TLEN 5410 – Network Management and Automation

Lab 2

Amazon Web Services (AWS)

University of Colorado Boulder

Interdisciplinary Telecom Program

Professor Levi Perigo, Ph.D.

# Summary

Amazon Web Services (AWS) is a cloud-computing platform offered by Amazon. Cloud technology and tools are popular for network engineering, and the skills learned in this lab will enhance students’ resumes in a desirable skillset in the current market. The AWS tools and skills used throughout this lab include EC2, security policies, S3 buckets, SNS, and Cloud Watch. Although these services could be managed using the AWS Management Console, AWS also offers a powerful tool (Software Development Kit) known as Boto3. This gives the power to manage the above mentioned services for a large number of resources using simple Python scripts integrated with the Boto module.

# Objectives

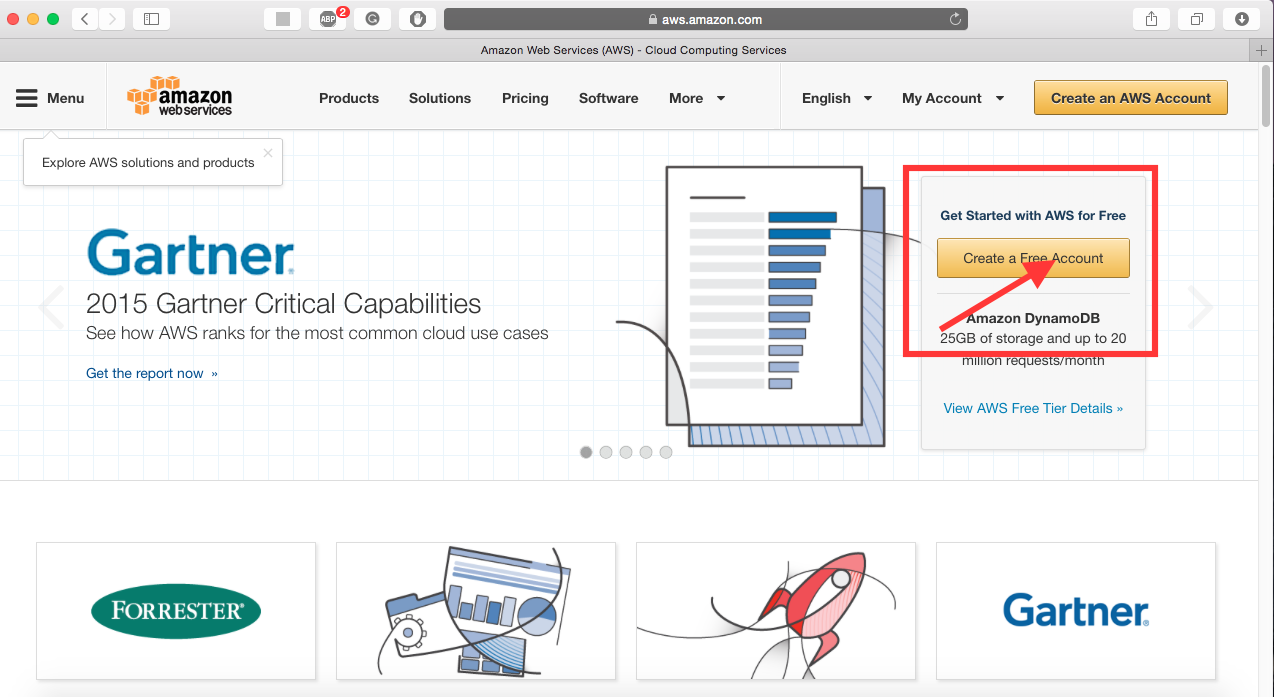
1. Learn about Amazon Web Services (AWS) tools
2. Lean how to deploy EC2 server instances
3. Learn how to deploy applications on those instances
4. Learn how to create security policies/firewall rules
5. Learn how to backup configurations into S3 buckets
6. Learn how to setup and use Simple Notification Service (SNS)
7. Learn how to setup Cloud Watch monitoring system
8. Manage AWS resources using Boto3 (Python based SDK)

# Part 1

# Objective 1.1 – Create an AWS account

In order to use Amazon Web Services, you must first create a user account. Amazon provides one year of free usage (limited resources).

1. Click on [**https://aws.amazon.com**](https://aws.amazon.com). Create a free account and enter all the relevant information. If you already have an Amazon account, sign-in using your credentials.



[**NOTE:** You might need to provide your debit/credit card number for future billing]

1. After creating the account, you will be directed to AWS management console which presents the user with a number of AWS tools and services. Provide a screenshot of your management console. [**1 point**]



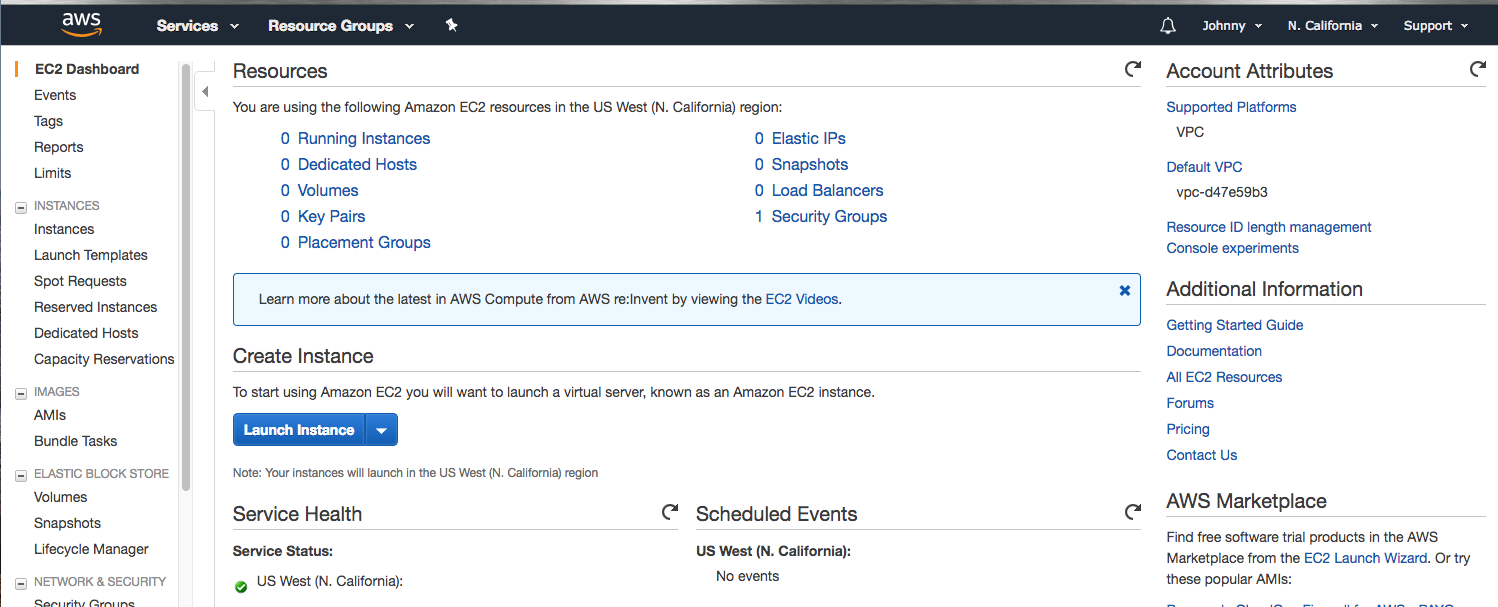
Objective 1.2 – Tweaking AWS settings

We shall now look at tweaking the AWS management settings to make efficient use of these resources.

1. We do not want Amazon to charge us for either using their premium services (Multi-Core CPU’s, Terabytes of data, CDN services, etc.) or breaching their ONE-year free usage tier.

**To prevent extra charges on your account please shut down and remove all instances at the end of the lab.**

To ensure you get the most efficient performance, we need to change our region to any of the US West regions (N. California/Oregon). You can do this from the AWS management console. Provide a screenshot of the region you selected [**1 point**].



1. Set up security credentials for your account: <https://aws.amazon.com/premiumsupport/knowledge-center/create-access-key/> (Please ensure that you download and save the keys on your machine, as it will be required for further objectives).

# Part 2

# Objective 2.1 – Deploying EC2 Instances

1. What is EC2? [**5 points**]

Amazon Elastic Compute Cloud(EC2) is a web service that provides secure, resizable compute capacity in the cloud. It’s designed to make web-scale could computing easier for developers.

1. Briefly explain the below types of EC2 instances. [**15 points**]

* Spot instances

Spot instance is an unused EC2 instance that is available for less than the On-Demand price. If you are flexible about when your applications run and your applications can be interrupted. It will fit for data-analysis, batch jobs, background processing, and optional tasks

* Reserved instances

Reserved instances are not physical instances, but rather a billing discount applied to the use of On-Demand instances in your account.

* Dedicated hosts

This is a physical server with EC2 instance with full capacity dedicated to your user’s use. This allow your existing per-socket, per-core, or per-VM software licenses such as Windows Server, Microsoft SQL Server, and Linux. [8]

* Dedicated instances

This type of instances are Amazon EC2 instances running in a VPC on hardware to a single customer dedicatedly. These dedicated instances may share hardware with other instances from the same AWS accounts that are not dedicated instances. [10]

* Elastic GPUs

This type of services allow you to attach low-cost graphic acceleration to a wide range of EC2 instances over the network. User can choose the right amount of compute, memory, and storage for their application. [9]

To deploy an EC2 instance, click on Services> EC2 > Launch Instance.

