DevOps Exercise Book

Contents

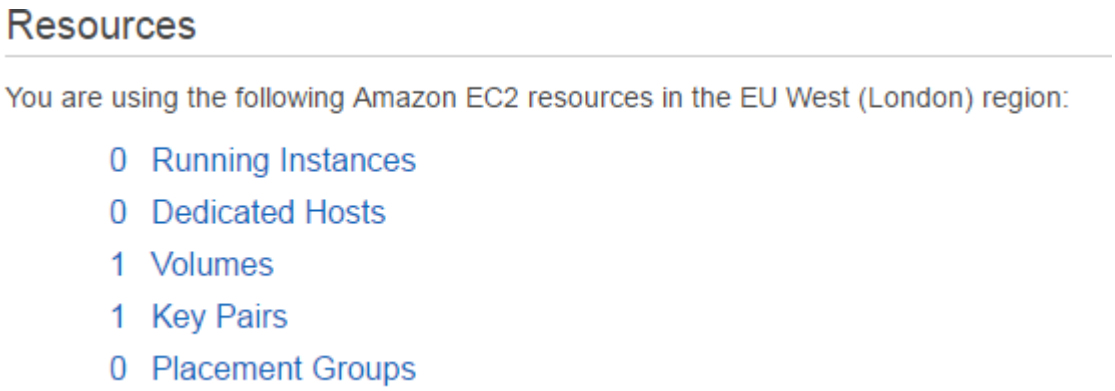
Linux

Task1: Your first Virtual Machine

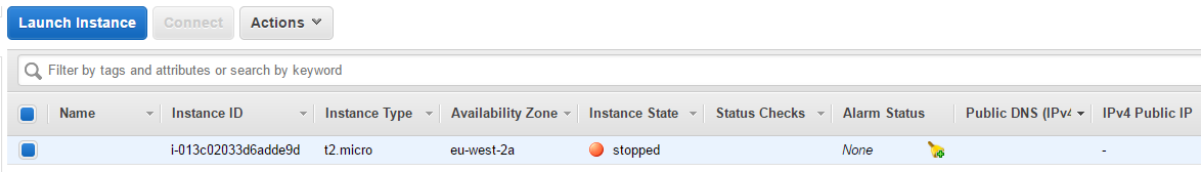
Using AWS EC2 Ubuntu box

Create instance of Ubuntu box according to needs.

Launch instance from Running Instances

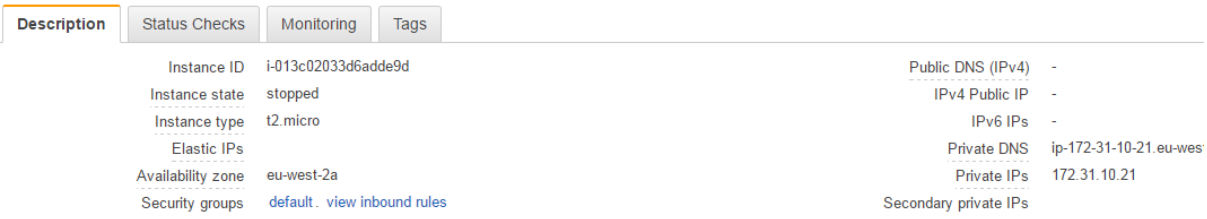


Following instance is present.

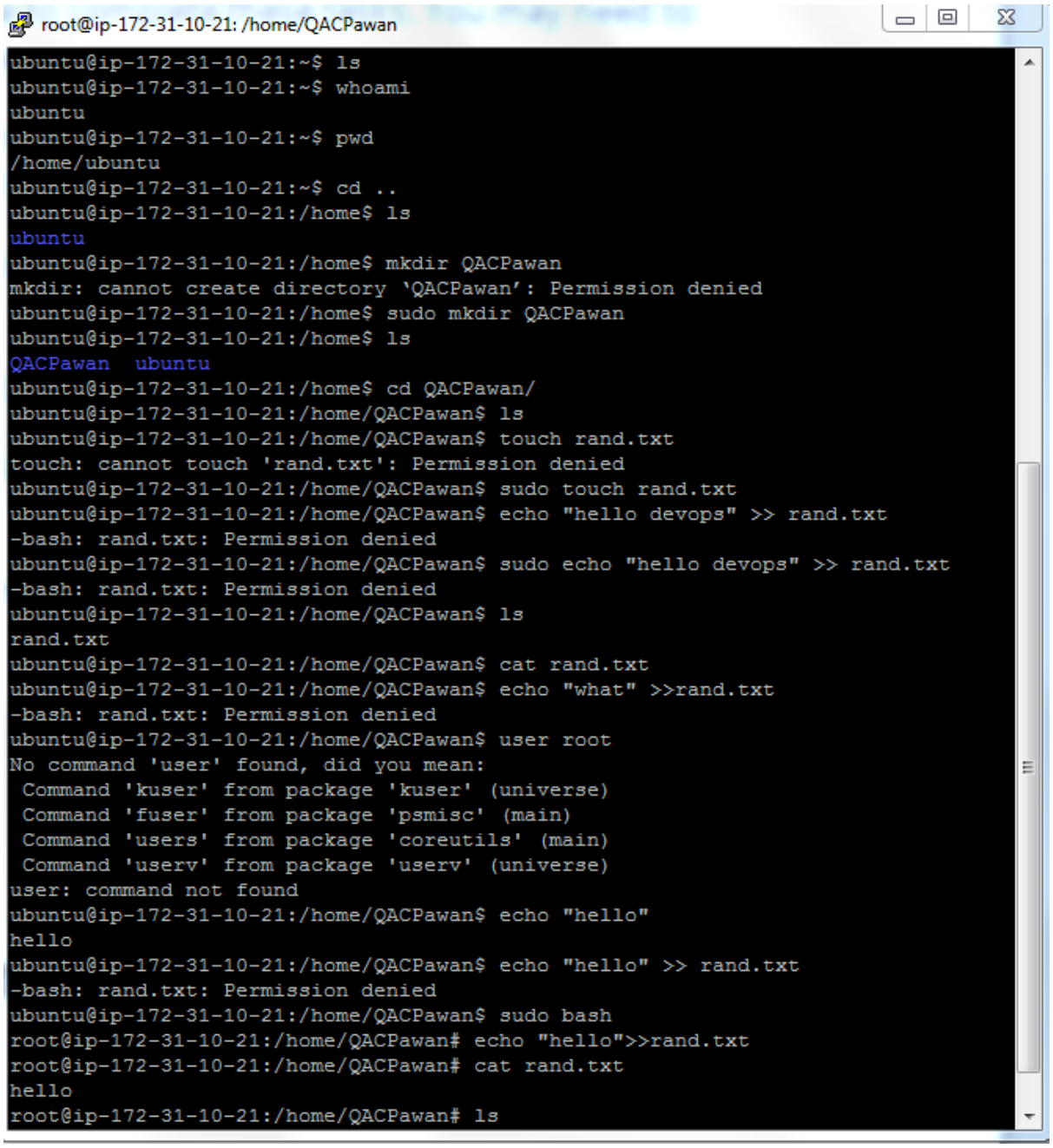


SSH To

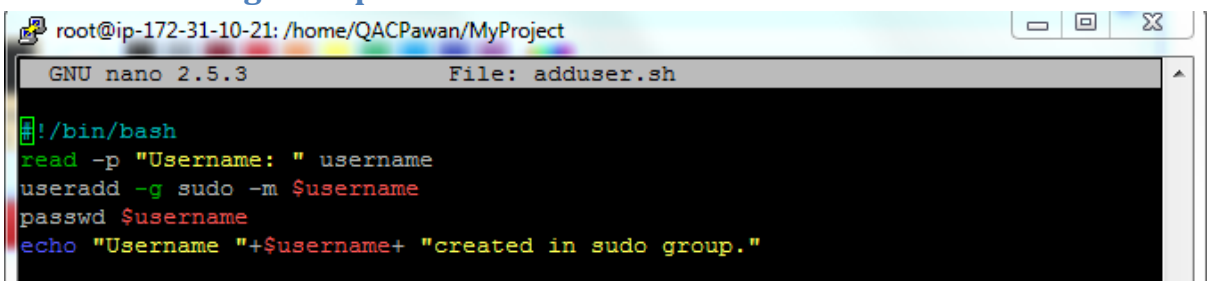
username@publicdns; ubuntu@...



Task 2: Terminal Exploration



Task 3: Creating a script file



```
 GNU nano 2.5.3           File: task1challenge.sh

#/bin/bash
date
w
uptime
```

Task 4: Configuring the Linux environment

apt-get update

apt-get install maven

(Maven installed itself and java as well)

```
Processing triggers for libc-bin (2.23-0ubuntu7) ...
Processing triggers for systemd (229-4ubuntu16) ...
Processing triggers for ureadahead (0.100.0-19) ...
Processing triggers for ca-certificates (20160104ubuntu1) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...

done.
done.
root@ip-172-31-10-21:/home/QACPawan/MyProject# java -version
openjdk version "1.8.0_131"
OpenJDK Runtime Environment (build 1.8.0_131-8u131-b11-0ubuntu1.16.04.2-b11)
OpenJDK 64-Bit Server VM (build 25.131-b11, mixed mode)
root@ip-172-31-10-21:/home/QACPawan/MyProject# mvn -v
Warning: JAVA_HOME environment variable is not set.
Apache Maven 3.3.9
Maven home: /usr/share/maven
Java version: 1.8.0_131, vendor: Oracle Corporation
Java home: /usr/lib/jvm/java-8-openjdk-amd64/jre
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "4.4.0-1013-aws", arch: "amd64", family: "unix"
root@ip-172-31-10-21:/home/QACPawan/MyProject#
```

Task 5: Using Iptables

Start iptables:

sudo ufw enable

Stop iptables:

sudo ufw disable

Show status:

sudo ufw status

```
root@ip-172-31-10-21:/home/QACPawan/MyProject# ufw status
Status: active
root@ip-172-31-10-21:/home/QACPawan/MyProject# ufw disable
Firewall stopped and disabled on system startup
root@ip-172-31-10-21:/home/QACPawan/MyProject# ufw status
Status: inactive
root@ip-172-31-10-21:/home/QACPawan/MyProject# ufw enable
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
root@ip-172-31-10-21:/home/QACPawan/MyProject#
```

Allowing ftp at port 21 rule:

```
root@ip-172-31-10-21:/home/QACPawan/MyProject# iptables -A INPUT -p tcp --dport ftp -j ACCEPT
root@ip-172-31-10-21:/home/QACPawan/MyProject# iptables -L -v -n
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
  421 34971 ufw-before-logging-input  all  --  *      *       0.0.0.0/0            0.0.0.0/0
  421 34971 ufw-before-input  all  --  *      *       0.0.0.0/0            0.0.0.0/0
   40  2944 ufw-after-input  all  --  *      *       0.0.0.0/0            0.0.0.0/0
   40  2944 ufw-after-logging-input  all  --  *      *       0.0.0.0/0            0.0.0.0/0
   40  2944 ufw-reject-input  all  --  *      *       0.0.0.0/0            0.0.0.0/0
   40  2944 ufw-track-input  all  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0           tcp dpt:21
```

