Secure Data Storage and in Cloud

A PROJECT REPORT

Submitted by

Ankit Kumar Sinha 17BIT0045

Prashant Jadaun 17BIT0364

Abhishek Sharma 17BIT0026

Akash Kumar 17BIT0055

Course Title -: Cloud Computing and Virtualization (ITE3007)

Slot -: A1 + TA1

Under the guidance of

Prof ASIS KUMAR TRIPATHY

Professor, SITE,

VIT University, Vellore

Table of Contents

Objective

- To provide encrypted data storage and data sharing system to improve security.

Introduction

"Nowadays, Data is moving to the cloud at a record pace."

Cloud-based solutions are increasingly in demand around the globe. These solutions incorporate everything from secure data storage to entire business processes. Cloud-based internet security is an outsourced solution for storing data. Instead of saving data onto local hard drives, users store data on Internet-associated servers. Data Centers manage these servers to keep the data safe and secure to access. Enterprises turn to cloud storage solutions to solve a variety of problems. Small businesses use the cloud to cut costs. IT specialists turn to the cloud as the best way to store sensitive data. Cloud based data storage provides us the any time access to the file as these files are stored remotely on server. Email is a prime example. Most email users don't bother saving emails to their devices because those devices are connected to the Internet. As we store our confidential documents on cloud, we need some type of encryption to be safe from security breaches.

Security breaches caused by poor cloud data protection. More than 40% of data security breaches occur due to employee error. the issues related to the cloud data storage such as data breaches, data theft, and unavailability of cloud data.

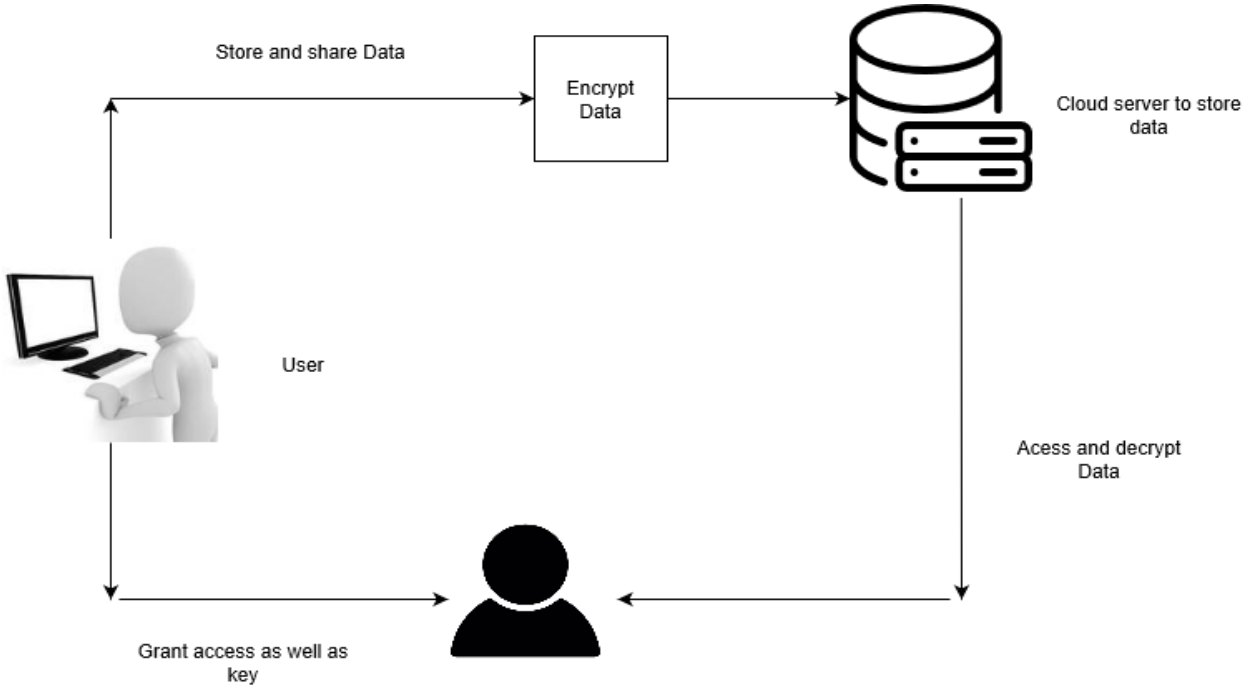
Factors affecting security breaches -:

- Awareness
- Authentication
- Phishing Protection
- Security Measures

"Negligent use can compromise even the best protection"

Project Description

As data security is important we in this project are providing the secure data storage of your confidential files on the cloud storage system. By encrypting the file using AES 128-bit encryption using a predefined key securely stored not known by any other person on AWS S3 bucket.



Technology Used -:

- Cloud provider – Amazon Web Services
- Cloud Services – Aws ec2 instance, Aws s3 storage bucket, boto3 – to store and retrieve data programmatically.
- Programming Language Used – Python
- Framework and packages used – Flask, cryptography

Amazon web services

Amazon Web Services (AWS) is a subsidiary of Amazon that provides on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a metered pay-as-you-go basis. In aggregate, these cloud computing web services provide a set of primitive abstract technical infrastructure and distributed computing building blocks and tools.

