Docker at Lyft

Speeding up development

Matthew Leventi @mleventi #dockercon

dockercon 15

SF JUNE 22-23

Lyft Engineering

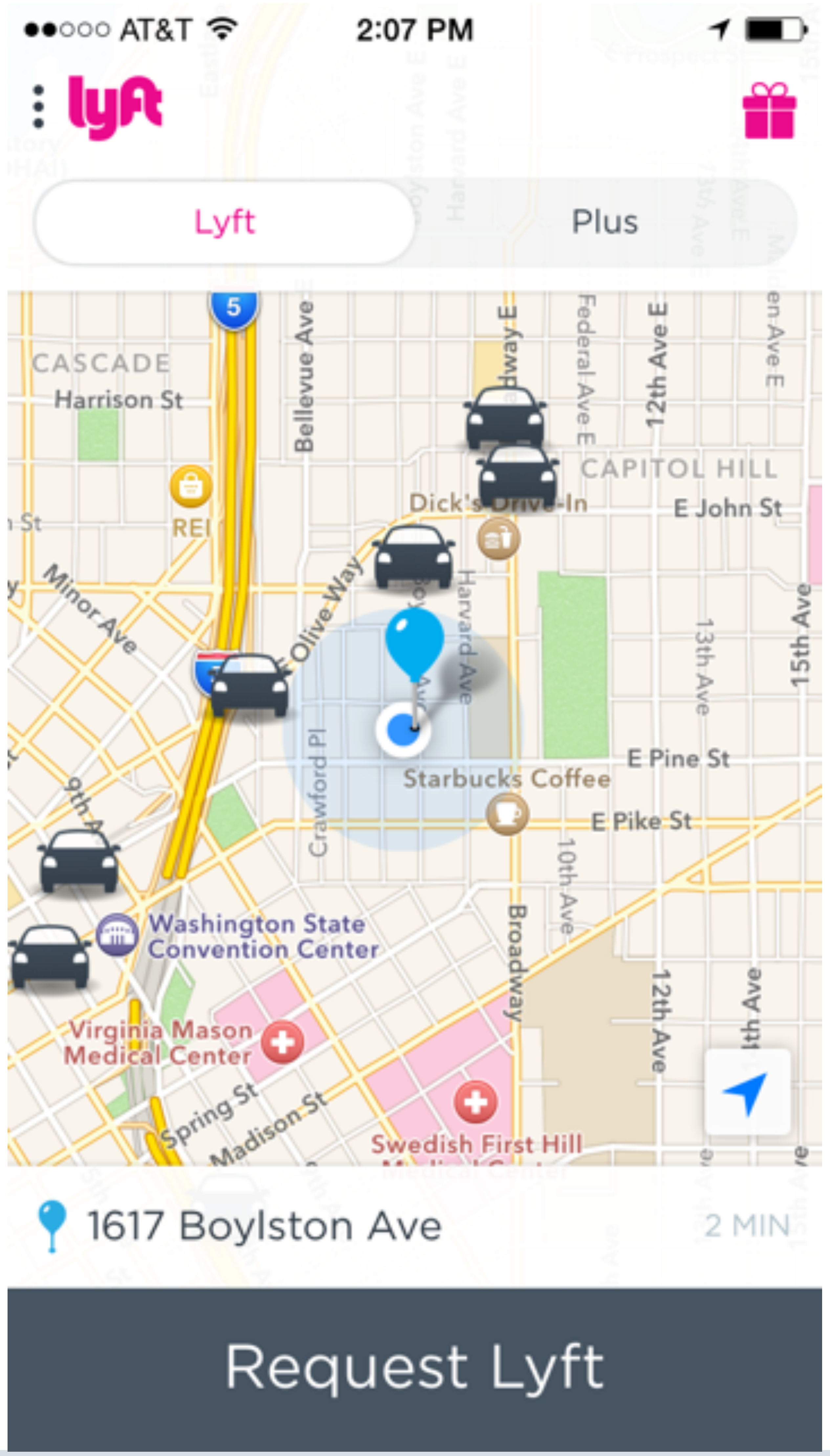
Lyft Engineering

Organization

- Rapidly growing headcount
- Fluid teams
- Everyone does devops

Technology

- 50+ microservices
- 25 server deploys a day
- 2 client pushes a week
- Highly available