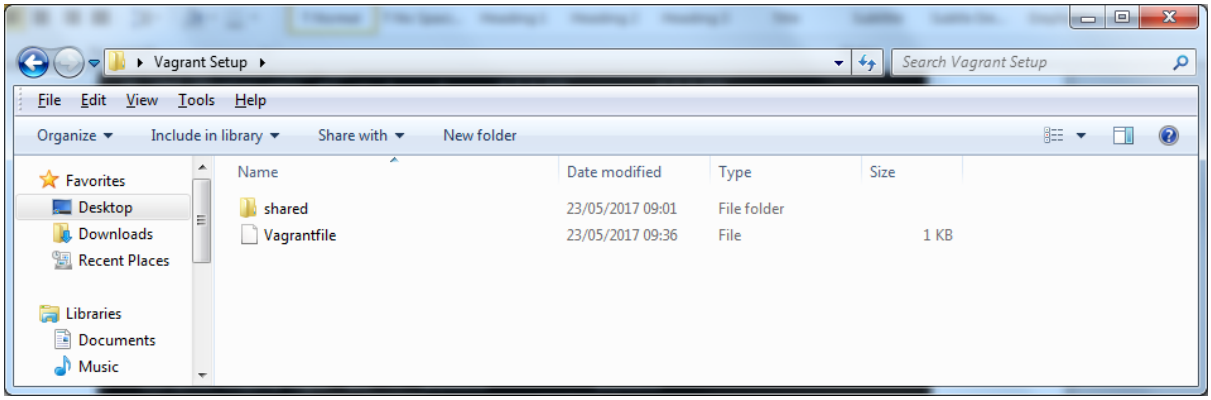
View rule by input chain and by line number, delete by input chain+linenumber

```
Chain ufw-user-output (1 references)
num  target     prot opt source               destination
root@ip-172-31-10-21:/home/QACPawan/MyProject# iptables -L INPUT --line-number
Chain INPUT (policy DROP)
num  target     prot opt source               destination
1    ufw-before-logging-input  all  --  anywhere             anywhere
2    ufw-before-input  all  --  anywhere             anywhere
3    ufw-after-input  all  --  anywhere             anywhere
4    ufw-after-logging-input  all  --  anywhere             anywhere
5    ufw-reject-input  all  --  anywhere             anywhere
6    ufw-track-input  all  --  anywhere             anywhere
7    ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ftp
root@ip-172-31-10-21:/home/QACPawan/MyProject# iptables -D INPUT 7
root@ip-172-31-10-21:/home/QACPawan/MyProject# iptables -L INPUT --line-number
Chain INPUT (policy DROP)
num  target     prot opt source               destination
1    ufw-before-logging-input  all  --  anywhere             anywhere
2    ufw-before-input  all  --  anywhere             anywhere
3    ufw-after-input  all  --  anywhere             anywhere
4    ufw-after-logging-input  all  --  anywhere             anywhere
5    ufw-reject-input  all  --  anywhere             anywhere
6    ufw-track-input  all  --  anywhere             anywhere
root@ip-172-31-10-21:/home/QACPawan/MyProject#
```

Task 6: Vagrant scripting

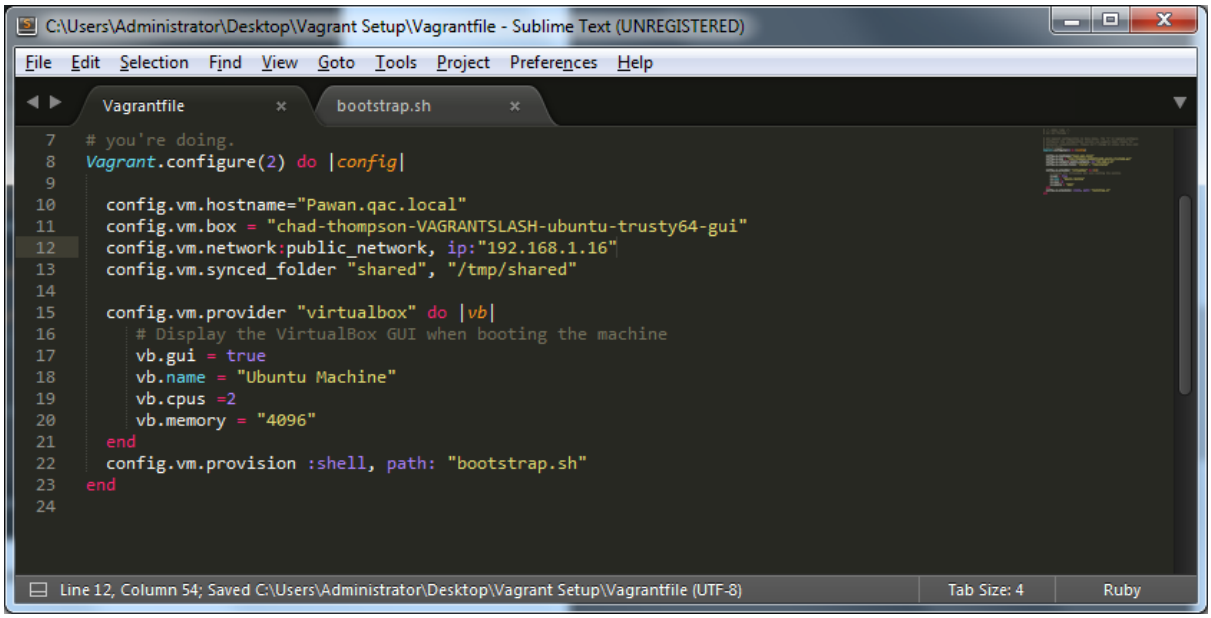
Create directory “Vagrant Setup” - in here open Git bash and use the command, *vagrant init*, to initialise a new vagrant repository. This should add a Vagrantfile to the directory

Create a shared folder to use with VM.

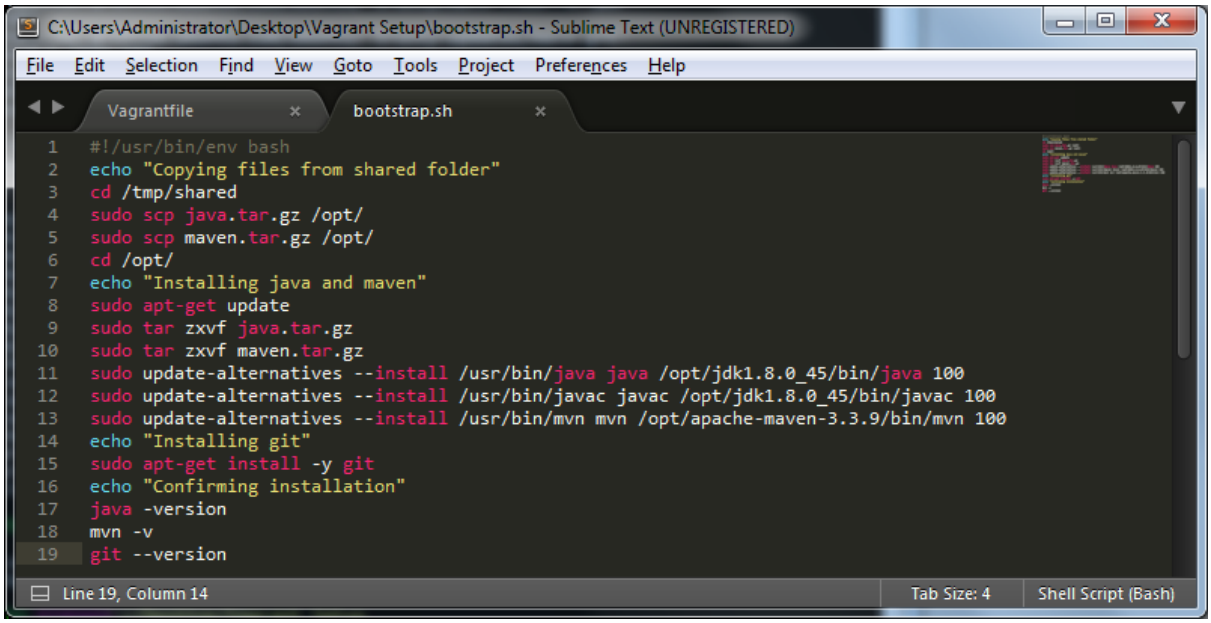


For single VM installation

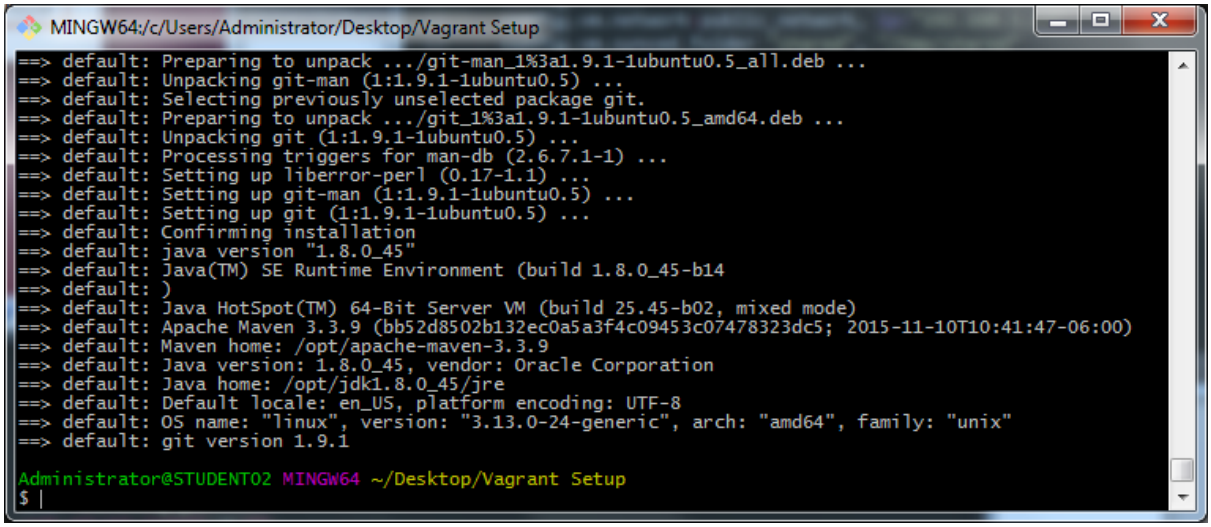
Edited vagrantfile to setup the VM with required specification.



Bash script used to install JAVA, Maven and Git on VM.



Use “vagrant up” on the folder with vagrantfile to create VM and run bash script.