**Step 1:** Search for ubuntu and select Ubuntu Server 16.04 LTS 64-bit (x86) image.

[Or you can select an image of your choice.]  
**Step 2:** t2.micro (Free tier eligible)

**NOTE:** Selecting anything else will result in OS costs.

**Step 3:** Deploy **TWO** instances.

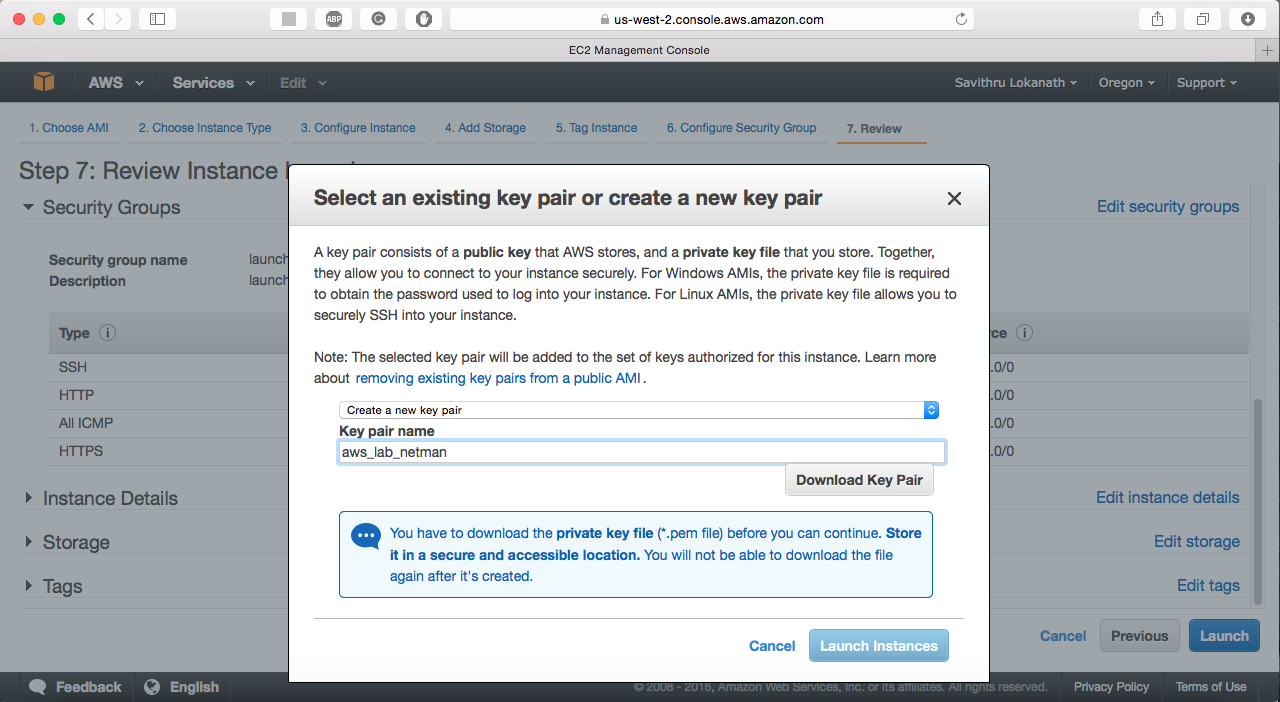
**Step 4:** Add storage (8 GB is fine, can add more later)

**Step 5:** Add name as NetMan-Lab2

This uniquely identifies the instance and will come handy in a clustered environment. You can define something like a **“webserver”** if you are deploying a webserver or **“appserver”** for an application server, etc.

**Step 6:** Create a new security group and select the source to be your IP address for SSH rule. Add security rules allowing ICMP, HTTP and HTTPS traffic from your IP address.

**Step 7:** Review the configuration parameters before launching the instance. Create a new key-pair and click on download. [**NOTE:** If you miss this step, you will have to repeat the whole process again.] After the download completes, click on the **“Launch”** button.



1. Explain how an AMI is related to the instance. [**3 points**]

Amazon Machine Image (AMI) offers information to launch an instance that is a virtual server in the cloud. It must specify a source AMI when user launch one or more instances[1]. AMI provides a template for the root volumn (operating system, applications, server…etc), and permissions to control AWS accounts that user the AMI launching instances. Linux AMI has some features like the followings:

* A stable, secure, and high-performance execution environment for applications running on Amazon EC2. [1]
* Provided at no additional charge to Amazon EC2 users. [1]
* Repository access to multiple versions of MySQL, PostgreSQL, Python, Ruby, Tomcat, and many more common packages. [1]
* Updated on a regular basis to include the latest components, and these updates are also made available in the **yum** repositories for installation on running instances.[1]
* Includes packages that enable easy integration with AWS services, such as the AWS CLI, Amazon EC2 API and AMI tools, the Boto library for Python, and the Elastic Load Balancing tools.[1]

1. What are the disadvantages of allowing SSH traffic into the server from anywhere? Is this a security concern? [**5 points**]

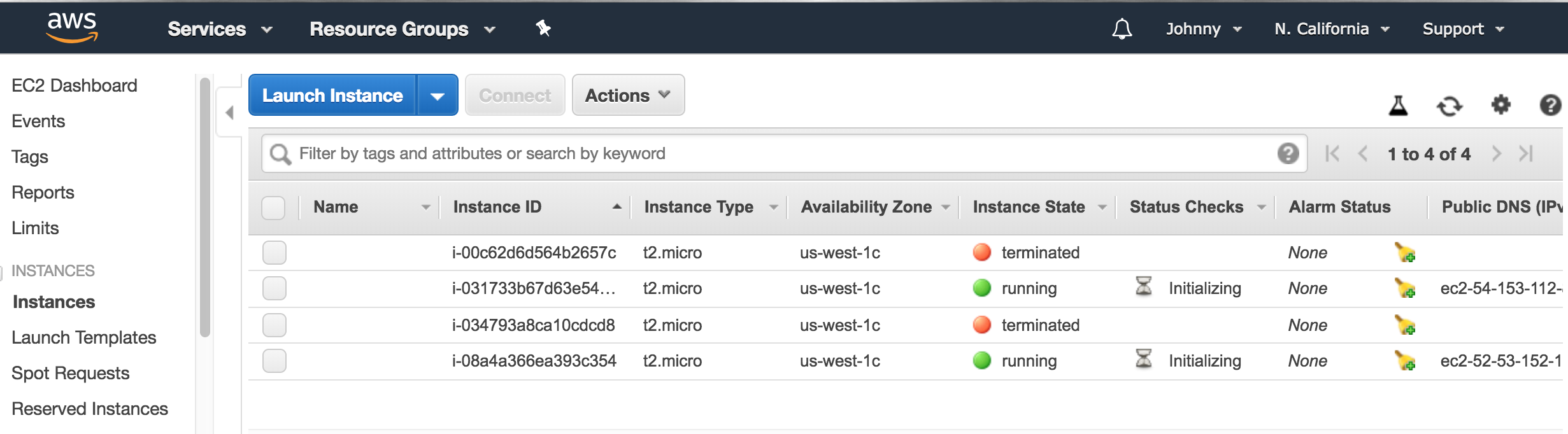
Yes, it is. Because when we add security rules before launching the instances, we specify SSH from anywhere, meaning we enable all IPv4 addresses to access my instance using SSH. Also, when we add the rule of anywhere, it also added the ::/0, it means I enable all IPv6 address to access my instance. It’s fine if I use this for a short time in a test environment, but this is not safe for production environments.

1. What are some best practices one should follow to secure the north-south and east-west traffic in the cloud? [**5 points**]

One of the best security practices is visibility. This is critical for organizations to determine their networks and data centers. It is important for organizations to monitor internal traffic patterns for malware that has infiltrated the network and threats while organizations tend to focus on securing external traffic [2].

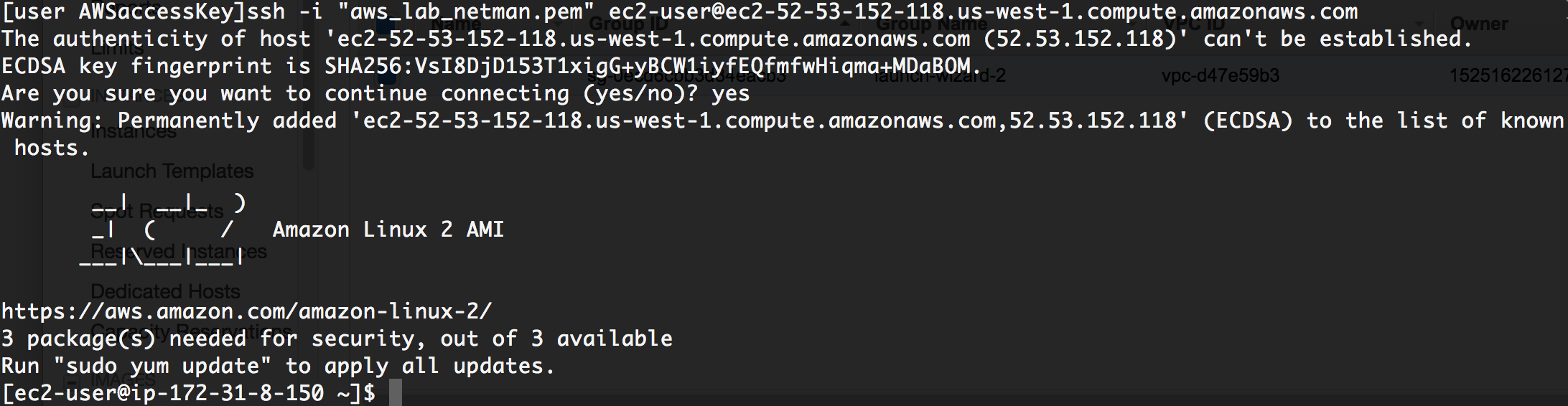
Another best practice is using Software-defined networking (SDN). It provides another level of control and management to traffic. With SDN controller, it could manage edge policies for each port, policies can be moved with a workload. This approach is more responsive and agile to user’s need.

