

Secure File Storage using Neo-Hybrid Cryptography

A PROJECT REPORT

Submitted by

DEEPACH CHANDRU M (513419104010)

RAGHUL J (513419104035)

THILLAI VALAVAN A S (513419104049)

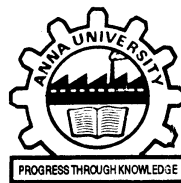
in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING



University College of Engineering Kancheepuram

ANNA UNIVERSITY : CHENNAI 600 025

May 2023

ANNA UNIVERSITY: CHENNAI- 600 025

BONAFIDE CERTIFICATE

Certified that this project report titled **“SECURE FILE STORAGE USING NEO-HYBRID CRYPTOGRAPHY”** is the bonafide work of **“DEEPACH CHANDRU M (513419104010), RAGHUL J (513419104035), THILLAI VALAVAN A S (513419104049)”** of Computer Science & Engineering whose carried out this project work under my supervision

SIGNATURE

**Dr. K. Selvabhuvaneswari, ME,
Ph.D.,**

HEAD OF THE DEPARTMENT,

Assistant Professor,

Department of CSE,

University College of Engineering,

Kancheepuram-631552.

SIGNATURE

Mr.J. Devanathan, M.Tech,

SUPERVISOR,

Teaching Fellow,

Department of CSE,

University College of Engineering,

Kancheepuram-631552.

Submitted for the Project Viva Voice held on:

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We are always thankful to our college Dean **Dr.V. KAVITHA, M.E, Ph.D., PROFESSOR**, who endorsed us throughout this project.

Our heartfelt thanks to HOD **Dr.K. SELVABHVANESWARI, M.E, Ph.D., PROFESSOR**, Department of Computer Science and Engineering, University College of Engineering, Kancheepuram, for the prompt and limitless help in providing the excellent infrastructure to do the project and to prepare the thesis.

We express our deep sense of gratitude to our guide **Mr.J.DEVANATHAN, M.Tech, Teaching Fellow**, Department of Computer Science & Engineering, for his invaluable support and guidance and encouragement for successful completion of this project. His vision and spirit will always inspire and enlighten us.

We express our sincere thanks to the project committee members **Mr.G.MANI, M.E, ASSISTANT PROFESSOR, Mrs.T.KALA, M.E, ASSISTANT PROFESSOR**, Department of Computer Science and Engineering, Kancheepuram, for their invaluable guidance and technical Support.

We thank all the faculty members of the Department of Computer Science and Engineering for their valuable help and guidance. We thank the almighty whose showers of blessings made this project a reality.

DEEPACH CHANDRU M	RAGHUL J	THILLAIVALAVAN AS
(513419104010)	(513419104035)	(513419104049)

ABSTRACT

Security is a major concern in a wide range of applications, from cloud storage to messaging via chat. Many different approaches have also been proposed to provide data protection in the cloud, such as AES, DES, and RSA, but Existing systems often fail when only a certain form of encoding is utilized, either AES, OR DES, OR RSA depending on a consumer requirement. Cryptographic techniques such as DES and AES are used in order to provide Security to the data but using a single technique sometimes doesn't provide high-level security. So here the proposed work focused on introducing a hybrid cryptographic mechanism that involves multiple techniques to encrypt and decrypt the data.

Steganography differs from encryption, but using both sets help to raise the level of security of protected information and prevent the discovery of secret communications. The keys of the used algorithms are encrypted and then stored in a cover image using LSB Steganography. Thus, both the data and the key are secured.

TABLE OF CONTENTS

CHAPTER NO:	TITLE	PAGE NO:
	ABSTRACT	i
	LIST OF FIGURES	v
	LIST OF SYMBOLS	v
	LIST OF ABBREVIATION	vi
1.	INTRODUCTION	1
	1.1 Objective	2
	1.2 Problem Statement	2
	1.3 Proposed Solution	3
	1.4 Need For the System	4
	1.5 System Analysis	4
2.	LITERATURE SURVEY	12
	2.1 Literature of Related Work	12
	2.2 Issues Identified	14
	2.3 Summary of Related Works	14

