Thinking Inside the Container: A Continuous Delivery Story

Jenkins World
2016
Who’s This Guy?

Jenkins World 2016
OUR MISSION

WE ASPIRE TO BE THE MOST PLAYER-FOCUSED GAME COMPANY IN THE WORLD
A Quick Note on Riot Games
The Scale of League

MORE THAN 67 MILLION MONTHLY ACTIVE PLAYERS

MORE THAN 27 MILLION DAILY ACTIVE PLAYERS

7.5 MILLION PEAK CONCURRENT PLAYERS
1.25 Million Builds a Year
Thinking Inside the Container
10,000 - 14,000 Containers A Week
120 Jobs An Hour
A Containerized Build Farm

JENKINS

SSH

Docker API

SWARM

Docker API

BUILD HOSTS

DOCKERHOST

DOCKERHOST

DOCKERHOST

DOCKERHOST

DOCKERHOST

CentOS 7.2/Docker 1.10.3

cAdvisor

Docker-GC

Container Metrics

4 Core/32GB RAM/120GB LVS

DRYDOCK

Docker API
STORY TIME!

(aka. Presenting the Problem)
650+ Builds an Hour
90+ Build Slaves
Expecting 100 More!
Story Time....

Engineers → Ticket → Build Team
Story Time....

Engineers

Build Team
What Did We Want?

- Teams needed to move fast
- Products had to own their stack
- Configuration as code
Maybe We Want...
“Along Came a Whale”

(More Story Time!)
Yeah, if you could put Docker in the Build Farm

That’d be great.
Oh Look! Another Way to Deploy!

How Standards Proliferate:
(See: A/C chargers, character encodings, instant messaging, etc.)

**Situation:**
There are 14 competing standards.

14?! RIDICULOUS!
We need to develop one universal standard that covers everyone's use cases. Yeah!

**Soon:**

Situation:
There are 15 competing standards.
Docker For Newbz
Jenkins Primer
Jenkins Primer

JENKINS

Master

“Win32” + “Java” + ”TeamA”

“Cent7” + “Java” + ”TeamB”

“Cent7” + “GoLang” + ”TeamB”

Slaves
Jenkins Primer

Build Job

“Win32” + “Java” + “TeamA”

Master

JENKINS

“Win32” + “Java” + “TeamA”

“Cent7” + “Java” + “TeamB”

“Cent7” + “GoLang” + “TeamB”

Slaves
Jenkins Primer

Build Job
“Win32” + “Java” + “TeamA”

Build Job

Build Job

Build Job

Queue

Master

JENKINS

Slaves

“Win32” + “Java” + “TeamA”

“Cent7” + “Java” + “TeamB”

“Cent7” + “GoLang” + “TeamB”
Jenkins Primer

Build Job
“Win32” + “Java” + “TeamA”

Queue
Build Job
Build Job
Build Job
Build Job

JENKINS

Master

Slaves

“Win32” + “Java” + “TeamA”

“Cent7” + “Java” + “TeamB”

“Cent7” + “GoLang” + “TeamB”

#JenkinsWorld
What Did We Need?

- CONTAINER RUNNING AS A BUILD SLAVE
- WAY TO HAVE ENGINEERS PROVIDE DOCKERFILES
- HOW TO SCALE IT
- PROFIT?
FROM centos:7
MAINTAINER Maxfield Stewart

# Install Essentials
RUN yum update -y && \
    yum clean all

# Install Packages
RUN yum install -y git && \
    yum install -y wget && \
    yum install -y openssh-server && \
    yum install -y java-1.8.0-openjdk && \
    yum install -y sudo && \
    yum clean all

# gen dummy keys, centos doesn't autogen them
RUN /usr/bin/ssh-keygen -A

# Set SSH Configuration to allow remote login
RUN sed -ri 's/^session\s+required\s+pam_loginuid\n    $/session\s+required\n        pam_loginuid\n        pam_secure砜/sh

# Add the jenkins user to sudoers
RUN echo "jenkins  ALL=(ALL)  ALL" >> etc/sudoers

# Set Name Servers
COPY /files/resolv.conf /etc/resolv.conf

# Create Jenkins User
RUN useradd jenkins -m -s /bin/bash

# Add public key for Jenkins login
RUN mkdir /home/jenkins/.ssh
COPY /files/authorized_keys /home/jenkins/.ssh

RUN chown -R jenkins /home/jenkins
RUN chgrp -R jenkins /home/jenkins
RUN chmod 600 /home/jenkins/.ssh/authorized_keys
RUN chmod 700 /home/jenkins/.ssh

