Lab 1

Saif Ali Athyaab - 23810756

Lab 1 covers the basics to set up our system for upcoming labs to accomplish the AWS tasks.

1. Registering for the AWS account is straightforward and access can be changed through reset password
2. Next we obtain the API key and secret key. These are 2 strings which are generated by AWS.
3. Complex cryptography algorithms like RSA employing SHA hash functions are using in the backend by AWS
4. In my case, Ubuntu OS was physically installed on my laptop through Windows dual-boot.
5. AWS Command Line Interface(CLI) is installed on our machine to interact with our AWS resources right from the Terminal.
6. We configure the Python version and install Boto3, which is an AWS SDK alllowing to run instances from Python script.
7. Finally, we verify AWS CLI through Terminal as well as Python file through Boto.

AWS Accounts and Log In

[1] AWS Account Login and Reset Password

Using the provided login link: https://489389878001.signin.aws.amazon.com/console

I was able to successfully login and reset my password.

AWS Console Sign-In Page

[2] Ensuring AWS Console Login

Upon entering my new password I was able to access the AWS Console

I have logged in as an **IAM User**. An IAM user enables to login to the Console without full-system access like an admin user.

Successful Login

[3] Identity Access Management

On the AWS Console **Search Bar**, I have typed in *Identity* and upon clicking the first link, I am directed to the IAM Console.

Identity Access

The **IAM Console** is an important resource for all things security like

- Security Keys
- Creation of IAM users
- Security groups

Below I have created an access key named **cclab**

Access Key

Dual-Boot Ubuntu on a Windows Laptop

I am already using an Ubuntu on my Windows Laptop as a dual-boot. So I shall provide a brief write-up of the process, since I will not be able to perform it again.

[1] Download the Ubuntu .iso file

Navigate to the Ubuntu Downloads page

Ubuntu 23.04

We will use this file to create a bootable usb in the next step.

[2] Download an etching software like Balena Etcher

Balena Etcher basically enables a normal USB removable drive to become a bootable drive.

Now when we advanced start-up our laptop, we can choose to **boot from drive** by changing the boot-order.

Download Balena

Once downloaded, we simply select the drive and the location of the .iso file and Balena automatically makes a bootable drive in less than 30 minutes.

[3] Boot from removable drive and Install Ubuntu

Once we advance start-up our laptop, we get the Ubuntu interface and we can choose to Install the OS.

Other details such as disk partitioning also to be considered. I have included a screenshot of the same below

Partitions

See below my installed Ubuntu config

Ubuntu About

AWSCLI, Boto and Python 3.8.x

[1] Install Python 3.8.x

I run the below commands on my machine to update to the latest Python 3 version. *sudo* is the keyword used to get Admin access while executing any command on Ubuntu.

```
sudo apt update
sudo apt -y upgrade
```

In the first command we get the package lists of all apps sudo apt update **upgrade** command reads the package lists upgrade Then it displays the ones which will be removed and updated upgrade 2 Finally, it displays the size of files to be downloaded. Since we already included **-y** in our command arguments, it takes in YES. upgrade 3 To check the latest version:

```
python3 -V
```

Python version Now we need to install `pip3`, which is a tool that will allow us to install and manage python libraries.

```
sudo apt install -y python3-pip
```

I have already installed it pip3

[2] Install awscli

AWS CLI is the most important part of this lab and the upcoming ones. Basically, it allows developers to access all the features of the AWS user console right from the CLI. The advantage of this being, allowing a developer to access and manage multiple AWS resources from the command line itself.

```
sudo snap install aws-cli --classic
```

I have already installed earlier aws cli install

Configure aws using aws configure

The AWS **configure** command allows us to modify the AWS CLI settings from the CLI. Important parameters include

- Secret Key ID
- Secret Access Key
- Region
- Receiving format - JSON/ CSV

Here is my .csv file containing the key info

csv key

We can also configure IAM role assumptions, environment varibales, etc.

aws configure

[3] Install boto3

Boto 3 is a system to set up the AWS SDK on CLI and access the various functions related to AWS.

```
pip3 install boto3
```

Boto3

We are now in place to call the decribe-regions function

Exploring and testing the environment

[1] Test the aws environment by running:

```
aws ec2 describe-regions --output table
```

The **describe-regions** command provides three important details about our AWS account

1. The URL endpoint for the region
2. The Opt-In status from user-side
3. The Region Name

By adding **--output-table** argument we can specify the output to be pretty-formatted in a visual tabular table.

Regions

[2] Test the python environment

```
python3
>>> import boto3
>>> ec2 = boto3.client('ec2')
>>> response = ec2.describe_regions()
>>> print(response)
```

Python output This will create an un-tabulated response.

[3] Python File with explanation

```
import boto3
```

Importing the Pretty Table package to automate the line-drawing process

```
from prettytable import PrettyTable
```

Getting the response about ec2 regions

```
ec2=boto3.client('ec2')
response=ec2.describe_regions()
```

Filtering out the unnecessary details from received dictionary

```
regions=response["Regions"]
```

Storing the column names in a list variable

```
columns=["Endpoint","RegionName"]
```

Initialising pretty table object

```
region_table=PrettyTable()
```

Creating empty lists and appending all rows data into respective lists

```
endpt, regn =[], []
for i in regions:
    endpt.append(i["Endpoint"])
    regn.append(i["RegionName"])
```

Calling the add-column function to pass in 2 arguments

1. Column name from the column list
2. Respective column data from the list

```
region_table.add_column(columns[0],endpt)
region_table.add_column(columns[1],regn)
```

Finally printing our desired output

```
print(region_table)
```

Output

Output