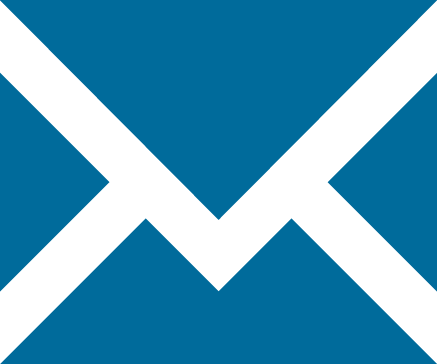
|  |
| --- |
| Macintosh HD:Users:hrente:Desktop:QA_branding:Logo:QA Consulting Logo_solo-01.jpg |
| Lecture Exercise  Booklet |
| Continuous integration and devops |

#### Prepared by

Aaron Mulholland

07783706895

[Aaron.Mulholland@qa.com](mailto:Aaron.Mulholland@qa.com)

Contents

[DevOps – Week One Tasks 2](#_Toc477341398)

[Introduction to Continuous Integration and Deployment 2](#_Toc477341399)

[Task 1 – Creating Example Systems 3](#_Toc477341400)

[Linux 3](#_Toc477341401)

[Task 1 – Your first Virtual Machine 4](#_Toc477341402)

[Task 2 – Terminal Exploration 4](#_Toc477341403)

[Task 3 – Creating a Script file 4](#_Toc477341404)

[Task 4 – Configuring the Linux Environment 4](#_Toc477341405)

[Task 5 – Using IPTables 5](#_Toc477341406)

[Task 6 – Vagrant Scripting 5](#_Toc477341407)

[Task 7 – Repository Management 5](#_Toc477341408)

[Tooling 6](#_Toc477341409)

[Task 1a – Setting up Jenkins 6](#_Toc477341410)

[Task 1b – Setting up Bamboo 6](#_Toc477341411)

[Task 2 – Setting up Jira 7](#_Toc477341412)

[Task 3 – Configuring Jira and Git 8](#_Toc477341413)

[Task 4 – Installing Nexus 8](#_Toc477341414)

[Task 4 – Installing Zabbix 8](#_Toc477341415)

[Configuration Management – Week Two Tasks 9](#_Toc477341416)

[Task 1 – Puppet Quest 9](#_Toc477341417)

[Provisioning with Puppet 9](#_Toc477341418)

[Task 2 – Setting up your Puppet Instances 9](#_Toc477341419)

[Task 3 – Creating your first Puppet Module (Java) 9](#_Toc477341420)

[Task 4 – More nodes & modules 9](#_Toc477341421)

[Provisioning with Ansible 10](#_Toc477341422)

[Task 2 – Setting up to use Ansible 10](#_Toc477341423)

[Task 3 – Managing your hosts. 10](#_Toc477341424)

[Task 4 – Write your Playbook. 10](#_Toc477341425)

[Task 5 – More Nodes & Playbooks 10](#_Toc477341426)

[Docker and Kubernetes – Week Two Tasks 11](#_Toc477341427)

[Task 1 – Install Docker on your System 11](#_Toc477341428)

[Task 2 - Deploy a Jenkins Container 11](#_Toc477341429)

[Task 3 – Create your own Dockerfile 11](#_Toc477341430)

[Task 4 – Create your own linked container 11](#_Toc477341431)

[Task 5 – Create your own docker-compose file 11](#_Toc477341432)

[Task 6 – Install Kubernetes 11](#_Toc477341433)

[Task 7 – Creating your first Single Container Pod 12](#_Toc477341434)

[Task 8 – Creating a Multi-Container Pod 12](#_Toc477341435)

[Introduction to AWS – Week Three Tasks 12](#_Toc477341436)

# DevOps – Week One Tasks

## Introduction to Continuous Integration and Deployment

This series of slides has introduced the concepts and basic principles of CI and CD and should have provided you with an initial understanding of why, how and where to apply them.

The following exercise builds on this section and aims to provide insight and experience in how you can tailor different development tools to create a Continuous Integration Server. The following image represents an example of what is expected.