# Set Name Servers
COPY /files/resolv.conf /etc/resolv.conf
Add a Bit of Secret Sauce...

<table>
<thead>
<tr>
<th>Base Slave Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>Universal Riot Tools</td>
</tr>
<tr>
<td>Product/App Specific Build Stuff</td>
</tr>
</tbody>
</table>
A Real Example

Dockerfile

FROM registry.rcluster.io/pipe/centosbuildslave:v4.0.2
MAINTAINER Competitive-Dev <competitive-dev@riotgames.com>

# Build Slave For Loot

# This Dockerfile defines the build environment for Loot-server. It contains a:

# Install Yum stuff and Mysql Stuff
RUN yum makecache && \
    yum install -y git && \
    yum install -y unzip && \
    yum install -y bzip2 && \
    yum install -y http://www.percona.com/downloads/percona-release/redhat/0.1 \
    yum install -y Percona-Server-client-55 && \
    yum install -y Percona-Server-devel-55 && \
    yum install -y Percona-Server-server-55 && \
    yum install -y Percona-Server-shared-55 && \
    yum clean all

# Install the docker and setup sudo
RUN wget https://get.docker.com/builds/linux/x86_64/docker-1.10.3 && \
    mv docker-1.10.3 /usr/bin/docker && chmod +x /usr/bin/docker && \
    echo "jenkins ALL=(ALL) NOPASSWD: /usr/bin/docker" >> /etc/sudoers

# Install Java Oracle 8
ENV j8DownloadUrl http://download.oracle.com/otn-pub/java/jdk/8u73-b02/jdk-8u73-linux-x64.bin \
ENV j8Short 8u73 \
ENV j8Long 1.8.0_73 \
ENV j8Unzip /usr/bin/unzip

# Install Maven
RUN wget http://mirror.cogentco.com/pub/apache/maven/maven-3/3.1.1/binaries \
    tar xvf apache-maven-3.1.1-bin.tar.gz -C /usr/local && \
    ln -s /usr/local/apache-maven-3.1.1 /usr/local/maven && \
    rm apache-maven-3.1.1-bin.tar.gz && \
    mv /usr/local/maven/conf/settings.xml /usr/local/maven/conf/settings_development.xml

# Install Ant
RUN wget http://mirror.nexcess.net/apache/ant/binaries/apache-ant-1.9.6-bin.tar.gz \
    tar xvf apache-ant-1.9.6-bin.tar.gz -C /usr/local && \
    ln -s /usr/local/apache-ant-1.9.6 /usr/local/ant && \
    rm apache-ant-1.9.6-bin.tar.gz && \
    chown -R jenkins:jenkins /usr/local/apache-ant-1.9.6

# Make sure the jenkins user as clear sudo rights to mysql
RUN sed -i '/Defaults requiretty/#Defaults requiretty/' /etc/sudoers \
    echo "jenkins ALL=(ALL) NOPASSWD: /usr/bin/mysql" >> /etc/sudoers \
    echo "jenkins ALL=(ALL) NOPASSWD: /bin/mysql_safe" >> /etc/sudoers \
    echo "jenkins ALL=(ALL) NOPASSWD: /usr/bin/mysqladmin" >> /etc/sudoers \
    echo "jenkins ALL=(ALL) NOPASSWD: /usr/bin/mysql_install_db" >> /etc/sudoers \
    echo "jenkins ALL=(ALL) NOPASSWD: /usr/bin/mysql" >> /etc/sudoers \
    echo "jenkins ALL=(ALL) NOPASSWD: /usr/bin/kill" >> /etc/sudoers

# Get the mysql start script in for easier use
COPY scripts/mysqlstart.sh /home/jenkins/mysqlstart.sh
RUN chmod +x /home/jenkins/mysqlstart.sh && \
    chown jenkins:jenkins /home/jenkins/mysqlstart.sh

# Copy bash profile to get ENV settings correct
COPY config/.bash_profile /home/jenkins/.bash_profile
RUN chown jenkins:jenkins /home/jenkins/.bash_profile && \
    chmod +x /home/jenkins/.bash_profile
Provisioning and Plugins
Of Whales and Plugins

