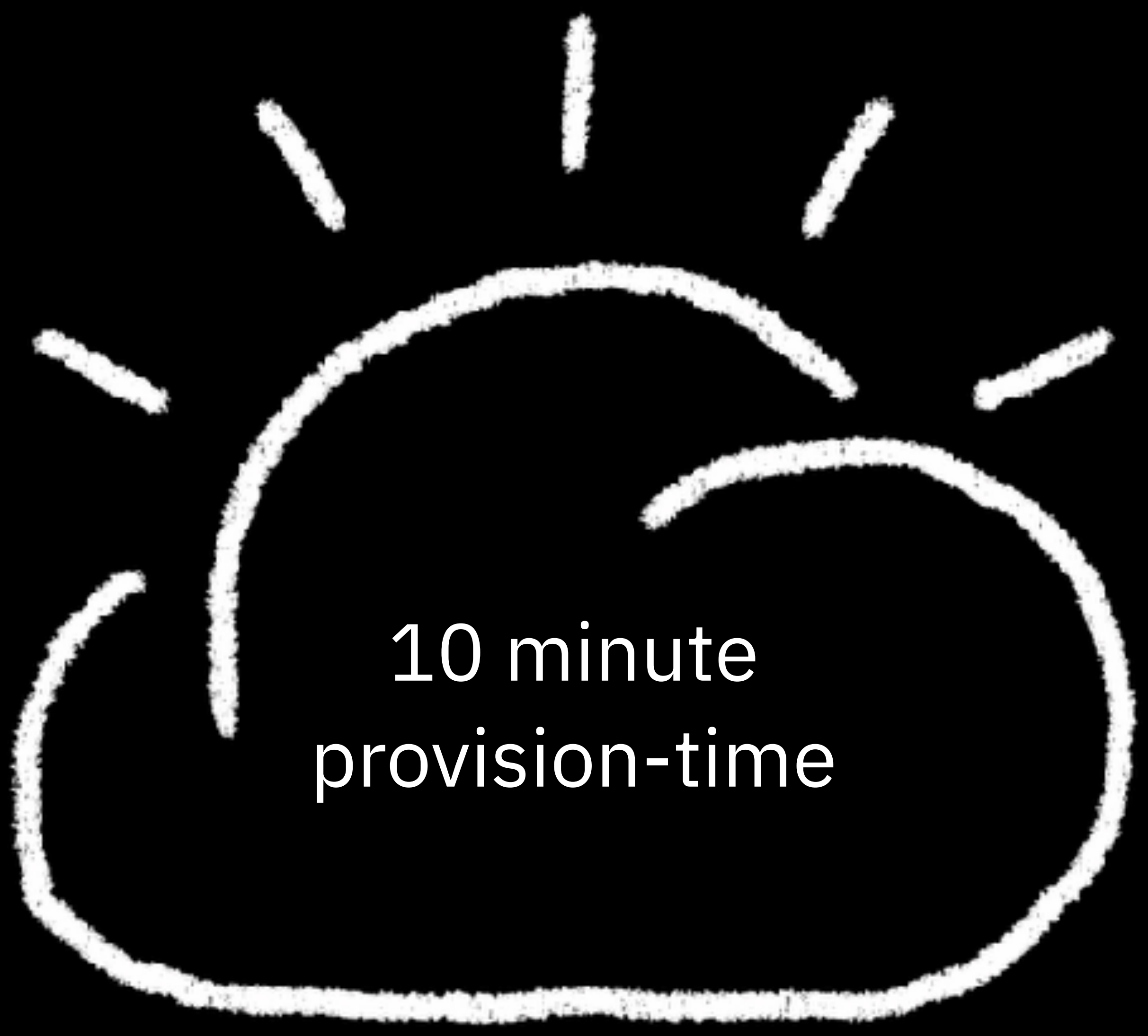


what we sold

“this provisioning software is broken”

