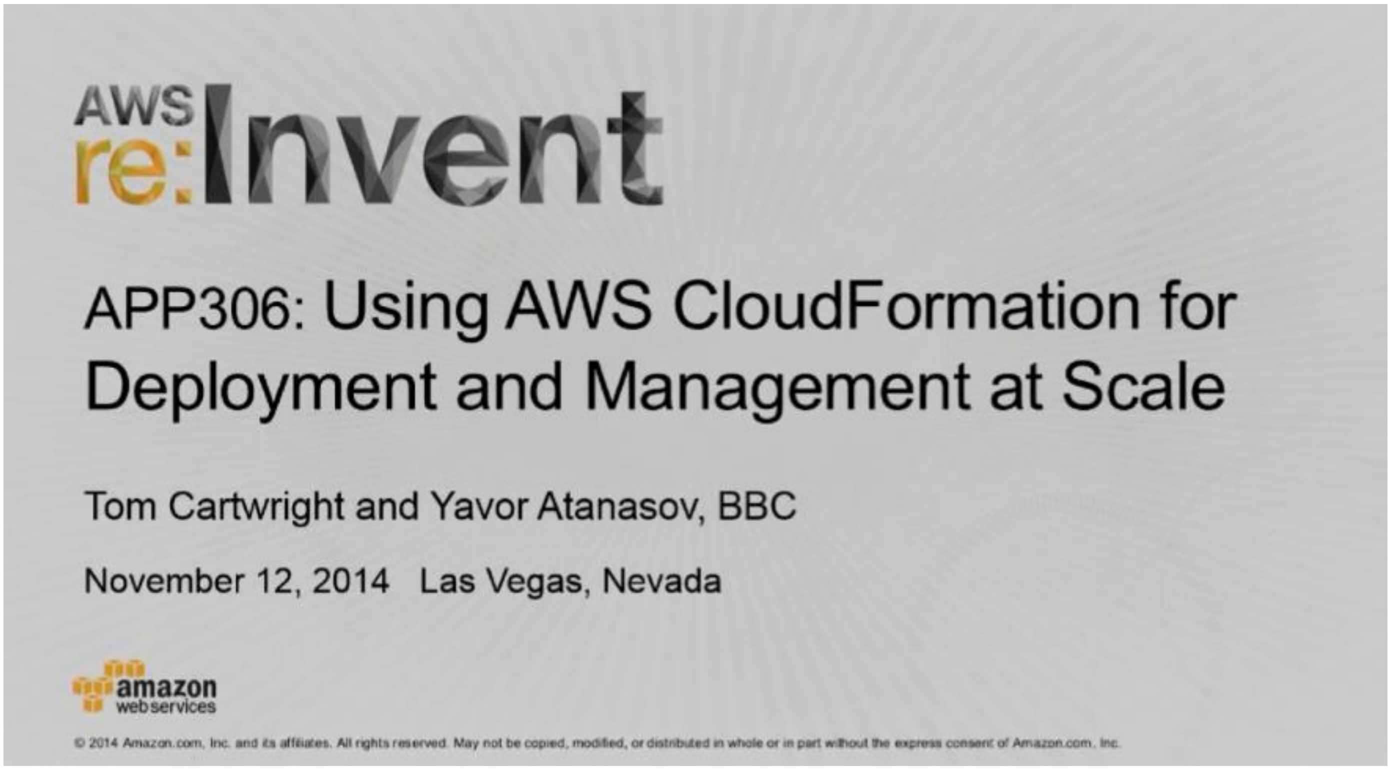


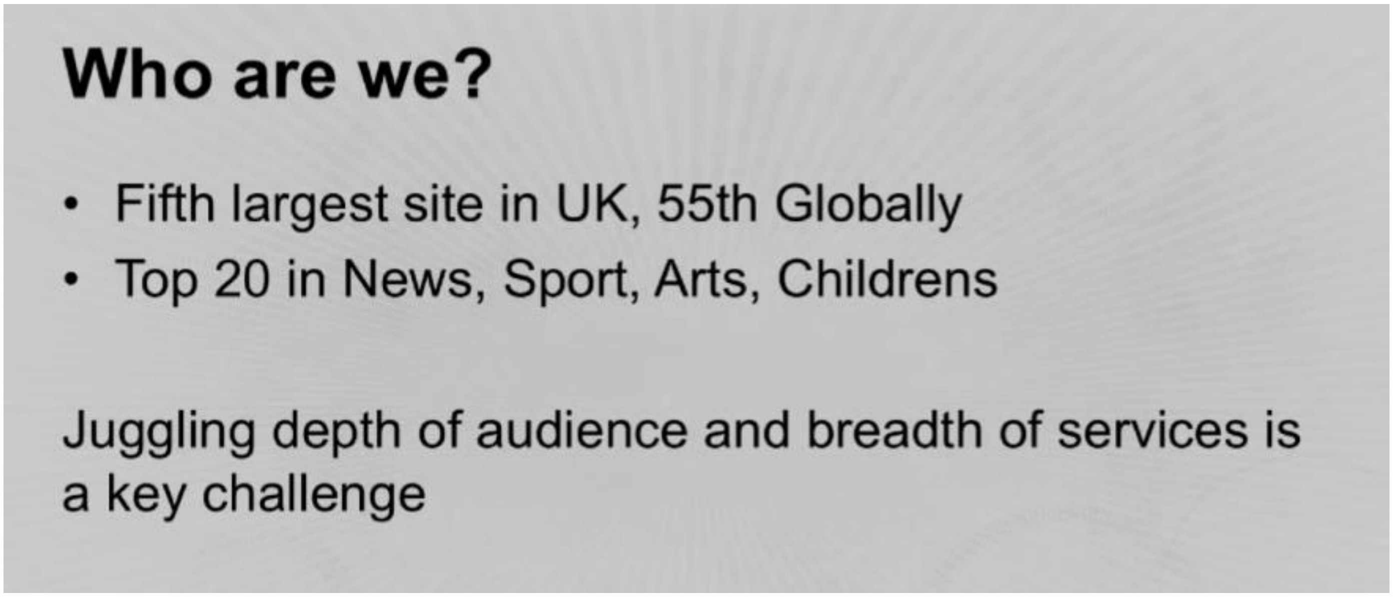
APP306: Using AWS CloudFormation for Deployment and Management at Scale

Tom Cartwright and Yavor Atanasov, BBC

November 12, 2014 Las Vegas, Nevada

© 2014 Amazon.com, Inc. and its affiliates. All rights reserved. May not be copied, modified, or distributed in whole or in part without the express consent of Amazon.com, Inc.

With AWS CloudFormation you can model, provision, and update the full breadth of AWS resources. You can manage anything from a single Amazon EC2 instance to a multi-tier application. The British Broadcasting Corporation (BBC) uses AWS and CloudFormation to help deliver a range of services, including BBC iPlayer. Learn straight from the BBC team on how they developed these services with a multitude of AWS features and how they operate at scale. Get insight into the tooling and best practices developed by the BBC team and how they used CloudFormation to form an end-to-end deployment and management pipeline. If you are new to AWS CloudFormation, get up to speed for this session by completing the Working with CloudFormation lab in the self-paced Labs Lounge.



Who are we?

- Fifth largest site in UK, 55th Globally
- Top 20 in News, Sport, Arts, Childrens

Juggling depth of audience and breadth of services is a key challenge

What do our services do?

Deploy at scale

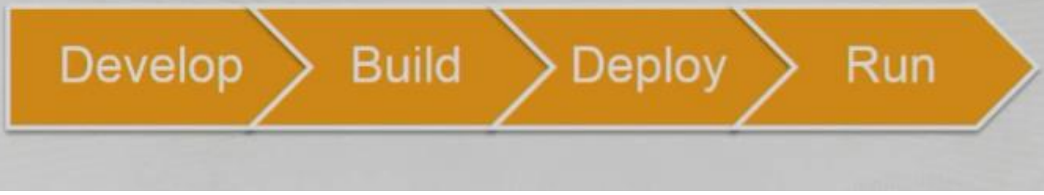
- > 300 deployments per day
- 60,000 deployments in first 18 months

Deploy robustly

- All key video transcoding and packaging for BBC iPlayer
- Pipeline delivering election results to BBC News
- Live text for BBC Sport events

How are Yavor and I involved?

We build tools for the full development lifecycle



Develop

Build

Deploy

Run

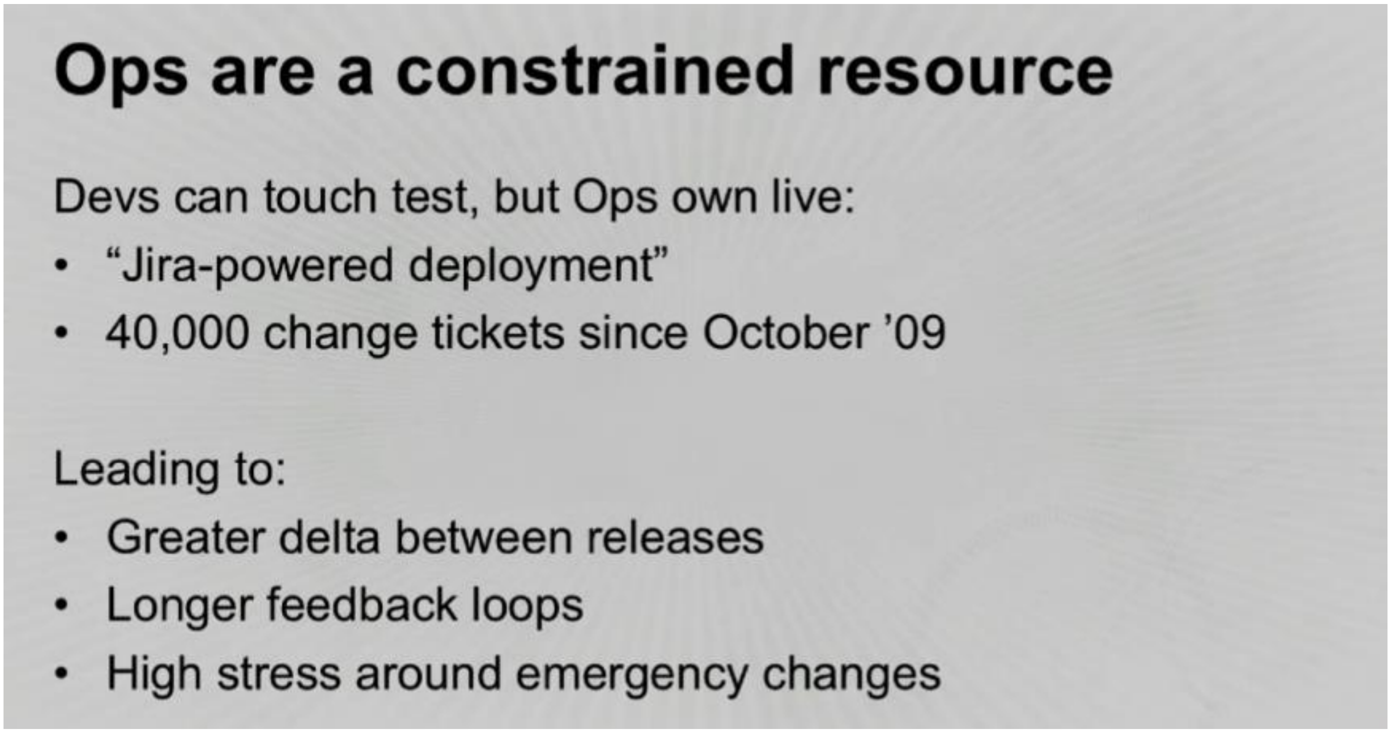
And what are we going to talk about?

- **Part One** – Where did we come from and how did we get where we are?
- **Part Two** – What have we built and how do we use AWS CloudFormation to keep it running?

The beginning

The beginning — 2012

- Olympics dominating our planning and capacity
- On-premises platforms running key BBC Online properties
- Hard to get focus on other projects



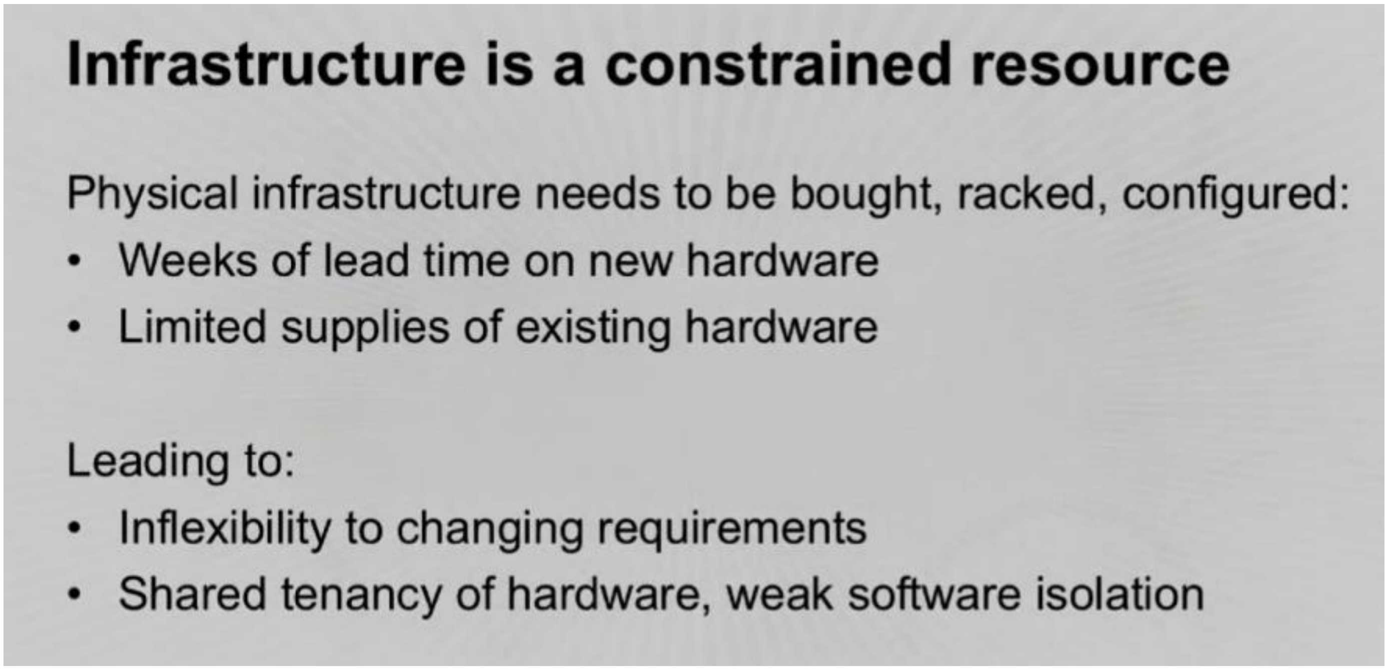
Ops are a constrained resource

Devs can touch test, but Ops own live:

- “Jira-powered deployment”
- 40,000 change tickets since October ’09

Leading to:

- Greater delta between releases
- Longer feedback loops
- High stress around emergency changes



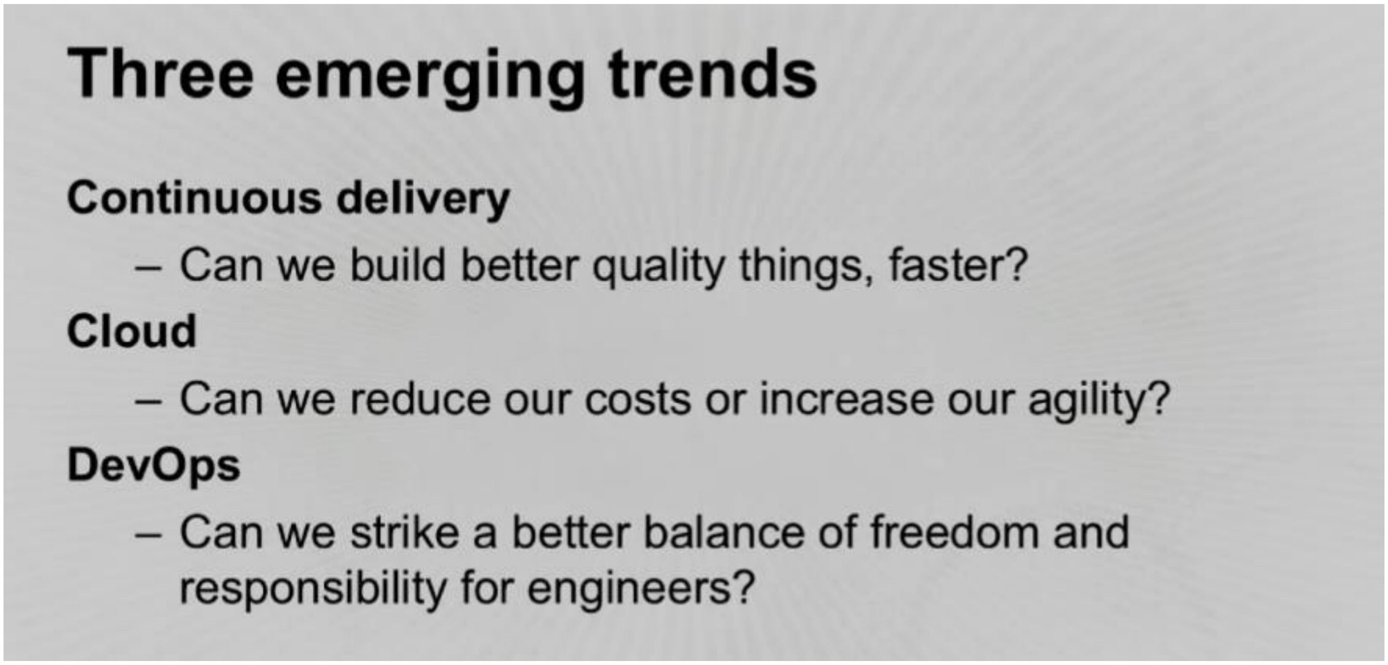
Infrastructure is a constrained resource

Physical infrastructure needs to be bought, racked, configured:

- Weeks of lead time on new hardware
- Limited supplies of existing hardware

Leading to:

- Inflexibility to changing requirements
- Shared tenancy of hardware, weak software isolation



Three emerging trends

Continuous delivery

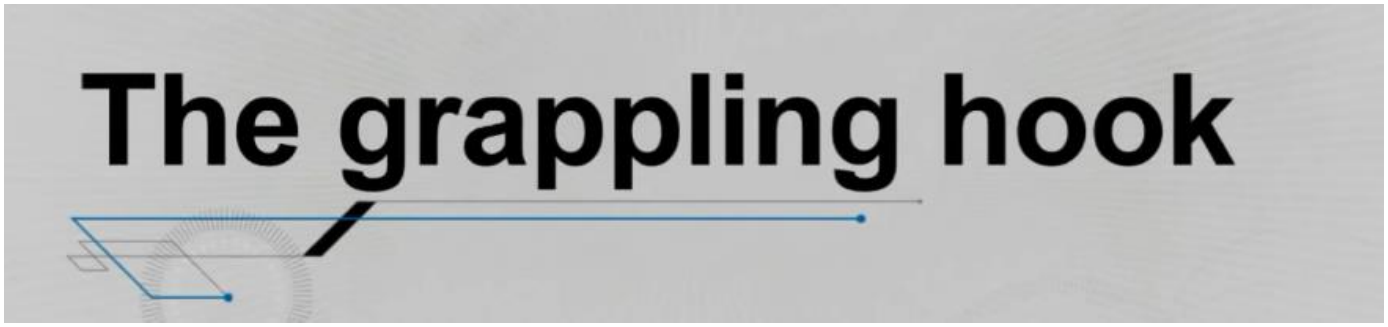
– Can we build better quality things, faster?

Cloud

– Can we reduce our costs or increase our agility?

DevOps

– Can we strike a better balance of freedom and responsibility for engineers?



