Secure File Storage Using Hybrid Cryptography

A COURSE PROJECT REPORT

Submitted By

Yashu Youwaraj (RA2011029010062)
Kishlay Shekhar (RA2011029010008)
Aryan Pandey (RA2011029010019)

Under the guidance of

Dr.P.VISALAKSHI

(Asst.Professor(Selection Grade))

Department of Network and communications

In partial fulfilment for the Course

in

18CSE378T- Principles of Cloud Computing

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Chenpalpattu District

NOVEMBER 2022

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this project report titled "Secure File Storage Using Hybrid Cryptography " is the bonafide work of "YASHU YOUWARAJ (RA2011029010062), KISHLAY SHEKHAR(RA2011029010008), ARYAN PANDEY (RA2011029010018) " who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported here in does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Mrs. P. Visalakshi
Guide
Asst. Professor(Section Grade)
Networking and Communications

SIGNATURE

HOD(NWC)

Signature of the Internal Examiner

SRM Institute of Science and Technology

ACKNOWLEDGEMENT

We express our heartfelt thanks to our honorable Vice Chancellor

Dr. C. MUTHAMIZHCHELVAN, for being the beacon in all our endeavors.

We would like to express my warmth of gratitude to our Registrar

Dr. S. Ponnusamy, for his encouragement for the course project

We express our profound gratitude to our Dean (College of Engineering and Technology) Dr. T. V.Gopal, for bringing out novelty in all executions.

We would like to express my heartfelt thanks to Chairperson, School of Computing Dr. Revathi Venkataraman, for imparting confidence to complete my course project

We are highly thankful to our Course project Faculty Dr.P Visalakshi , Asst Professor(Section Grade) , Networking and Communications , for her assistance,timely suggestion and

guidance throughout the duration of this course project.

We extend my gratitude to our HoD,Mrs. Annapurani Panaiyappan,Head of Department, NWC(Networking and Communications) and my Departmental colleagues for their Support.

Finally, we thank our parents and friends near and dear ones who directly and

indirectly contributed to the successful completion of our project.

Above all, I thank the almighty for showering his blessings on me to complete my Course project.

Yashu Youwaraj

Kishlay Shekhar

Aryan Pandey

TABLE OF CONTENTS

1.ABSTRACT

Nowadays the growing use of mobile devices and advancement in networking technology is leading us to secure file storage over the network. Cryptography is the most popular technology used for all types of data security. This discussed paper is a broad survey of the different approach which is used for securely storing files, and sharing it over the network. This proposed scheme will also ensure the whole model to have confidentiality, integrity, and availability mechanisms to be implemented in it. With the increasing reliance on cloud storage by many organizations and entities to keep the most important and confidential data, concerns are growing about the security strength of cloud storage against various threats. This paper is mainly discussing the security of cloud computing and cloud storage using hybrid cryptosystems. An analysis will be performed to compare various hybrid cryptosystems proposed for cloud storage.

2.INTRODUCTION

The aim of the project is to create an encrypted and secured file storage system to transfer files within users in a remote location. This system will require an input that is successfully encrypted using any of the algorithm techniques and store them anywhere. The uploaded file can be downloaded by other users, but to read the data present in it, they have to decrypt the file using the decryption algorithm and the information provided about the file within the users by the owner. The system uses public-key cryptographic techniques like RSA and Symmetric key cryptography like AES. Hashing techniques like static hashing and dynamic hashing are used for performing integrity. Due to the encryption of data, confidentiality is also achieved in the process. The project is also open to new challenges and future changes to other advanced technologies in keeping the data secured.

3.LITERATURE SURVEY

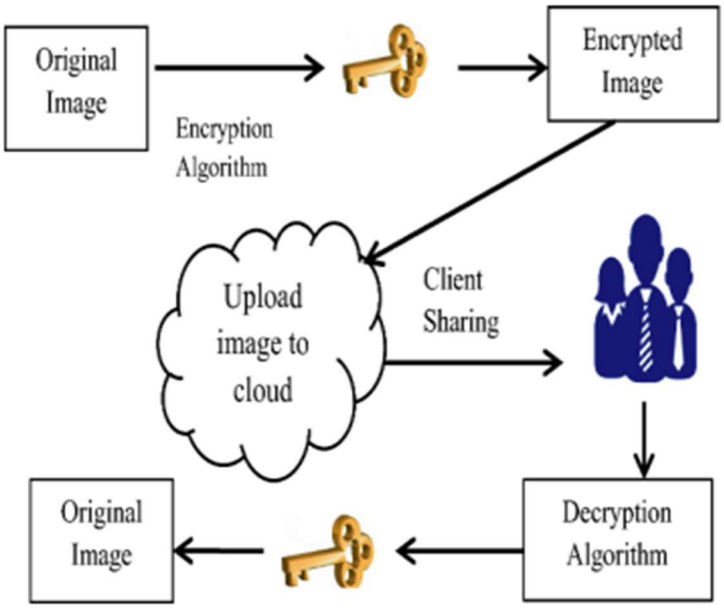
A literature review is nothing but an objective, aim, or summary of whatever research has done relevant to a certain topic. The following published articles have been referred to create a base for my project. Following are some papers been referred to:- Secure file storage in the cloud using Hybrid Cryptography; [1] Author - Punam V. Maitri, Aruna Verma, Year – 2016 Description – The paper focuses on how files are securely stored on a cloud platform. Also, it discusses the problem of using only a single algorithm to encrypt the file and how ineffective it will be on the cloud. This paper splits the file into blocks and each block is encrypted using AES, blowfish, RC6algorithm. The key information about which file uses which algorithm is sent to the receiver using steganography modern approach to file system integrity checking [2] Author – M. Malarvizhi, J. Angela JennifaSujana, T. Revathi, Year – 2014 Description - The main focus of the paper is on the integrity of files and restoring the files if integrity is violated. The proposed system uses a pattern of each protected file to determine its modification. The method used for pattern generation is cryptographic hash functions. The system also uses a database that stores the files that need to be protected and their hash codes. To check the integrity of the file the hash code of the file is produced and checked with one in the database. If the file is successfully tested positively then access is granted otherwise the administrator gets alerted and if it is saved copy is available of the same file then the file is restored.

