CloudBees

Continuous Delivery with Amazon ECS and Jenkins

Tracy Kennedy - 8/17/2016

Who is Tracy?

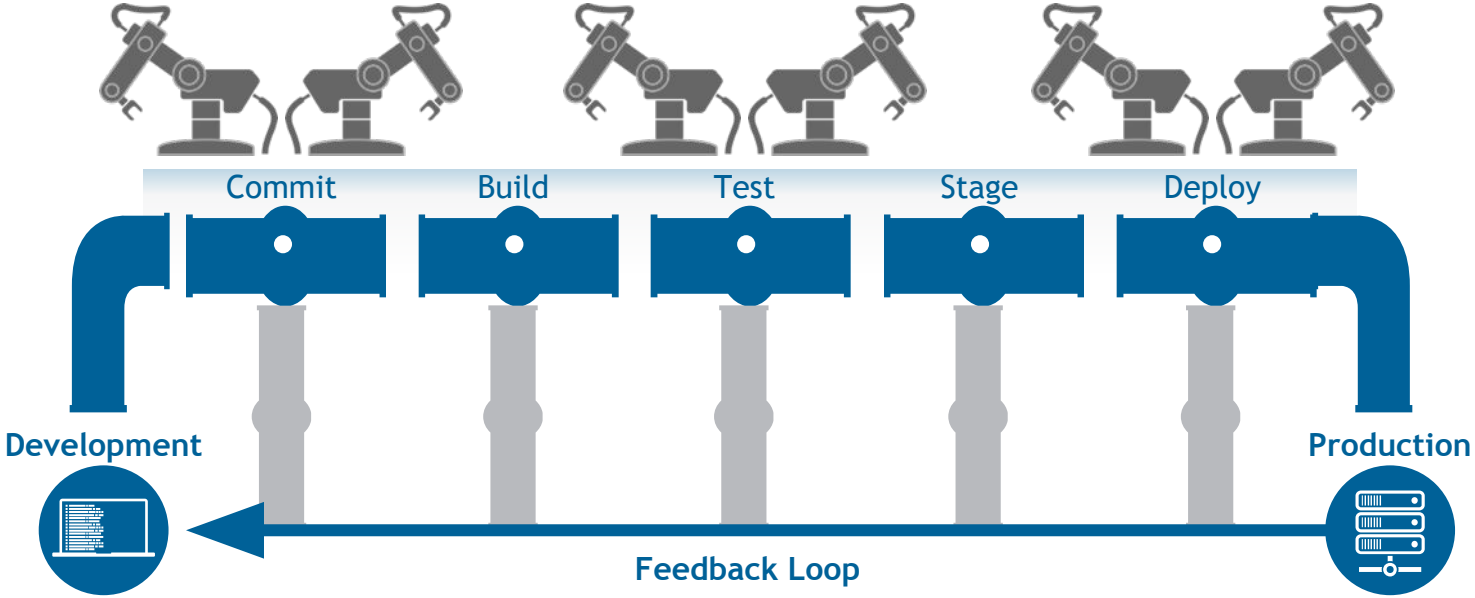
- Product manager @CloudBees
- Owns CloudBees Network

Email: tkennedy@cloudbees.com
Git|Docker Hub: lavaliere
Twitter: @Tracy_Kennedy

Who is Cyrille?

- Director of product management @ CloudBees
- In charge of our Enterprise Edition
- Built the CloudBees partnerships with AWS, Azure,Pivotal, Docker...
- Open source at night: jmxtrans

What is continuous delivery?



What is Jenkins?

Over 1000 Jenkins Plugins

Integration with over 100 DevOps Tools

Orchestration of the DevOps Toolchain

End-to-End CD Pipeline Management

Code & Commit

git, IBM Rational, GitHub, ca technologies, Stash, SERENA, SUBVERSION, Borland StarTeam, PERFORCE, artifactory, mercurial, Nexus, Visual Studio, eclipse

Build & Config

maven, gradle, docker, Apache Ant, CHEF, puppet labs, Gant, ANSIBLE, VAGRANT, Visual Studio, amazon web services, IBM Rational, SALTSTACK

Scan & Test

gerrit Gerrit Code Review, sauce LABS, sonar, BlazeMeter, SOASTA, Gatling, RedLine13.com, Selenium WebDriver, Loadrunner, FitNesse, JUnit, cucumber

Release

uDeploy, SERENA, ca technologies, CollabNet, MidVision Release the innovation, XL RELEASE