3.	SYSTEM DESIGN	15
	3.1 System Architecture	15
	3.2 System Specification	
	3.2.1 Hardware Specification	17
	3.2.2 Software Specification	17
4.	DETAILED ARCHITECTURE	19
	4.1 List of Modules	19
	4.1.1 Data Preprocessing	19
	4.1.2 cryptography Algorithm	21
	4.1.3 Steganography and key Transfer	24
	4.1.4 Decryption and Extraction	25
5.	IMPLEMENTATION AND RESULTS	30
	5.1 Data Preprocessing	30
	5.2 Cryptography Algorithm	34
	5.3 Steganography and Key Transfer	35
	5.4 Decryption and Extraction	37

6.	SYSTEM TESTING	40
	6.1 System Testing	40
	6.1.1 Test Cases	40
	6.1.2 Testing Techniques	40
7.	CONCLUSION AND FUTURE WORK	44
	7.1 Conclusion	44
	7.2 Future Enhancement	45
	7.3 Performance Evaluation	46
	7.4 Output	47
	REFERENCES	48

LIST OF FIGURES

Fig. No:	Description	PAGE.NO:
1.5.1	AES Block diagram	6
1.5.2	Blowfish Block Diagram	7
1.5.3	TripleDES Block Diagram	8
1.5.4	International Data Encryption Algorithm (IDEA) Block Diagram	9
1.5.5	Fernet Block Diagram	10
3.1.1	System Architecture - Encryption	15
3.1.2	System Architecture - Decryption	16
4.1.1	Module-1 Block Diagram	19
4.1.2	Module-2 Block Diagram	21
4.1.3	Module-3 Block Diagram	24
4.1.4	Module-4 Block Diagram	26
5.3.2	LSB Steganography Images	36
7.3.1	Performance Evaluation Graph	46
7.4.1	Output	47

LIST OF SYMBOLS

\wedge - append string

\oplus – Logical XOR function

LIST OF ABBREVIATION

AES- Advanced Encryption Standard

DES- Data Encryption Standard

IDEA- International Data Encryption Algorithm

LSB- Least Significant Bit

IV-Initialization Vectors

CBC-Cipher Block Chaining Mode

SHA- Secure Hash Algorithm

CHAPTER - 1

1. INTRODUCTION

The widened handling of digital media for information transmission through secure and unsecured channels exposes messages sent via networks to intruders or third parties. Encryption of messages in this modern age of technology becomes necessary for ensuring that data sent via communications channels become protected and made difficult for deciphering. Enormous number of transfer of data and information takes place through internet, which is considered to be most efficient though it's definitely a public access medium. Therefore to counterpart this weakness, many researchers have come up with efficient algorithms to encrypt this information from plain text into ciphers. In information security, encryption is the process of transforming information using an algorithm to make it unreadable to anyone except those possessing special knowledge, usually referred to as a key. The result of the process is encrypted information. The reverse process is referred to as decryption. Single cryptosystems are nowadays cracked with the latest hardware capabilities so a hybrid system is proposed in this paper.

1.1 OBJECTIVE

The main objective of this project is to build a Hybrid Crypto-system that secures data on multiple layers and also ensures security of keys. The Hybrid cryptosystem also securely stores the keys so that they don't lead to any vulnerabilities. To create a crypto-system that provides excellent security without compromising on performance and speed. To overcome the performance-security tradeoffs of cryptographic algorithms when used separately. To create a crypto-system that provides excellent security without compromising on performance and speed.

1.2 PROBLEM STATEMENT

Various Encryption Algorithms are used in apps and services to secure data. But the advent of new and sophisticated technologies is making these existing systems obsolete. Advancements in Hardware have significantly reduced the time required to break a cryptographic system. Various kinds of attacks have weakened the existing systems

Crypto-analysis and special mathematical attacks have made these systems quite vulnerable to being broken by cryptographers. Key security is another vulnerability that modern systems face. Ensuring safe storage and transmission of sensitive keys is a major fault of existing systems. Another key aspect of securing data is to ensure that performance is not compromised.

Generally, encryption algorithms to provide higher levels of security use larger key lengths, but that hampers the performance of the system.

A single layered standalone crypto-system can sometimes have trade-offs that might lead to data leaks, and also hamper key security. A standalone system has vulnerabilities that often effect the security of data. The various pitfalls of standalone systems at times compromise the performance and speed.