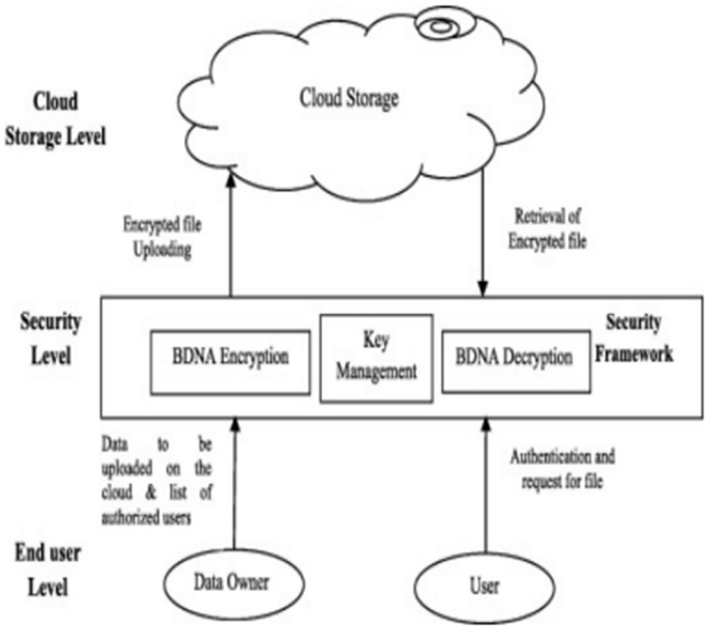
4.Innovation idea

Here are some of my ideas for this like we can introduce three encryption algorithms which are AES, DES and RC6 for encrypting the file system because of not one but three encryption algorithm this will give extra protection to our files. We will also introduce key to access the system. The key can be anything a physical pen drive or a key in which it embeds the key in image using LSB. This will make the unauthorized access to the system not possible easily and makes the system more and more secure and robust in nature. We will also allow access the file when it is online for real time checking the authority of the person.

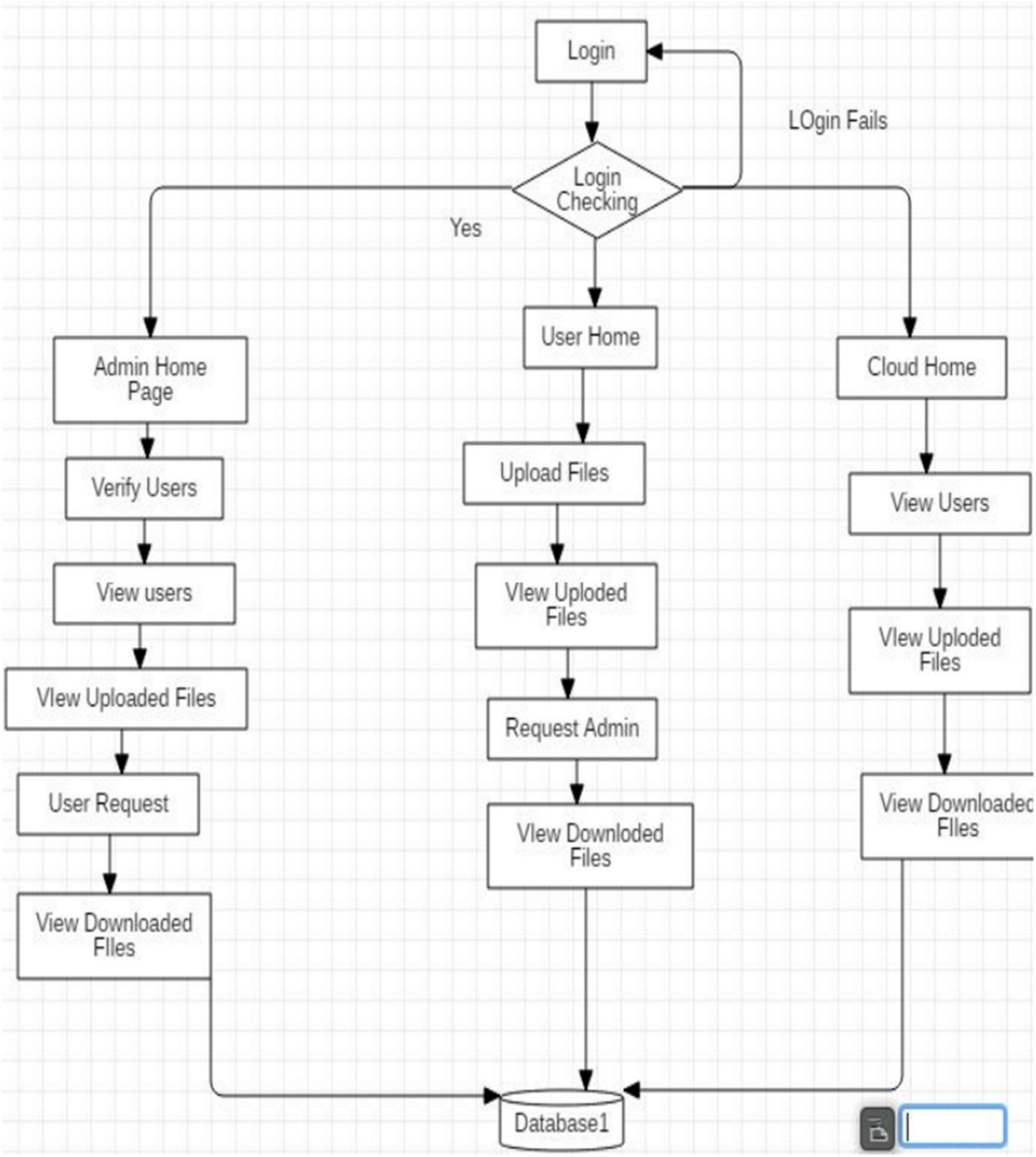
5.Architecture Diagram

A conceptual framework shows the three stage of data input, process and output. Input it represents the data as input to the system that includes: student information, student transcript, and types of appeal, schedule, exam questionnaires and breakdown marks. This data represents different dataset in the system process. it pertains to the particularly viable for application to information bases in such areas as the board handle that comprise of few sections comparing various organizations





6.Data Flow Diagram



7.Technology Used

The technology used in this Project are:

1. Advanced Encryption Standard (AES) ~> The AES algorithm is related to Rijndael`s encryption. Rijndael is a family of encryption algorithms with different keys and block sizes. It consists of a continue serial operations, some of them involve the input of certain outputs (substitutions) and others the mixing of bits (permutations).

