Build Server configuration

@ Mary Walsh McGinty

# Op System: Ubuntu 16.04

# Build\_Jenkins\_server IP address : (Gold) 172.28.25.136 , (Sandbox) 172.28.25.123 Java

### Download Site:

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

Version to download: jdk-11.02\_linux-x64\_bin.tar.gz

x64 for 64 bit systems and tar.gz for Ubuntu Op Systems

### Steps:

Cd Downloads directory

/Downloads$ sudo tar -xzvf jdk-11.02\_linux-x64\_bin.tar.gz -C /usr/lib/jvm

X = extract, v= verbose, z= gunzip/unzipping f= extract in filesystem tar= to create and extract tape archives

-C specifies a different directory other than the current working directory

1. Add the file path for java to the /etc/environment file as JAVA\_HOME

cd /etc

sudo nano environment

#Edit the file with this line.

JAVA\_HOME=”usr/lib/jvm/java-8-oracle”

1. Next show Ubuntu a link to java

sudo update-alternatives --install "/usr/lib/jvm" "java" "/usr/lib/jvm/java-8-oracle/bin/java" 1

Set this version of java as the default

sudo update-alternatives --set java /usr/lib/jvm/java-8-oracle/bin/java

1. To check that java is install

java -version This should return:

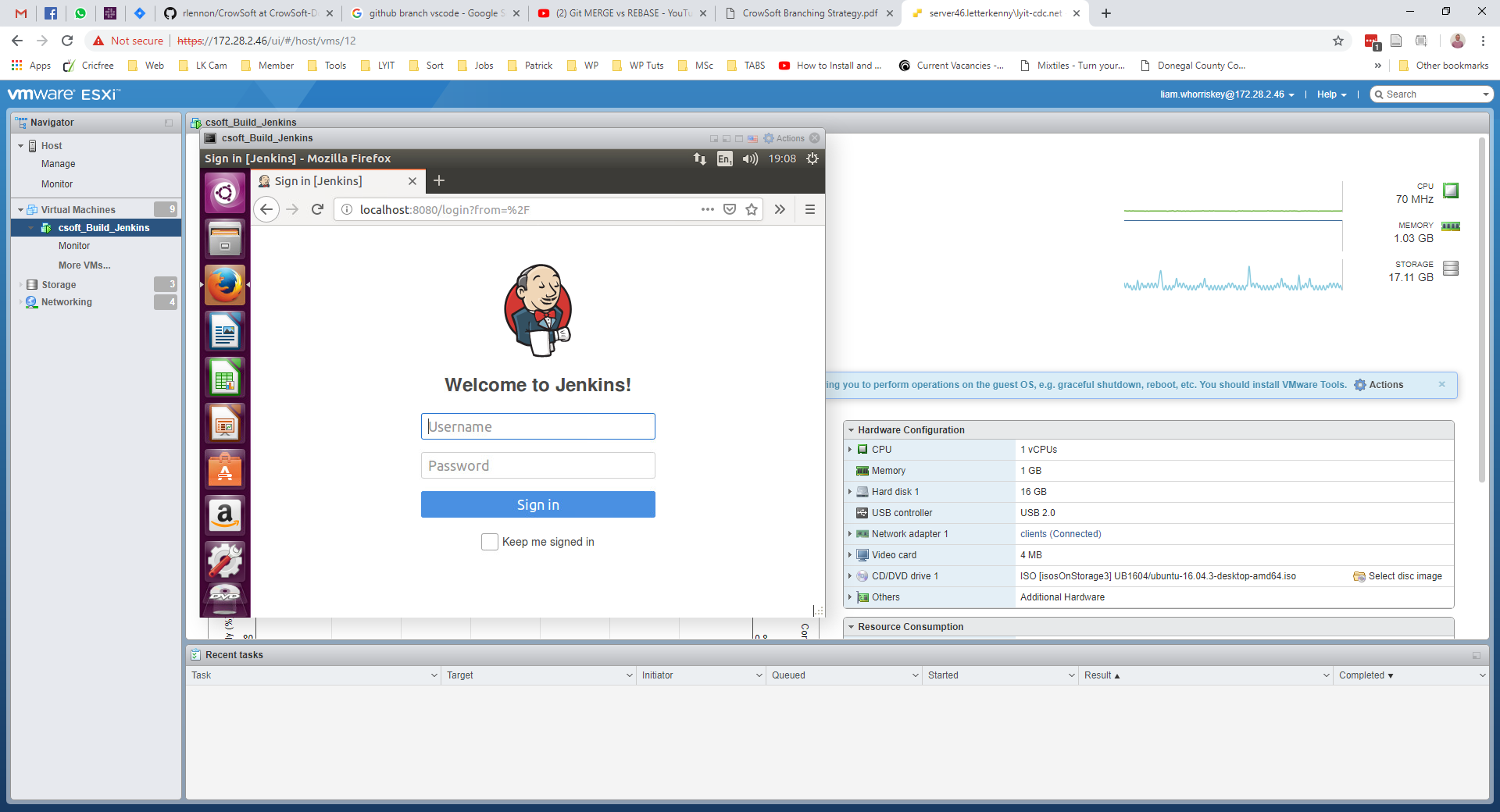
java version "1.8.0\_201"

