AWS Security and Server Infrastructure

A demo of the basic setup of this lab is at: https://youtu.be/rhf4_1E_wAU

A Outline

Increasingly, we use the public cloud to build our information systems, and which reduces the cost in the investment in data centre costs, while providing the opportunity to quickly scale our server, network and data infrastructure. It is generally as pay-as-you-go model, and where we pay for CPU time, network bandwidth and data costs. The most popular public cloud provider is AWS (Amazon Web Services), and which provides EC2 (for compute), S3 (for data buckets), RDS (for databases) and AWS Network Firewall (for firewalls). Some of these services are outlined in Figure 1.

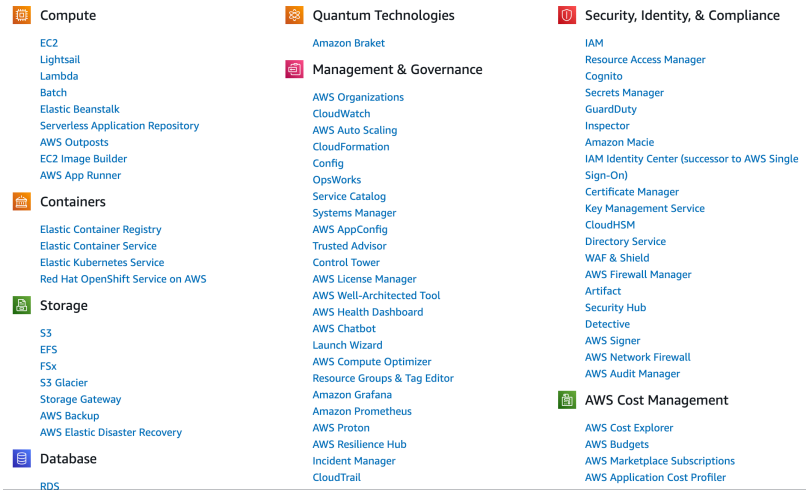

Figure 1: AWS Services

B Enabling your lab

You should have an AWS Academy login, so go to: https://awsacademy.instructure.com/ and log into the system and select **AWS Academy Learner Lab** (Figure 2).

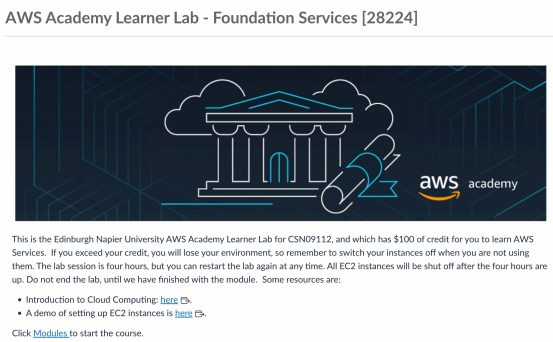

Figure 2: AWS Academy Learner Lab

Next, select “Modules”, and then “Learner Lab - Foundational Services”, and should have the lab environment (Figure 3).

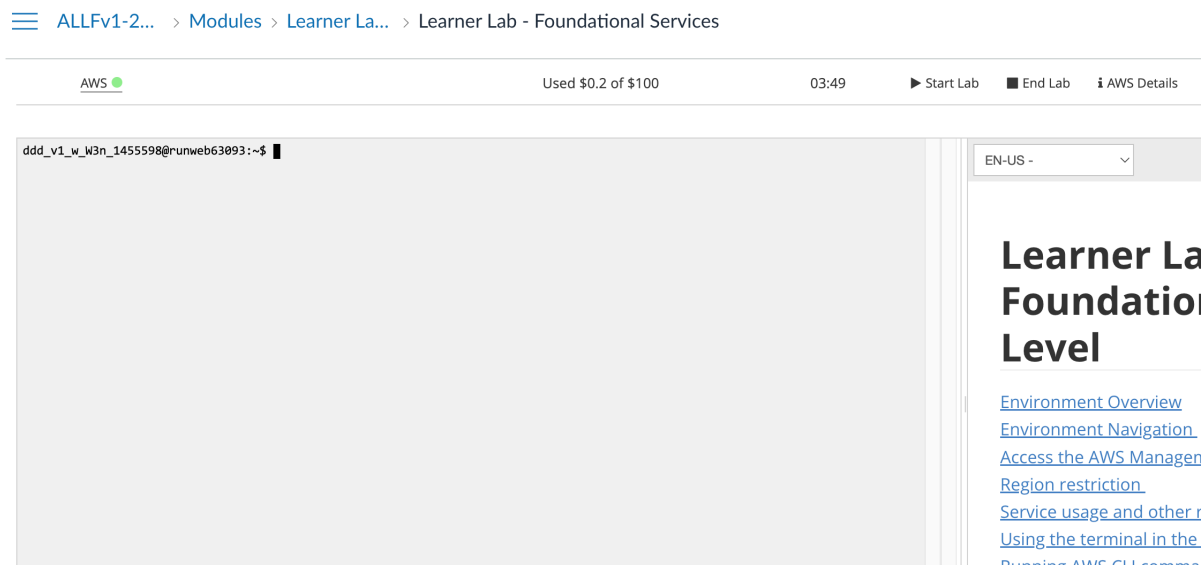

Figure 3: AWS Academy Learner Lab environment

Your unique account will have been generated, and you can access it with **aws_access_key_id** and **aws_secret_access_key** (from AWS details).

In the console you can interact with your AWS though the console (as you are already logged into AWS). Now, press the "Start Lab" button, and wait for the AWS light to go green. Once, green, you can click on it, and open up your AWS Management console. After this, just select EC2, and you should see your EC2 environment.