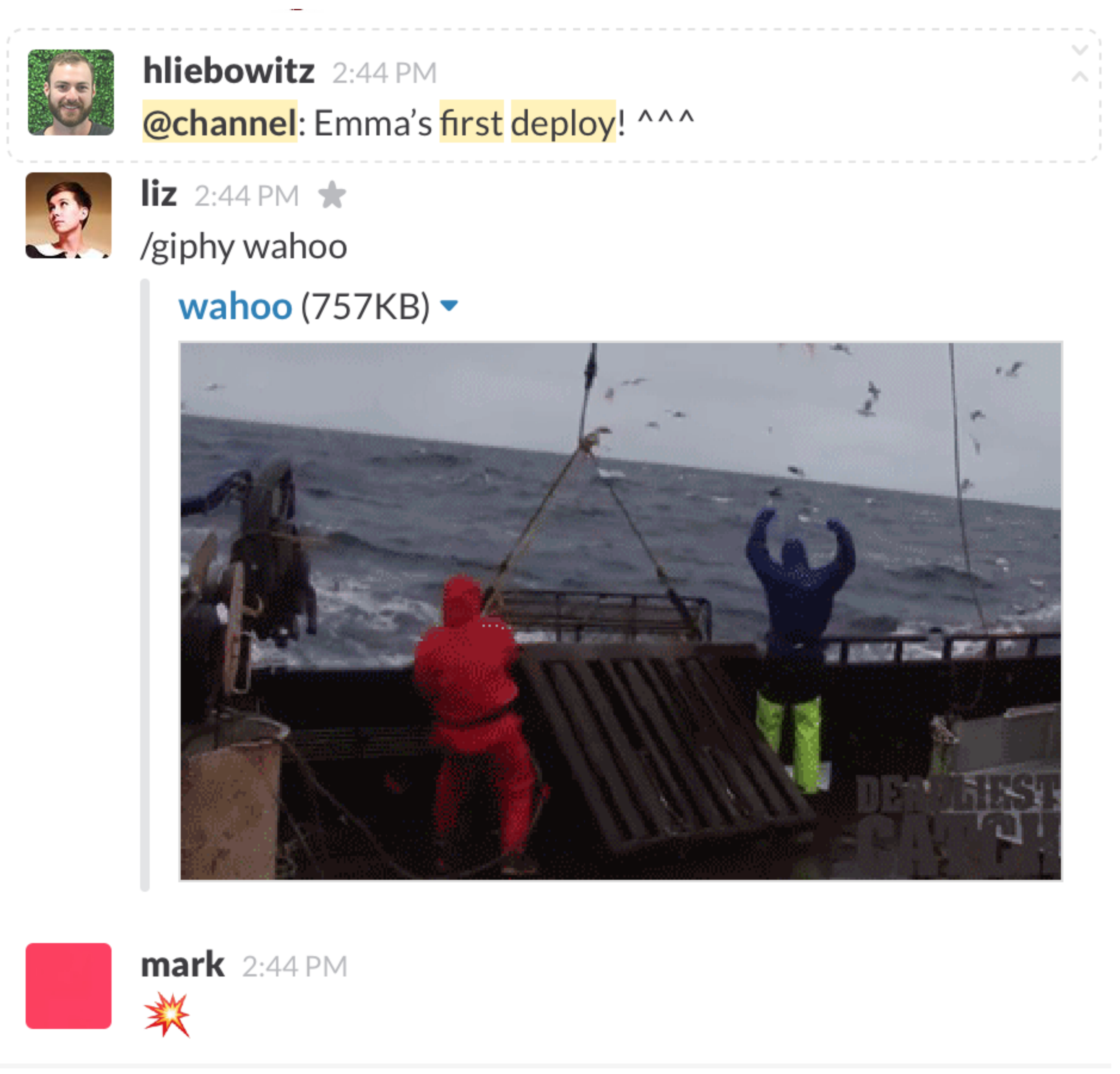
Systems Engineering

Developer Productivity

- New developers ship on Day 1
- Seamless team switches
- Faster feature development

