CHEATING GALL'S LAW
HOW WE SPLIT A MONOLITH
AND LIVED TO TELL THE TALE

LAURIE VOSS

CTO, npm

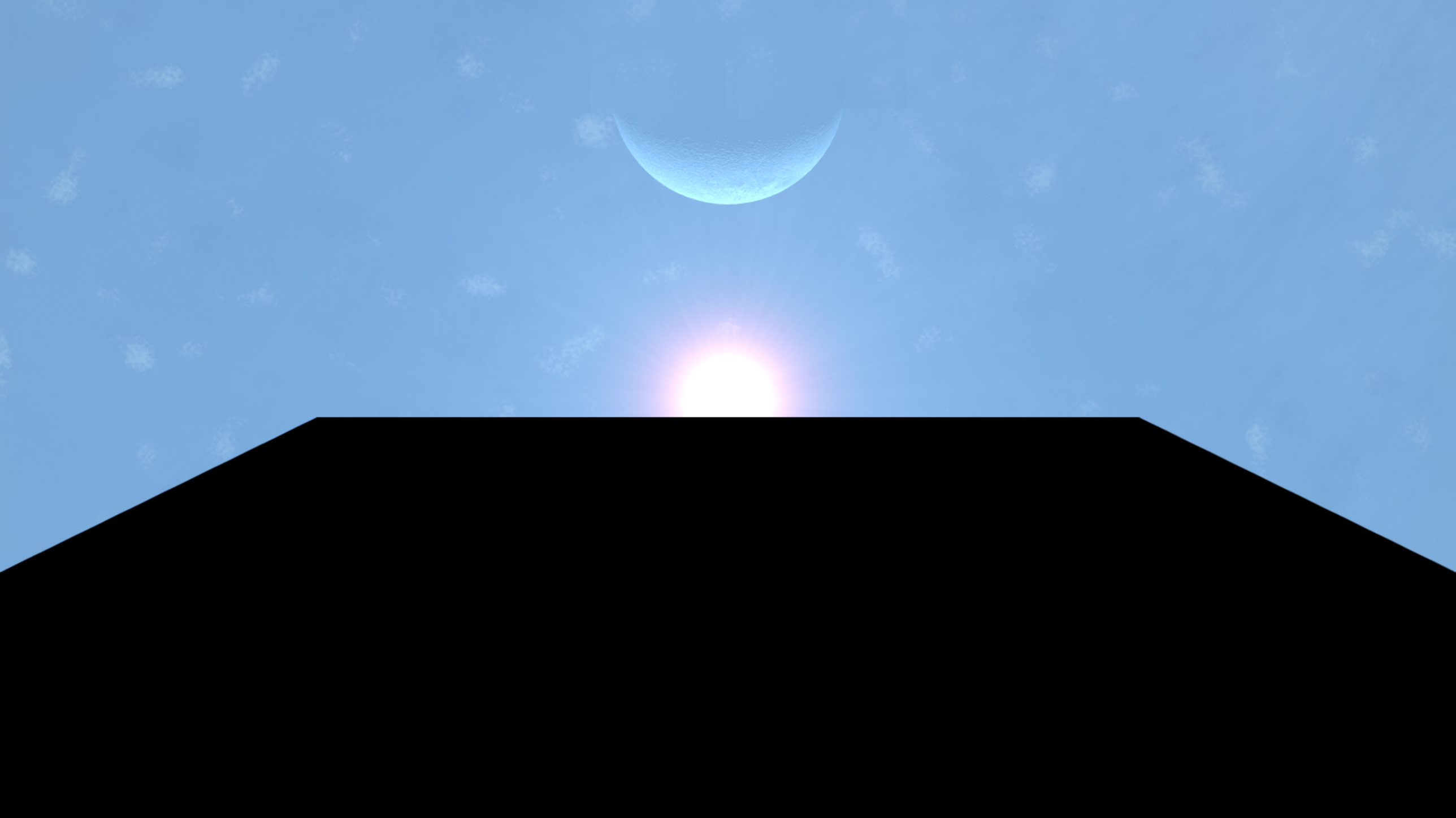
@SELDO

GALL'S LAW

A COMPLEX SYSTEM THAT WORKS IS INVARIABLY FOUND TO HAVE EVOLVED FROM A SIMPLE SYSTEM THAT WORKED.