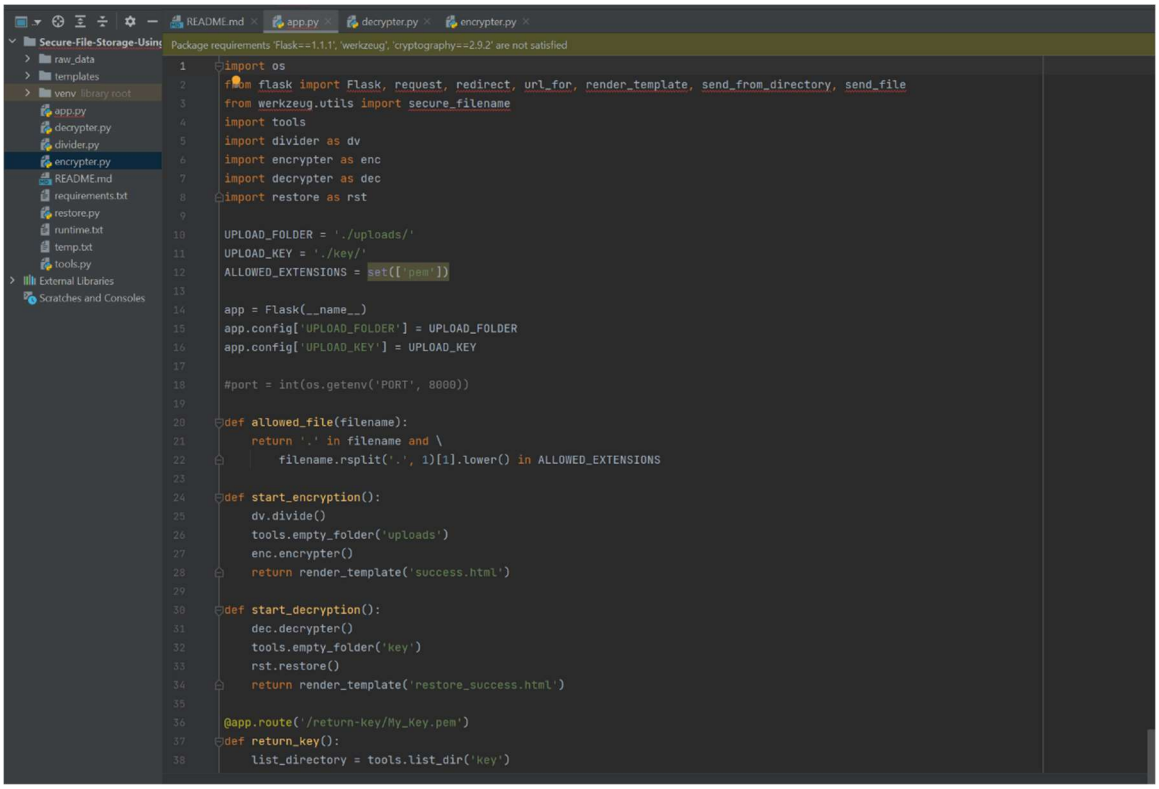
2. Triple Data Encryption Standard (3DES) ~> In cryptography, 3DES is an inherited enhanced version of DES (Data Encryption Standard). In the Triple DES algorithm, DES is used trice to increase the security level. Triple DES is also referred to as TDES or Triple Data Encryption Algorithm (TDEA).

3. Rivest Cipher 6 (RC6) ~> RC6 is a symmetric key block cipher. RC6 (Rivest Cipher 6) is an enhanced version of the old RC5 algorithm. RC6 – w/r/b means that four w-bit-word plaintexts are encrypted with r-rounds by b-bytes keys. It is a proprietary algorithm patented by RSA Security.

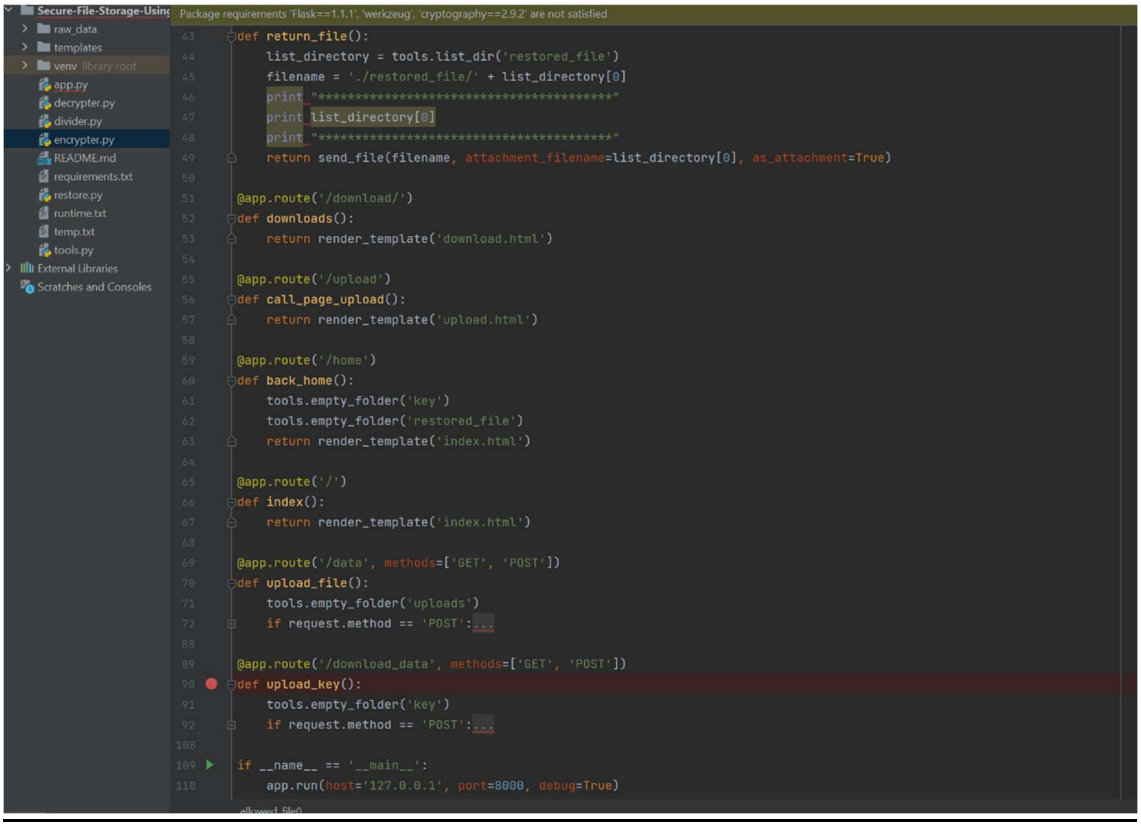
4. Blowfish ~> Blowfish is a symmetric block cipher which uses a Fiestal network, 16 rounds of iterative encryption and decryption functional design. The block size used is of 64-bits and key size can vary from any length to 448.Blowfish cipher uses 18 sub arrays each of 32-bit commonly known as P-boxes and four Substitution boxes each of 32-bit, each having 256 entries.