Operational Stability

- everything must scale
- nothing goes down

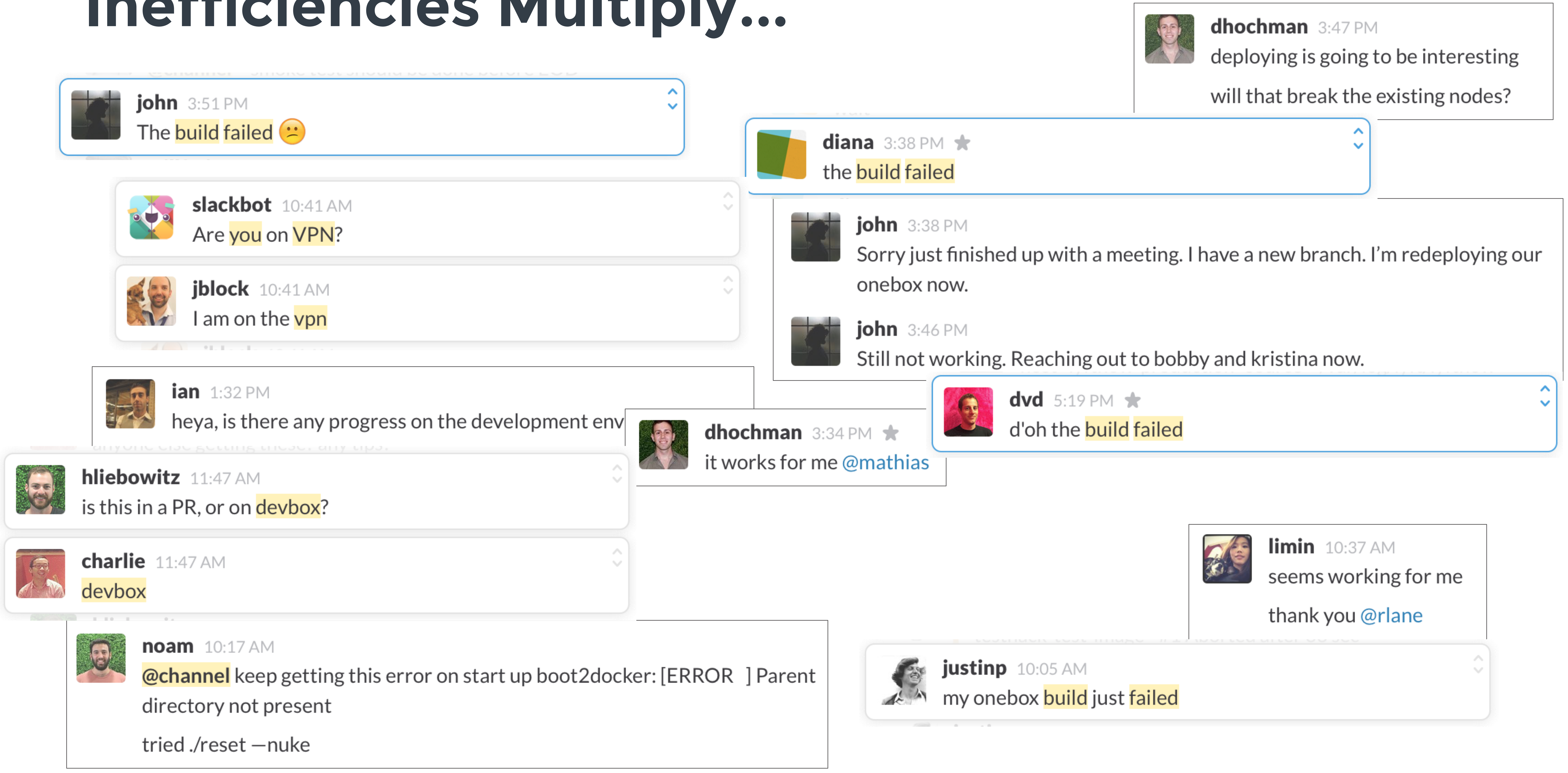

hliebowitz 2:44 PM
@channel: Emma's first deploy! ^^^

liz 2:44 PM
/giphy wahoo
wahoo (757KB)

mark 2:44 PM
💥

Developer Productivity

Inefficiencies Multiply...