SYSTEMANTICS:
HOW SYSTEMS REALLY WORK AND HOW THEY FAIL

STARTING SIMPLE OFTEN MEANS STARTING WITH A MONOLITH



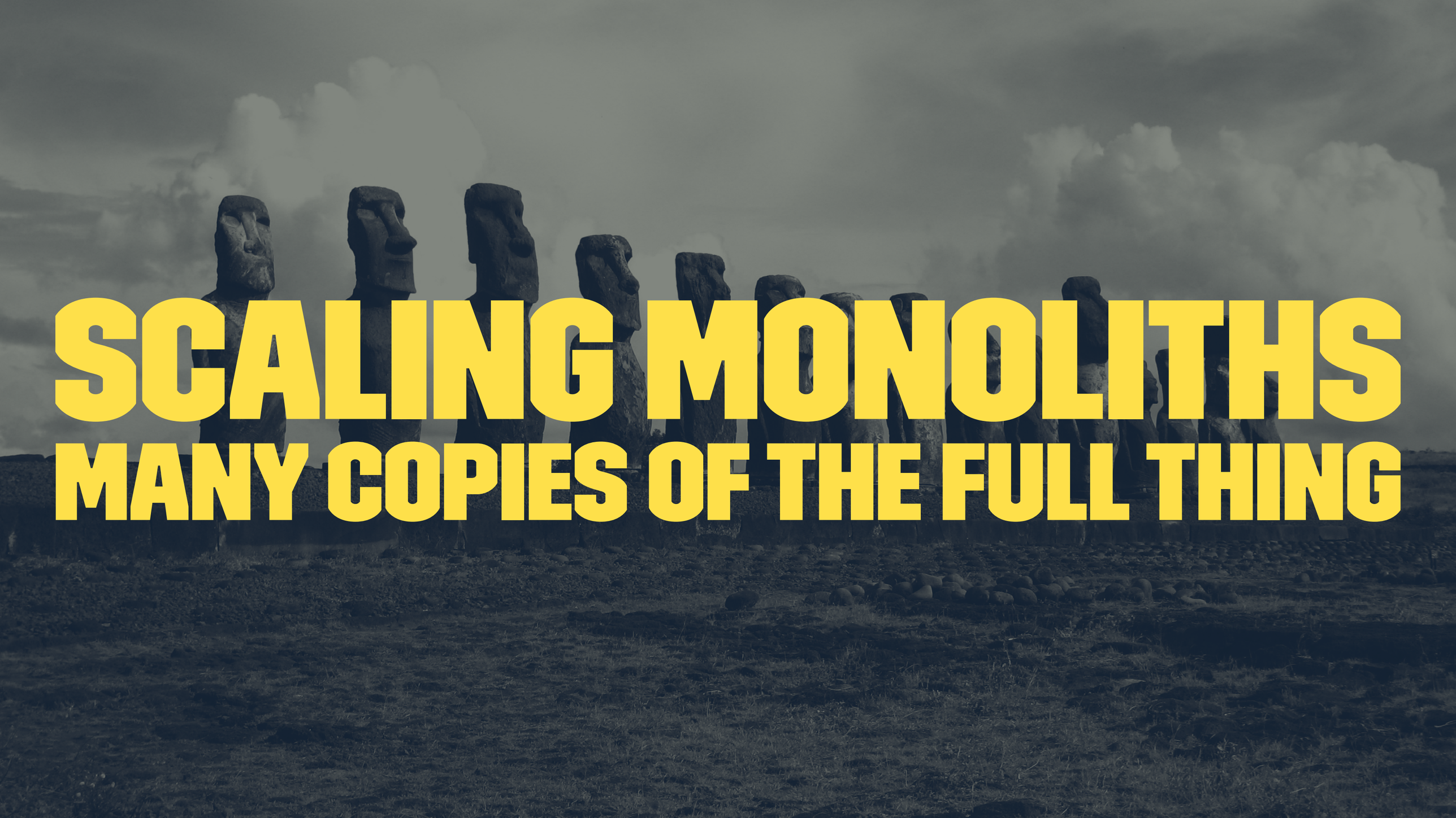
MONOLITH

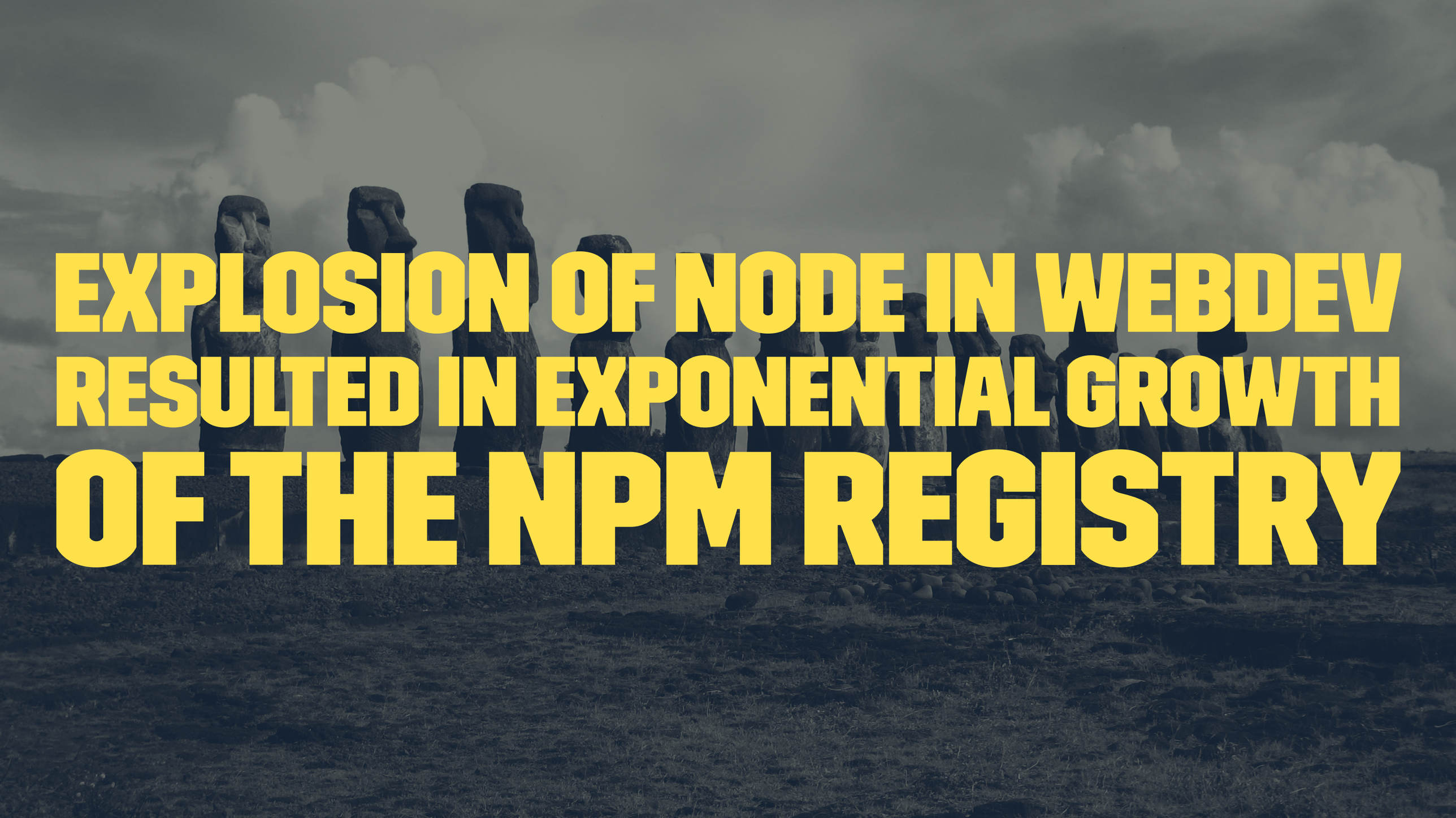
EVERYTHING IN ONE PROCESS

MONOLITHS WORK JUST FINE

WHATEVER IT TAKES TO BUILD A SYSTEM THAT SATISFIES YOUR USERS

SUCCESS!
NOW SCALE IT.


SCALING MONOLITHS
MANY COPIES OF THE FULL THING


EXPLOSION OF NODE IN WEBDEV
RESULTED IN EXPONENTIAL GROWTH
OF THE NPM REGISTRY


EXPONENTIAL MONOLITHS
WERE GOING TO BE EXPENSIVE

SPLITTING THE MONOLITH

YAY MICROSERVICES?

YOUR MONOLITH IS COMPLEX.
A SPLIT SYSTEM IS MORE COMPLEX.

WHAT DID THAT GALL GUY SAY ABOUT COMPLEX WORKING SYSTEMS?

HOW DO YOU SPLIT A MONOLITH SUCCESSFULLY?

LET'S CHEAT.

Q: HOW DO YOU CHEAT?

A: BY NOT REWRITING THE WHOLE THING.

