Hybrid cloud lessons

at Spotify



Spotify

November 2013

About Spotify

Streaming music service

Bare metal, baby

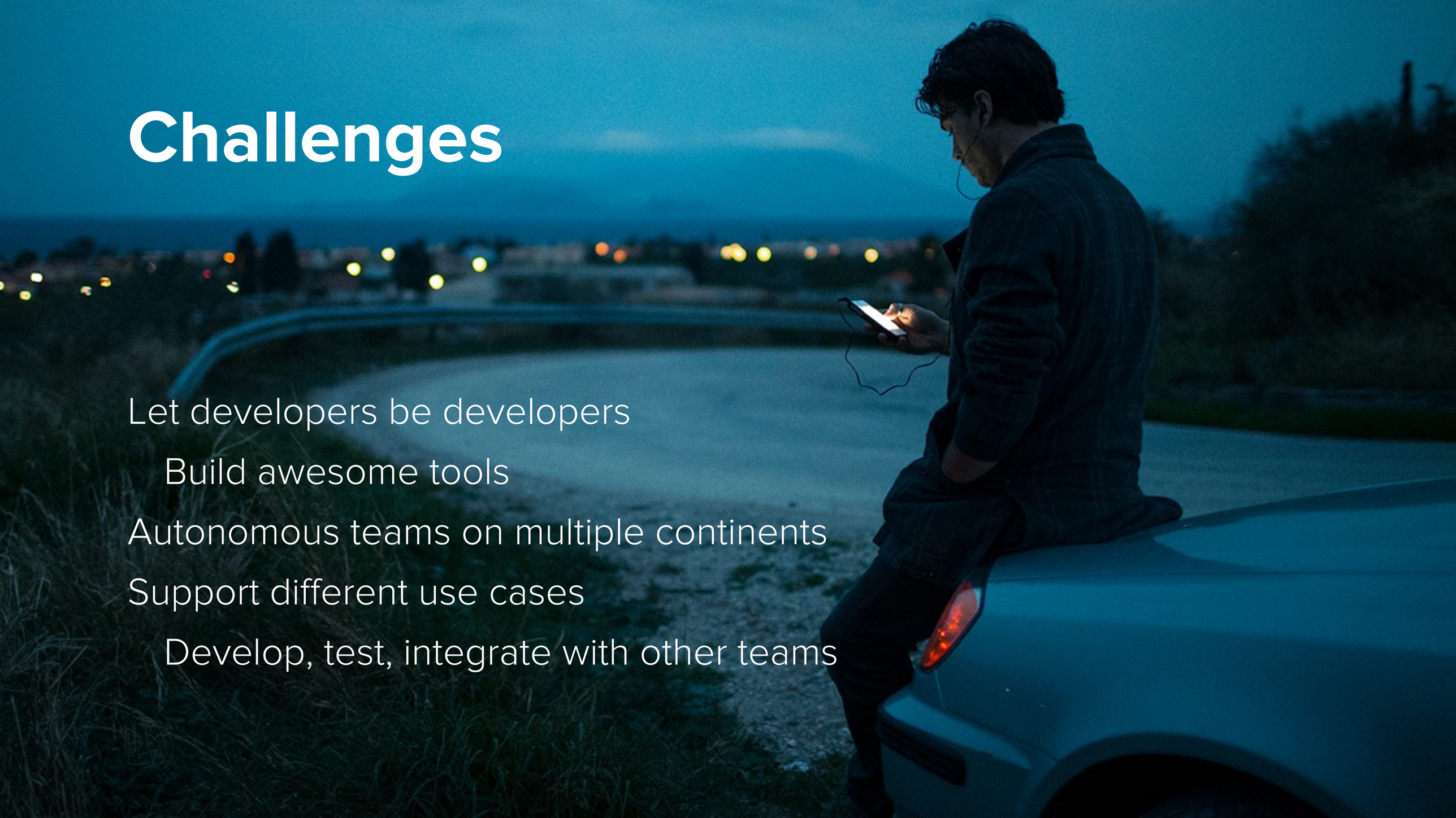
4 Engineering offices, Sweden and USA

Hundreds of engineers

Lots of services

Highly managed configuration

Challenges