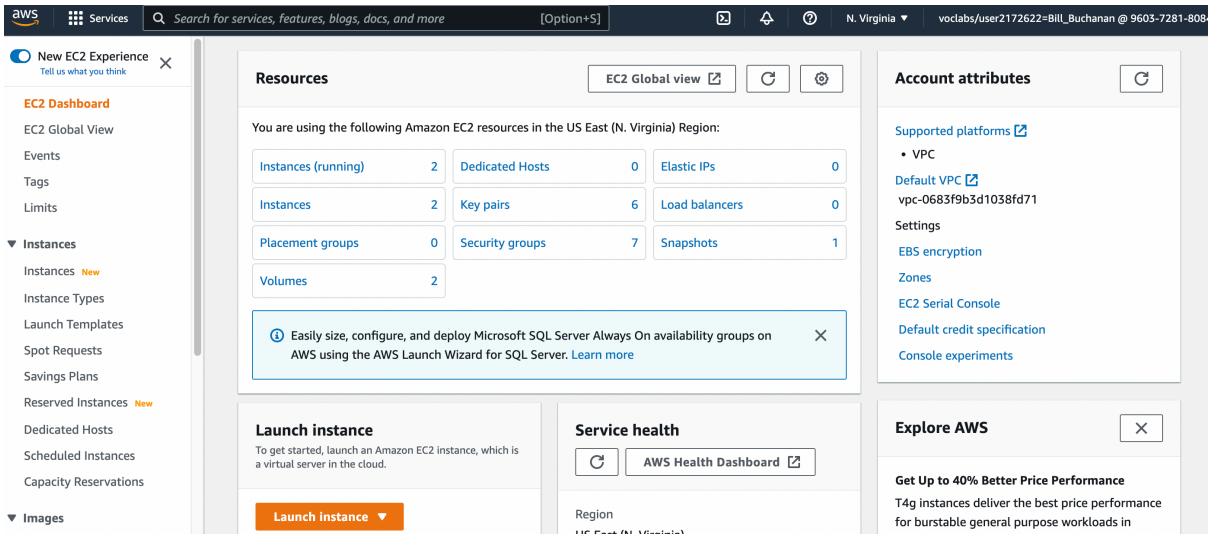

Figure 4: AWS Management Console (EC2)

C Creating and Securing a Linux Server

We will now create a Linux Server, and which should be accessible from the Internet. For this select "Launch Instance", and then give it a name (such as "My Linux Server") and select the Amazon Linux instance for the AMI (Amazon Machine Instance) – as shown in Figure 5.

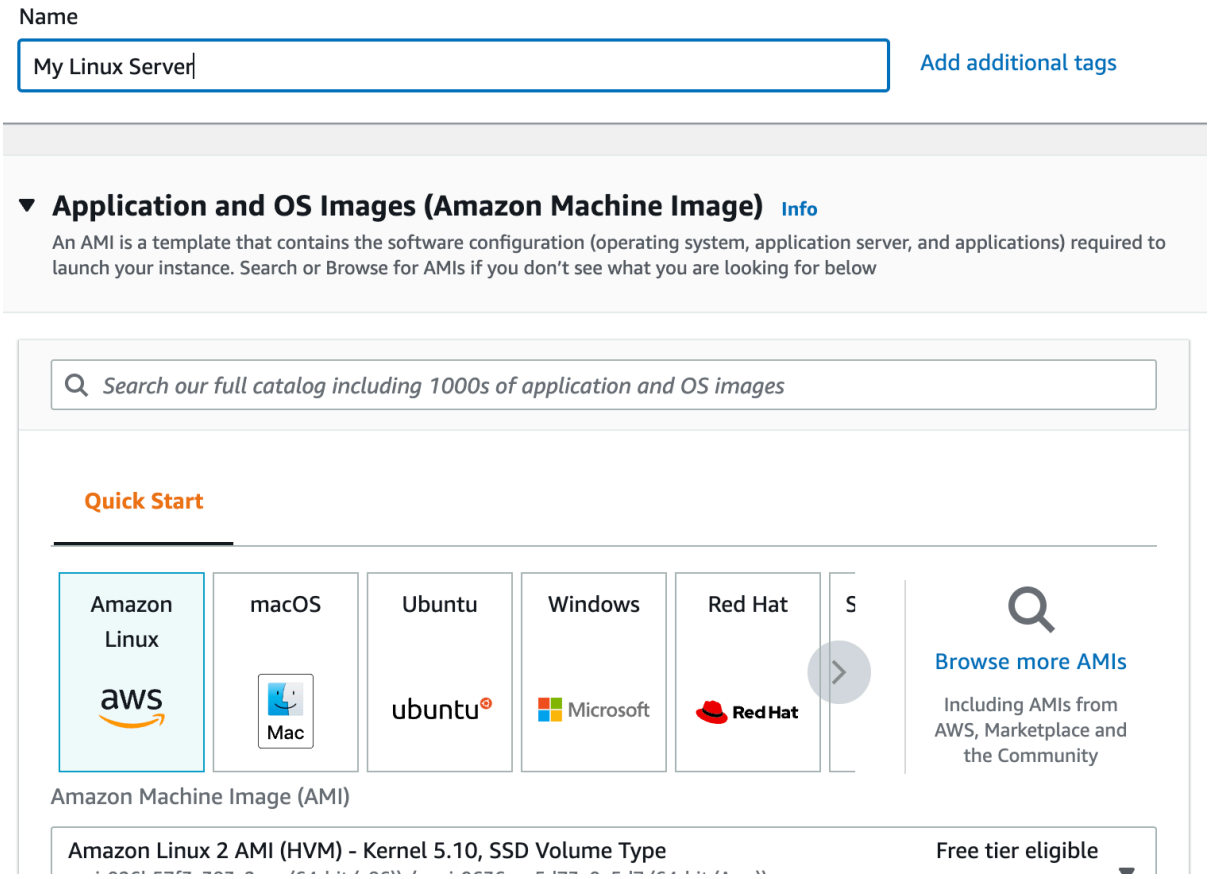

Figure 5: Creating Amazon Linux instance

Now select **t2.micro** for the instance type.