Two of these services is -:

- Amazon ec2
- s3 bucket

Amazon ec2 (Elastic compute cloud)

Amazon Elastic Compute Cloud (EC2) forms a central part of Amazon.com's cloud-computing platform, Amazon Web Services (AWS), by allowing users to rent virtual computers on which to run their own computer applications. EC2 encourages scalable deployment of applications by providing a web service through which a user can boot an Amazon Machine Image (AMI) to configure a virtual machine, which Amazon calls an "instance", containing any software desired. A user can create, launch, and terminate server-instances as needed, paying by the second for active servers – hence the term "elastic". EC2 provides users with control over the geographical location of instances that allows for latency optimization and high levels of redundancy.

s3 bucket (Simple Storage Service)

Amazon S3 or Amazon Simple Storage Service is a service offered by Amazon Web Services (AWS) that provides object storage through a web service interface. Amazon S3 uses the same scalable storage infrastructure that Amazon.com uses to run its global e-commerce network.

Amazon S3 can be employed to store any type of object which allows for uses like storage for Internet applications, backup and recovery, disaster recovery, data archives, data lakes for analytics, and hybrid cloud storage. In its service-level agreement, Amazon S3 guarantees 99.9% uptime, which works out to less than 43 minutes of downtime per month. Although Amazon Web Services (AWS) does not publicly provide the details of S3's technical design, Amazon S3 manages data

with an object storage architecture which aims to provide scalability, high availability, and low latency with 99.999999999% durability and between 99.95% to 99.99% availability (though there is no service-level agreement for durability).

Flask

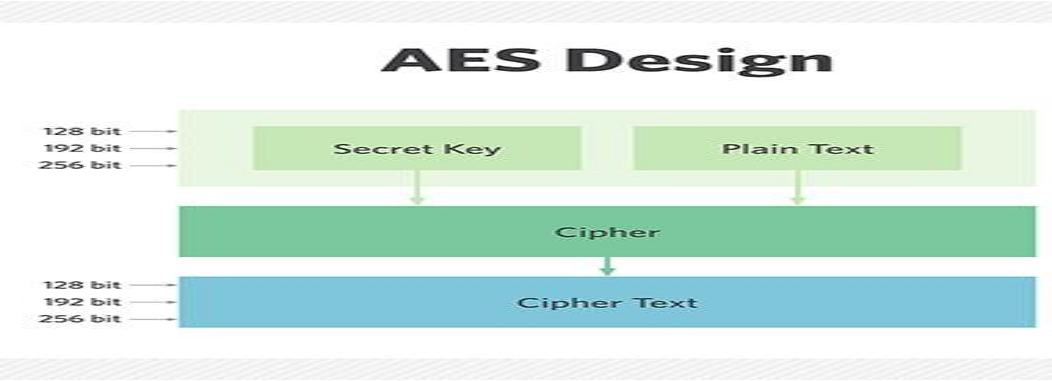
Flask is a Python web framework built with a small core and easy-to-extend philosophy. Flask is an implementation of the web frameworks concept. Flask was originally designed and developed by Armin Ronacher as an April Fool's Day joke in 2010. Despite the origin as a joke, the Flask framework became wildly popular as an alternative to Django projects with their monolithic structure and dependencies.

AES

AES stands for advance encryption standard. AES is a cryptographic technique used for encryption and decryption. AES is a symmetric key block cipher. In AES data size is 128 bit and it is stronger and faster than triple DES. In AES the key size is 128 bit also with 10 rounds of process on the plain text.

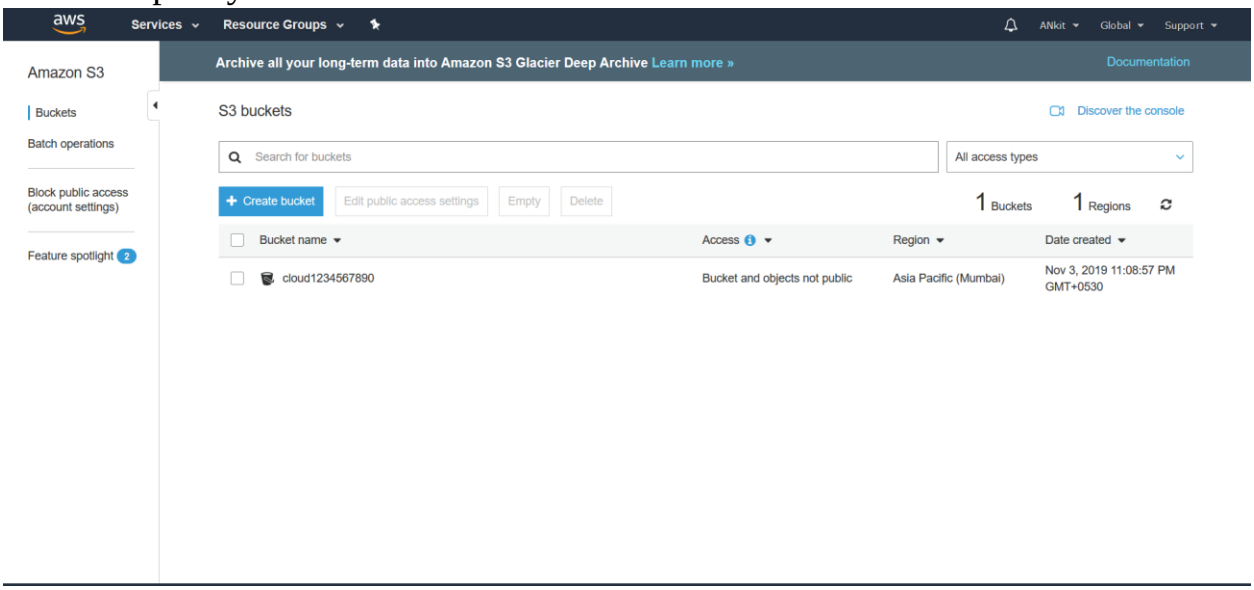
Features of AES –

- Block encryption implementation
- 128-bit group encryption with 128, 192 and 256-bit key lengths
- Symmetric algorithm requiring only one encryption and decryption key
- Data security for 20-30 years
- Worldwide access
- No royalties
- Easy overall implementation