what the client thought they'd got



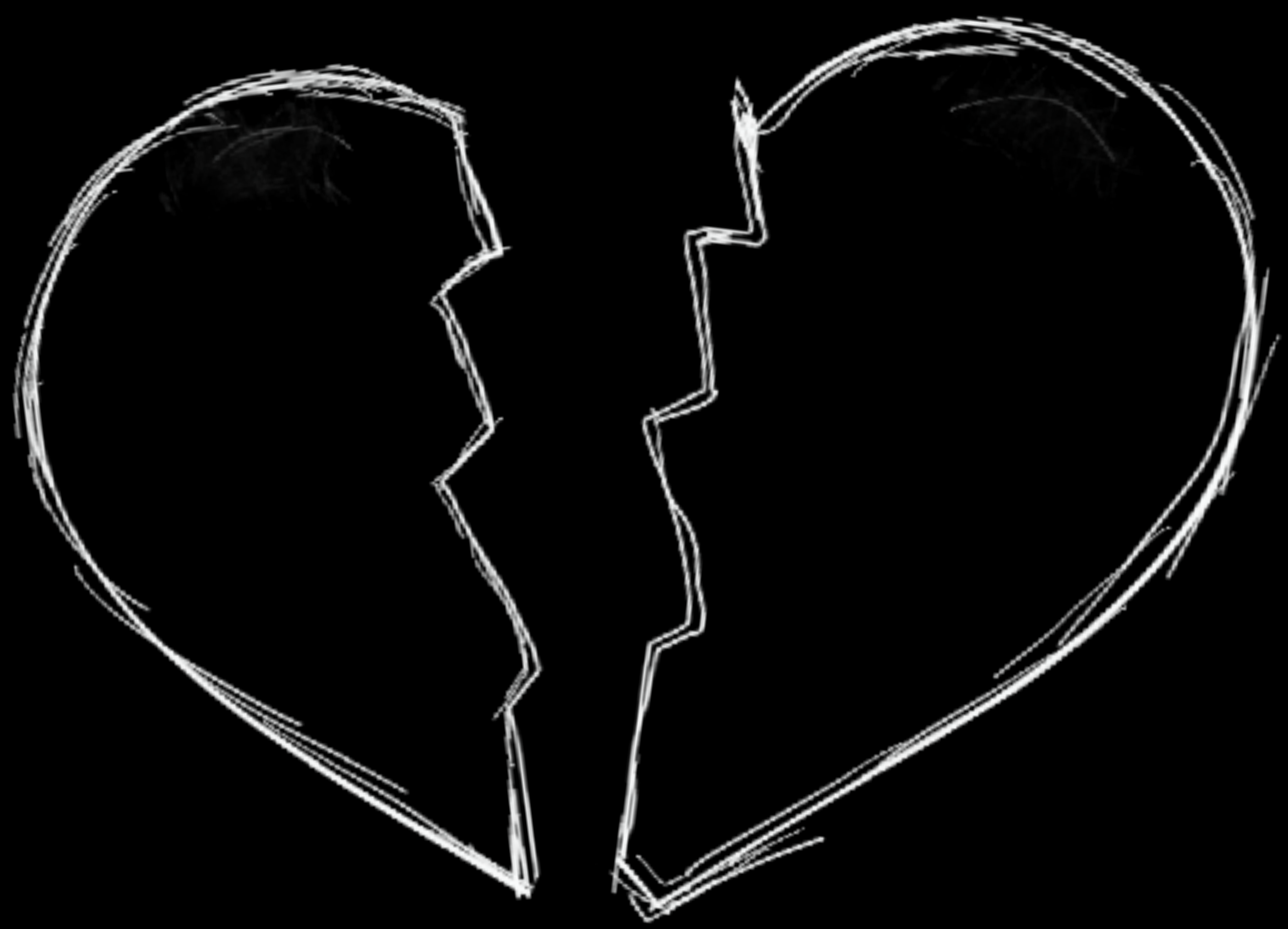


what we sold

“this provisioning software is broken”

what the client thought they'd got
3 month provision-time
the reason
10 minute provision-time
84-step pre-approval process
what we sold

"this provisioning software is broken"