john 3:51 PM
The build failed 😕

slackbot 10:41 AM
Are you on VPN?

jblock 10:41 AM
I am on the vpn

ian 1:32 PM
heya, is there any progress on the development env

hliebowitz 11:47 AM
is this in a PR, or on devbox?

charlie 11:47 AM
devbox

noam 10:17 AM
@channel keep getting this error on start up boot2docker: [ERROR] Parent directory not present
tried ./reset —nuke

dhochman 3:47 PM
deploying is going to be interesting
will that break the existing nodes?

diana 3:38 PM
the build failed

john 3:38 PM
Sorry just finished up with a meeting. I have a new branch. I'm redeploying our onebox now.

john 3:46 PM
Still not working. Reaching out to bobby and kristina now.

dhochman 3:34 PM
it works for me @mathias

dvd 5:19 PM
d'oh the build failed

limin 10:37 AM
seems working for me
thank you @rlane

justinp 10:05 AM
my onebox build just failed

General Problems

“It doesn’t work on my box!”

“I don’t understand how the client got into that state!”

“It worked in development!”

“How do I get service X to talk to service Y?”

“How do I test this feature from the client?”

“How do I get started working on a new team?”

Invest in Dev Environments

In the past…

AWS Dev EC2 Instances - 1 per dev per service

- NFS syncing for code changes
- Service discovery through dev config sections
- Manual task to stay up to date on changes
- Individual SQS, Dynamo resources per developer
- Expensive to orchestrate

Vagrant VM Images

- Hard to run more than 2 on a mac
- Hard to interface with cloud resources.

Development Environment

Devbox: Everyone has the same up to date local environment

Onebox: All of lyft, in the cloud, running any combination of builds

CI: Cross service integration testing, deploys

Devbox

Start a set of services easily:

```
./service start api dispatch eta
```

Automatically mount repos into services:

```
ls .
api dispatch eta payments python-sdk
```

Load and save state snapshots:

```
./service snap issue519
./service apply issue519
```

