**Documentation on what we did till date:**

Very first thing, I configured virtual machine which is currently running Ubuntu 15.10 server on it.

**To install JDK and configure GIT:**

In order to install GIT, first we need to have java running on it.

1. Install java using command:

**#apt-get install openjdk-7-jre-headless**

**(Note: Since that version of java works well with Jenkins)**

1. Ensure the version of java which you have installed by using :

**#java –version**

1. To install GIT run the following command:

**#apt-get install git-core**

1. Check the version of GIT which we have installed by using the command:

**#git version**

1. Every git user must introduce himself to git, by running these twoi commands:

**#git config –global user.email** [**Sameer.shukur.m@gmail.com**](mailto:Sameer.shukur.m@gmail.com)

**#git config –global user.name “Sameer\_shukur”**

1. Run: **#ssh-keygen –t rsa**
2. Run: **#cd /root/.ssh**
3. Run: **#cat id\_rsa.pub**
4. Copy the entire key and go to <https://github.com>
5. Click on profile -> settings -> SSH and GPG keys -> New SSH key -> Enter Title -> <Paste the key in key section> -> click on Add SSH Key
6. Clone the Game-Of-Life git repository by using the following command:

**#git clone** [**https://github.com/sameer-shukur/Game-of-life.git**](https://github.com/sameer-shukur/Game-of-life.git)

**To install and configure Maven:**

1. Run the command to install Maven:

**#apt-get install maven**

1. Now we need to set the environmental variables for that edit **.bash\_profile** using **vi editor:**

vi ~/.bash\_profile

if [ -f ~/.bashrc ]; then

. ~/.bashrc

fi

if [ -d ~/bin ] ; then

PATH=~/bin:"${PATH}"

fi

JAVA\_HOME=/usr/lib/jvm/java-7-openjdk-amd64/jre

export JAVA\_HOME

PATH=$PATH:$JAVA\_HOME/bin

export M2\_HOME=/usr/share/maven

export M2=$M2\_HOME/bin

export MAVEN\_OPTS="-Xms256m -Xmx512m"

export PATH

1. Check whether environmental variables have been set correctly by using command:

**#echo $JAVA\_HOME**

**#echo $M2**

1. Now check the maven version by using :

**#mvn –version**

**To install Apache Tomcat:**

1. Download **ApacheTomcat.8.0.30.tar.gz** file in local system
2. Use **FileZilla** or **Winscp** software to transfer the downloaded Apache file to remote machine
3. Unzip the file using the command:

**#tar –zxvf ApacheTomcat.8.0.30.tar.gz**

1. Open Browser in your local machine and go to <https://portal.azure.com/> and login to your azure account
2. Now we need to set Inbound values, for that:

Select **VM** -> **Network Interfaces** -> select your **Network** -> click on **Network Security Group** -> under settings you will find **Inbound security rules**

1. Open **Inbound security rules**, click on **ADD**, give any name and set Destination port range to **8080** and click on OK.
2. After unzip enter into ApacheTomcat directory and then start the service using the following command:

**#./startup.sh**

1. Go to browser in your local system and type **13.76.47.21:8080** to check whether Apache server is working or not.
2. Now most importantly we need to install maven in Game-Of-Life directory where **pom.xml** file is present by using following command:

**#mvn clean install**

**To install and configure Jenkins:**

1. First run these commands to install Jenkins:

**#wget -q -O - https://jenkins-ci.org/debian/jenkins-ci.org.key | sudo apt-key add -**

**#sudo sh -c 'echo deb http://pkg.jenkins-ci.org/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'**

**#sudo apt-get update**

**#sudo apt-get install jenkins**

1. Enter into ApacheTomcat directory and then start the service using the following command:

**#./startup.sh**

1. Go to browser in your local system and type **13.76.47.21:8080/jenkins**
2. It will ask for the Initial Administrator Password. To get the password Run: **cat cd /root/.jenkins/secrets/initialAdminPassword**
3. Copy the entire password and paste in the required field in the browser.
4. Now configure Git and Maven in Jenkins by installing their **plugins**.

**To install and configure JAVA and MYSQL for SonarQube:**

1. The **pre-requisites** for sonar installation are:
   1. JAVA jre
   2. DBMS ( MySQL )
2. First **install JAVA jre**. Update the package index:

**#sudo apt-get update**

1. Then make a java runtime environment:
2. **#sudo apt-get install default-jre** and then the java development kit: **#sudo apt-get install default-jdk**
3. Now **install MySQL server** in our Ubuntu server. First update your system: **#sudo apt-get update** and then run: **#sudo apt-get upgrade**
4. For installing MySQL server: **#sudo apt-get install MySQL-server**. It will ask to generate password.
5. Run the **mysql\_secure\_installation** script to address several security concerns in a default MySQL installation: **#sudo mysql\_secure\_installation**.
6. You will be given the choice to change the MySQL root password, remove anonymous user accounts, disable root logins outside of localhost, and remove test databases. You must answer **yes** to all the constraints.
7. To login as root user: **#mysql -u root –p**. It will ask for your password.
8. Then create the database and a user:

**CREATE DATABASE sonar CHARACTER SET utf8 COLLATE utf8\_general\_ci;**

**CREATE USER 'sonar' IDENTIFIED BY 'sonar';**

**GRANT ALL ON sonar.\* TO 'sonar'@'%' IDENTIFIED BY 'sonar';**

**GRANT ALL ON sonar.\* TO 'sonar'@'localhost' IDENTIFIED BY 'sonar';**

**FLUSH PRIVILEGES;**

**To install and configure SonarQube:**

1. Go to your **virtual machine > settings > Network Interfaces > <user\_name> > Network Security Group > Inbound Security Rules.**
2. Give **Name > SonarQube** & **Port > 9000**.
3. Download and unzip SonarQube distribution: **wget** [**http://dist.sonar.codehaus.org/sonarqube-5.1.zip**](http://dist.sonar.codehaus.org/sonarqube-5.1.zip)
4. **Unzip** that package: **unzip sonarqube-5.1.zip**.
5. Now **move** the whole directory to /opt/sonar : **mv sonarqube-5.1 /opt/sonar**
6. Then we have to **edit** sonar.properties: **vi /opt/sonar/conf/sonar.properties**
7. **Uncomment** these lines from the text:

* **sonar.jdbc.username=sonar**
* **sonar.jdbc.password=sonar**
* **sonar.jdbc.url=jdbc:mysql://localhost:3306/sonar?useUnicode=true&characterEncoding=utf8&rewriteBatchedStatements=true&useConfigs=maxPerformance**

1. Now we have to **change** web server settings by editing these setting on web server settings part:

* **sonar.web.host=127.0.0.1**
* **sonar.web.context=/sonar**
* **sonar.web.port=9000**

1. To implement **SonarQube server as a service**, **copy** sonar.sh file to etc/init.d/sonar: **sudo cp bin/linux-x86-64/sonar.sh /etc/init.d/sonar**
2. Open this file by **sudo vi /etc/init.d/sonar** and **insert** these two lines:

* **SONAR\_HOME=/opt/sonar**
* **PLATFORM=linux-x86-64**

And **modify** the following lines:

* **WRAPPER\_CMD="${SONAR\_HOME}/bin/${PLATFORM}/wrapper"**
* **WRAPPER\_CONF="${SONAR\_HOME}/conf/wrapper.conf"**
* **PIDDIR="/var/run"**

1. Now register **SonarQube as a linux service**:

* **sudo update-rc.d -f sonar remove**
* **sudo chmod 755 /etc/init.d/sonar**
* **sudo update-rc.d sonar defaults**