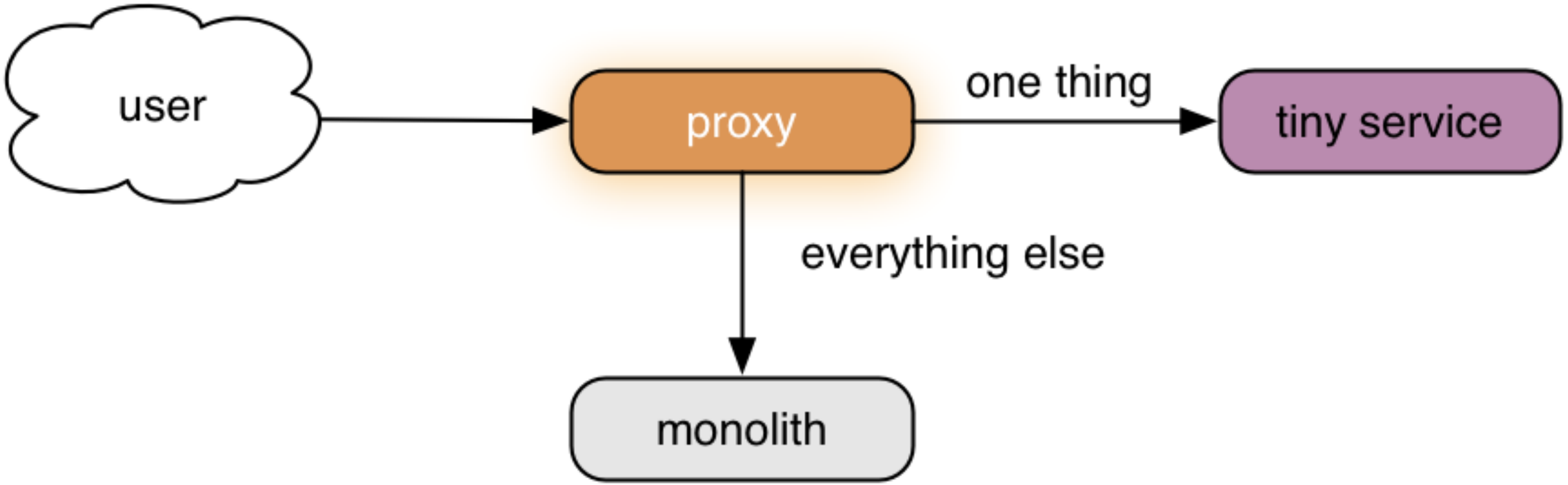
SLICE OFF A PART OF THE SYSTEM

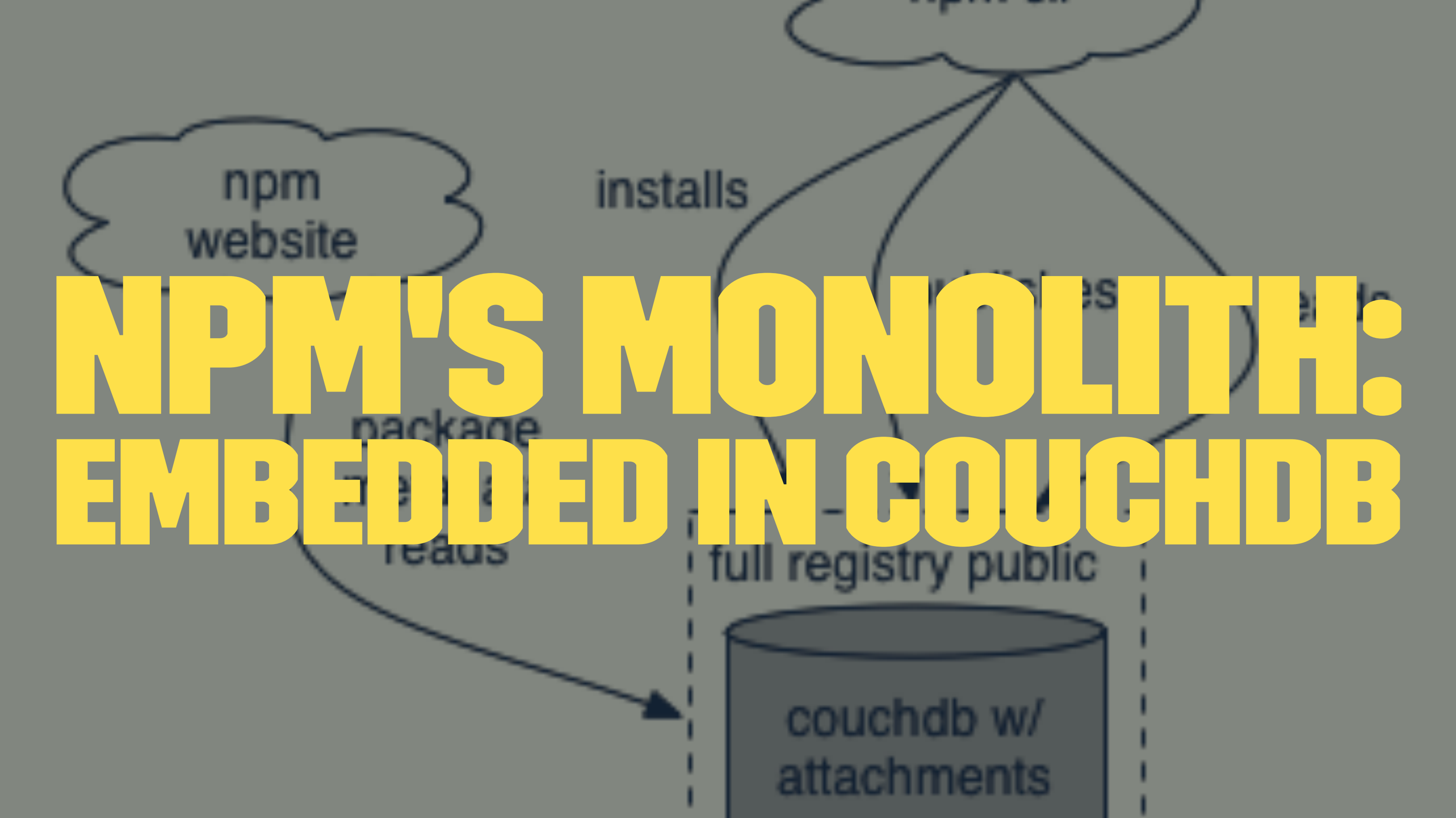
INTO A MODULE