Open websites locally

```
./service open api
```

Build new services locally

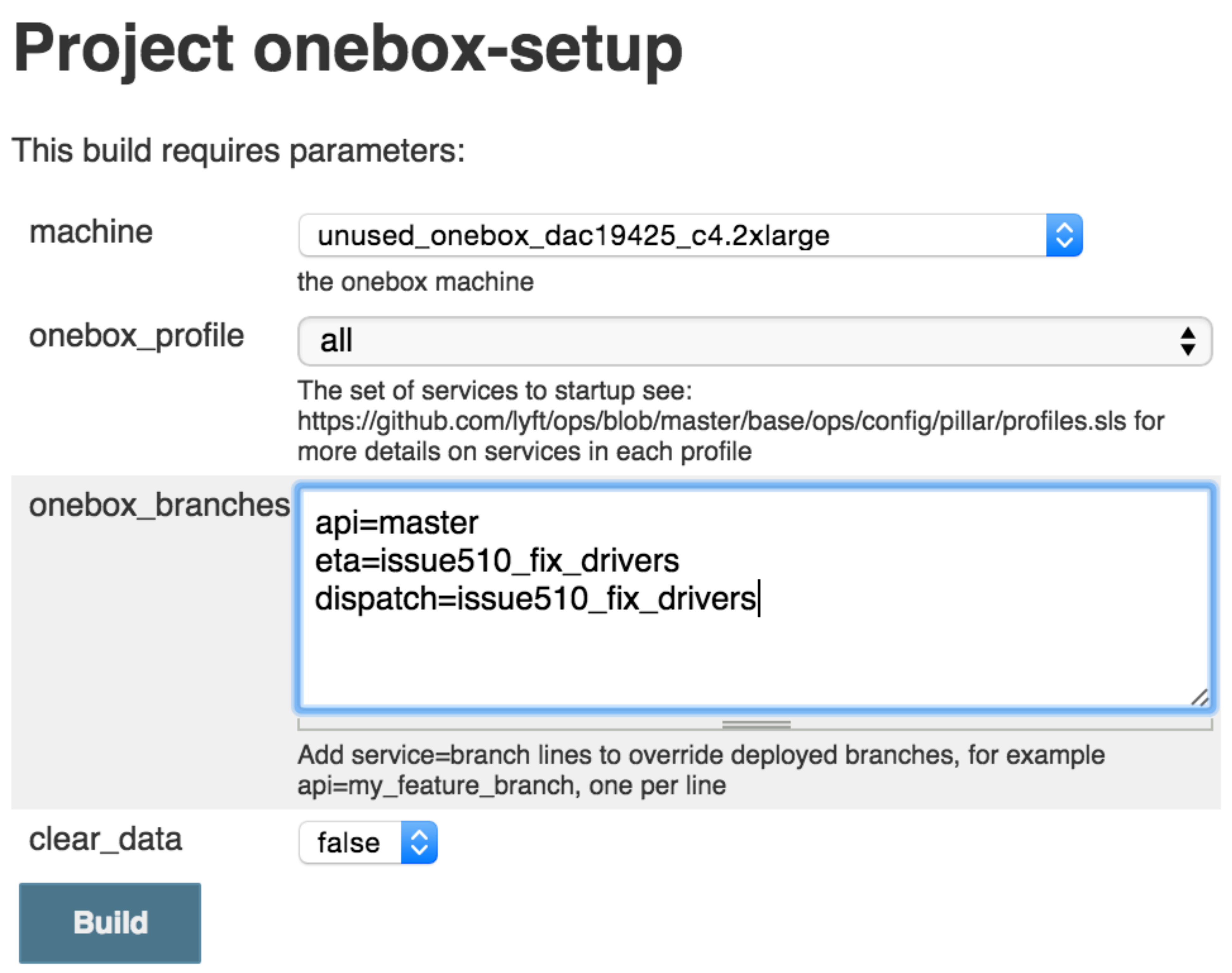
```
./service build new_service_X
```

Onebox

Every QA engineer has their own environment.

No mocking needed for client development.

Easy to share state between developers.



Project onebox-setup

This build requires parameters:

machine	unused_onebox_dac19425_c4.2xlarge the onebox machine
onebox_profile	all The set of services to startup see: https://github.com/lyft/ops/blob/master/base/ops/config/pillar/profiles.sls for more details on services in each profile

onebox_branches	api=master
	eta=issue510_fix_drivers
	dispatch=issue510_fix_drivers
	Add service=branch lines to override deployed branches, for example api=my_feature_branch, one per line