How many vCPUs will the instance have?

How much memory will it have?

How much will it cost per day to run?

If you selected, t2.medium, how much would it cost per day?

If you selected, t2.large, how much would it cost per day?

Now create a new key pair and save it to your local drive. This file contains your private key, and which you will need to connect to your instance. Accept all the other defaults.

Observe the **firewall group** that will be applied.

Which firewall ports are open on the instance?

What do you think is the main issue with this firewall setting?

How would you change it, once you have created the instance?

Observe the disk storage setting for the instance.

What type of disk will be used? [HDD/SSD]

What do you think is the advantage of using SSD?

For disk storage, what is the default size of the disk that you will create?

What is the maximum storage size for a free tier storage of the AMI instance we are creating?

C.1 Creating the instance

Go ahead and create the instance. Then go back to the AWS Management Console, and find your instance. Wait for it to set its state to running.

Now we will connect to it. For this we need to create an SSH connection and use the private key we have generated. The public key will be stored on the instance and will authenticate our access. We do not need a username or password to access the instance, as this is often insecure. Our PEM file will give us access (or you can use Putty for the connection).

Now, we will examine the details of our instance (Figure 6). On the instance summary, determine the following:

The public IP address:

The private IP address:

The instance type:

The public IPv4 DNS:

From your local host, can you ping the public IP address? [Yes/No]

Why can't you successfully ping your instance?

Which region of the world is your instance running in?

C.2 Enabling ICMP on firewall

Now, we will enable ICMP on the instance. First click on the Security tab of the instance summary, and then on the security group.

What is the firewall rule that is applied to the instance?

[SSH/Telnet/FTP/HTTP/HTTPs] for [0.0.0.0/0 or 0.0.0.0/8 or 0.0.0.0/16 or 0.0.0.0/32]

What does 0.0.0.0/0 represent?

Now go ahead and add an ICMP rule for all hosts (Figure 7).

Can you now successfully ping your instance? [Yes/No]

Now, lock your ICMP rule down to just your IP address (you need to use a /32 address for this). Can you still successfully ping the instance? [Yes/No]

Ask you neighhour or one of the lab tutors to ping your instance. Can they successfully ping it? [Yes/No]

What is the advantage of applying the firewall in AWS, rather than in the instance?

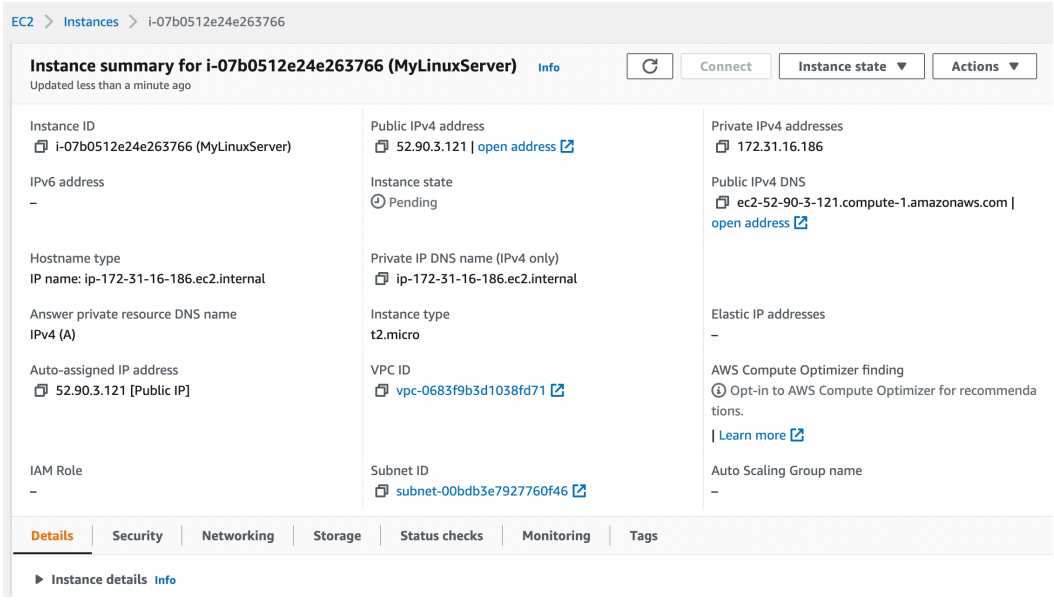

Figure 6: Details of instance

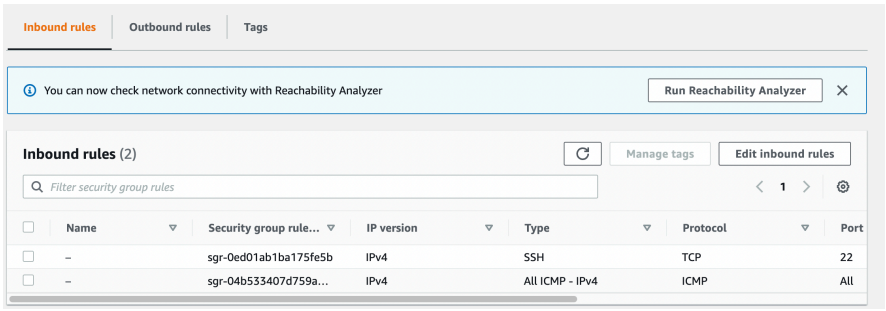

Figure 7: Enable ICMP

C.3 Accessing your instance

Now we will connect to our instance. For this you need SSH (such as provided by OpenSSH). This may be installed on the host you are using (such as in vSoC 2), or from Apps Anywhere. Once you have SSH, press **Connect** on the summary page, and you should then have tabs for **Connect to instance** (Figure 8). Next select the SSH client tab, and you will see the details of connecting to your instance with SSH.