The grappling hook

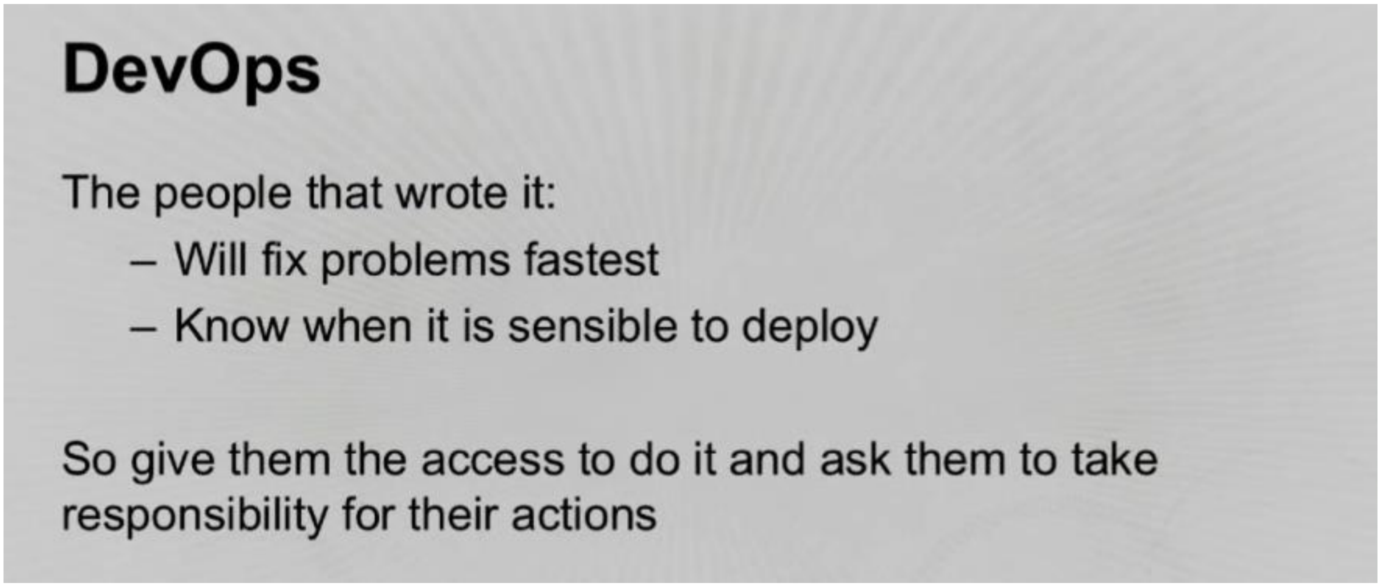
The grappling hook

- Take two teams: one product, one platform
- Product team takes advantage of features as they become available from platform and feeds requirements in
- Platform team builds features based on need but looks to make them scale to many users
- Get the learning in software, not slideware

Continuous delivery

- Automate everything
- Keep *everything* in source control
- Build your binaries once
- Use the same mechanism to deploy to all environments
- If anything fails, stop the line

Think continuous improvement — direction not position

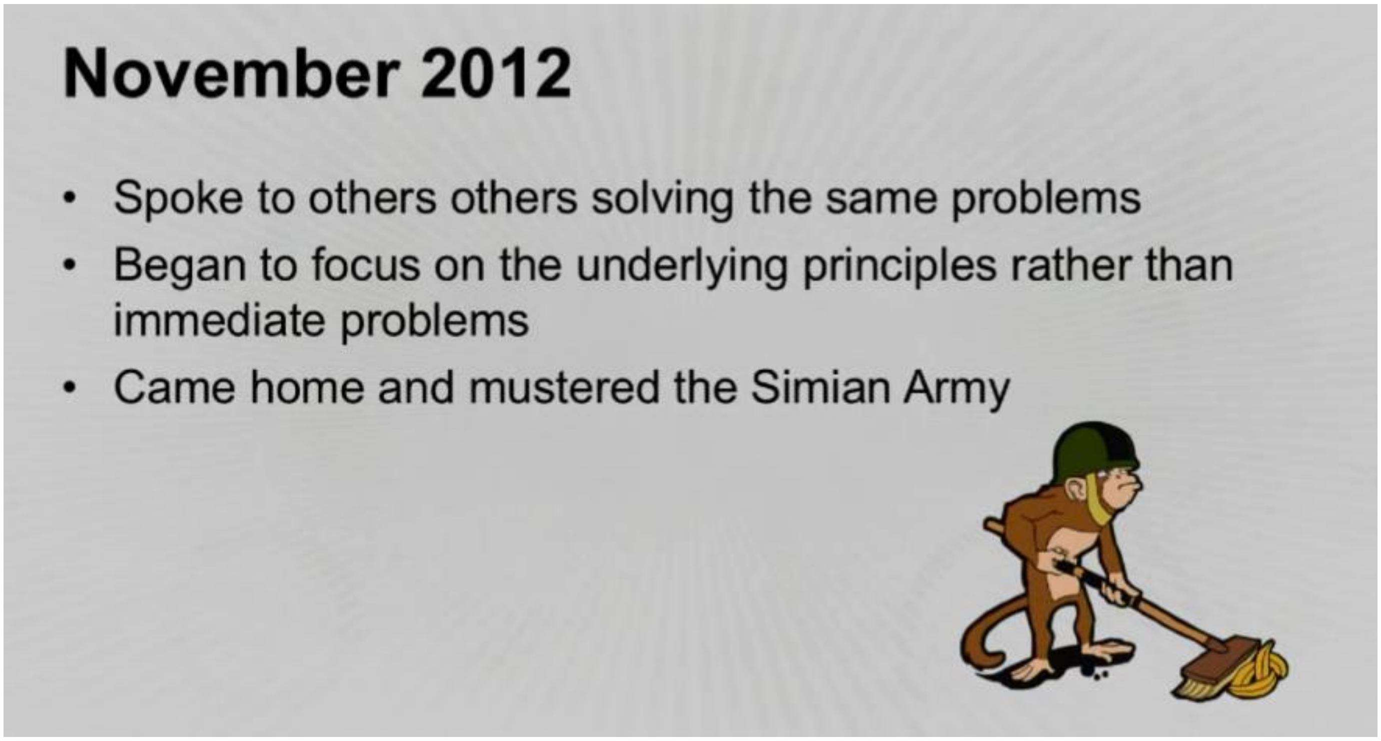


DevOps

The people that wrote it:

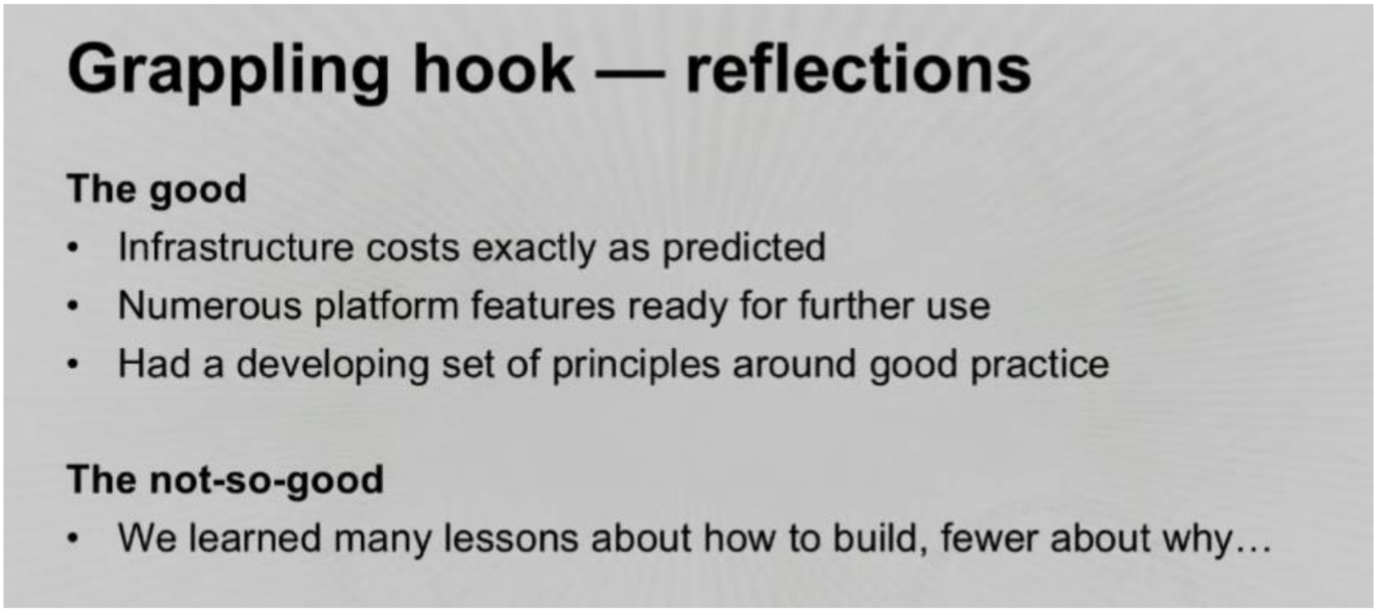
- Will fix problems fastest
- Know when it is sensible to deploy

So give them the access to do it and ask them to take responsibility for their actions



November 2012

- Spoke to others others solving the same problems
- Began to focus on the underlying principles rather than immediate problems
- Came home and mustered the Simian Army



Grappling hook — reflections

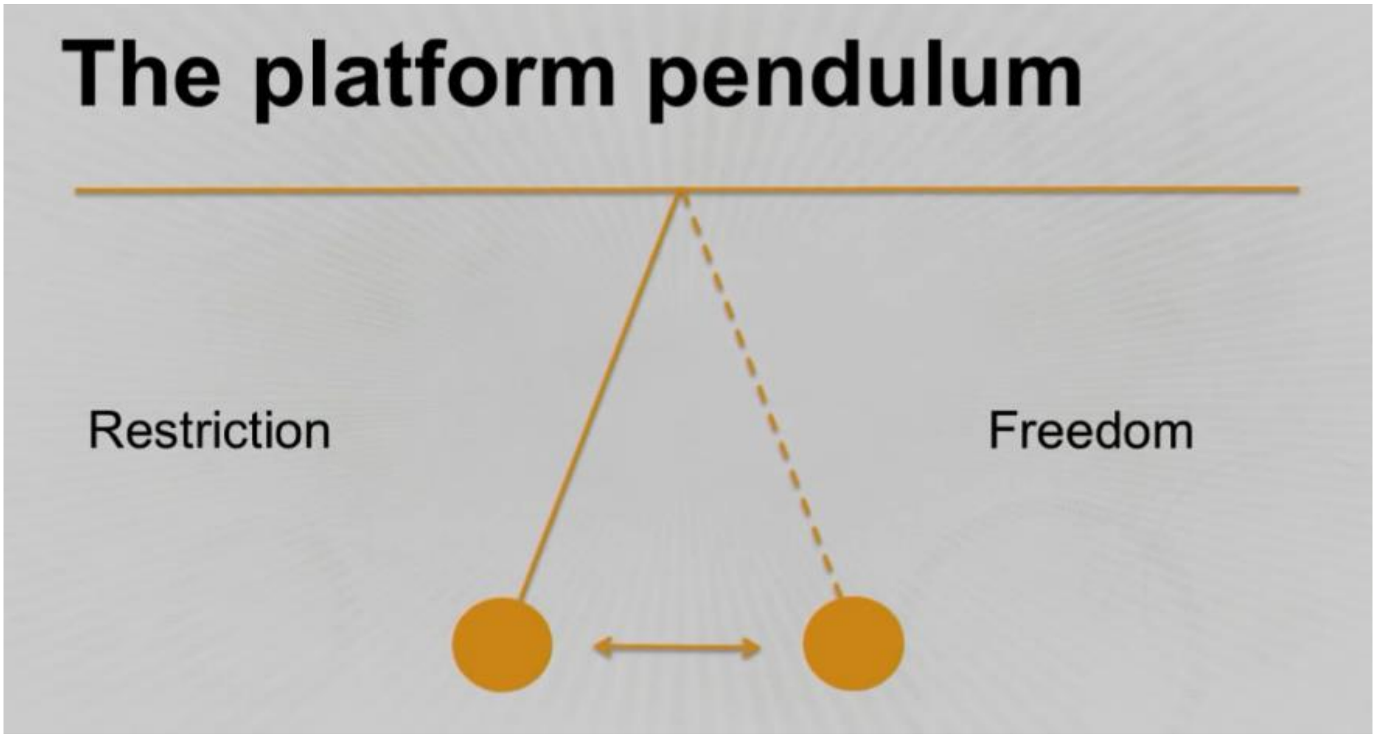
The good

- Infrastructure costs exactly as predicted
- Numerous platform features ready for further use
- Had a developing set of principles around good practice

The not-so-good

- We learned many lessons about how to build, fewer about why…

Storming the tower

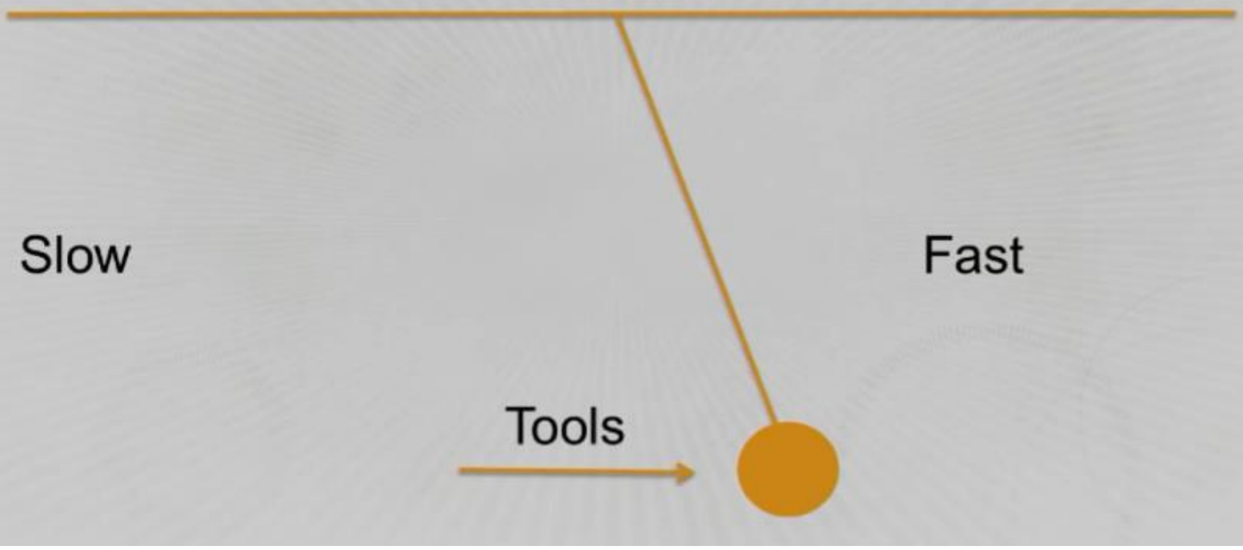


The platform pendulum

Restriction

Freedom

The platform pendulum

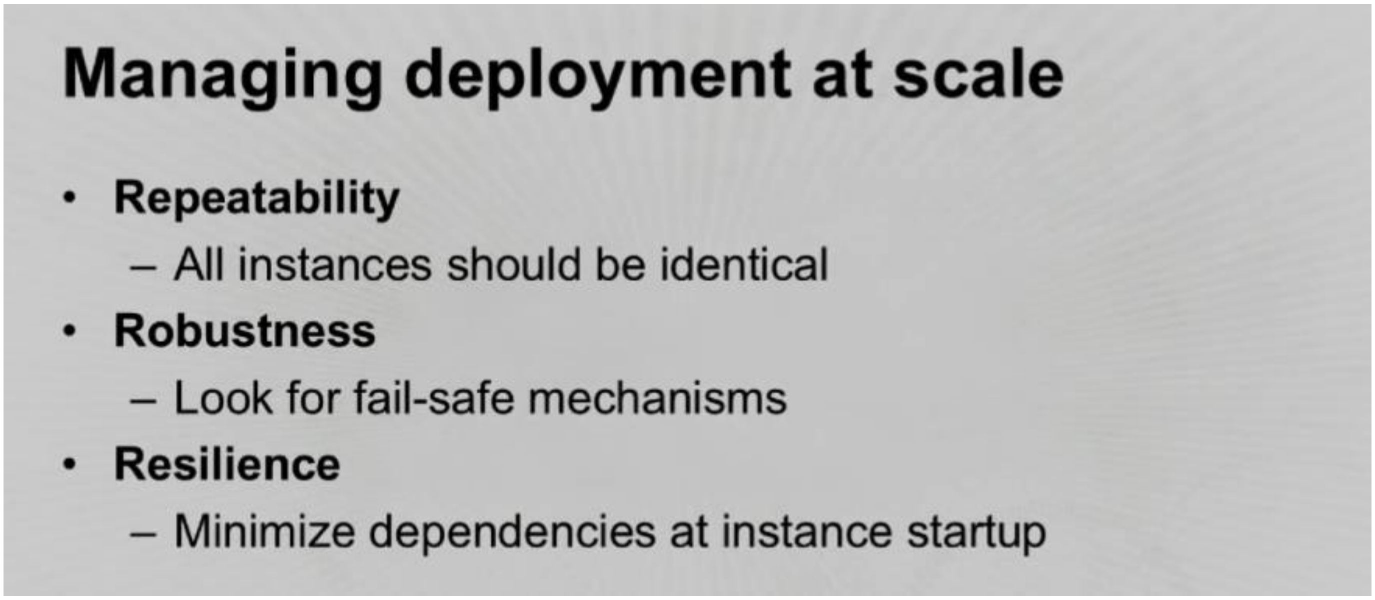


Establishing principles

- Establish strong defaults for the way things get built and create tooling for that
- Assume that there will be use cases where the defaults don’t fit

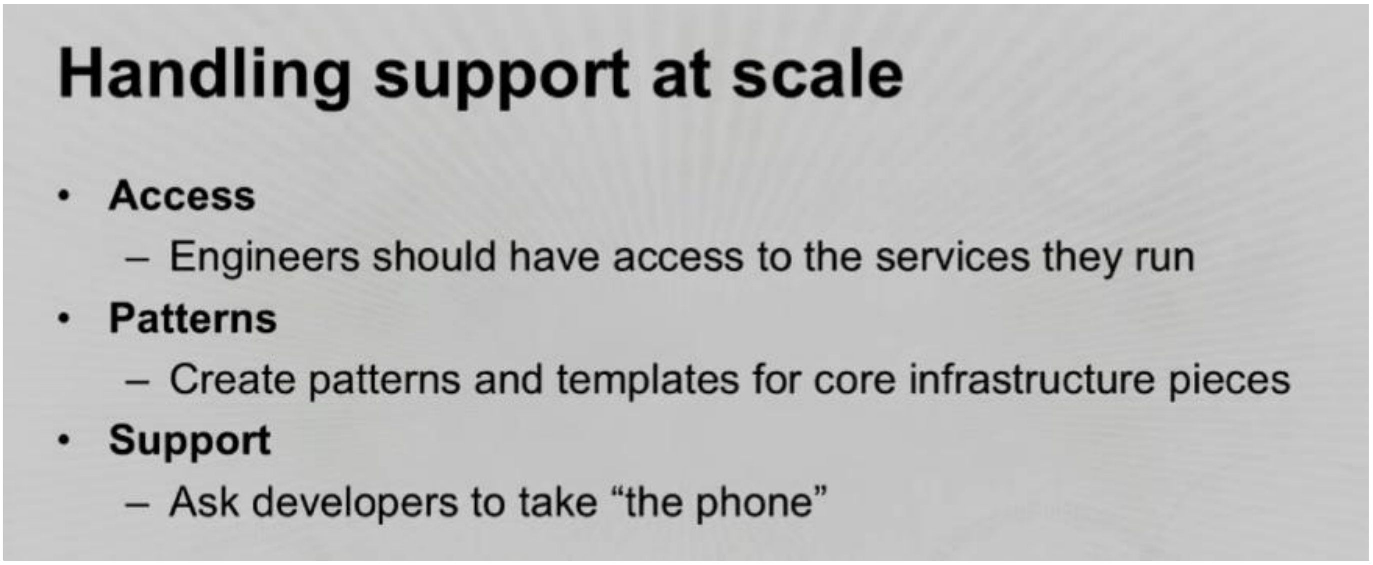
Managing infrastructure at scale

- **Repeatability**
  - Never “spin it up in the console and hope”
- **Flexibility**
  - Teams *are* going to need that obscure service
- **StackOverflow-ability**
  - If there is a well-known way of expressing it in the world, use it



Managing deployment at scale

- **Repeatability**
  - All instances should be identical
- **Robustness**
  - Look for fail-safe mechanisms
- **Resilience**
  - Minimize dependencies at instance startup

