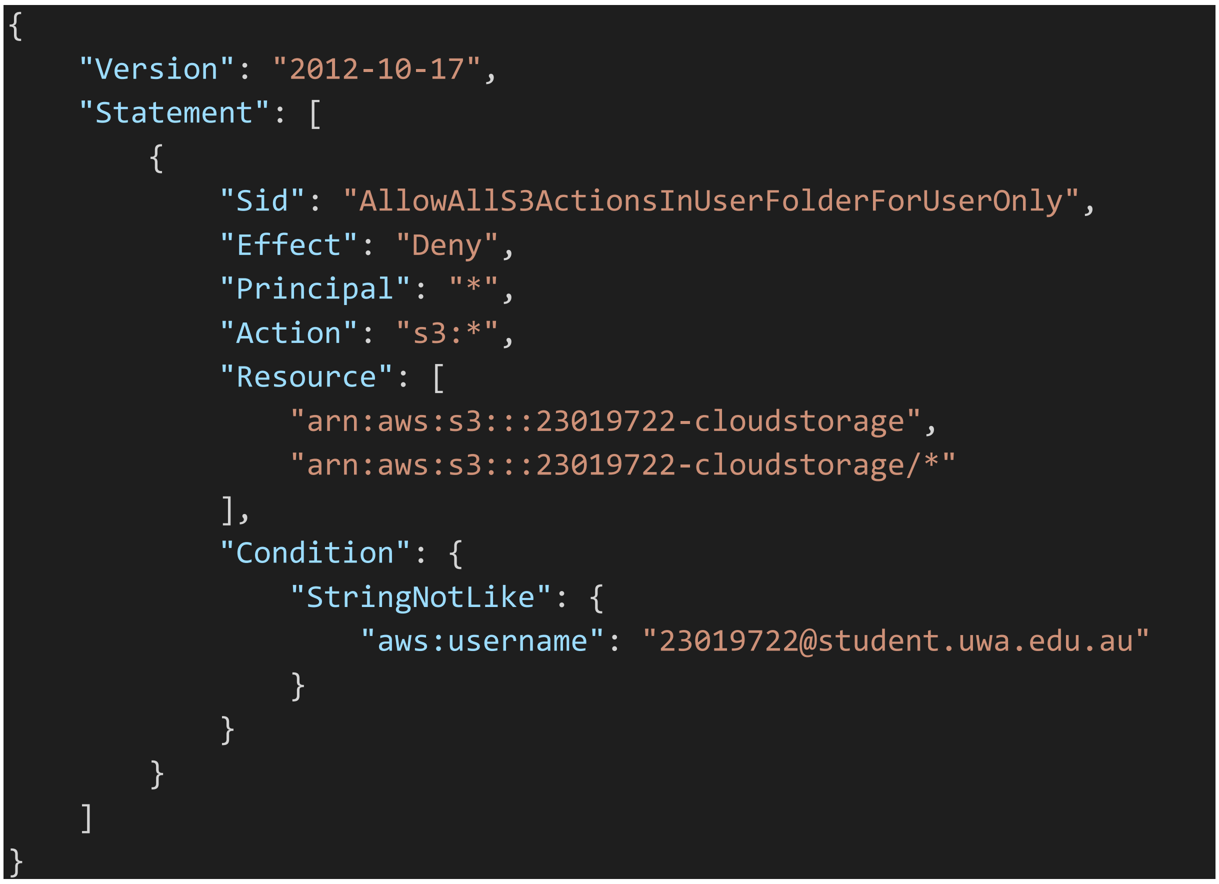
#-------------------
# Lab 4
# Isaac Huang
# 23019722
#-------------------