governance

Provider A

Provider B

“we’re going to change cloud provider

Provider A

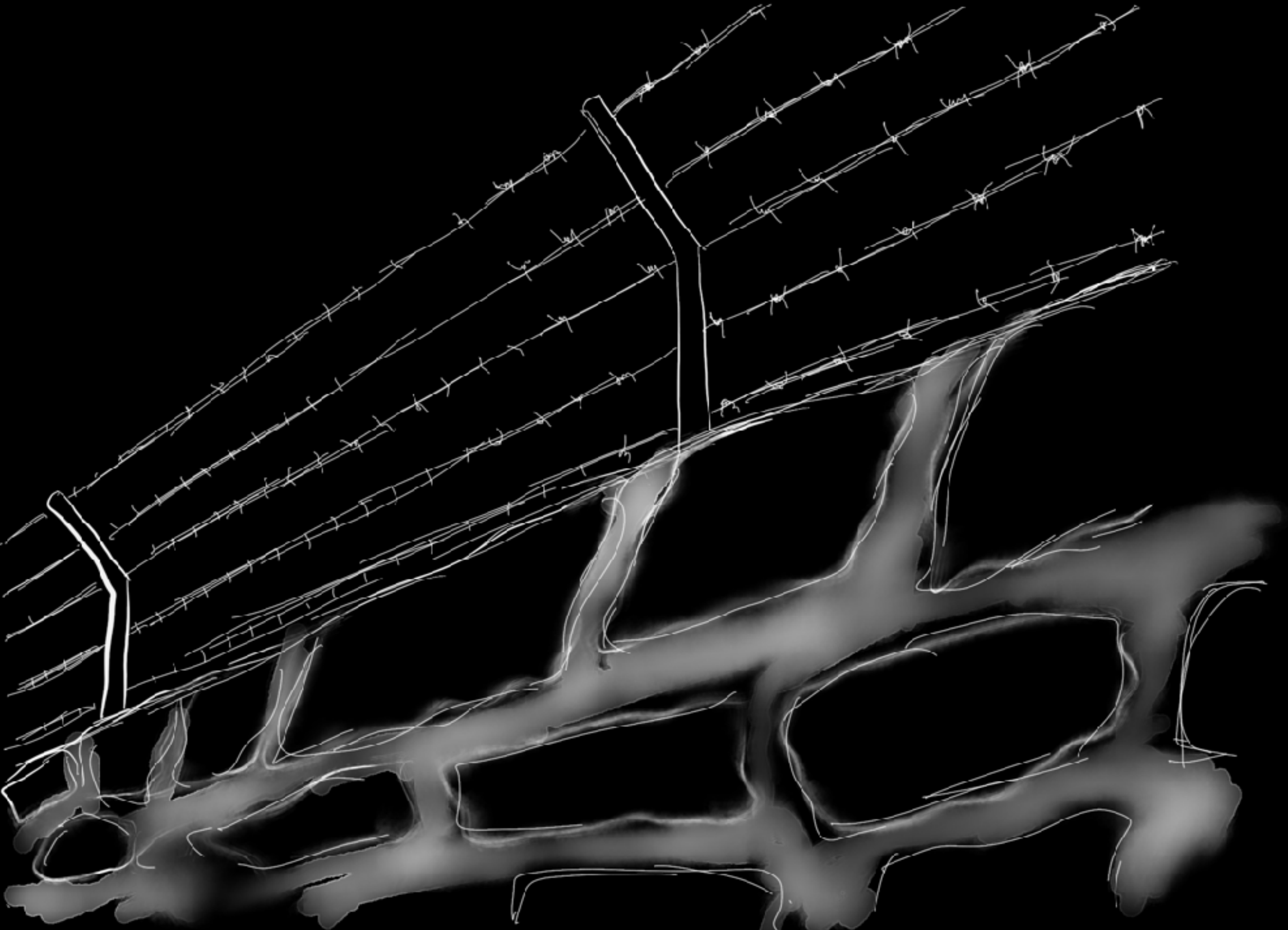
Provider B

“we’re going to change cloud provider to fix our procurement process!”