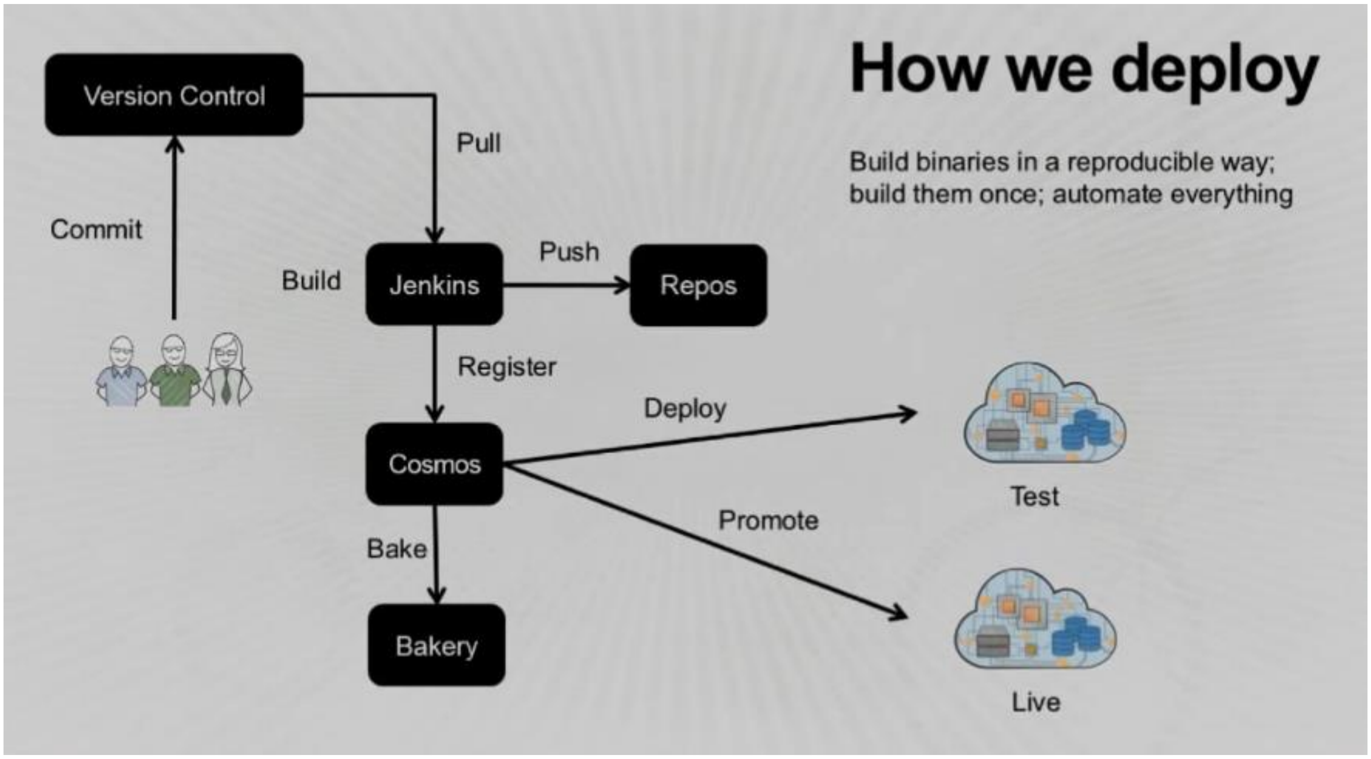


Handling support at scale

- **Access**
  - Engineers should have access to the services they run
- **Patterns**
  - Create patterns and templates for core infrastructure pieces
- **Support**
  - Ask developers to take “the phone”

The rest is just software…

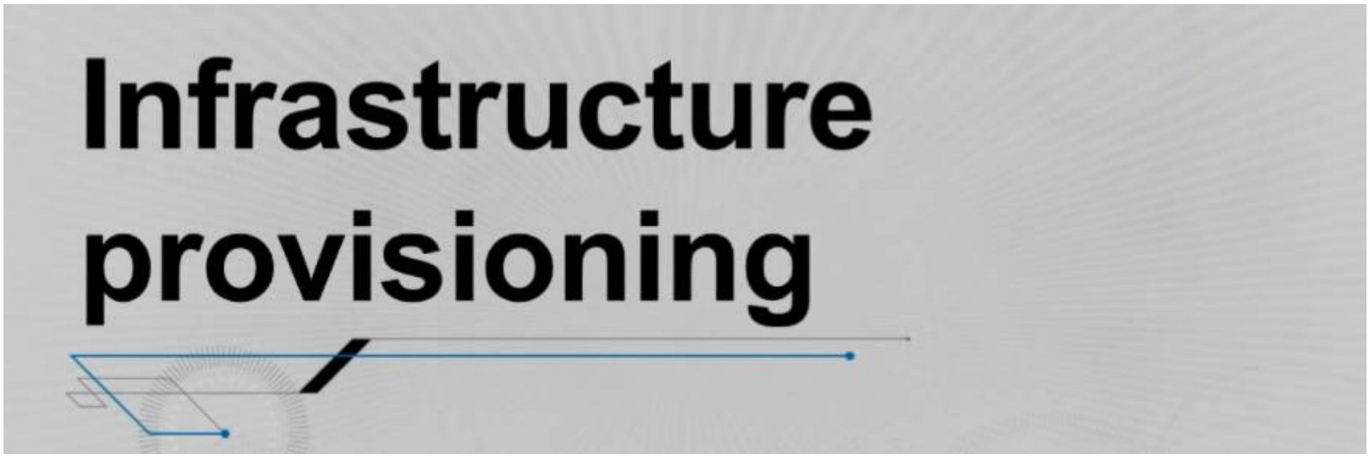
Inside the machine



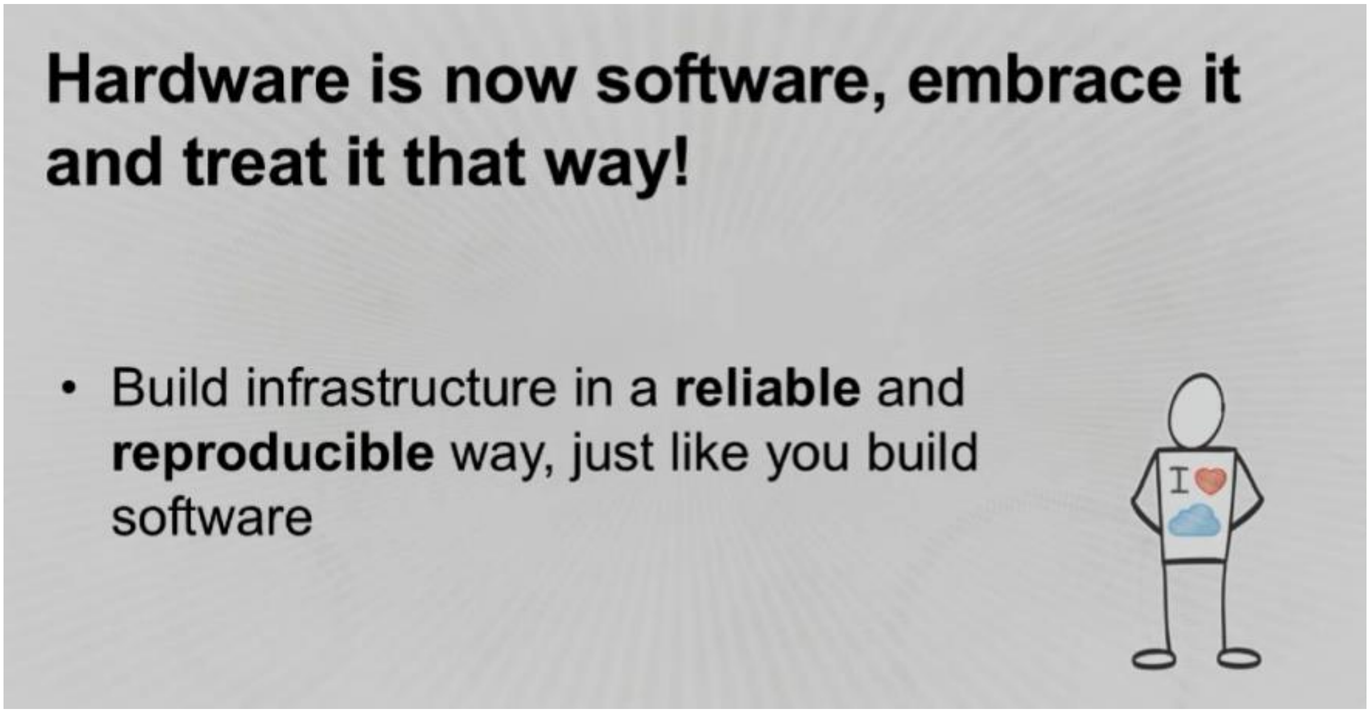
How we deploy

Build binaries in a reproducible way; build them once; automate everything

This is the high-level overview of our deployment platform

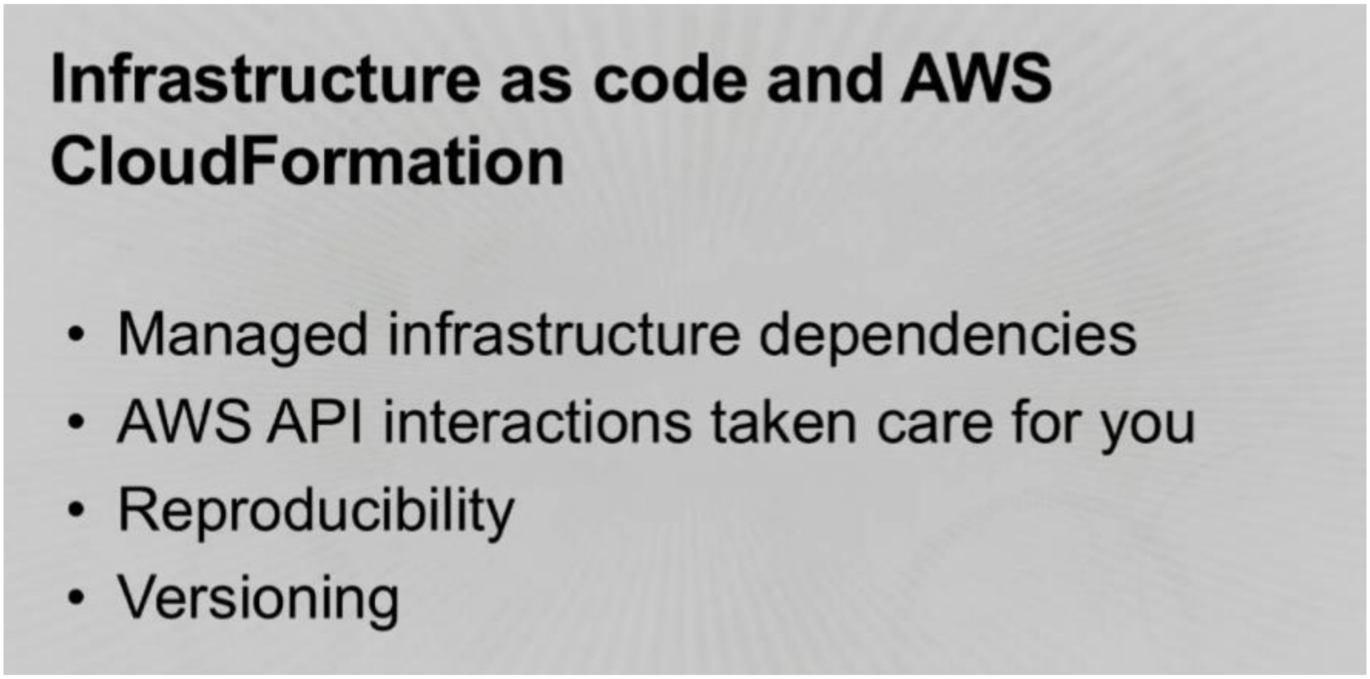


Infrastructure provisioning



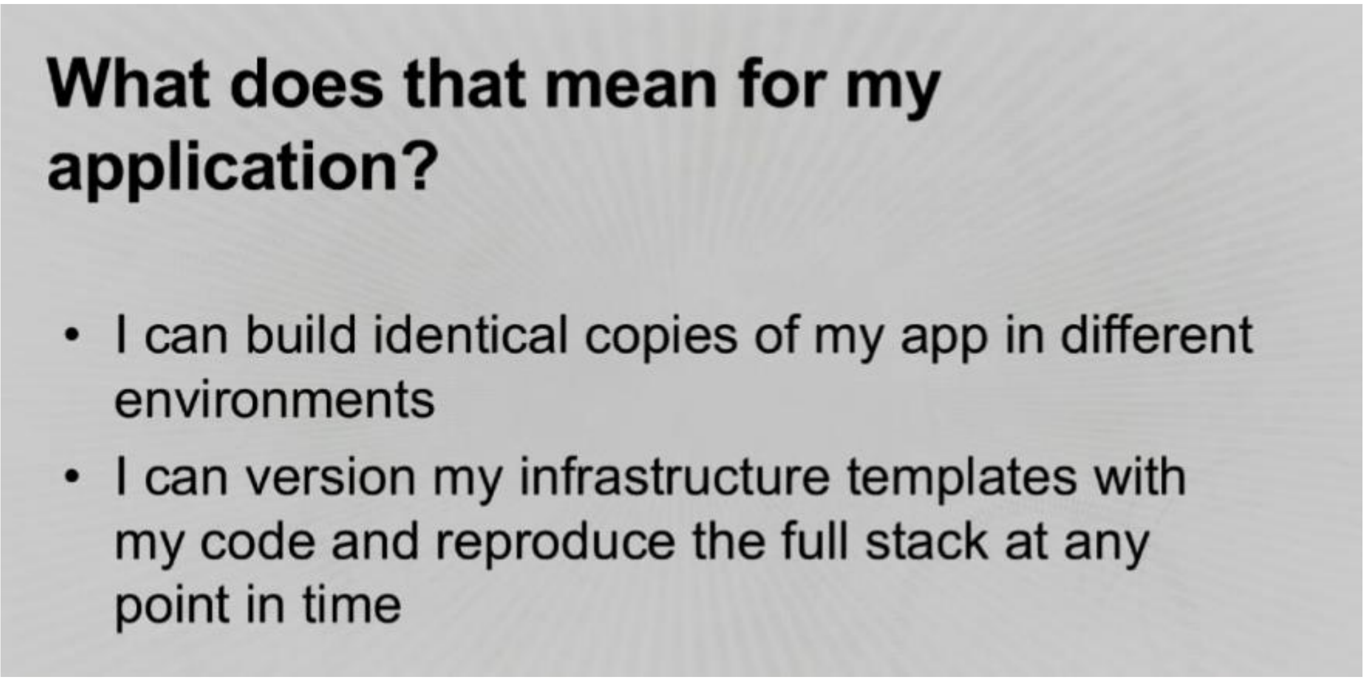
Hardware is now software, embrace it and treat it that way!

- Build infrastructure in a **reliable** and **reproducible** way, just like you build software



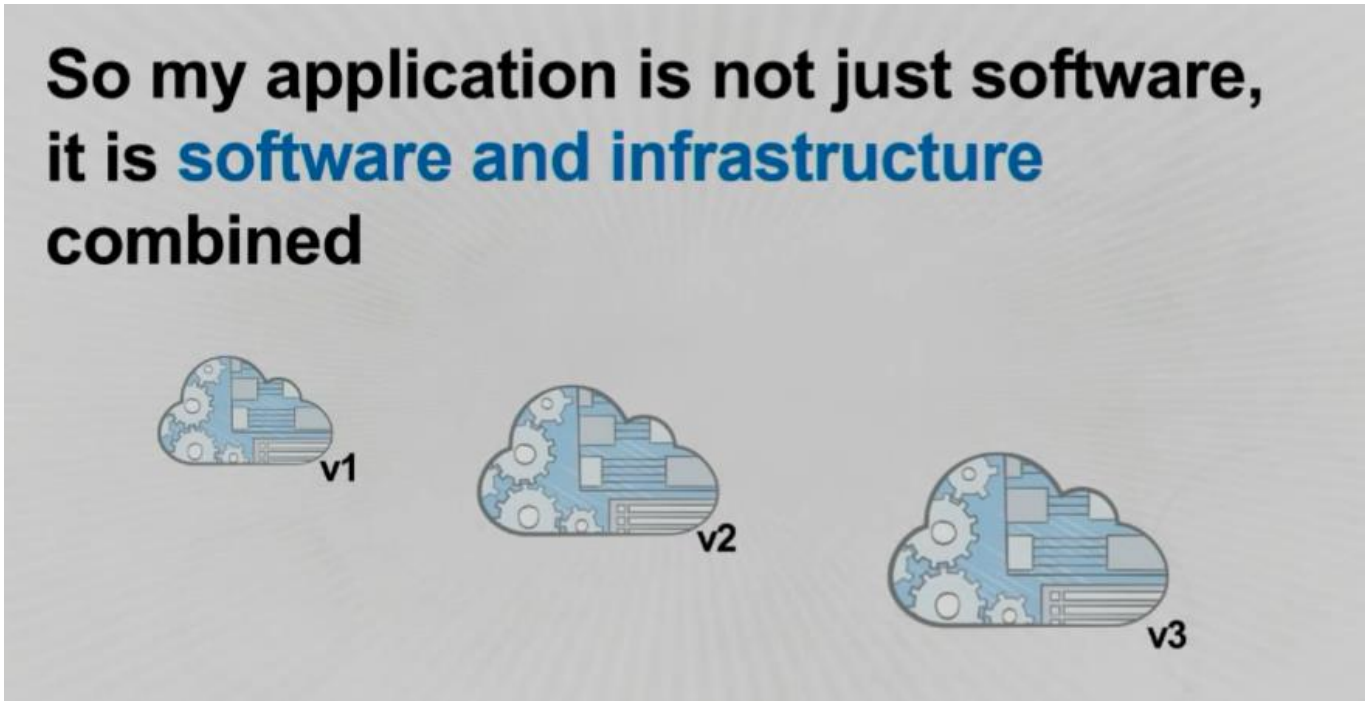
Infrastructure as code and AWS CloudFormation

- Managed infrastructure dependencies
- AWS API interactions taken care for you
- Reproducibility
- Versioning

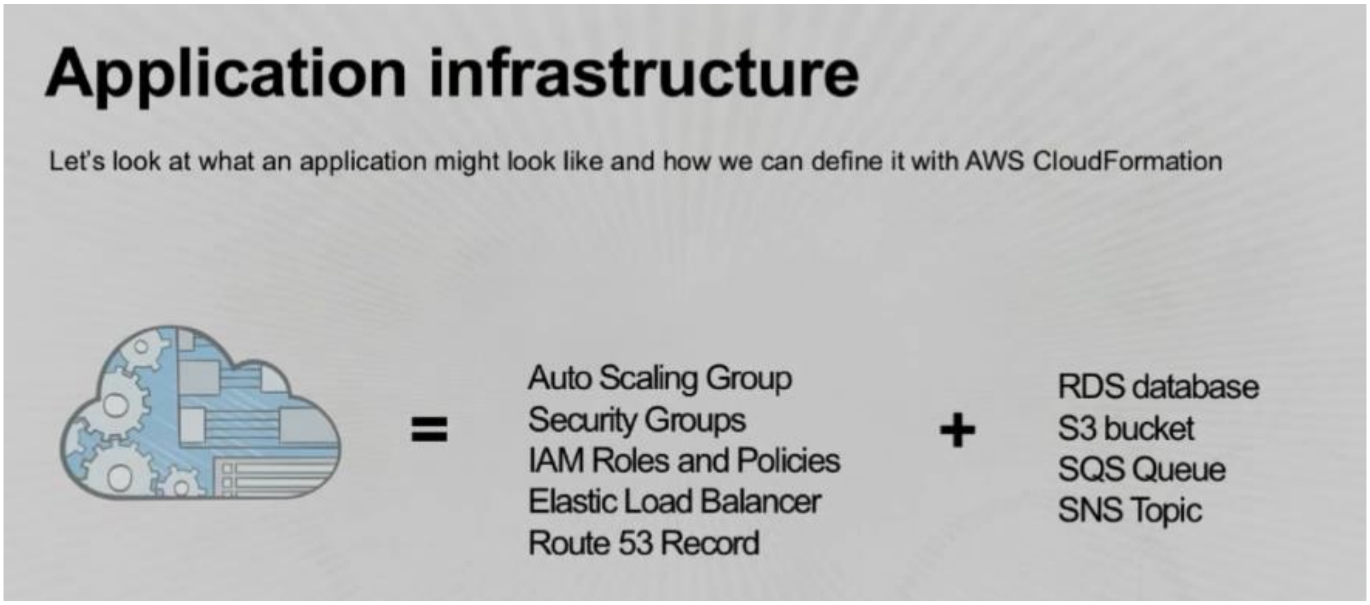


What does that mean for my application?