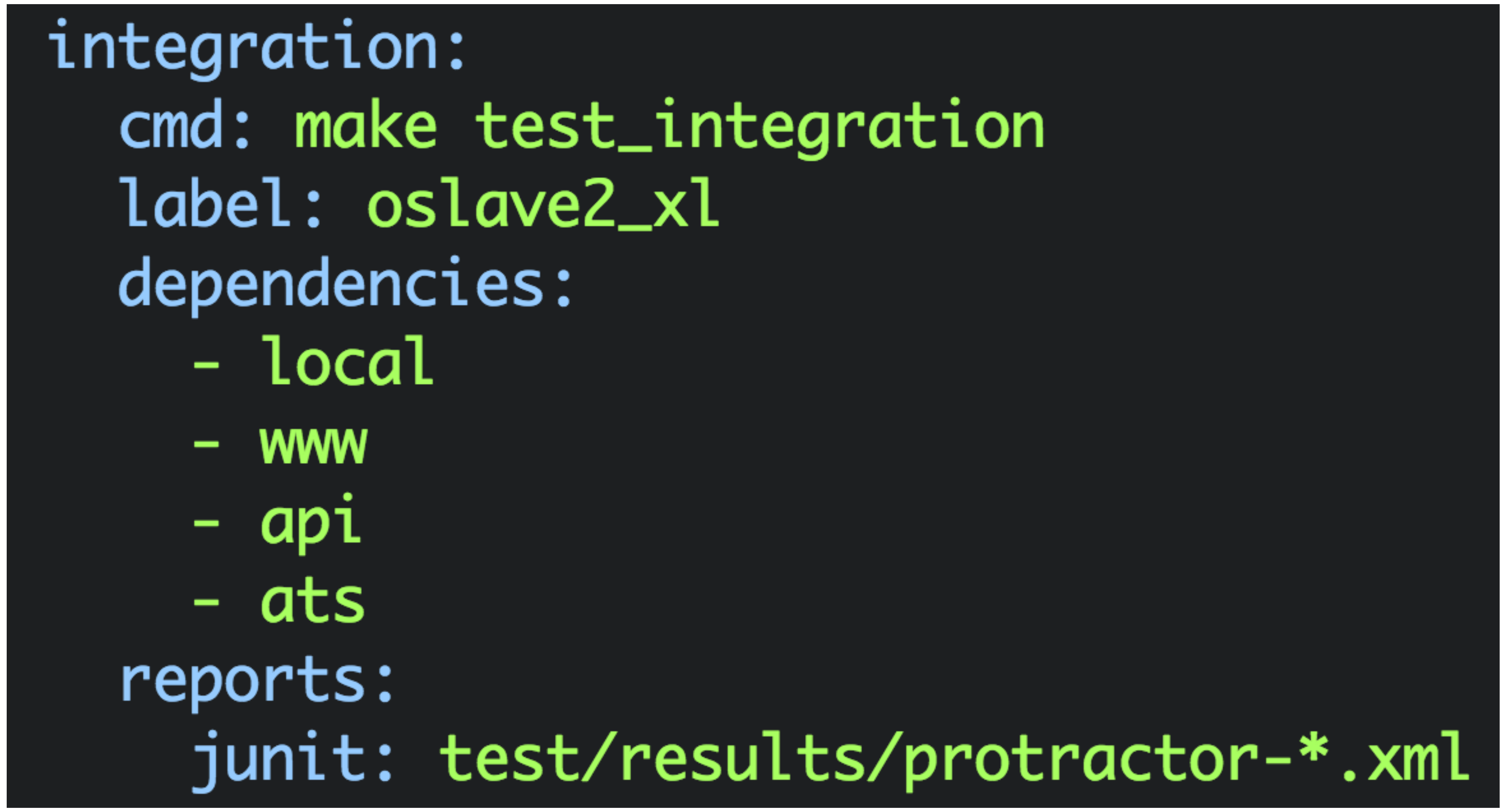
clear_data	false

CI

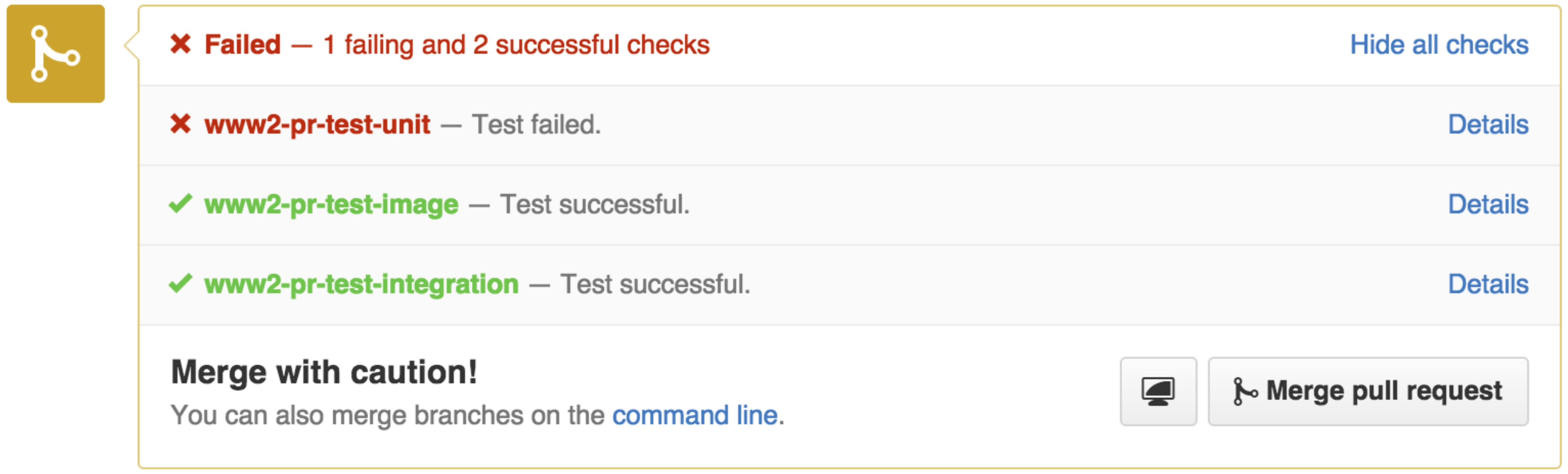
Every service defines test suites with dependent services.

Tests are run per pull request and on master commits.

Isolated cross service integration tests.



```
integration:
  cmd: make test_integration
  label: oslave2_xl
  dependencies:
    - local