1. Paste a screenshot showing the running instances. [**10 points**]

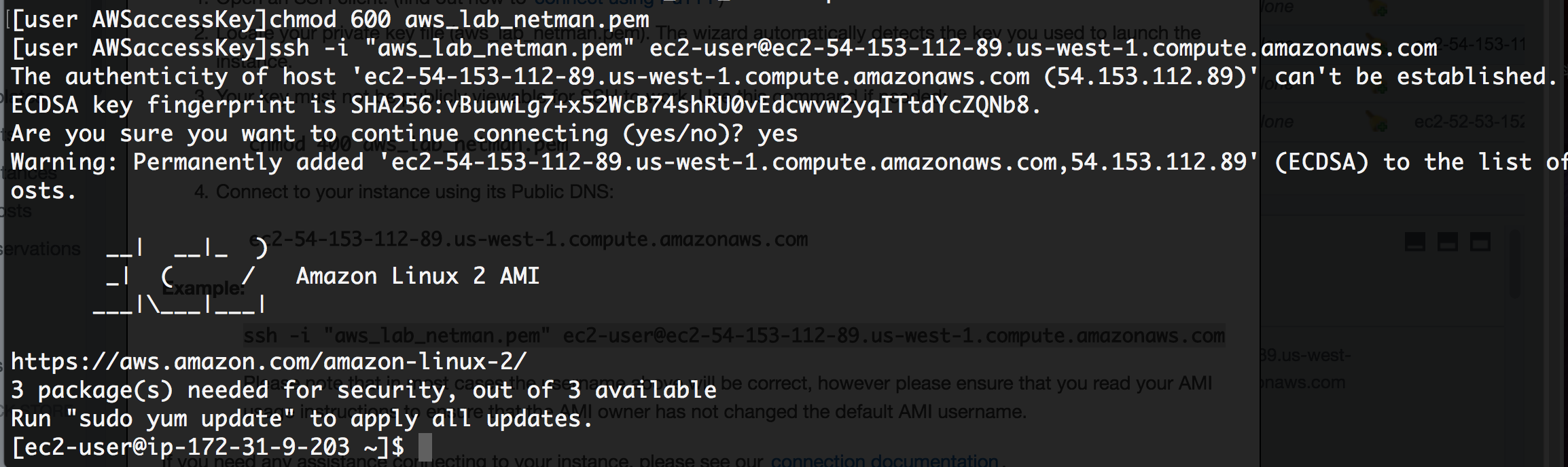


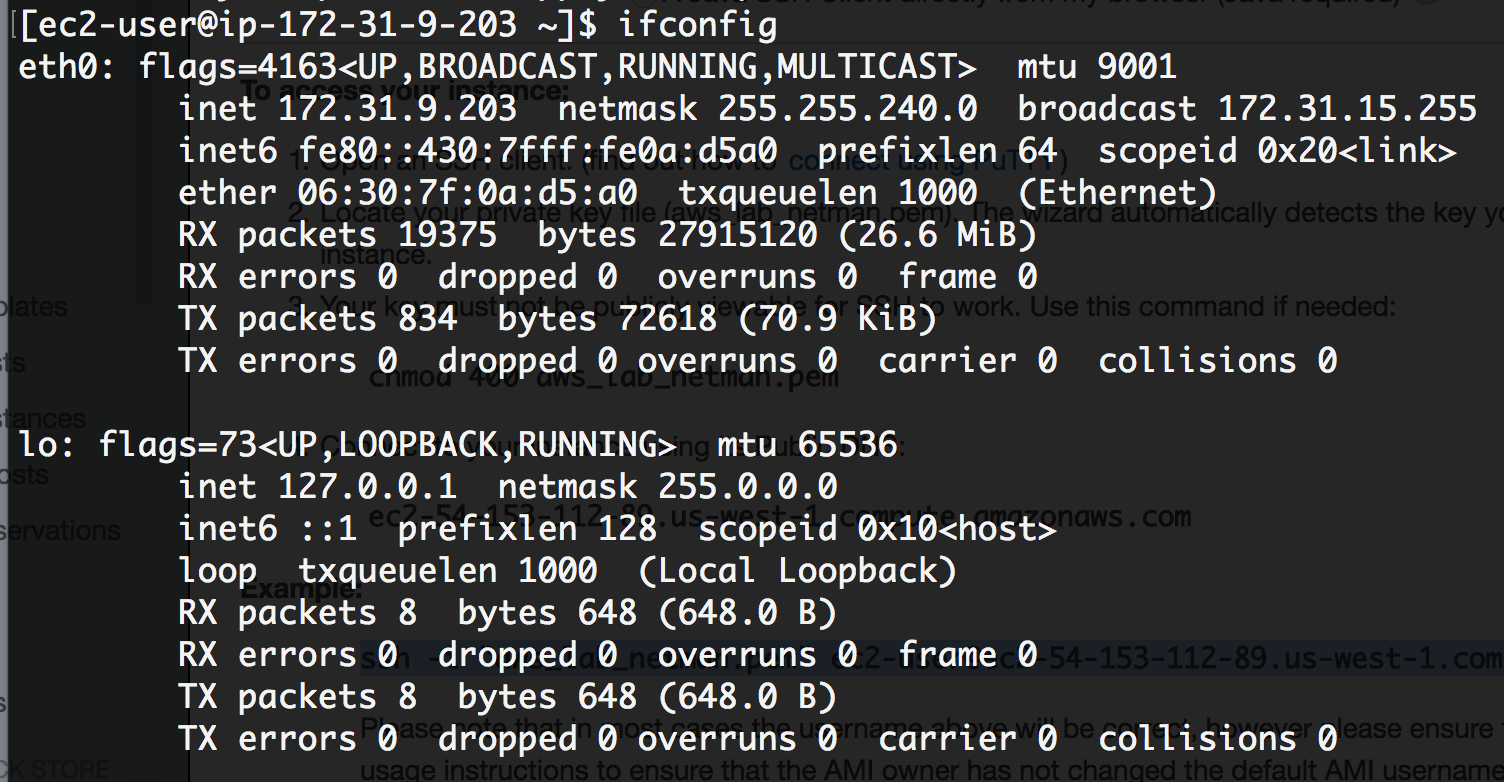
1. Select an instance and click on connect. Follow the instructions and SSH into one of the EC2 instances. Paste a screenshot showing the bash prompt. [Login for the Ubuntu instance is: ubuntu] [**5 points**]

Note: this ip address is different than the next picture because I came back in a different day to do this object and the public ip address has changed.



1. What IP address did you SSH to? And what is the IP address on the interface of the instance? Explain the flow of traffic from your laptop to the EC2 instance. [**5 points**]

The IP address I ssh to is 54.153.112.89. The IP address on the interface of the instance is eth0



I send packet from school, which is private subnet (NAT), and then the packet will transmit to router, the router will route the packet to internet gateway, and then goes to the AWS.

1. Create another user in your instance with the username being your name and enable SSH with password for just your user. Explain how you achieved this. [**3 points**]

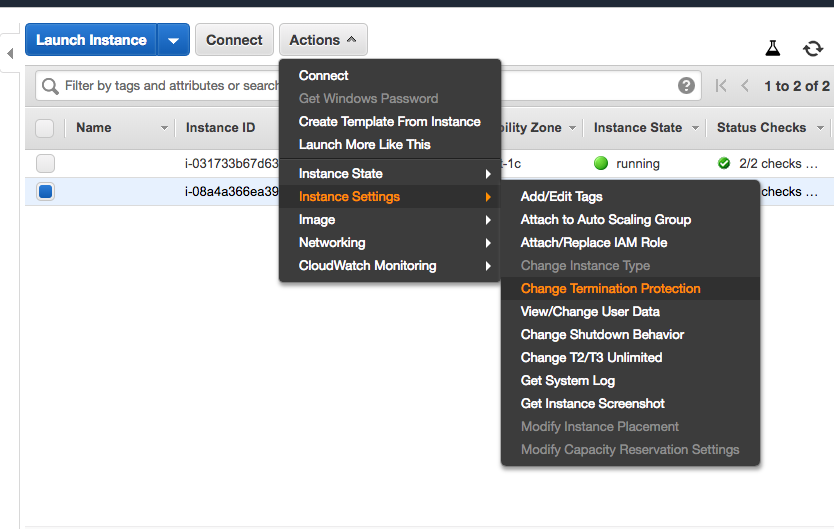
* On ec2 server, create a user by doing ‘sudo useradd chenhao\_cheng’ on ec2 server
* Change the user password by ‘sudo passwd chenhao\_cheng’ [pw: netman123]
* Switch to new user I just created by ‘sudo su – chenhao\_cheng’
* Go to console web page and on the left navigation, select Key Pairs, and select Create Key Pair, and create name of key pair (chenhao\_cheng-keypair), and click create
* Download the key pair pem file and save the location you want
* Go back to ec2 with the account you correctly switching (chenhao\_chen)
* Make a directory by ‘mkdir .ssh’, Chmod 700 .ssh and go inside of .ssh
* Create a file call ‘authorized\_keys’, and make permission 600 for that file
* Go back to the directory of chenhao\_cheng-keypair and set 400 permission
* And in MAC, use the ‘ssh-keygen -y’ command to generate ssh key, when it prompt file, ENTER the absolute path with the new pem file(chenhao\_cheng-keypair)
* Copy the content of public key and paste into authorized\_key in ec2 server
* Done, test with your new user account

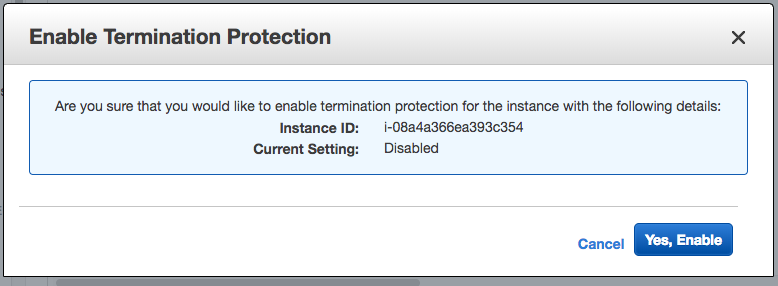
1. What is the difference between stopping and terminating an instance? [**2 points**]

The key difference between stopping and terminating an instance is that the attached bootable EBS volume will be deleted. The volume will stay after stopping while all information on the local hard drive will be lost as usual.