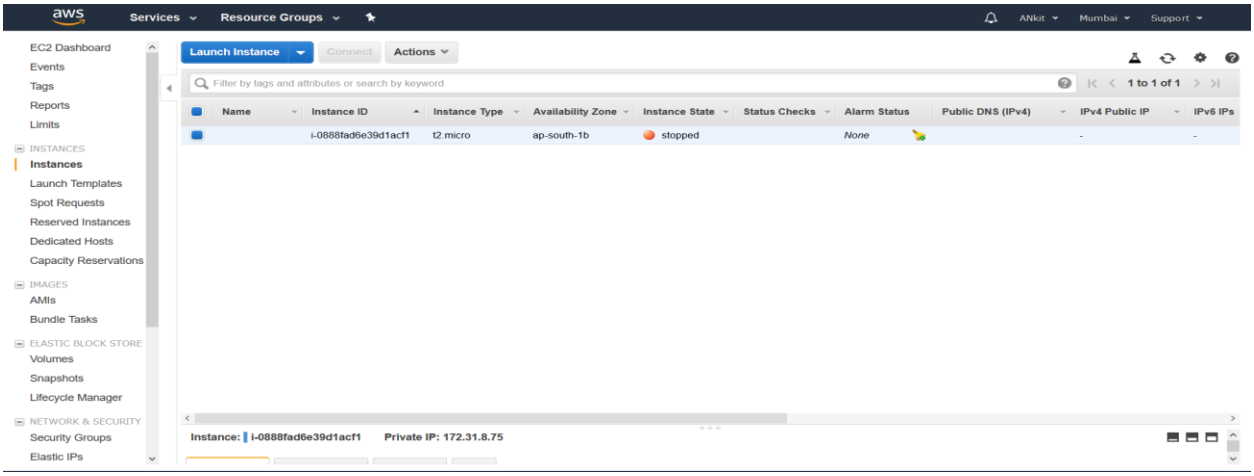
Procedure

- Create a S3 bucket on the Aws. S3 bucket name should be unique and read and write policy must be checked.



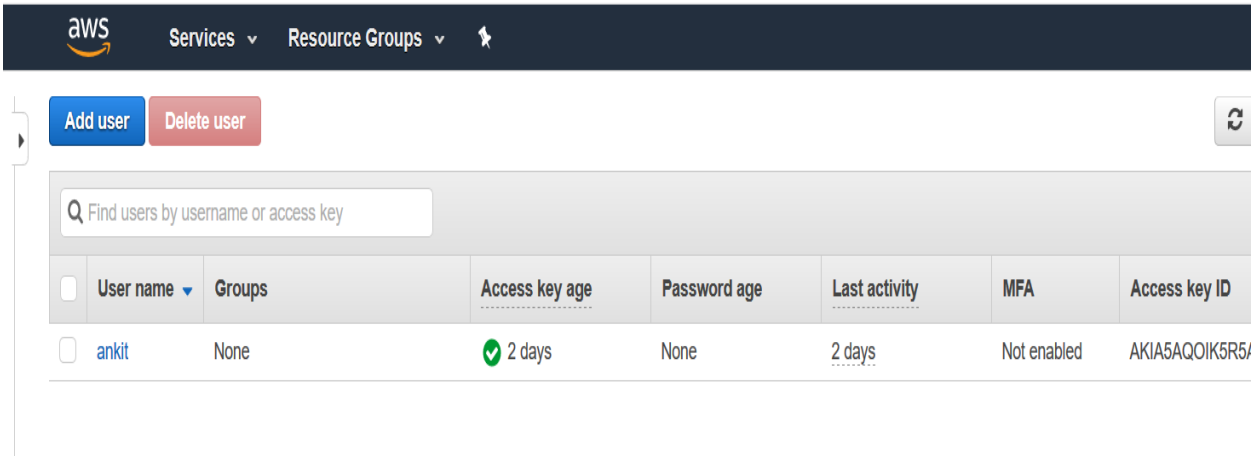
- Create an EC2 virtual machine instance on Aws using Aws web console.