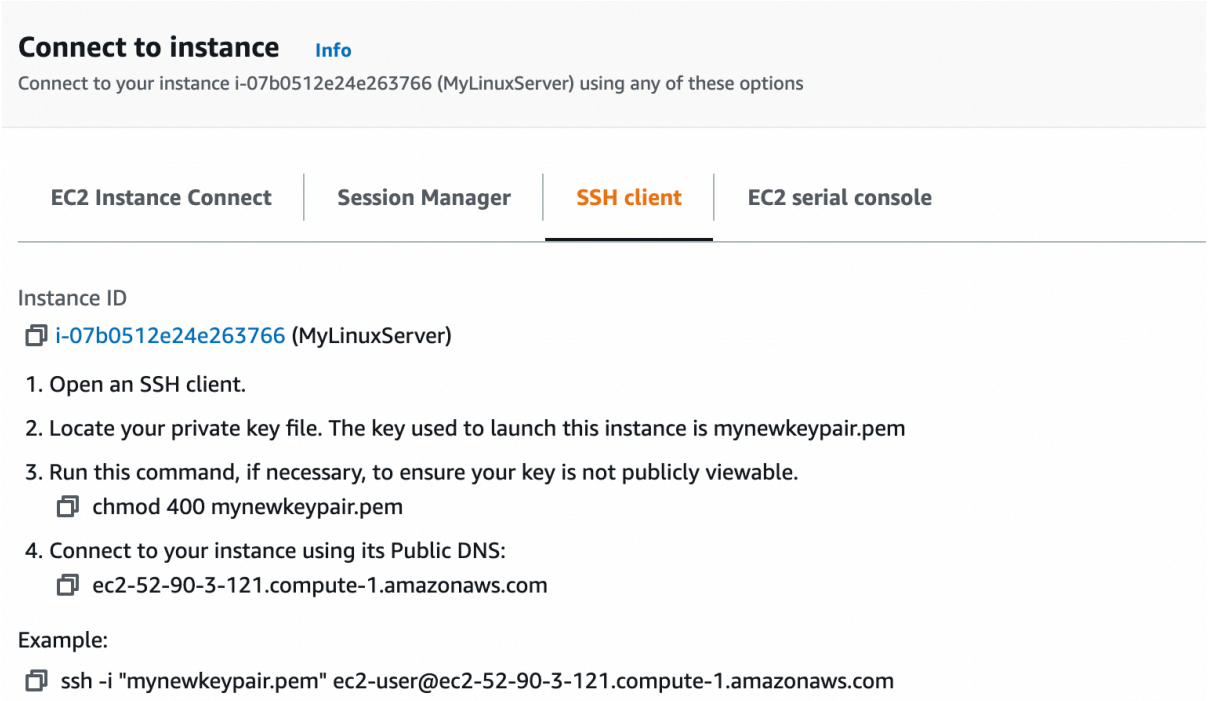

Figure 8: Connect to instance

Now find your PEM file on your local machine (from the command line), and protect it with:

```
chmod 400 myfile.pem
```

What protection does this put on your private key?

Next, use the SSH connection with the name of your PEM file and with the DNS (or IP address) for your instance. For example, in the case in Figure 8, we have:

```
ssh -i "mynewkeypair.pem" ec2-user@ec2-52-90-3-121.compute-1.amazonaws.com
```

What is the name of the user that logs in?

An example of connecting is:

```
% ssh -i "mynewkeypair.pem" ec2-user@ec2-52-90-3-121.compute-1.amazonaws.com
The authenticity of host 'ec2-52-90-3-121.compute-1.amazonaws.com (52.90.3.121)'
can't be established.
ED25519 key fingerprint is SHA256:/c5UOK6gprKL19XCptNQ1brb9MpYR5wEeqhD/6t+/Wk.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:48: ec2-3-90-189-201.compute-1.amazonaws.com
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-52-90-3-121.compute-1.amazonaws.com' (ED25519) to
the list of known hosts.
```

```
Last login: Fri Sep 30 17:07:00 2022 from ec2-18-206-107-27.compute-1.amazonaws.com

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-16-186 ~]$
```

Have you managed to connect? [Yes/No]

By using “ip addr show” or “ifconfig” in your instance, what is the private IP address of it?

Can you ping 8.8.8.8 from your instance? [Yes/No]

Is there a folder named .ssh? [Yes/No]

What do you think is the purpose of the file contained in .ssh?

Now create a folder in the top level named “mytestfolder”, and put a new file in there named “mytext.txt” (and put some text in this file).

Now go to the EC2 Instance Connect (Figure 8), and press on the Connect button. You should now get a console terminal in the browser.

From your console (Figure 9), verify that your file has been created. Has it been created in the instance? [Yes/No]