WITH A CLEARLY-DEFINED INTERFACE

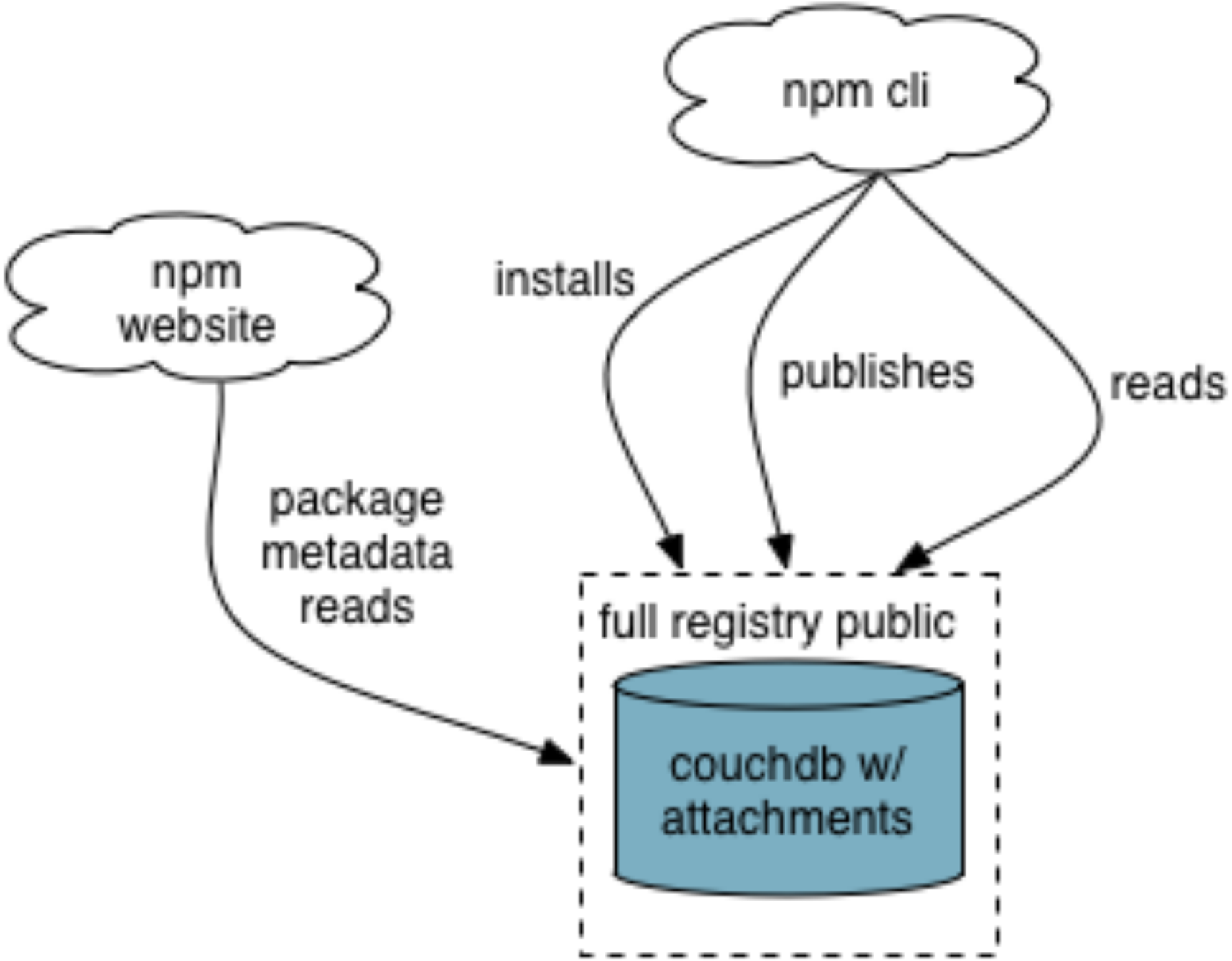
THEN WRITE A SECOND IMPLEMENTATION OF THAT INTERFACE