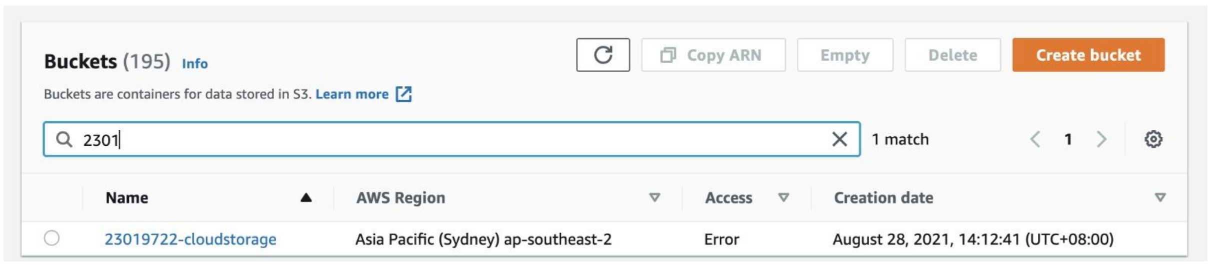
1. Bucket Permissions

1.1 Policy



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllS3ActionsInUserFolderForUserOnly",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::23019722-cloudstorage",
                "arn:aws:s3:::23019722-cloudstorage/*"
            ],
            "Condition": {
                "StringNotLike": {
                    "aws:username": "23019722@student.uwa.edu.au"
                }
            }
        }
    ]
}
```

1.2 Result



Buckets (195) Info

Buckets are containers for data stored in S3. Learn more

Search: 2301 — 1 match

	Name	AWS Region	Access	Creation date
○	23019722-cloudstorage	Asia Pacific (Sydney) ap-southeast-2	Error	August 28, 2021, 14:12:41 (UTC+08:00)

2. KMS

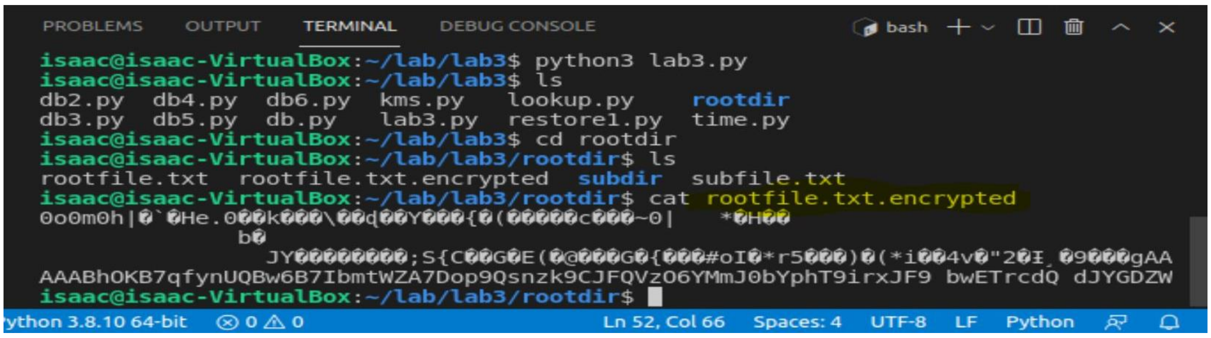
2.1 Encrypt

2.1.1 Script

```
import os
from Crypto.Cipher import AES
from Crypto import Random
import boto3, botocore
from botocore.exceptions import ClientError
import base64
from cryptography.fernet import Fernet

client = boto3.client('s3')
s3 = boto3.resource('s3')
bucket_name = '23019722-cloudstorage'
cmk_id = 'arn:aws:kms:ap-southeast-2:622578507161:key/084f2b57-93ab-
4ed5-8f2a-0163d9daa0f4'
kms_client = boto3.client('kms')
NUM_BYTES_FOR_LEN = 4
ROOT_DIR = '.'

def create_data_key(cmk_id, key_spec="AES_256"):
    response = kms_client.generate_data_key(KeyId=cmk_id, KeySpec=ke
y_spec)
    return response["CiphertextBlob"], base64.b64encode(response["Pl
aintext"])

def encrypt_file(filename, cmk_id):
    with open(filename, "rb") as file:
      file_contents = file.read()
    data_key_encrypted, data_key_plaintext = create_data_key(cmk_id)
    if data_key_encrypted is None:
        return
    f = Fernet(data_key_plaintext)
    file_contents_encrypted = f.encrypt(file_contents)
    with open(filename + '.encrypted', 'wb') as file_encrypted:
        file_encrypted.write(len(data_key_encrypted).to_bytes(NUM_BY
TES_FOR_LEN,
                                                              byteor
der='big'))
        file_encrypted.write(data_key_encrypted)
```

