From Commit to Cloud

The why and how of an efficient build pipeline

Daniel Hall

LIFX Cloud Engineer

@smarthall

Deployments should be

Fast

- 10 minutes is too long

Small

- Ideally a single commit
- You must be aware of the whole change

Easy

- As little human involvement as possible
- Minimal context switching
- Simple to understand

We believe this because

- There are less things to break
- You can focus on one thing at a time
- The changes are fresh in everyone's memory
- Devs should be focused on the app
- Each project should be easy to learn
- Nobody should be afraid to deploy

We can back it up too

- We have 30 separate microservices
- 88 Docker machines running across 15 workers
- Around 7 deploys to prod every working day
- 4 emergency rollbacks in 1.5 years
- Maintained by myself and a single developer

How did we do it?

You'll need:

- A microservices architecture (kinda optional)
- A git repo that can run triggers on push
- A build agent that allows several steps
- A packaging format that tracks dependencies
- Something to manage your cluster of machines

Deployment steps

1. Write some code
2. Push to the git repo
3. **Application is built**
4. **Automated tests are run**
5. **Application is packaged**
6. **Deployment to staging**
7. Test in staging
8. Single click to approve for prod
9. **Deployment to production**

3. Application is built

- Happens on trigger
- Not all apps need to be built
- We have a build.sh script in each repo
- We wanted to be consistent across all our services
- You could also use Makefiles

4. Automated tests are run

- We actually do this at the end of build.sh
- Again consistency is key
- Don't do end to end testing here
- We do end to end testing continuously against staging and prod

5. Application is packaged

- We build docker containers with a Dockerfile
- Push it to an internal docker repository on GCS
- Anyone should be able to pull this for testing
- You could also use RPMs, Debs, and others
- If you do make per app repos, to make repo metadata creation time short

6. Deployment to staging

- We use Apache Mesos with Marathon
- For us we essentially assemble a JSON file from a template
- Then submit it to Marathon using curl
- You can use these with or without docker

8. Single click approval

- At this point our change is well tested
- Trigger the deployment to staging
- Most build agents have this, or let you build it easily
- Audit who clicks this!

9. Deployment to production

- Should be very similar to prod right?
- Remember, consistency!
- We use Marathon and Mesos in prod too

Thanks!

LIFX is seeking a **Cloud Engineer** and **Firmware Engineer** to join our **Melbourne** team.

Either see this link, email me at daniel.hall@lifx.com or come talk to me at LCA all week.

See: https://goo.gl/0KOX1C and https://goo.gl/YY30Gh

This Talk

Will be available at:

- https://slides.com/smarthall/from-commit-to-cloud
- I'll post this URL on my Twitter (@smarthall)