Let developers be developers

- Build awesome tools

Autonomous teams on multiple continents

Support different use cases

- Develop, test, integrate with other teams

Some history: spvirtenv

KVM hypervisors

DNS as database

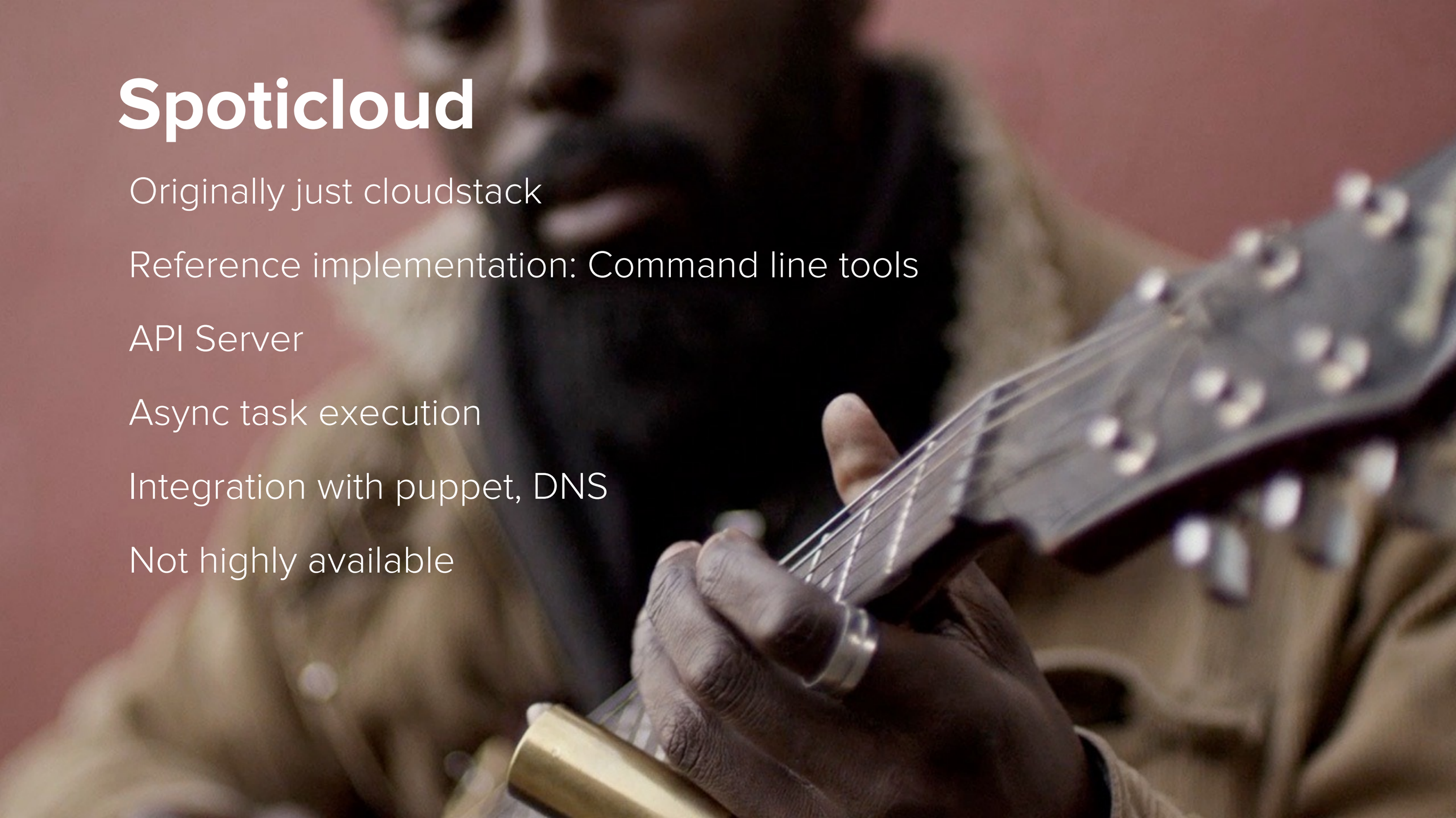
Warning: Bad idea!