```python
        file_encrypted.write(file_contents_encrypted)

def upload_file(file):
    client.upload_file(file,bucket_name,file[2:])
    print("Uploading %s" % file)

for dir_name, subdir_list, file_list in os.walk(ROOT_DIR, topdown=Tr
ue):
    if dir_name != ROOT_DIR:
        for fname in file_list:
            encrypt_file(fname, cmk_id)
            file = os.path.join(dir_name, fname) + '.encrypted'
            upload_file(file)
```

2.1.2 Result



```
isaac@isaac-VirtualBox:~/lab/lab3$ python3 lab3.py
isaac@isaac-VirtualBox:~/lab/lab3$ ls
db2.py  db4.py  db6.py  kms.py   lookup.py    rootdir
db3.py  db5.py  db.py   lab3.py  restore1.py  time.py
isaac@isaac-VirtualBox:~/lab/lab3$ cd rootdir
isaac@isaac-VirtualBox:~/lab/lab3/rootdir$ ls
rootfile.txt  rootfile.txt.encrypted  subdir  subfile.txt
isaac@isaac-VirtualBox:~/lab/lab3/rootdir$ cat rootfile.txt.encrypted
0o0m0h|�`�He.0��k���\��d��Y���{�(�����c���~0|    *�H��
           b�
             JY���������;S{C��G�E(�@���G�{���#oI�*r5���)�(*i��4v�"2�Ɪ ��9���gAA
AAABhOKB7qfynUQBw6B7IbmtWZA7Dop9Qsnzk9CJFQVzO6YMmJ0bYphT9irxJF9 bwETrcdQ dJYGDZW
isaac@isaac-VirtualBox:~/lab/lab3/rootdir$ 
```

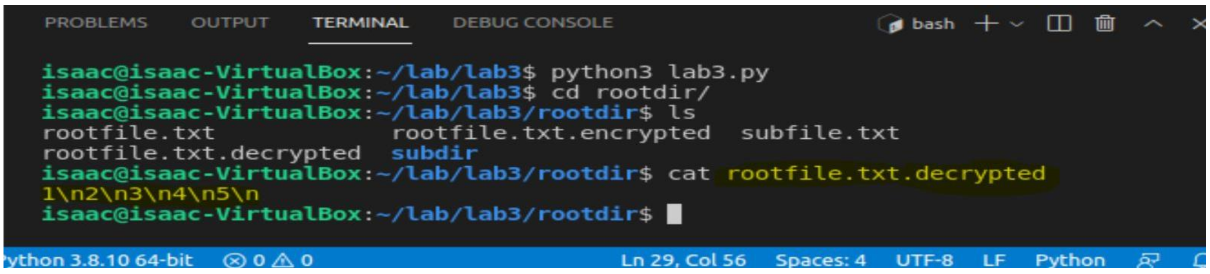
2.2 Decrypt

2.2.1 Script

```python
import os
import errno
import boto3
import errno
import base64
from cryptography.fernet import Fernet

client = boto3.client('s3')
s3 = boto3.resource('s3')
bucket_name = '23019722-cloudstorage'
cmk_id = 'arn:aws:kms:ap-southeast-2:622578507161:key/084f2b57-93ab-
4ed5-8f2a-0163d9daa0f4'
kms_client = boto3.client('kms')
NUM_BYTES_FOR_LEN = 4
ROOT_DIR = '.'
```