Thus, stop an instance if you plan to start it again within a reasonable timeframe. Otherwise, you could terminate an instance for cost saving purposes. [3]

1. Protect the second instance that you created from unauthorized termination. How did you achieve this? [**2 points**]





Select yes

Objective 2.2 – Deploying an application on your instance

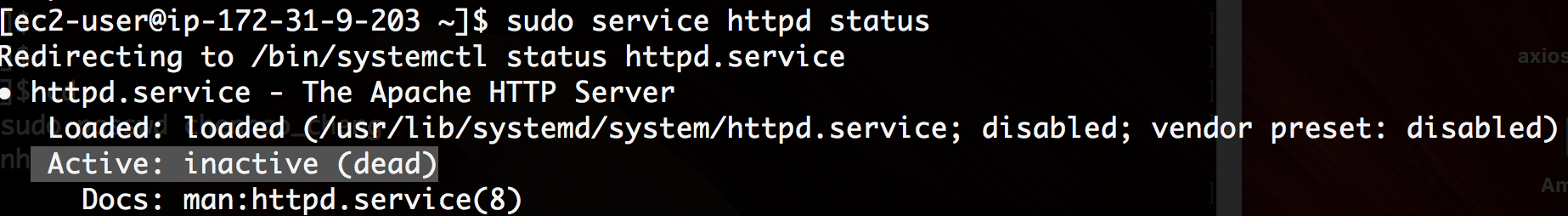
Install Apache web server on one of the instances. Create an index.html file in the /var/www/html/ directory which displays your name and ‘NetMan Lab2’. You can use the below sample.

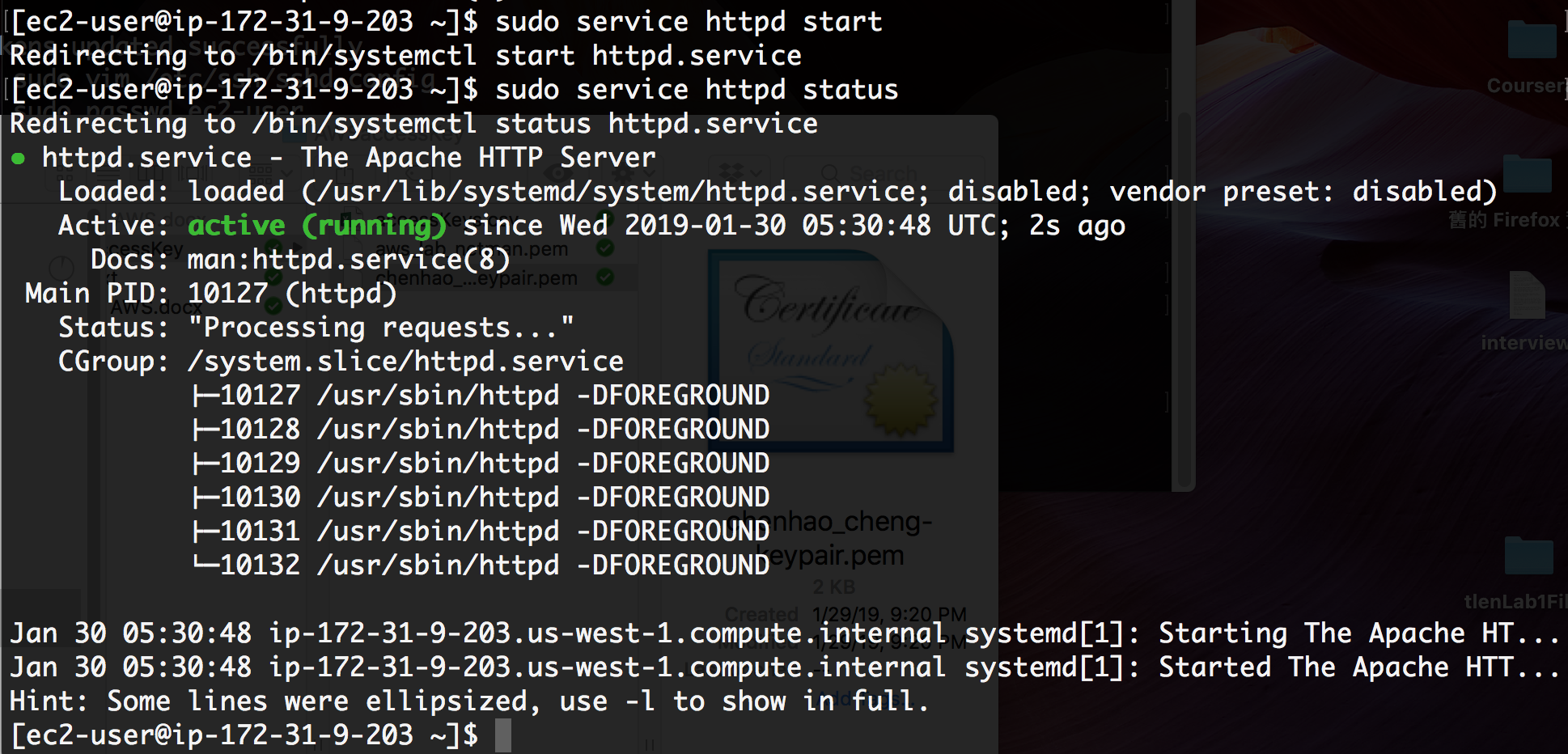
|  |
| --- |
| <HTML>  <HEAD>  <TITLE>NetMan</TITLE>  </HEAD>  <BODY>  <H2>Your\_Name</H2>  </BODY>  </HTML> |

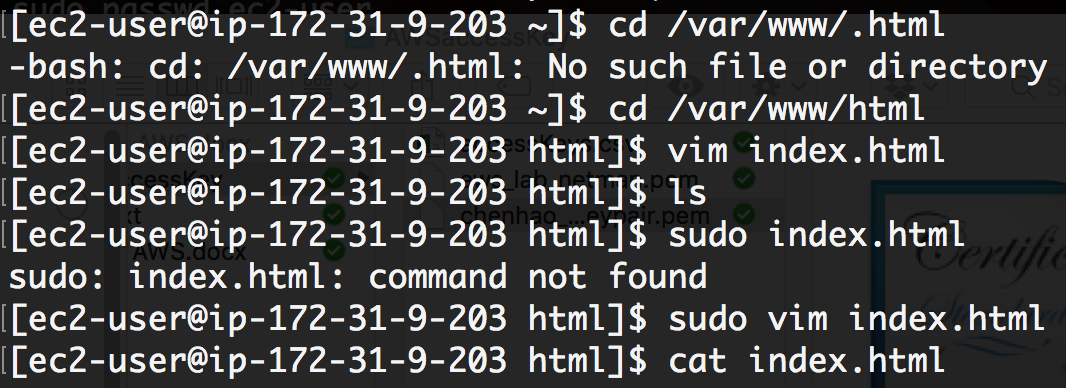
How do you access this webpage hosted on the instance from your laptop? Paste relevant screenshots. Can you access it using an IP address or DNS name or both? [**20 points**]

You can do it both. Follow the following pictures.

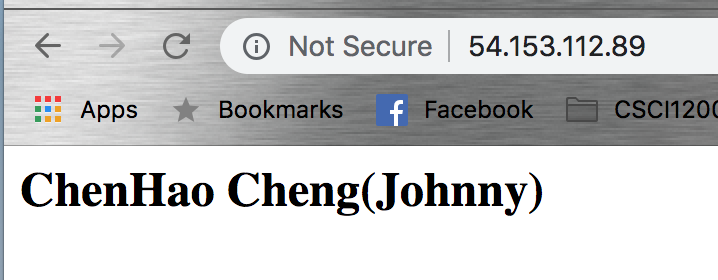
Do ‘sudo yum update -y’ if needed, then do ‘sudo yum install httpd -y’ to instead Apache

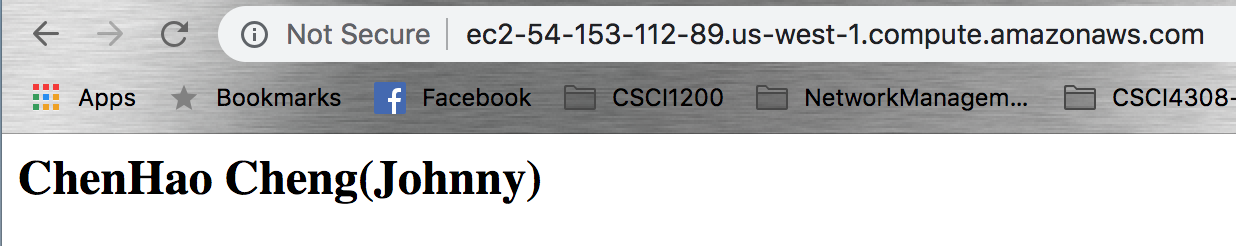






Go back and copy the Public DNS (IPv4) from instances from Dashboard, and paste it to the web browser, you will see the content.





# Part 3

Objective 3.1 – Deploying S3 Backups

**NOTE:** It is preferred to use the VM provided to complete this objective

1. What is S3? [**5 points**]

Amazon Simple Storage Service is storage for the Internet. It is designed to make web-scale computing easier for developers. This module has web services interface that user can use to store and retrieve any amount of data anytime they want, from anywhere on the web.[7]

1. You shall now backup some of our router configurations to the cloud. Before you proceed, ensure you have configuration files of your routers stored in a separate folder (Eg. /home/netman/routerConfigs)

[**NOTE:** If you do not have configuration files present on your system, feel free to use other files (images, files, text,etc.)]