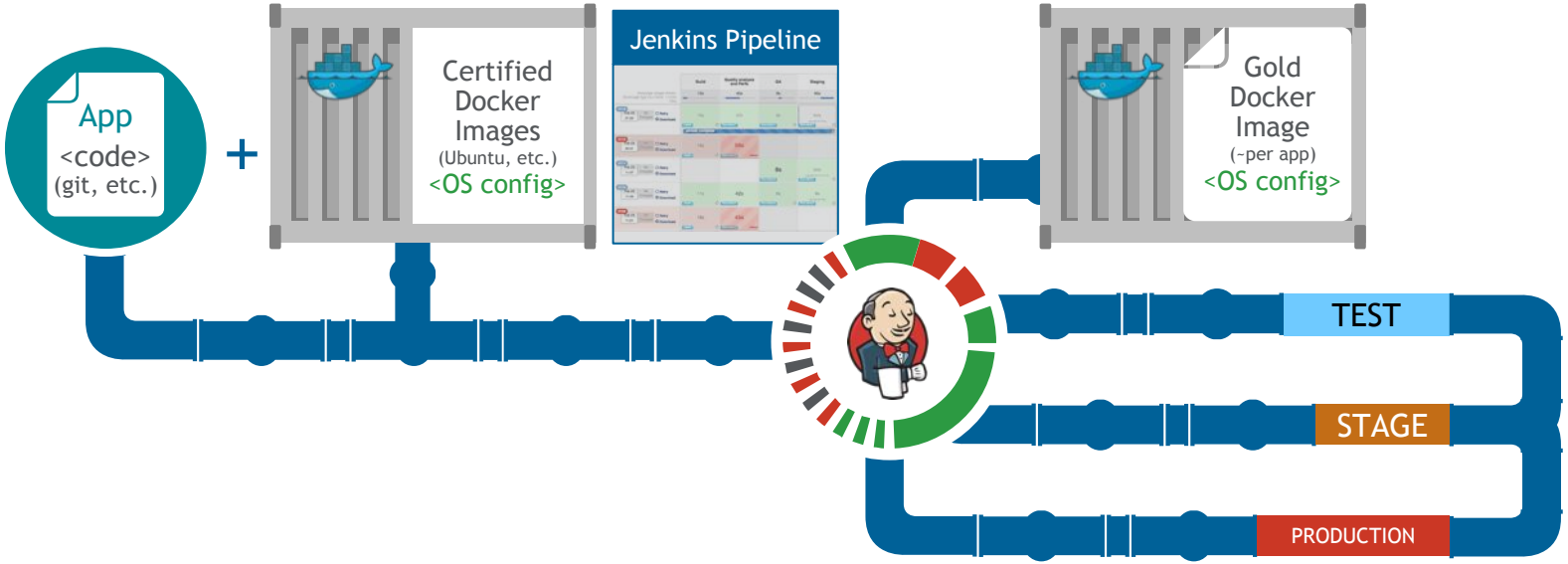
Deploy

amazon web services, Pivotal, Azure, docker, vmware, OPENSHIFT by Red Hat, JBoss by Red Hat, Google Compute Engine, Google app engine, Microsoft .net, openstack, WebSphere

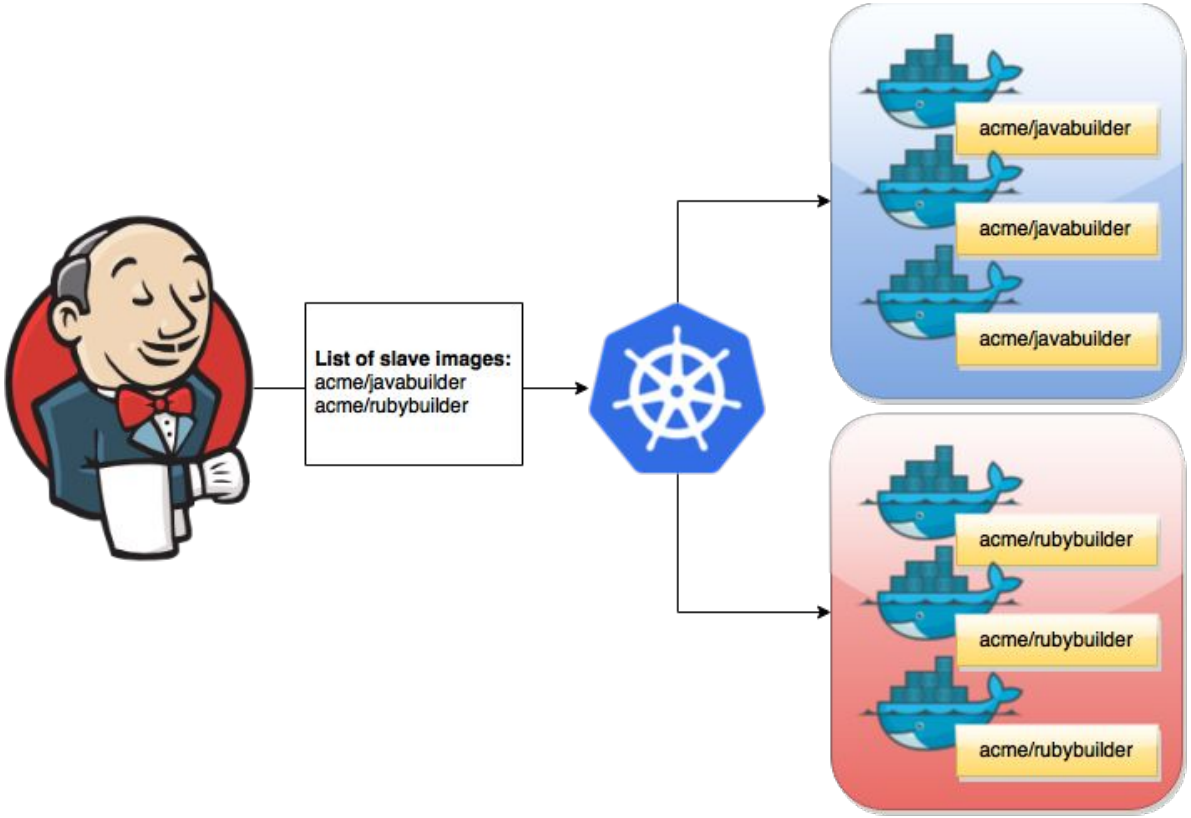
What does CloudBees do?