For multiple VM installation

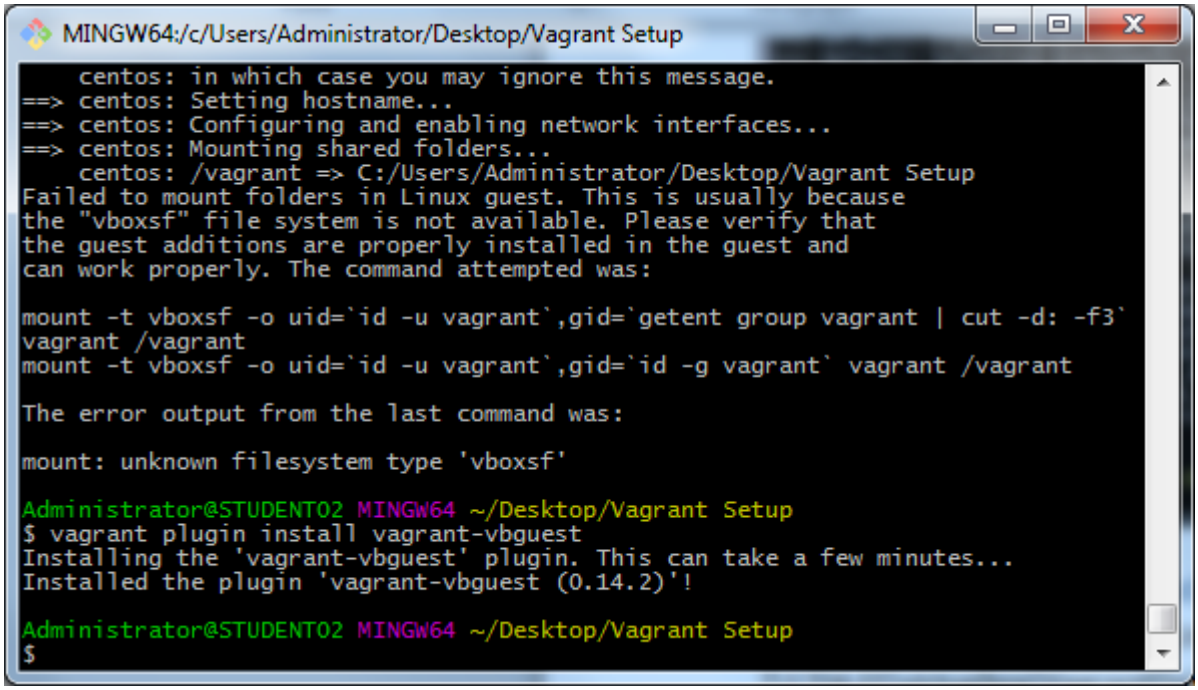
For CentOS installation, edit C:\Users\Administrator\.vagrant.d\boxes\centos-VAGRANTSLASH-7\1704.01\virtualbox\Vagrantfile

config.vm.synced_folder ".", "/vagrant", type: "rsync"

config.vm.synced_folder ".", "/vagrant", type: "virtualbox"

Rsync isn’t installed.

Run **vagrant plugin install vagrant-vbguest** on git bash as the shared folder encounters problem with centOS due to mismatch of GuestAdditions between centOS and virtualbox.



Vagrantfile for multiple VM

```
Vagrant.configure(2) do |config|
  config.vm.define "ubuntu" do |ubuntu|
      ubuntu.vm.hostname="Pawan.qac.local"
      ubuntu.vm.box = "chad-thompson-VAGRANTSLASH-ubuntu-trusty64-gui"
      ubuntu.vm.network:public_network, ip:"192.168.1.17"
      ubuntu.vm.synced_folder "shared", "/tmp/shared"

      ubuntu.vm.provider "virtualbox" do |vb|
         # Display the VirtualBox GUI when booting the machine
         vb.gui = true
         vb.name = "Ubuntu Machine"
         vb.cpus =2
         vb.memory = "4096"
      end
  end
  config.vm.define "centos" do |centos|
      centos.vm.hostname="Pawan1.qac.local"
      centos.vm.box = "centos-VAGRANTSLASH-7"
      centos.vm.network:public_network
      centos.vm.synced_folder "shared", "/tmp/shared"

      centos.vm.provider "virtualbox" do |vb|
         # Display the VirtualBox GUI when booting the machine
         vb.gui = true
         vb.name = "CentOS"
         vb.cpus =2
         vb.memory = "4096"
      end
  end
  config.vm.provision :shell, path: "bootstrap.sh"
end
```

Line 38, Column 4; Saved C:\Users\Administrator\Desktop\Vagrant Setup\Vagrantfile (UTF-8) | Tab Size: 4 | Ruby

Script file for multiple VM

C:\Users\Administrator\Desktop\Vagrant Setup\bootstrap.sh - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

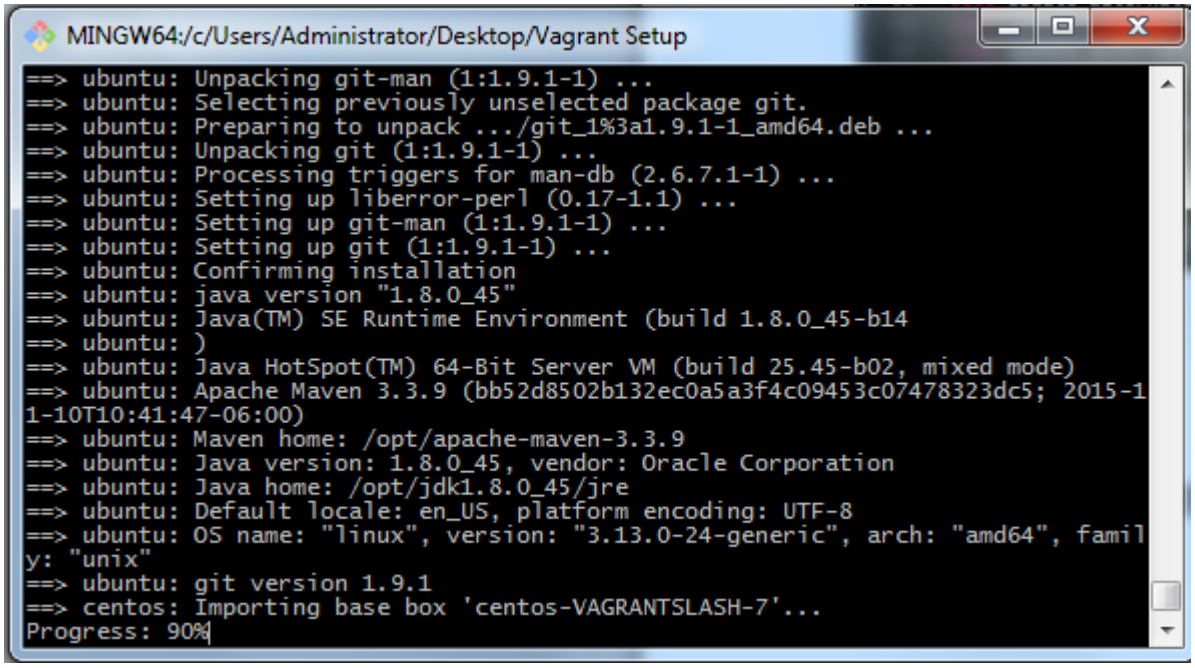
Vagrantfile — Desktop\Vagrant Setup | Vagrantfile — .vagrant.d\...\virtualbox | bootstrap.sh

```
#!/usr/bin/env bash
echo "Copying files from shared folder"
cd /tmp/shared
sudo scp java.tar.gz /opt/
sudo scp maven.tar.gz /opt/
cd /opt/
echo "Installing java and maven"
if [ "$(. /etc/os-release; echo $NAME)" = "Ubuntu" ]; then
  sudo apt-get install updates
else
  sudo yum update
fi
sudo tar zxvf java.tar.gz
sudo tar zxvf maven.tar.gz
sudo update-alternatives --install /usr/bin/java java /opt/jdk1.8.0_45/bin/java 100
sudo update-alternatives --install /usr/bin/javac javac /opt/jdk1.8.0_45/bin/javac 100
sudo update-alternatives --install /usr/bin/mvn mvn /opt/apache-maven-3.3.9/bin/mvn 100
echo "Installing git"
if [ "$(. /etc/os-release; echo $NAME)" = "Ubuntu" ]; then
    sudo apt-get install -y git
else
  sudo yum install -y git
fi
echo "Confirming installation"
java -version
mvn -v
git --version
```

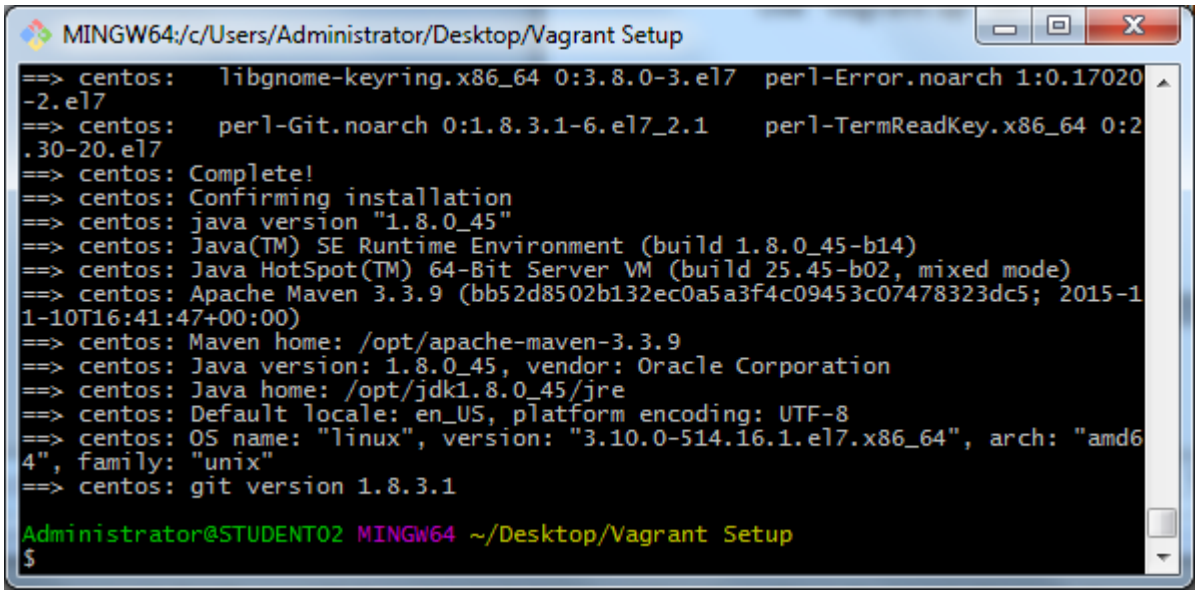
Line 22, Column 26 | Tab Size: 4 | Shell Script (Bash)

Use “vagrant up” on the folder with vagrantfile to create VM and run bash script.

Ubuntu installed with JAVA, MAVEN, and GIT.

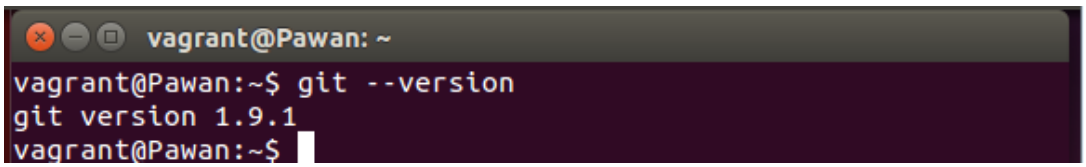


CentOS installed with JAVA, MAVEN, and GIT.