Provider A

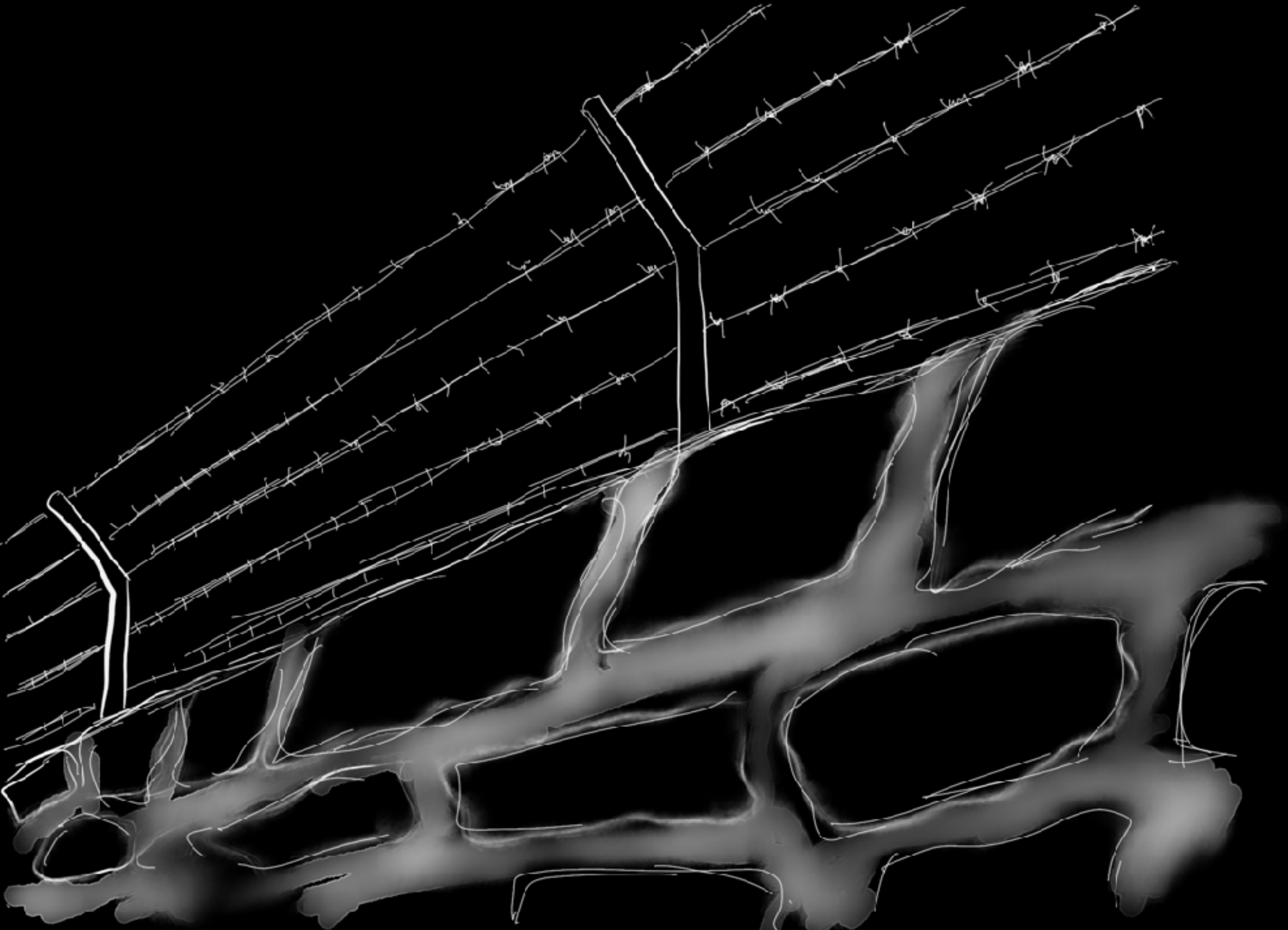
Provider B

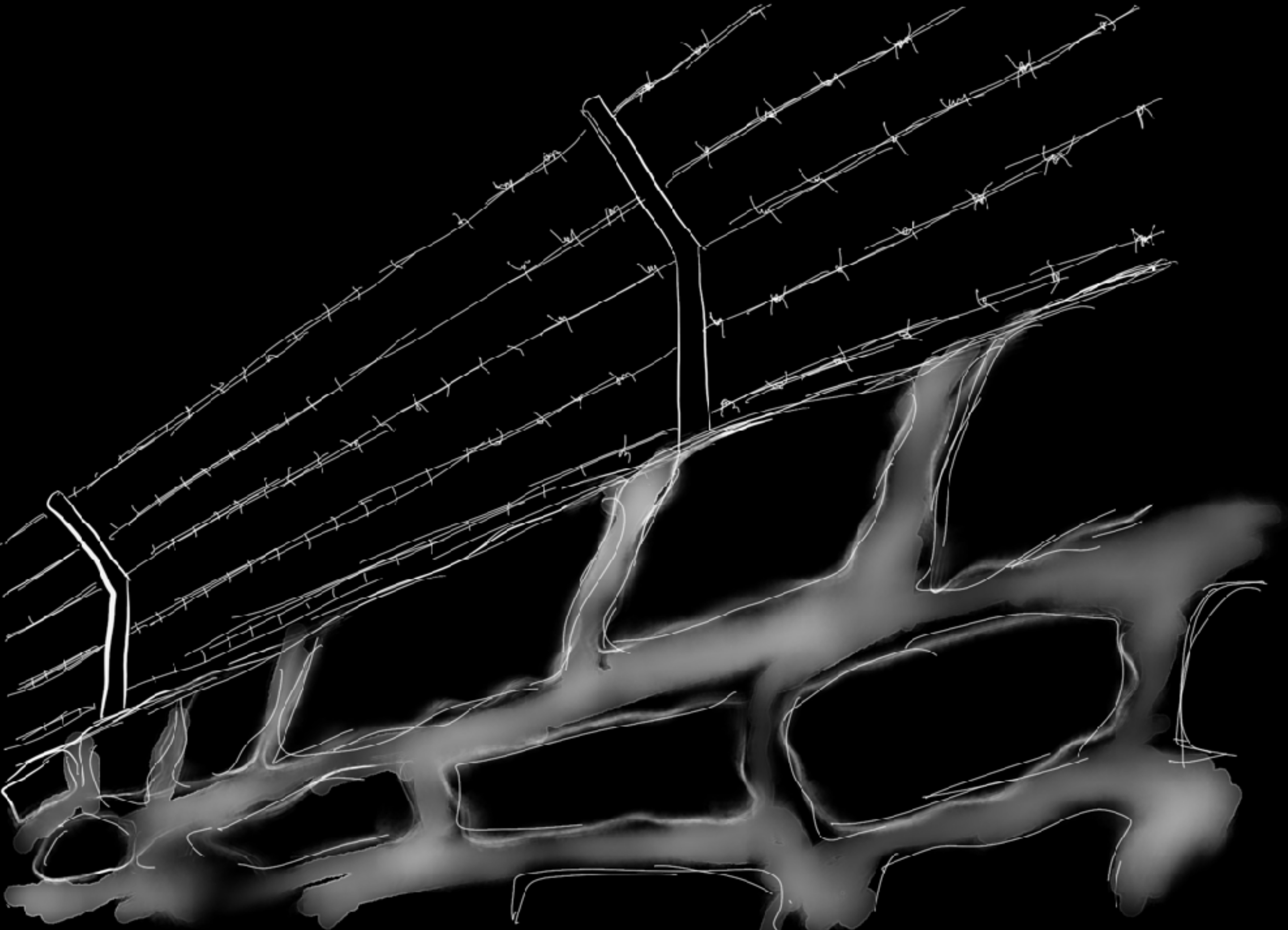
“we’re going to change cloud provider to fix our procurement process!”



@holly_cummins

“we’ve configured our network!