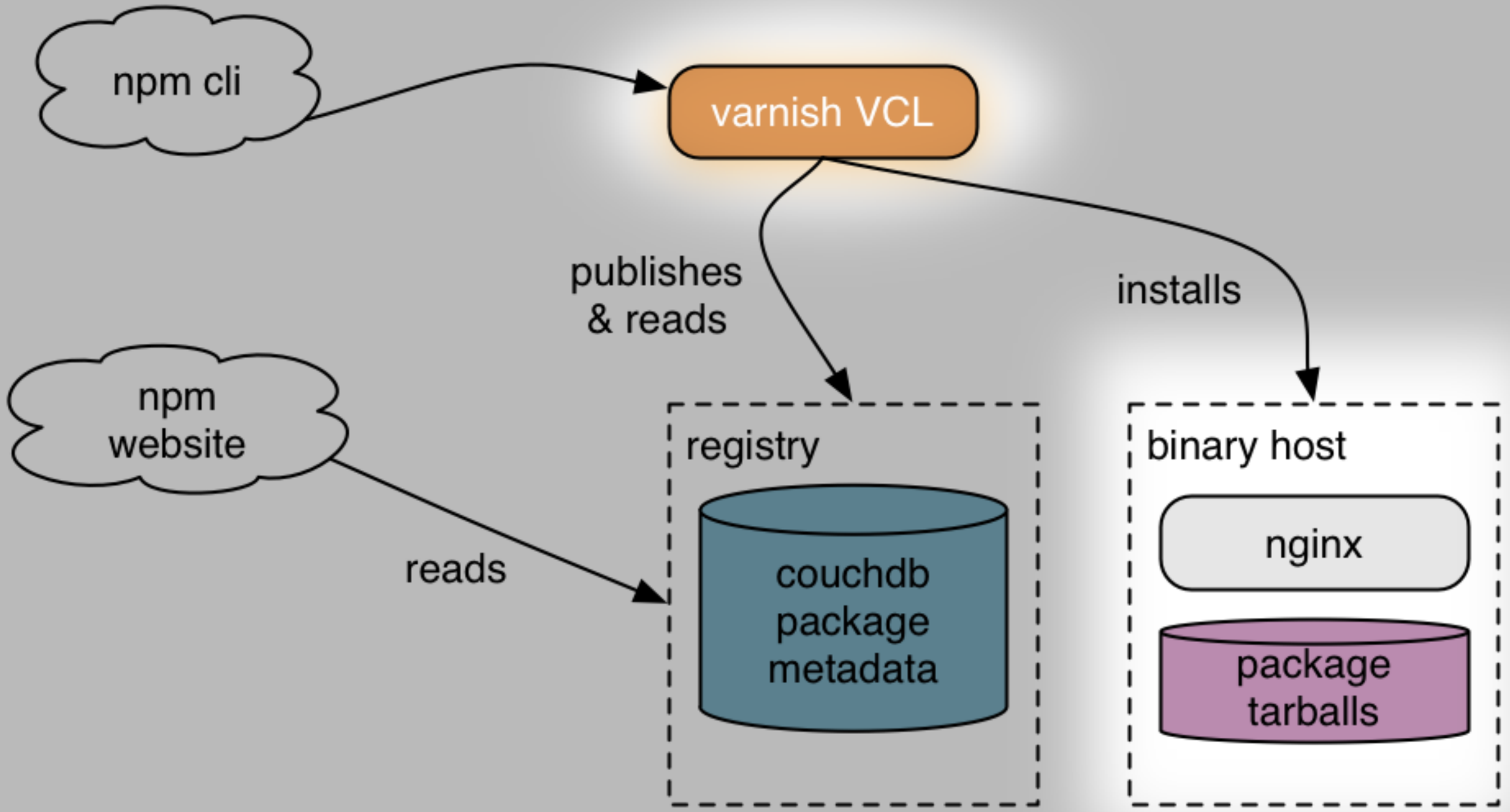
SEND REQUESTS TO THAT SECOND IMPLEMENTATION WITH A PROXY


user
proxy
one thing
tiny service
everything else
monolith



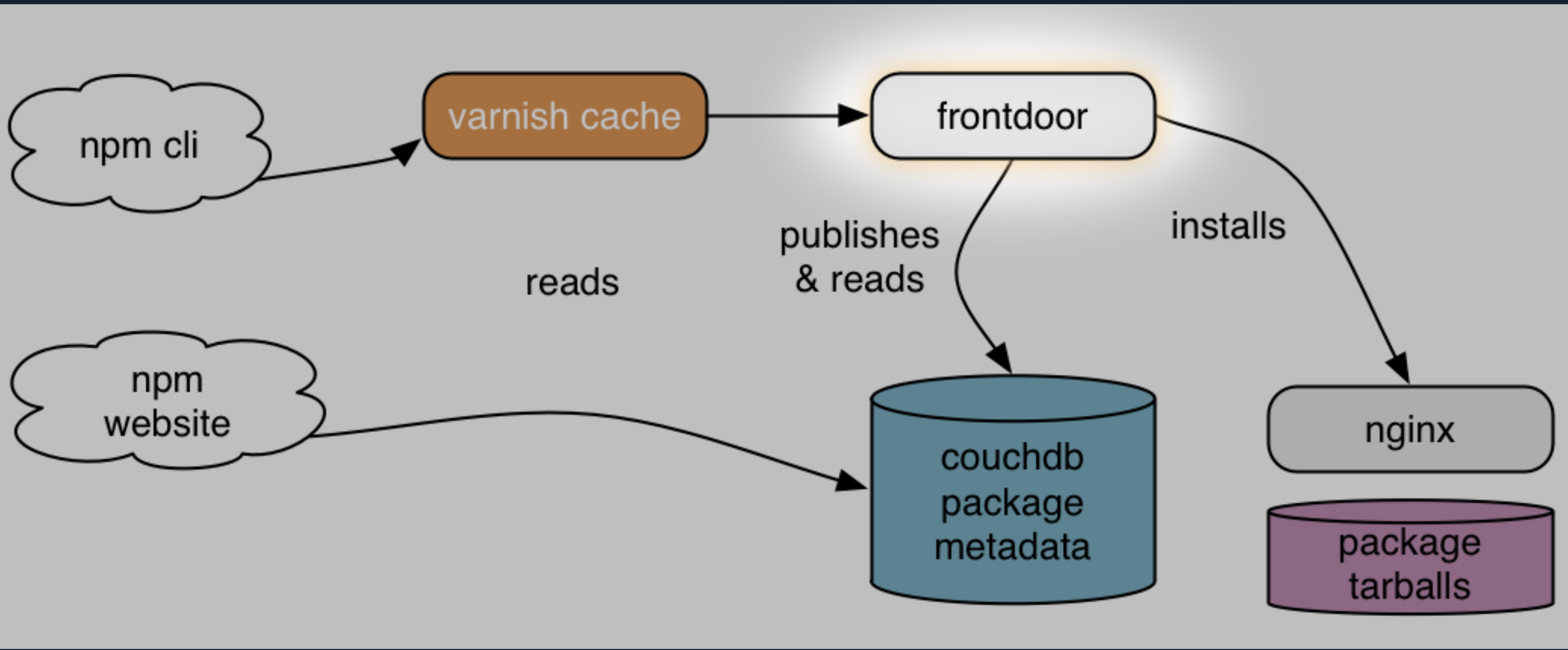
NPM'S MONOLITH: EMBEDDED IN COUCHDB


npm cli
npm
website