- I can build identical copies of my app in different environments
- I can version my infrastructure templates with my code and reproduce the full stack at any point in time



So my application is not just software, it is software and infrastructure combined

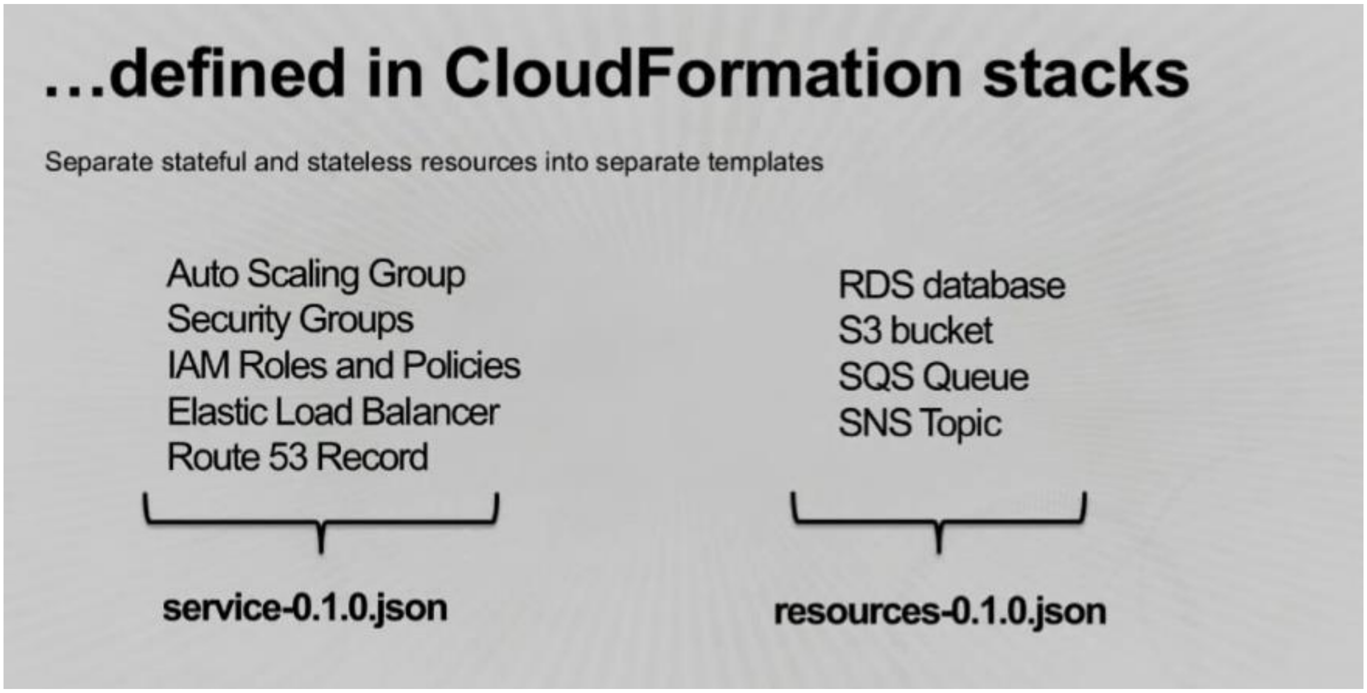


Application infrastructure

Let's look at what an application might look like and how we can define it with AWS CloudFormation

= Auto Scaling Group
Security Groups
IAM Roles and Policies
Elastic Load Balancer
Route 53 Record

+ RDS database
S3 bucket
SQS Queue
SNS Topic



…defined in CloudFormation stacks

Separate stateful and stateless resources into separate templates

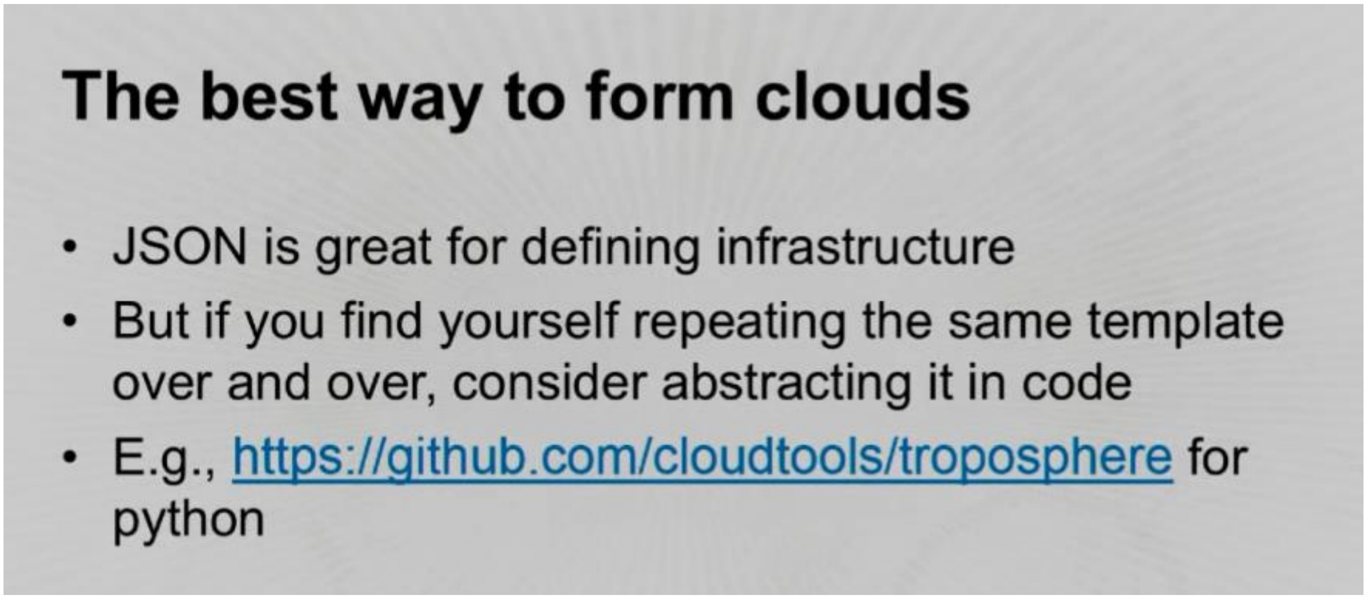
Auto Scaling Group
Security Groups
IAM Roles and Policies
Elastic Load Balancer
Route 53 Record

service-0.1.0.json

RDS database
S3 bucket
SQS Queue
SNS Topic

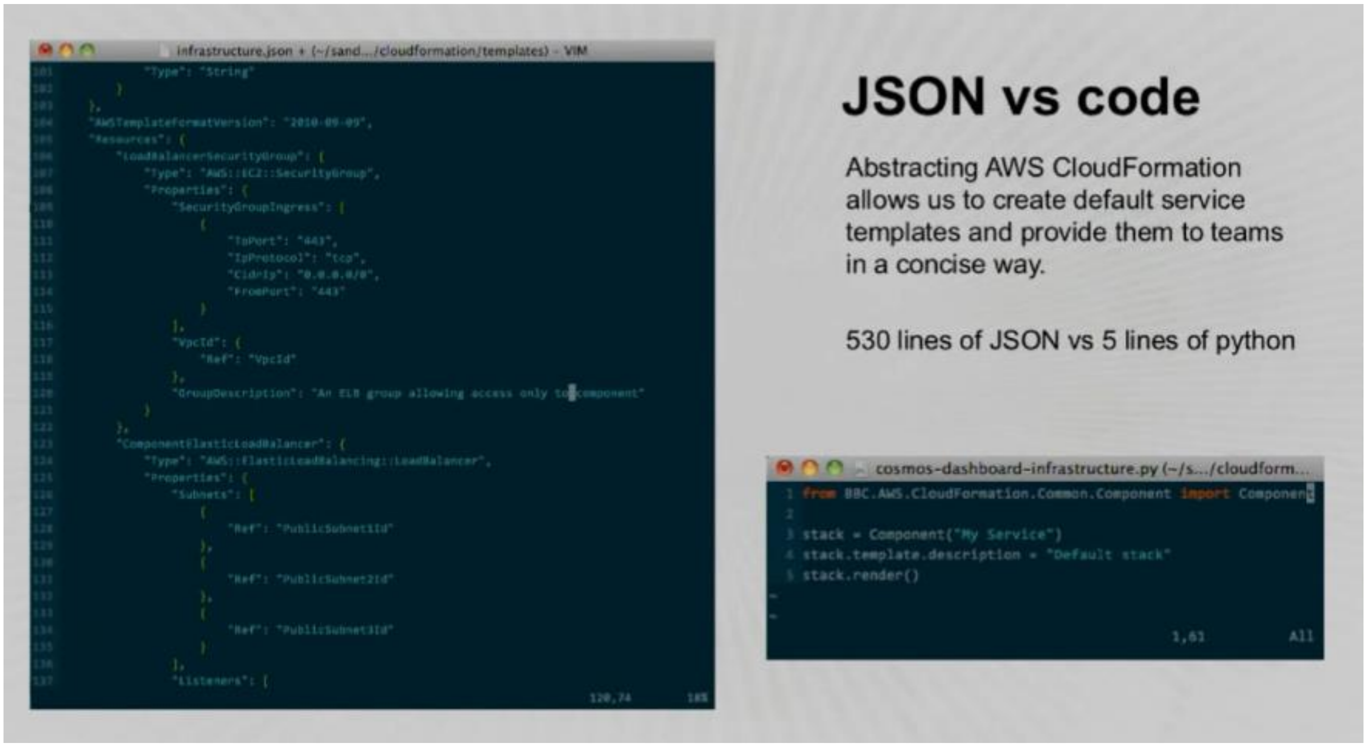
resources-0.1.0.json

They have different management lifecycle regarding how often they change or are revised.



The best way to form clouds

- JSON is great for defining infrastructure
- But if you find yourself repeating the same template over and over, consider abstracting it in code
- E.g., https://github.com/cloudtools/troposphere for python



infrastructure.json + (~/sand.../cloudformation/templates) - VIM

```
            "Type": "String"
        }
    },
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "LoadBalancerSecurityGroup": {
            "Type": "AWS::EC2::SecurityGroup",
            "Properties": {
                "SecurityGroupIngress": [
                    {
                        "ToPort": "443",
                        "IpProtocol": "tcp",
                        "CidrIp": "0.0.0.0/0",
                        "FromPort": "443"
                    }
                ],
                "VpcId": {
                    "Ref": "VpcId"
                },
                "GroupDescription": "An ELB group allowing access only to component"
            }
        },
        "ComponentElasticLoadBalancer": {
            "Type": "AWS::ElasticLoadBalancing::LoadBalancer",
            "Properties": {
                "Subnets": [
                    {
                        "Ref": "PublicSubnet1Id"
                    },
                    {
                        "Ref": "PublicSubnet2Id"
                    },
                    {
                        "Ref": "PublicSubnet3Id"
                    }
                ],
                "Listeners": [
```

JSON vs code

Abstracting AWS CloudFormation allows us to create default service templates and provide them to teams in a concise way.

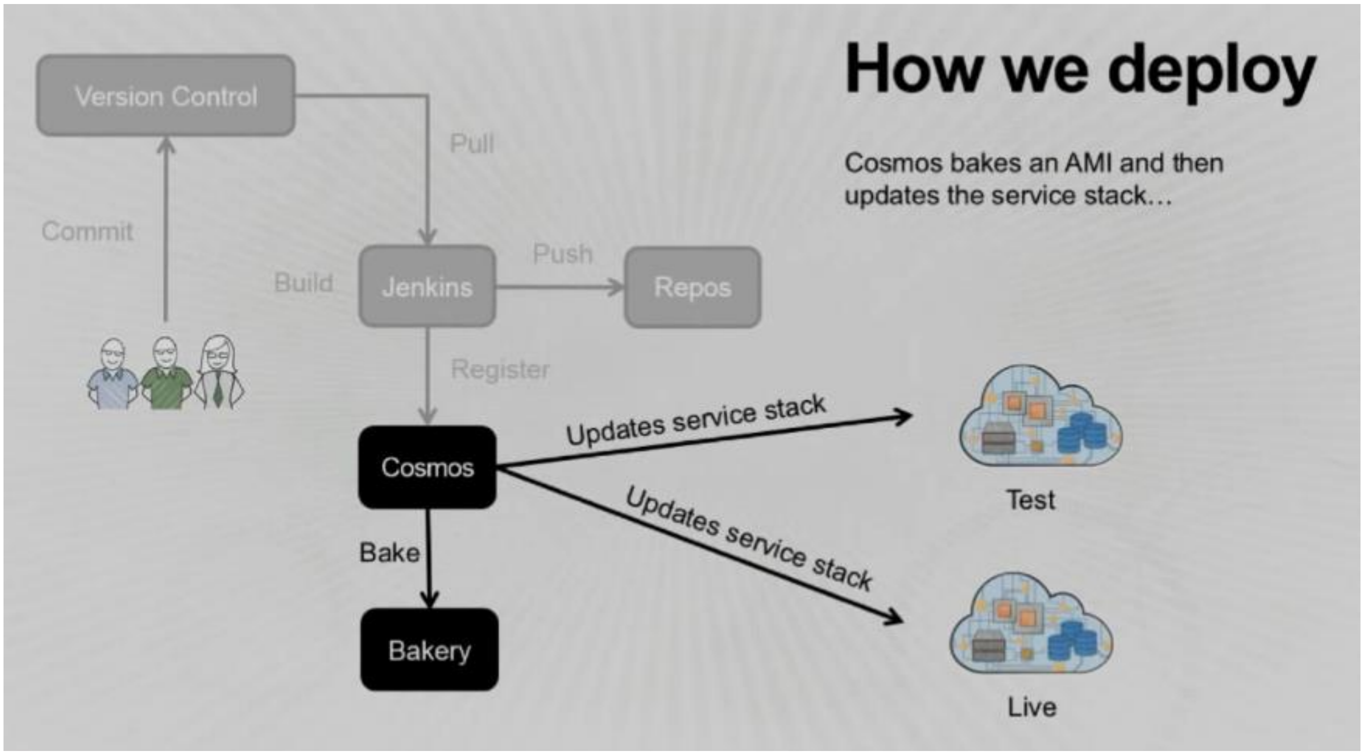
530 lines of JSON vs 5 lines of python

```
from BBC.AWS.CloudFormation.Common.Component import Component

stack = Component("My Service")
stack.template.description = "Default stack"
stack.render()
```

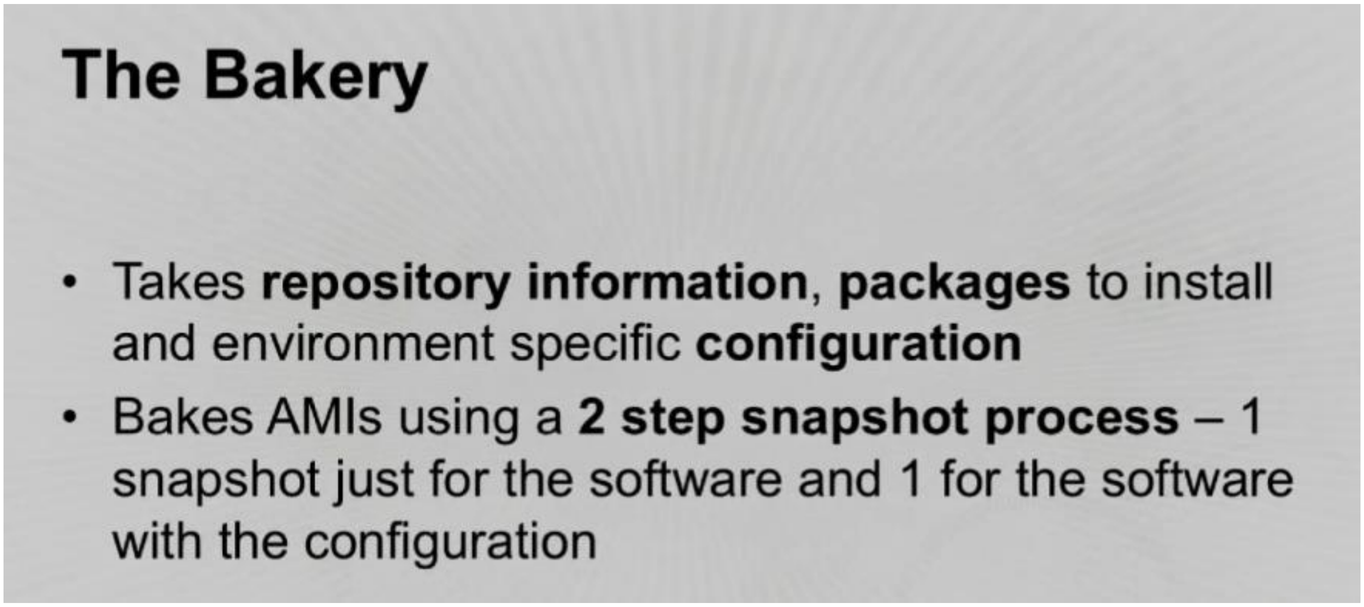
The left side contains a sample template but we then abstracted it into 5 lines on the right

AWS CloudFormation and deployments



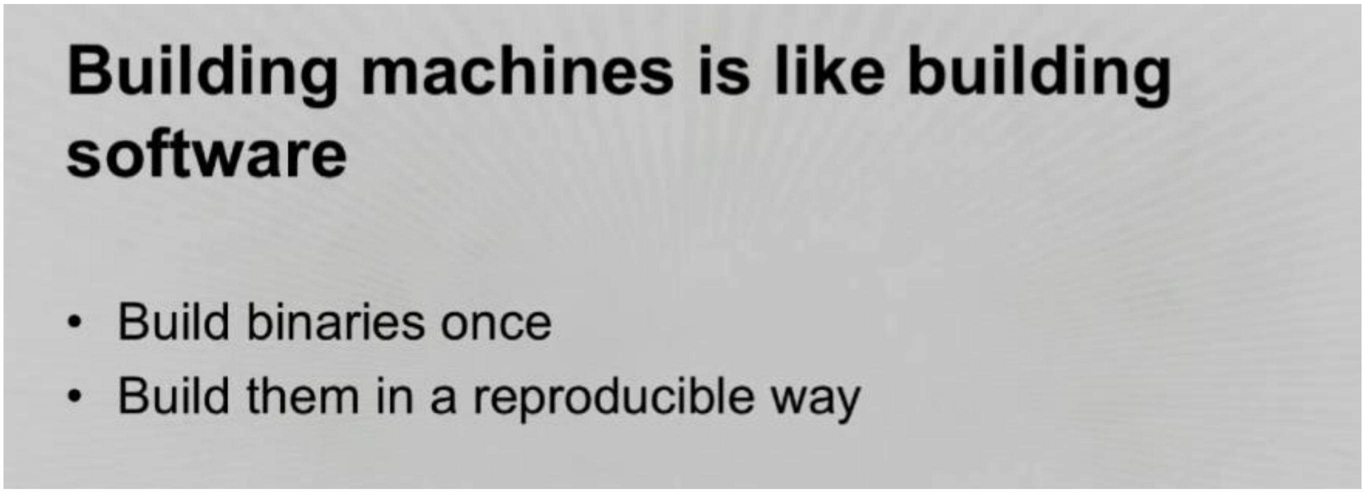
How we deploy

Cosmos bakes an AMI and then updates the service stack…



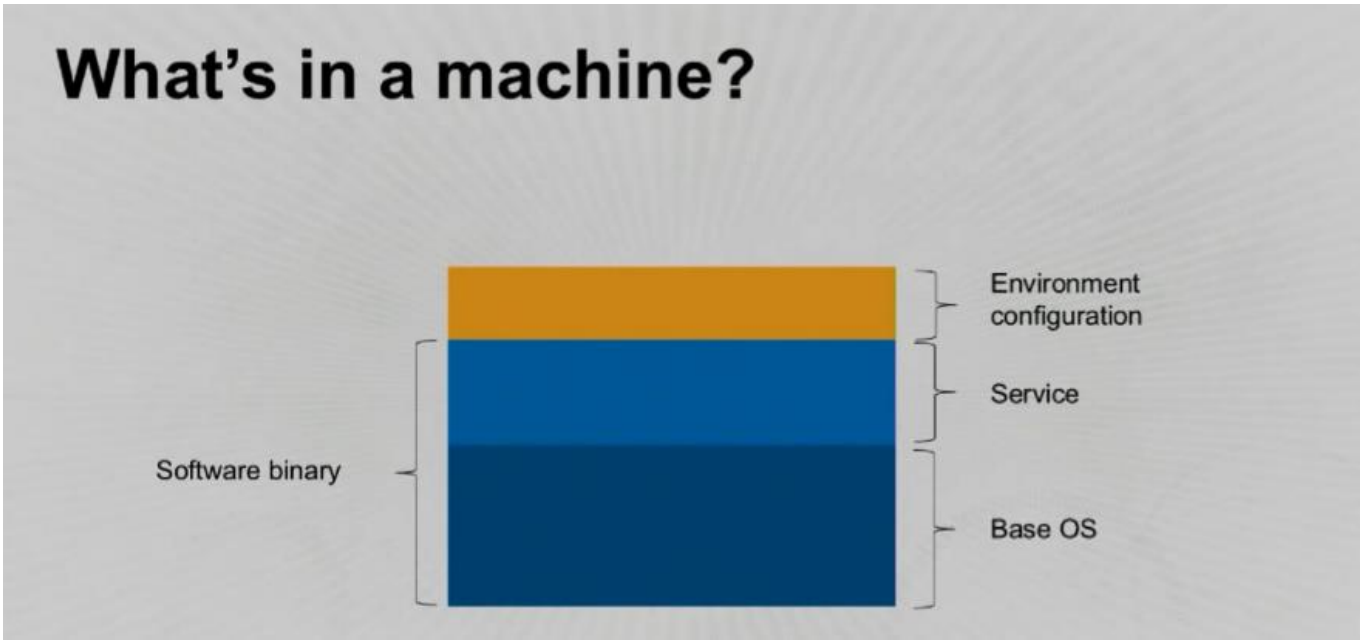
The Bakery

- Takes **repository information**, **packages** to install and environment specific **configuration**
- Bakes AMIs using a **2 step snapshot process** – 1 snapshot just for the software and 1 for the software with the configuration