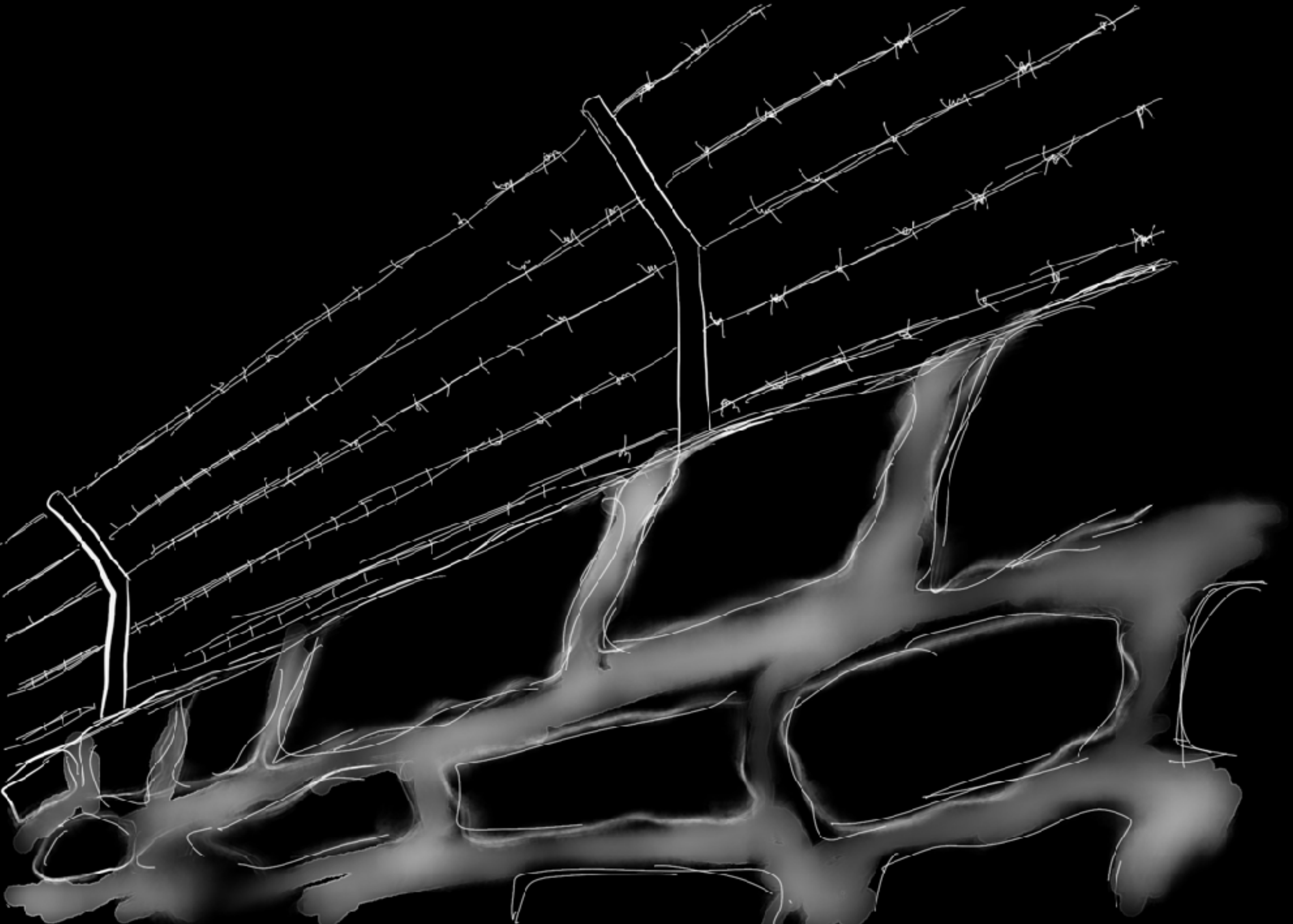




“we’ve configured our network!

you can either access the cloud servers … or access jira.

@holly_cummins



"we've configured our network!

you can either access the cloud servers … or access jira.

to access both you'd need two machines."

@holly_cummins

“it takes us a week
to start coding.”

“it takes us a week to start coding.”

“two days to get a repo …

two days to get a pipeline …”

there is a **cost**:

developers flee

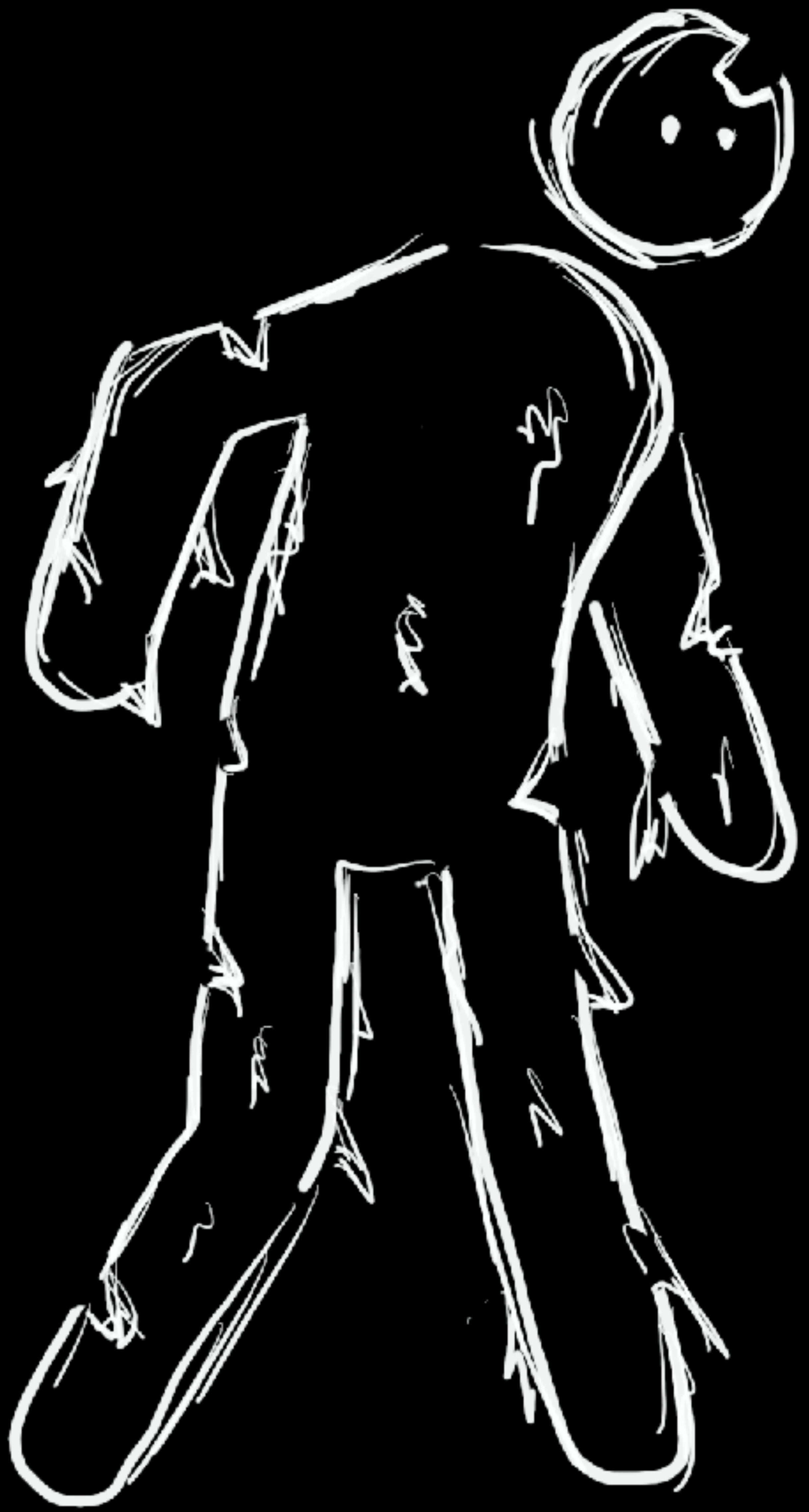
doom!

THE MYSTERY MONEY PIT

the cloud makes it so **easy** to provision hardware.

that doesn’t mean the hardware is free.

or useful.