What is Amazon Elastic Container Service (ECS)?

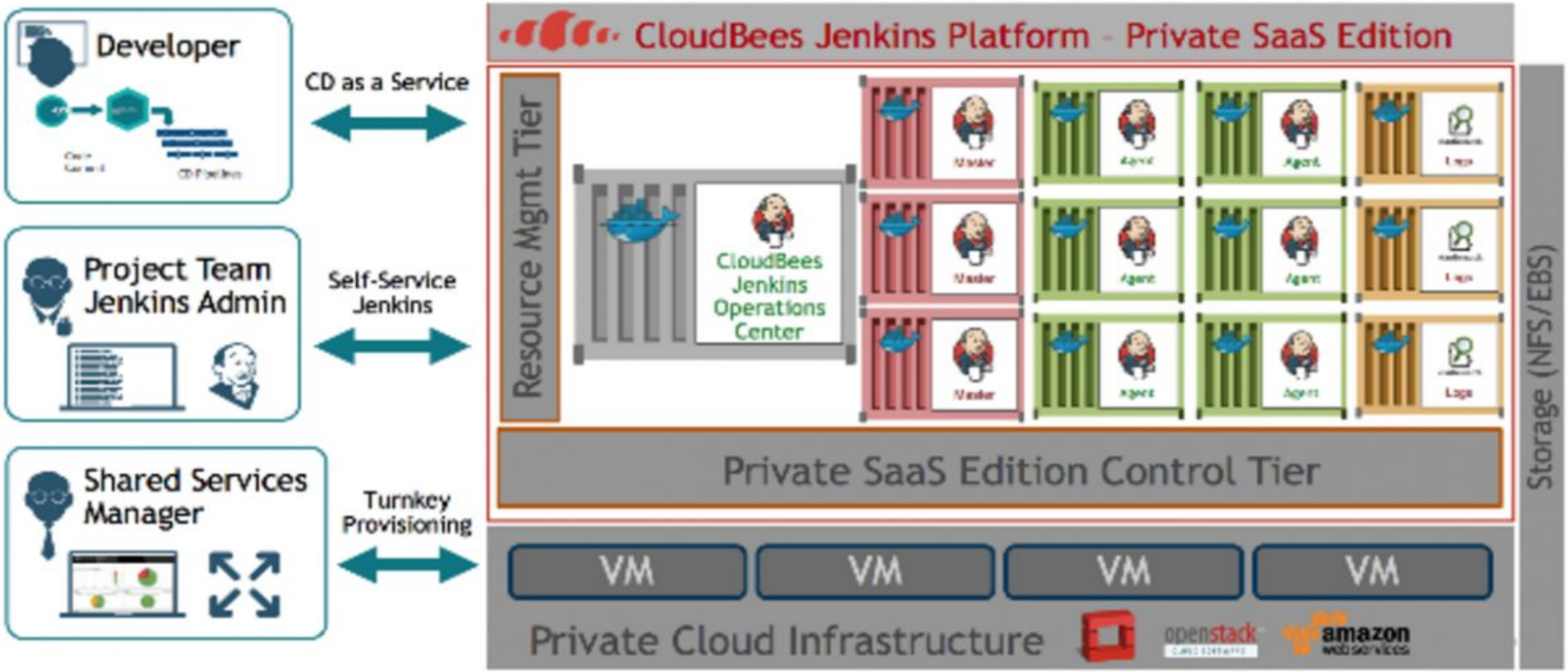
Using Docker with Jenkins



Jenkins and container management



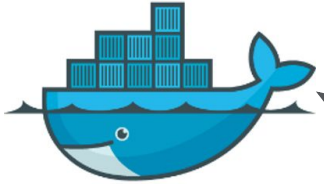
Jenkins and container management



Private SaaS Edition Architecture

Target development pipeline

```
FROM ubuntu
MAINTAINER Eugene Kalinin

# base update
RUN echo "deb http://archive.ubuntu.com/ubuntu precise main universe" > /etc/apt/sources.list
RUN aptitude update
RUN apt-get -y upgrade

# node.js
RUN apt-get install python-software-properties python g++ make -y
RUN add-apt-repository ppa:chris-lea/node.js
RUN apt-get update
RUN apt-get install nodejs

# mongodb
RUN apt-get install mongodb

# uptime
ADD https://github.com/fzaninotto/uptime/archive/master.zip /uptime.zip
```