installs
publishes
reads
package
metadata
reads
full registry public
couchdb w/
attachments

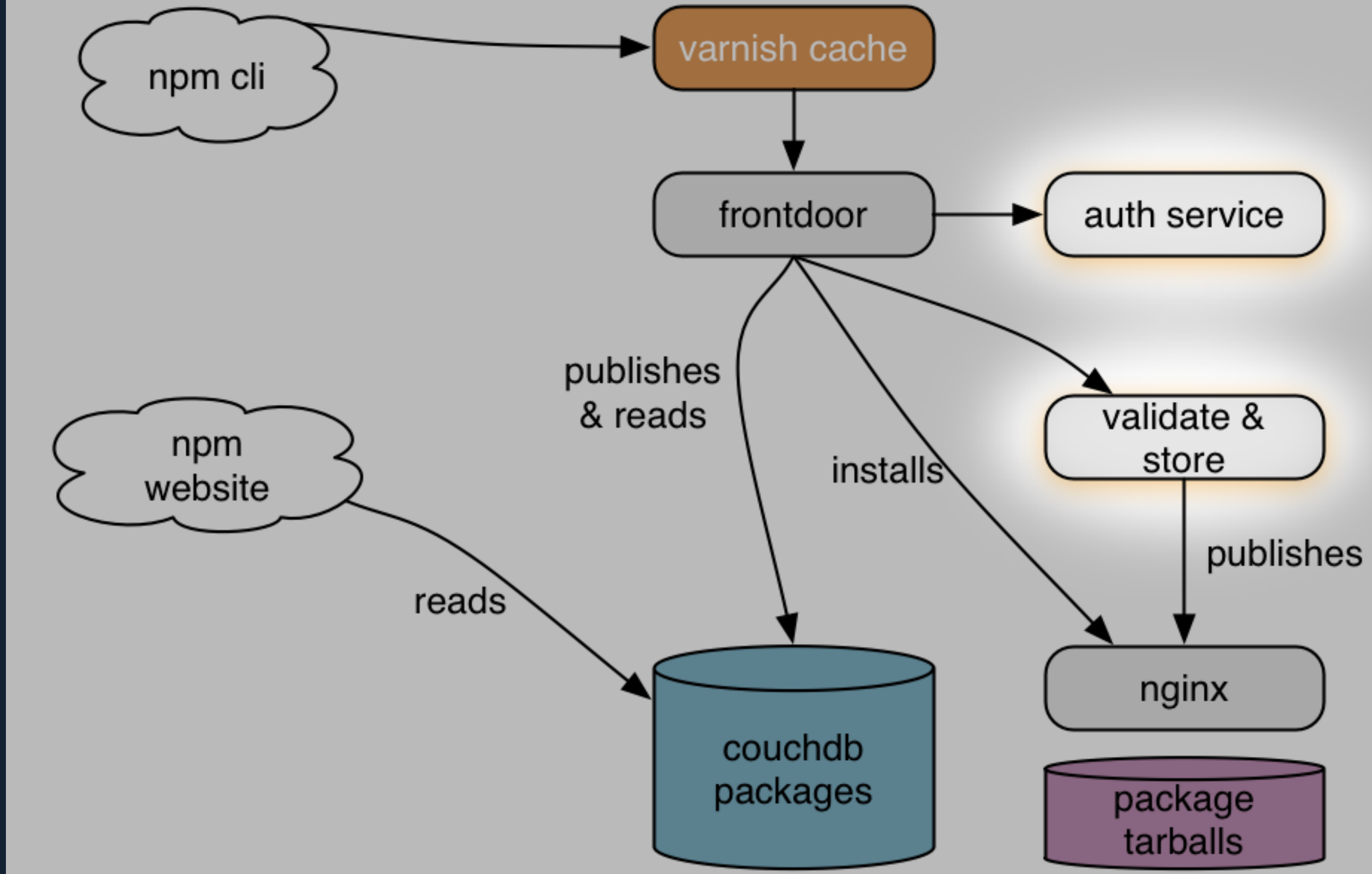

npm cli
varnish VCL
publishes
& reads
installs
npm
website
reads
registry
couchdb
package
metadata
binary host
nginx
package
tarballs