```python
def decrypt_data_key(data_key_encrypted):
    kms_client = boto3.client("kms")
    response = kms_client.decrypt(CiphertextBlob=data_key_encrypted)
    return base64.b64encode((response["Plaintext"]))

def decrypt_file(filename):
    with open(filename + ".encrypted", "rb") as file:
      file_contents = file.read()
    data_key_encrypted_len = int.from_bytes(file_contents[:NUM_BYTES
_FOR_LEN],
                                            byteorder="big") \
                             + NUM_BYTES_FOR_LEN
    data_key_encrypted = file_contents[NUM_BYTES_FOR_LEN:data_key_en
crypted_len]
    data_key_plaintext = decrypt_data_key(data_key_encrypted)
    if data_key_plaintext is None:
        return False
    f = Fernet(data_key_plaintext)
    file_contents_decrypted = f.decrypt(file_contents[data_key_encry
pted_len:])
    with open(filename + '.decrypted', 'wb') as file_decrypted:
      file_decrypted.write(file_contents_decrypted)

def assert_dir_exists(path):
    try:
        os.makedirs(path)
    except OSError as e:
        if e.errno != errno.EEXIST:
            raise

def download_dir(bucket, path, target):
    # Handle missing / at end of prefix
    if not path.endswith('/'):
        path += '/'
    paginator = client.get_paginator('list_objects_v2')
    for result in paginator.paginate(Bucket=bucket, Prefix=path):
        # Download each file individually
        for key in result['Contents']:
            # Calculate relative path
            rel_path = key['Key'][len(path):]
            if not key['Key'].endswith('/'):
                local_file_path = os.path.join(target, rel_path)
                local_file_dir = os.path.dirname(local_file_path)
                assert_dir_exists(local_file_dir)
                client.download_file(bucket, key['Key'], local_file_
path)

download_dir(bucket_name, 'rootdir/', '/home/isaac/lab/lab3/rootdir/
')

for dir_name, subdir_list, file_list in os.walk(ROOT_DIR, topdown=Tr
ue):
    if dir_name != ROOT_DIR:
        for fname in file_list:
            decrypt_file(fname)
```

2.2.2 Result



```
isaac@isaac-VirtualBox:~/lab/lab3$ python3 lab3.py
isaac@isaac-VirtualBox:~/lab/lab3$ cd rootdir/
isaac@isaac-VirtualBox:~/lab/lab3/rootdir$ ls
rootfile.txt              rootfile.txt.encrypted   subfile.txt
rootfile.txt.decrypted    subdir
isaac@isaac-VirtualBox:~/lab/lab3/rootdir$ cat rootfile.txt.decrypted
1\n2\n3\n4\n5\n
isaac@isaac-VirtualBox:~/lab/lab3/rootdir$
```

3. AES

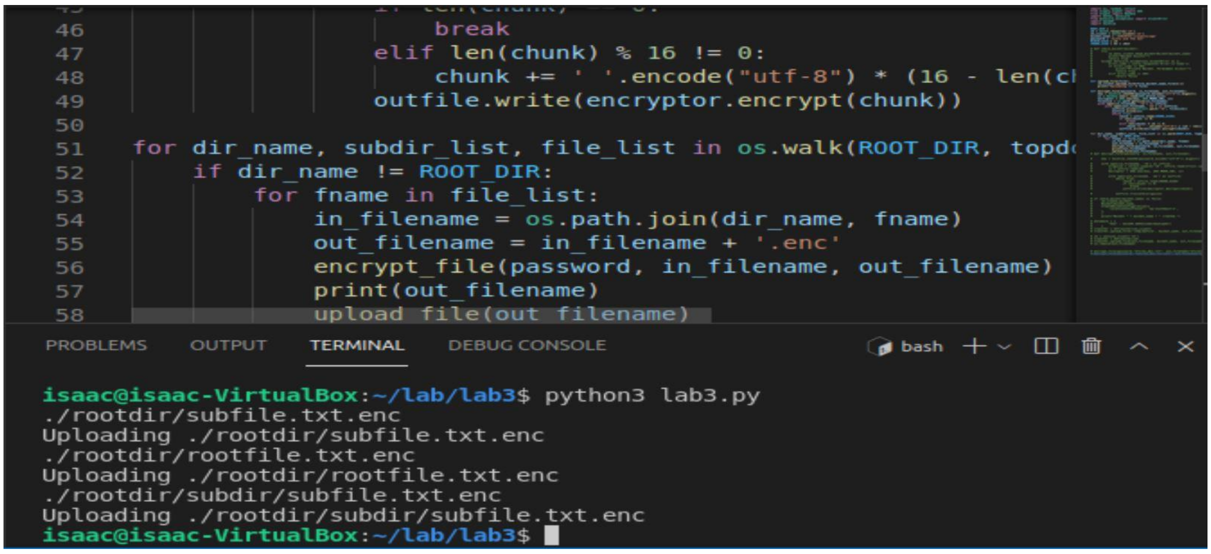
3.1 Encrypt

3.1.1 Script