Maven

Jenkinsfile

```
standardBuild {
  environment = 'golang:1.5.0'
  mainScript = '''
go build -v hello-world.go
'''
  postScript = '''
ls -l
./hello-world
'''
}
```

ECS Terminology

Clusters - groups of container instances in an AWS region

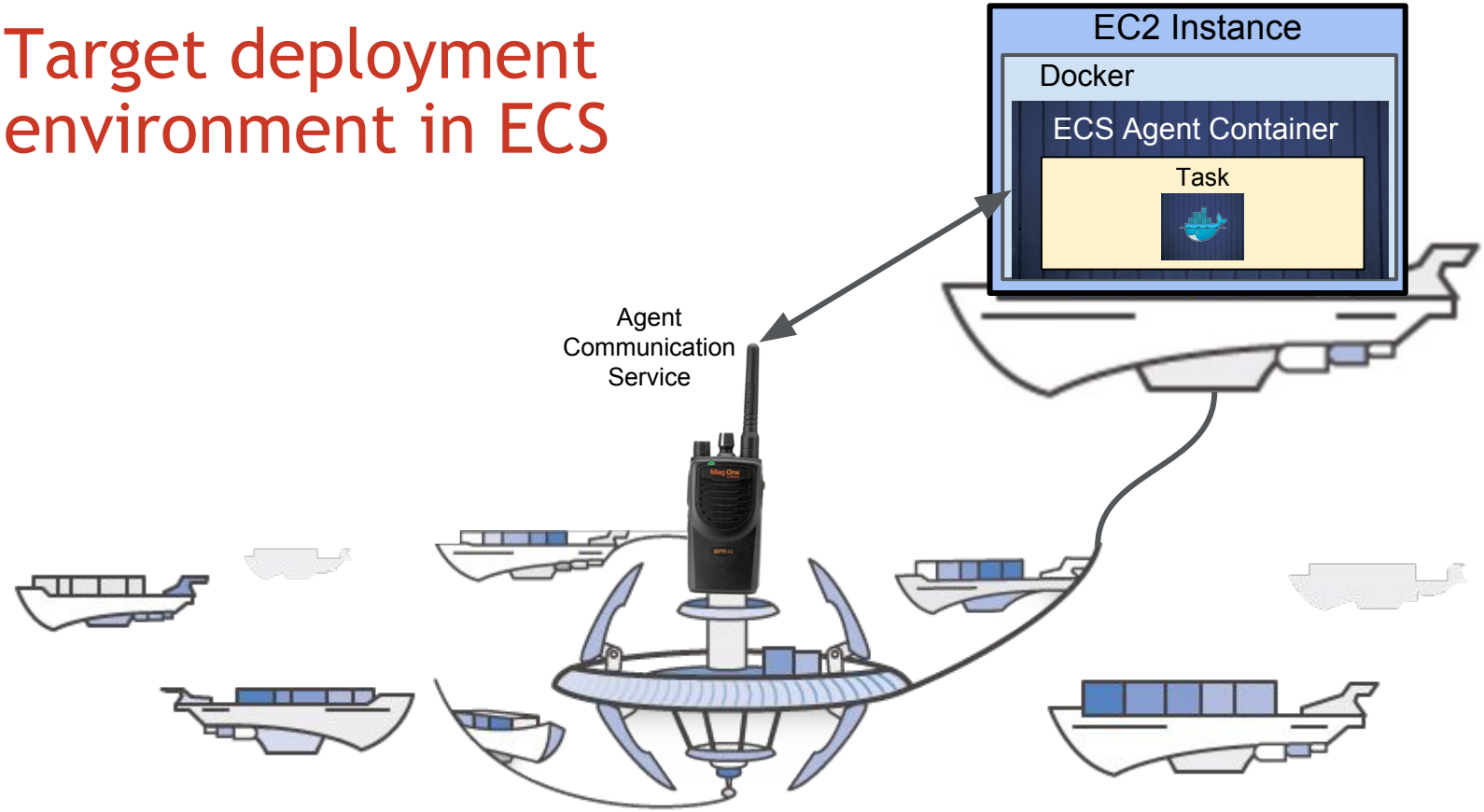
Tasks - defines what Docker container should be launched on an EC2 instance

Services - defines the desired number of tasks/containers to run simultaneously in an ECS cluster