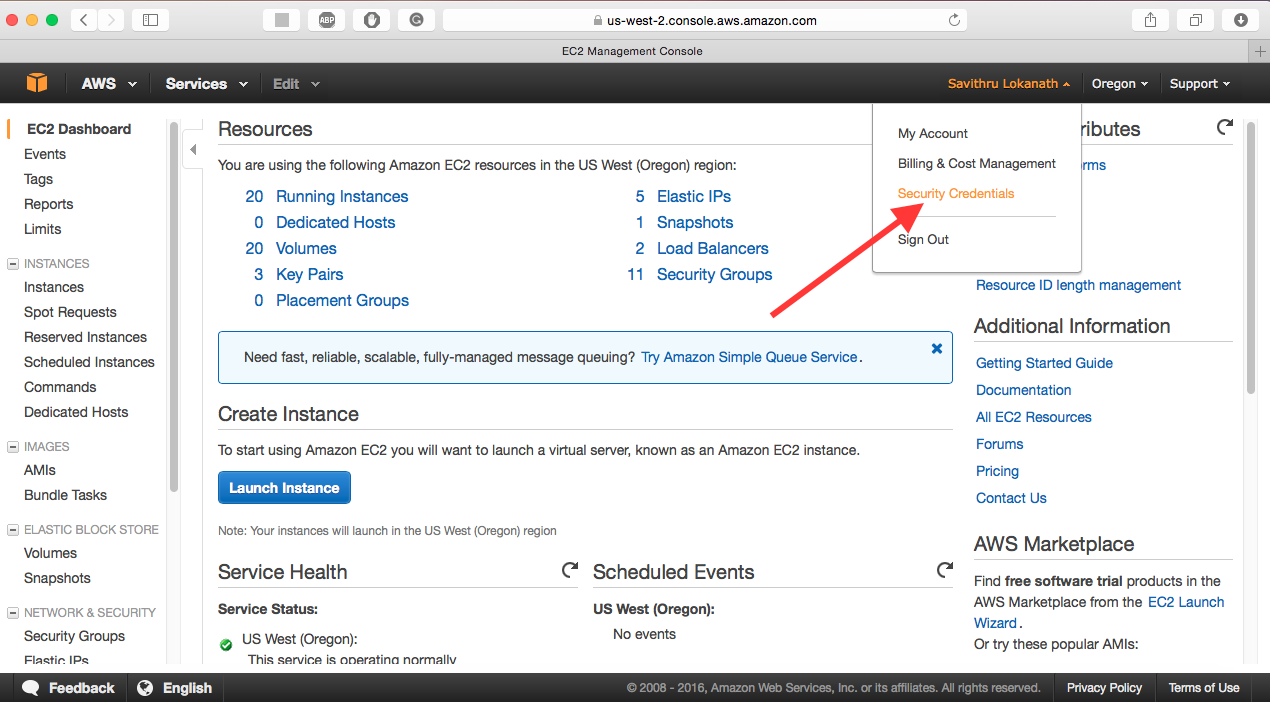
1. Update your VM and install S3 command-line utility

**(VM)# sudo apt-get install s3cmd -y**

1. Configure S3 parameters by entering your AWS access/secret credentials and enter the encryption password (Your choice) when prompted. Save the settings.

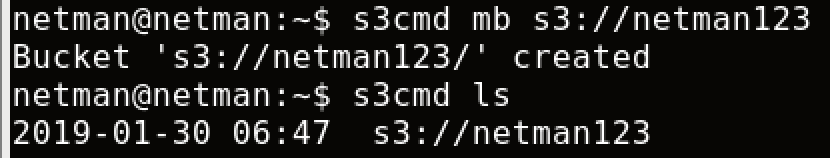
**(VM)# s3cmd --configure**

You can create/find an AWS access/secret key on your “username” > Security Credentials > Continue > Access Keys > Create/Use



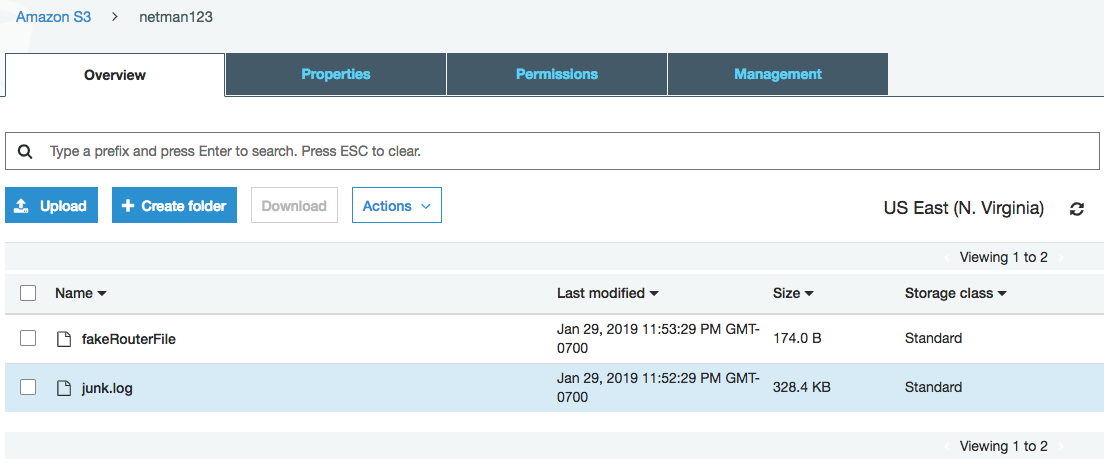
1. Create an S3 bucket

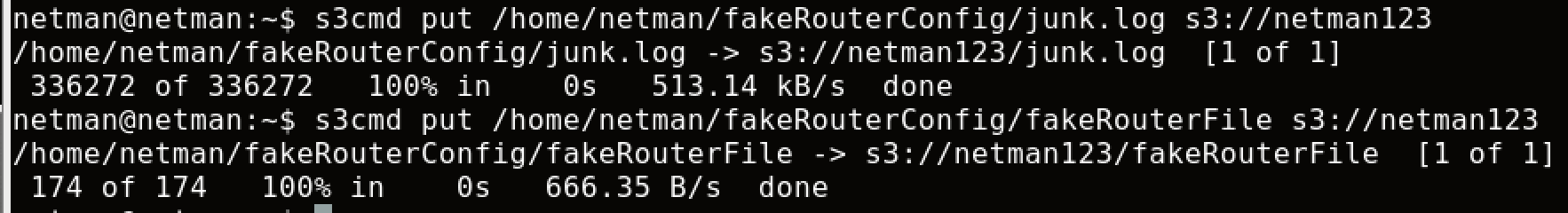
**(VM)# s3cmd mb s3://<S3\_BUCKET\_NAME\_CREATED>**  
Confirm by issuing **“s3cmd ls”**. You can also check using the AWS Management Console. S3 is located under Storage and Content Delivery.



1. Push the configuration folder into the bucket

**(VM)# s3cmd put <PATH\_TO\_LOCAL\_CONFIG\_FOLDER> s3://< S3\_BUCKET\_NAME\_CREATED>**



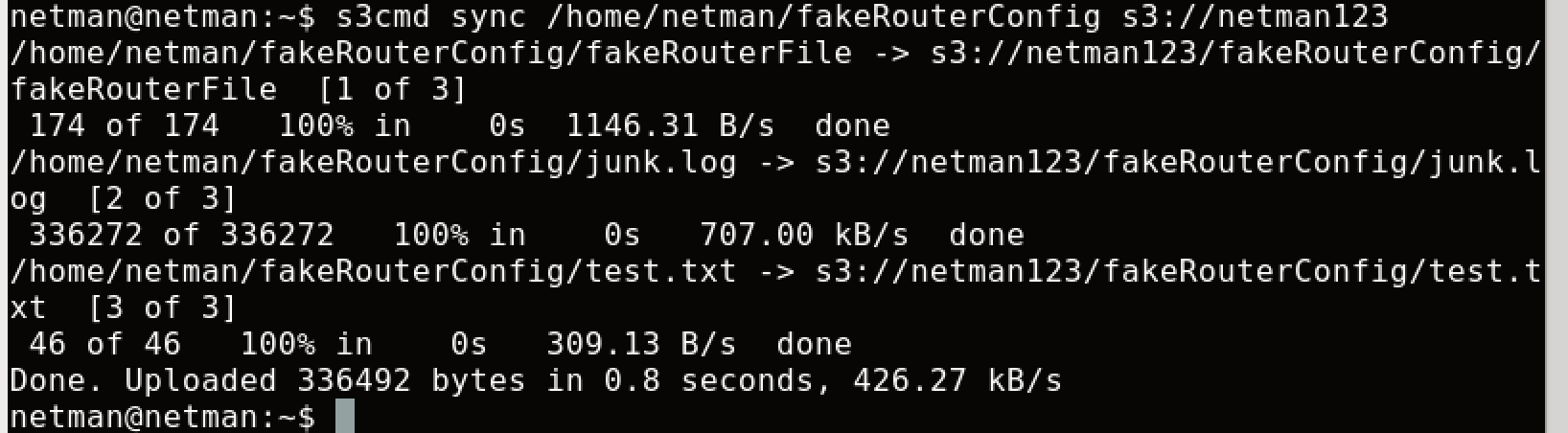
  
Verify that the files are updated. Paste a screenshot showing the same. [**10 points**]

1. What is another way of transferring data to your Amazon S3 bucket? [**2 points**]

﻿ Synchronize a directory tree to S3

s3cmd sync LOCAL\_DIR s3://BUCKET[/PREFIX] or s3://BUCKET[/PREFIX] LOCAL\_DIR

1. Create a **cronjob** to sync every night. Paste a screenshot of the **cronjob** created. [**5 points**]





1. What is Amazon Glacier and how is it different from S3? [**3 points**]

Amazon Glacier is cloud storage service to archive data that is secure, durable, and low-cost. And a very high percentage of the data stored in Amazon Glacier today comes directly from customers using S3 Lifecycle policies to move cooler data into Amazon Glacier [4].

Amazon Glacier is able to offload administrative burdens of operating and scaling storage to AWS for customers, and they don’t need to worry about capacity planning, hardware provisioning, data replication, repair, and migrations for hardware.

# Part 4

# Objective 4.1 – Monitoring using CloudWatch

# Click on any of the running EC2 instances and check the monitoring tab for EC2 metrics. What are the key metrics that you see and why is this important for an organization? [5 points]

CPU utilization: This can help you track your ec2 instances to determine if your instacnces are over or undersize for your workload.