```
import os, struct
from Crypto.Cipher import AES
from Crypto import Random
import boto3, botocore
from botocore.exceptions import ClientError
import hashlib

ROOT_DIR = '.'
s3 = boto3.resource('s3')
s3_client = boto3.client('s3')
bucket_name = '23019722-cloudstorage'
password = 'kitty and the kat'
BLOCK_SIZE = 16
```

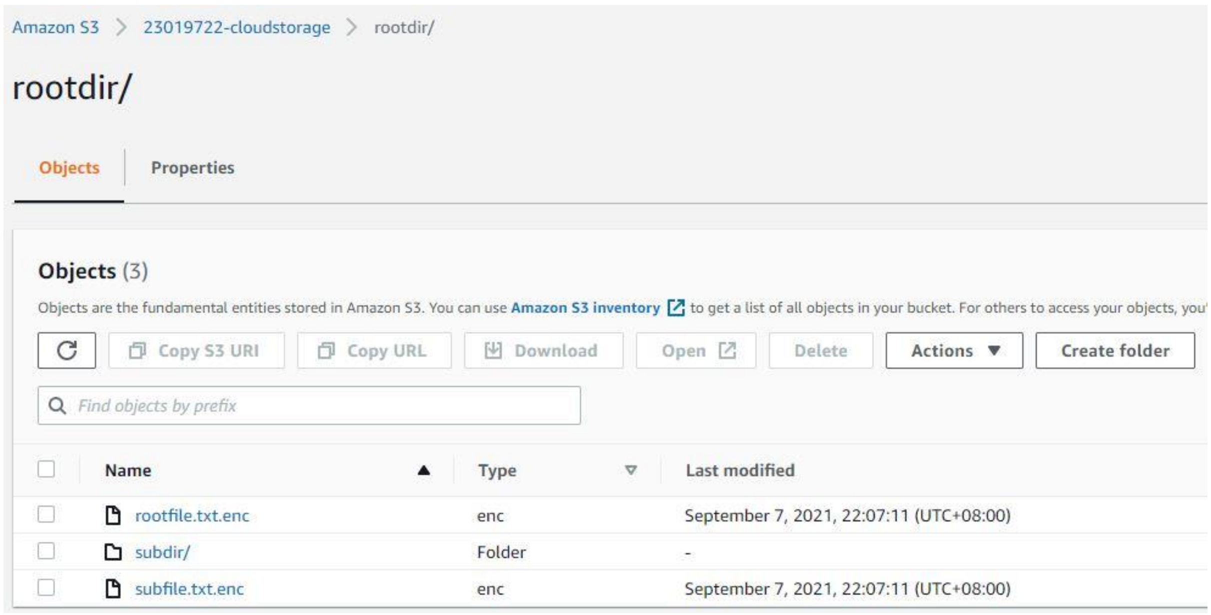
```python
CHUNK_SIZE = 64 * 1024

def upload_file(file):
    s3_client.upload_file(file,bucket_name,file[2:])
    print("Uploading %s" % file)

def encrypt_file(password, in_filename, out_filename):
    key = hashlib.sha256(password.encode("utf-8")).digest()
    iv = Random.new().read(AES.block_size)
    encryptor = AES.new(key, AES.MODE_CBC, iv)
    filesize = os.path.getsize(in_filename)
    with open(in_filename, 'rb') as infile:
        with open(out_filename, 'wb') as outfile:
            outfile.write(struct.pack('<Q', filesize))
            outfile.write(iv)
            while True:
                chunk = infile.read(CHUNK_SIZE)
                if len(chunk) == 0:
                    break
                elif len(chunk) % 16 != 0:
                    chunk += ' '.encode("utf-
8") * (16 - len(chunk) % 16)
                outfile.write(encryptor.encrypt(chunk))