Container agent - registers an EC2 instance to an ECS cluster

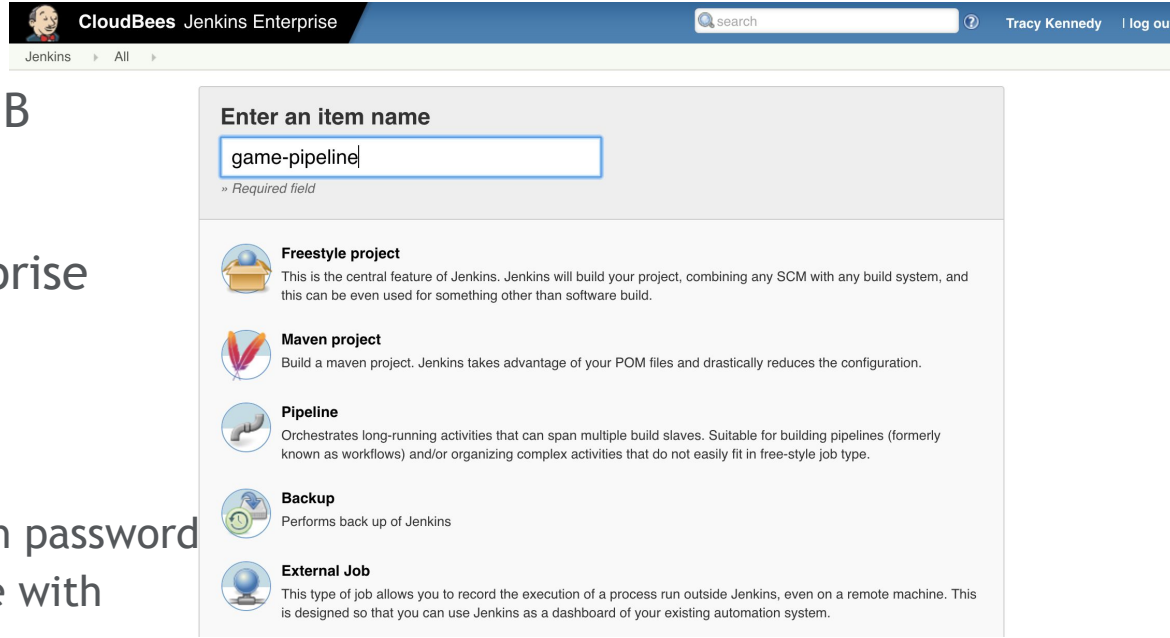
Container instances - EC2 instance registered to an ECS cluster

Target deployment environment in ECS

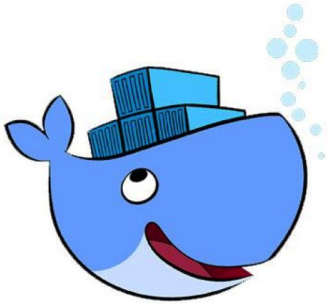


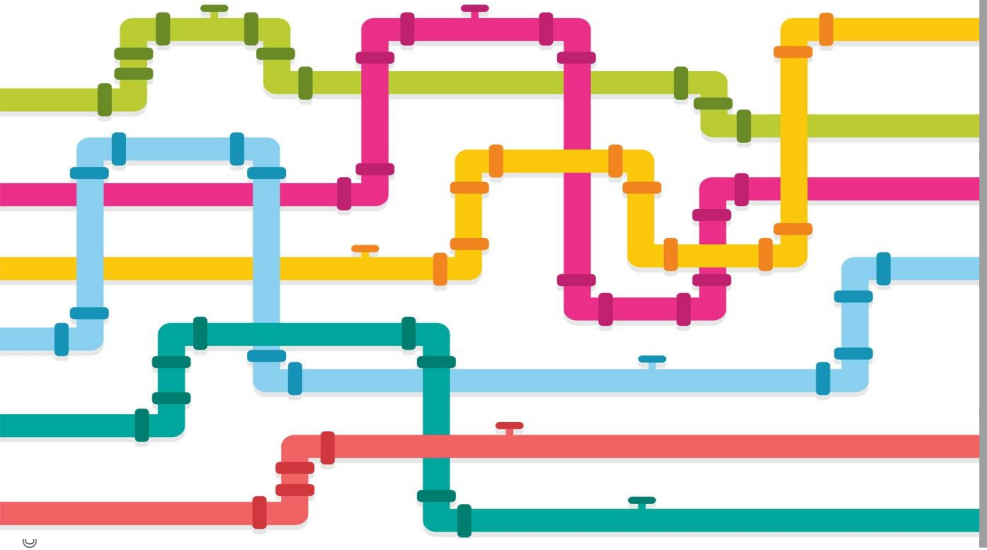
Setting up Jenkins in AWS

- Master spec:
  - T2.medium - 2 vCPU, 4 GB memory
  - Amazon Linux
  - CloudBees Jenkins Enterprise 2.7.2 WAR
  - Tools installed:
    - Git - 2.7.4-1.47.amzn1
  - Credentials set up:
    - Github - Username with password
    - Docker Hub - Username with password
    - EC2 - SSH username with private key



Plugins Used





- CloudBees Docker Pipeline Plugin
- CloudBees Docker Build and Publish Plugin
- CloudBees AWS CLI Plugin*
- Pipeline Plugin
- Docker Plugin
- Pipeline Stage View Plugin
- Git Plugin
- Docker Hub Plugin

Setting up Jenkins in AWS

- Agent spec:
  - AMI ID: ami-07b3b36d
  - T2.medium - 2 vCPU, 4 GB memory
  - Ubuntu
  - Installed tools:
    - Docker - 1.10.3 Ubuntu package
    - Maven - 3.0.5 Ubuntu package
    - Git - 2.7.3 Ubuntu package
    - Java 8 - OpenJDK 1.8.0_72-internal

cloudbees-jenkins-agent-1458232747 - ami-07b3b36d

Root device type: ebs Virtualization type: hvm

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Tim
	ec2	Linux (amd64)	In sync	6.41 GB	0 B	6.41 GB	2532m
	master	Linux (amd64)	In sync	6.28 GB	0 B	6.28 GB	0m
	Data obtained	42 sec	42 sec	42 sec	42 sec	42 sec	42