8. CODE

APP:

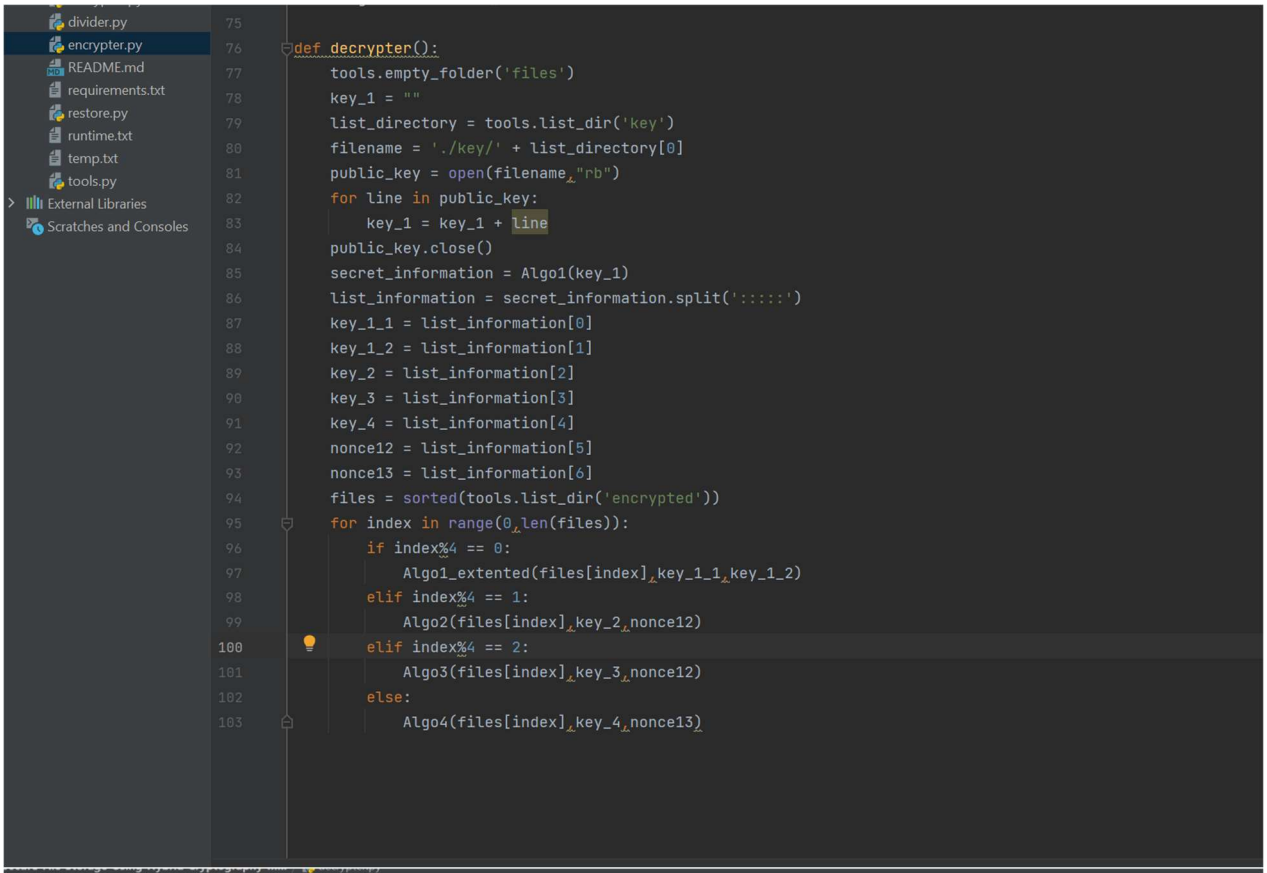


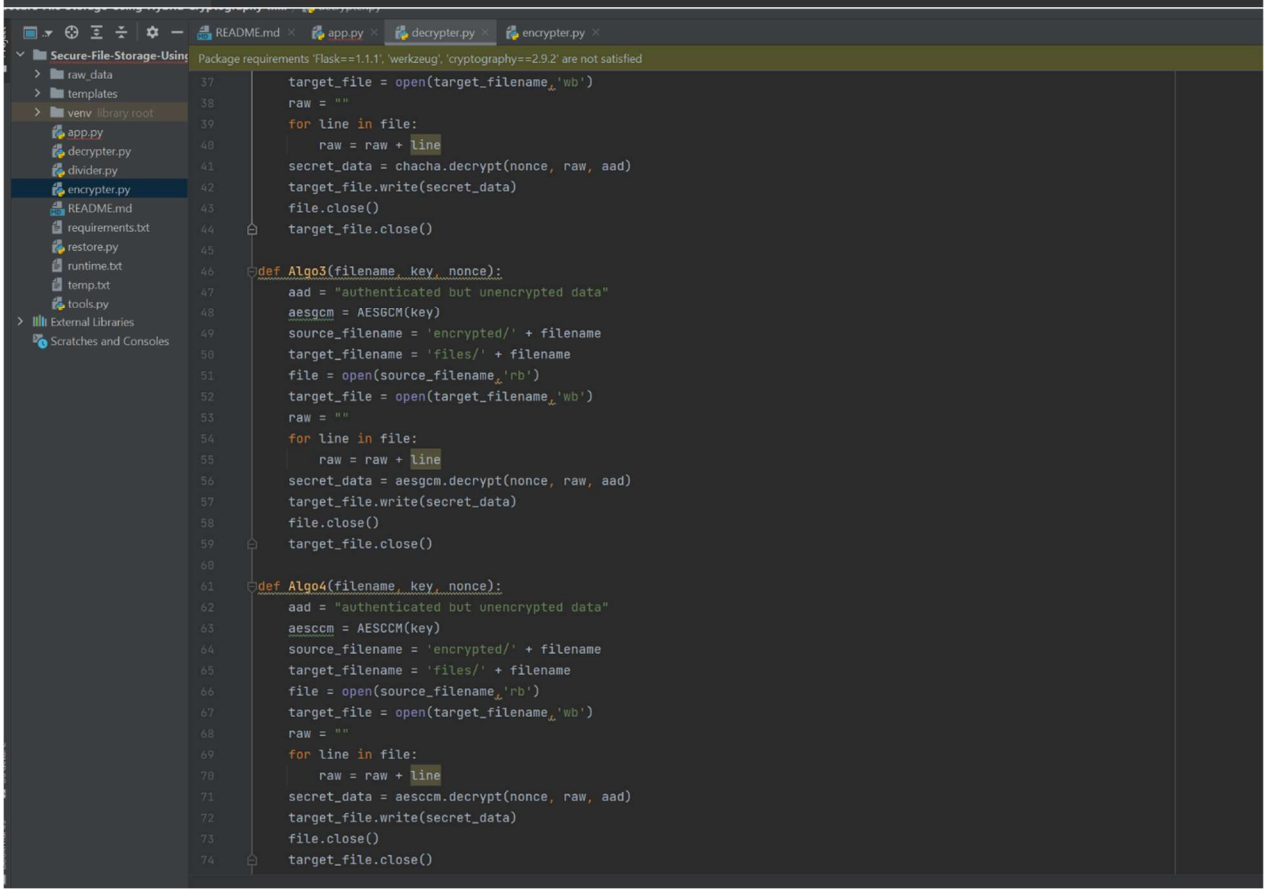
```
import os
from flask import Flask, request, redirect, url_for, render_template, send_from_directory, send_file
from werkzeug.utils import secure_filename
import tools
import divider as dv
import encrypter as enc
import decrypter as dec
import restore as rst

UPLOAD_FOLDER = './uploads/'
UPLOAD_KEY = './key/'
ALLOWED_EXTENSIONS = set(['pem'])

app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
app.config['UPLOAD_KEY'] = UPLOAD_KEY

#port = int(os.getenv('PORT', 8000))

def allowed_file(filename):
    return '.' in filename and \
        filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS

def start_encryption():
    dv.divide()
    tools.empty_folder('uploads')
    enc.encrypter()
    return render_template('success.html')

def start_decryption():
    dec.decrypter()
    tools.empty_folder('key')
    rst.restore()
    return render_template('restore_success.html')

@app.route('/return-key/My_Key.pem')
def return_key():
    list_directory = tools.list_dir('key')
```