### Task 1 – Creating Example Systems

In pairs, start working on a new version of the above System Example. In your examples, make sure you are using at least two of the tools in the list below and note you will need to use some in the example above. You will need to do some research into the individual tool before making use of it.

* Zabbix
* MCollective
* Vagrant
* A Database
* Puppet

## Linux

This series of tasks is presented to you to compliment the series of lectures received on Working with Linux.

The following four tasks have you work through setting up your own running Linux Ubuntu system, and configuring your environment ready for use.

We will also explore how to use the terminal, and how to use IPTables.

### Task 1 – Your first Virtual Machine

Using the new ISO file provided you are to begin setting up your first virtual machine within Virtual Box.

You need to consider the name and type of machine.

You need to set the correct network (bridged connection) and you need to specify a good amount of RAM (4096).

It is advised you set 100GB of Disk Space too.

### Task 2 – Terminal Exploration

Now that you have experienced a flavour of what exactly bash scripting is now it’s time to get some hands-on experience with working through a series of tasks.

Remember you **must** use the terminal to complete these tasks. You may need to research the correct commands to use.

First, navigate through the terminal to your home directory, and create a directory called “QAC<name>” – e.g. “QACAaron”.

Inside this directory – insert a new file, and enter some random text into this file. Save and close the file.

Print the contents of the file – without going into the file to edit it.

Now create a new directory as follows; /home/QAC<name>/MyProject/

E.g; /home/QACAaron/MyProject

Move the file you created before, from /QAC<name> to /QAC<name>/MyProject

### Task 3 – Creating a Script file

Conduct independent research into how you can create a new user using commands. Create a .sh script file that will meet the following requirements;

* Take user input for information such as user name and password
* Assign the user to the administrators (sudoers) group

You will need to pay consideration to the new user accounts home directory, bash terminal, and group.

**CHALLENGE TASK:** Write a script that upon invocation shows the time and date, lists all logged-in users, and gives the system uptime. The script then saves this information to a logfile.

### Task 4 – Configuring the Linux Environment

Now your system is set up and you have had chance to use the terminal it’s time for you to configure the environment ready for use. This includes ensuring Java and Maven are both set up and configured correctly on your system. (Using apt-get)

This is quite simple; you just need to check for the following things;

* You have admin permissions (can you execute sudo commands)
* The system is at its most updated (sudo apt-get update)

### Task 5 – Using IPTables

Your lecture session should have provided you a brief overview as to what IPTables are and how they work. Now it’s time for some practical experience with them on your Ubuntu Virtual Machine.

Using the list below (Also seen on the lecture slides), please try out each of these commands, and check that the firewall has been disabled.

One very easy check, is to try and SSH into the virtual machine – If you cannot connect, then the firewall is still active and is blocking the access. If you can, you’ve successfully disabled/set up the firewall correctly.

To connect to your machine using SSH, you can use the puTTY tool located at C:\Local Install\DevOps\PuTTY– and follow the instructions to connect as described in the lecture slides.

Work out how the following tasks can be completed with IPTables;

* Stop/disable the IPTables
  + Start/enable the IPTables
  + Add a rule to allow an application through the firewall
  + List the rules in a chain
  + Displaying the Firewall status
  + Delete the Firewall rules

Some helpful links include;

<http://www.cyberciti.biz/faq/ubuntu-start-stop-iptables-service/>

<http://ipset.netfilter.org/iptables.man.html>

### Task 6 – Vagrant Scripting

Using the new provisioning tool Vagrant, you are tasked with creating a new Ubuntu environment and CentOS environment. You will need to create a directory and call it “Vagrant Setup” – in here open Git bash and use the command, *vagrant init*, to initialise a new vagrant repository. This should add a Vagrantfile to the directory.

Modify this file to ensure you are setting up an Ubuntu box (the vagrant box will be provided to you by your trainer, so you don’t have to wait for the download) and the CentOS box.