Java(TM) SE Runtime Environment (build 1.8.0\_201-b09)

Java HotSpot(TM) 64-Bit Server VM (build 25.201-b09, mixed mode)

@Liam Whorriskey

# Jenkins



**Jenkins Setup Complete with suggested plugins.**

* Name: Dev Ops
* User: admin
* Pass: \*\*\*\*\*\*\*\*
* email: [l00113360@student.lyit.ie](mailto:l00113360@student.lyit.ie)
* Jenkins Url: <http://localhost:8080/>

**Plugins Installed**

* Ant Plugin
* Build Timeout
* Email Extension Plugin
* GitHub Branch Source Plugin
* Gradle Plugin
* LDAP Plugin
* Matrix Authorisation Strategy Plugin
* SWASP Markup Formatter Plugin
* PAM Authentication Plugin
* Pipeline
* Pipeline: GitHub Groovy Libraries
* SSH Slaves Plugin
* Subversion plugin
* Timestamper
* Workspace Cleanup Plugin

@Mary Walsh McGinty

* Artifactory plugin
* Ansible
* Audit Trail
* Backup plugin

A screenshot of a cell phone

Description automatically generated A screenshot of a cell phone

Description automatically generated

Figure 1 Thin Backup

* CodeSonar
* OWASP Dependency-Check
* Release
* Selenium Builder
* Selenium
* SonarQube Scanner for Jenkins
* Visual Studio Code Metrics

@Mary Walsh McGinty

Jenkins must be running before it can be accessed through LOCALHOST

To start Jenkins: sudo systemctl start jenkins

To check if Jenkins is running: sudo systemctl status jenkins

A picture containing outdoor, text, building

Description automatically generated

To stop Jenkins: sudo systemctl stop jenkins

### Jenkins Security

* Added Users with permissions
* Disabled “Remember Me” option
* Set up Matrix-based security

# Artifactory

### Prerequisites : Java JDK

Install: JFrog Artifactory-OSS Version 5.8.3. An open source version of JFrog Artifactory

Version to download : jfrog-artifactory-oss-5.8.3.deb

### Steps

1. You must be running as root to download

wget <https://bintray.com/artifact/download/jfrog/artifactory-debs/pool/main/j/jfrog-artifactory-oss-deb/jfrog-artifactory-oss-5.8.3.deb>

1. Install artifactory

$ gpg -keyserver pgpkeys.mit.edu -recv-key6B219DCCD7639232

$ gpg -a -export 6B219DCCD7639232 | sudo apt-key add –

$ apt-get update

$ dpkg -i jfrog-artifactory-oss-5.8.3.deb

1. Start the service

sudo systemctl start artifactory.service

Artifactory must be running before it can be accessed through LOCALHOST

To start Artifactory: sudo systemctl start artifactory.service

To check if Artifactory is running: sudo systemctl status artifactory.service A picture containing text, table, outdoor

Description automatically generated

To stop Artifactory: sudo systemctl stop artifactory.service

## Configure Artifactory

Artifactory can be accessed through the default Artifactory URL: localhost:8081/artifactory

### Steps

1. Initial setup activates a setup wizard where basic configuration is completed.
2. Setup administration username and password. Artifactory comes with a default set.
3. Create repositories. Created: Maven, Gradle and Generic Repositories.
4. Create a group; Admin section -> Security -> Groups

A screenshot of a cell phone

Description automatically generated

Click Save to save the group.