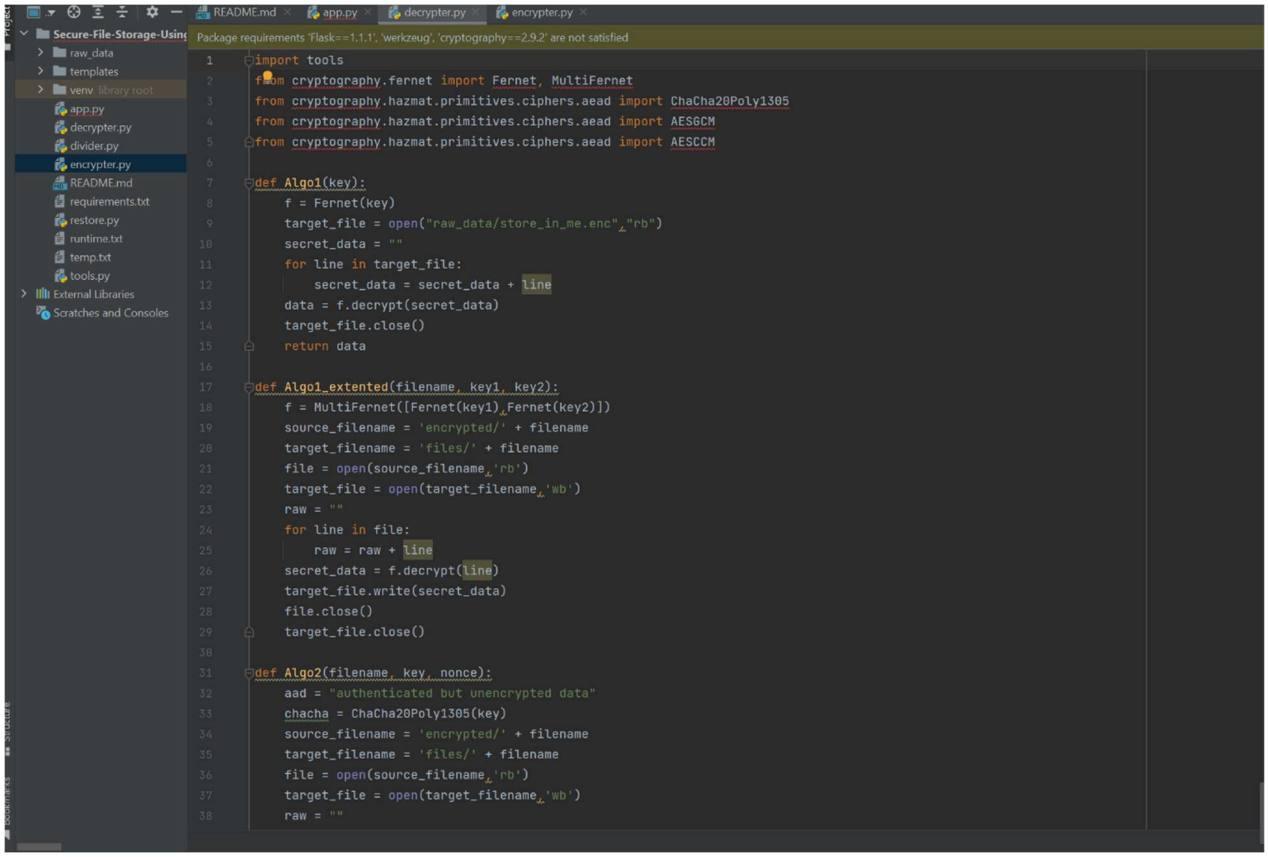


```
def return_file():
    list_directory = tools.list_dir('restored_file')
    filename = './restored_file/' + list_directory[0]
    print "**************************************"
    print list_directory[0]
    print "**************************************"
    return send_file(filename, attachment_filename=list_directory[0], as_attachment=True)

@app.route('/download/')
def downloads():
    return render_template('download.html')

@app.route('/upload')
def call_page_upload():
    return render_template('upload.html')

@app.route('/home')
def back_home():
    tools.empty_folder('key')
    tools.empty_folder('restored_file')
    return render_template('index.html')

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/data', methods=['GET', 'POST'])
def upload_file():
    tools.empty_folder('uploads')
    if request.method == 'POST':...

@app.route('/download_data', methods=['GET', 'POST'])
def upload_key():
    tools.empty_folder('key')
    if request.method == 'POST':...

if __name__ == '__main__':
    app.run(host='127.0.0.1', port=8000, debug=True)
```

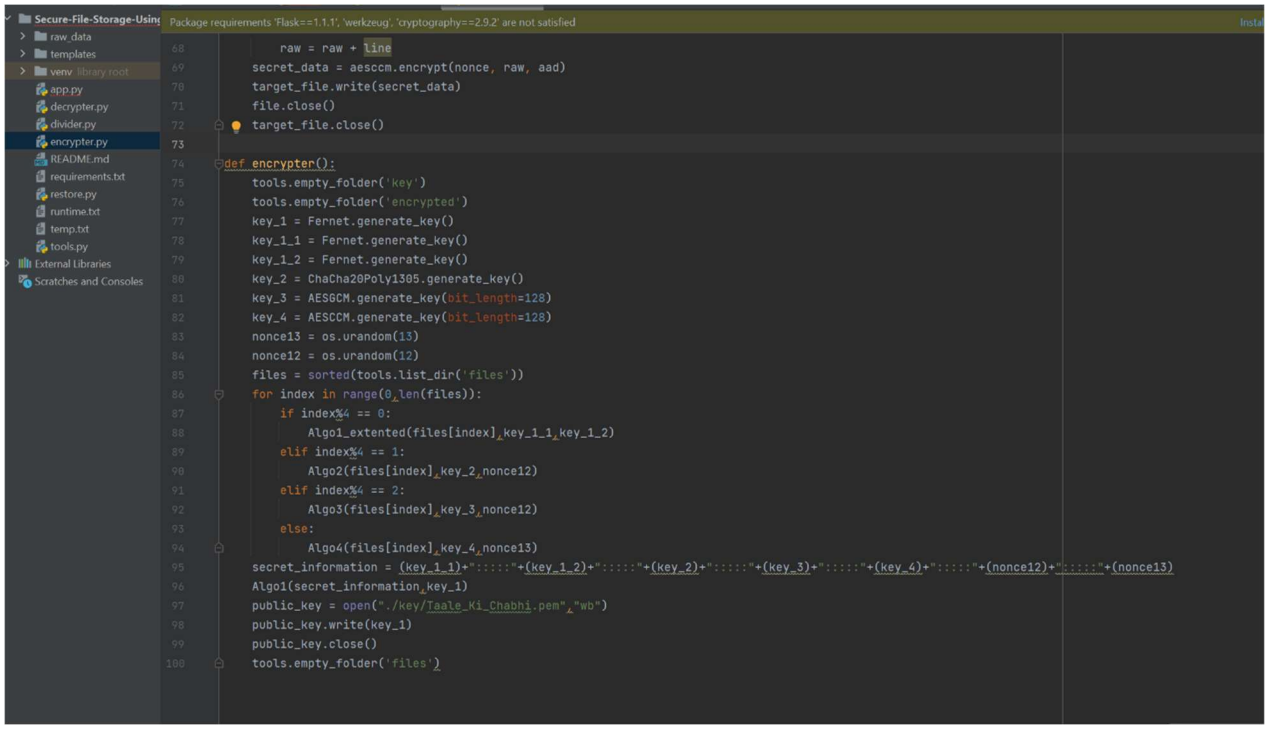
Decrypter:

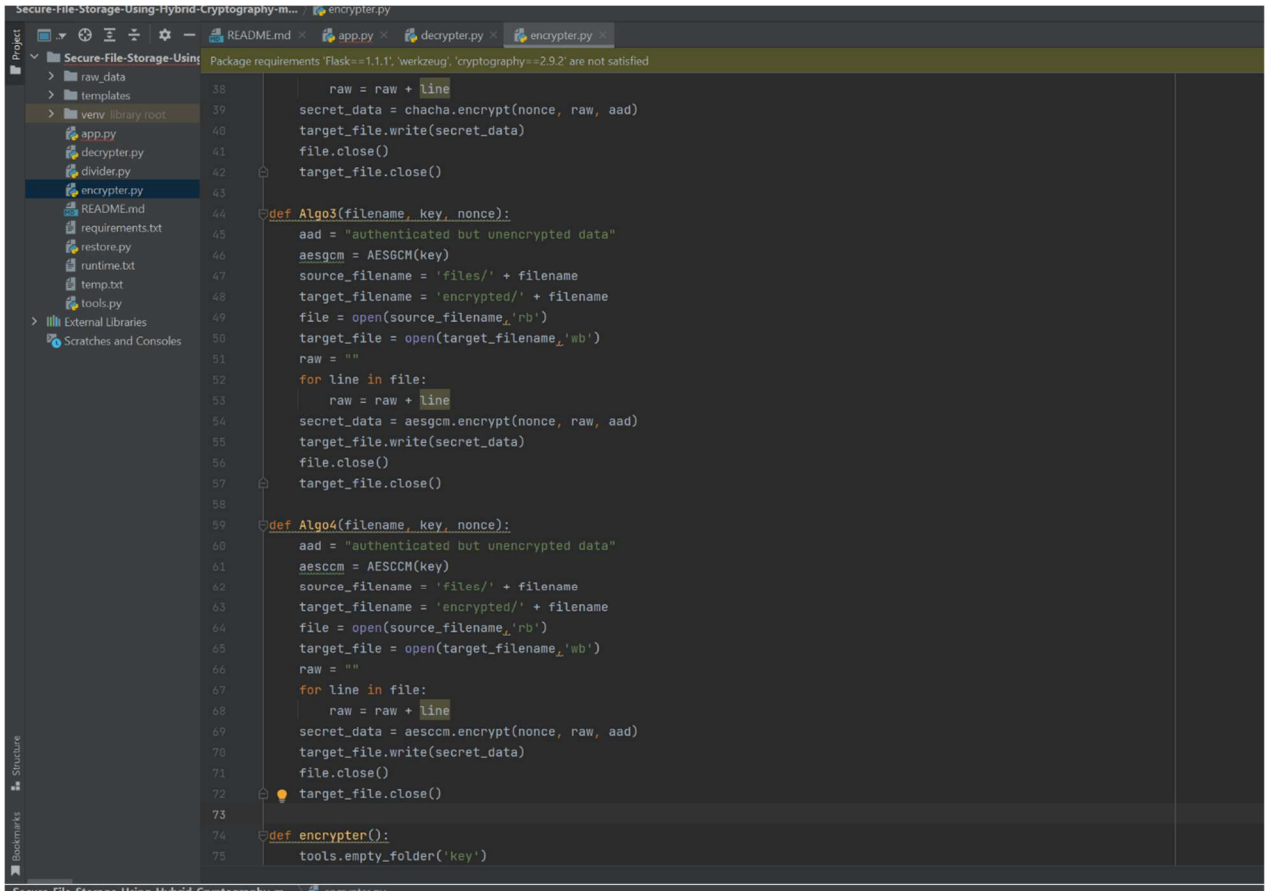

divider.py
encrypter.py
README.md
requirements.txt
restore.py
runtime.txt
temp.txt
tools.py
External Libraries
Scratches and Consoles
def decrypter():
tools.empty_folder('files')
key_1 = ""
list_directory = tools.list_dir('key')
filename = './key/' + list_directory[0]
public_key = open(filename,"rb")
for line in public_key:
key_1 = key_1 + line
public_key.close()
secret_information = Algo1(key_1)
list_information = secret_information.split('::::::')
key_1_1 = list_information[0]
key_1_2 = list_information[1]
key_2 = list_information[2]
key_3 = list_information[3]
key_4 = list_information[4]
nonce12 = list_information[5]
nonce13 = list_information[6]
files = sorted(tools.list_dir('encrypted'))
for index in range(0,len(files)):
if index%4 == 0:
Algo1_extented(files[index],key_1_1,key_1_2)
elif index%4 == 1:
Algo2(files[index],key_2,nonce12)
elif index%4 == 2:
Algo3(files[index],key_3,nonce12)
else:
Algo4(files[index],key_4,nonce13)