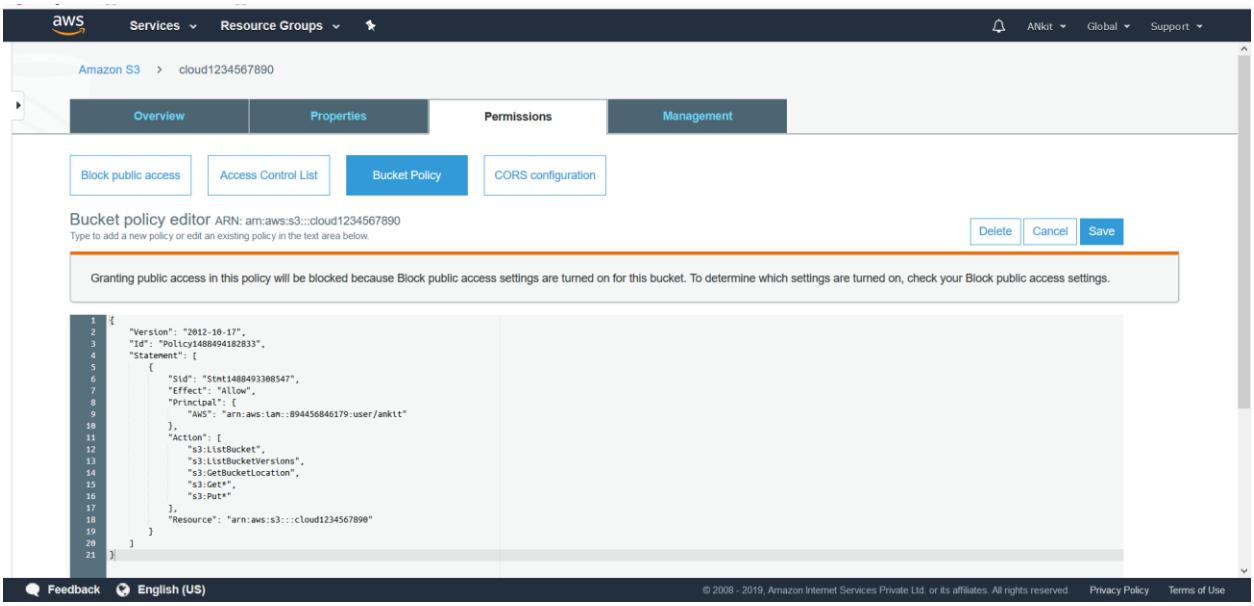
Ec2 instance configuration -:
Operating System = Ubuntu Server 18.004 LTS
Storage – SSD Type 10GB
Ram – 1 GB



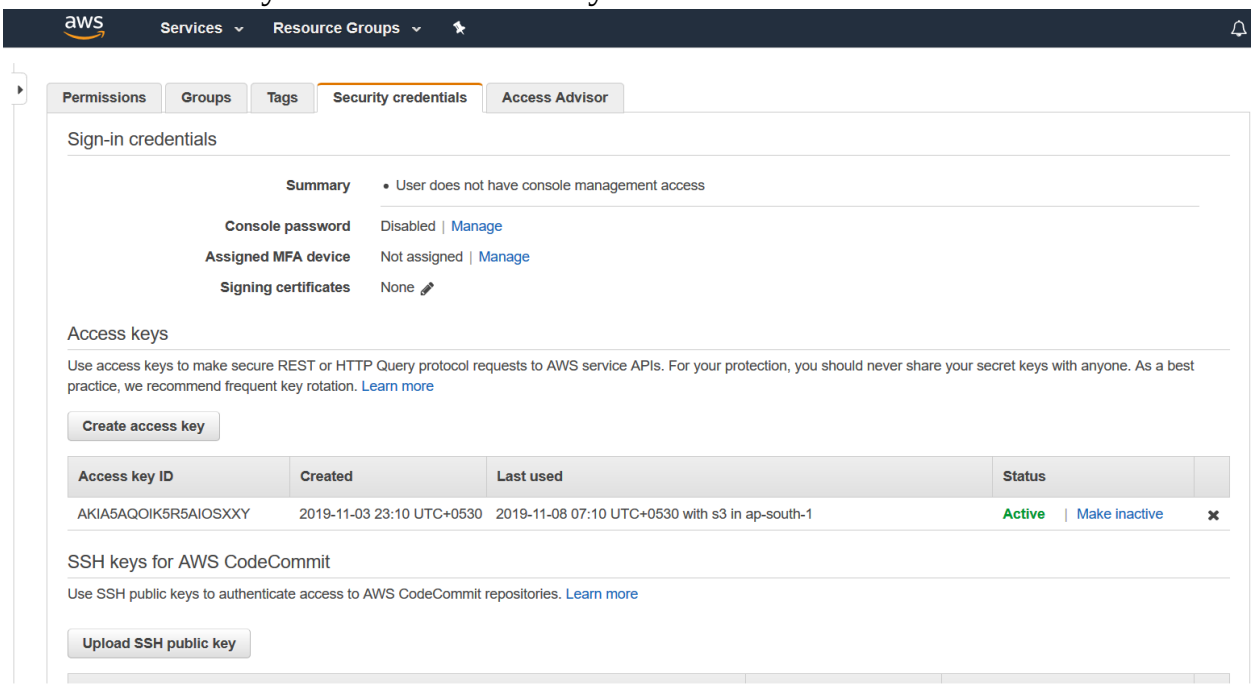
- Create an IAM user policy to read and write Aws S3 programmatically and link it to S3.

```
{
   "Version": "2012-10-17",
   "Id": "Policy1488494182833",
   "Statement": [
      {
         "Sid": "Stmt1488493308547",
         "Effect": "Allow",
         "Principal": {
            "AWS": "arn:aws:iam::894456846179:user/ankit"
         },
         "Action": [
            "s3:ListBucket",
            "s3:ListBucketVersions",
            "s3:GetBucketLocation",
            "s3:Get*",
            "s3:Put*"
         ],
         "Resource": "arn:aws:s3:::cloud1234567890"
      }
   ]
}
```