Everything in the client

Access everywhere

No error handling

Created in 2009 when roughly 30 engineers worked at Spotify



Spoticloud

Originally just cloudstack

Reference implementation: Command line tools

API Server

Async task execution

Integration with puppet, DNS

Not highly available

ssh key authentication

ssh-agent

Challenge/Response over http

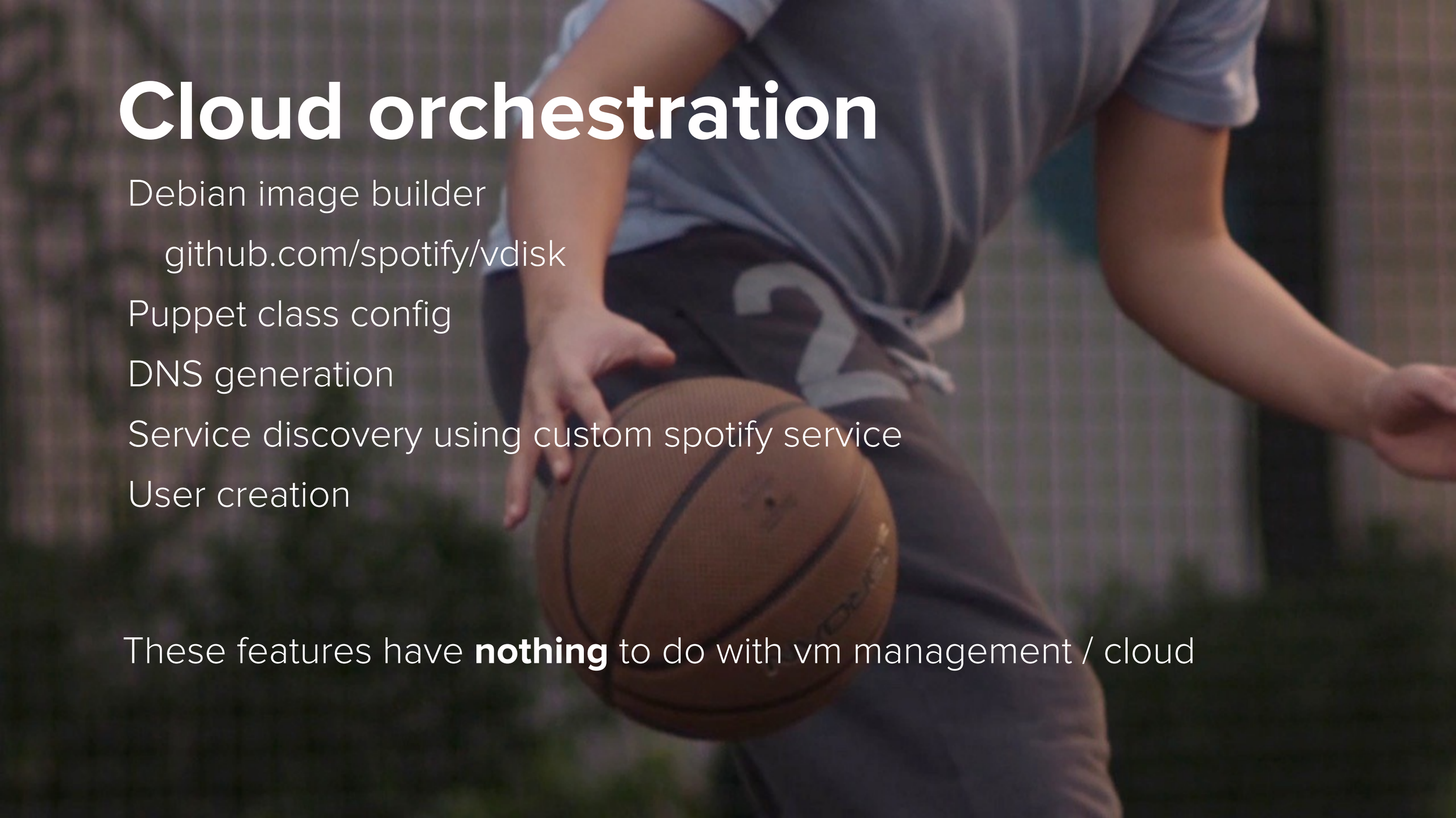
Token gives access

LDAP key & group

github.com/spotify/crtauth



We did/do not want to run a full blown CA/PKI solution



Cloud orchestration

Debian image builder

github.com/spotify/vdisk

Puppet class config

DNS generation

Service discovery using custom spotify service

User creation

These features have **nothing** to do with vm management / cloud

Experiences with Cloudstack

Has served us well

Moving fast

Lots of features you don’t use

Still an appliance mindset

Feedback could be improved

Spotify: trouble scaling the hardware backend of cloudstack

Cloudstack: Running cloudstack is more difficult than it should be

Enter AWS

Mostly solid operation

- Sometimes it takes a while

API tools are surprisingly clunky

Networking has been the main pain point

Getting Amazon engineers on site has been valuable

Resource management

Removing friction makes it easier to spend

Expiring machines

- Creating machines is a very easy operation
- Instances die

Email cost summary

- **“Spending is fine, wasting is not”**

Dashboard



Asynchronous execution

A requirement for robustness

RabbitMQ using Celery

Alternatives?