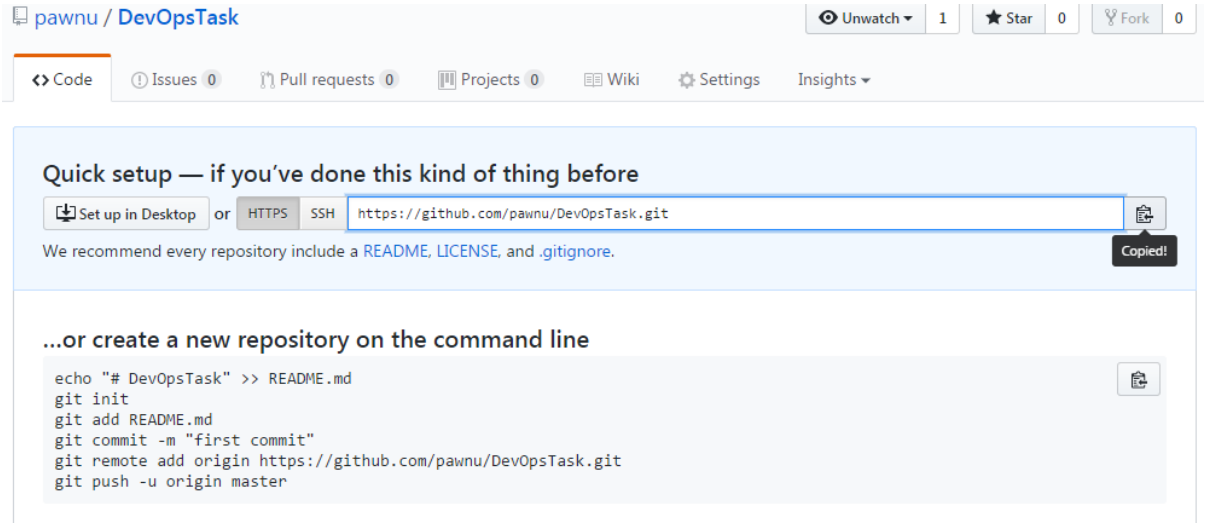


Task 7: Repository Management

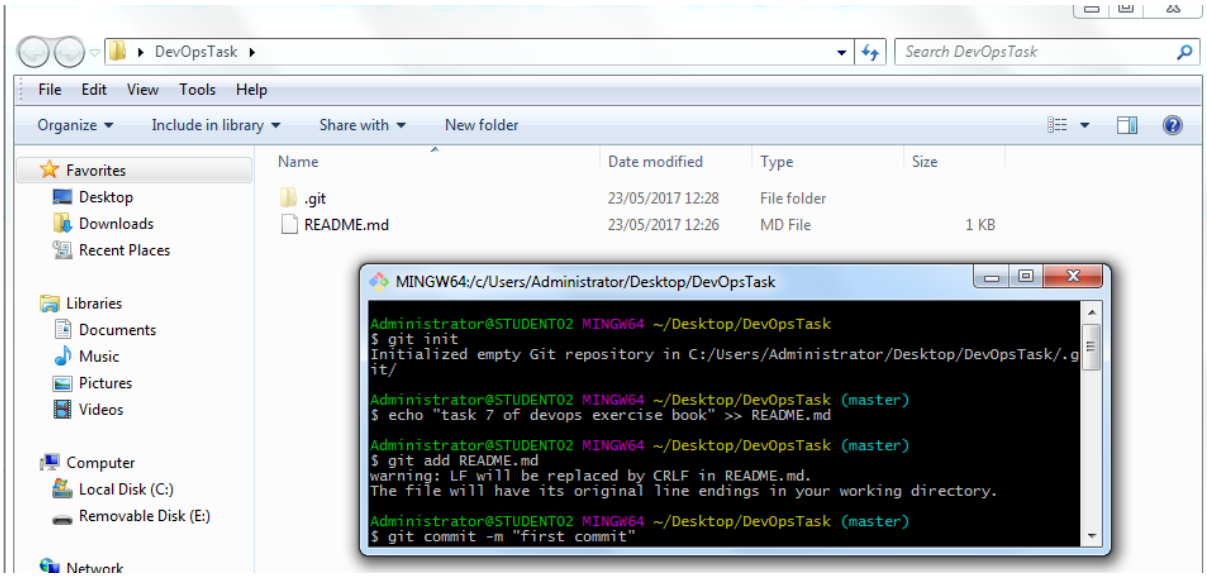
Confirmed Git is installed in VM



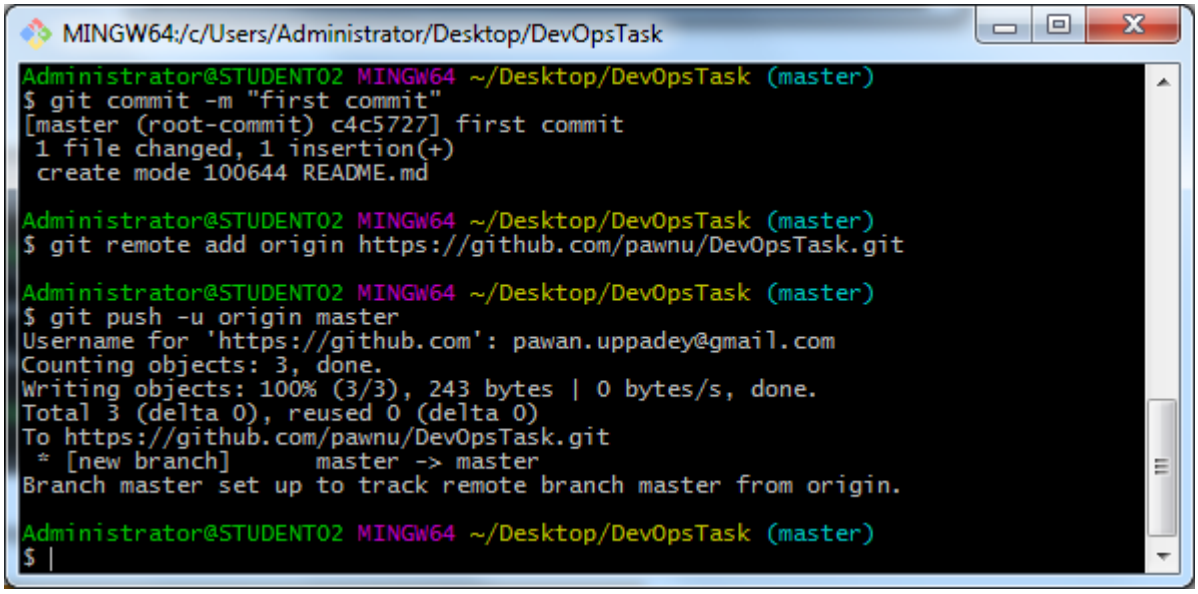
On Windows host machine, create a repository on github.



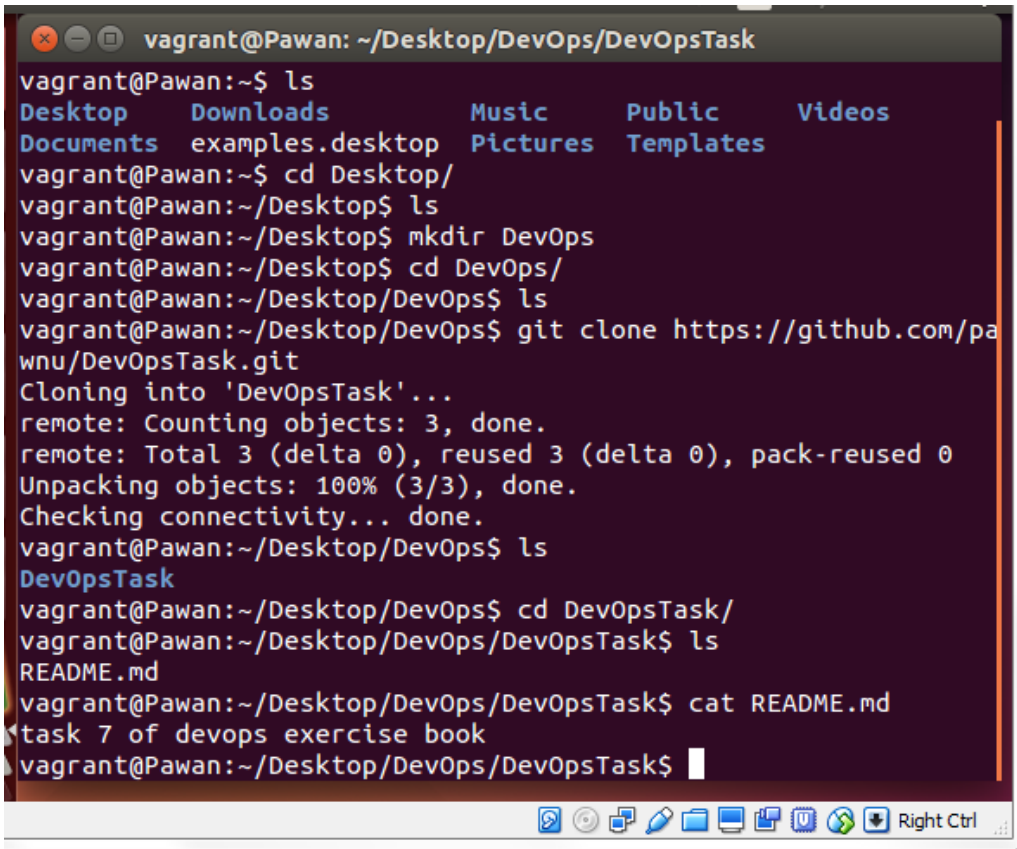
On host windows machine, create a directory for git and use “git init” command on that folder. Create a file, add and commit it.



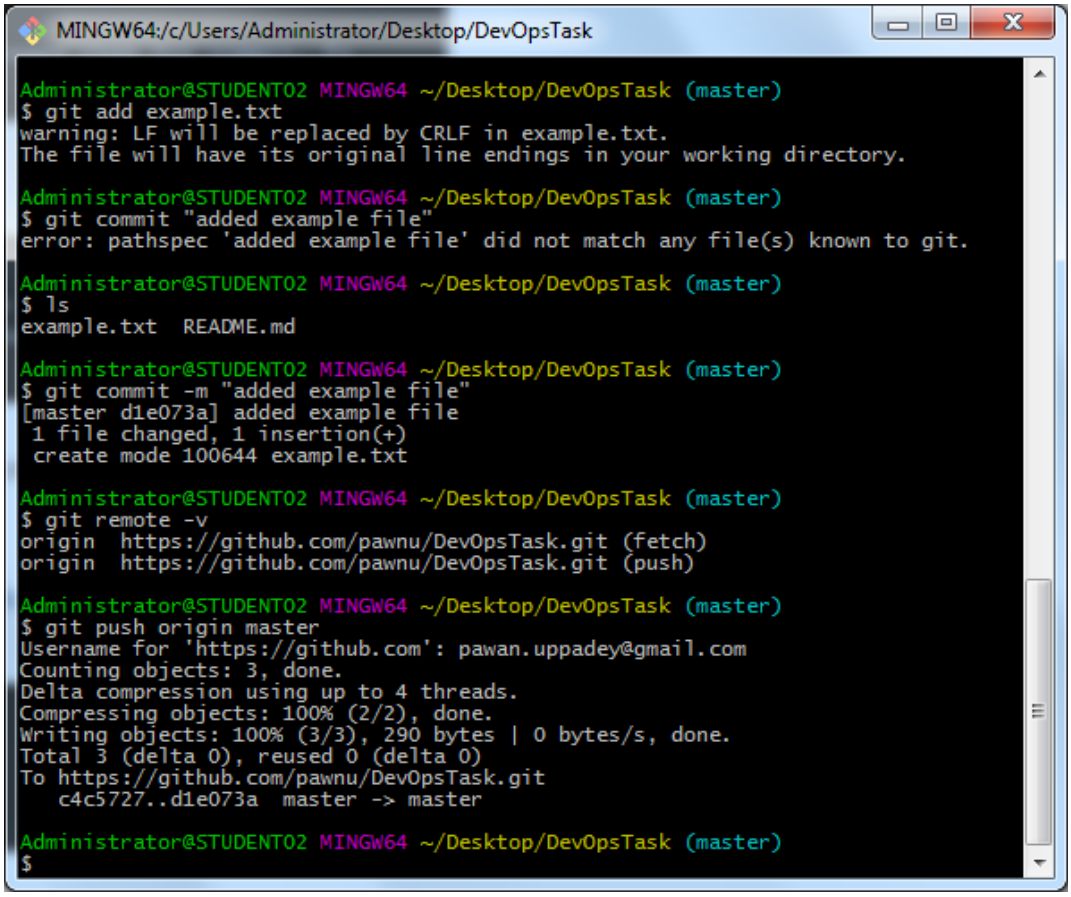
Add the commit to your repository on github with HTTPS link and push the changes.