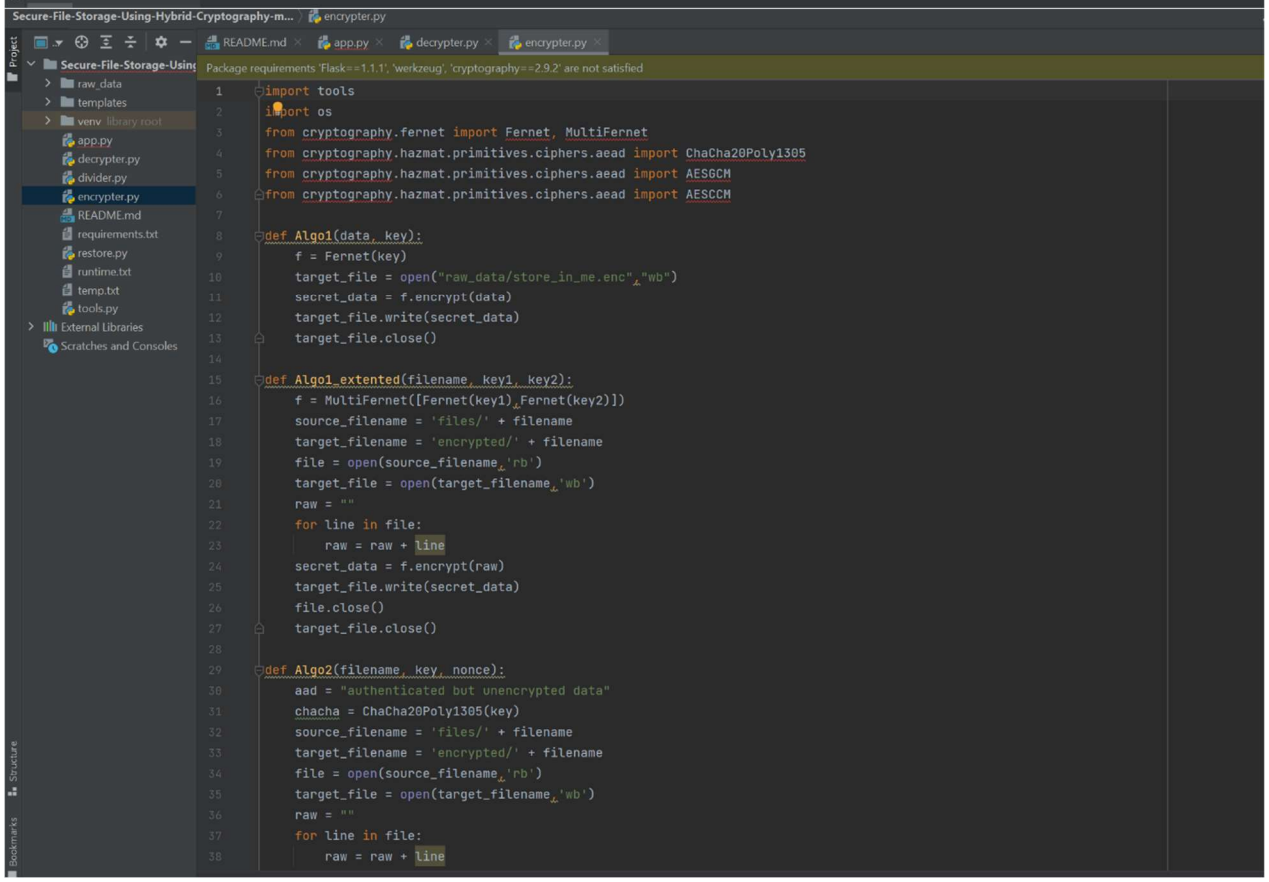

README.md
app.py
decrypter.py
encrypter.py
Package requirements 'Flask==1.1.1', 'werkzeug', 'cryptography==2.9.2' are not satisfied
raw_data
templates
venv library root
app.py
decrypter.py
divider.py
encrypter.py
README.md
requirements.txt
restore.py
runtime.txt
temp.txt
tools.py
External Libraries
Scratches and Consoles
target_file = open(target_filename,'wb')
raw = ""
for line in file:
raw = raw + line
secret_data = chacha.decrypt(nonce, raw, aad)
target_file.write(secret_data)
file.close()
target_file.close()
def Algo3(filename, key, nonce):
aad = "authenticated but unencrypted data"
aesgcm = AESGCM(key)
source_filename = 'encrypted/' + filename
target_filename = 'files/' + filename
file = open(source_filename,'rb')
target_file = open(target_filename,'wb')
raw = ""
for line in file:
raw = raw + line
secret_data = aesgcm.decrypt(nonce, raw, aad)
target_file.write(secret_data)
file.close()
target_file.close()
def Algo4(filename, key, nonce):
aad = "authenticated but unencrypted data"
aesccm = AESCCM(key)
source_filename = 'encrypted/' + filename
target_filename = 'files/' + filename
file = open(source_filename,'rb')
target_file = open(target_filename,'wb')
raw = ""
for line in file:
raw = raw + line
secret_data = aesccm.decrypt(nonce, raw, aad)
target_file.write(secret_data)
file.close()
target_file.close()


README.md
app.py
decrypter.py
encrypter.py
Package requirements 'Flask==1.1.1', 'werkzeug', 'cryptography==2.9.2' are not satisfied
Secure-File-Storage-Using
raw_data
templates
venv library root
app.py
decrypter.py
divider.py
encrypter.py
README.md
requirements.txt
restore.py
runtime.txt
temp.txt
tools.py
External Libraries
Scratches and Consoles
import tools
from cryptography.fernet import Fernet, MultiFernet
from cryptography.hazmat.primitives.ciphers.aead import ChaCha20Poly1305
from cryptography.hazmat.primitives.ciphers.aead import AESGCM
from cryptography.hazmat.primitives.ciphers.aead import AESCCM
def Algo1(key):
f = Fernet(key)
target_file = open("raw_data/store_in_me.enc","rb")
secret_data = ""
for line in target_file:
secret_data = secret_data + line
data = f.decrypt(secret_data)
target_file.close()
return data
def Algo1_extented(filename, key1, key2):
f = MultiFernet([Fernet(key1),Fernet(key2)])
source_filename = 'encrypted/' + filename
target_filename = 'files/' + filename
file = open(source_filename,'rb')
target_file = open(target_filename,'wb')
raw = ""
for line in file:
raw = raw + line
secret_data = f.decrypt(line)
target_file.write(secret_data)
file.close()
target_file.close()
def Algo2(filename, key, nonce):
aad = "authenticated but unencrypted data"
chacha = ChaCha20Poly1305(key)
source_filename = 'encrypted/' + filename
target_filename = 'files/' + filename
file = open(source_filename,'rb')
target_file = open(target_filename,'wb')
raw = ""

Encrypter:


Secure-File-Storage-Using
Package requirements 'Flask==1.1.1', 'werkzeug', 'cryptography==2.9.2' are not satisfied
raw_data
templates
venv library root
app.py
decrypter.py
divider.py
encrypter.py
README.md
requirements.txt
restore.py
runtime.txt
temp.txt
tools.py
External Libraries
Scratches and Consoles
raw = raw + line
secret_data = aesccm.encrypt(nonce, raw, aad)
target_file.write(secret_data)
file.close()
target_file.close()
def encrypter():
tools.empty_folder('key')
tools.empty_folder('encrypted')
key_1 = Fernet.generate_key()
key_1_1 = Fernet.generate_key()
key_1_2 = Fernet.generate_key()
key_2 = ChaCha20Poly1305.generate_key()
key_3 = AESGCM.generate_key(bit_length=128)
key_4 = AESCCM.generate_key(bit_length=128)
nonce13 = os.urandom(13)
nonce12 = os.urandom(12)
files = sorted(tools.list_dir('files'))
for index in range(0,len(files)):
if index%4 == 0:
Algo1_extented(files[index],key_1_1,key_1_2)
elif index%4 == 1:
Algo2(files[index],key_2,nonce12)
elif index%4 == 2:
Algo3(files[index],key_3,nonce12)
else:
Algo4(files[index],key_4,nonce13)
secret_information = (key_1_1)+":::::"+(key_1_2)+":::::"+(key_2)+":::::"+(key_3)+":::::"+(key_4)+":::::"+(nonce12)+":::::"+(nonce13)
Algo1(secret_information,key_1)
public_key = open("./key/Taale_Ki_Chabhi.pem","wb")
public_key.write(key_1)
public_key.close()
tools.empty_folder('files')