Gradually improving failures

Asynchronous failure modes are “different”

Providing users with relevant feedback on asynchronous execution is “different”

Start early!

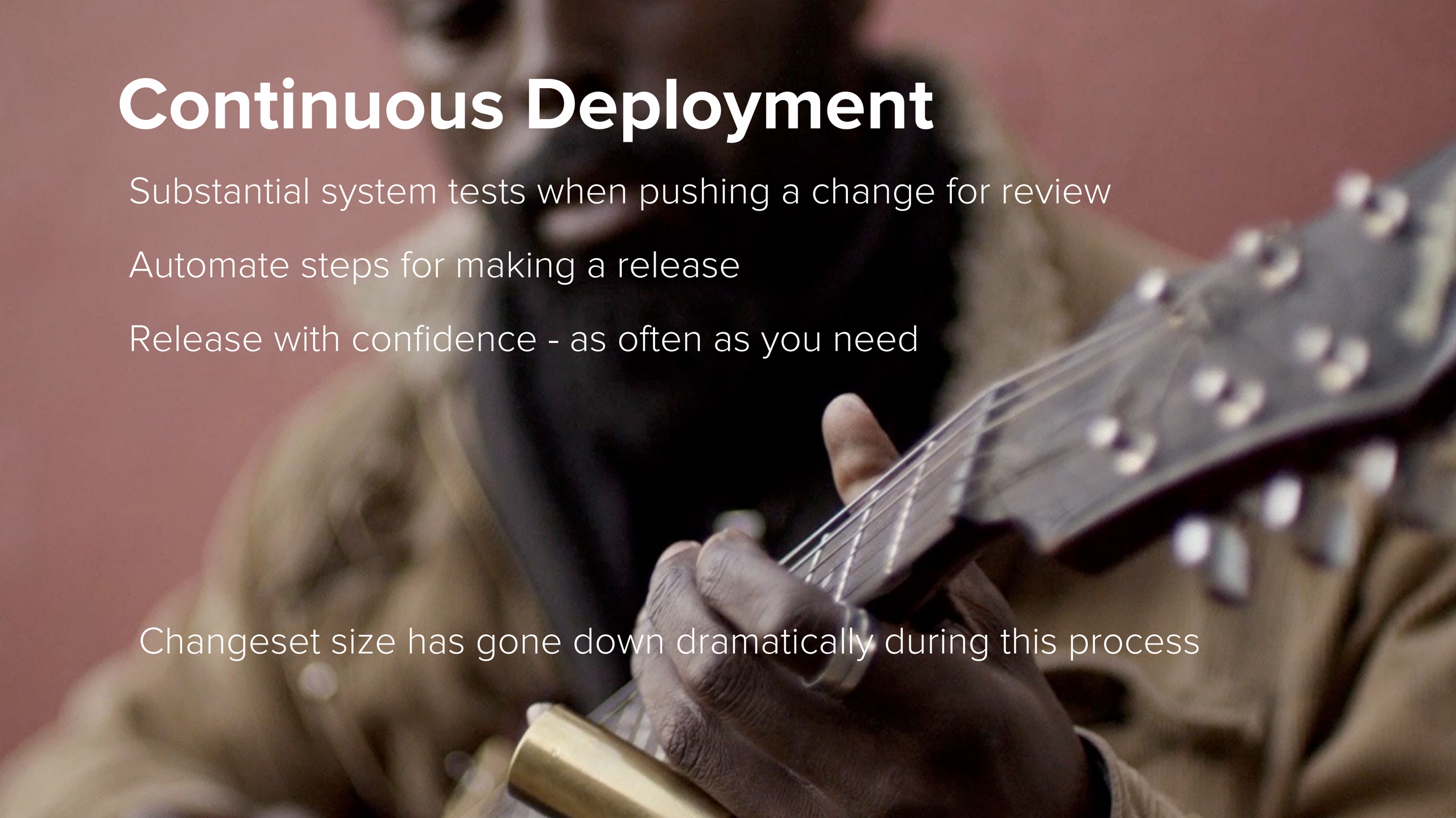


Continuous Deployment

Substantial system tests when pushing a change for review

Automate steps for making a release

Release with confidence - as often as you need



Changeset size has gone down dramatically during this process



Looking ahead

Keeping up with production!