1. Now we have to **start** SonarQube server. Run: **sudo /opt/sonar/bin/linux-x86-64/sonar.sh start**

And to **stop** SonarQube server. Run: **sudo /opt/sonar/bin/linux-x86-64/sonar.sh stop**

1. Visit SonarQube **web page** at **http://<server\_ip>:9000/sonar**

**Creating AWS account:**

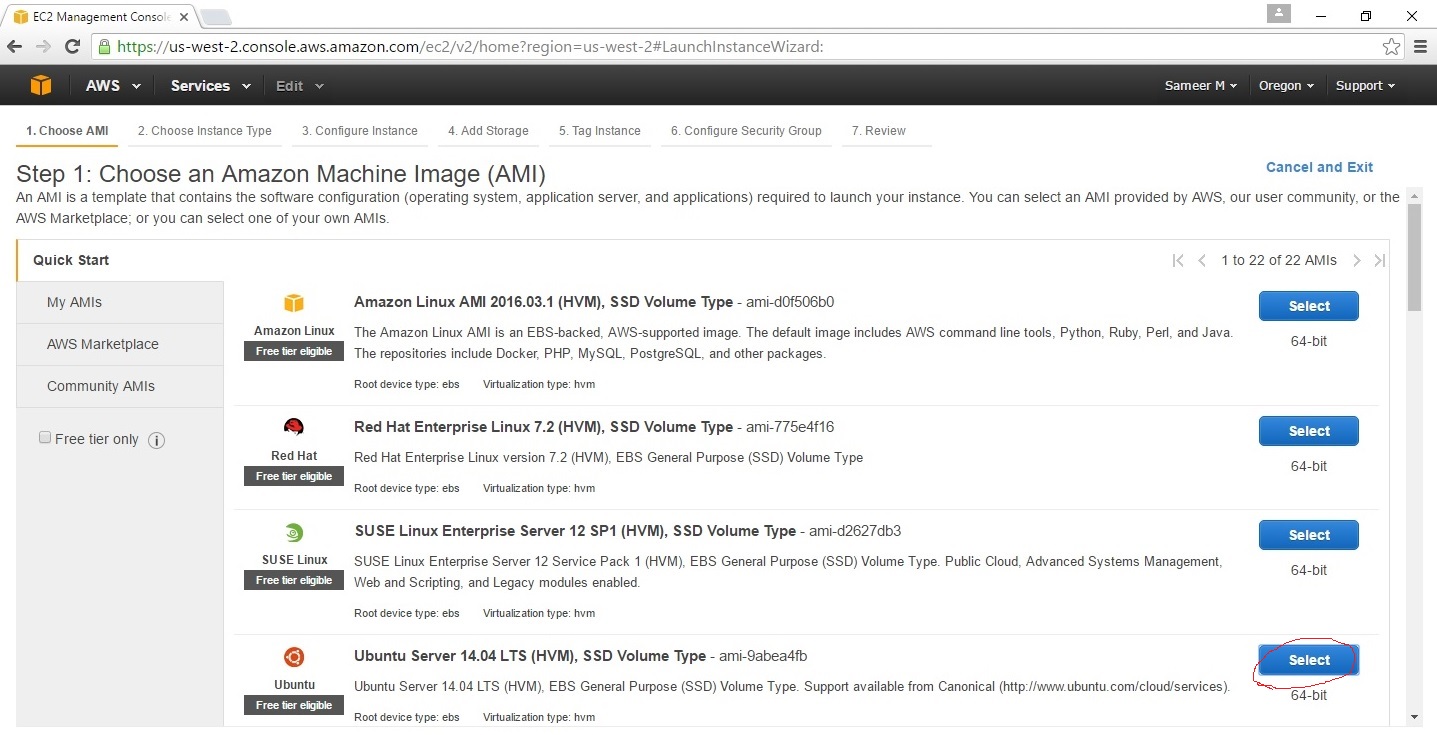
1. Go to the website:

<https://aws.amazon.com/>

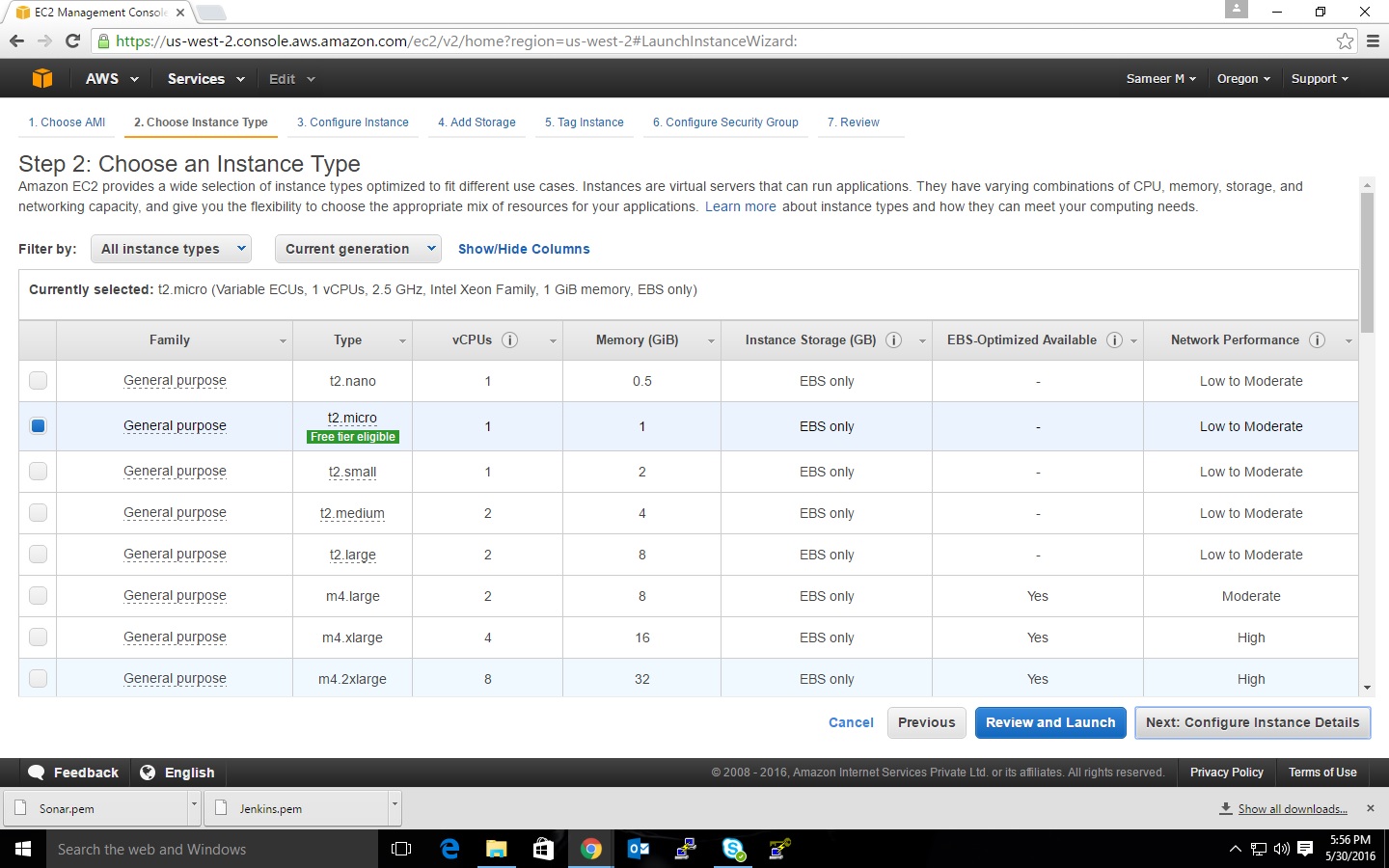
1. Click on sign in to console option
2. If you are a new user then click on **I am a new user** option
3. If you are an existing user then then enter your email ID or registered mobile number with password to sign in
4. Once we login it will prompt to enter all the credit details in order to use the amazon web service.
5. Amazon web service will charge Rs. 2.00 for account activation and provides 750 Hrs of free duration to access the web service and post charges will be applicable, so it’s better stop the instance whenever not in use.