Connected to Jenkins with an SSH Connector

Label = “ec2”

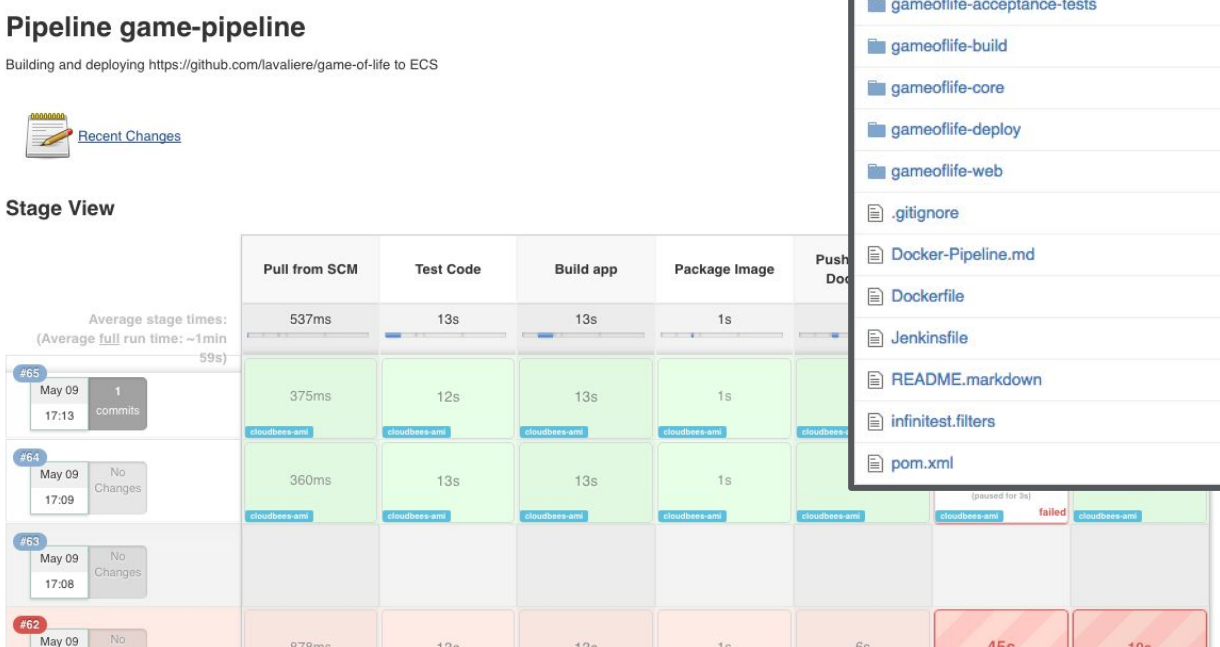
Docker daemon to start on boot

Ensure “ec2-user” in “docker” group

- Connect agent to Jenkins only *after* doing all this!

The Jenkinsfile

- Kept in GitHub project's root directory, Jenkins executes
- Scripts out the target deployment pipeline
- Triggered by pipeline job



gameoflife-acceptance-tests	webapp root context should be "/" and not "/gameoflife"
gameoflife-build	Updated versions
gameoflife-core	Restored working tests
gameoflife-deploy	Tidied up code
gameoflife-web	Add Dockerfile for gameoflife.war onto of Tomcat8
.gitignore	Moved web tests into a separate module.
Docker-Pipeline.md	changed dockerfile for ecs demo
Dockerfile	changed dockerfile for ecs demo
Jenkinsfile	final fixes for deployment
README.markdown	Changed README to markdown
infinitest.filters	First commit
pom.xml	Updated dependencies

The Dockerfile

- Kept in GitHub project's root directory - copied into the Jenkins workspace
- Copies built artifact into a Docker container running Tomcat 8
- Triggered by pipeline step

Branch: master ▾ game-of-life / Dockerfile Find file Copy path

lavaliere changed dockerfile for ecs demo

2 contributors

8 lines (5 sloc) | 177 Bytes

```
FROM tomcat:8-jre8

RUN rm -rf /usr/local/tomcat/webapps/*

COPY /gameoflife-web/target/gameoflife.war /usr/local/tomcat/webapps/ROOT.war

EXPOSE 9090
CMD ["catalina.sh", "run"]
```

```
stage 'Build app'
//Running the maven build and archiving the war
sh 'mvn install'
archive 'target/*.war'

stage 'Package Image'
//Packaging the image into a Docker image
def pkg = docker.build ('lavaliere/game-of-life', '.')
```

Target Docker registry - Docker Hub

- Docker pipeline step “withRegistry” specifies target registry for push
- “push” step sends built image to target registry, tagged “docker-demo”