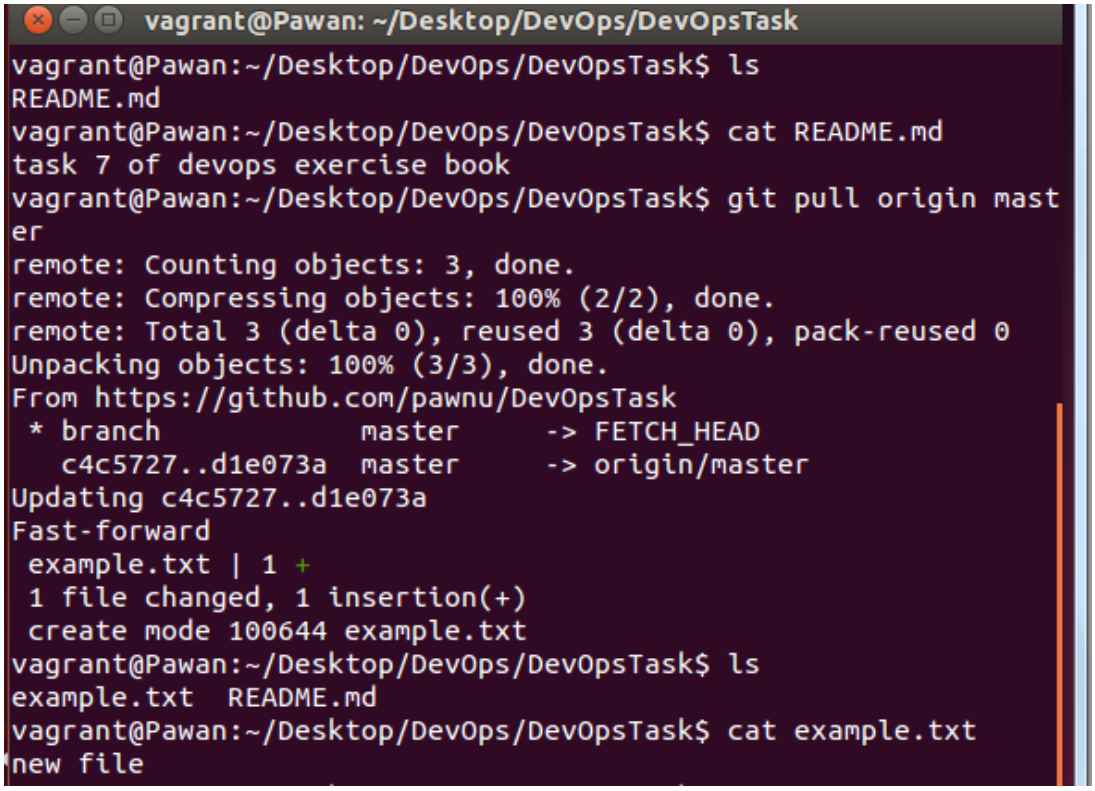
On Ubuntu guest machine, clone and confirm the file created earlier is present.



Create a new file on Windows host machine and push change to repository.



Confirm the file present on Ubuntu guest machine.

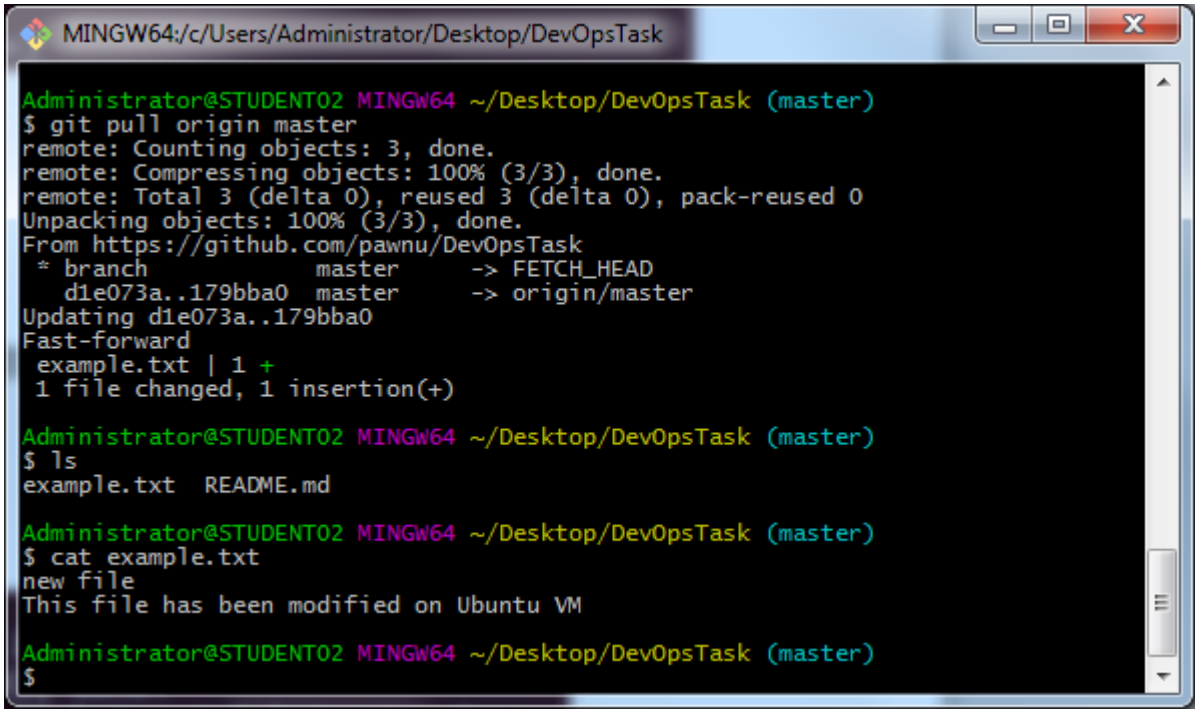


Make change to a file on Ubuntu and push to repository

```
vagrant@Pawan:~/Desktop/DevOps/DevOpsTask$ git add example.txt
vagrant@Pawan:~/Desktop/DevOps/DevOpsTask$ git commit -m "made
change to example text file"
[master 179bba0] made change to example text file
 Committer: Vagrant <vagrant@Pawan.qac.local>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accur
ate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

After doing this, you may fix the identity used for this commit
 with:

    git commit --amend --reset-author

 1 file changed, 1 insertion(+)
```

```
Counting objects: 5, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 339 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/pawnu/DevOpsTask.git
   d1e073a..179bba0  master -> master
vagrant@Pawan:~/Desktop/DevOps/DevOpsTask$
```

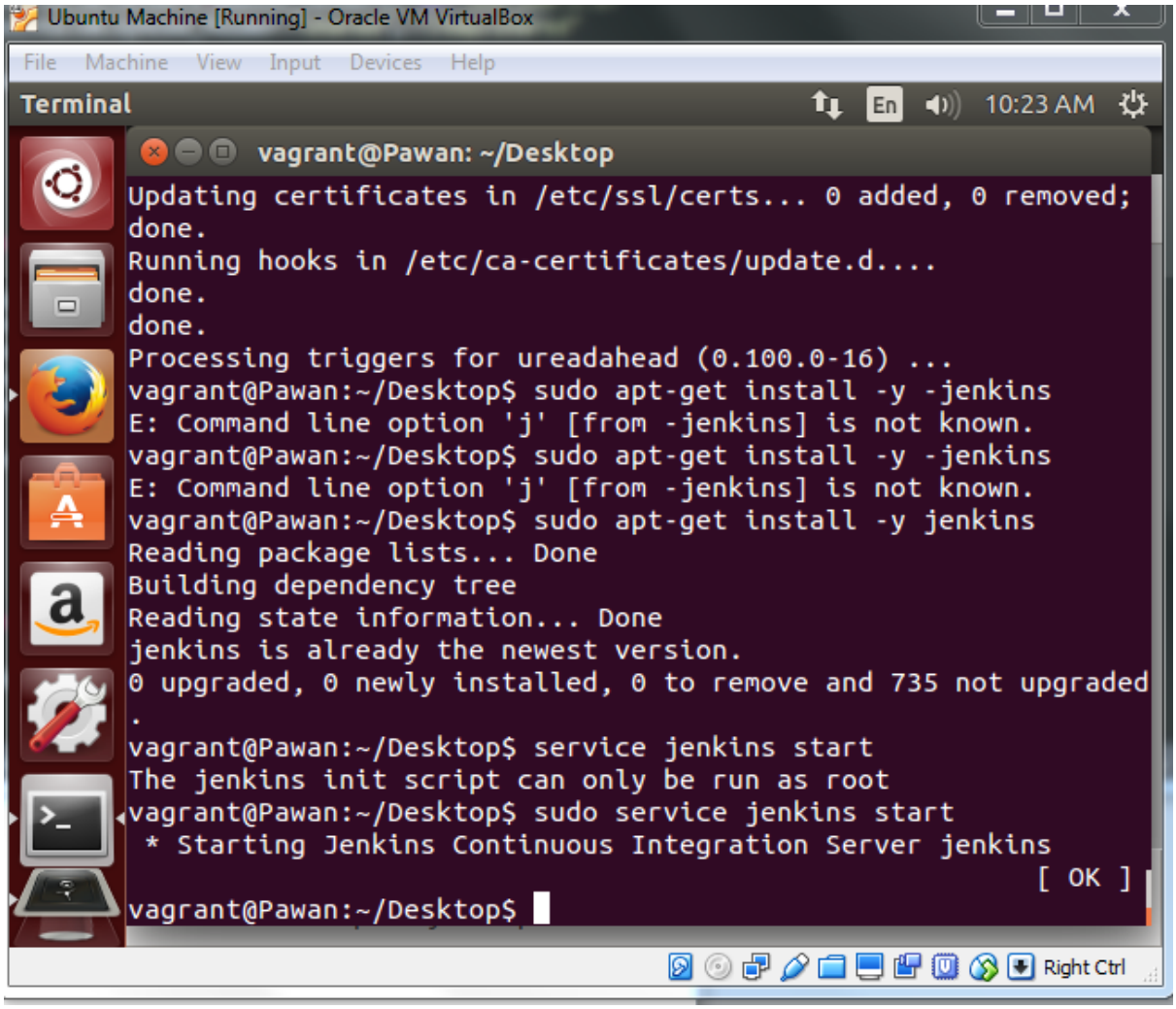
Confirm changes on Windows host machine side