**Creating instance in AWS:**

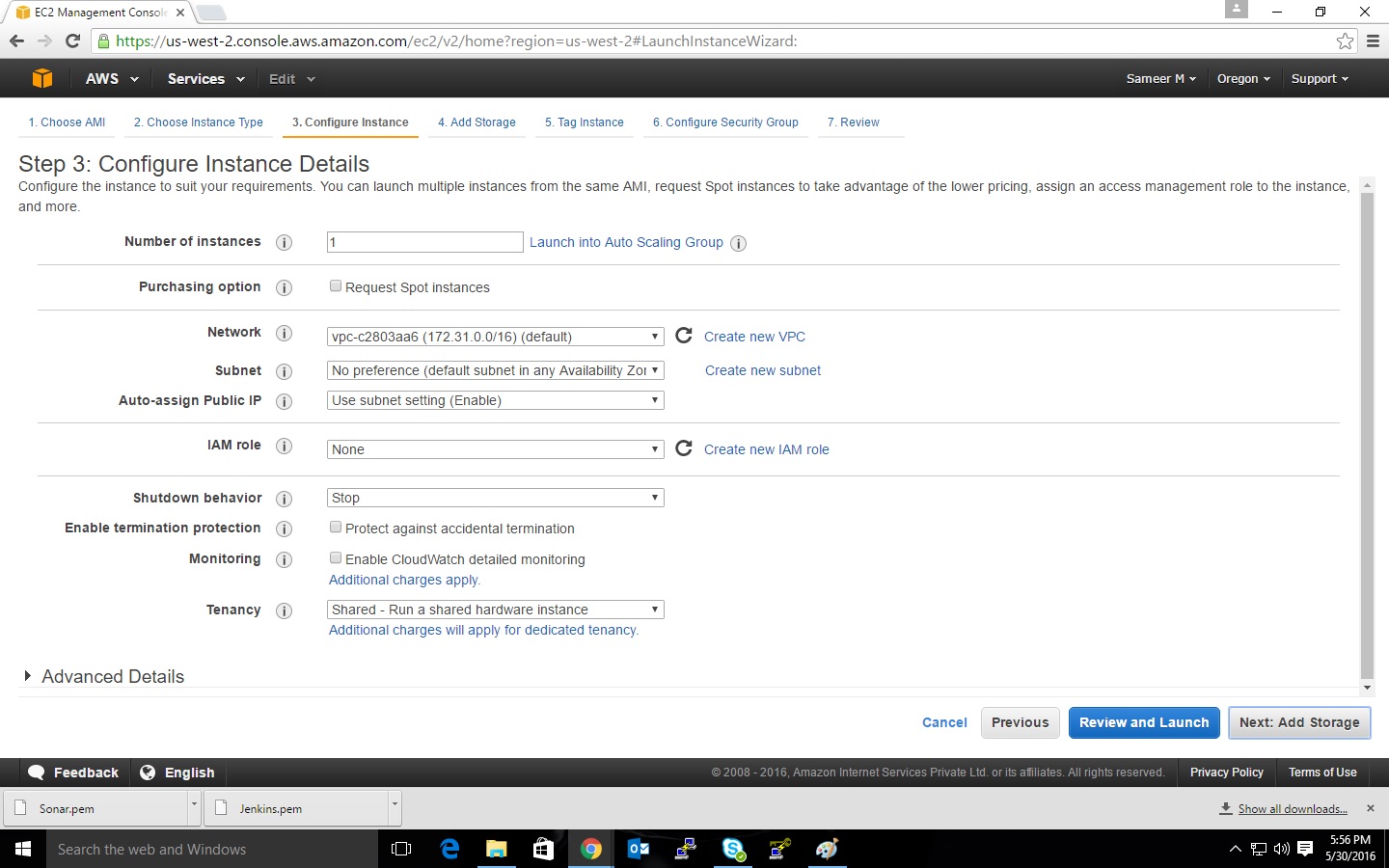
1. Once we login to AWS account we will be redirected to multiple stages to deploy a server or VM.
2. The first step is to choose an Amazon Machine Image (AMI), see the snapshot for reference. Since we are deploying a Linux server, I am using Ubuntu server 14.04



1. Next step is to choose an Instance Type. Since we are using a free account ensure that we are selecting General Purpose t2.micro free tier eligible instance as shown in the below snapshot:



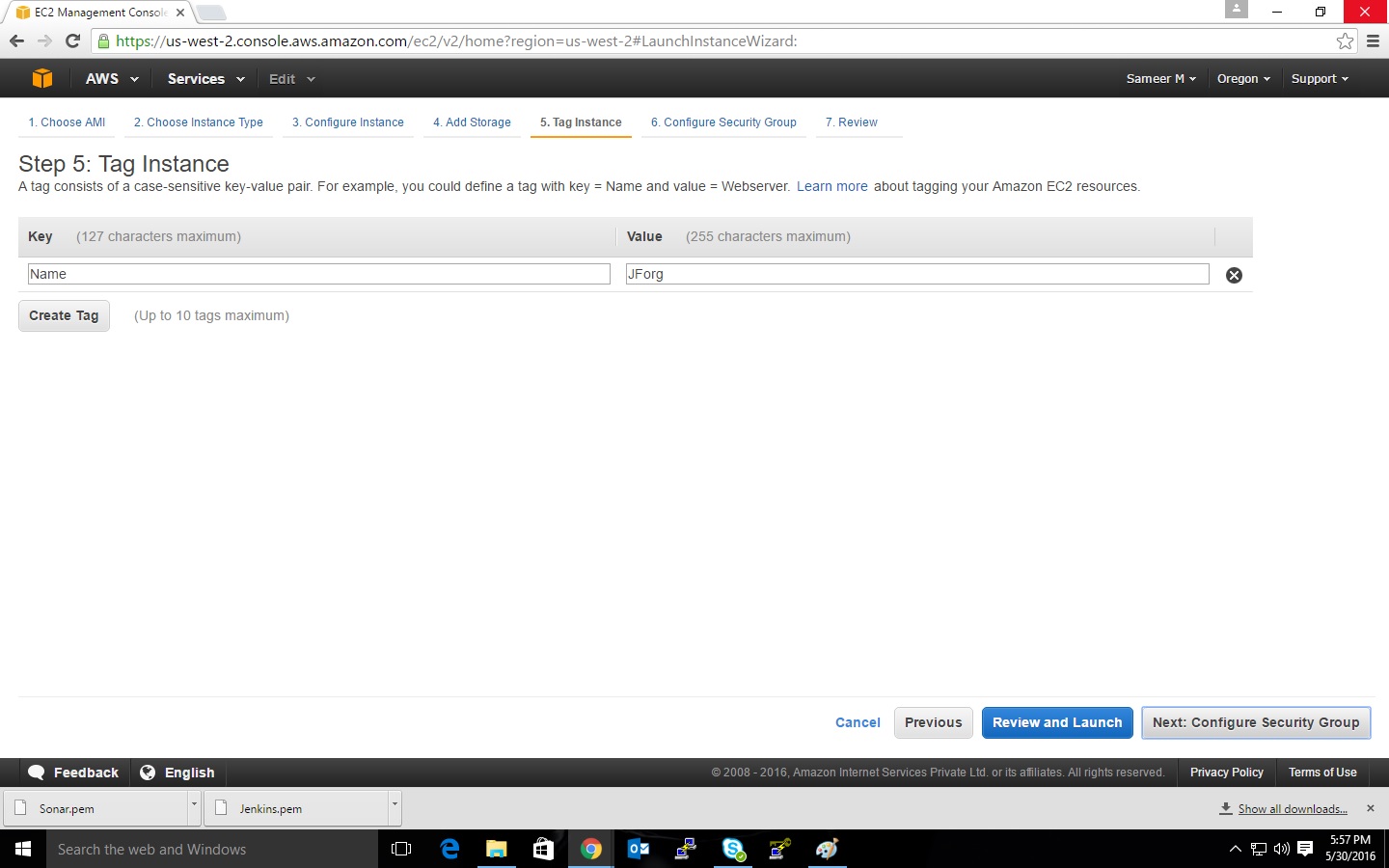
1. The next step is to configure Instance details. Since we are using free account no need to make any changes in this step and click on next button as shown below:



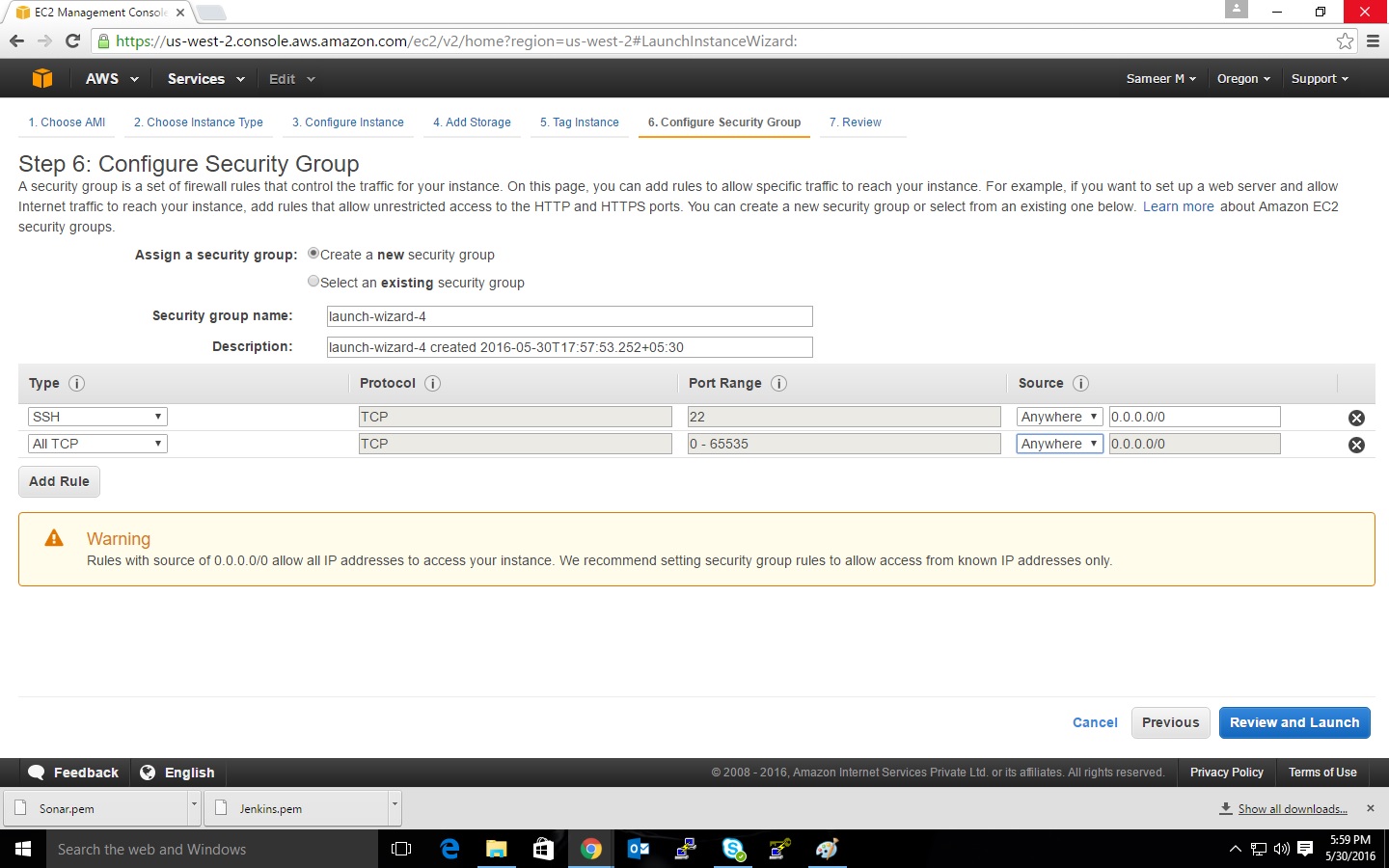
1. The next step is to add storage:



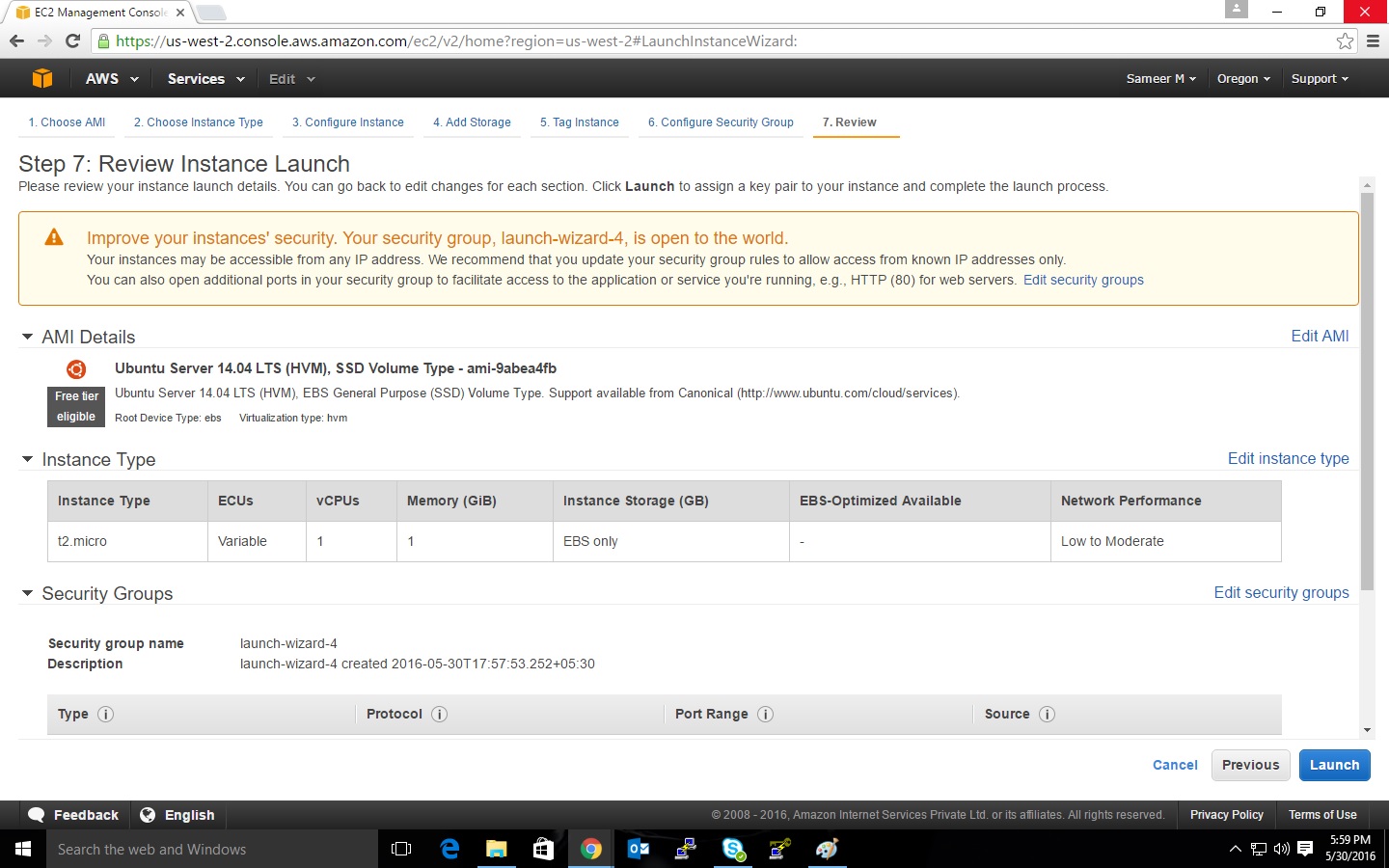
1. The next step is to tag instance consisting of key-pair value which is a case sensitive, here key= Name and value= Webserver, as shown in the below snapshot.