zombie workload

2017 survey

25%

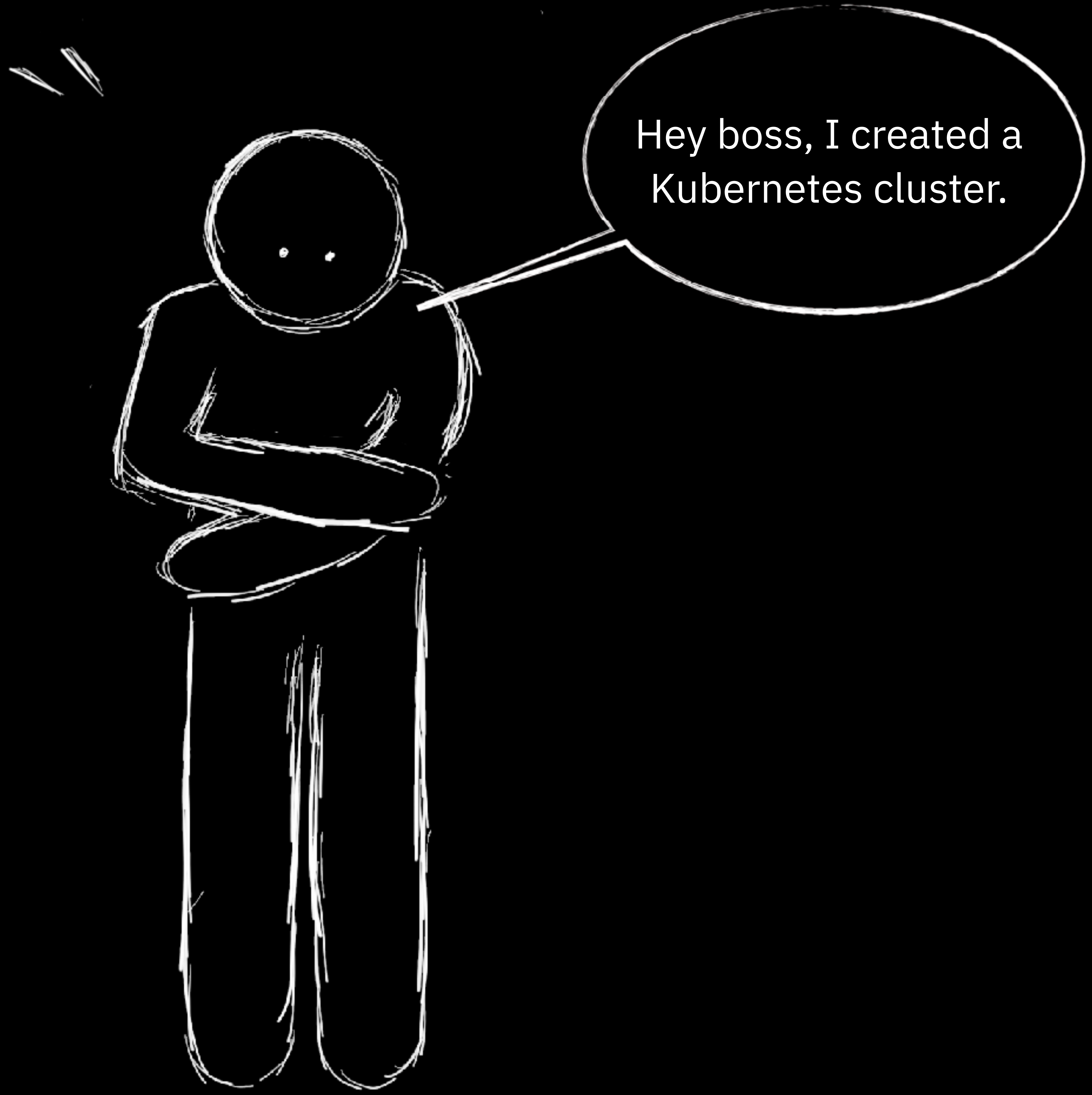
of 16,000 servers
doing no useful work

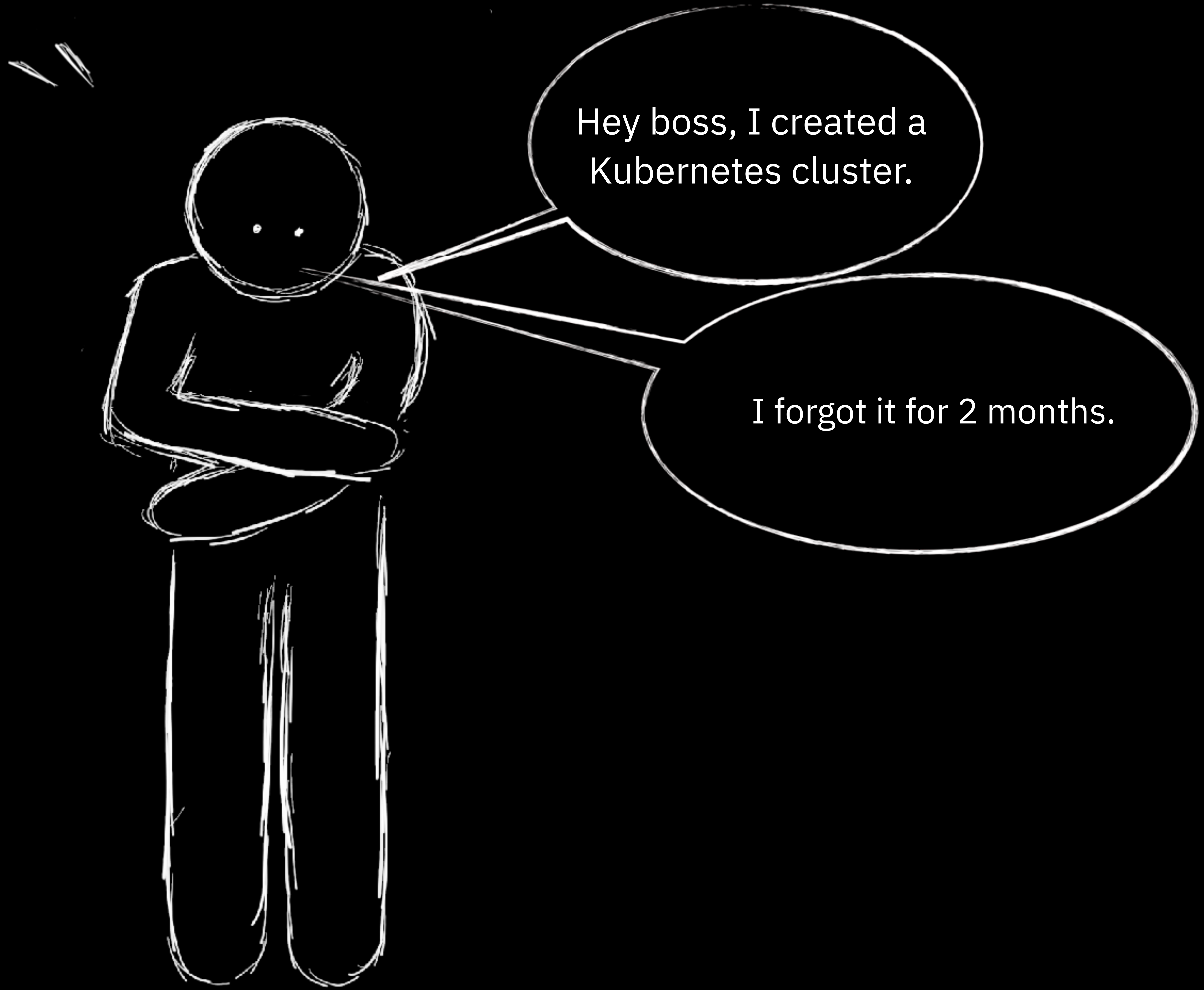
2017 survey

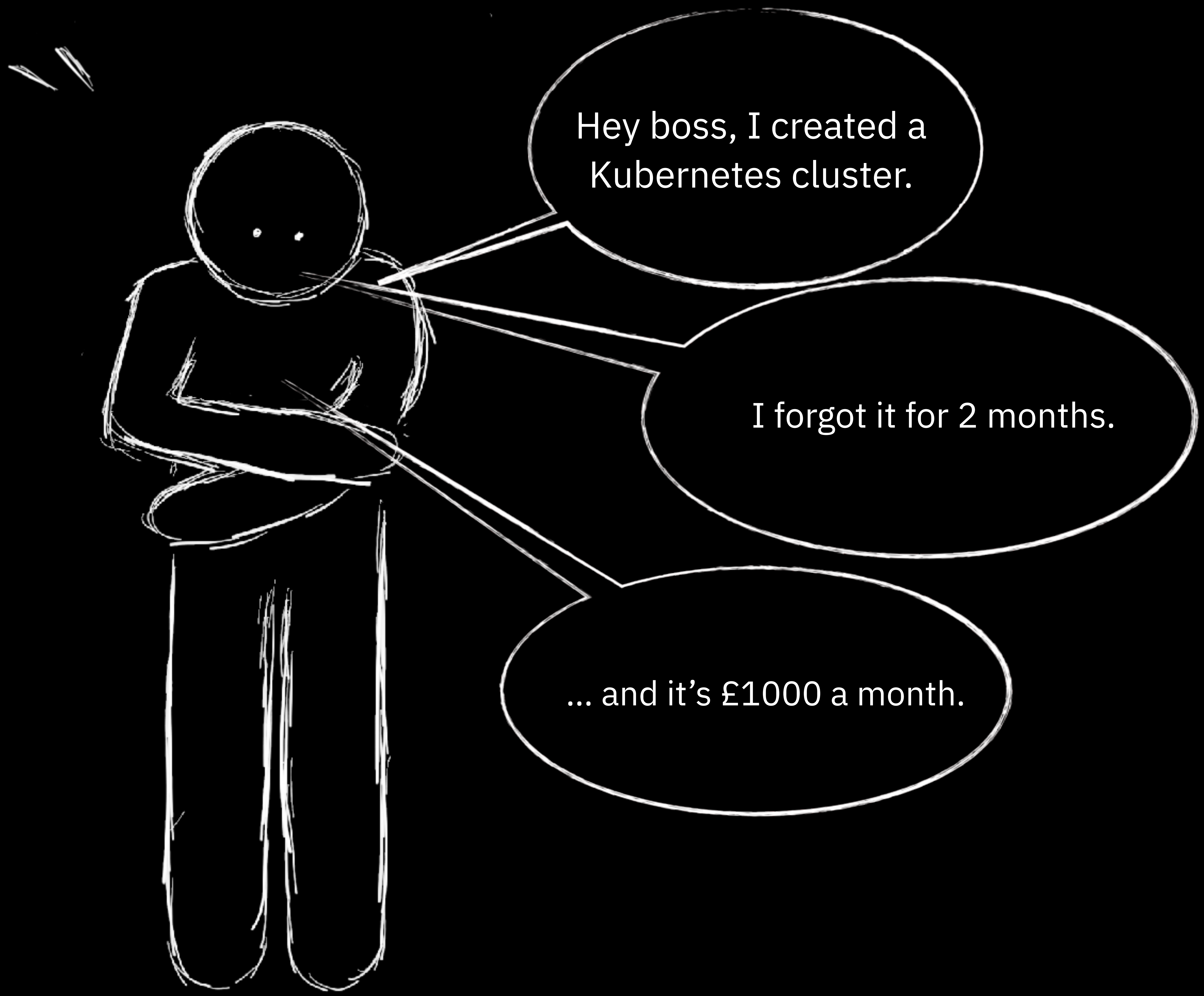
25%

of 16,000 servers doing no useful work

"perhaps someone forgot to turn them off"


Hey boss, I created a Kubernetes cluster.


Hey boss, I created a Kubernetes cluster.
I forgot it for 2 months.


Hey boss, I created a Kubernetes cluster.
I forgot it for 2 months.
... and it's £1000 a month.

“we have 28 cloud instances.

or maybe it’s 35.”

“we have **no idea** how much we’re spending on cloud.”

finops

multicloud management

@holly_cummins