for dir_name, subdir_list, file_list in os.walk(ROOT_DIR, topdown=Tr
ue):
    if dir_name != ROOT_DIR:
        for fname in file_list:
            in_filename = os.path.join(dir_name, fname)
            out_filename = in_filename + '.enc'
            encrypt_file(password, in_filename, out_filename)
            print(out_filename)
            upload_file(out_filename)
```

3.1.2 Result



```
                if len(chunk) == 0:
                    break
                elif len(chunk) % 16 != 0:
                    chunk += ' '.encode("utf-8") * (16 - len(cl
                outfile.write(encryptor.encrypt(chunk))

for dir_name, subdir_list, file_list in os.walk(ROOT_DIR, topdo
    if dir_name != ROOT_DIR:
        for fname in file_list:
            in_filename = os.path.join(dir_name, fname)
            out_filename = in_filename + '.enc'
            encrypt_file(password, in_filename, out_filename)
            print(out_filename)
            upload_file(out_filename)
```

```
isaac@isaac-VirtualBox:~/lab/lab3$ python3 lab3.py
./rootdir/subfile.txt.enc
Uploading ./rootdir/subfile.txt.enc
./rootdir/rootfile.txt.enc
Uploading ./rootdir/rootfile.txt.enc
./rootdir/subdir/subfile.txt.enc
Uploading ./rootdir/subdir/subfile.txt.enc
isaac@isaac-VirtualBox:~/lab/lab3$ 
```



Objects (3)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'

Name	Type	Last modified
rootfile.txt.enc	enc	September 7, 2021, 22:07:11 (UTC+08:00)
subdir/	Folder	-
subfile.txt.enc	enc	September 7, 2021, 22:07:11 (UTC+08:00)

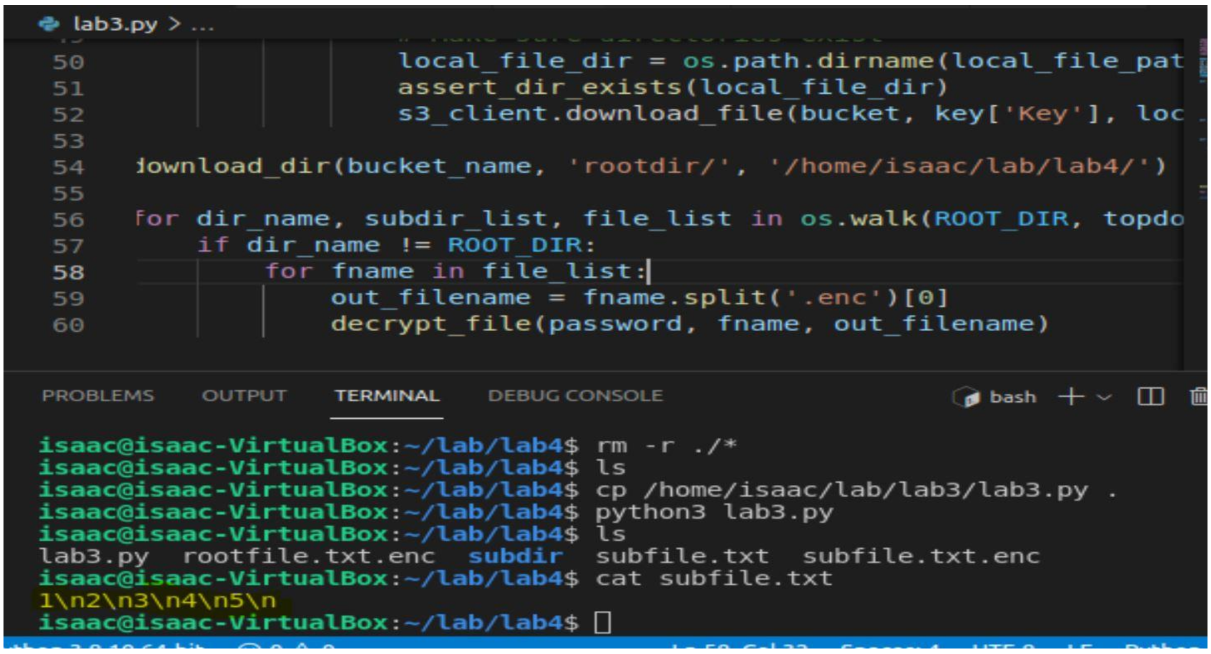
3.2 Decrypt

3.2.1 Script