NOW LET'S MAKE OUR PROXY SMART

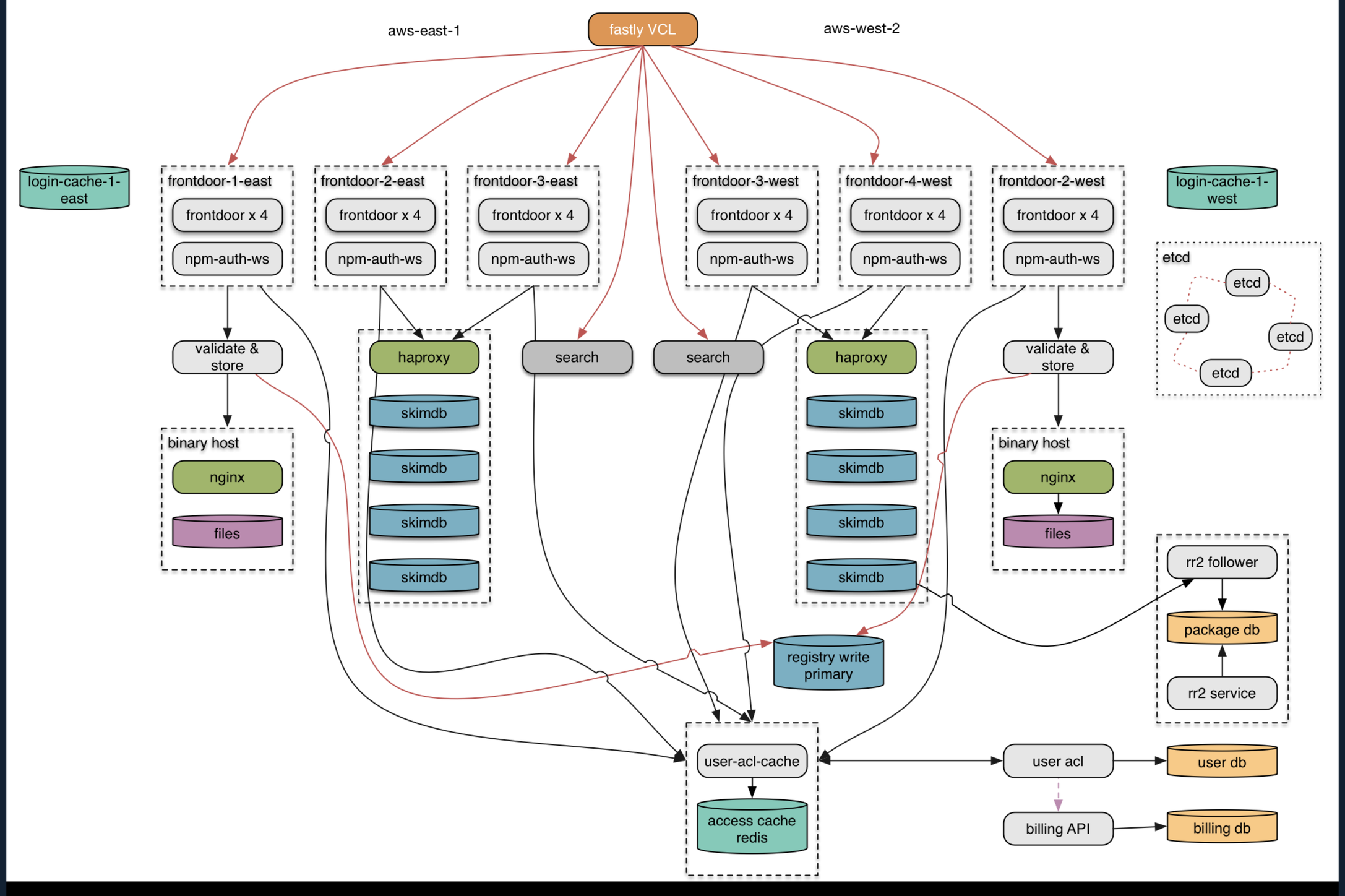
REPLACE VARNISH WITH A MICROSERVICE

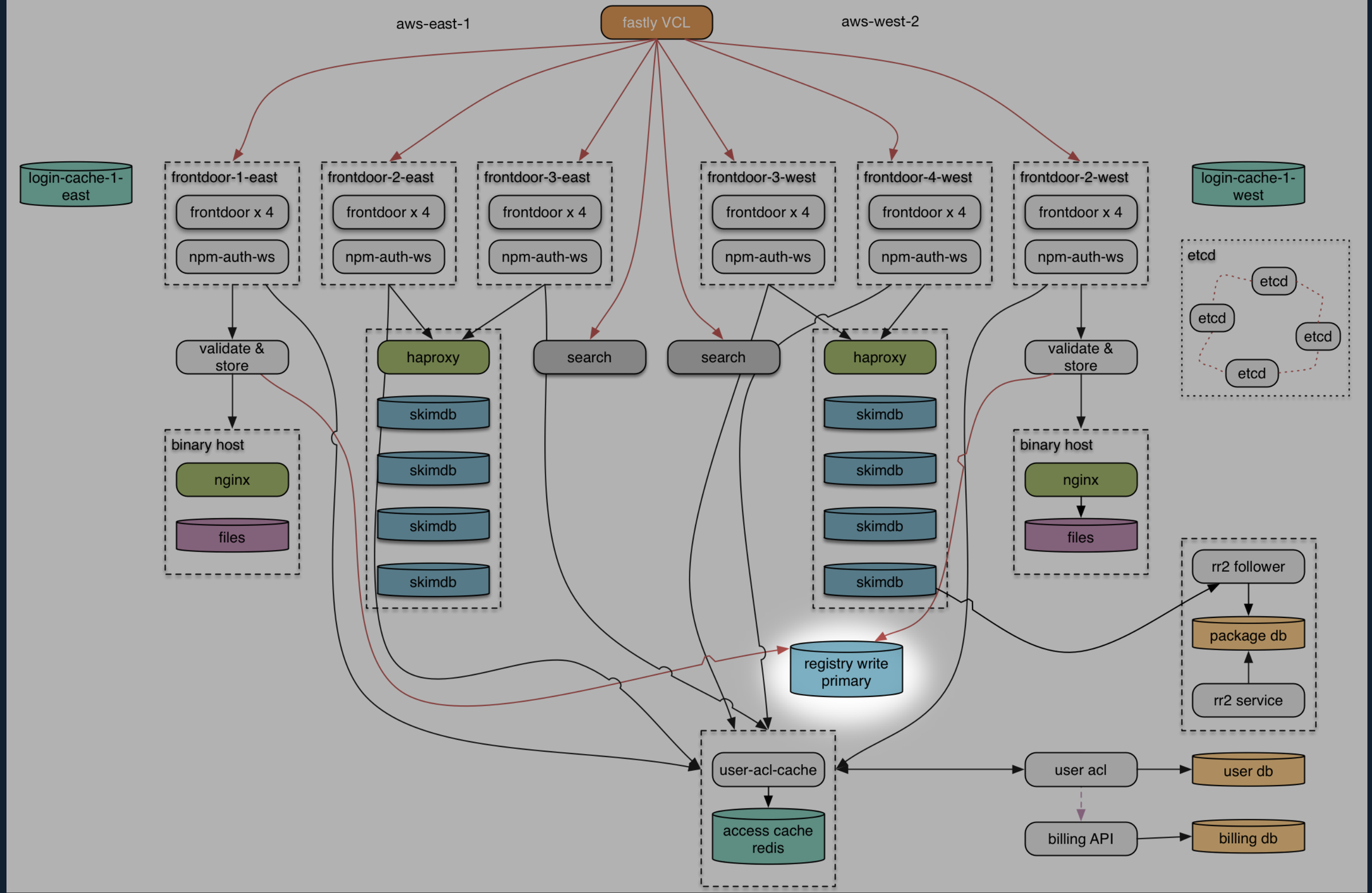

npm cli
varnish cache
frontdoor
reads
publishes
& reads
installs
npm
website
couchdb
package
metadata
nginx
package
tarballs

NOW DO IT AGAIN


npm cli
varnish cache
frontdoor
auth service
publishes
& reads
installs
validate &
store
npm
website
reads
publishes
couchdb
packages
nginx
package
tarballs

AND AGAIN


aws-east-1
fastly VCL
aws-west-2
login-cache-1-east
frontdoor-1-east
frontdoor x 4
npm-auth-ws
frontdoor-2-east
frontdoor x 4
npm-auth-ws
frontdoor-3-east
frontdoor x 4
npm-auth-ws
frontdoor-3-west
frontdoor x 4
npm-auth-ws
frontdoor-4-west
frontdoor x 4
npm-auth-ws
frontdoor-2-west
frontdoor x 4
npm-auth-ws
login-cache-1-west
etcd
etcd
etcd
etcd
etcd
validate & store
haproxy
search
search
haproxy
validate & store
skimdb
skimdb
skimdb
skimdb
skimdb
skimdb
skimdb
skimdb
binary host
nginx
files
binary host
nginx
files
rr2 follower
package db
rr2 service
registry write primary
user-acl-cache
access cache redis
user acl
user db
billing API