    - www
    - api
    - ats
  reports:
    junit: test/results/protractor-*.xml
```



✗ Failed — 1 failing and 2 successful checks — Hide all checks

✗ www2-pr-test-unit — Test failed. — Details

✓ www2-pr-test-image — Test successful. — Details

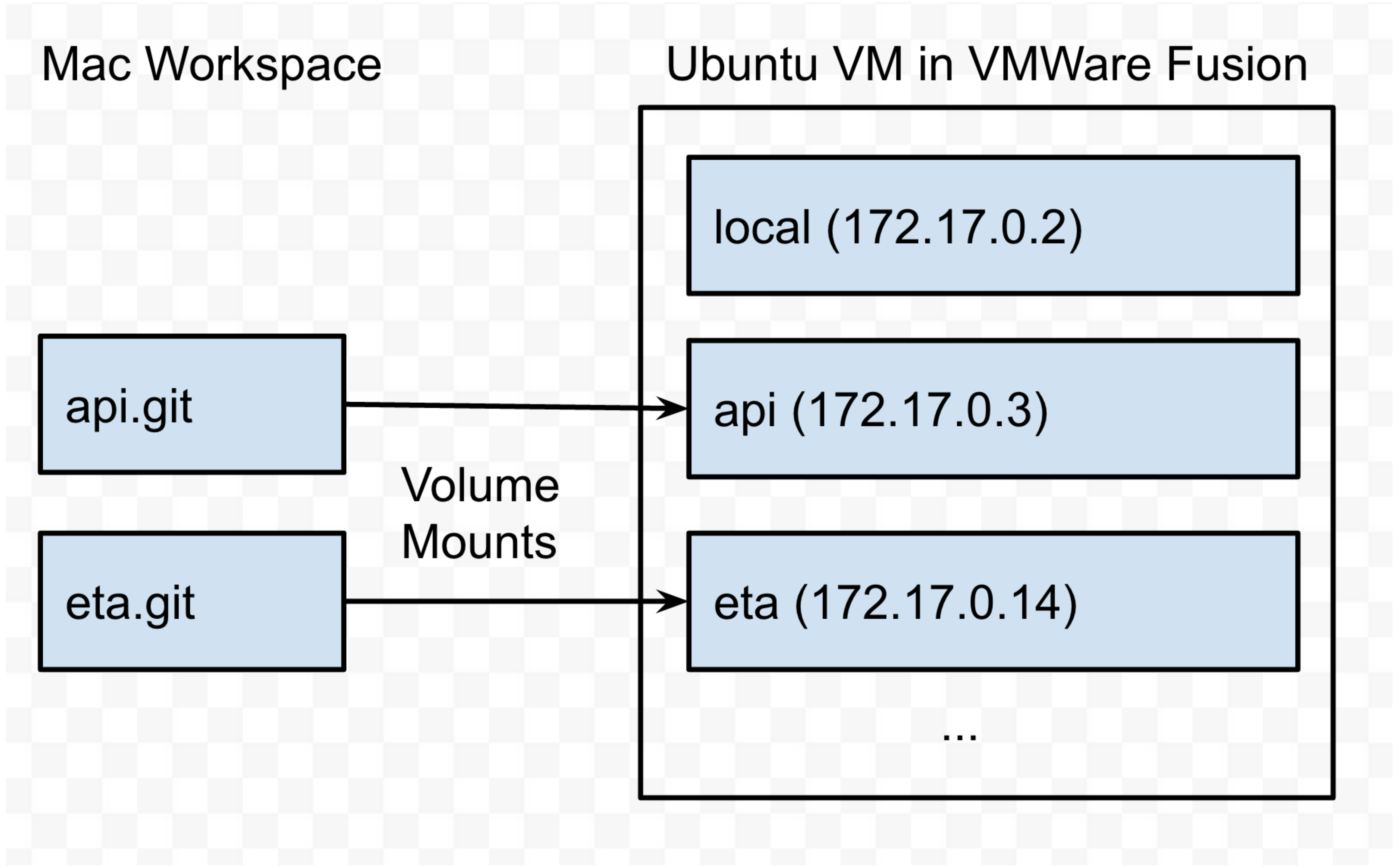
✓ www2-pr-test-integration — Test successful. — Details

