## **Docker at Spotify**

Rohan Singh @rohansingh





### **Spotify**

"Music for every moment"

40+ million active users

20+ million songs available (we're in 55+ countries, differs by market)

Live in the US for 2 years

### What are we working with?

100+ distinct backend services

5000+ production servers in 4 sites worldwide

Almost 300 servers per ops engineer

# Deployment at Spotify

Build Debian packages

Requisition new hardware

Use combination of Puppet and internal tooling (deployify)

### old way:

SSH to a bunch of machines, apt-get install II upgrade

#### deployify:

Uses fabric, which just SSH's, runs apt-get install II upgrade

# What's wrong with that?

Requisitioning new hardware is slow

Hardware utilization is low

We've automated and improved, but still too slow

Deployments often fail...

...though usually only partially

Machines get denormalized easily

## Debian packaging is its own special hell

So many damn files!

debhelper

Distro & version specific

apt-get update

# What do we want?

### Deployments that are:

Repeatable

Straightforward

Fault-tolerant

## Why Docker?

It gives us "repeatable", right out of the box

And it's fairly straightforward, too!

## Repeatability?

The code you test isn't necessarily the code that runs in prod

## Repeatability — Docker

Build an image

Run tests against the image

Run the tested image in prod

## Straightforward

Just one Dockerfile

Pretty simple syntax

Immutable images — easy to reason about

### Fault-tolerance?

# Even if your code is perfect, deployments might fail

Puppet or Debian repos are down

Network issues

The machine dies

### Fault-tolerance — Docker

Just one thing has to work — pulling & running an image

If it fails, just try again

No machines left in weird states\*

\*(in theory)

## What more do we need?

# Deploying Docker containers across our entire fleet of servers

Some way to deal with Docker or container failures

That's pretty much it

## Looking around for a Docker platform

Several really feature-full offerings

Pretty opinionated and all-encompassing

All we need is reliably deploying containers to a fleet of Docker hosts

# Helios

# Makes sure your containers are deployed & running exactly where you want them...

...and that's pretty much it

github.com/spotify/helios

### What's in the box?

Helios master

frontend that you talk to

Helios agent

runs alongside each Docker instance

Zookeeper

Straightforward CLI

#### How does it work?

#### Create a job —

tell Helios which Docker image to deploy

#### Deploy the job —

tell Helios where to run the Docker containers

# See it in action

# What's next?

#### Helios is just the beginning

Brought up plenty of Helios+Docker agents in our private cloud for our devs

Integrated Helios with JUnit — write JUnit tests run against Docker containers

# Dockerized integration tests

#### This week —

first Helios+Docker-based service in production

In the future —

Containerizing more of our infrastructure

### Give it a go

#### Open source:

github.com/spotify/helios

#### Try it out:

git clone https://github.com/spotify/helios.git \
&& cd helios && vagrant up

Come play with us (spotify.com/jobs)