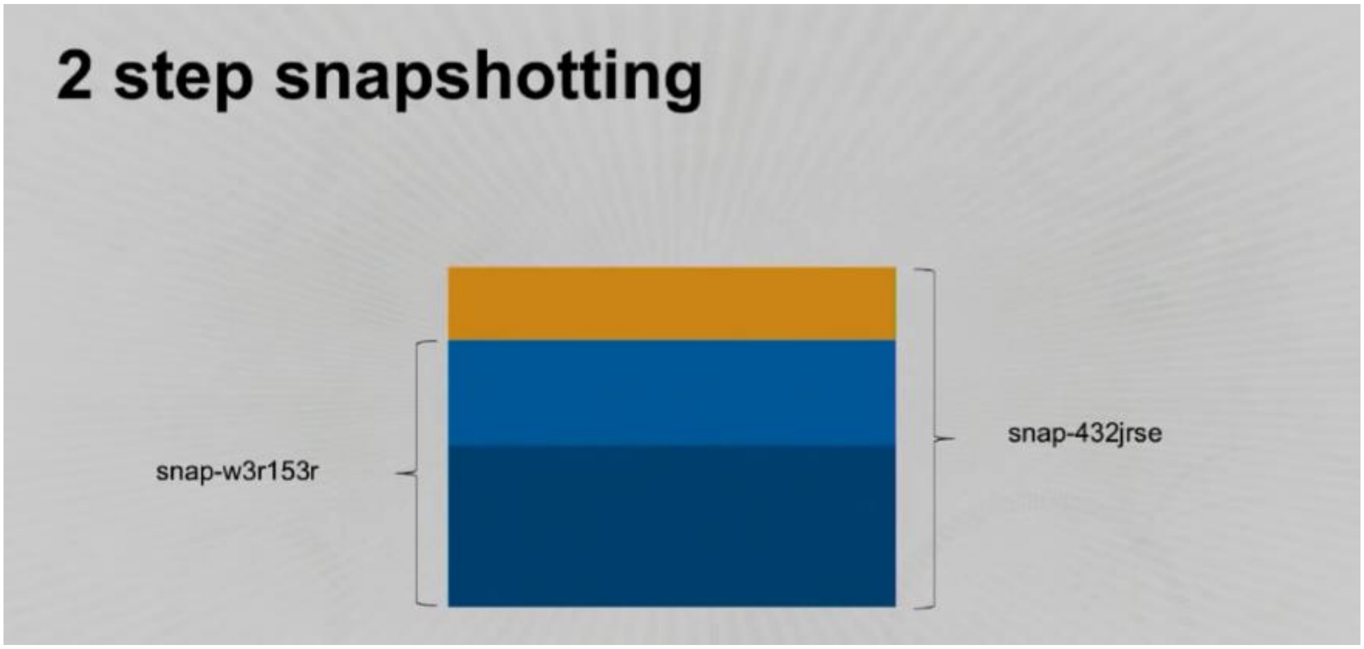
Building machines is like building software

- Build binaries once
- Build them in a reproducible way

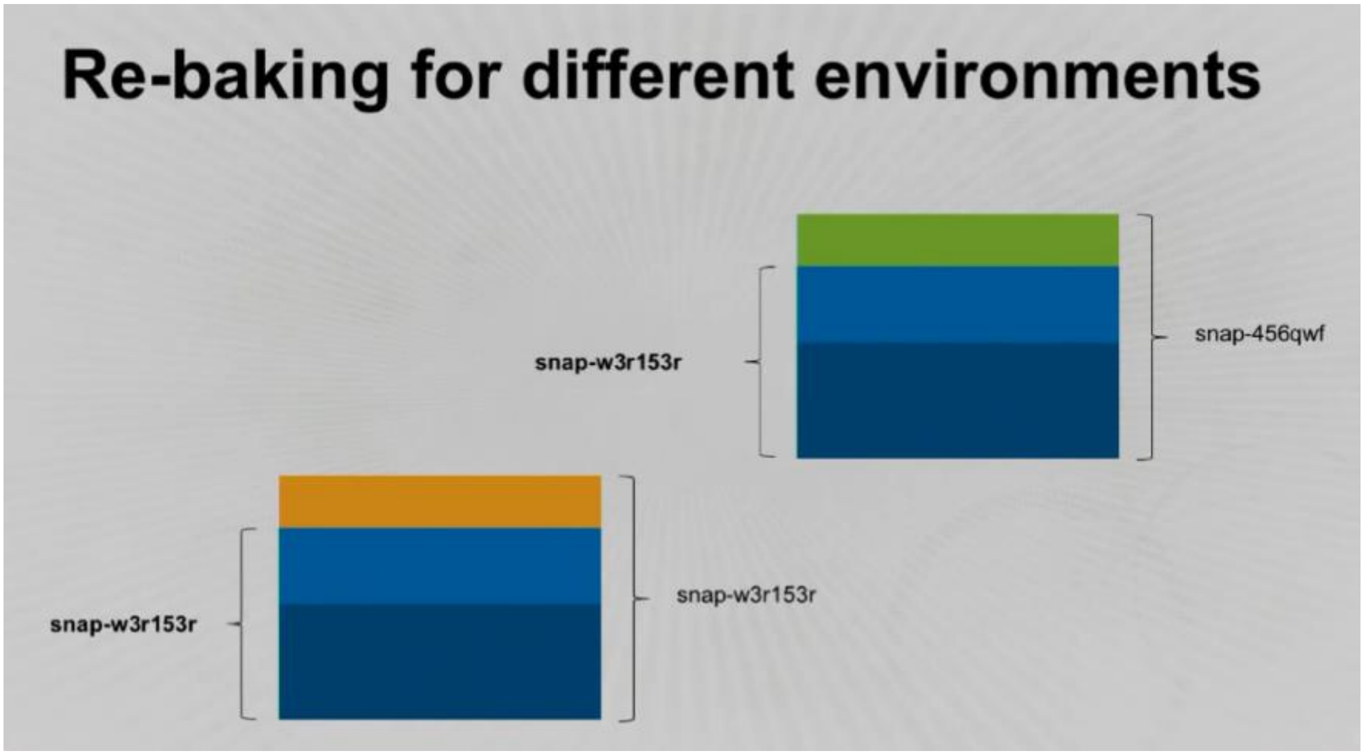


What’s in a machine?

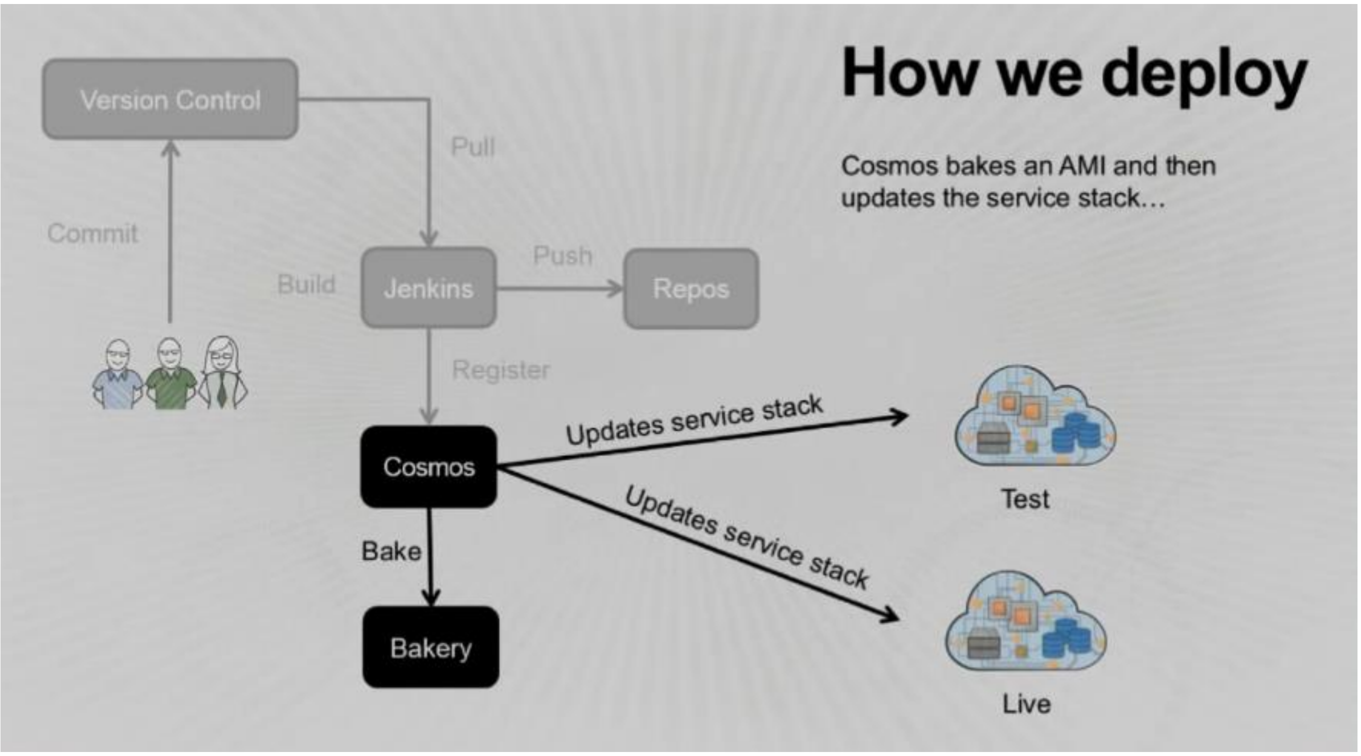
Machine is what runs a service, the Service is a software binary that doesn’t change. The only thing that changes is the configuration



2 step snapshotting



Re-baking for different environments



How we deploy

Cosmos bakes an AMI and then updates the service stack…

From this point on, it is a CloudFormation call to update the service stack

…what actually happens

- Cosmos updates the ImageId property of the Auto Scaling Group's LaunchConfiguration
- Based on the specified UpdatePolicy, the ASG starts refreshing the instances with new ones using the new AMI

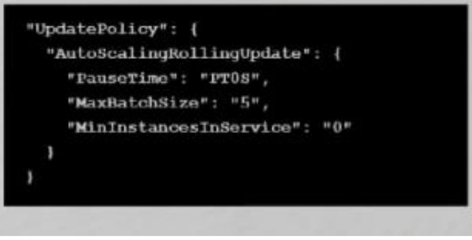
Optimizing the ASG UpdatePolicy

- On test environments you can **optimize for speed** and replace all instances at once
- Once live, you should update the ASG in batches making sure you don't have downtime

…for example

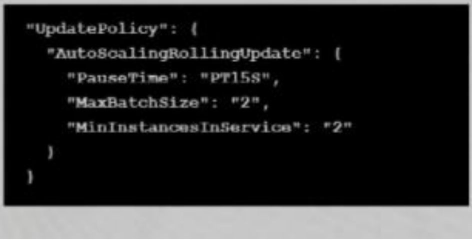
For a service with an ASG with 5 instances…

TEST



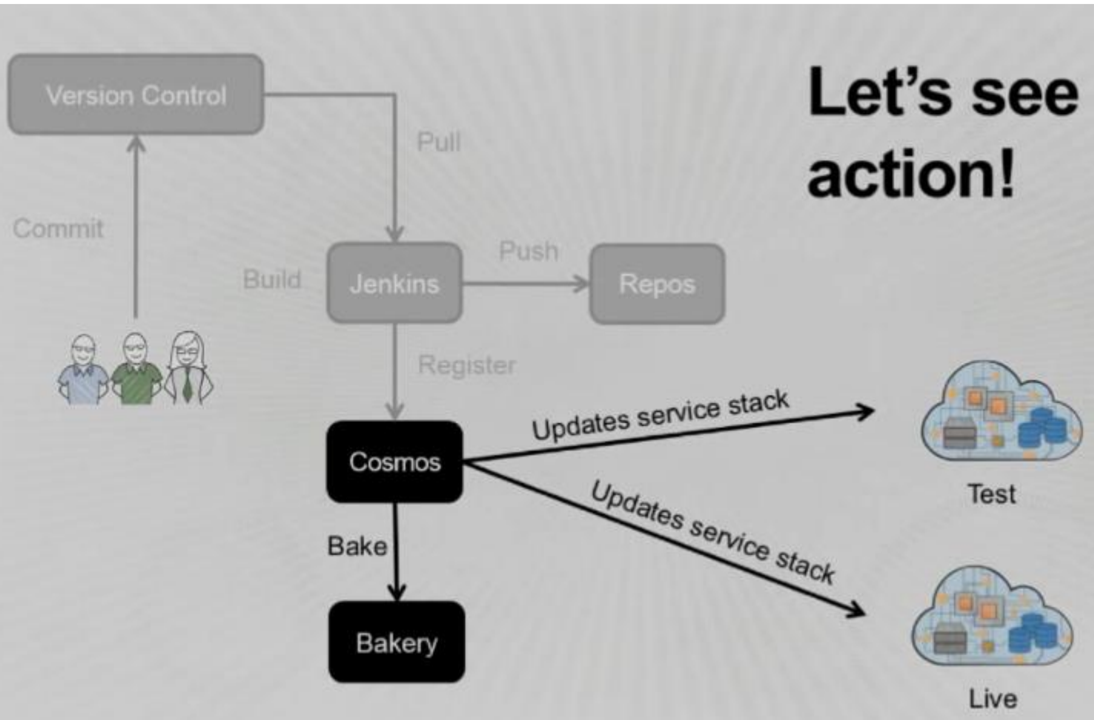
```
"UpdatePolicy": {
  "AutoScalingRollingUpdate": {
    "PauseTime": "PT0S",
    "MaxBatchSize": "5",
    "MinInstancesInService": "0"
  }
}
```

LIVE

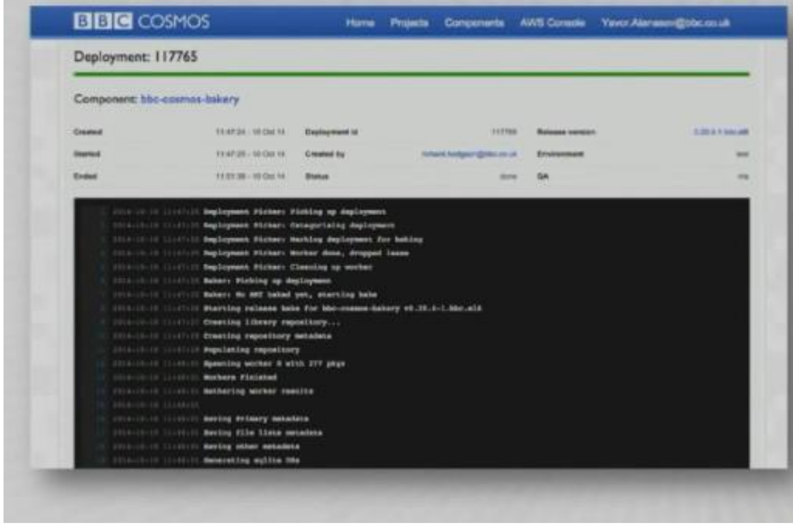


```
"UpdatePolicy": {
  "AutoScalingRollingUpdate": {
    "PauseTime": "PT15S",
    "MaxBatchSize": "2",
    "MinInstancesInService": "2"
  }
}
```

Let's see it in action!



Demo time



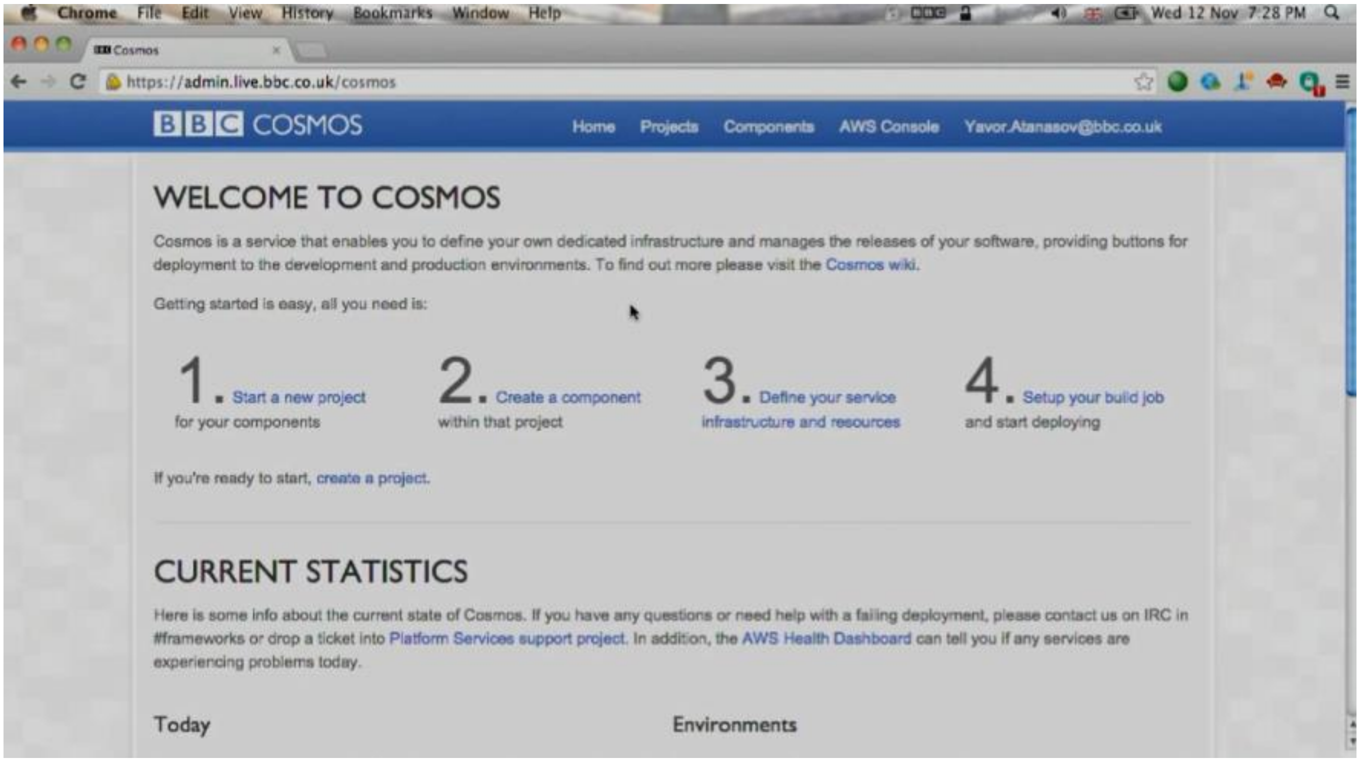
Deployment: 117765

Component: bbc-cosmos-bakery

Created	11:47:24 - 10 Oct 14	Deployment id	117765	Release version	0.20.0-1.bbc.el6
Started	11:47:25 - 10 Oct 14	Created by	richard.hodgson@bbc.co.uk	Environment	test
Ended	11:51:38 - 10 Oct 14	Status	done	QA	n/a

```
2014-10-10 11:47:25 Deployment Picker: Picking up deployment
2014-10-10 11:47:25 Deployment Picker: Categorizing deployment
2014-10-10 11:47:25 Deployment Picker: Marking deployment for baking
2014-10-10 11:47:25 Deployment Picker: Worker done, dropped lease
2014-10-10 11:47:25 Deployment Picker: Cleaning up worker
2014-10-10 11:47:25 Baker: Picking up deployment
2014-10-10 11:47:25 Baker: No AMI baked yet, starting bake
2014-10-10 11:47:26 Starting release bake for bbc-cosmos-bakery v0.20.0-1.bbc.el6
2014-10-10 11:47:27 Creating library repository...
2014-10-10 11:47:27 Creating repository metadata
2014-10-10 11:47:28 Populating repository
2014-10-10 11:48:31 Spawning worker 0 with 277 pkgs
2014-10-10 11:48:31 Workers Finished
2014-10-10 11:48:31 Gathering worker results
2014-10-10 11:48:31
2014-10-10 11:48:31 Saving Primary metadata
2014-10-10 11:48:31 Saving file lists metadata
2014-10-10 11:48:31 Saving other metadata
2014-10-10 11:48:31 Generating sqlite DBs
```

Let's deploy one of our services and see what happens…



WELCOME TO COSMOS

Cosmos is a service that enables you to define your own dedicated infrastructure and manages the releases of your software, providing buttons for deployment to the development and production environments. To find out more please visit the Cosmos wiki.