1.3 PROPOSED SOLUTION

The solution is to secure the data for storage and transmission by building an application that takes the confidential data as input and undergo splitting of data, involve in hybrid cryptography algorithm[Blowfish , AES, TripleDES, IDEA and Fernet] hide the encrypted the keys in an image using LSB steganography , and decryption application decrypts the data from image and encrypted files.

Since we are splitting the data and encrypting them the hackers find it difficult to determine which encryption algorithm is used to which part. When a user uploads data, it is divided into five sections, the first of which is encrypted with AES, the second with BlowFish, the third with Triple-DES, the fourth part with IDEA and the fifth part with Fernet Algorithm.LSB steganography is used to store the keys in the image, and the encrypted files are stored in the cloud. Users must first recover the keys from the image before they can import all data from the server. These keys are then used to decrypt the data once more with AES, Triple-DES, BlowFish, IDEA and Fernet. This approach increases the security of records.

The system randomly generates two Keys and two Initialization vectors(IV) for encryption. These two keys and two IVs are collaged as a single key file and then embedded in a cover image using LSB-Steganography.

1.4 NEED FOR THE SYSTEM

The file is being encrypted by using symmetric key cryptography and steganography techniques. The system is very secure and robust. Data of the users is highly secured on hybrid cryptography algorithms and steganography which helps in avoiding unauthorized access from the outside world.. Data security is a major priority. This system can be implemented in the banking and corporate sectors to securely transfer confidential data.

1.5 SYSTEM ANALYSIS

1.5.1 CRYPTOGRAPHY

Cryptography is the study of techniques and methods of securing data and communication between different parties, from malicious intruders, eavesdroppers, and adversaries. Cryptography involves various aspects of securing data communication like creating protocols to facilitate secured and safe connectivity, analyzing cryptosystems for vulnerabilities, and performance and checking for the fulfillment of information security aspects like integrity, authenticity, confidentiality, and non-repudiation.

Cryptography is the backbone of Information security and Information Security is essential for a user's digital presence. Cryptography is primarily associated with Encryption. Encryption is the procedure of converting readable information/data to unreadable gibberish. Encryption is facilitated by the use of Cryptographic Algorithms or Cryptosystems. Modern cryptosystems are heavily dependent on mathematical theories and functions as well as theoretical computer

science practices. Encryption is facilitated using mathematical functions that require a user-known secret, called the Key.

Decryption reverses what encryption does, that is, decryption converts the unintelligible gibberish back into readable information. Using the Key. Cryptography algorithms can be classified into two categories - Asymmetric Cryptography and Symmetric-Key Cryptography. Symmetric-Key algorithms for Cryptography uses the identical keys for both encryption, as well as for decryption, whereas Asymmetric Cryptography uses different keys.

1.5.2 Advanced-Encryption Standard(AES)

Rijndael, proposed by Belgian cryptographers, Vincent Rijmen and Joan Daemen, is Symmetric-key Block-Cipher that has been established as the Advanced Encryption Standard by the National Institute of Standards and Technology (NIST) of The United States of America, in the year of 2001.

The AES may have keys varying in size between 128, 192, 256 bits, & having 10, 12, 14 rounds respectively. The n-Bits key is expanded using AES Key-Scheduling into several subkeys depending on the number of rounds. In the beginning, the input block is XORed with an Initial Round-Key. Then, for the first N-1 rounds, 4 Round Functions are applied on each block. The first-round function is Substitute Bytes where every byte is substituted by another, from the lookup table. Followed by Shift-Rows, where the last three rows are cyclically shifted by a certain number of steps. Shift-Rows is followed by Mix-Columns, where a linear mixing operation is executed on the columns, combining the 4-bytes of each column. Lastly, the Add-RoundKey function is executed on the current state, where each byte, and a byte of the round key are combined using bitwise XOR operation.

For the Nth round, i.e. the last round all the above functions are applied except the Mixed Columns step.

AES is one of the most extensively used and secure algorithms for data security. Even though it is slower than blowfish, it provides a higher level of data security.

