**CSU4400 Internet Applications – Laboratory 2**

**Introduction to Using Cloud Computing with AWS  
Prof Donal O’Mahony**

The objective of this exercise is to gain some minimal experience with Cloud Computing using the tools offered by the most popular providers in the field: Amazon Web Services (AWS)

**Phase 1 – Creating a bare-bones virtual machine running Linux**

Login to RosettaHub.com – take note of what credit you have remaining. Click on “Go to AWS Console”. On the AWS Console select “EC2” to get the Elastic Compute Cloud(EC2) dashboard. This dashboard allows you to create virtual machines at Amazon cloud facilities around the world and to monitor and control their status

To create a machine instance click on “**Launch Instance**” button

Any machine instance needs to be created with a disk image – this controls what operating system is installed and what software is present when it boots up.

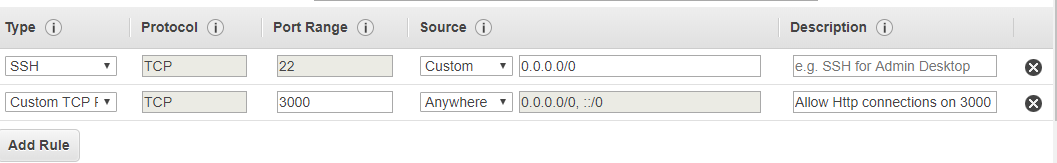
Select “**Amazon Linux 2 AMI….**”

This will create a machine that will boot up with just Amazon’s version of Linux installed

You are next asked what kind of (virtual) hardware you’d like in your machine – We will select a

**t2.micro** instance – This machine will get launched with a local disk containing your selected machine image.

Then hit “Review and Launch”

On the next screen, under “Security Groups”, we can see a summary of the Security settings for the machine – by default, it will accept SSH (Secure Shell) connections on Port 22 from any host on the internet. We’d like to be able to  make TCP connections in to Port 3000, so “Edit Security Groups” and add a rule that says:

now hit **Review &Launch**

As soon as the machine is booted, we are going to want to connect to it, so before we launch we need to setup the public/private keys that you will use to later SSH to the machine.

**Setting up the Key Pair – to allow you to SSH (using PuTTY) from a Windows machine**

A window appears entitled “Select an existing key pair or create a new key pair”

If this is the first time you have done this, select “**Create a new key pair**”

This will generate a public/private key pair – keep the public part up on Amazon’s AWS and install it on your newly created virtual machines. The private part gets downloaded as a file by your browser.

Give it a Name e.g. “CSU4400TestKeyPair” and click “**Download Key Pair**”

This will cause a file to be downloaded called csu4400TestKeyPair.pem which contains the private key. You need to save this file somewhere in your home directory for later use or email it to yourself (ignoring good security practice!)

The only problem is that it is in .pem (Privacy Enhanced Mail) format, and we need to convert the format before we can use it for our SSH program (Putty).. directions on how to do this later.

Hit “**Launch**” – this will create your machine wherever in the world you asked for it and start booting it up with a plain Linux machine image.

If you hit “View Instances” – you will be able to see your virtual machine – it tells you the status of your machine (“running, terminated etc”) and where in the world it is running.

Notice that it takes a few minutes to instantiate the machine – boot up with Linux etc.

**Connecting to and Logging into your newly created machine**

There are a number of ways of connecting to your newly created machine and we will start with the most convenient. Recently (June 2019), Amazon introduced a facility called AWS Instance-Connect that by-passes the normal Public Key Authentication method. ( it implants a temporary public key in the instance meta-data)

Select your running machine in the console and hit the “Connect” button at the top – select “EC2 Instance Connect” and hit “Connect” – a terminal window will appear off the browser with the shell prompt from the machine. Have a poke around in your new machine!

**Connecting Using your Public/Private Key Pair**

We will connect to the virtual machine using the Secure Shell (SSH) protocol. The client we will use on windows should already be installed on lab machines and is called PuTTY.

We want to connect to the DNS Name (say

“**ec2-34-247-69-249.eu-west-1.compute.amazonaws.com**”), login as the pre-configured user “ec2-user” – instead of using a password, we will authenticate using our private key.

First we need to convert the private key into a form that PuTTY can use. Invoke the utility “PuTTYgen” – hit “Load” and load up the .pem keyfile from earlier (if you did not move it, it will be in your Downloads directory) – when it is successfully loaded – “Save Private Key” to save it in .ppk format – say “CS7052TestKeyPair.ppk”.

Now start PuTTY – click on SSH..Auth – then “Browse” to load the ppk file

Click on session and enter the DNS name of your virtual machine in the “Host Name (or IP Address)” box and hit “Open” -when prompted with “login as” enter the username “ec2-user”

If all that worked, you are now logged in as a linux user to your virtual machine – poke around the file system to verify that all works as well as if you had a dedicated linux machine on your desk.

A very similar procedure is used when you want to copy files using WinSCP – in the WinSCP login dialog, enter user “ec2-user” – click on “Advanced” and point the “private key file” at your private key in .ppk format.

**Phase 2 – Getting a Node.js based Webserver running on your instance**

Your machine instance will need to have Node.js installed – you can do this with the following command sequence – entered through your putty session or your EC-2 Instance connect.

curl https://raw.githubusercontent.com/nvm-sh/nvm/v0.34.0/install.sh | bash

nvm install 10.16

npm install express

Use Winscp to transfer a copy of ExpressRandomAPI.js – included on Blackboard with this assignment

Run this program under Node.js on your created AWS instance

[ec2-user@ip-172-31-43-76 ~]$ node ExpressRandomAPI.js

Example app listening on port 3000!

Now invoke a browser (Chrome) on your local machine and, using the DNS name of your machine fetch a URL like:

<http://ec2-34-247-69-249.eu-west-1.compute.amazonaws.com:3000/random/10/20>

Fetch a randomly generated number from your Cloud-based server.

**[Optional] Try it out on a Windows-based cloud machine**

If there is time, you could leave the server running. Instantiate another machine instance – this time running windows. When you try to connect to this, your browser will download an .RDP file. When this is opened, a Remote Desktop Session will be opened to the Windows Server. Use a browser on that machine to connect to your Linux instance. [ See if you can circumvent the security restrictions!]

**Cleaning Up**

The machine instances you have created will continue to run, consuming rental costs – both for the running machine and the attached storage. RosettHub has some mechanisms for stopping idle machines, but when you are finished with the virtual machines you have created, it is best to go back to the EC2 console and right-click…Terminate (not just Stop!) any machines you have started – the resources will then go back to the pool.