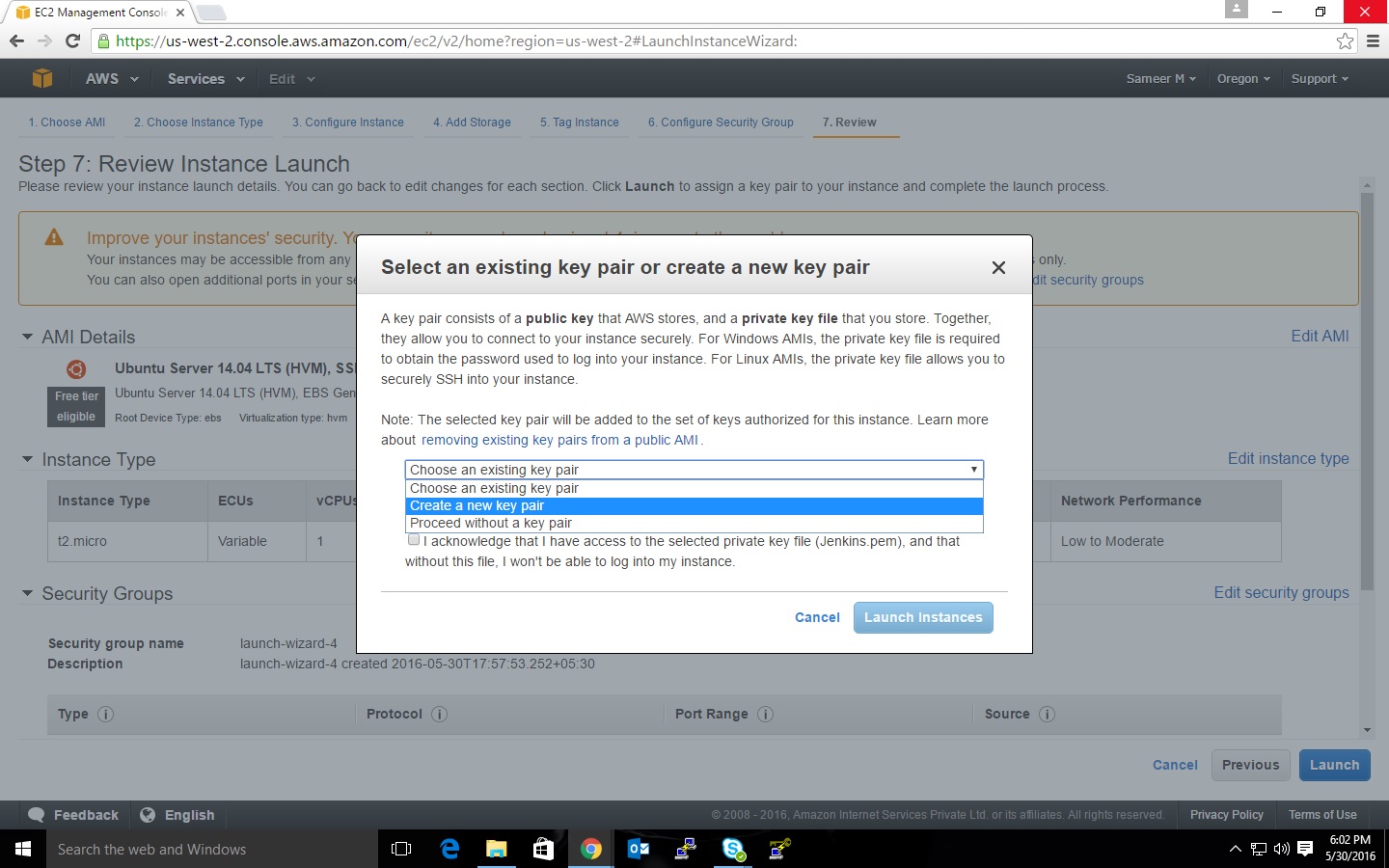
1. Next step is to configure security group in which we have to open ports, for that click on **Add Rule** button. Under “Type” field select **All TCP** and under “Protocol” select **TCP** and in “port range” select **0-65535** and under “source” select it as **anywhere.** Next click on **Review and lunch** as shown in the below snapshot:



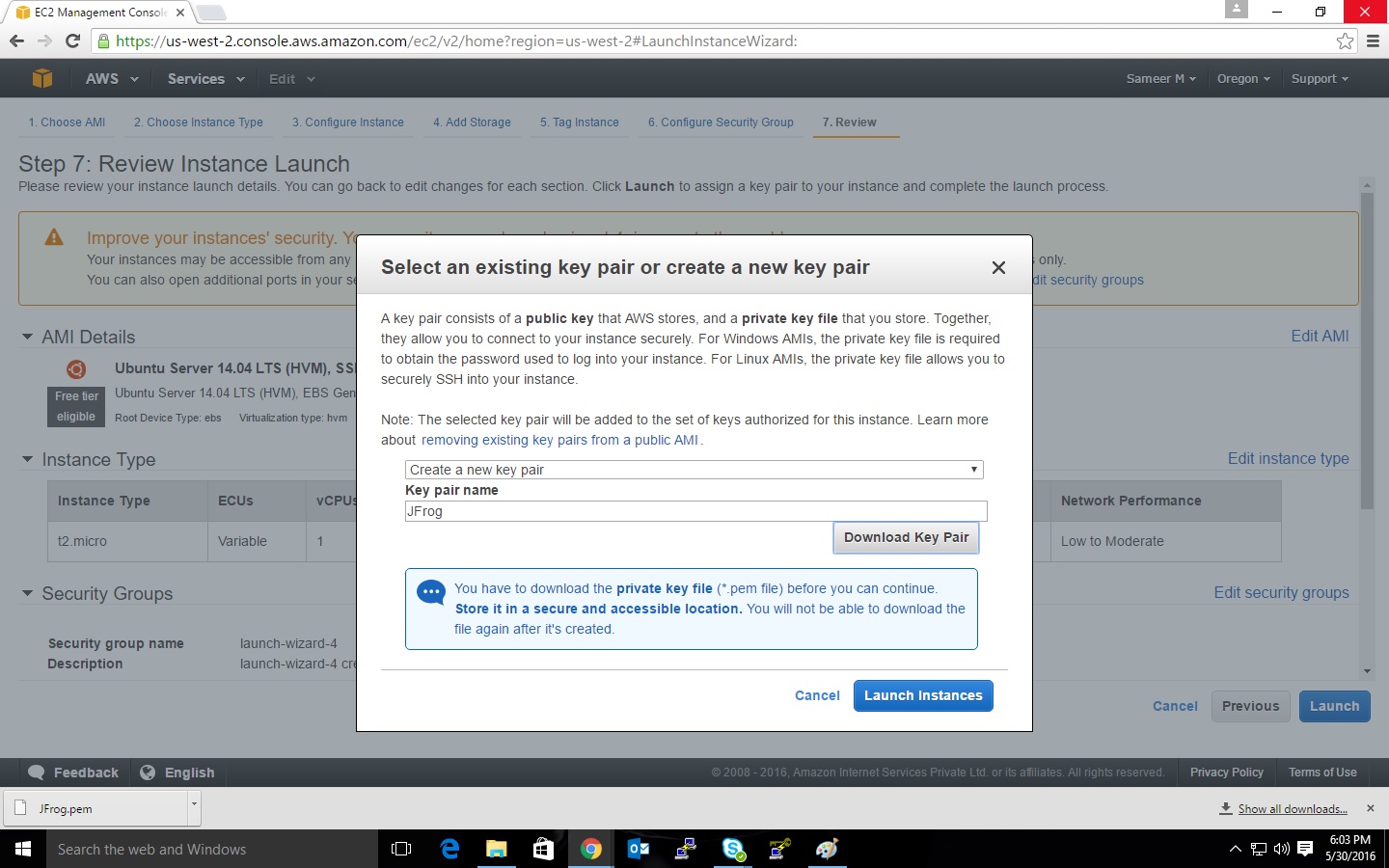
1. Next step is to Review the Instance Launch, in this step verify everything you have selected is correct or not, if not then click on previous to change it.



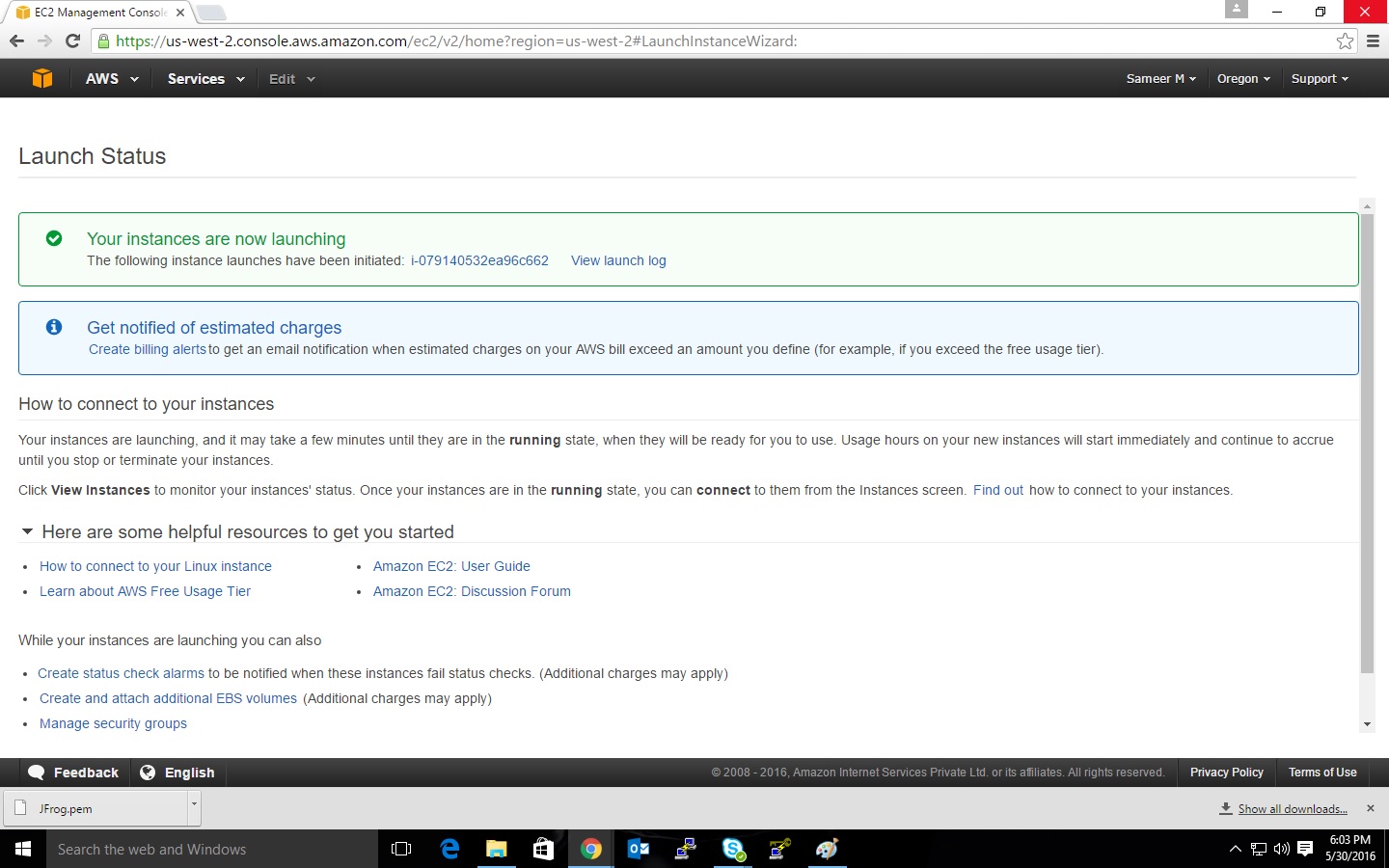
1. Once we click on launch button, a dialogue box will be prompted asking us to select an existing key pair or to create a new key pair. In this we have to select on that drop down list to select option **“create a new key pair”** which will create a (**.pem**) file that can be used to generate a (**.ppk**) file. We will discuss about .**ppk** file in future steps.