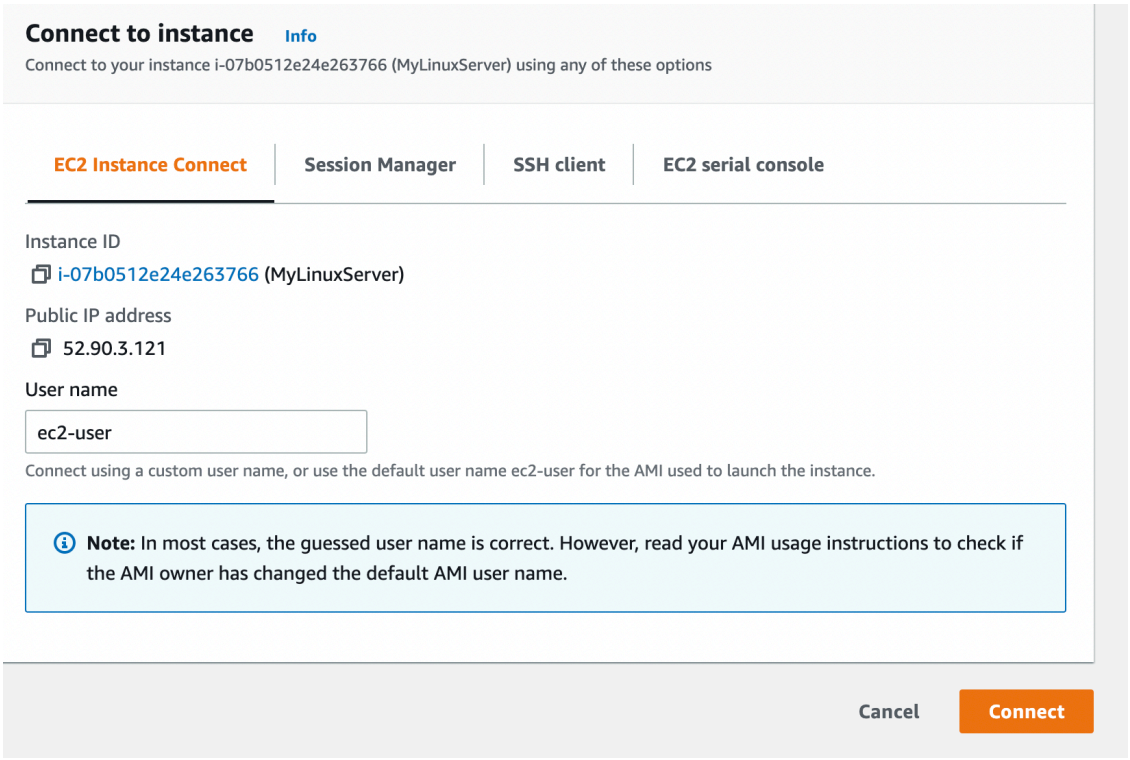

Figure 9: EC2 Instance Connect

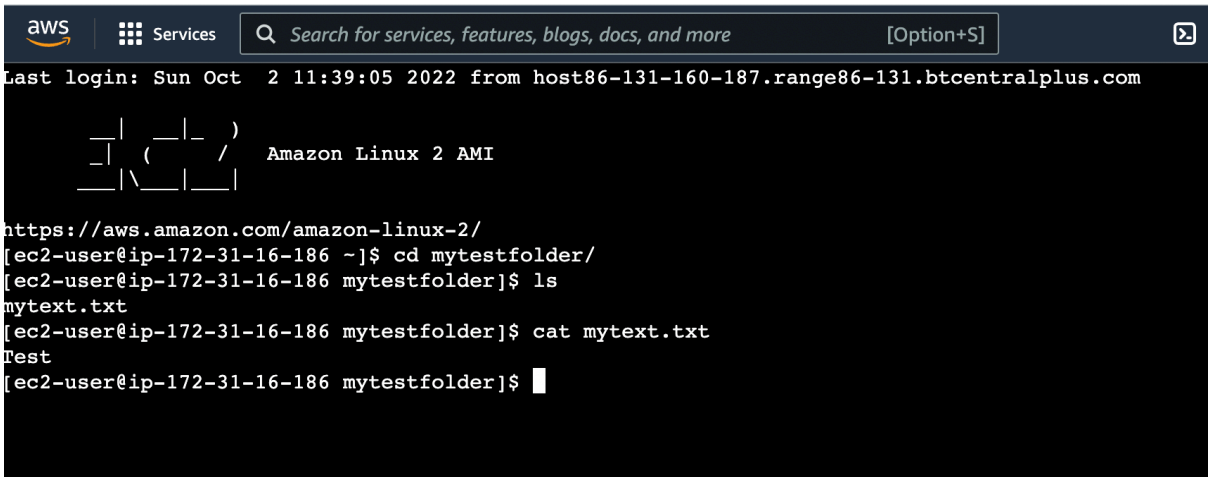

Figure 10: EC2 Instance Connect terminal

Now examine the running services on the instance with:

```
$ netstat -i | grep tcp
$ netstat -i | grep udp
```

Which of the main services are running:

C.4 Installing a Web server

Now we will install a Web server on the instance with:

```
sudo yum update -y
sudo yum install -y httpd.x86_64
sudo systemctl start httpd.service
sudo systemctl enable httpd.service
```

Next open up a browser on your computer and access your instance for Web access.

Can you connect to it? [Yes/No]

Why can't you connect to it?

Now enable a firewall rule on Port 80 and Port 443 and allow access for Web traffic (see Figure 10).

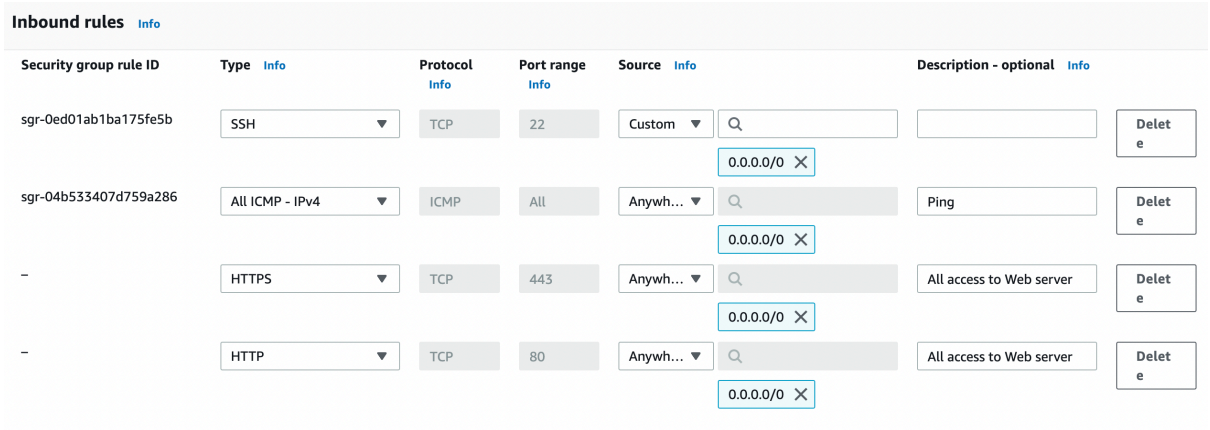

Figure 11: Enable HTTP and HTTPs rules

Can you now connect to your Web site? [Yes/No] (see Figure 11)