These machines should both have;

1. A designated machine name
2. A designated hostname
3. A Static IP (Ubuntu Machine Only)
4. 4096MB RAM
5. 2 CPUs
6. Visible Desktop GUI
7. Shared folder access on both environments

You should also include a *bootstrap.sh* file which will install and configure an installation of Java, Maven and Git.

### Task 7 – Repository Management

For this task, first ensure you’ve got Git installed on your virtual machine. You can install git using the command;

*sudo apt-get install -y git*

Now, if you don’t already have one, create a GitHub account and create a new repository.

You’ll need to initialise a new repository (git init) on your windows training machine. Using code created previously or sample code, submit this all to the stack (git add) and then commit all the changes (git commit -m).

Now clone that repository onto your Ubuntu VM, using the SSH/Terminal. Navigate into the directory to ensure that the correct files are there.

Add or change something within the directory on your windows machine, and push the changes through the git repository.

On your virtual machine, make sure you pull the changes you’ve just made and ensure it all works correctly still.

Apply a second change to this repository via your Linux VM, and push the changes. Ensure the change is applied at the windows end of the repository too.

You can explore other git options at this point, make sure you’re comfortable with using this tool.

## Tooling

We have looked at several different tools over the course of the Tooling section from build management, project tracking and Version Control Systems.

### Task 1a – Setting up Jenkins

This tasks requires you to install a new instance of Jenkins to your virtual machine. You can do this either command-by-command or by writing out a .SH script to automate this process.

The following commands will prove useful for installing Jenkins;

1. *Get the files required for installation*
   1. *wget -q -O - https://jenkins-ci.org/debian/jenkins-ci.org.key | sudo apt-key add -*
   2. *sudo sh -c 'echo deb http://pkg.jenkins-ci.org/debian binary/ > /etc/apt/sources.list.d/jenkins.list'*
   3. *sudo apt-get update*
   4. *sudo apt-get install jenkins*
2. *Next login as the Jenkins user, using the following commands*
   1. *sudo su jenkins -s /bin/bash*
   2. *ssh-keygen*
3. *Next start the Jenkins Service*
   1. *sudo service jenkins start*

You should then be able to go to <http://machineip:8080>

### Task 2 – Setting up Jira

This task requires you to install a new trial instance of Jira on your virtual machine. You can do this either using an automated script or command-by-command manually.

The following commands will prove useful for installing Jenkins;

1. Obtain the files for Jira
   1. wget http://aaronmulholland.co.uk/jira.bin
2. Ensure that the file is executable using the following
   1. chmod a+x jira.bin
3. Run the executable
   1. sudo ./jira.bin

At this point we want to run through a series of options;

1. Choose Express Install
2. Choose set custom value for HTTP and Control ports
   1. HTTP Port = 8081
   2. Control Port = 8006
3. Finish the installation process

Now complete the final setup configuration by going to <http://machineip:8081> and fill out the additional setup options.

### Task 3 – Configuring Jira and Git

You’ll first need to set up a DVCS account between Jira and Git.

Click on the settings tab, and select Add-Ons. From there, click DVCS account and Click “Link Bitbucket or GitHub account” – Fill in your username and password for your git account.

You’ll need to go to Github.com – Select settings and click applications. From there select Developer Applications and register a new application.

Provide the IP and port of your Jira instance e.g. <http://ip:port> for Jira, in the Homepage URL and the Callback URL.

Then copy the Client ID to the Secret Key box in Jira, and the Client secret to the OAuth secret areas on your Jira setup page.

Select your relevant Java project, and ensure you leave Auto Link new repositories and Enable smart commits as checked.

Now that JIRA and Git have been configured to work with each other. You need to set up Jira to track the commits you make to your git repository.

Tips:

* You’ll need to create a simple issue tracking project, give it a name and Make note of the KEY generated when you create the project.
* In your created project you’ll need to then create an issue based on your user stories from JSE.
* Then add a test text file to your Repository and push this change to Jira.

Example push command on a project titled gameOfLife, with the key GEM and one issue.

*git commit –m “GEM-1 #comment Added a text File”*

### Task 4 – Installing Nexus

You’ll need to configure either manually or with a script, the installation of Nexus on your virtual machine.