PUBLIC REPOSITORY

lavaliere/game-of-life ☆

Last pushed: 23 minutes ago

Repo Info | Tags | Collaborators | Webhooks | Settings

Tag Name		
docker-demo	140 MB	23 minutes ago
latest	140 MB	2 hours ago

```
stage 'Push Image'
//Pushing the packaged app in image into DockerHub
docker.withRegistry ('https://index.docker.io/v1/',
    sh 'ls -lart'
    pkg.push 'docker-demo'
}
```

Target flow in ECS

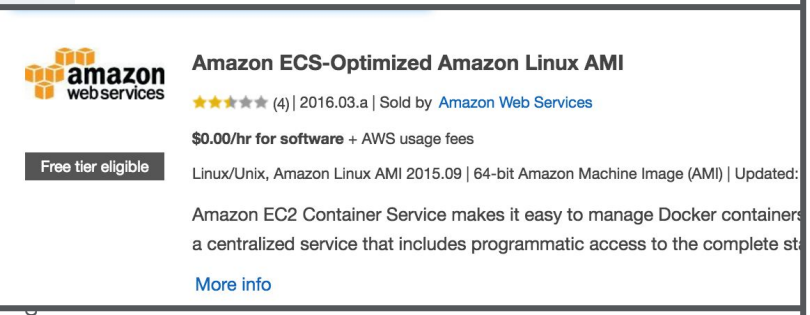
- ECS steps run in an agent container (cloudbees/java-build-tools:0.0.7.1) with AWS CLI
  - AWS credentials in Jenkins
  - Wraps AWS-related services
  - Calls ECS agent to trigger task that pulls the newly built app image
  - Checks ECS service status
  - “update-service” = “docker stop”
  - Manually gated deployment

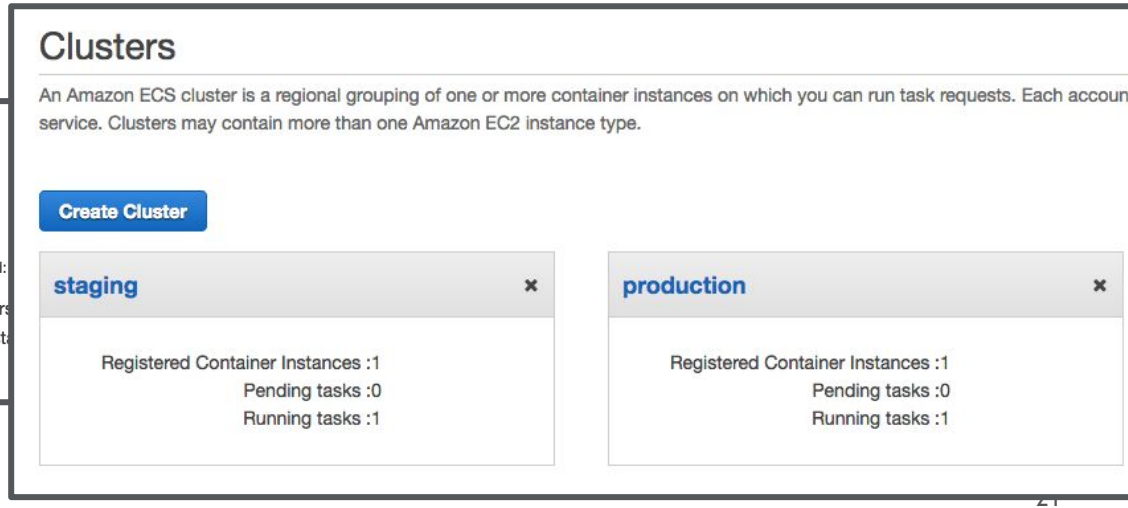
```
#===================================
# AWS CLI
#===================================
RUN pip install awscli

# compatibility with CloudBees AWS CLI Plugin which expects pip
RUN mkdir -p /home/jenkins/.local/bin/ \
  && ln -s /usr/bin/pip /home/jenkins/.local/bin/pip \
  && chown -R jenkins:jenkins /home/jenkins/.local
```

```
26  stage 'Stage image'
27  //Deploy image to staging in ECS
28  def buildenv = docker.image('cloudbees/java-build-tools:0.0.7.1')
29  buildenv.inside {
30    wrap([$class: 'AmazonAwsCliBuildWrapper', credentialsId: '20f6b2e4-7fbe-4655-8b4b-9842ec81bce2', defaultRegion: 'us-east
31        sh "aws ecs update-service --service staging-game  --cluster staging --desired-count 0"
32        timeout(time: 5, unit: 'MINUTES') {
33            waitUntil {
34                sh "aws ecs describe-services --services staging-game  --cluster staging  > .amazon-ecs-service-status.json"
35
36                // parse `describe-services` output
37                def ecsServicesStatusAsJson = readFile(".amazon-ecs-service-status.json")
38                def ecsServicesStatus = new groovy.json.JsonSlurper().parseText(ecsServicesStatusAsJson)
39                println "$ecsServicesStatus"
40                def ecsServiceStatus = ecsServicesStatus.services[0]
41                return ecsServiceStatus.get('runningCount') == 0 && ecsServiceStatus.get('status') == "ACTIVE"
42            }
43        }
44        sh "aws ecs update-service --service staging-game  --cluster staging --desired-count 1"
45        timeout(time: 5, unit: 'MINUTES') {
46            waitUntil {
47                sh "aws ecs describe-services --services staging-game --cluster staging > .amazon-ecs-service-status.json"
48
49                // parse `describe-services` output
50                def ecsServicesStatusAsJson = readFile(".amazon-ecs-service-status.json")
51                def ecsServicesStatus = new groovy.json.JsonSlurper().parseText(ecsServicesStatusAsJson)
52                println "$ecsServicesStatus"
```