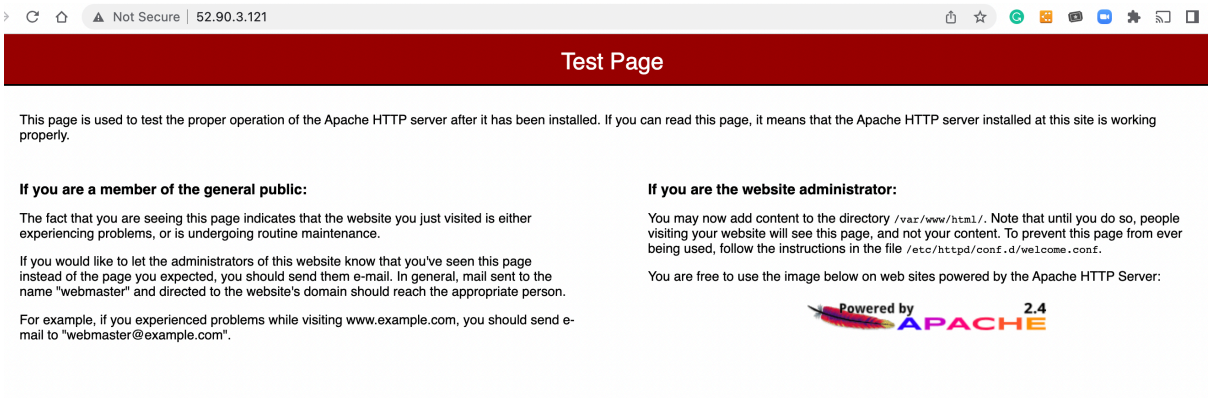

Figure 12: Sample access to Web site

Now go into the /var/www/html folder, and create a file named “index.html”, and add:

```
<h1>Main Web site</h1>
<p>Hello to you</p>
```

And then save the file.

Has it changed the welcome? [Yes/No]

C.6 Auditing

The main logging output is in the /var/log folder. Go into this folder and observe some of the files in there. Identify the contents of the following files:

What are the likely contents of the “secure” file? What are the likely contents of the “boot.log” file? List the log/httpd/access_log file. What are its contents? Can you identity your browser access? (see Figure 12). Which browser type accessed your Web server?

Now try with another browser type (such as Firefox or Chrome) and re-examine the log/httpd/access_log file. Did it detect the new browser type?

Now access a file that does not exist in your site (such as http://AWSIP/test.htm). Now re-examine the log/httpd/access_log file. What is the status code returned for the access?

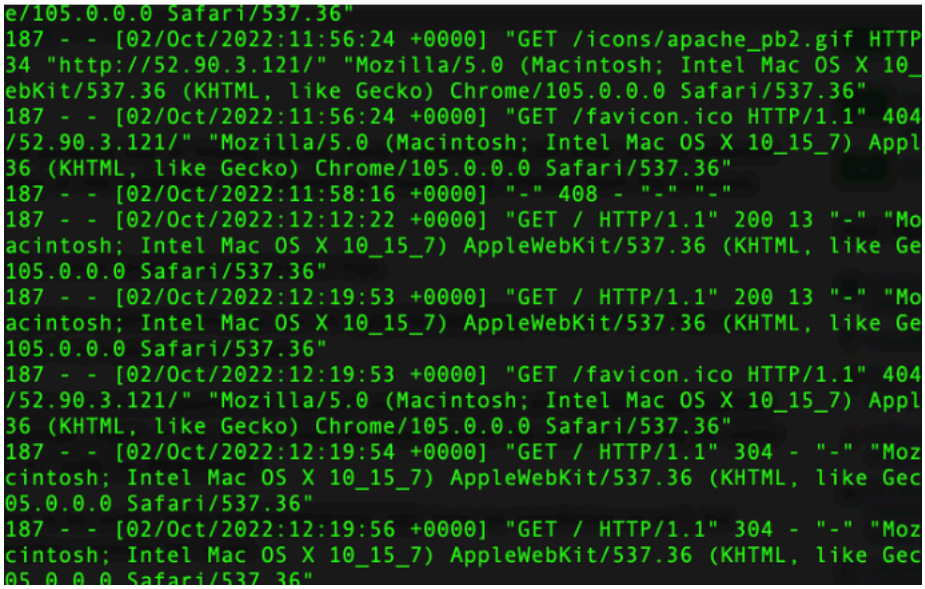

Figure 13: Sample list of log/httpd/access_log

C.7 Adding a new user

The ec2_user can be used to connect back into the server using access authenticated with the private key. We will now create a new user named "napier", and which can connect to the instance with SSH. For this we use adduser and passwd on the Linux instance:

```
[ec2-user@ip-172-31-16-186 ~]$ sudo adduser napier
[ec2-user@ip-172-31-16-186 ~]$ sudo passwd napier
Changing password for user napier.
New password:  <yourpass>
Retype new password:  <yourpass>
passwd: all authentication tokens updated successfully.
```

Now we will add the new user to the login. For this, we use:

```
[ec2-user@ip-172-31-16-186 .ssh]$ sudo nano /etc/ssh/sshd_config
Add line of (see Figure 13):
AllowUsers ec2-user napier
Change the following to "yes" (see Figure 14):
PasswordAuthentication yes
```