You will need to conduct research into how this works.

### Task 4 – Installing Zabbix

Monitoring your resources is crucial in the world of DevOps. Install and configure Zabbix so you can monitor what’s happening on your nodes.

You will need to conduct research into how this works.

# Docker and Kubernetes – Week Two Tasks

### Task 1 – Install Docker on your System

You’re now tasked with installing docker onto your AWS Virtual Machine. Use the Docker-toolbox to install this system.

Be sure to unselect (Kitematic as you won’t need this and Virtualbox as it should already be installed)

$ wget -qO- https://get.docker.com/ | sh

$ sudo usermod -aG docker ubuntu

### Task 2 - Deploy a Jenkins Container

Using the local registry, you need to pull the Jenkins container and then run it locally. Take a moment to explore this Jenkins container. See what you can do with it.

$ docker pull jenkins

$ docker run -p 8080:8080 --name=jenkins-master -d jenkins

### Task 3 – Create your own Dockerfile

You should now consider how to make your own Dockerfile. Your task is to deploy a java application within a virtualised environment using the base ubuntu image.

What should be in the file? What order should each instruction be in?

FROM ubuntu:16.04

#Always update your running system

# Install OpenJDK-8

RUN apt-get update && \

apt-get install -y openjdk-8-jdk && \

apt-get install -y ant && \

apt-get clean;

# Fix certificate issues

RUN apt-get update && \

apt-get install ca-certificates-java && \

apt-get clean && \

update-ca-certificates -f;

# Setup JAVA\_HOME -- useful for docker commandline

ENV JAVA\_HOME /usr/lib/jvm/java-8-openjdk-amd64/

RUN export JAVA\_HOME

$ sudo docker build -t ubuntu:16.04 .

$ docker run -it ubuntu:16.04 bash

### Task 4 – Create your own linked container

Create a container for a mysql database. Store the data in a volume.

sudo docker run --detach --name=test-mysql --env="MYSQL\_ROOT\_PASSWORD=mypassword" --publish 6603:3306 mysql #runs

Then create a container for a PHP Project. You should link this to the database container.

sudo docker run --detach --name test-wordpress --link test-mysql:mysql -p 8080:80 -d wordpress #install wordpress link to Mysql and assign to port 8080

### Task 5 – Create your own docker-compose file

Rewrite the linked container exercise (task 4) so that it uses docker-compose. This should mean we can start both containers using the simple docker-compose up command.

|  |
| --- |
|  |
| |  | | --- | | version: '2' | |  |  | |  | services: | |  | db: | |  | image: mysql:5.7 | |  | volumes: | |  | - db\_data:/var/lib/mysql | |  | restart: always | |  | environment: | |  | MYSQL\_ROOT\_PASSWORD: somewordpress | |  | MYSQL\_DATABASE: wordpress | |  | MYSQL\_USER: wordpress | |  | MYSQL\_PASSWORD: wordpress | |  |  | |  | wordpress: | |  | depends\_on: | |  | - db | |  | image: wordpress:latest | |  | ports: | |  | - "8000:80" | |  | restart: always | |  | environment: | |  | WORDPRESS\_DB\_HOST: db:3306 | |  | WORDPRESS\_DB\_USER: wordpress | |  | WORDPRESS\_DB\_PASSWORD: wordpress | |  | volumes: | |  | db\_data: | |
|  |

### Task 6 – Install Kubernetes

In groups of two or three, you are tasks with beginning the setup and installation of your own instance of Kubernetes. You will need to consider the environment this is being installed on, whether you’re doing it in AWS or locally through vagrant.

The following links will help

<https://kubernetes.io/docs/getting-started-guides/ubuntu/>

### Task 7 – Creating your first Single Container Pod

You’re tasked with writing a Kubernetes Pod configuration which will launch a Jenkins server in a container, using a docker image. You’ll need to be sure to expose the right ports so that you can access this container once it’s launched.

### Task 8 – Creating a Multi-Container Pod

You’re tasked with writing a Kubernetes Pod configuration which will launch multiple containers that have a shared volume.