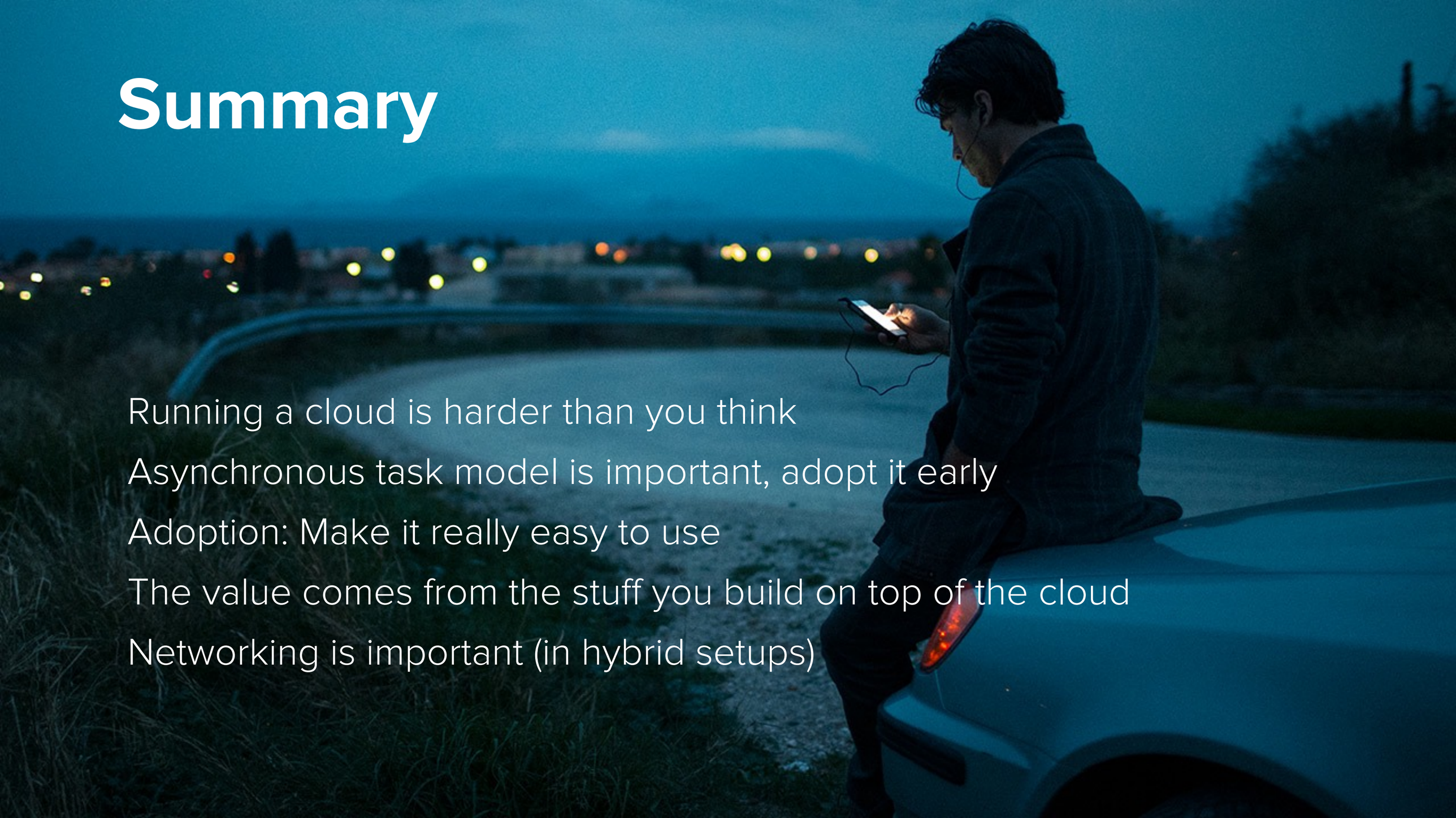
On demand build servers

Integration testing

More clouds?

Containers

Summary



Running a cloud is harder than you think

Asynchronous task model is important, adopt it early

Adoption: Make it really easy to use

The value comes from the stuff you build on top of the cloud

Networking is important (in hybrid setups)

Thank you!

Noa Resare
noa@spotify.com
@blippie

Ramon van Alteren
ramon@spotify.com
@ramonvanalteren



September 2013

Yes, we're hiring

spotify.com/jobs

Spotify

November 2013