Secure-File-Storage-Using-Hybrid-Cryptography-m...
encrypter.py
README.md
app.py
decrypter.py
encrypter.py
Package requirements 'Flask==1.1.1', 'werkzeug', 'cryptography==2.9.2' are not satisfied
raw_data
templates
venv library root
app.py
decrypter.py
divider.py
encrypter.py
README.md
requirements.txt
restore.py
runtime.txt
temp.txt
tools.py
External Libraries
Scratches and Consoles
raw = raw + line
secret_data = chacha.encrypt(nonce, raw, aad)
target_file.write(secret_data)
file.close()
target_file.close()
def Algo3(filename, key, nonce):
aad = "authenticated but unencrypted data"
aesgcm = AESGCM(key)
source_filename = 'files/' + filename
target_filename = 'encrypted/' + filename
file = open(source_filename,'rb')
target_file = open(target_filename,'wb')
raw = ""
for line in file:
raw = raw + line
secret_data = aesgcm.encrypt(nonce, raw, aad)
target_file.write(secret_data)
file.close()
target_file.close()
def Algo4(filename, key, nonce):
aad = "authenticated but unencrypted data"
aesccm = AESCCM(key)
source_filename = 'files/' + filename
target_filename = 'encrypted/' + filename
file = open(source_filename,'rb')
target_file = open(target_filename,'wb')
raw = ""
for line in file:
raw = raw + line
secret_data = aesccm.encrypt(nonce, raw, aad)
target_file.write(secret_data)
file.close()
target_file.close()
def encrypter():
tools.empty_folder('key')


Secure-File-Storage-Using-Hybrid-Cryptography-m...
encrypter.py
README.md
app.py
decrypter.py
encrypter.py
Package requirements 'Flask==1.1.1', 'werkzeug', 'cryptography==2.9.2' are not satisfied
import tools
import os
from cryptography.fernet import Fernet, MultiFernet
from cryptography.hazmat.primitives.ciphers.aead import ChaCha20Poly1305
from cryptography.hazmat.primitives.ciphers.aead import AESGCM
from cryptography.hazmat.primitives.ciphers.aead import AESCCM
def Algo1(data, key):
f = Fernet(key)
target_file = open("raw_data/store_in_me.enc","wb")
secret_data = f.encrypt(data)
target_file.write(secret_data)
target_file.close()
def Algo1_extented(filename, key1, key2):
f = MultiFernet([Fernet(key1),Fernet(key2)])
source_filename = 'files/' + filename
target_filename = 'encrypted/' + filename
file = open(source_filename,'rb')
target_file = open(target_filename,'wb')
raw = ""
for line in file:
raw = raw + line
secret_data = f.encrypt(raw)
target_file.write(secret_data)
file.close()
target_file.close()
def Algo2(filename, key, nonce):
aad = "authenticated but unencrypted data"
chacha = ChaCha20Poly1305(key)
source_filename = 'files/' + filename
target_filename = 'encrypted/' + filename
file = open(source_filename,'rb')
target_file = open(target_filename,'wb')
raw = ""
for line in file:
raw = raw + line

9. Conclusion

The main aim of this system is to securely store and retrieve data on the cloud that is only controlled by the owner of the data. Cloud storage issues of data security are solved using cryptography and steganography techniques. Data security is achieved using RC6, 3DES and AES algorithm. Key information is safely stored using LSB technique (Steganography). Less time is used for the encryption and decryption process using multithreading technique. With the help of the proposed security mechanism, we have accomplished better data integrity, high security, low delay, authentication, and confidentiality. In the future we can add public key cryptography to avoid any attacks during the transmission of the data from the client to the server. System performance is evaluated by calculating time of encryption, time of key generation and decryption time. The time consumed by KP-ABE outsourcing scheme can be measured on IDE which refers to Integrated Development Environment JAVA when the database is held in MYSQL. Performance is analyzed by calculating time of Encryption and upload, time of download and Decryption and time of key generation. With the help of multiple sizes of files, Performance analysis is performed and by taking note of encryption time of files having multiple sizes and decryption time of the files and key generation time, the results will be attained. Hybrid cryptography systems currently use combinations of RSA and AES, AES and Blowfish, Blowfish and ECC and Triples DES among various others. Such combinations ensure that those CSPs can harness both algorithms advantages in a hybrid system to ensure access control, authorization, authentication, and confidentiality

10. References

[1] M. S. Abbas, S. S. Mahdi and S. A. Hussien, "Security Improvement of Cloud Data Using Hybrid Cryptography and Steganography," 2020 International Conference on Computer Science and Software Engineering (CSASE), 2020, pp. 123-127, doi: 10.1109/CSASE48920.2020.9142072. [2] A. Rashid and A. Chaturvedi, "Cloud Computing Characteristics and Services A Brief Review", International Journal of Computer Sciences and Engineering, vol. 7, no. 2, pp. 421-426, 2019. Available: Link. [3] "Hybrid encryption | Tink | Google Developers", Google Developers, 2022. [Online]. Available: Link. [4] P. P. Chinnasamy, s. Padmavathi, R. Swathy and S. Rakesh, "Efficient Data Security Using Hybrid Cryptography on Cloud Computing", Inventive Communication and Computational Technologies, vol. 145, 2021. [5] K. Sudharson, M. Akshaya, M. Lokeswari, K. Gopika, "Secure Authentication scheme using CEEK technique for Trusted Environment", 2022 International Mobile and Embedded Technology Conference (MECON), pp.66-71, 2022. [6] R. , P. , Y. , and P. , “Use of digital signature with diffie hellman key exchange and AES encryption algorithm to enhance data security in cloud computing,” in 2013 International Conference on Communication Systems and network technologies: 6-8 april, 2013, Gwalior, India, Piscataway, NJ: Institute of Electrical and Electronics Engineers, 2013, pp. 437–439. [7] A. K. Dubey et al, "Cloud-user security based on RSA and MD5 algorithm for resource attestation and sharing in java environment," in 2012, . DOI: 10.1109/CONSEG.2012.6349503. [8] C. Jian-quan, H. Du, X. Zhang, Y. Lin and L. Zeng, "Ensure Data Security in Cloud Storage", The Quality of Higher Education, vol. 10, pp. 284-287, 2011.