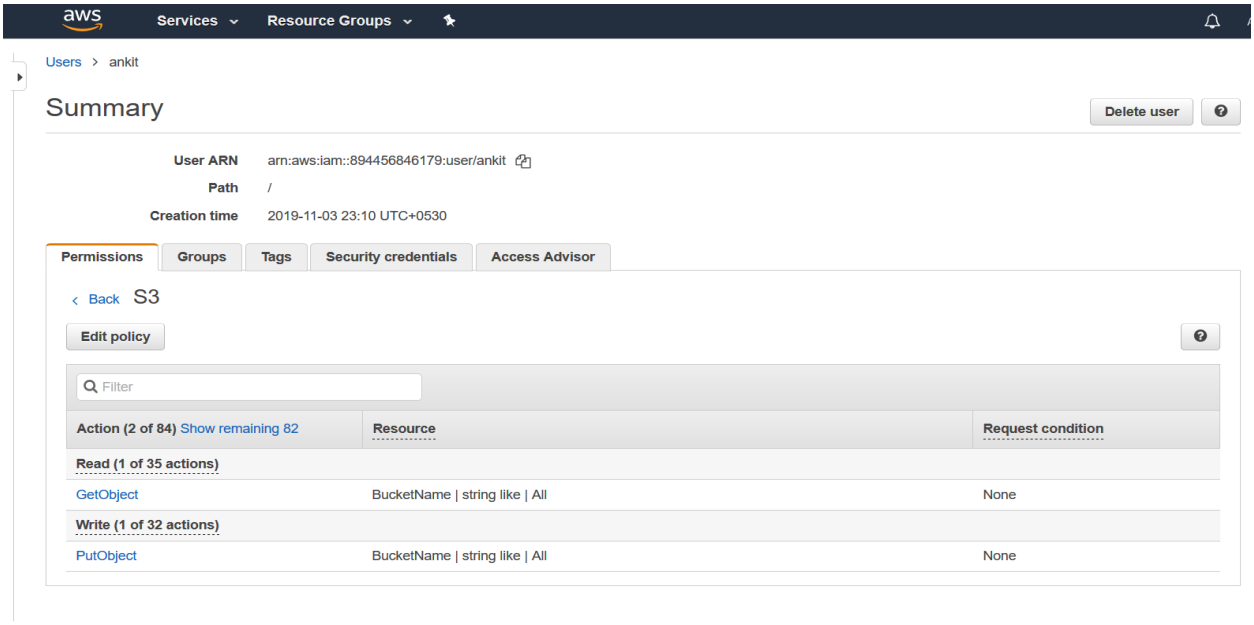




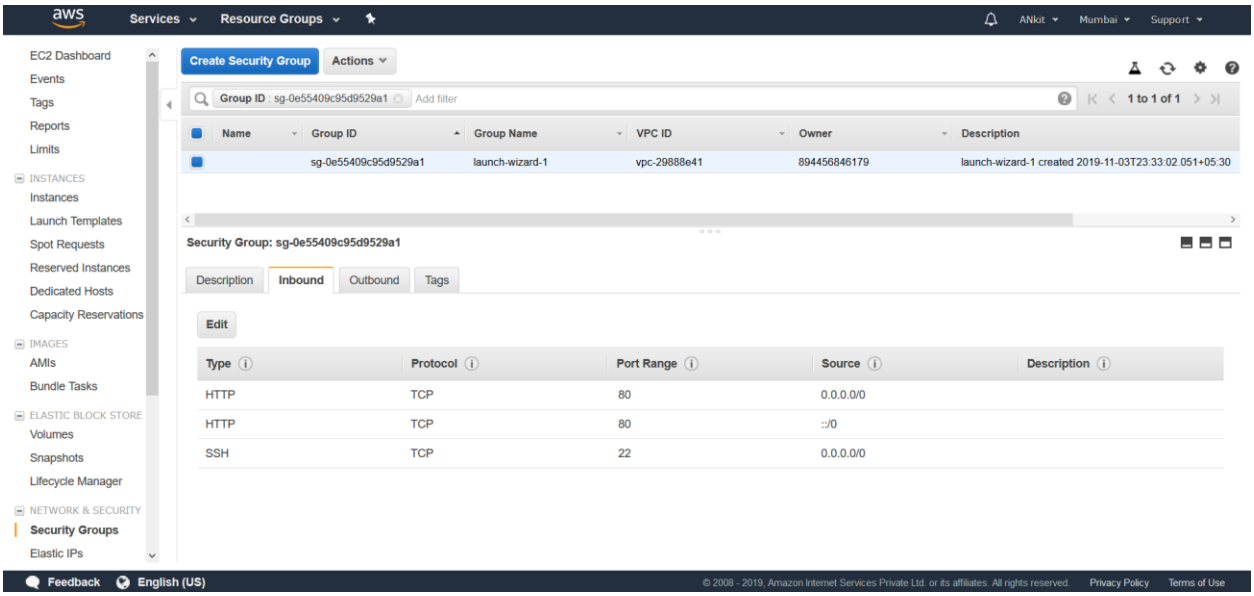
- Get the access key and secret access key.



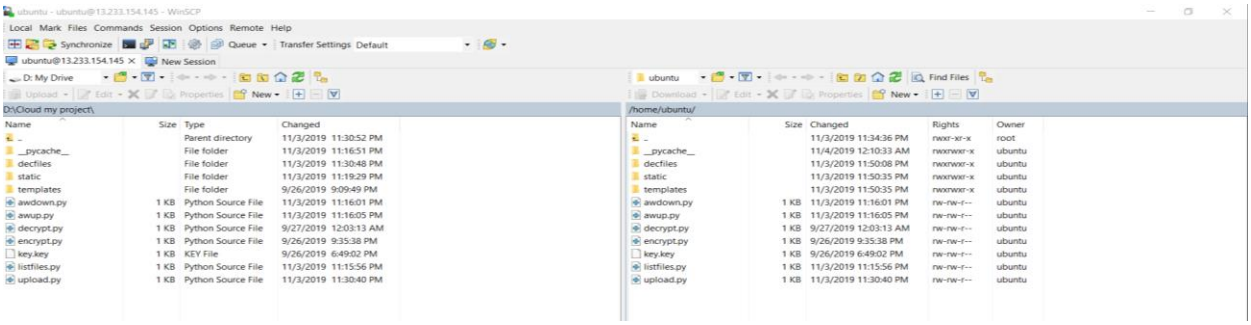
- Verify that the access to S3 is enabled or not in the IAM policy permission section.