Now restart the SSH service with:

[ec2-user@ip-172-31-16-186 .ssh]$ sudo systemctl restart sshd

Can you now connect to your instance with the new user and password:

```
ssh   napier@54.209.145.85
```

Can you connect with the new user? [Yes/No]

```
#       $OpenBSD: sshd_config,v 1.100 2016/08/15 12:32:04 naddy Exp $
AllowUsers ec2-user napier
# This is the sshd server system-wide configuration file.  See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/usr/bin

# The strategy used for options in the default sshd_config shipped with
```

Figure 14: Accessing instances

```
# To disable tunneled clear text passwords, change to no here!
#PasswordAuthentication yes
#PermitEmptyPasswords no
PasswordAuthentication yes

# Change to no to disable s/key passwords
#ChallengeResponseAuthentication yes
ChallengeResponseAuthentication no
```

Figure 15: Accessing instances

C.8 Accessing from AWS prompt

We can also access our instance from the AWS terminal prompt. For this return to your AWS Academy console, and enter the command (Figure 15):

```
$ aws ec2 describe-instances
```

From the results, can you identify the following.

Instance type:

Public IP address:

Private IP address:

State:

Figure 16: Accessing instances

Now try we will stop our instance using an AWS EC2 command. Run the following with your instance ID (see Figure 17):

`aws ec2 stop-instances --instance-ids [My-instance-ID]`

From the AWS Management Console, has your instance stopped? [Yes/No]

```
ddd_v1_w_W3n_1455598@runweb62964:~$ aws ec2 stop-instances --instance-ids i-07b0512e24e263766
{
    "StoppingInstances": [
        {
            "CurrentState": {
                "Code": 64,
                "Name": "stopping"
            },
            "InstanceId": "i-07b0512e24e263766",
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
ddd_v1_w_W3n_1455598@runweb62964:~$
```

Figure 17: Stopping an instance

Now we will restart the instance, with:

`aws ec2 start-instances --instance-ids [My-instance-ID]`

Has the instance re-started? [Yes/No]

Now we will change the instance type from t3.micro to t3.small. To do this, run the following commands:

```
aws ec2 stop-instances --instance-ids [My-instance-ID]
aws ec2 wait instance-stopped --instance-ids [My-instance-ID]
aws ec2 modify-instance-attribute --instance-id [My-instance-ID] --instance-
type "{\"Value\": \"t3.small\"}"
aws ec2 start-instances --instance-ids [My-instance-ID]
```

Did it change the instance type? [Yes/No]

Can you still get access to your instance?

By observing the script, and investigate what t3.micro and t3.small are, can you determine what has changed about your instance?

Now, revert the instance back to t3.micro, and suspend the instance.

D Python Access

Your unique account will have been generated, and you can access it with **aws_access_key_id** and **aws_secret_access_key** (from AWS details). You will also find that your console has been setup with the details already setup for you. For this, there is a hidden folder named **.aws**, and there is a file named **credentials** in there:

```
ddd_v1_w_W3n_1455598@runweb63277:~$ ls -al
drwxrwx--- 5 ddd_v1_w_W3n_1455598 apache              6144 Oct  2 10:13 .
drwxrwx--- 5 ddd_v1_w_W3n_1455598 apache              6144 Sep 29 10:32 ..
dr-xr-xr-x 2 ddd_v1_w_W3n_1455598 apache              6144 Sep 29 12:08 .aws
-rw-rw-r-- 1 ddd_v1_w_W3n_1455598 apache                51 Sep 29 10:32 .gitconfig
drwxr-x--- 2 ddd_v1_w_W3n_1455598 apache              6144 Sep 29 12:08 .ssh
-rw-r--r-- 1 root                 root                3851 Oct  4 02:44 .termrc
dr-xr-xr-x 2 root                 root                6144 Sep 29 10:32 .voc
ddd_v1_w_W3n_1455598@runweb63277:~$ cd .aws
ddd_v1_w_W3n_1455598@runweb63277:~/.aws$ ls -al
dr-xr-xr-x 2 ddd_v1_w_W3n_1455598 apache 6144 Sep 29 12:08 .
drwxrwx--- 5 ddd_v1_w_W3n_1455598 apache 6144 Oct  2 10:13 ..
-r--r--r-- 1 ddd_v1_w_W3n_1455598 apache   29 Oct  4 00:19 config
-r--r--r-- 1 ddd_v1_w_W3n_1455598 apache  501 Oct  4 00:19 credentials
ddd_v1_w_W3n_1455598@runweb63277:~/.aws$ cat credentials
```

List the contents of the credentials file, and verify that it contains the same credentials as from the AWS details button.

Are they the same? [Yes/No]

Now create a Python file which will show your instances in the terminal window (such as 1.py):

```
import boto3
ec2 = boto3.client('ec2', region_name='us-east-1')
ec2.describe_instances()
```

Figure 28: Python file creation

Save the file, and then run the file with Python3 and prove that it shows your instances (see Figure 29).

Figure 29: Running the Python3 file

Does the Python3 program show your instances? [Yes/No]

Now we will stop one of our instances. For this, get an instance name, and add it to the following file:

```
import boto3
ec2 = boto3.client('ec2', region_name='us-east-1')
```

```
ec2.stop_instances(InstanceIds=["i-07b0512e24xxxxxx"])
```

Now run the Python file, and prove that it has stopped your instance.

Does the Python3 program stop your instance? [Yes/No]

Now we will restart one of our instances. For this, get an instance name, and add it to the following file:

```
import boto3
ec2 = boto3.client('ec2', region_name='us-east-1')
ec2.start_instances(InstanceIds=["i-07b0512e24xxxxxx"])
```

Now run the Python file, and prove that it has stopped your instance.