DiskRead (Bytes) and DiskWrite (Bytes): Those portions of metric can help you identify application-level problems. For instance, when you see a large amount of data is being read from disk consistently, maybe, you could improve application performance by adding a caching layer. [5]

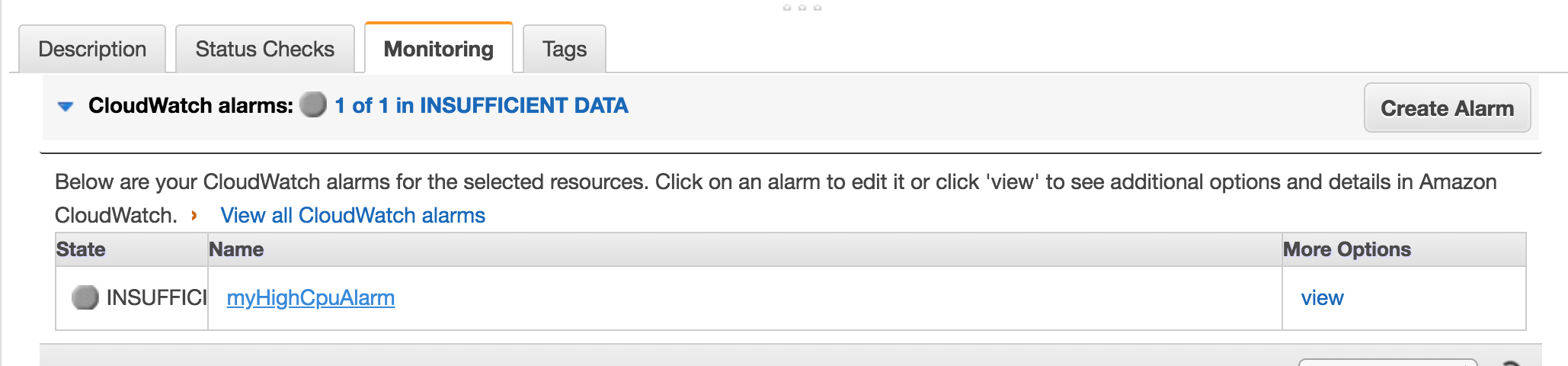
StatusCheck Failed: There are two status checks running at one minute intervals by CloudWatch. The first one check that queries the availability of each ec2 instance, and the second report information about the system hosting the instance. They all provide ec2 health, and also help you determine whether the scource of an issue lies with the instance itself or the underlying infrastructure that supports the instance. [5]

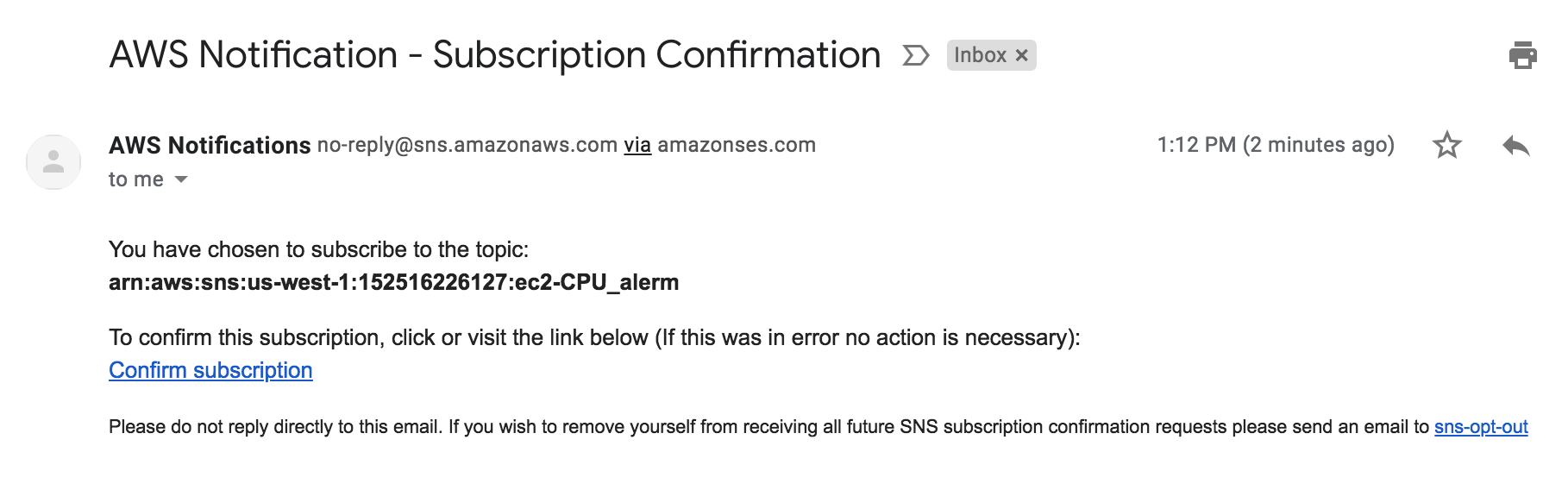
1. Enable detail monitoring for CloudWatch metrics. What is the difference between Basic monitoring and Detailed monitoring? [**5 points**]

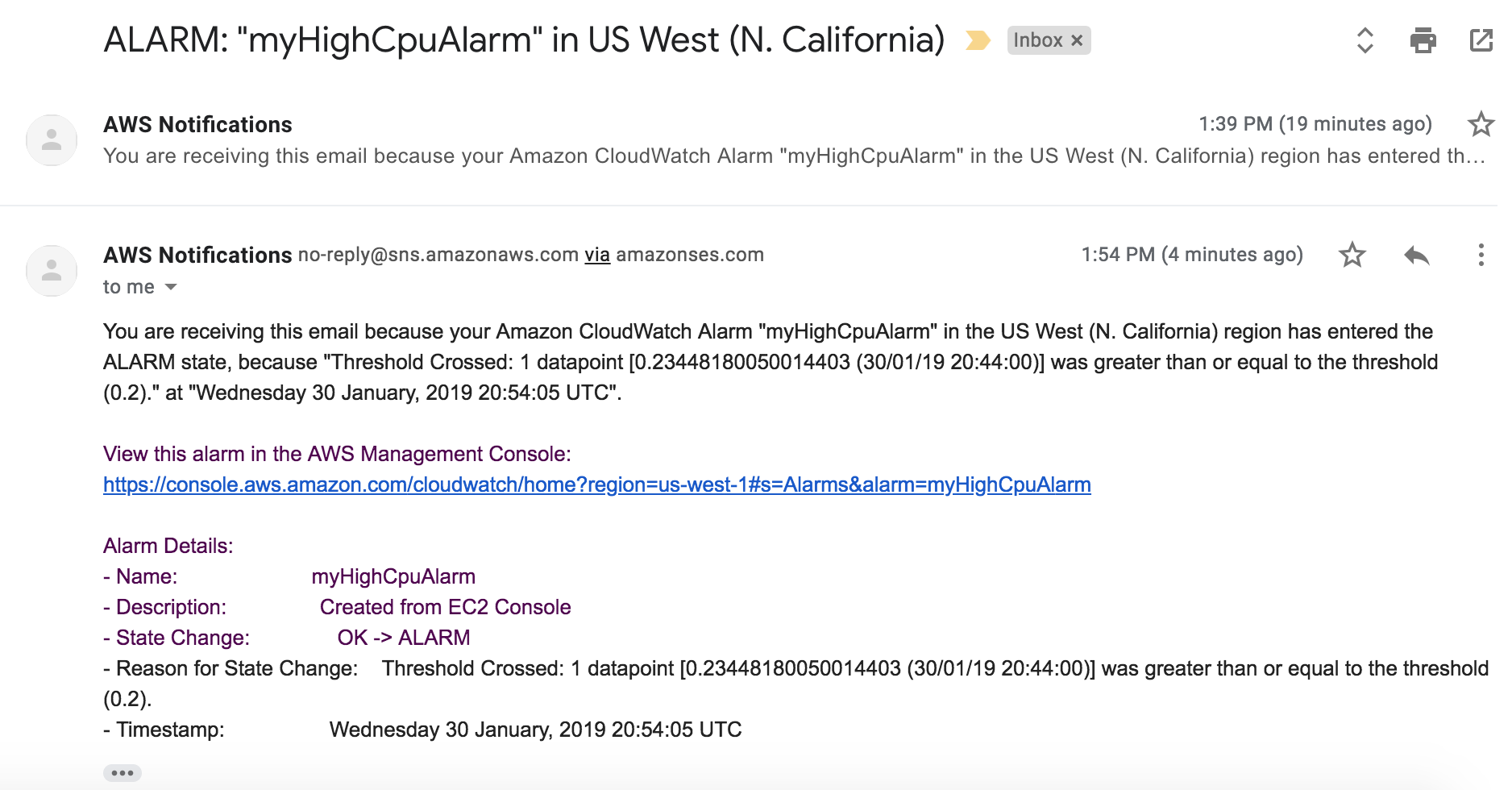
Basic: Data is available automatically in 5 minutes periods at no charge. [6]

Detailed: Data is available for additional cost in 1 minute periods. You must specifically enable it for the instance in order to get this level of data. [6]

1. Create a new **CloudWatch alarm** to monitor **average CPU utilization**. The alarm should take effect when average CPU utilization is greater than a user-defined threshold. Alarm should send an **E-Mail** to the recipients entered during the alarm configuration. Paste a screenshot of the alarm created and the Email that you received. [**10 points**]







[**NOTE:** For simplicity, enter the threshold to be less than 1%.]

1. Which is the service used by CloudWatch to send out E-Mail notifications? [**5 points**]

Amazon use Simple Notification Service to send emails.