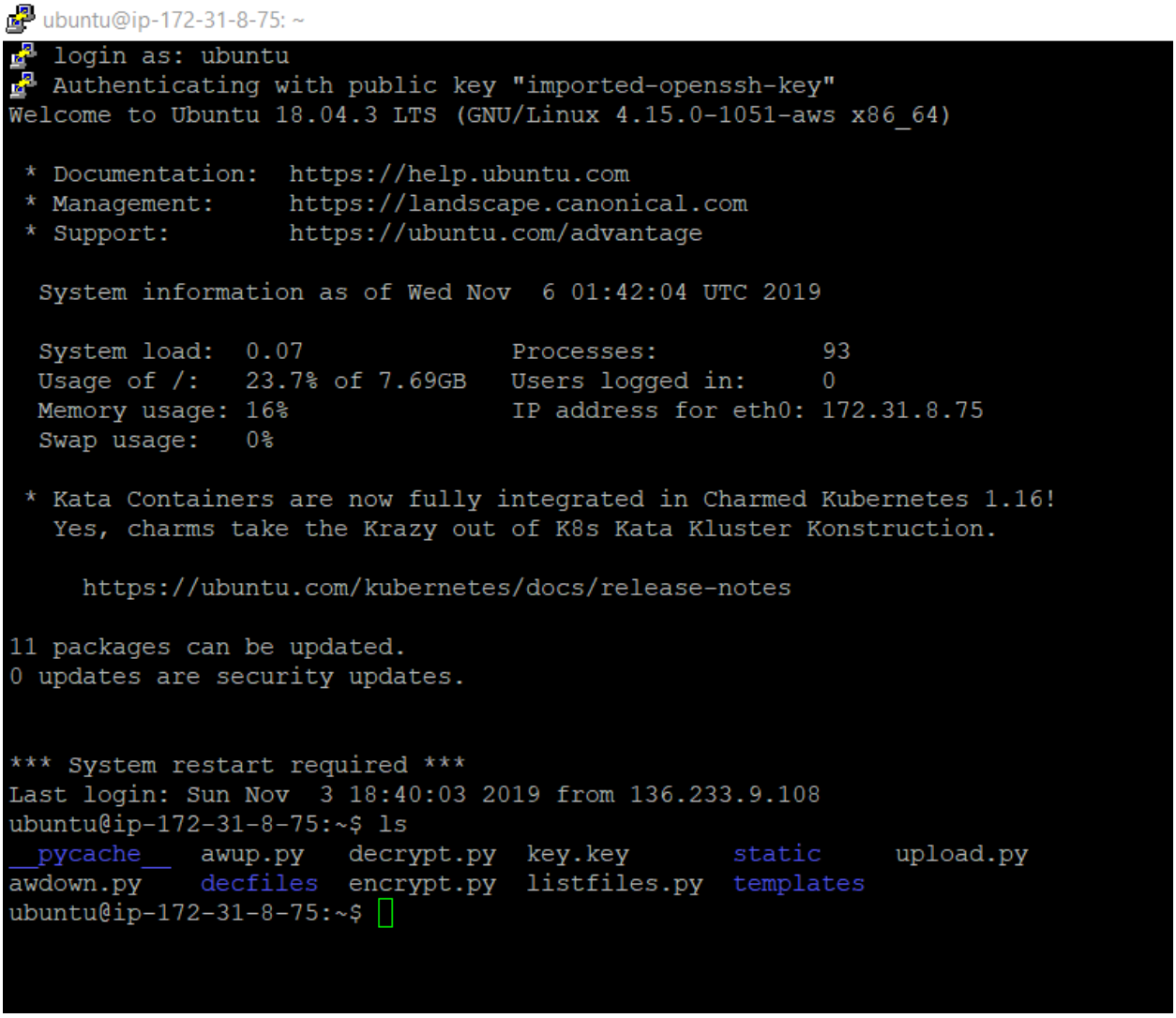
- Configure the ec2 instance settings to serve to the HTTP in the inbound section of the instance.



- Upload the files on Aws ec2 instance using the WinScp file transfer program.