Does the Python3 program start your instance? [Yes/No]

Finally, write a Python3 program which will start both of your instances, and another one to stop them both.

Do your Python3 programs work? [Yes/No]

You can also use the AWS prompt. Now try to start and stop your instances with:

```
aws ec2 stop-instances --instance-ids i-07b0512e24xxxxxx
```

and

```
aws ec2 start-instances --instance-ids i-07b0512e24xxxxxx
```

Do these command line programs work? [Yes/No]

Now we will create a keypair with Python, and then create a new Linux instance. First create the keypair with the Python file of:

```
import boto3
ec2 = boto3.client('ec2', region_name='us-east-1')
outfile = open('mykeypair.pem','w')

key_pair = ec2.create_key_pair(KeyName='mykeypair2')
MyKeyPair = key_pair["KeyMaterial"]

print(MyKeyPair)
```

What is the name of your key pair? Can you find it in your AWS Management console? [Yes/No]

Now we will create a Linux instance. Take a note of the AMI for your Linux instance, and check that it is the same as the instance below Now create create.py, and save the file:

```
import boto3
```

```
ec2 = boto3.resource('ec2')

# create a new EC2 instance
instances = ec2.create_instances(
     ImageId='ami-026b57f3c383c2eec',
     MinCount=1,
     MaxCount=2,
     InstanceType='t2.micro',
     KeyName='mykeypair2'
 )
```

Finally run the instance. Has it created the instance? [Yes/No]

If it has created it, now terminate it. Has it been terminated? [Yes/No]

```
ddd_v1_w_W3n_1455598@runweb63485:~$ python3 keypair.py
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEA11fo8IBjSKYxVBFsrvoLKOO8z+JrhFgL9dvJSuyxlxnP5HtS
SQawOKN/OKONGiu0rOp8BQNwkhTcwem1jOh6vss5vmyaXq4uJe3OiCACoKfHtsjW
OMgxbWyAaT2KkFF3B7sdIOK4tBHUFSU/STzn4IpU6XktAxC/8TuUIx7sh8RAKsGR
6qxWyeUh8jVsP0Q+5HPtx+3ObyZGtGDeiqhYxW4iRBkugG2ukX7KsLnEpNdv0mS8
w4wTMObB0Pgwnun7MNF+gM+6/VtnraJia19amTMhvtk4F6bG6qgyH3DXQmm3VP64
MvJQR3QigMtq9WKYP3deASN7LOzKJzVgXHso1QIDAQABAoIBAEPl/BSY5rUpcbRI
KjShspMZ+5awlaYM/PhwbG+rYfN6uAoauD8+VKIHKz34k60EytsatDAAfaYR3elm
f1AGnW07NuHL7ICd+v8KlGxna2yselQvzxvoQ211IKgON3p5tUZCwVR1KDfexzhD
nkjBY3txIoTbKE4nId42IudQANRtgJ2GN+iw+T6UgQ6dchRw7Z35V+txw1lpSEQk
V68ofai3MD8+Nj1bJrKjDJSIb4ScaV/KtIq53eXX8bHKi/3CL78mjVC0kPb5Ir8Z
7YKo02Xw9W4jOghjHsh22bmLsbEJnuICci8Z5h+4Im+3lSVwQFdjq7pTsCQkNGYQ
QSV7yaECgYEA6uwTvBWDNUTH6cf18SRZ4zplRKDWSuufAvhiPH3oq96o5NTILxXM
UDnFmW8x43MH4aG5LjLtaosnwAIDAgFuLwbQDZPsbNaHy8+tlNxW2iAldc/5v3vQ
qMGCoy+ak3/boye7X16WuYbXWDXxYAKaYo+hxWqp9MOdNw/FmPHmnU8CgYEA6qog
1YyAnuYrmEM9/FIVDH1FyrEIEUzck79oCcYuONRGqkbMt7dS7DIjpVGGPmTnEcFr
LlyCAiWfyxQVa5BTGCogNXgDsYPD7aEk6b+Etc8gKdA9bMUZ60Xq9R0pegNuRABy
93wZhGWmxghS/st3wqAGtzzJrJIXE0+sWD5Q9psCgYBtst2oR3dkiYC8tyYue6YJ
yHv/ZweKJqqzDV96FeVvVGMTsBTmf8xPbhrKgjUOm+mDQCyj2R4qUV63d3kGF7rm
9Mx0eq29SRRSZOEOkTFxW++CKyLZeqy+ENa68wDAfcUHVAdYCmhhrAik5VOYXL7Q
5XdKI3vHkwy36IKB3/urWwKBgQCu/VTkyhRz6cIilioYCYGmwZgU0TTp6b4m8zn6
5+059ZoIiilI6o0Mk0sfkY6g4o+pHun1NYjBXxsszfo3LvW39Fn2ZVujuVWW0276
XBXzmdQrOh1ECisM1iOiyy9NzFcioI8a3FEO+NqHnFd/p2zUFo9LfyoI5RwOC4DC
gR3Q9wKBgQChAlNSOmuZ8sSyJqTr2XxwJ5wI5RDHWQHhJ5maDJQHASYDWcRcD841
c+f817A9G3E3CSytM1BxLudt7INe9pFXf+FSooUtVqGiAPAtAhS7iGnb9Vxhxx27
47hh8xhtwUqSf6/yaY7WCKuGEUAh3iIJpiyt7Me71aUTox/+jelKgg==
-----END RSA PRIVATE KEY-----
ddd_v1_w_W3n_1455598@runweb63485:~$ python create.py
```

Figure 30: Creating an instance

NOW TERMINATE YOUR NEWLY CREATED INSTANCE (and any others you have created with Python)!

At the end of the lab, you should only have two instances. Please either terminate these, or stop them.