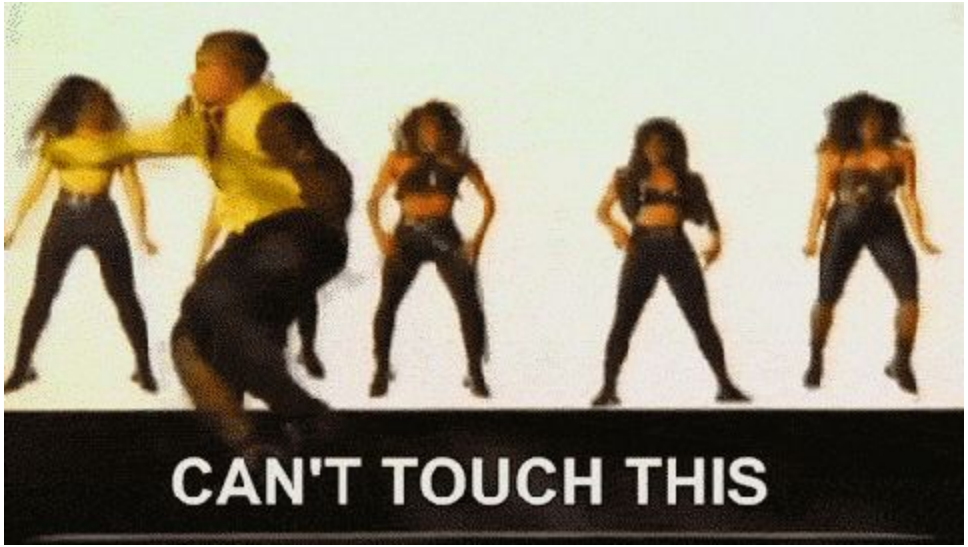
Target environments in ECS

- 2 clusters: “production” and “staging” for the webapp
  - Service per cluster
    - Task definitions for pulling the app image
  - EC2 instance per cluster
    - “ECS-optimized”
    - IAM roles





Demo time!



ECS: https://console.aws.amazon.com/ecs/home?region=us-east-1#/clusters
Jenkins: http://52.72.46.249:8080/job/game-pipeline/
Github: https://github.com/lavaliere/game-of-life
Docker Hub: https://hub.docker.com/r/lavaliere/game-of-life/
Staging: http://52.200.92.100/
Production: http://52.202.249.4/

Curious to try this out?

https://www.cloudbees.com/get-started

CloudBees

Search | Blog | About | Login | Free Trial

Jenkins | Products | Support | Continuous Delivery | Customers | Partners | Resources

Home / Get Started with the CloudBees Jenkins Platform!

Get Started with the CloudBees Jenkins Platform!

With this 14-day free trial, you will get:

- CloudBees Jenkins Platform - Enterprise Edition in your choice of installation type: Docker image, on-premise installation, or in the Amazon or Azure cloud environments
- Technical assistance from CloudBees engineers
- A full-feature trial license valid for 14 days

Choose your installation type:

Docker
Run CloudBees Jenkins Platform in a pre-configured Docker image

On-Premise
Install CloudBees Jenkins Platform manually for your operating system

AWS or Azure
Deploy CloudBees Jenkins Platform on AWS or Azure infrastructure

Platform Features

- ✓ Jenkins Open Source Continuous Delivery Engine
- ✓ Advanced Enterprise Features for scalability, security, and manageability
- ✓ Expert Technical Support

- We've made it easy to get started with a pre-configured CloudBees Jenkins Platform image

Jenkins World

Sept 13-15 in Santa Clara, CA

Register with the code JWTKENNEDY for 20% off any ticket

Talks, training, hackathon, certification

Jenkins goodies

- **Join our meetup groups!**
  - San Francisco
  - Lima
  - Tel Aviv
  - London
  - Bangalore
  - Washington DC
  - Boston
  - Dallas
  - Barcelona
  - Los Angeles
  - Paris
  - Amsterdam
  - St. Petersburg
  - Guadalajara
- Toulosse
- Seattle
- Atlanta
- Hamburg
- Seville
- New York
- Moscow
- Austin
- Boulder
- Breizh
- Zurich
- Sydney
- Milano
- Hengelo
- Albuquerque

Additional Resources

- Jenkins Pipeline
- How to point Jenkins to a custom registry (e.g. local)
- Setting up Jenkins slaves on AWS
- Game of Life pipeline deployment to ECS
- AWS's approach to building a pipeline with Jenkins and ECS
- Creating an ECS cluster with ELB and autoscaling
- Building multiple containers with Docker Compose and pipeline
- Blue-green ECS deployments