Getting started is easy, all you need is:

1. Start a new project for your components

2. Create a component within that project

3. Define your service infrastructure and resources

4. Setup your build job and start deploying

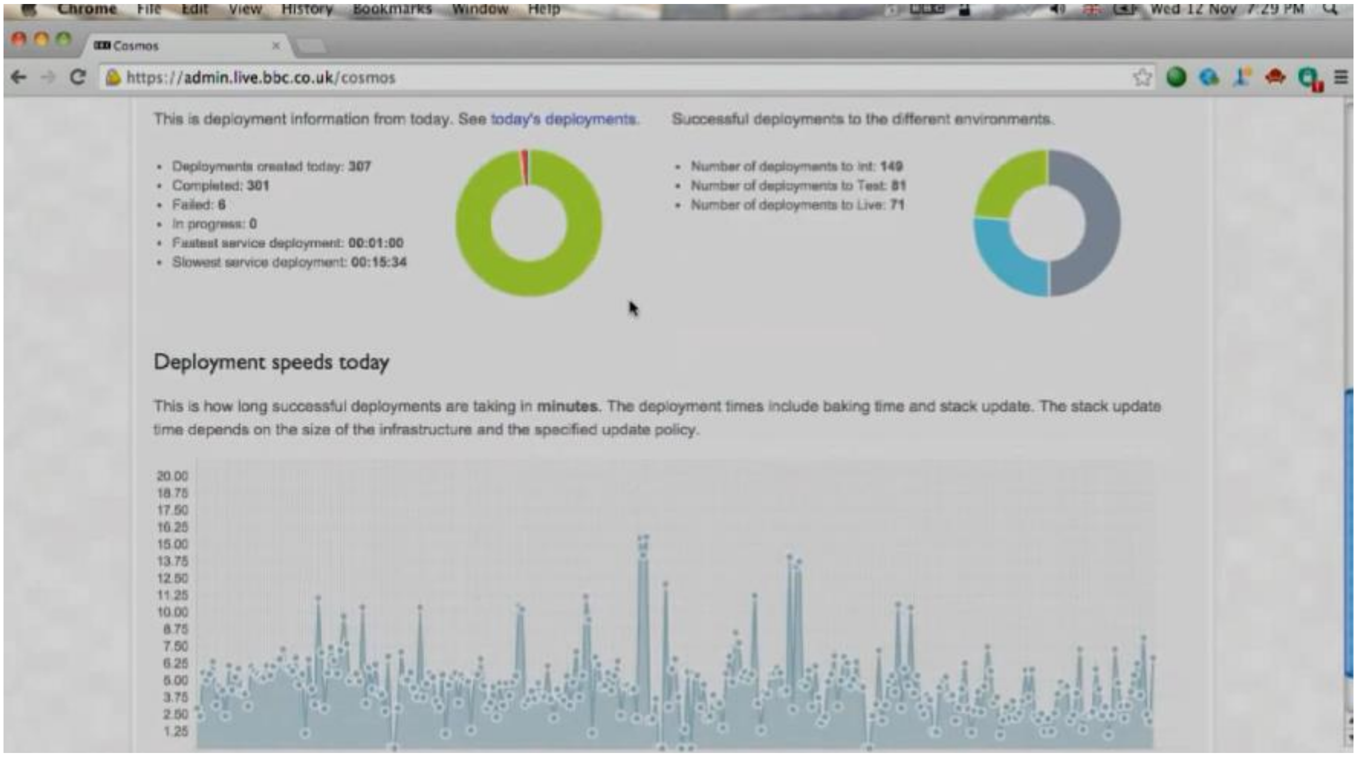
If you're ready to start, create a project.

CURRENT STATISTICS

Here is some info about the current state of Cosmos. If you have any questions or need help with a failing deployment, please contact us on IRC in #frameworks or drop a ticket into Platform Services support project. In addition, the AWS Health Dashboard can tell you if any services are experiencing problems today.

Today

Environments



Chrome File Edit View History Bookmarks Window Help
Wed 12 Nov 7:29 PM
Cosmos
https://admin.live.bbc.co.uk/cosmos

This is deployment information from today. See today's deployments.

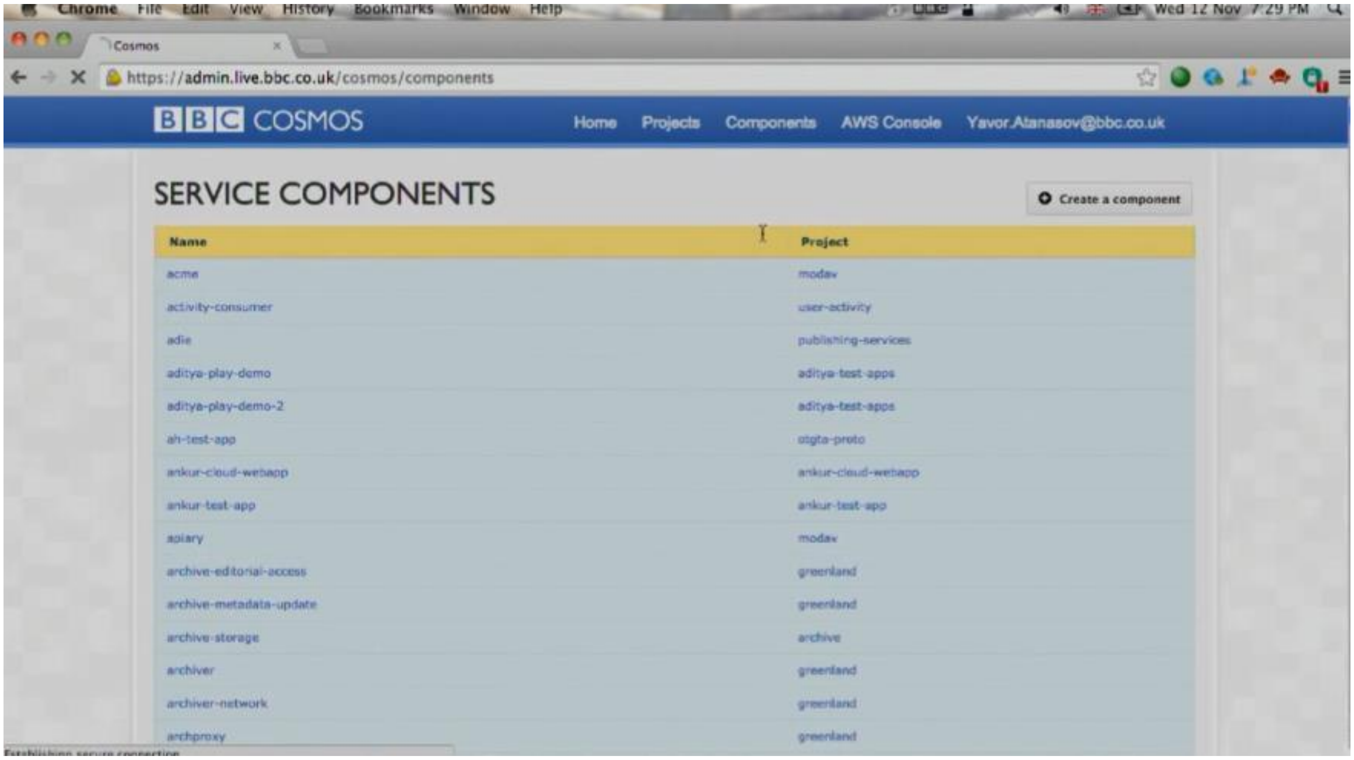
- Deployments created today: **307**
- Completed: **301**
- Failed: **6**
- In progress: **0**
- Fastest service deployment: **00:01:00**
- Slowest service deployment: **00:15:34**

Successful deployments to the different environments.

- Number of deployments to Int: **149**
- Number of deployments to Test: **81**
- Number of deployments to Live: **71**

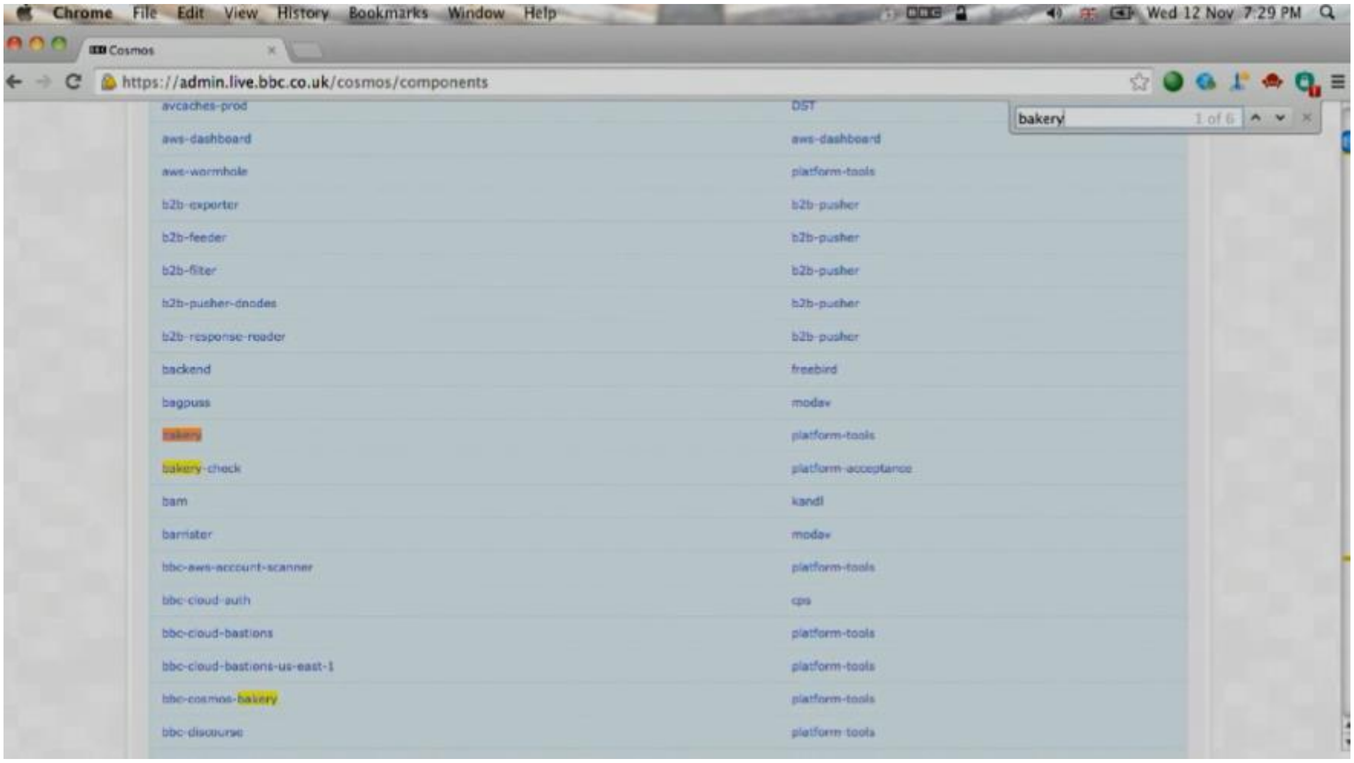
Deployment speeds today

This is how long successful deployments are taking in **minutes**. The deployment times include baking time and stack update. The stack update time depends on the size of the infrastructure and the specified update policy.

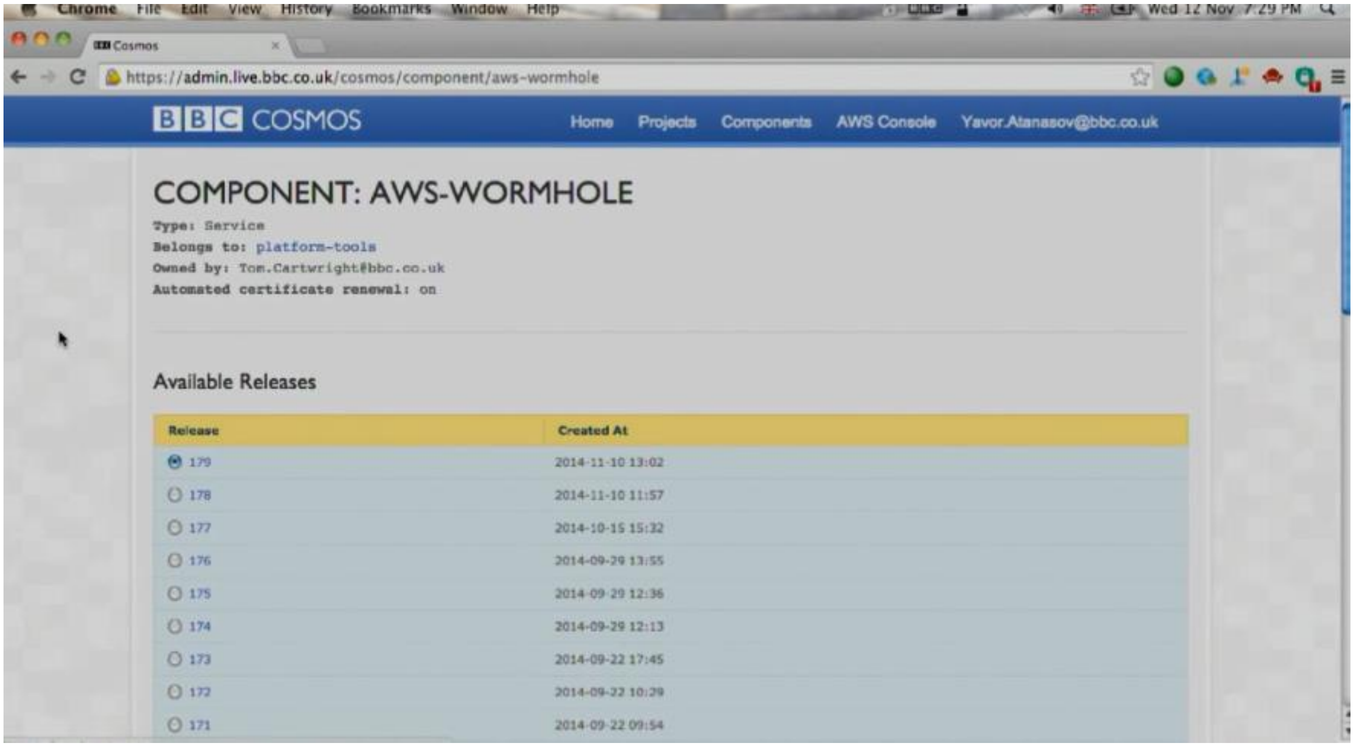

Chrome File Edit View History Bookmarks Window Help
Wed 12 Nov 7:29 PM
Cosmos
https://admin.live.bbc.co.uk/cosmos/components
BBC COSMOS
Home
Projects
Components
AWS Console
Yavor.Atanasov@bbc.co.uk
Create a component
Establishing secure connection

SERVICE COMPONENTS

Name	Project
acme	modav
activity-consumer	user-activity
adie	publishing-services
aditya-play-demo	aditya-test-apps
aditya-play-demo-2	aditya-test-apps
ah-test-app	otgta-proto
ankur-cloud-webapp	ankur-cloud-webapp
ankur-test-app	ankur-test-app
apiary	modav
archive-editorial-access	greenland
archive-metadata-update	greenland
archive-storage	archive
archiver	greenland
archiver-network	greenland
archproxy	greenland


Chrome File Edit View History Bookmarks Window Help
Wed 12 Nov 7:29 PM
Cosmos
https://admin.live.bbc.co.uk/cosmos/components
bakery
1 of 6

avcaches-prod	DST
aws-dashboard	aws-dashboard
aws-wormhole	platform-tools
b2b-exporter	b2b-pusher
b2b-feeder	b2b-pusher
b2b-filter	b2b-pusher
b2b-pusher-dnodes	b2b-pusher
b2b-response-reader	b2b-pusher
backend	freebird
bagpuss	modav
bakery	platform-tools
bakery-check	platform-acceptance
bam	kandl
barrister	modav
bbc-aws-account-scanner	platform-tools
bbc-cloud-auth	cps
bbc-cloud-bastions	platform-tools
bbc-cloud-bastions-us-east-1	platform-tools
bbc-cosmos-bakery	platform-tools
bbc-discourse	platform-tools

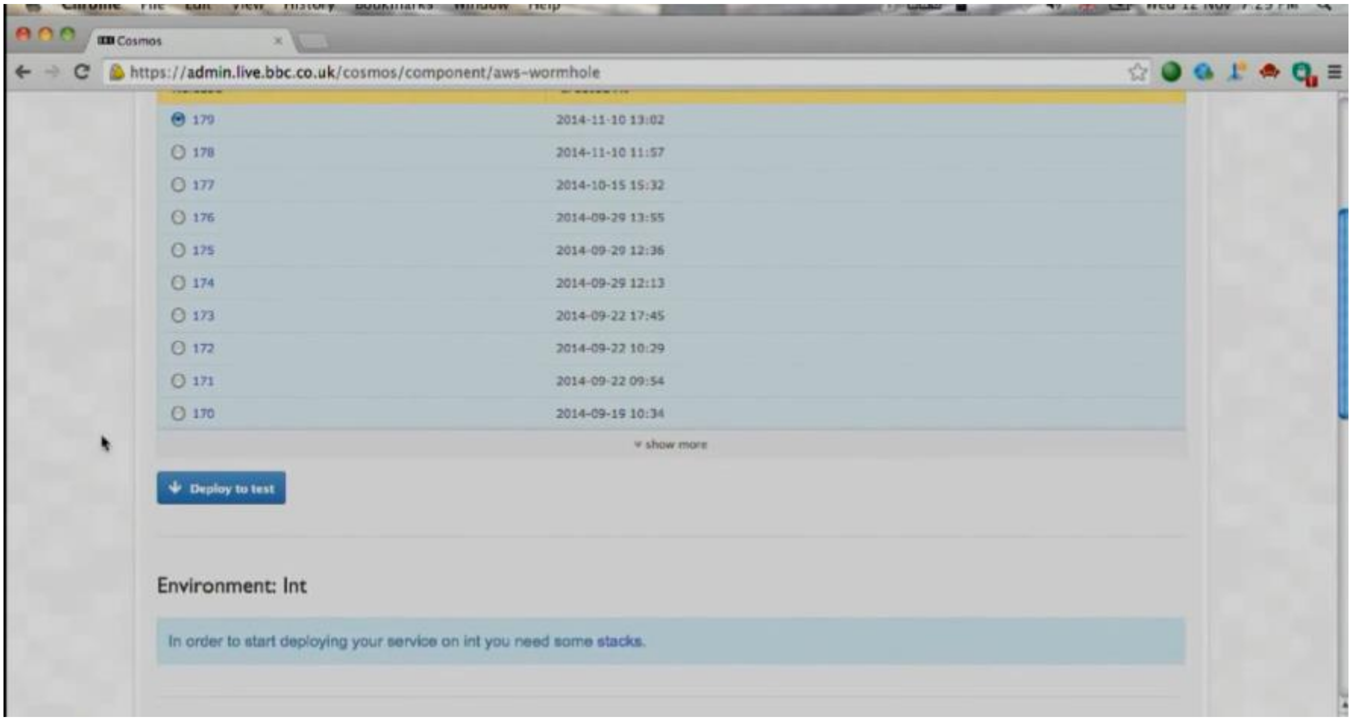

Chrome File Edit View History Bookmarks Window Help
Wed 12 Nov 7:29 PM
Cosmos
https://admin.live.bbc.co.uk/cosmos/component/aws-wormhole
BBC COSMOS
Home
Projects
Components
AWS Console
Yavor.Atanasov@bbc.co.uk

COMPONENT: AWS-WORMHOLE

Type: Service
Belongs to: platform-tools
Owned by: Tom.Cartwright@bbc.co.uk
Automated certificate renewal: on

Available Releases

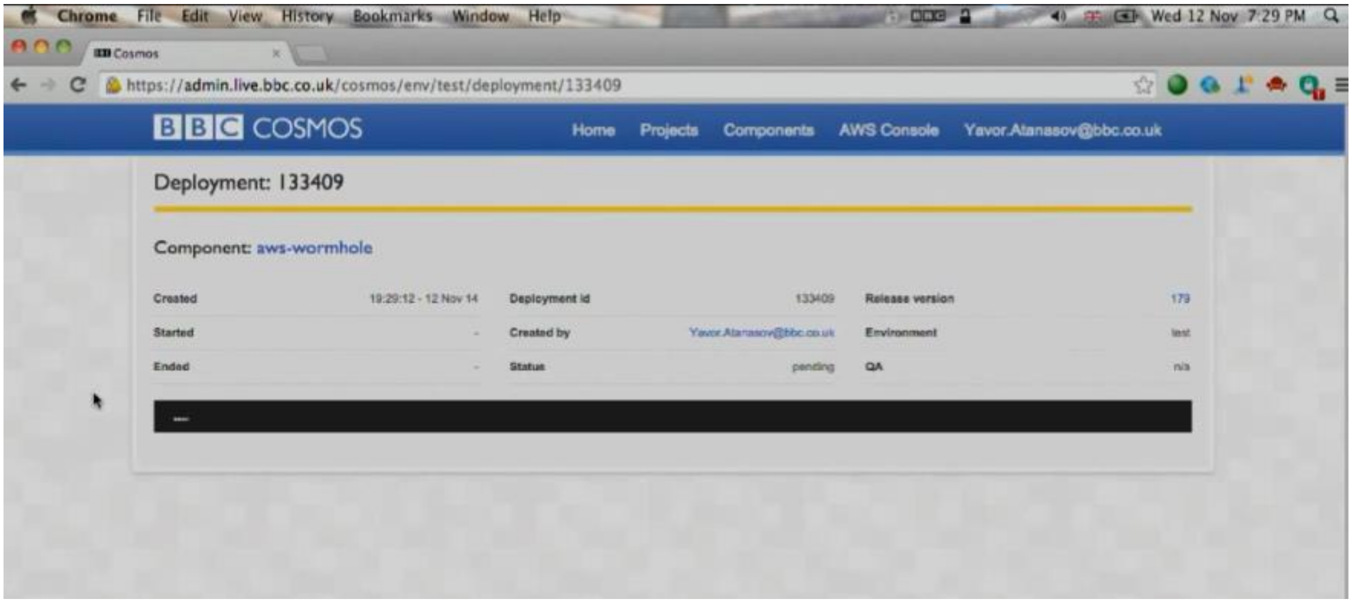
Release	Created At
● 179	2014-11-10 13:02
○ 178	2014-11-10 11:57
○ 177	2014-10-15 15:32
○ 176	2014-09-29 13:55
○ 175	2014-09-29 12:36
○ 174	2014-09-29 12:13
○ 173	2014-09-22 17:45
○ 172	2014-09-22 10:29
○ 171	2014-09-22 09:54



● 179	2014-11-10 13:02
○ 178	2014-11-10 11:57
○ 177	2014-10-15 15:32
○ 176	2014-09-29 13:55
○ 175	2014-09-29 12:36
○ 174	2014-09-29 12:13
○ 173	2014-09-22 17:45
○ 172	2014-09-22 10:29
○ 171	2014-09-22 09:54
○ 170	2014-09-19 10:34

Environment: Int

In order to start deploying your service on int you need some stacks.