DOCKER PLUGIN

MESOS PLUGIN

KUBERNETES PLUGIN
Of Whales and Plugins

DOCKER PLUGIN

MESOS PLUGIN

KUBERNETES PLUGIN

KEEP CALM
Keep It
Simple
Stupid
## Quick Look

<table>
<thead>
<tr>
<th>Cloud</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Docker</td>
<td>Docker Swarm</td>
</tr>
<tr>
<td>Docker URL</td>
<td><a href="http://tas1swarm006.tfs1.riotgames.com:4444">http://tas1swarm006.tfs1.riotgames.com:4444</a></td>
</tr>
<tr>
<td>Credentials</td>
<td>- none -</td>
</tr>
<tr>
<td>Connection Timeout</td>
<td>5</td>
</tr>
<tr>
<td>Read Timeout</td>
<td>35</td>
</tr>
<tr>
<td>Container Cap Images</td>
<td>100</td>
</tr>
</tbody>
</table>

| Docker Template | docker.s.riotgames.com/chai/ehabberd-buildslave:kg |
| Docker Image |  |
| Instance Capacity |  |
| Remote File System Root | home/jenkins |
| Labels | social-infrastructure-buildslave |
| Usage | Only build jobs with label restrictions matching this node |
| Launch method | Docker SSH computer launcher |
| Credentials | jenkins (Jenkins with a private key) |
| Remote FS Root Mapping |  |
| Remove volumes |  |
| Pull strategy | Pull once and update latest |
Provisioning and Plugins

Image Name

Label

API
/*
   = ADDING NEW DOKER HOST CODE =
   */

// injected variables from jenkins envinject:
// cloudName existing or current docker cloud name
// label desired label for build node
// image image location (eg. dockers.tf.riotgames...)

def dockerPlugin = Jenkins.instance.pluginManager.getPlugin("docker-plugin").getPlugin()
def dockerServers = dockerPlugin.getServers()
def cloudExists = false
def labelExists = labelInCloud(dockerServers)

if (label.isEmpty()) {
    println("Label cannot be empty. Slave not added")
} else if (labelExists) {
    println("Specified label already exists. Slave not added.")
} else {
    dockerServers.each() { dockerServer ->
        // label checking
        ArrayList<DockerCloud> templates = dockerServer.templates
        if (dockerServer.name.toLowerCase().equals(cloudName.toLowerCase())) {
            cloudExists = true
            println("Adding "+ label + " to cloud["+ cloudName + "]...")
            def retentionStrategy = new DockerOnceRetentionStrategy(idleMinutes)
            def sshConnector = new SSHConnector(22, credentialsId, jvmOptions, javaPath, prefixStartSlaveCmd,
                suffixStartSlaveCmd, launchTimeoutMinutes * 60)
        }
    }
}
We Created A Monster

MODEL CALCULATIONS

"Garbage In-garbage Out" Paradigm

GARBAGE DATA → PERFECT MODEL → GARBAGE RESULTS

PERFECT DATA → GARBAGE MODEL → GARBAGE RESULTS
We Need to Inspect Our Containers
We Need to Inspect Our Containers

```
./harbormaster --registry_user "$REGISTRY_USER" --registry_token "$REGISTRY_TOKEN" --jenkins_user $USER --jenkins_pass $PASSWORD --jenkins --dockerhost $TEST_DOCKER_HOST "$IMAGENAME" "$CLOUDNAME" "$LABELNAME"
```

**Overall Results:** 7 out of 7 Passed

- **Container CMD:** Passed
  - Verify that the container has `automated` as its CMD.

- **Container Run:** Passed
  - Verify that after the container starts, it stays running. If it doesn’t, it can’t work as a build slave.

- **Java:** Passed
  - Verify that Java 1.7 is the primary Java. Jenkins requires this minimum to control the node.
    - Your Version: 1.7.0.101
    - Minimum Expected Version: 1.7

- **SSH:** Passed
  - Verify that you can SSH in to your container as a Jenkins user. Jenkins needs this in order to talk to the slave. Jenkins will also create a workspace using the jenkins user.