Merge with caution!
You can also merge branches on the command line.

How?

Service Model

- Single fat containers
- Stateless
- Fixed static ip address
- Single “stateful” local container
- Auto detect code changes



Building a Service Image

- Docker image is a fs snapshot of config management.
- Each image has:
  - git clone of a central ops codebase
  - git clone of a service codebase
  - a salt stack provisioning run.
  - runit configuration for processes

```
ID = $(docker run --env SERVICE=api --env SERVICE_SHA=abc --env OPS_SHA=def lyft/base)
docker commit $ID api
docker push api
```

No dockerfiles!

Running a Service Image

- Rerun salt provisioning on new SHAs
- Start runit processes
- Terminate the container if initial runit checks fail

Allows

- Developers can easily apply ops modifications
- Testing PRs are a matter of changing env variables
- Don’t need to wait for an image build, deltas are applied during runtime
- Easy to mount code volumes and trigger changes

Single Host

DevBox

Mac docker host using vmware fusion with shared folders

CI Slave

AWS ubuntu docker host for short lived containers

Onebox

AWS ubuntu docker host for long lived environments

Managing State

All stateful processes run inside the same container.

- Redis
- MongoDB
- DynamoLocal
- SQS Local
- Fake Kinesis

Standard import/export scripts to S3 tar files.

All developers, qa, slaves get their own data environment.

Demo

Results

Results

Productivity

Majority of new hires push to production on day one.

Feature development is no longer blocked by devops.

QA client testing is parallelized.

Stability

99% of deploys are successful.

Every PR on every service is integration tested.

Lessons Learned

Lessons Learned

VMWare Fusion can be unstable under load

Frequent image downloads take time

Bugs in config management can freeze development

Easy service creation leads to unnecessary services

Approach limits on what can run on a single box

Static IP allocation not supported in docker

Future

Future Ideas

Tons of t2.smalls to replace VMWare:

- One container per host in the cloud
- NFSv4 code syncing
- Same static ip private network using libnetwork
- docker-machine

Exploring production docker usage:

- ETL jobs in docker
- Containers to reduce ASG spin up/down times
- Containers for atomic deploys

Thank you

Matthew Leventi

mleventi@lyft.com @mleventi #dockercon

dockercon 15