Tooling

Task 1a – Setting up Jenkins

sudo scp jenkins_2.1_all.deb /home/vagrant/Desktop/
cd /home/vagrant/Desktop
sudo dpkg –i jenkins_2.1_all.deb
sudo apt-get install –y –f
sudo apt-get install –y jenkins
sudo service jenkins start



Task 2 – Setting up Jira

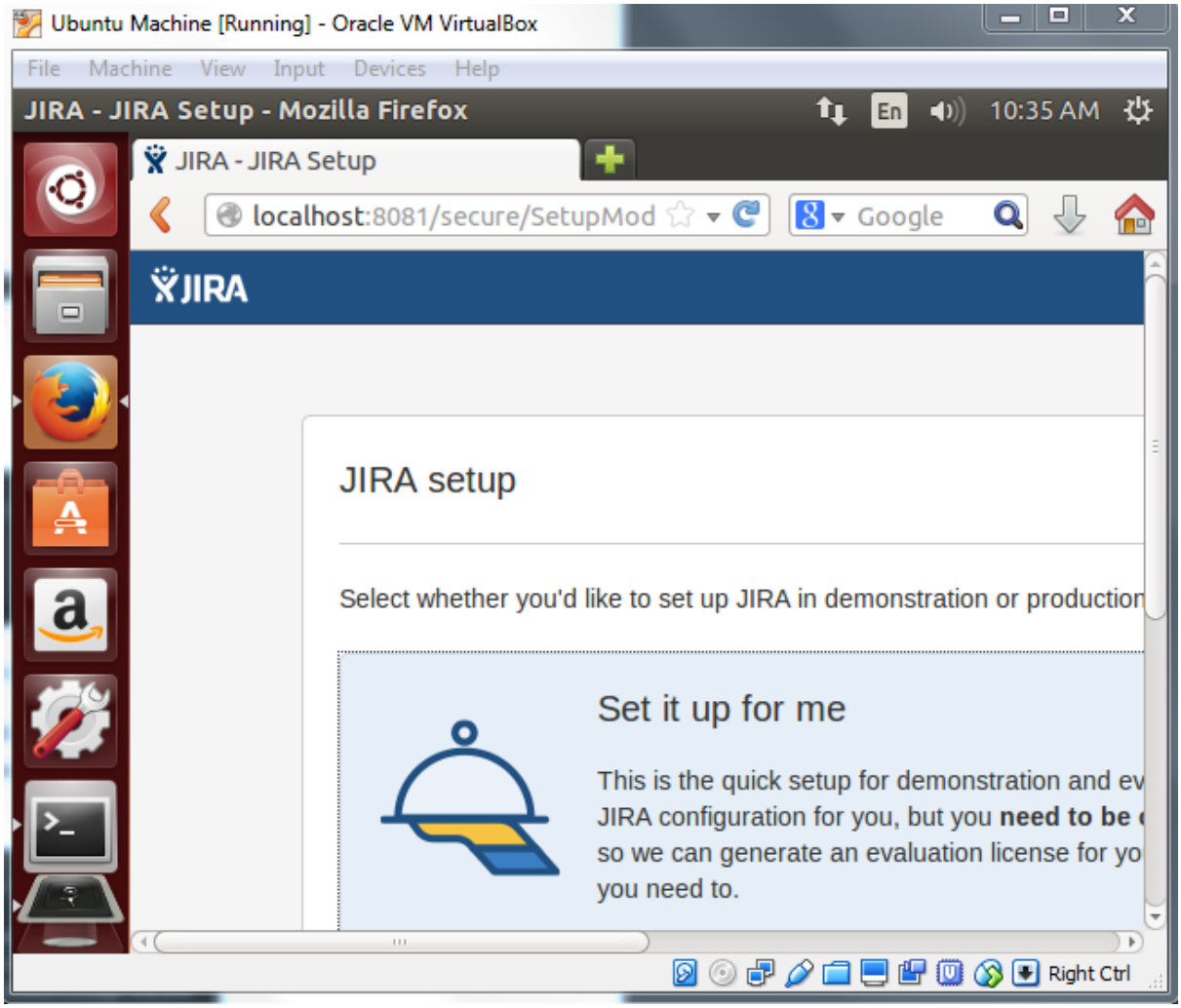
cd /opt/

sudo chmod a+x jira.bin

sudo ./jira.bin

Choose custom install, install on port 8081, custom port 80

```
Please wait a few moments while JIRA starts up.
Launching JIRA ...
Installation of JIRA 6.4.9 is complete
Your installation of JIRA 6.4.9 is now ready and can be accesse
d via your
browser.
JIRA 6.4.9 can be accessed at http://localhost:8081
Finishing installation ...
vagrant@Pawan:/opt$
```



Task 4 – Installing Nexus

cd /opt/
sudo scp nexus-2.14.4-03-bundle.tar.gz /usr/local
cd /usr/local
sudo tar xvzf nexus-2.14.4-03-bundle.tar.gz
sudo ln –s nexus-2.14.4-03 nexus
echo "1" | sudo update-alternatives --config java
sudo chowm -R vagrant nexus* sonatype-work
cd /usr/local/nexus
./bin/nexus console
./bin/nexus start

Notes:
nexus-3.* didn't work with this setup
nexus won't run with sudo, change a user to be owner of nexus and sonawork directory to run
nexus requires JVM 1.8. JVM might show as 1.7. Make sure java 1.8 is installed.

Use **sudo update-alternatives - -config java** and select the 1.8 version of java to run nexus

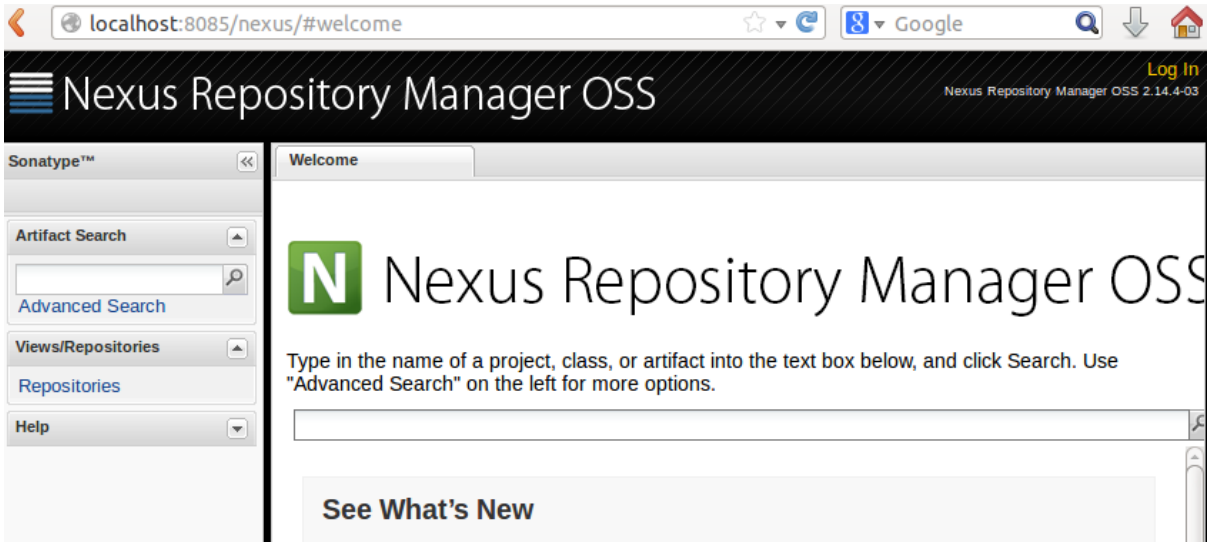
```
vagrant@Pawan: /opt
vagrant@Pawan:/usr/local/nexus$ sudo update-alternatives --conf
ig java
There are 2 choices for the alternative java (providing /usr/bi
n/java).

  Selection    Path
Priority   Status
------------------------------------------------------------
* 0            /usr/lib/jvm/java-7-openjdk-amd64/jre/bin/java
 1071      auto mode
  1            /opt/jdk1.8.0_45/bin/java
 1         manual mode
  2            /usr/lib/jvm/java-7-openjdk-amd64/jre/bin/java
 1071      manual mode

Press enter to keep the current choice[*], or type selection nu
mber: 1
update-alternatives: using /opt/jdk1.8.0_45/bin/java to provide
 /usr/bin/java (java) in manual mode
vagrant@Pawan:/usr/local/nexus$ java -version
java version "1.8.0_45"
Java(TM) SE Runtime Environment (build 1.8.0_45-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.45-b02, mixed mode)
```

Nexus by default runs on port 8081, to change it edit the **conf/properties** file

```
# Sonatype Nexus
# ==============
# This is the most basic configuration of Nexus.

# Jetty section
application-port=8085
application-host=0.0.0.0
nexus-webapp=${bundleBasedir}/nexus
nexus-webapp-context-path=/nexus
```

Run nexus as follows:

```
vagrant@PUPPADEY:/usr/local/nexus$ ./bin/nexus stop
Stopping Nexus OSS...
Nexus OSS was not running.
vagrant@PUPPADEY:/usr/local/nexus$ ./bin/nexus start
Starting Nexus OSS...
Started Nexus OSS.
vagrant@PUPPADEY:/usr/local/nexus$ ./bin/nexus status
Nexus OSS is running (18921).
```