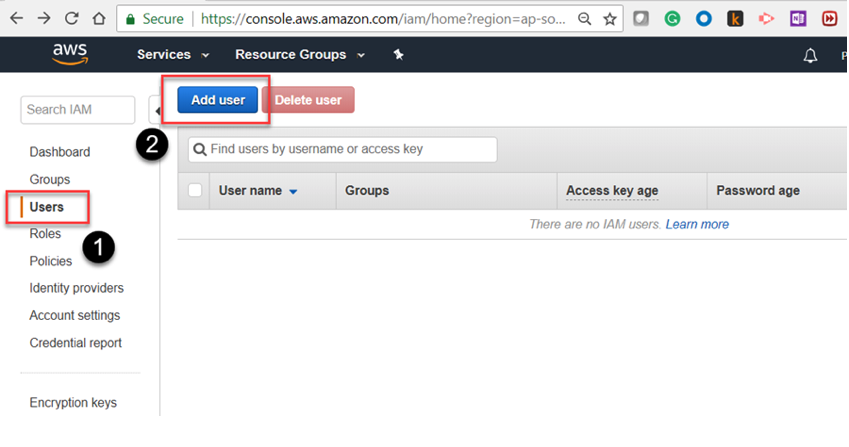
# Part 5

# Objective 5.1 Setting up BOTO3 for AWS resource automation

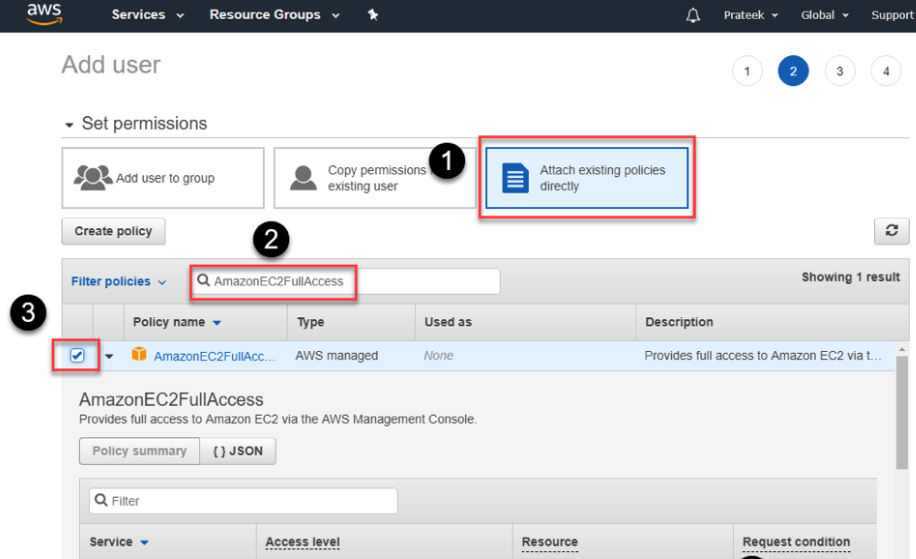
1. Install Boto 3 on your machine using “sudo -H pip install boto3” to download the required packages.
2. Before we start using Boto, it is mandatory to setup the necessary authentication to the AWS management console. In order to do this, we would need to download the AWS CLI and put in the AWS Access Keys which we have already downloaded in objective 1.3.
3. You may install the awscli for ease of authentication using “sudo -H pip install awscli” command. Further steps to achieve this are found in the below mentioned link: [https://boto3.amazonaws.com/v1/documentation/api/latest/guide/quickstart.html#installation](https://boto3.amazonaws.com/v1/documentation/api/latest/guide/quickstart.html%23installation).

After download aws cli with ‘﻿sudo pip install awscli --upgrade --user’ , check if it’s properly installed by ‘﻿~/.local/bin/aws –version’ , then create .bash\_profile and add the following line: ﻿export PATH=~/.local/bin:$PATH ,then do ‘﻿source ~/.bash\_profile’ to refresh the terminal.

Next, go to this link: <https://console.aws.amazon.com/iam/home#/home> to add new user,

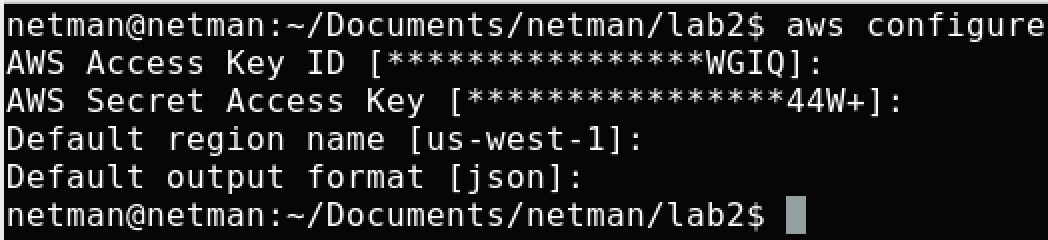


Then pick a username, and check the programmatic access, go next:



Then click next:

Set Key and value (optional), click Review and confirm. Then download the .csv file if you want. Now you can configure aws cli with Access Key ID and secret Key by doing ‘aws configure’ in terminal like the following:



Note: region should be [us-west-1] instead of [us-west-1x], output format could be json or others. Json is more readable.

Now, you can start programming.

1. If you opt to not set up one-time authentication with AWS CLI, ensure you add Var\_name = Session (aws\_access\_key\_id = ' ', aws\_secret\_access\_key = ‘ ‘ , region\_name = ' ') in your python code for further objectives.

# Objective 5.2 Controlling EC2 using Boto3

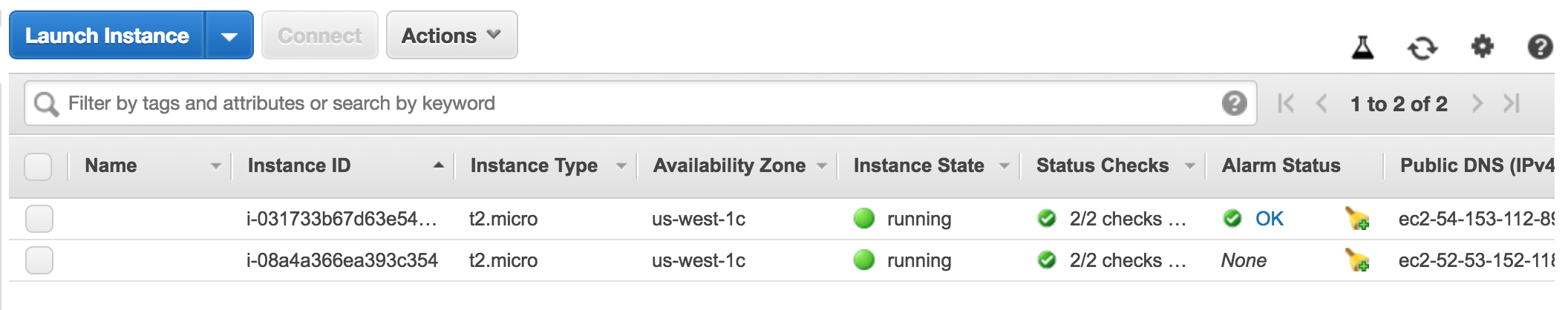
1. Write a python script to launch two new EC2 instances, stop one of the instances, and then fetch the details of all instances using the Boto3 module. Sample output:

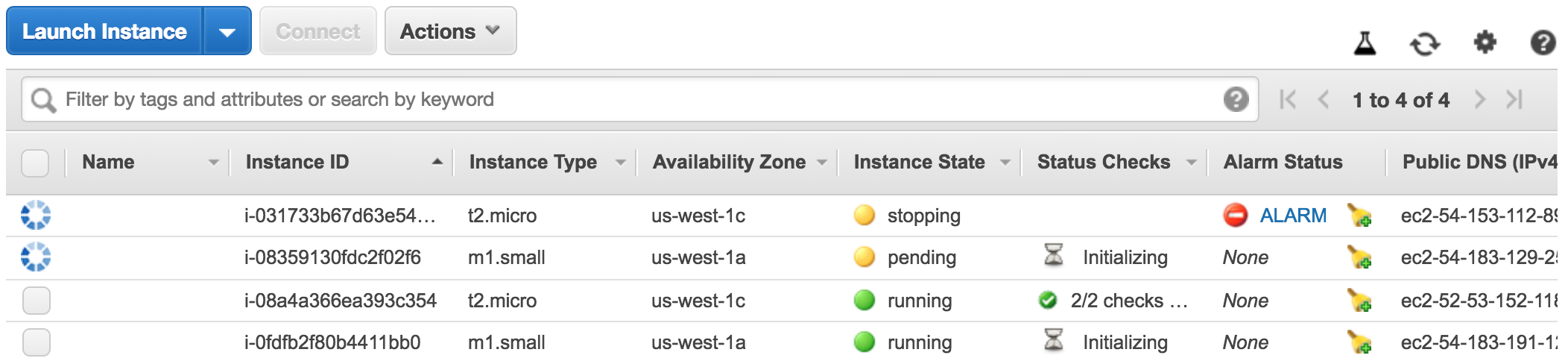
**[Instance Id] [Instance\_type] [Instance\_ip\_address] “Running/Stopped”**

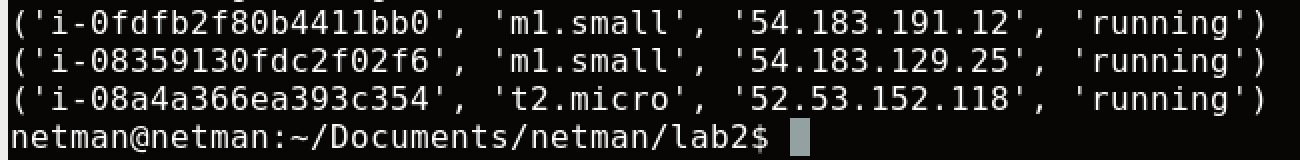
[Hint: refer <https://boto3.readthedocs.org/en/latest/guide/migrationec2.html>]

The ami-value should be the same as one that’s currently running on Amazon.