- **Write Permissions:** Passed
  - Verify that the user Jenkins exists and has write permissions to /home/jenkins. This is where Jenkins creates a workspace and puts files it generates (/home/jenkins/workspace/<job name>)

- **SCM:** Passed
  - Verify that a version of Git is installed. This is needed to use Jenkins Git features to sync with Git. Perforce is currently not supported.
    - Your Git Version: version 2.8.1

- **Label:** Passed
  - Verify that your label is unique. We check to make sure your label doesn’t already exist on the farm. You restrict your builds to this label.
    - Specified label and image already exists therefore slave not added. No configuration update needed. Latest will be pulled if all tests succeeded
Are We Done?

Donkey: Are we there yet?
Shrek: No
Donkey: Are we there yet?
Fiona: No not yet!
Donkey: Are we there, yet?
Shrek: Yes!
Donkey: Really?
Shrek: NO!!!

- Shrek 2
But It Needs to Scale!
If Only...
If Only...
But It Needs to Scale!

Build hosts:
- DOCKERHOST
- DOCKERHOST
- DOCKERHOST
- DOCKERHOST

JENKINS -> SSH -> Docker API -> SWARM -> Docker API

CentOS 7.2/Docker 1.10.3
- cAdvisor
- Docker-GC
- Container Metrics

4 Core/32GB RAM/120GB LVS
Putting It All Together

Image

REGISTRY

Engineer
Putting It All Together

Image Name + Jenkins Label

Engineer
Putting It All Together

Engineer

Runs Tests

REGISTRY

HARBOR MASTER
Putting It All Together

Image Pulled to Swarm
Putting It All Together

Groovy API Called

Jenkins

REGISTRY

HARBOR MASTER

SWARM

Engineer

DOCKERHOST

DOCKERHOST

DOCKERHOST
node ('Awesome-Build-Label') {

git branch: "master", url: "git@github.com:maxfields2000/awesome.git"

// This script could come from source control!
sh './buildme.sh'

}
DOCKERCEPTION

WE HEARD YOU LIKE DOCKER

SO WE PUT A DOCKER IN YOUR DOCKER
Where to Build Containers?

JENKINS -> SWARM

JENKINS

SSH

Docker API

SWARM

Docker API

BUILD HOSTS

DOCKERHOST

DOCKERHOST

DOCKERHOST

DOCKERHOST

DOCKERHOST

DRYDOCK

Docker API

CentOS 7.2/Docker 1.10.3

cAdvisor

Docker-GC

Container Metrics

4 Core/32GB RAM/120GB LVS
Mai Tai’s On the Beach

Build Engineer
How Do You Actually Build It?
How Do You Actually Build It?

(engineering.riotgames.com)
https://github.com/maxfields2000/dockerjenkins_tutorial
You Make This Sound Simple.

Step 1
Draw two circles

Step 2
Draw the rest of the owl
Lesson 1 - Docker Isn’t “Simple”

DOCKERFILES REQUIRE DECENT SYSTEMS ADMIN KNOWLEDGE
DOCKER IMAGES THAT BUILD DOCKER IMAGES IS DOCKERCEPTION
DOCKER “VOODOO AND BLACK MAGIC” QUESTIONS
Lesson 2 - Containers != VM’s

- CANNOT MOUNT REMOTE FILE SYSTEMS
- EPHEMERAL CONTAINERS DON’T MAINTAIN STATE
- HAVE RULES FOR DOCKERFILE VS RUN TIME
Lesson 3 - Garbage Collection

DOCKER BUILD

DOCKER PULL

DOCKER RUN

VOLUME
Lesson 4 - Maintenance/Failure

- Pull Hosts On/Offline
- Update All Images
- Rolling Re-Wraps
Lesson 5 - How Will You Upgrade?

BRACE YOURSELVES

DEPLOYMENT TO PRODUCTION IS COMING.
<table>
<thead>
<tr>
<th>Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>WHAT SHOULD YOU PUT IN YOUR BASE IMAGE?</td>
</tr>
<tr>
<td>SSH KEYS IN YOUR REGISTRY?</td>
</tr>
<tr>
<td>PASSWORDS IN YOUR SOURCE CODE?</td>
</tr>
<tr>
<td>PASSING EVERYTHING IN AS A ENVIRONMENT VARIABLE?</td>
</tr>
</tbody>
</table>
“What if we don’t change at all ... and something magical just happens?”
Over 1200 New Build Jobs Created
30% of All Environments Are Containers
Environment Change/Create/Fix

Tickets Dissappeared
We No Longer Have
“Just” A Build Team
QUESTIONS?