Task 5 – Install Zabbix

cd /opt/

wget http://repo.zabbix.com/zabbix/2.4/ubuntu/pool/main/z/zabbix-release/zabbix-release_2.4-1+trusty_all.deb

sudo dpkg -i zabbix-release_2.4-1+trusty_all.deb

sudo apt-get install -y zabbix-server-mysql zabbix-frontend-php php5-mysql

Edit **/etc/php5/apache2/php.ini**

post_max_size = 16M

max_execution_time = 300

max_input_time = 300

date.timezone = Europe/London

sudo service apache2 restart

Create **/etc/zabbix/apache.conf**

 # Define /zabbix alias, this is the default

<IfModule mod_alias.c>

Alias /zabbix /usr/share/zabbix

</IfModule>

sudo cp /etc/zabbix/apache.conf /etc/apache2/conf-available/zabbix.conf

sudo a2enconf zabbix.conf

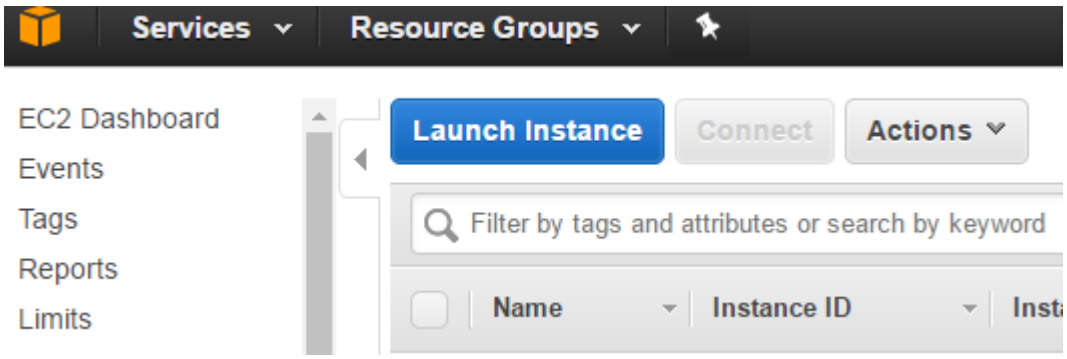
sudo service apache2 restart

sudo service zabbix-server start

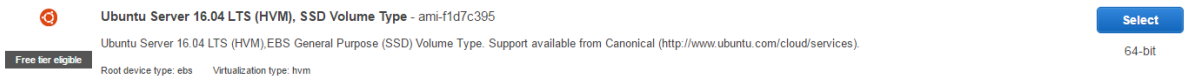
Docker and Kubernetes

Creating AWS instance

EC2 – Launch Instance

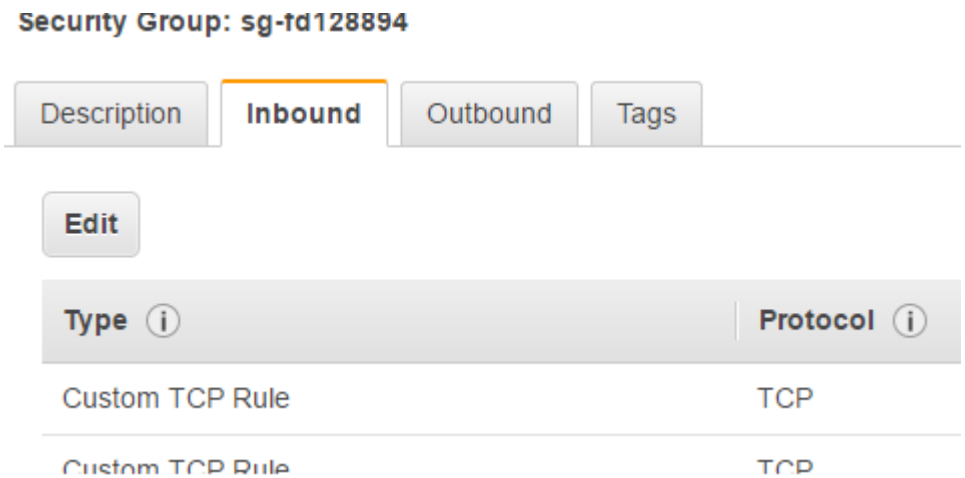


Install Ubuntu

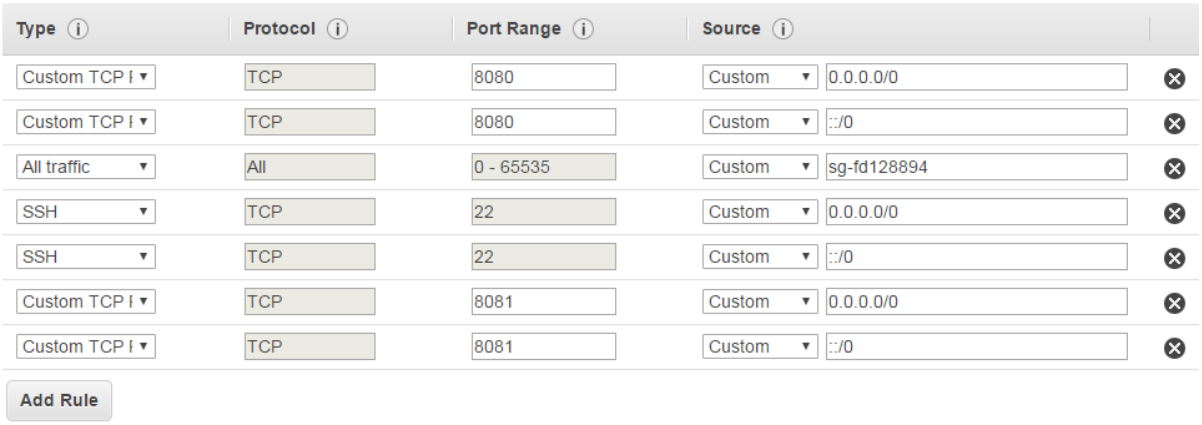


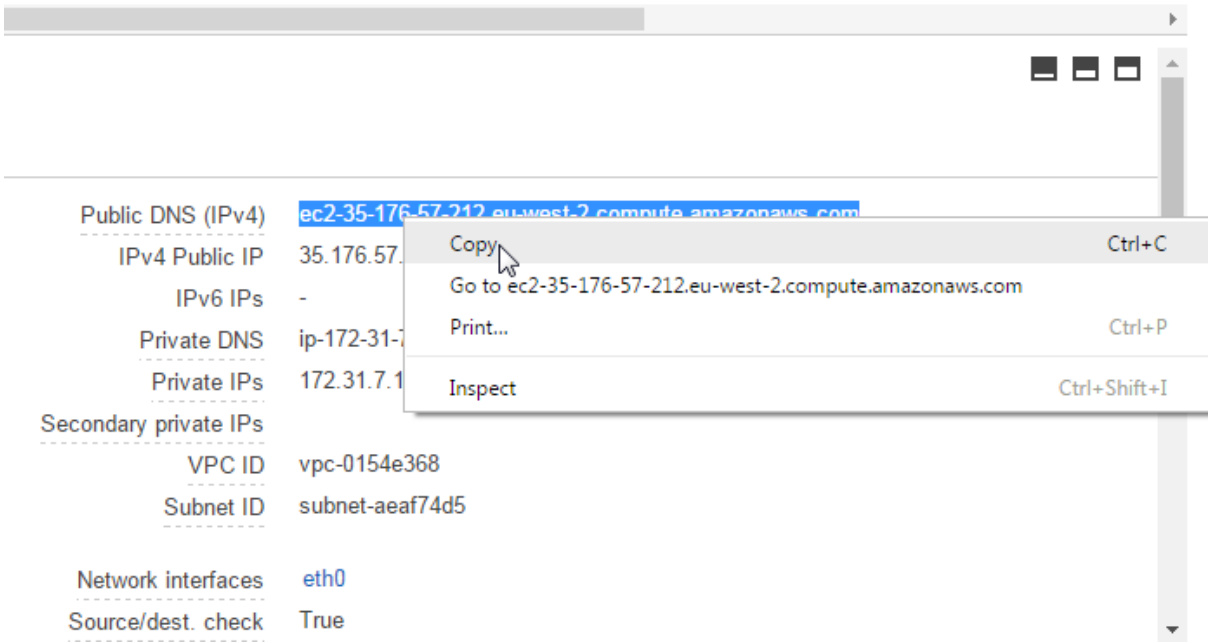
Go to security groups – default to configure firewall

Public DNS (IPv4	IPv4 Public IP	IPv6	Key Name	Monitoring	Launch Time	Security Groups
	-	-	Keypair2	disabled	May 24, 2017 at 3:44:30 PM...	default
	-	-	dockerpair	disabled	May 25, 2017 at 10:34:30 AM..	default
ec2-35-176-57-21...	35.176.57.212	-	dockerpair	disabled	May 25, 2017 at 10:52:25 AM..	default



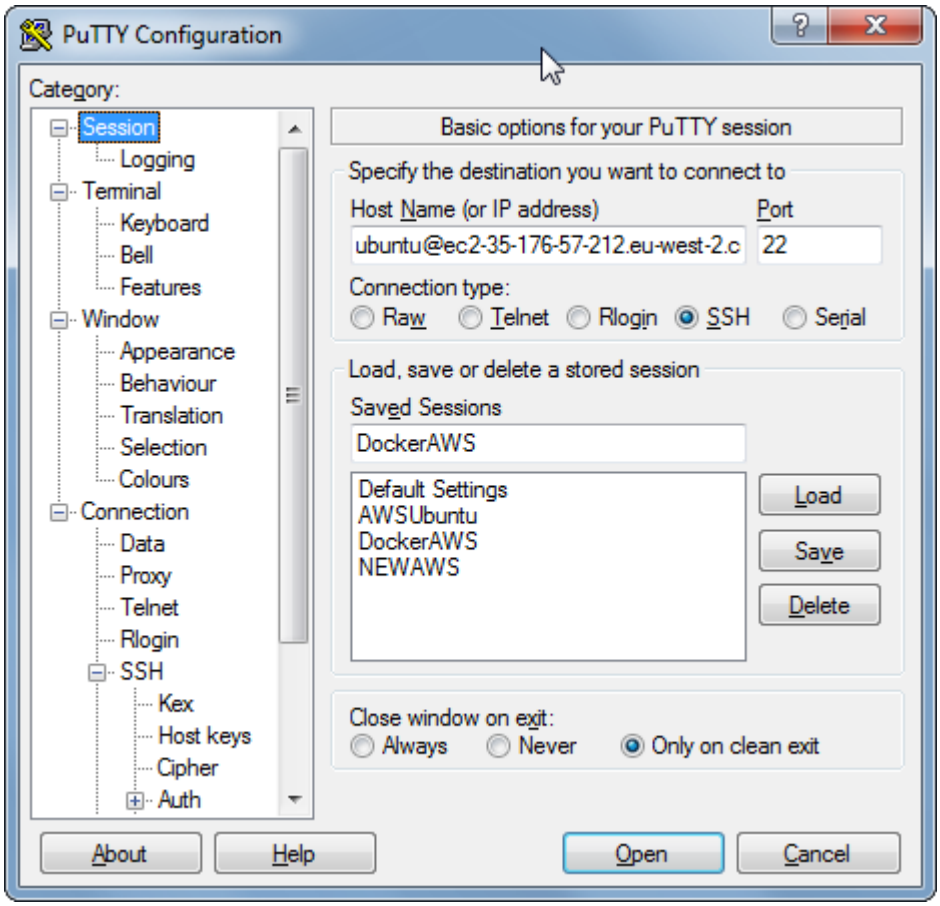
Add port 22 for SSH, and others as required e.g. 8080





The .pem file provided by AWS to access SSH can be broken down into a private/public key pair. The private key can be created by using the .pem file on PuTTYgen – click generate private key.

Put your privatekey to SSH-Auth on PuTTY Config. Insert the username (Ubuntu) public IP address and port 22 to access SSH.



Task 1 – Install docker

Get the script file to install docker and its dependencies from docker website and pipe it to the shell.