```
import os, errno, struct
from Crypto.Cipher import AES
from Crypto import Random
import boto3, botocore
from botocore.exceptions import ClientError
import hashlib

ROOT_DIR = '.'
s3 = boto3.resource('s3')
```

```python
s3_client = boto3.client('s3')
bucket_name = '23019722-cloudstorage'
password = 'kitty and the kat'
BLOCK_SIZE = 16
CHUNK_SIZE = 64 * 1024

def decrypt_file(password, in_filename, out_filename):
    key = hashlib.sha256(password.encode("utf-8")).digest()
    with open(in_filename, 'rb') as infile:
        origsize = struct.unpack('<Q', infile.read(struct.calcsize('
Q')))[0]
        iv = infile.read(16)
        decryptor = AES.new(key, AES.MODE_CBC, iv)
        with open(out_filename, 'wb') as outfile:
            while True:
                chunk = infile.read(CHUNK_SIZE)
                if len(chunk) == 0:
                    break
                outfile.write(decryptor.decrypt(chunk))
            outfile.truncate(origsize)

def assert_dir_exists(path):
    try:
        os.makedirs(path)
    except OSError as e:
        if e.errno != errno.EEXIST:
            raise

def download_dir(bucket, path, target):
    if not path.endswith('/'):
        path += '/'
    paginator = s3_client.get_paginator('list_objects_v2')
    for result in paginator.paginate(Bucket=bucket, Prefix=path):
        # Download each file individually
        for key in result['Contents']:
            # Calculate relative path
            rel_path = key['Key'][len(path):]
            if not key['Key'].endswith('/'):
                local_file_path = os.path.join(target, rel_path)
                local_file_dir = os.path.dirname(local_file_path)
                assert_dir_exists(local_file_dir)
                s3_client.download_file(bucket, key['Key'], local_fi
le_path)

download_dir(bucket_name, 'rootdir/', '/home/isaac/lab/lab4/')

for dir_name, subdir_list, file_list in os.walk(ROOT_DIR, topdown=Tr
ue):
    if dir_name != ROOT_DIR:
        for fname in file_list:
            out_filename = fname.split('.enc')[0]
            decrypt_file(password, fname, out_filename)
```

3.2.2 Result



```
                local_file_dir = os.path.dirname(local_file_pat
                assert_dir_exists(local_file_dir)
                s3_client.download_file(bucket, key['Key'], loc

download_dir(bucket_name, 'rootdir/', '/home/isaac/lab/lab4/')

for dir_name, subdir_list, file_list in os.walk(ROOT_DIR, topdo
    if dir_name != ROOT_DIR:
        for fname in file_list:
            out_filename = fname.split('.enc')[0]
            decrypt_file(password, fname, out_filename)
```

```
isaac@isaac-VirtualBox:~/lab/lab4$ rm -r ./*
isaac@isaac-VirtualBox:~/lab/lab4$ ls
isaac@isaac-VirtualBox:~/lab/lab4$ cp /home/isaac/lab/lab3/lab3.py .
isaac@isaac-VirtualBox:~/lab/lab4$ python3 lab3.py
isaac@isaac-VirtualBox:~/lab/lab4$ ls
lab3.py  rootfile.txt.enc  subdir  subfile.txt  subfile.txt.enc
isaac@isaac-VirtualBox:~/lab/lab4$ cat subfile.txt
1\n2\n3\n4\n5\n
isaac@isaac-VirtualBox:~/lab/lab4$ 
```