1. Submit the .py file that you created to accomplish this objective, screenshots showing the new instances created on the console, and the instance details as specified in the sample output. [**30 points**]







# Objective 5.3 Fetching Cloudwatch metrics using Boto3

1. Write a python script to create a new AWS session using access keys (refer objective 1), create a cloudwatch session, and fetch the following metrics for one running EC2 instance over a specific time period (at least 30 minutes): Status\_Check, CPU\_Utilization, Network\_In and Network\_Out. Sample output:

**Instance ID: <value>**

**Status Check: <value>**

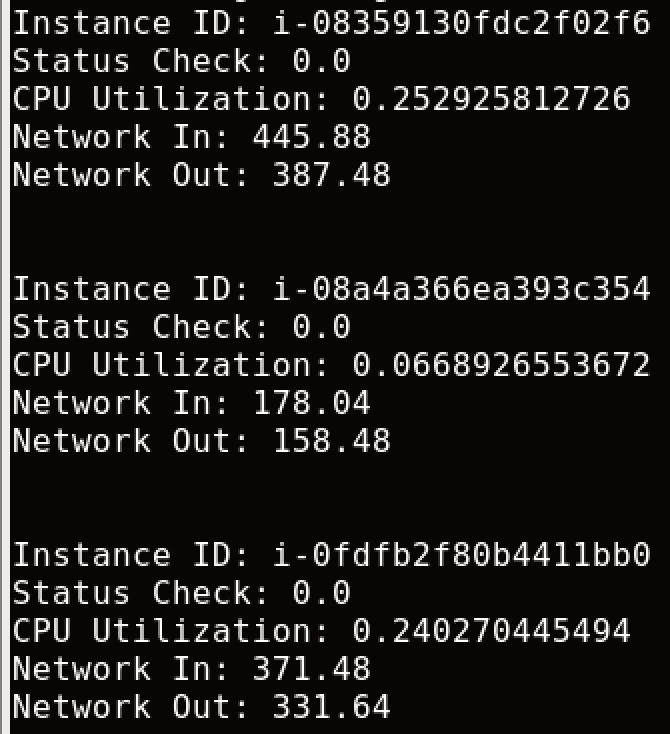
**CPU Utilization: <value>**

**Network In: <value>**

**Network Out: <value>**

[Hint: [https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/cloudwatch.html#CloudWatch.Client.get\_metric\_statistics](https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/cloudwatch.html%23CloudWatch.Client.get_metric_statistics)]

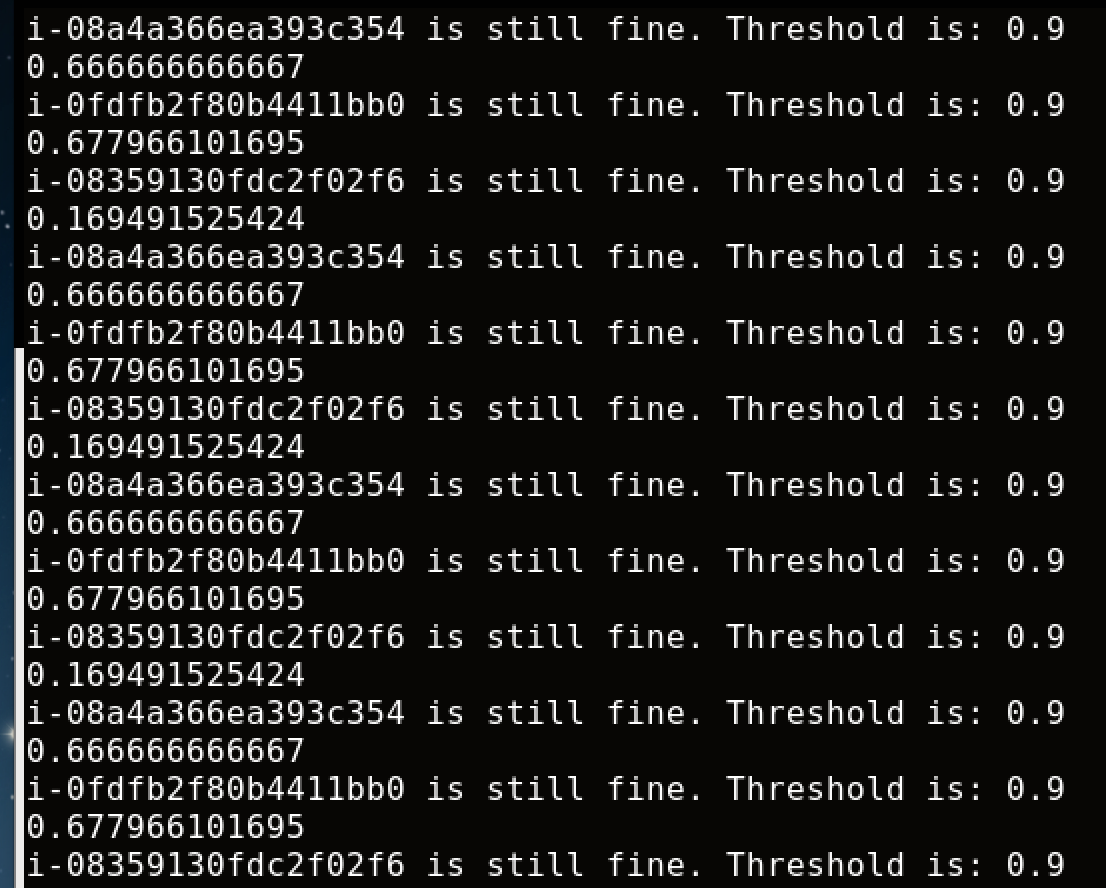
1. Submit the .py file that you created to accomplish this objective and screenshots of the details as specified in the sample output. [**20 points**]



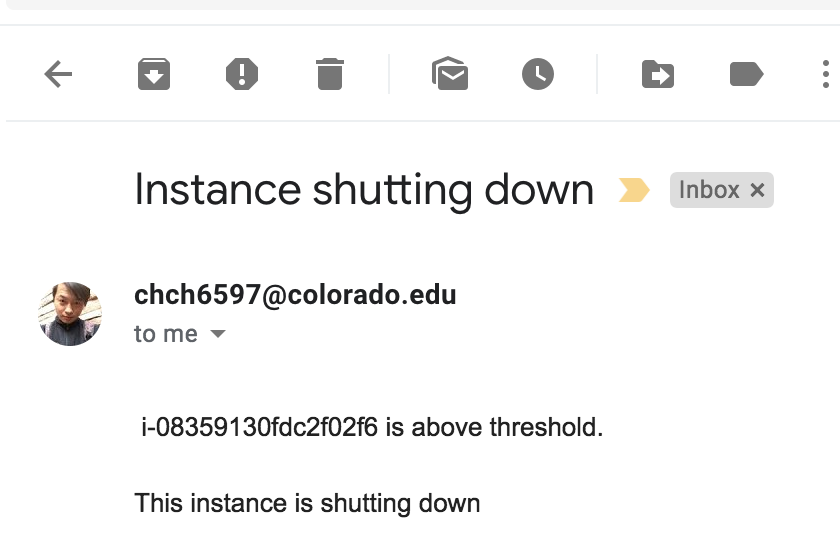
Objective 5.4 Spinning new instances based on CPU utilization

Write a python script that uses Boto3 to continuously fetch CPU utilization of two running EC2 instances. When a specific threshold is reached, the script should automatically shut the instances down, spin up identical new instances and send out an alert email to your email id. Submit the .py file that you created to accomplish this and relevant screenshots. [**20 points**]

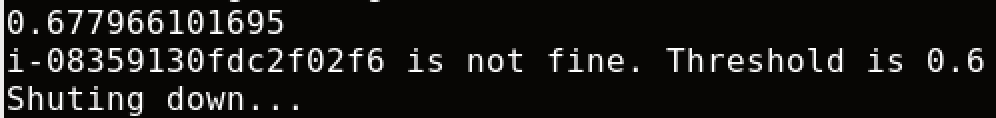
When I set threshold = 0.9, it will keep running.



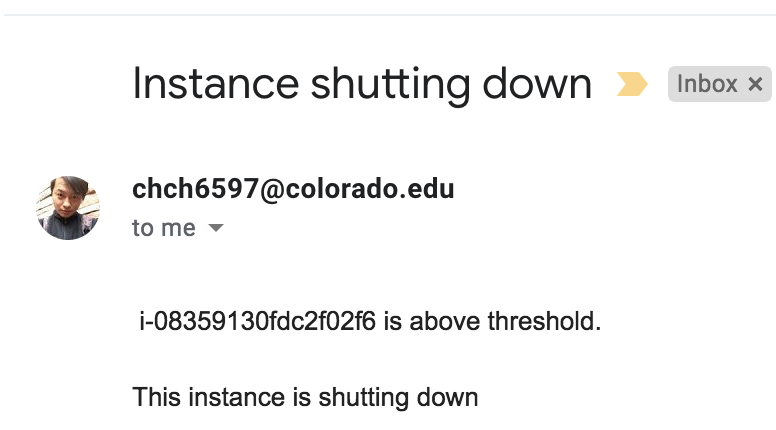
When I set threshold lower than 0.6:



Every time when I set an email, I need to make the program sleep for at least 90 second because google would think it’s spam if I keep sending email when it’s above threshold.







Reflection:

1. Now that you have learnt the basics, what do you think are the most important reasons for an organization to use AWS? [**5 points**]

* It’s pretty easy to use even me count as a beginner. Everyone should be able to learn it quickly
* AWS is pretty reliable. Their network is really unlikely down because it’s a huge company just like Google.
* As a company like Amazon, their security should be really taken a good care of.

1. Suggest any other AWS modules that you would like to learn about. Why?

I think what we have learned about modules and attributes are pretty good and solid knowledge to use. We can check the status of instances only on CLI/Python without go into GUI in Amazon. I’m pretty good and happy with what I have learned.

1. Suggest any other public cloud platforms that you would like to learn about. Why?

I’m not sure if we should learn other modules. As a IT guy, EC2 and S3 are most popular to use in industry. I personally think this is good enough.

# Total Points \_\_\_\_\_\_\_\_\_\_\_\_ / 207

References:

1. <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AMIs.html>
2. <https://searchnetworking.techtarget.com/definition/east-west-traffic>
3. <https://docs.rightscale.com/faq/clouds/aws/Whats_the_difference_between_Terminating_and_Stopping_an_EC2_Instance.html>
4. <https://aws.amazon.com/glacier/faqs/>
5. <https://www.datadoghq.com/blog/aws-monitoring/>
6. <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-cloudwatch-new.html>
7. <https://docs.aws.amazon.com/AmazonS3/latest/dev/Welcome.html>
8. <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/dedicated-hosts-overview.html>
9. <https://aws.amazon.com/ec2/elastic-graphics/>
10. https://aws.amazon.com/ec2/purchasing-options/dedicated-instances/