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

Task 2 – Deploy a Jenkins Container

Pull a Jenkins image
docker pull Jenkins
Run Jenkins in a container
docker run -p 8080:8080 -p 50000:50000 jenkins

A container of Jenkins has been created, use ctrl+z to get out of the terminal.

sudo docker ps –a

Lists all processes, find the container id of Jenkins

sudo docker start containerID

```
ubuntu@ip-172-31-7-183:~$ sudo docker start 3ee2780a2a10
3ee2780a2a10
ubuntu@ip-172-31-7-183:~$
```

```
ubuntu@ip-172-31-7-183:~$ sudo docker ps -a -f status=running
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
35f39df042c9        mysql:8.0           "docker-entrypoint..."   17 hours ago        Up 11 minutes
3ee2780a2a10        jenkins             "/bin/tini -- /usr..."   22 hours ago        Up 10 minutes
ubuntu@ip-172-31-7-183:~$
```

Task 3 – Create a dockerfile

```
FROM ubuntu:16.04

#Always update your running system
RUN sudo apt-get update -y

#You may or may not need to run these commands
RUN sudo apt-get install -y wget
RUN sudo apt-get install -y tar

#installs the libraries needed to run the GUI
RUN sudo apt-get install -y libgtk2.0
RUN sudo apt-get install -y mesa-utils
RUN sudo apt-get install -y libXtst6

#RUN sudo apt-get install -y openjdk-7-jre
```

#RUN java -version

#Now install the Java Compiler
#RUN sudo apt-get install -y openjdk-7-jdk
#RUN javac -version

#Add java from file and install
WORKDIR /opt

ADD files /opt

RUN sudo tar zxvf /opt/java.tar.gz

RUN sudo update-alternatives --install /usr/bin/java java /opt/jdk1.8.0_74/bin/java 100
RUN sudo update-alternatives --install /usr/bin/javac javac /opt/jdk1.8.0_74/bin/javac 100

```
# 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
```

Create java program

```
public class javaprogram{
        public static void main(String[] args){
                System.out.println("Hello world");
        }
}
```

Compile and run it.

```
root@a613e9cf0f1d:/opt# javac javaprogram.java
root@a613e9cf0f1d:/opt# ls
javaprogram.class  javaprogram.java
```

```
root@a613e9cf0f1d:/opt# java javaprogram
Hello world
```

Task 4 – Create your own linked container

To create mysql container that stores data in the host volume, create a directory

```
ubuntu@ip-172-31-7-183: ~
ubuntu@ip-172-31-7-183:/home$ ls
ubuntu
ubuntu@ip-172-31-7-183:/home$ cd ubuntu/
ubuntu@ip-172-31-7-183:~$ ls
dockerfile
ubuntu@ip-172-31-7-183:~$ pwd
/home/ubuntu
ubuntu@ip-172-31-7-183:~$ sudo mkdir mysqldir
ubuntu@ip-172-31-7-183:~$ ls
dockerfile  mysqldir
```

docker run --name some-mysql -v /home/ubuntu/mysqldir:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=my-secret-pw -d mysql:tag

where **my-secret-pw** is password, **tag** is version number e.g. 8.0

e.g. **docker run --name some-mysql -v /home/ubuntu/mysqldir:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=abcde -d mysql:8.0**

Find running process with

sudo docker ps -a -f status=running

Execute interactive terminal –it inside a container using exec and containerID

sudo docker exec -it containerID /bin/bash

```
ubuntu@ip-172-31-7-183:~/mysqldir$ sudo docker ps -a -f status=running
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
                    NAMES
2f8542bc24de        php:7.0-apache      "docker-php-entryp..."   4 minutes ago       Up 4 minutes        80/tcp
                    mystifying_pare
ea11b44952de        ubuntu              "bash"                   2 hours ago         Up 2 hours
                    clever_euclid
35f39df042c9        mysql:8.0           "docker-entrypoint..."   19 hours ago        Up 2 hours          3306/tcp
                    sqldirectory
3ee2780a2a10        jenkins             "/bin/tini -- /usr..."   24 hours ago        Up 2 hours          0.0.0.0:8080-
0:50000->50000/tcp  dreamy_jang
ubuntu@ip-172-31-7-183:~/mysqldir$ sudo docker exec -it 35f39df042c9 /bin/bash
root@35f39df042c9:/# ls
bin   dev                      entrypoint.sh  home  lib64  mnt  proc  run   srv  tmp  var
boot  docker-entrypoint-initdb.d  etc          lib   media  opt  root  sbin  sys  usr
root@35f39df042c9:/# cd /var/lib/mysql
root@35f39df042c9:/var/lib/mysql# ls
auto.cnf    client-cert.pem  ib_logfile0  ibtmp1                performance_schema  server-cert.pem  sys_4.SDI
ca-key.pem  client-key.pem   ib_logfile1  mysql                 private_key.pem     server-key.pem
ca.pem      ib_buffer_pool   ibdata1      performance_sche_3.SDI  public_key.pem      sys
root@35f39df042c9:/var/lib/mysql#
```

For PHP project, pull php image:
sudo docker pull php
When running the container mount the directory to a newdirectory in php file path
sudo docker run -it -v /home/ubuntu/mysqldir:/home/newdir php:7.0-apache bash

```
ubuntu@ip-172-31-7-183:~/mysqldir$ sudo docker run -it -v /home/ubuntu/mysqldir:/home/newdir php:7.0-apache bash
root@2f8542bc24de:/var/www/html# cd /home/newdir
root@2f8542bc24de:/home/newdir# ls
auto.cnf    client-cert.pem  ib_logfile0  ibtmp1                performance_schema  server-cert.pem  sys_4.SDI
ca-key.pem  client-key.pem   ib_logfile1  mysql                 private_key.pem     server-key.pem
ca.pem      ib_buffer_pool   ibdata1      performance_sche_3.SDI  public_key.pem      sys
```

Task 5 - Create your own docker-compose file

Create a directory to be used as shared folder.

/home/Ubuntu/mysqldir contains many files from task4

```
ubuntu@ip-172-31-7-183:~$ cd mysqldir/
ubuntu@ip-172-31-7-183:~/mysqldir$ ls
auto.cnf    ca.pem           client-key.pem  ibdata1      ib_logfile1  performance_sche_3.SDI  phptext.txt
ca-key.pem  client-cert.pem  ib_buffer_pool  ib_logfile0  mysql        performance_schema      private_key.pem
```

Create **docker-compose.yml** file as below:

```
version : '2'

services:
  db:
    image: mysql:8.0
    ports:
      - "3333:3333"
    volumes:
      - /home/ubuntu/mysqldir:/var/www/html
    environment:
      MYSQL_ROOT_PASSWORD: passwerd
  php:
    image: php:7.0-apache
    links:
      - db:db
    ports:
      - "80:80"
    volumes:
      - /home/ubuntu/mysqldir:/var/www/html
```

PHP and mysql are running

```
ubuntu@ip-172-31-7-183:~$ sudo docker-compose up -d
Starting ubuntu_db_1
Recreating ubuntu_php_1
ubuntu@ip-172-31-7-183:~$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
ac8da15d92a4        php:7.0-apache      "docker-php-entryp..."   6 seconds ago       Up 5 seconds        0.0.0.0:8
4557d923d586        mysql:8.0           "docker-entrypoint..."   22 minutes ago      Up 5 seconds        3306/tcp,
```

Execute the first container and check shared directory which contains files as expected.

```
ubuntu@ip-172-31-7-183:~$ sudo docker exec -ti ac8da15d92a4 bash
root@ac8da15d92a4:/var/www/html# ls
auto.cnf    ca.pem           client-key.pem  ib_logfile0  ibdata1  performance_sche_3.SDI  phptext.txt
ca-key.pem  client-cert.pem  ib_buffer_pool  ib_logfile1  mysql    performance_schema      private_key.pem
```

Execute the second container and it can be seen that the directory is successfully mounted as the files expected are present.

```
ubuntu@ip-172-31-7-183:~$ sudo docker exec -ti 4557d923d586 bash
root@4557d923d586:/# ls
bin  boot  dev  docker-entrypoint-initdb.d  entrypoint.sh  etc  home  lib  lib64  media  mnt  opt  proc  ro
root@4557d923d586:/# cd /var/www/html
root@4557d923d586:/var/www/html# ls
auto.cnf    ca.pem           client-key.pem  ib_logfile0  ibdata1  performance_sche_3.SDI  phptext.txt
ca-key.pem  client-cert.pem  ib_buffer_pool  ib_logfile1  mysql    performance_schema      private_key.pem
```

Task 6 – Install Kubernetes

Install conjure up which also installs juju

sudo snap install conjure-up --classic

Add credentials of aws using juju

juju add-credential aws

```
ubuntu@ip-10-0-0-7:~$ juju list-credentials
Cloud  Credentials
aws    AcademyTrainee10
```

Update cloud and check it

```
ubuntu@ip-10-0-0-7:~$ juju update-clouds
Fetching latest public cloud list...
Your list of public clouds is up to date, see `juju clouds`.
ubuntu@ip-10-0-0-7:~$ juju clouds
Cloud        Regions  Default          Type        Description
aws               14  us-east-1        ec2         Amazon Web Services
aws-china          1  cn-north-1       ec2         Amazon China
aws-gov            1  us-gov-west-1    ec2         Amazon (USA Government)
azure             24  centralus        azure       Microsoft Azure
azure-china        2  chinaeast        azure       Microsoft Azure China
cloudsigma         5  hnl              cloudsigma  CloudSigma Cloud
google             7  us-east1         gce         Google Cloud Platform
joyent             6  eu-ams-1         joyent      Joyent Cloud
oracle             5  uscom-central-1  oracle
rackspace          6  dfw              rackspace   Rackspace Cloud
localhost          1  localhost        lxd         LXD Container Hypervisor
```

Bootstrap a controller to manage our cluster

```
ubuntu@ip-10-0-0-7:~$ juju bootstrap aws/eu-west-2
Creating Juju controller "aws-eu-west-2" on aws/eu-west-2
Looking for packaged Juju agent version 2.1.3 for amd64
Launching controller instance(s) on aws/eu-west-2...
 - i-05759fcc1690399ff (arch=amd64 mem=4G cores=2)
Fetching Juju GUI 2.6.0
```

Deploy a cluster of 9 nodes

juju deploy canonical-kubernetes

```
ubuntu@ip-10-0-0-7:~$ juju deploy canonical-kubernetes
Located bundle "cs:bundle/canonical-kubernetes-38"
Deploying charm "cs:~containers/easyrsa-9"
added resource easyrsa
Deploying charm "cs:~containers/etcd-34"
added resource etcd
added resource snapshot
Deploying charm "cs:~containers/flannel-15"
added resource flannel
Deploying charm "cs:~containers/kubeapi-load-balancer-11"
application kubeapi-load-balancer exposed
Deploying charm "cs:~containers/kubernetes-master-19"
```

Check the status

```
ubuntu@ip-10-0-0-7:~$ juju status
Model    Controller      Cloud/Region   Version
default  aws-eu-west-2   aws/eu-west-2  2.1.3

App                    Version  Status   Scale  Charm                  Store       Rev  OS
easyrsa                         waiting    0/1  easyrsa                jujucharms    9  ubuntu
etcd                            waiting    0/3  etcd                   jujucharms   34  ubuntu
flannel                         waiting      0  flannel                jujucharms   15  ubuntu
kubeapi-load-balancer           waiting    0/1  kubeapi-load-balancer  jujucharms   11  ubuntu
kubernetes-master               waiting    0/1  kubernetes-master      jujucharms   19  ubuntu
kubernetes-worker               waiting    0/3  kubernetes-worker      jujucharms   23  ubuntu

Unit                     Workload  Agent       Machine  Public address  Ports  Message
easyrsa/0                waiting   allocating  0                               waiting for machi
etcd/0                   waiting   allocating  1                               waiting for machi
```

Task 7 – Creating your first Single Container Pod

Install kubectl using curl

curl -LO https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl

Make it executable

chmod +x ./kubectl

Move it to environmental PATH

sudo mv ./kubectl /usr/local/bin/kubectl