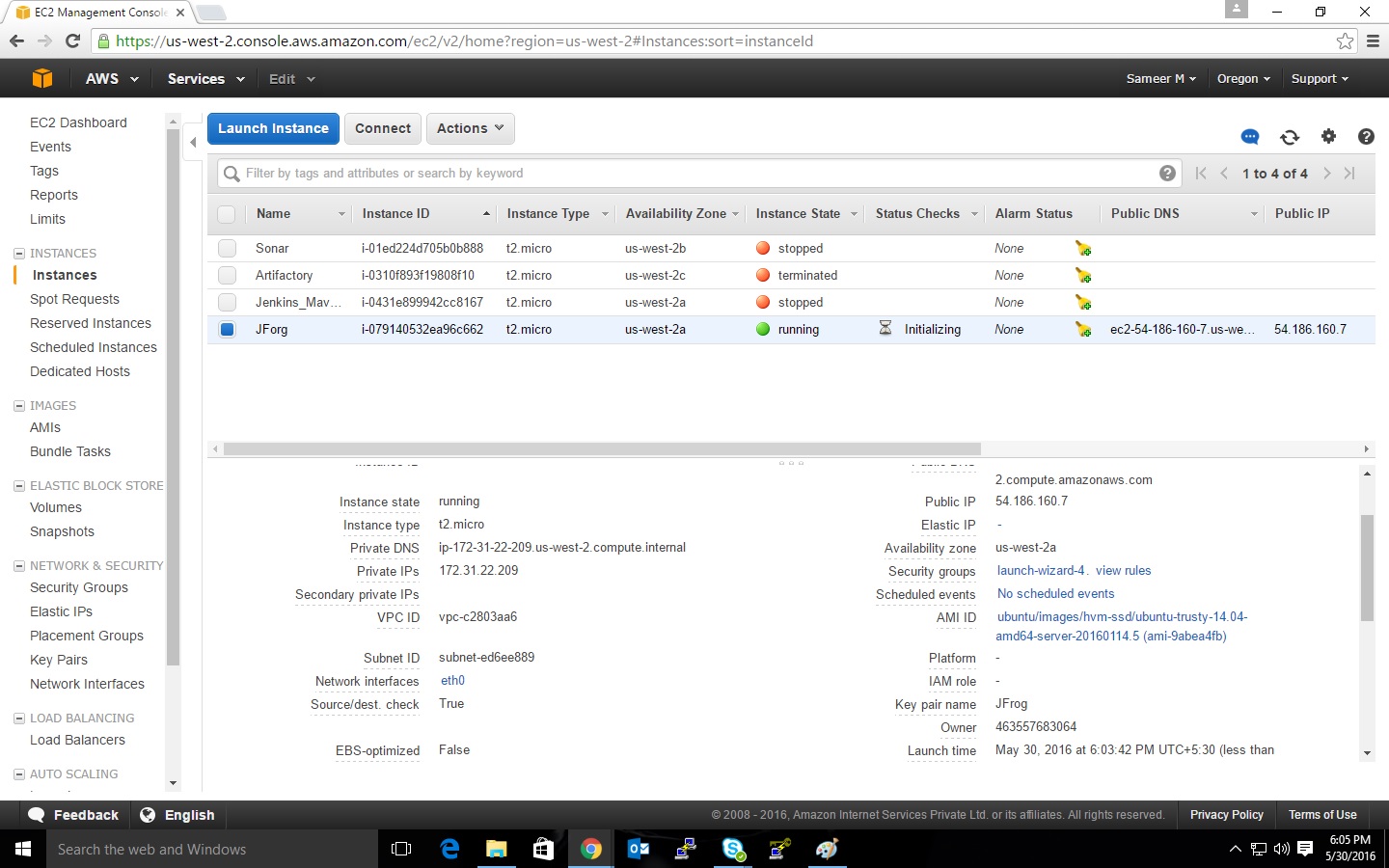
1. Once we select create a new key pair option, Next step is to click on **Download Key pair** button to download **.pem** file. It is important to download the **.pem** file, if we miss this step then we cannot download the file again. Now once we download that **.pem** file, click on launch instance.



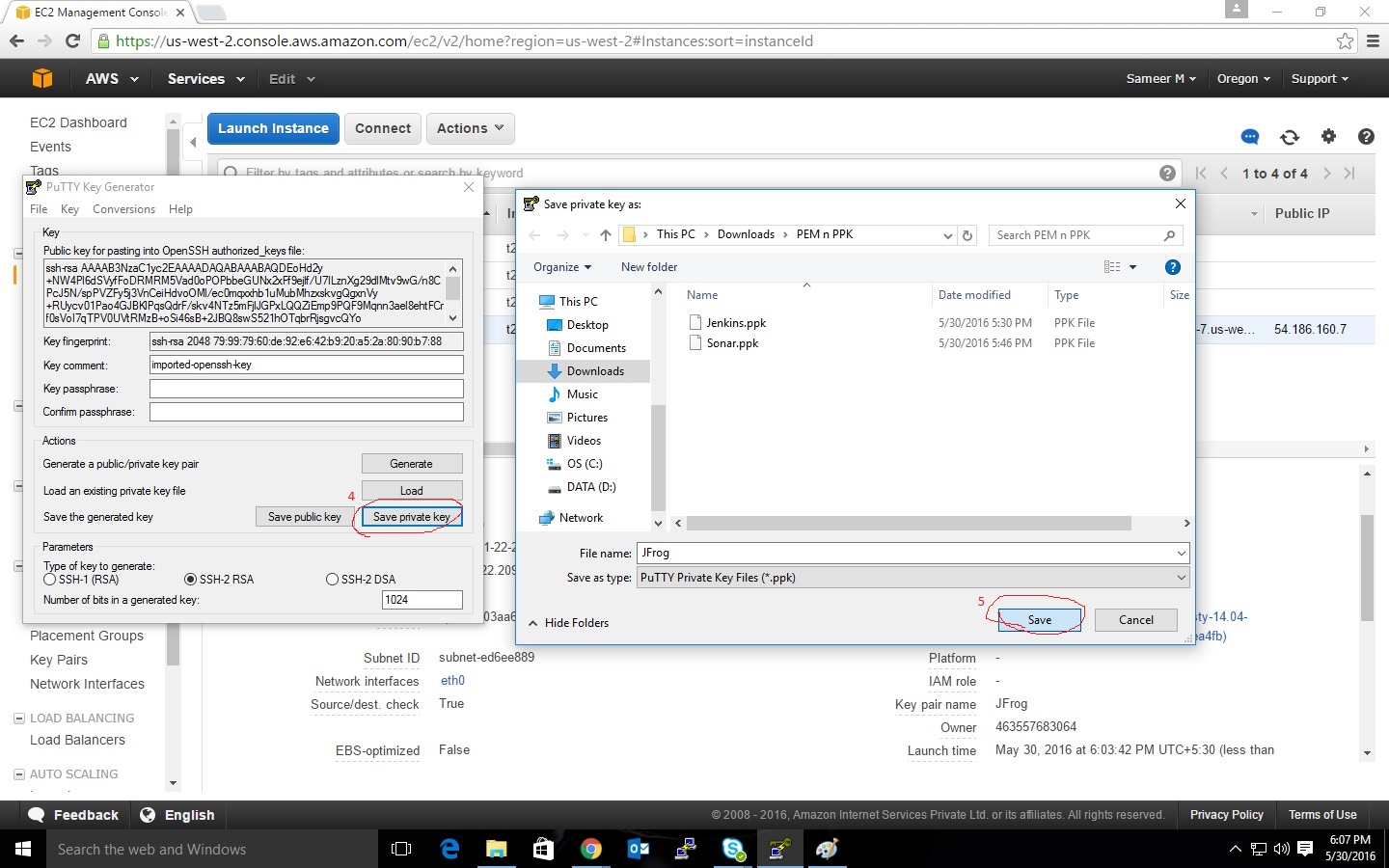
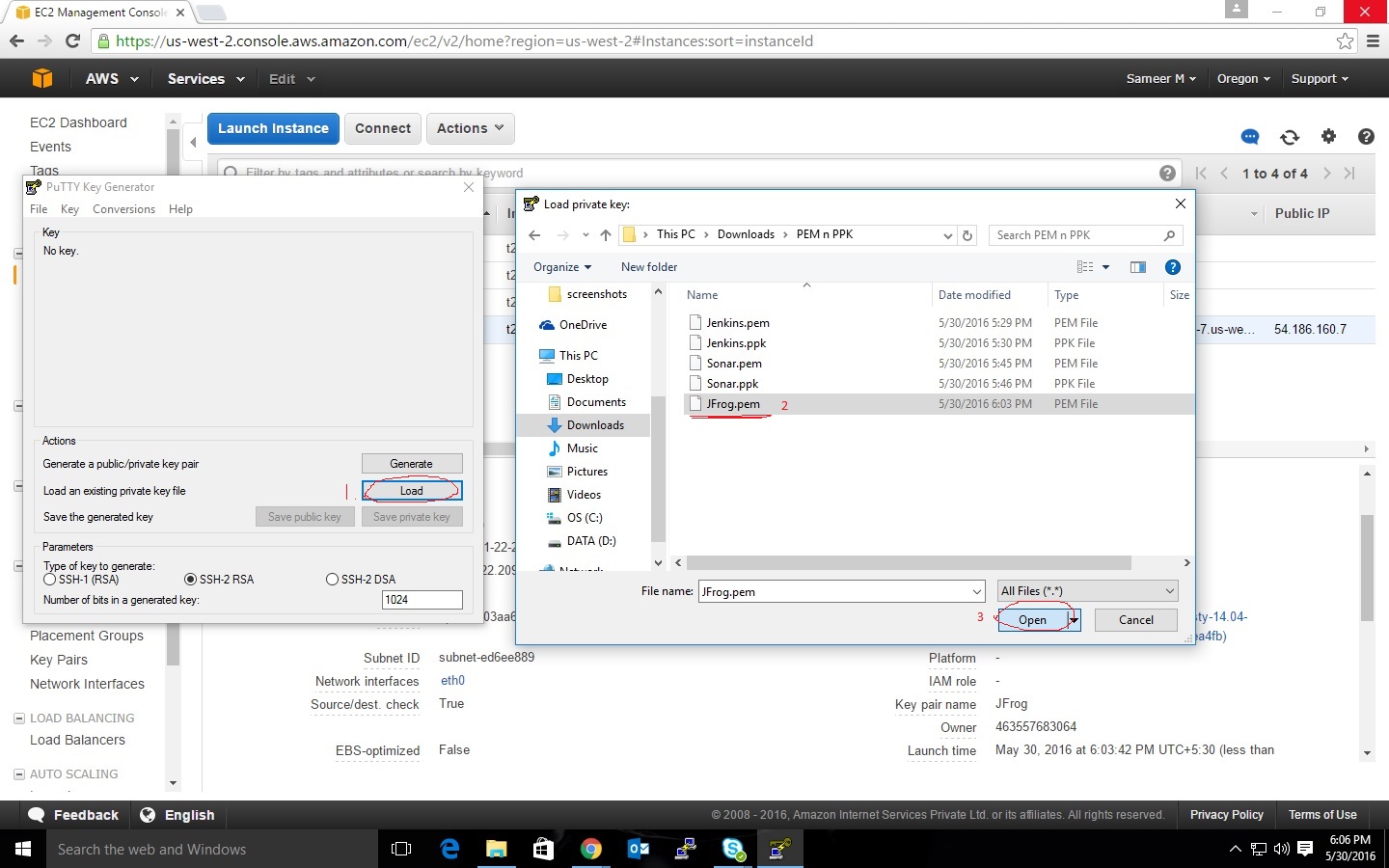
1. Once the instance is created you will see this page in your browser as shown in the below snapshot:



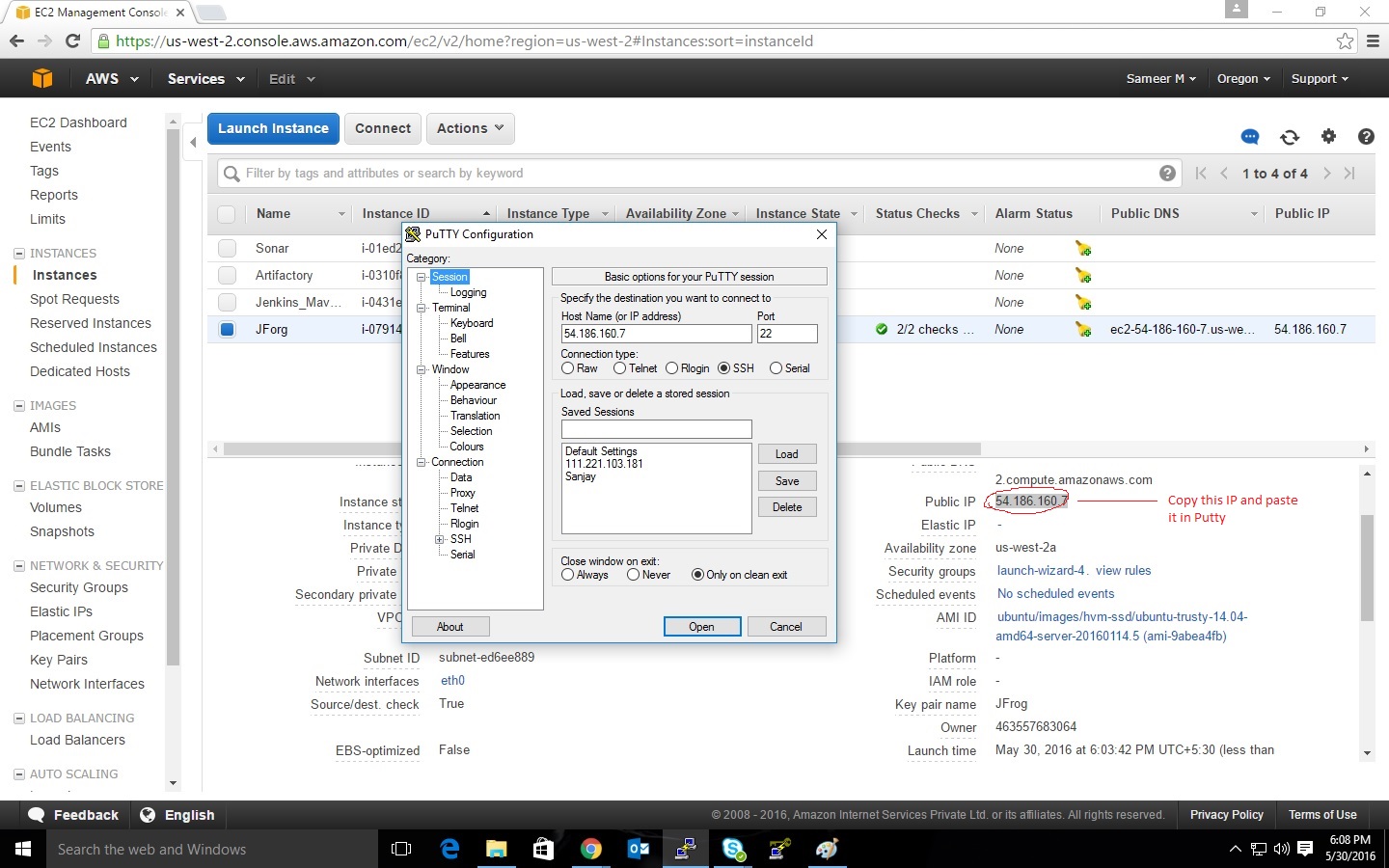
1. Next step is to click on view instances, which will show all the created instances. Suppose if we want to start any instance, then click on any instance and in the top we can see an option called **Actions.**  Click on that under which we will see an option called **Instance State**. In Instance State, click **start** to start an instance, **stop** to stop an instance, **Reboot** to restart server and **Terminate** to delete that server.



1. Once the server is started, we need to access it, which can be done by using SSH tools like **Putty** and **PuttyGen.** PuttyGen is used to generate a public/private key.
2. Open PuttyGen and perform the following steps to generate a **.ppk** file:



1. Now copy the public IP and open Putty and paste the IP address in Host Name.



1. Now navigate to **SSH** in left Pane, expand it. We can see **Authentication** click on it. Now on the right pane, we can see an option called **Allow agent forwarding** check that radio box. Now browse for the private key (**.ppk**) and click on **open**.