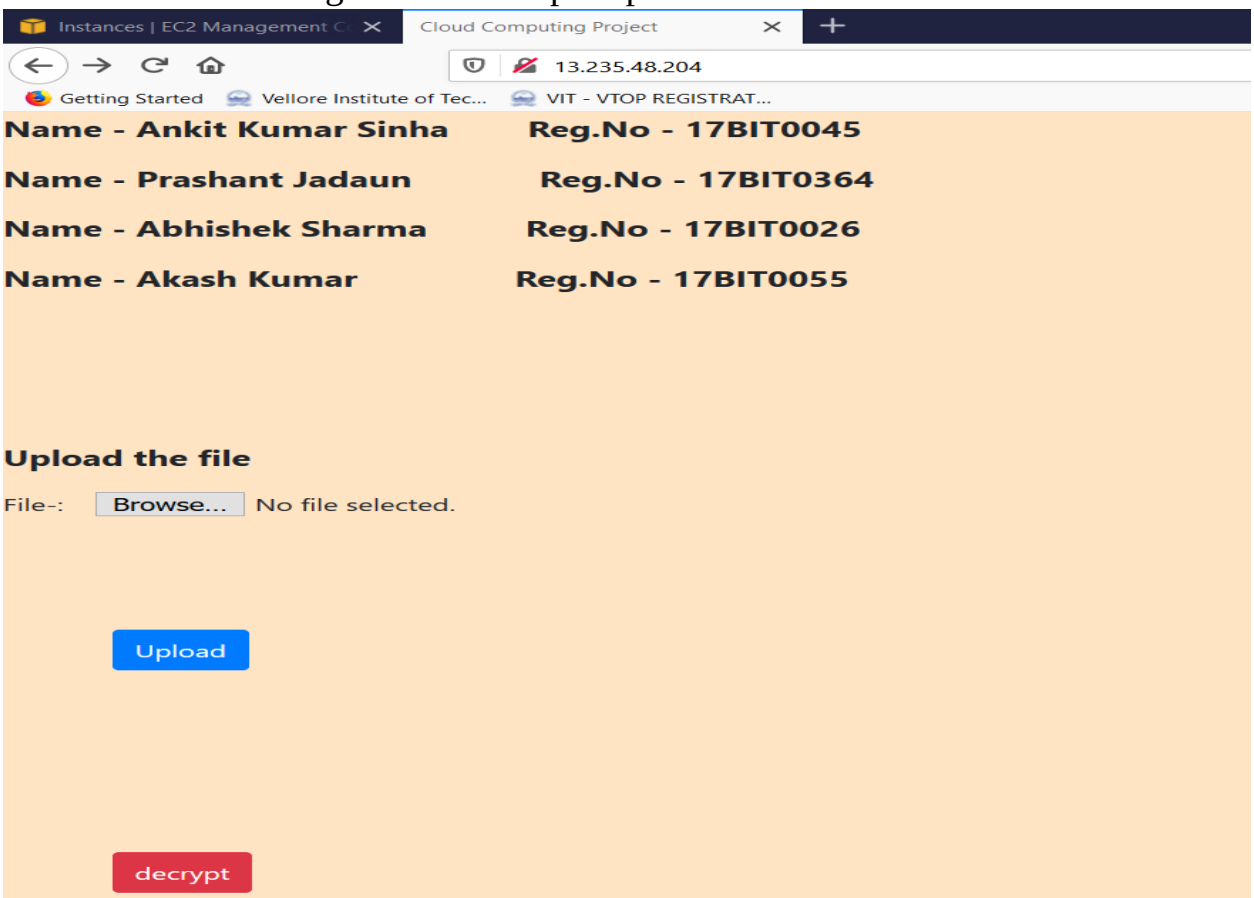
- Access the ec2 instance terminal on our system using putty



- Start the HTTP server.

```
*** System restart required ***
Last login: Sun Nov  3 18:40:03 2019 from 136.233.9.108
ubuntu@ip-172-31-8-75:~$ ls
__pycache__   awup.py   decrypt.py  key.key       static      upload.py
awdown.py     decfiles  encrypt.py  listfiles.py  templates
ubuntu@ip-172-31-8-75:~$ sudo python3 upload.py
 * Serving Flask app "upload" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 269-862-025
136.233.9.108 - - [06/Nov/2019 01:42:54] "GET / HTTP/1.1" 200 -
136.233.9.108 - - [06/Nov/2019 01:42:54] "GET /static/bootstrap.css HTTP/1.1" 200 -
136.233.9.108 - - [06/Nov/2019 01:42:55] "GET /favicon.ico HTTP/1.1" 404 -
136.233.9.108 - - [06/Nov/2019 01:43:03] "GET /dect HTTP/1.1" 200 -
```

- Visit the website using the instance ipv4 ip address.



- Display the files already uploaded.

Python code

For flask server

```
from encrypt import encrypt
from decrypt import decrypted
from awup import fileonaws
from awdown import filedownaws
from listfiles import oblist
from flask import *
app = Flask(__name__)
@app.route('/')
def home():
    return render_template('main.html')



@app.route('/success', methods = ['POST'])
def success():
    if request.method == 'POST':
        f = request.files['file']
```

```
        f.save(f.filename)
        encrypt(f.filename)
        fileonaws(f.filename)
        return render_template("success.html", name = f.filename)


@app.route('/dect')
def decrypt():
    z = oblist()
    return render_template('dec.html',z=z)


@app.route('/filedownload',methods = ['POST'])
def filedownload():
    if request.method == 'POST':
        g = request.form['finame']
        filedownaws(g)
        decrypted("decfiles/"+g)
        return send_file(g,attachment_filename="download.txt")


if __name__ == '__main__':
    app.run(host='0.0.0.0', port=80,debug = True)
```

For encryption and decryption

```
from cryptography.fernet import Fernet
def encrypt(input_file):
    file = open('key.key', 'rb')
    key = file.read()
    file.close()
    with open(input_file,'rb') as f:
        data=f.read()
    fernet = Fernet(key)
    encrypted = fernet.encrypt(data)
    with open(input_file,'wb') as f:
        f.write(encrypted)

from cryptography.fernet import Fernet
def decrypted(output_file):
    file = open('key.key', 'rb')
    key = file.read()
    file.close()
    with open(output_file,'rb') as f:
        data=f.read()
    fernet = Fernet(key)
    decryptedfile = fernet.decrypt(data)
```

```
    output_file=output_file.split("/")[1]
    with open(output_file,'wb') as f:
        f.write(decryptedfile)
```