If you launch something with a web interface (i.e Jenkins/JIRA) then be sure to expose the relevant ports, so you can access the dashboard.

AWS

### Task 1 – Create an EC2 Instance

Your first task is to login to the EC2 Console at the following link;

<https://qaconsulting.signin.aws.amazon.com/console> – Your trainer will give you your username and password.

Spend a few moments navigating around the console. Get to know where certain items are.

You should now look to launching your first EC2 instance. Your trainer just demoed this task, so see if you can remember.

Create an EC2 instance based on the following Specs;

* Ubuntu Server 16.04
* 1 CPU / 2GB Memory
* Hard disk size of 20GB
* Name Tag – Academy{date}:Name (e.g. Academy2305:Jacob)
* Academy Security Group
* Network – AcademyVPC
* Launch using the *AcademyKeyPair*

### Task 2 – Using the CLI

You will first need to install the Command Line interface on your system. Your trainer should provide you with this installation file.

You can confirm the successful installation by opening a CMD terminal and typing in aws --version this should tell you the current installed version of AWS CLI.

You will need to conduct research into how to complete these tasks.

#### Configuring your CLI

Remembering the command from the slide decks, you are to configure your command line interface access.

Don’t forget you’ll need to obtain your Access Key ID and Secret Access Key to do this – your trainer can give these to you.

$ aws configure

#### Creating a security group, key pair and role

First, create a new security group and add a rule that allows incoming traffic over port 22 for SSH. Note the security group ID for later use.

aws ec2 authorize-security-group-ingress --group-id sg-c1274ba8 --protocol tcp --port 22 --cidr 164.39.75.108/24

Next, create a key pair, which allows you to connect to the instance.

$ aws ec2 create-key-pair --key-name AzimKeyPair

#### Launch your EC2 Instance

Run the AWS CLI command to launch your instance. Don’t forget you’ll need to include the AMI you want to use, the security group id and the instance type.

$ aws ec2 run-instances --image-id ami-f1d7c395 --count 1 --instance-type t2.micro --key-name AzimKeyPair --security-group-ids sg-c1274ba8 --subnet-id subnet-7d578c06

#### Obtain the IP Address & SSH

Use the correct AWS command to obtain the IP Address of the instance you just launched. Using this IP address, you can use putty to SSH into the Virtual machine and test the setup.

$ aws ec2 describe-instances --instance-ids i-0b32daa14b5fb848e --query 'Reservations[].Instances[].PublicDnsName'

### Task 3 – Using Amazon S3

Amazon Simple Storage Service is storage for the Internet. It is designed to make web-scale computing easier.

#### Creating a Bucket in Amazon S3

In the AWS Management Console click on **S3** and then click **Create Bucket**.

In the dialog box that opened, enter a name for the bucket.

Select a region to use and click next.

$ aws s3api create-bucket --bucket azim-bucket --region eu-west-2 --create-bucket-configuration LocationConstraint=eu-west-2

Set up your properties by enabling/disabling the versioning and logging option, and setting any tags you want to use.

Versioning

$ aws s3api get-bucket-versioning --bucket azim-bucket

{

"Status": "Suspended"

}

Logging

$ aws s3api get-bucket-logging --bucket azim-bucket

{

"LoggingEnabled": {

"TargetPrefix": "",

"TargetBucket": "azim-bucket"

}

}

The next option allows you to select user permissions for people on your bucket. Leave this as default, click next then create. You can now click on the name of the bucket to view it.

#### Adding objects to your bucket

Inside the bucket view, click on Get Started (or upload) and browse to select a file you want to upload. A simple image will suffice.

Click next for the remaining windows and the uploaded image will be visible inside your bucket. If you select this image, you will be presented with its properties, and a link you can click to view it online.

$ aws s3 cp SonChuckle.jpg s3://azim-bucket/

Explore the remaining options, try creating a folder, moving your objects and deleting items.

Be sure to clean up Amazon S3 when you are finished. Delete any objects/buckets you created.