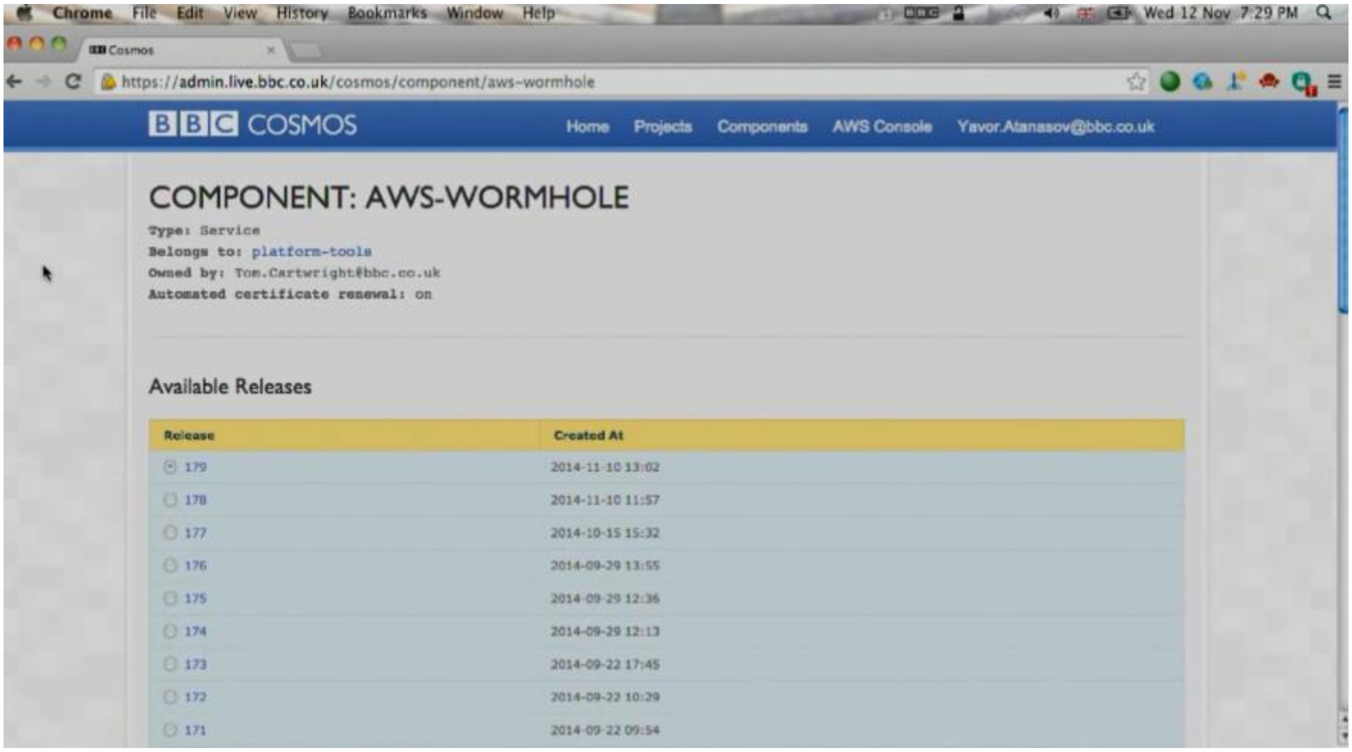

Chrome File Edit View History Bookmarks Window Help
Wed 12 Nov 7:29 PM
Cosmos
https://admin.live.bbc.co.uk/cosmos/env/test/deployment/133409
BBC COSMOS
Home Projects Components AWS Console Yavor.Atanasov@bbc.co.uk

Deployment: 133409

Component: aws-wormhole

Created	19:29:12 - 12 Nov 14	Deployment id	133409	Release version	179
Started	-	Created by	Yavor.Atanasov@bbc.co.uk	Environment	test
Ended	-	Status	pending	QA	n/a

```
---
```

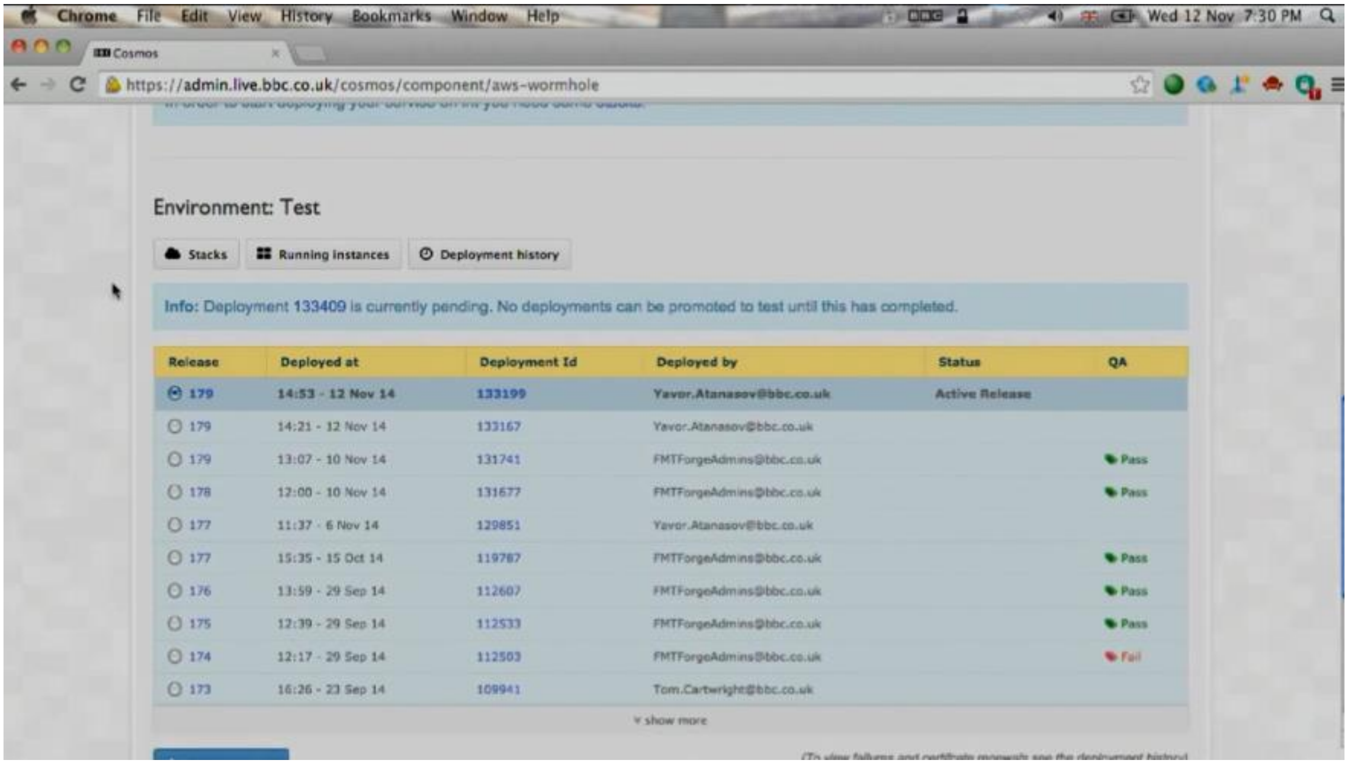

Chrome File Edit View History Bookmarks Window Help
Wed 12 Nov 7:29 PM
Cosmos
https://admin.live.bbc.co.uk/cosmos/component/aws-wormhole
BBC COSMOS
Home Projects Components AWS Console Yavor.Atanasov@bbc.co.uk

COMPONENT: AWS-WORMHOLE

Type: Service
Belongs to: platform-tools
Owned by: Tom.Cartwright@bbc.co.uk
Automated certificate renewal: on

Available Releases

Release	Created At
● 179	2014-11-10 13:02
○ 178	2014-11-10 11:57
○ 177	2014-10-15 15:32
○ 176	2014-09-29 13:55
○ 175	2014-09-29 12:36
○ 174	2014-09-29 12:13
○ 173	2014-09-22 17:45
○ 172	2014-09-22 10:29
○ 171	2014-09-22 09:54


Chrome File Edit View History Bookmarks Window Help
Wed 12 Nov 7:30 PM
Cosmos
https://admin.live.bbc.co.uk/cosmos/component/aws-wormhole

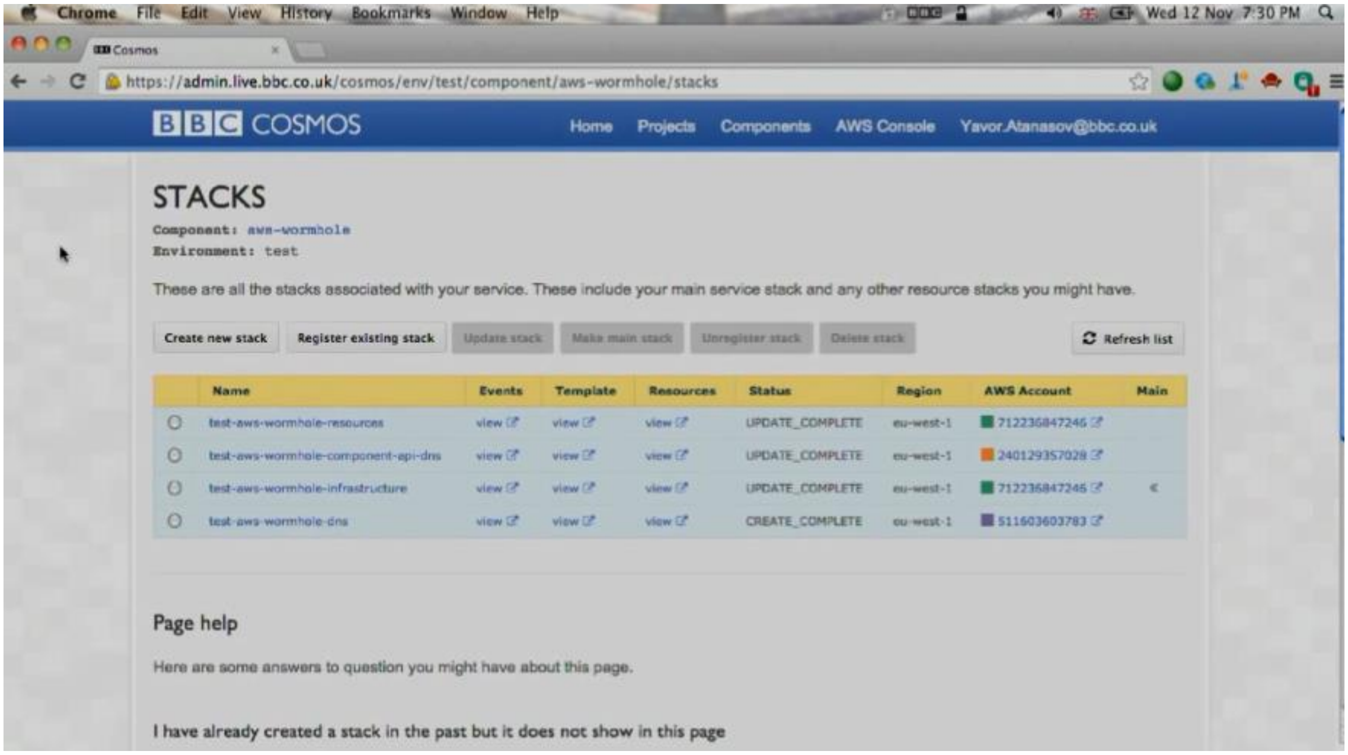
Environment: Test

Stacks | Running instances | Deployment history

Info: Deployment 133409 is currently pending. No deployments can be promoted to test until this has completed.

Release	Deployed at	Deployment Id	Deployed by	Status	QA
● 179	14:53 - 12 Nov 14	133199	Yavor.Atanasov@bbc.co.uk	Active Release	
○ 179	14:21 - 12 Nov 14	133167	Yavor.Atanasov@bbc.co.uk		
○ 179	13:07 - 10 Nov 14	131741	FMTForgeAdmins@bbc.co.uk		Pass
○ 178	12:00 - 10 Nov 14	131677	FMTForgeAdmins@bbc.co.uk		Pass
○ 177	11:37 - 6 Nov 14	129851	Yavor.Atanasov@bbc.co.uk		
○ 177	15:35 - 15 Oct 14	119787	FMTForgeAdmins@bbc.co.uk		Pass
○ 176	13:59 - 29 Sep 14	112607	FMTForgeAdmins@bbc.co.uk		Pass
○ 175	12:39 - 29 Sep 14	112533	FMTForgeAdmins@bbc.co.uk		Pass
○ 174	12:17 - 29 Sep 14	112503	FMTForgeAdmins@bbc.co.uk		Fail
○ 173	16:26 - 23 Sep 14	109941	Tom.Cartwright@bbc.co.uk		

show more


Chrome File Edit View History Bookmarks Window Help
Wed 12 Nov 7:30 PM
Cosmos
https://admin.live.bbc.co.uk/cosmos/env/test/component/aws-wormhole/stacks
BBC COSMOS
Home
Projects
Components
AWS Console
Yavor.Atanasov@bbc.co.uk

STACKS

Component: aws-wormhole
Environment: test

These are all the stacks associated with your service. These include your main service stack and any other resource stacks you might have.

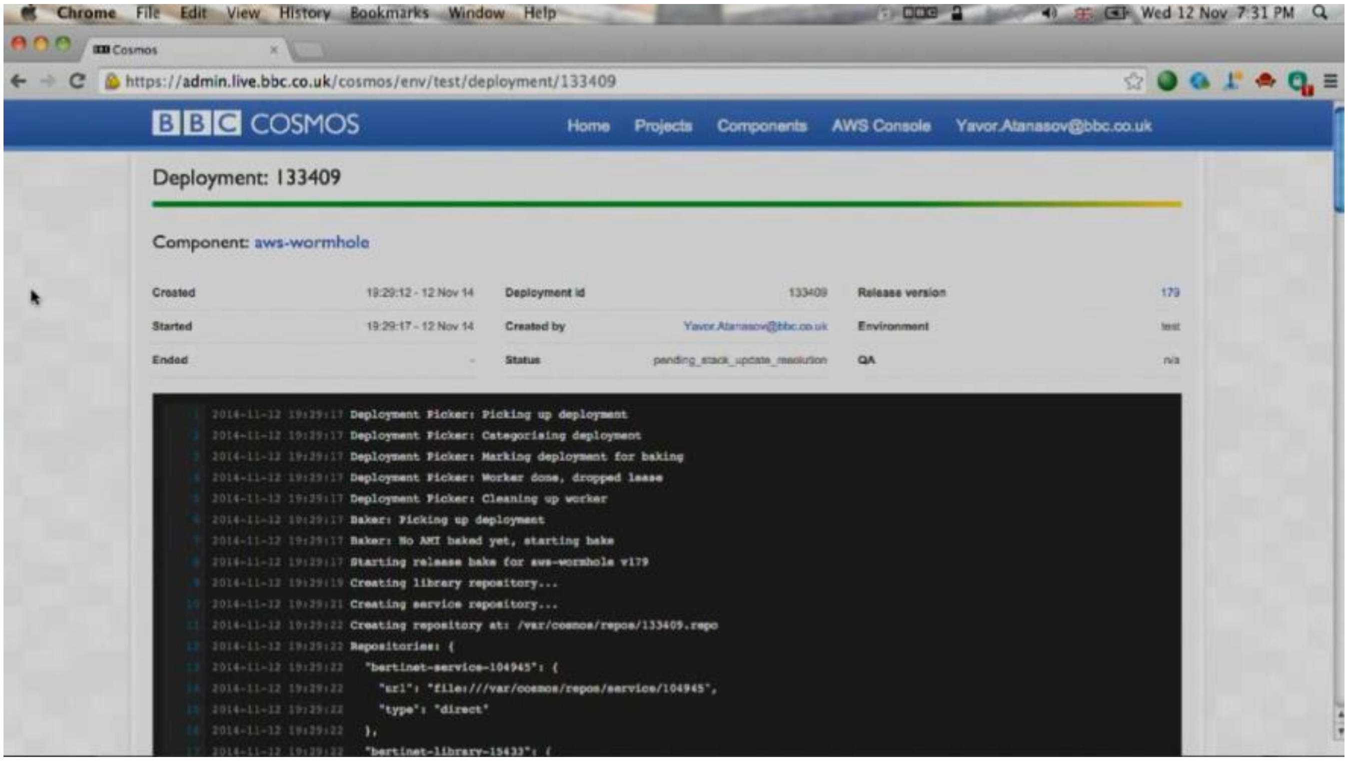
Create new stack | Register existing stack | Update stack | Make main stack | Unregister stack | Delete stack | Refresh list

	Name	Events	Template	Resources	Status	Region	AWS Account	Main
○	test-aws-wormhole-resources	view	view	view	UPDATE_COMPLETE	eu-west-1	🟩 712236847246	
○	test-aws-wormhole-component-api-dns	view	view	view	UPDATE_COMPLETE	eu-west-1	🟧 240129357028	
○	test-aws-wormhole-infrastructure	view	view	view	UPDATE_COMPLETE	eu-west-1	🟩 712236847246	«
○	test-aws-wormhole-dns	view	view	view	CREATE_COMPLETE	eu-west-1	🟪 511603603783	

Page help

Here are some answers to question you might have about this page.