For uploading the encrypted file on S3 programmatically

```
import boto3
from botocore.client import Config
ACCESS_KEY_ID = 'AKIA5AQOIK5R5AIOSXXY'
ACCESS_SECRET_KEY = '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'
BUCKET_NAME = 'cloud1234567890'
def fileonaws(file_name):
    data = open(file_name, 'rb')
    s3 = boto3.resource(

        's3',

        aws_access_key_id=ACCESS_KEY_ID,

        aws_secret_access_key=ACCESS_SECRET_KEY,

        config=Config(signature_version='s3v4')

    )
    s3.Bucket(BUCKET_NAME).put_object(Key=file_name, Body=data)
```

For downloading the decrypted files from s3

```
import boto3
from botocore.client import Config
ACCESS_KEY_ID = 'AKIA5AQOIK5R5AIOSXXY'
ACCESS_SECRET_KEY = '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'
BUCKET_NAME = 'cloud1234567890'
def filedownaws(FILE_NAME):
    s3 = boto3.resource(

        's3',

        aws_access_key_id=ACCESS_KEY_ID,

        aws_secret_access_key=ACCESS_SECRET_KEY,

        config=Config(signature_version='s3v4')
```

```
    )
    d = './decfiles/'+FILE_NAME
    s3.Bucket(BUCKET_NAME).download_file(FILE_NAME,d)


#cloud1234567890
```

For getting the list of files available on S3

```
import boto3
from botocore.client import Config
ACCESS_KEY_ID = 'AKIA5AQOIK5R5AIOSXXY'
ACCESS_SECRET_KEY = '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'
BUCKET_NAME = 'cloud1234567890'
def oblist():
    s3 = boto3.resource(

        's3',

        aws_access_key_id=ACCESS_KEY_ID,

        aws_secret_access_key=ACCESS_SECRET_KEY,

        config=Config(signature_version='s3v4')

    )
    listObjSummary = s3.Bucket(BUCKET_NAME).objects.all()
    return listObjSummary
```

HTML CODES OF DIFFERENT WEBPAGES

```
<!DOCTYPE html>
<html>
    <head>
        <title>Cloud Computing Project</title>
        <style>
            p{
                font-weight: bold;
                font-size: 20px;
            }
        </style>
        <link rel="stylesheet" href="{{ url_for('static', filename='bootstrap.css
') }}">
    </head>
    <body style="background-color: bisque;">
        <p>Name - Ankit Kumar Sinha        Reg.No -
 17BIT0045</p>
        <p>Name - Prashant Jadaun         &nbs
p;   Reg.No - 17BIT0364</p>
        <p>Name - Abhishek Sharma         &nbs
p;Reg.No - 17BIT0026</p>
        <p>Name - Akash Kumar          &n
bsp;     Reg.No - 17BIT0055</p>
        <br><br><br><br><br>
        <p> Upload the file</p>
        <form action="/success" method="POST" enctype="multipart/form-data">
            <label>File-
:    <input type="file" name="file"></label>
            <br><br><br><br><br>
                       &nb
sp;  <input type="submit" value="Upload" class="btn btn-primary">
        </form>
        <br><br><br><br><br><br><br>
                    &
nbsp; <a href="/dect"><button class="btn btn-danger">decrypt</button></a>
    </body>
</html>
```

```
<html>
<head>
<title>Cloud Computing Project</title>
<link rel="stylesheet" href="{{ url_for('static', filename='bootstrap.css') }}">
```

```
</head>
<body style="background-color: bisque;">
<h2>File uploaded successfully</h2>
<h2>File Name: {{name}}</h2>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
    <head>
        <title>Cloud Computing Project</title>
        <style>
            p{
                font-weight: bold;
                font-size: 20px;
            }
        </style>
        <link rel="stylesheet" href="{{ url_for('static', filename='bootstrap.css
') }}">
    </head>
    <body style="background-color: bisque;">
        <p>Name - Ankit Kumar Sinha        Reg.No -
 17BIT0045</p>
        <p>Name - Prashant Jadaun         &nbs
p;   Reg.No - 17BIT0364</p>
        <p>Name - Abhishek Sharma         &nbs
p;Reg.No - 17BIT0026</p>
        <p>Name - Akash Kumar          &n
bsp;     Reg.No - 17BIT0055</p>
        <br><br><br><br><br>
        <form action="/filedownload" method="POST" enctype="multipart/form-data">
            {%for i in z%}
                   <label><input type="radio" value={{i.key}} name="finame">{{i.k
ey}}</label>
            {%endfor%}
            <br><br><br><br><br>
                       &nb
sp;  <input type="submit" value="Decrypt" class="btn btn-danger">
        </form>
    </body>
</html>
```

```
<!DOCTYPE html>
```

```
<html>
    <head>
        <title>Nis Project</title>
        <link rel="stylesheet" href="{{ url_for('static', filename='bootstrap.css
') }}">
    </head>
    <body style="background-color: azure;">
        <h3>Name - Ankit Kumar Sinha     Reg.No -17BIT0045</h3>
        <h3>Name - Sujoy Dey     Reg.No -17BIT0327</h3>
        <h3>Name - Prashant Jadaun     Reg.No -17BIT0364</h3>
        <h3>Name - Abhishek Sharma     Reg.No -17BIT0026</h3>
        <br>
        <h3>Password Saved Sucessfully</h3>
    </body>
    </html>
```