billing db


aws-east-1
fastly VCL
aws-west-2
login-cache-1-east
frontdoor-1-east
frontdoor x 4
npm-auth-ws
frontdoor-2-east
frontdoor x 4
npm-auth-ws
frontdoor-3-east
frontdoor x 4
npm-auth-ws
frontdoor-3-west
frontdoor x 4
npm-auth-ws
frontdoor-4-west
frontdoor x 4
npm-auth-ws
frontdoor-2-west
frontdoor x 4
npm-auth-ws
login-cache-1-west
etcd
etcd
etcd
etcd
etcd
validate & store
haproxy
search
search
haproxy
validate & store
skimdb
skimdb
skimdb
skimdb
skimdb
skimdb
skimdb
skimdb
binary host
nginx
files
binary host
nginx
files
rr2 follower
package db
rr2 service
registry write primary
user-acl-cache
access cache redis
user acl
user db
billing API
billing db

BEAT GALL'S LAW WITH MODULARITY AKA INFORMATION HIDING

EACH SERVICE IS A MODULE

A SIMPLE SYSTEM

WHEN VIEWED IN ISOLATION

YOU HAVE A WORKING SYSTEM EVERY STEP OF THE WAY

SIMPLE WORKING SERVICE FIRST

SCALE IT LATER

RESPECT GALL: REWRITE IN PIECES

USE A PROXY
TO DIVIDE & CONQUER

BE BOLD

YOU CAN CHANGE YOUR SYSTEM

NOBODY NOTICED
THAT WE SLOWLY REPLACED
THE ENTIRE NPM REGISTRY

LOVES YOU

```
npm install -g npm@latest
```