A screenshot of a cell phone

Description automatically generated

Double click on

1. Set up: Users with username, password.
2. Set permissions for each User.
3. Configured general security

### Configuring Artifactory on the Jenkins Server

#### Steps

1. Log into Jenkins and click on Manage Jenkins
2. Click on Configure System
3. Scroll down the page to the Artifactoy section
4. Add the server details, Url: localhost:8081/artifactory,

@ Matthew McColgan

1. Add the Artifactory log in details, username and password, then test the connection

### A screenshot of a cell phone Description automatically generated

@Mary Walsh McGinty

### Configuring Jenkins Build

1. Log into Artifactory

A screenshot of a cell phone

Description automatically generated

1. Click on repository in Set me up and add your password

A screenshot of a cell phone

Description automatically generated

1. A curl command will be returned with a path to the repository you choose

A screenshot of a cell phone

Description automatically generated

1. Log into Jenkins and in the configuration file for the build go to the build section

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Copy the curl command into the Build section as shown, inserting the details relevant to you.

1. Run the Build Now in Jenkins

# Gradle

### Prerequisites : Java JDK

After Java JDK has been installed install Gradle.

### Steps

1. Download the zip Gradle file

cd /tmp

wget <https://services.gradle.org/distributions/gradle-4.10.2-bin.zip>

This will download the file into the tmp directory

sudo unzip -d /opt/gradle /tmp/gradle-\*.zip

This extracts the file into the /opt/gradle/gradle-4.10.2 directory

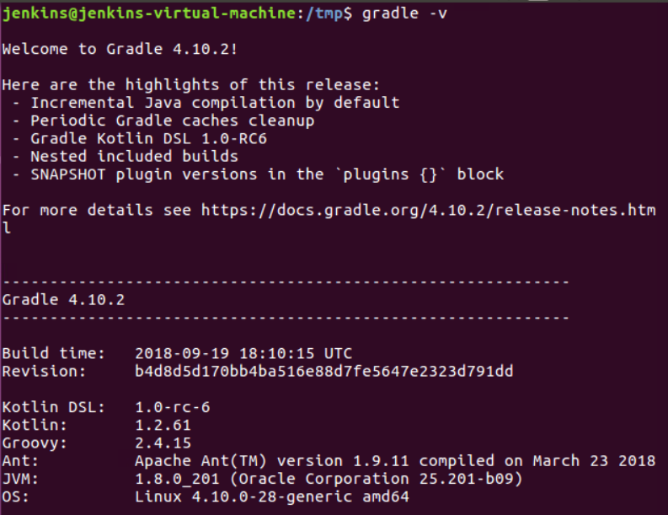
2. Configure Ubuntu Environment Variables

sudo nano /etc/profile.d/gradle.sh

This creates a new file called gradle.sh in the /etc/profile.d directory. Inside this file the following is added.

export GRADLE\_HOME=/opt/gradle/gradle-4.10.2

export PATH=${GRADLE\_HOME}/bin:${PATH}

 Exit and save the file ‘ctrl + x’ then y.

When done run the following commands to make the file executable.

sudo chmod +x /etc/profile.d/gradle.sh

source /etc/profile.d/gradle.sh

3. To check if Gradle installed correctly

gradle -v

It should return this →

## Gradle Freestyle Example

This pulls from Github and builds using Gradle then publishes to Artifactory

### Artifactory Configuration

In the artifactory configuration of the Freestyle project. These details were used.

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

In the More Details check the box to publish the artifacts.

A screenshot of a cell phone

Description automatically generated

### Build the project

In the Build section add the **Invoke Gradle Script**

A screenshot of a cell phone

Description automatically generated

Here the name of the gradle to use is added.

##### Example of Gradle script

A screenshot of a social media post

Description automatically generated

**NOTE:** The plugin to msBuild through Gradle

This allows for .NET projects to be built in Gradle

# Build Scan

This tool allows for a graph representation of the build output. It is a shareable and centralised record of a build. It provides a clear insight into the steps and outcome. This can be published to scans.gradle.com. This is created through a buildScan.gradle file which resides in the init.d directory of gradle.

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated A screenshot of a cell phone

Description automatically generated

Figure 2 Outcome of the BuildScan