I have already created a stack in the past but it does not show in this page



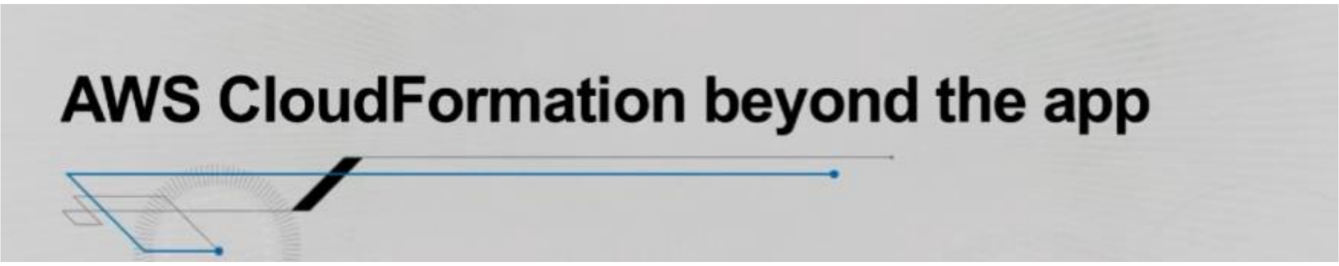
Chrome File Edit View History Bookmarks Window Help
Wed 12 Nov 7:31 PM
Cosmos
https://admin.live.bbc.co.uk/cosmos/env/test/deployment/133409
BBC COSMOS
Home
Projects
Components
AWS Console
Yavor.Atanasov@bbc.co.uk

Deployment: 133409

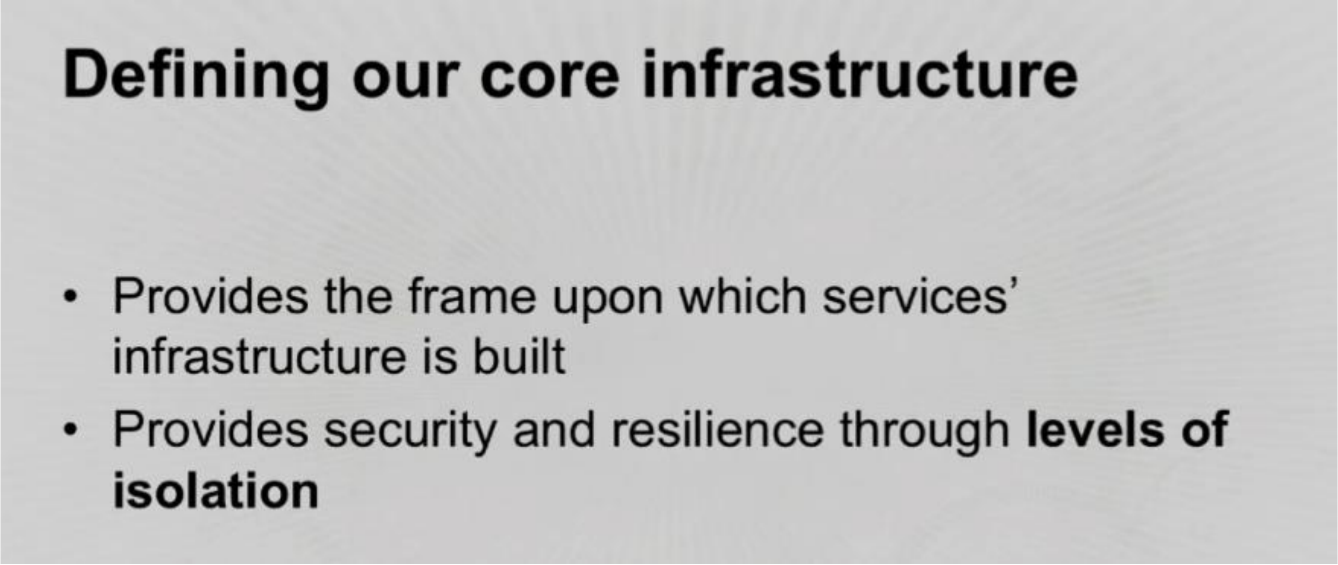
Component: aws-wormhole

Created	19:29:12 - 12 Nov 14	Deployment id	133409	Release version	179
Started	19:29:17 - 12 Nov 14	Created by	Yavor.Atanasov@bbc.co.uk	Environment	test
Ended	-	Status	pending_stack_update_resolution	QA	n/a

```
2014-11-12 19:29:17 Deployment Picker: Picking up deployment
2014-11-12 19:29:17 Deployment Picker: Categorising deployment
2014-11-12 19:29:17 Deployment Picker: Marking deployment for baking
2014-11-12 19:29:17 Deployment Picker: Worker done, dropped lease
2014-11-12 19:29:17 Deployment Picker: Cleaning up worker
2014-11-12 19:29:17 Baker: Picking up deployment
2014-11-12 19:29:17 Baker: No AMI baked yet, starting bake
2014-11-12 19:29:17 Starting release bake for aws-wormhole v179
2014-11-12 19:29:19 Creating library repository...
2014-11-12 19:29:21 Creating service repository...
2014-11-12 19:29:22 Creating repository at: /var/cosmos/repos/133409.repo
2014-11-12 19:29:22 Repositories: {
2014-11-12 19:29:22   "bertinet-service-104945": {
2014-11-12 19:29:22     "url": "file:///var/cosmos/repos/service/104945",
2014-11-12 19:29:22     "type": "direct"
2014-11-12 19:29:22   },
2014-11-12 19:29:22   "bertinet-library-15433": {
```

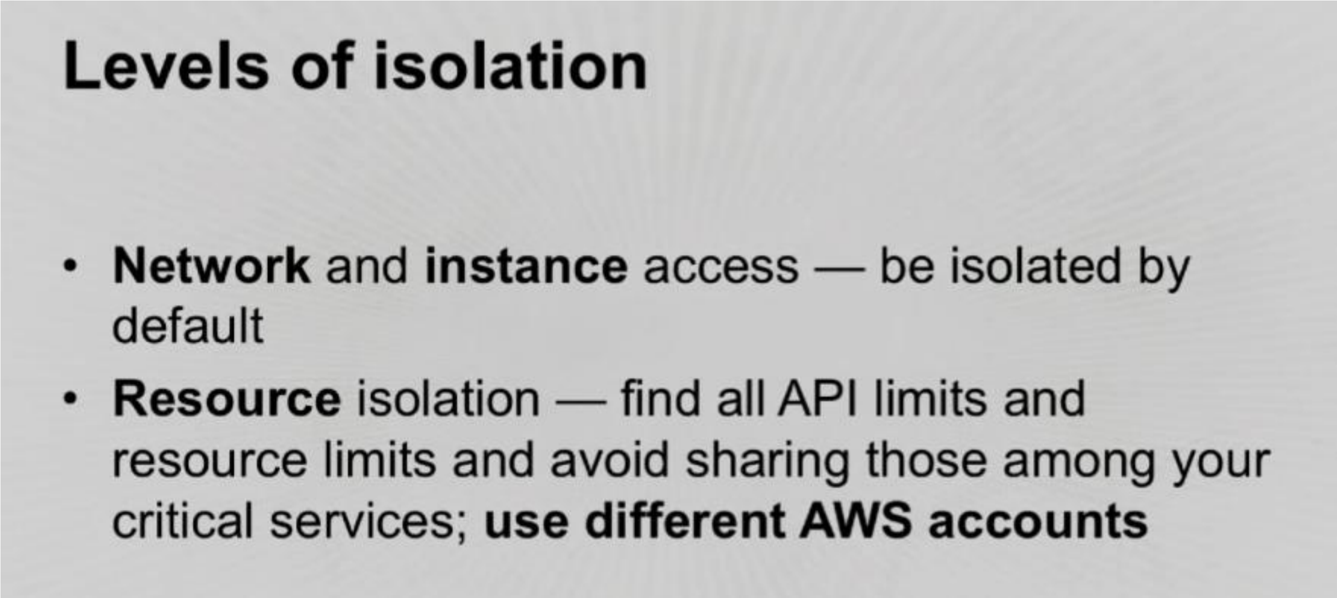


AWS CloudFormation beyond the app



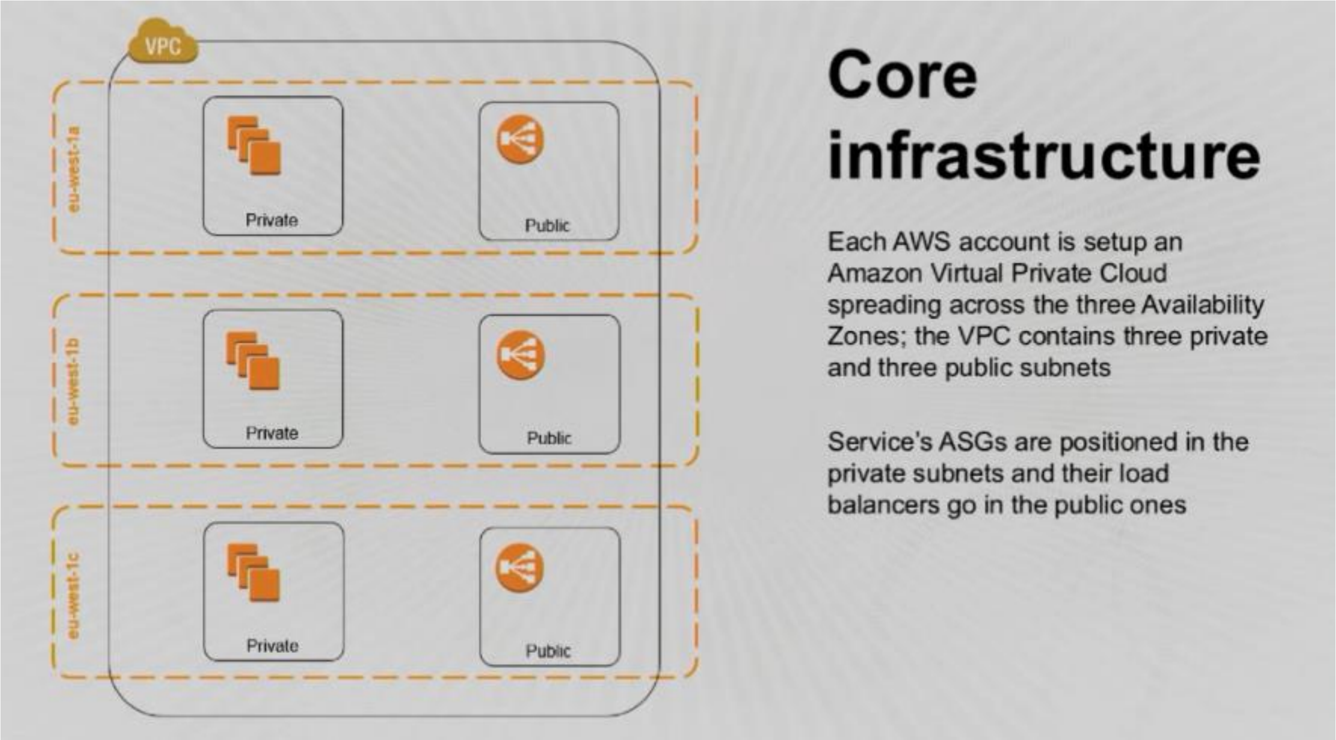
Defining our core infrastructure

- Provides the frame upon which services' infrastructure is built
- Provides security and resilience through **levels of isolation**



Levels of isolation

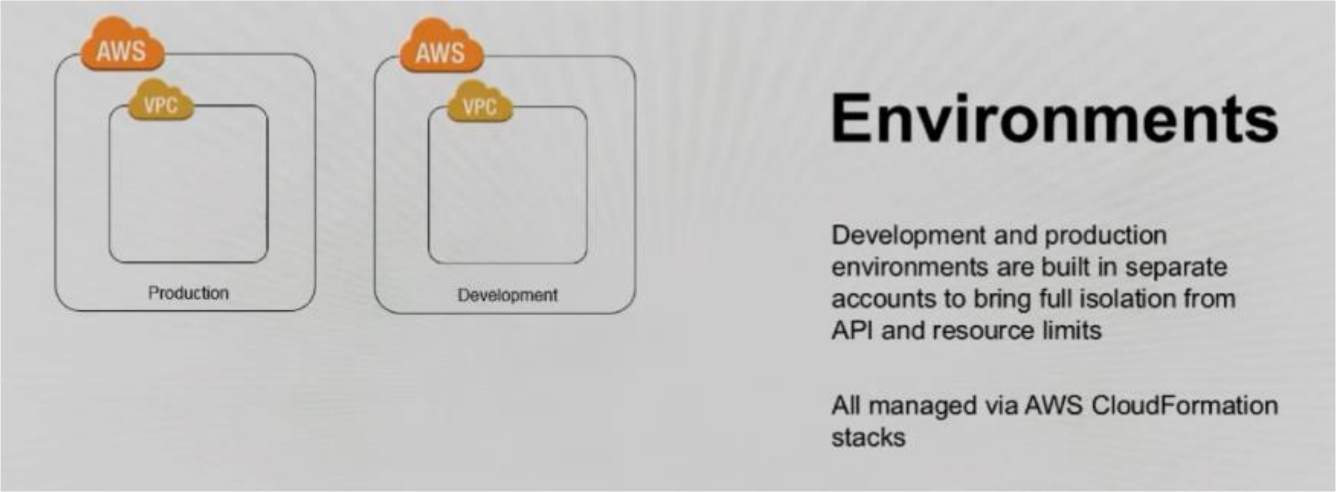
- **Network** and **instance** access — be isolated by default
- **Resource** isolation — find all API limits and resource limits and avoid sharing those among your critical services; **use different AWS accounts**



Core infrastructure

Each AWS account is setup an Amazon Virtual Private Cloud spreading across the three Availability Zones; the VPC contains three private and three public subnets

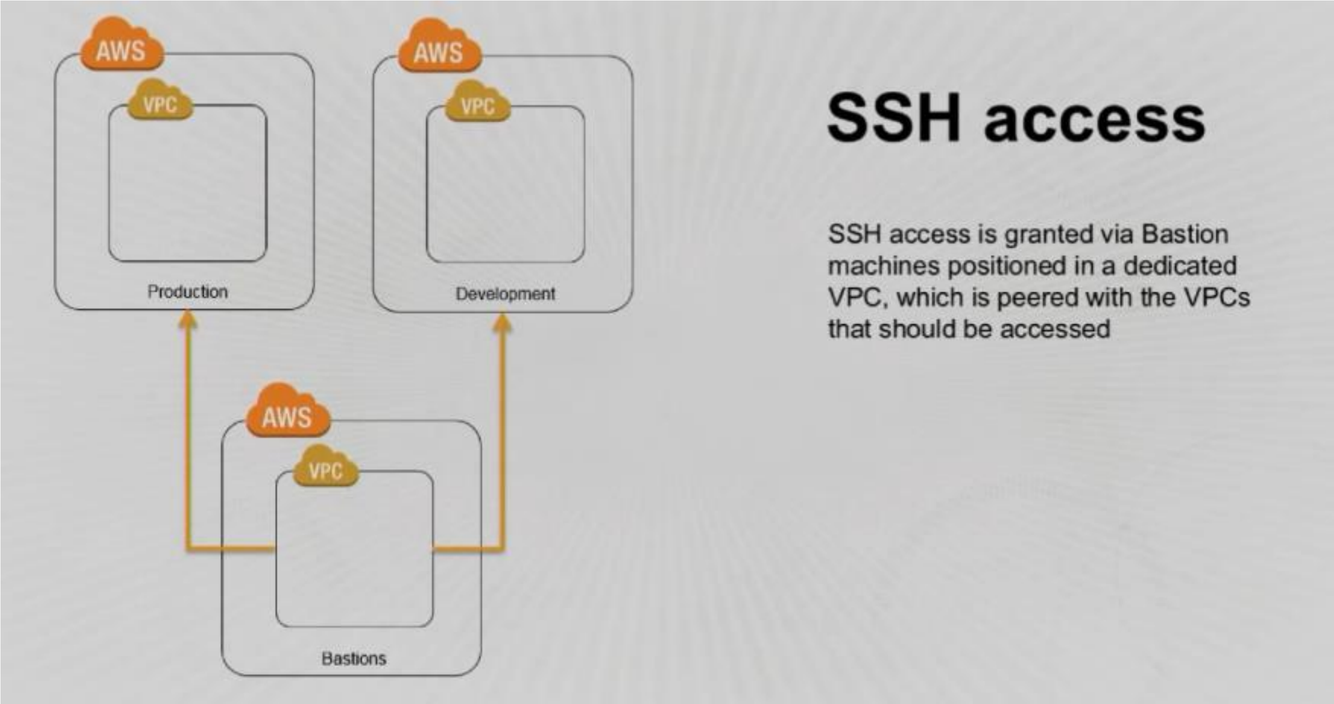
Service's ASGs are positioned in the private subnets and their load balancers go in the public ones



Environments

Development and production environments are built in separate accounts to bring full isolation from API and resource limits

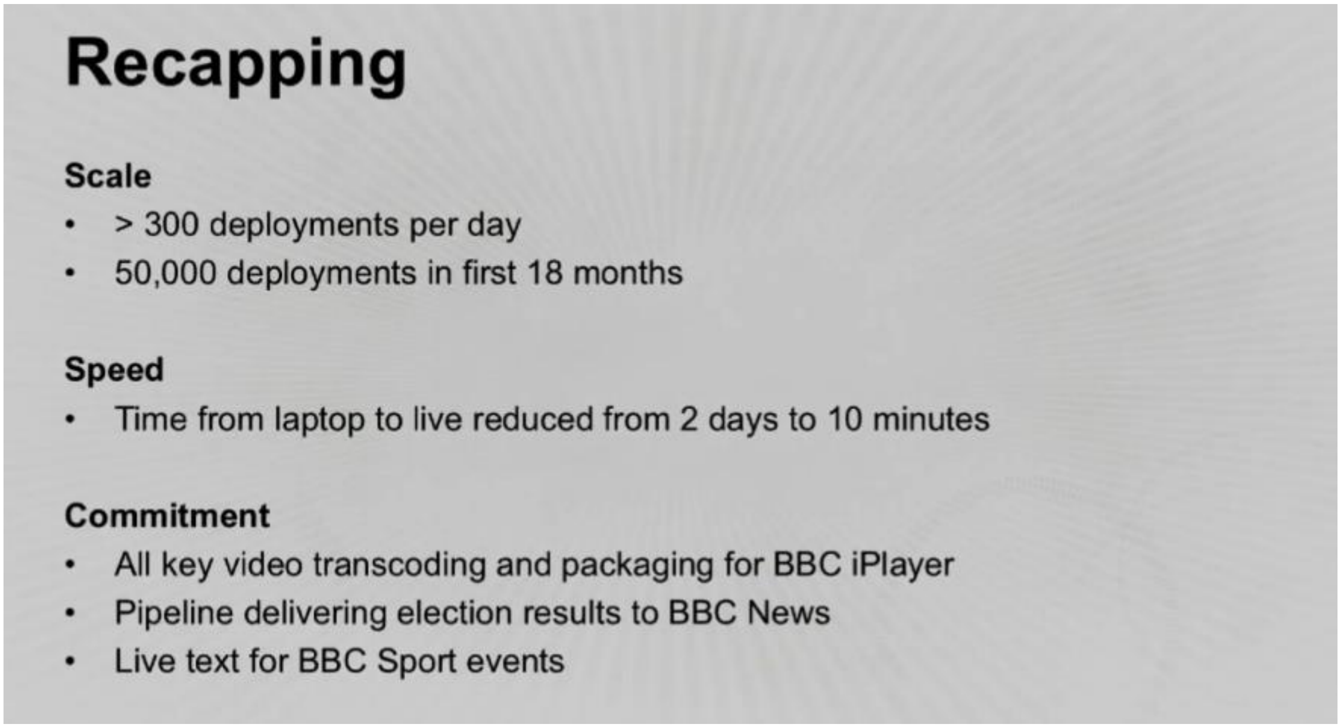
All managed via AWS CloudFormation stacks



SSH access

SSH access is granted via Bastion machines positioned in a dedicated VPC, which is peered with the VPCs that should be accessed

In Closing…



Recapping

Scale

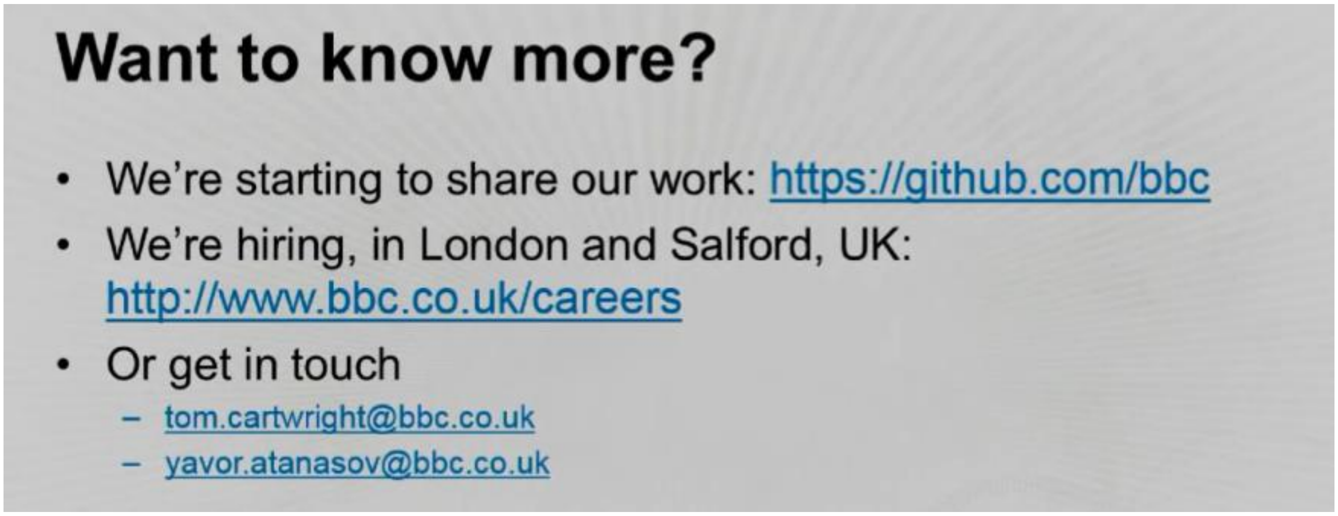
- > 300 deployments per day
- 50,000 deployments in first 18 months

Speed

- Time from laptop to live reduced from 2 days to 10 minutes

Commitment

- All key video transcoding and packaging for BBC iPlayer
- Pipeline delivering election results to BBC News
- Live text for BBC Sport events



Want to know more?

- We're starting to share our work: https://github.com/bbc
- We're hiring, in London and Salford, UK: http://www.bbc.co.uk/careers
- Or get in touch
  - tom.cartwright@bbc.co.uk
  - yavor.atanasov@bbc.co.uk