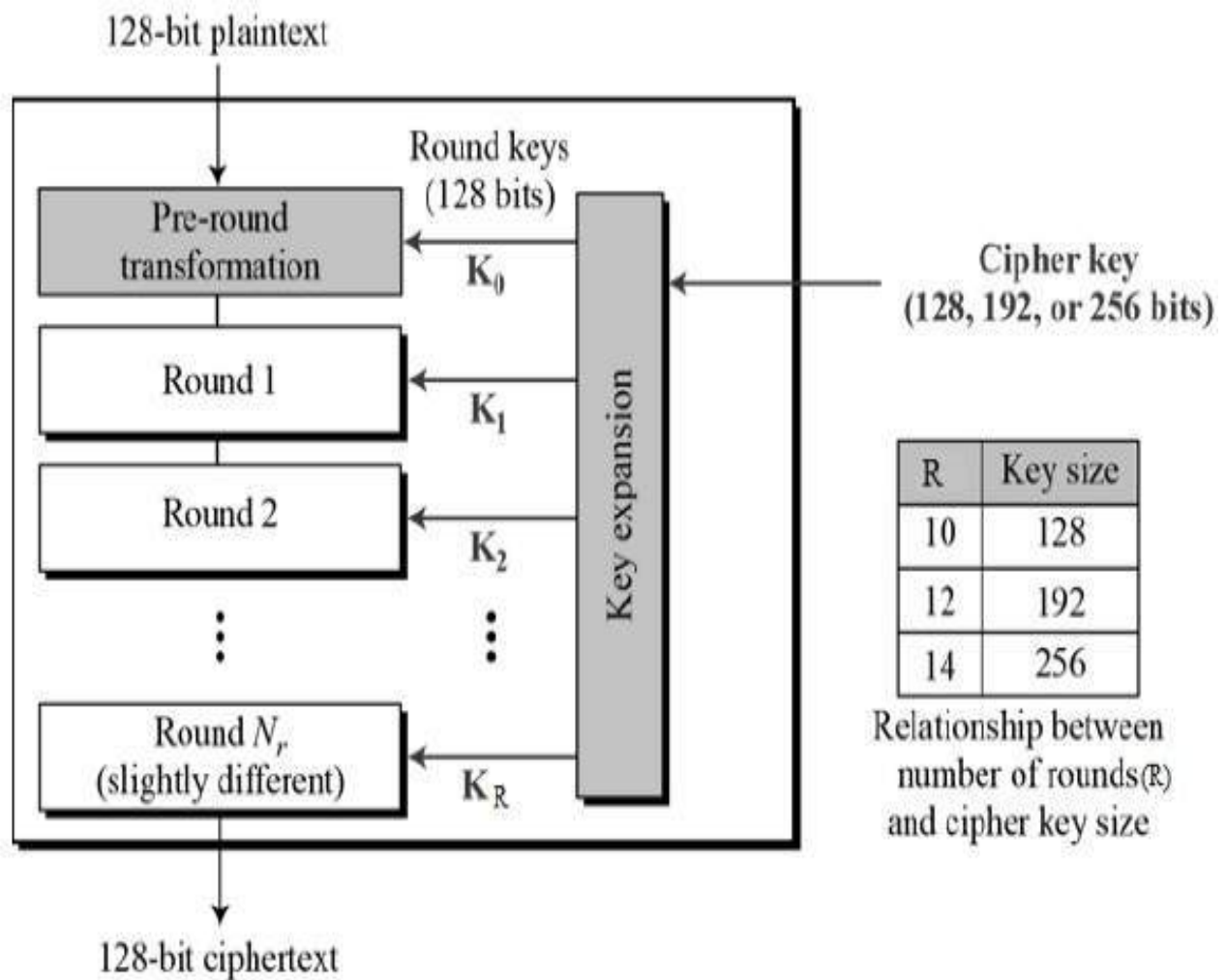


Fig.No.1.5.1 AES block diagram

1.5.3 Blowfish Algorithm

Blowfish, a Symmetric-Key Block Cipher, was developed by B. Schneier in the year 1993. Blowfish algorithm has 64 Bits block size and variable key length of 32 to 448 Bits. It is particularly known for its features like complicated key schedules and key dependent s-boxes. Being a Feistel cipher it has 16 rounds. Each round, consists of four steps. In n th round, the left half of the block and the n th element in the subkey-array are XORed \oplus followed by passing it to the round function F . The return from the function F and the right half of the initial block are XORed \oplus and then swapped. The round function F divides the 32-bit input into four 8-bit blocks that are then fed to 4 different S-Boxes.

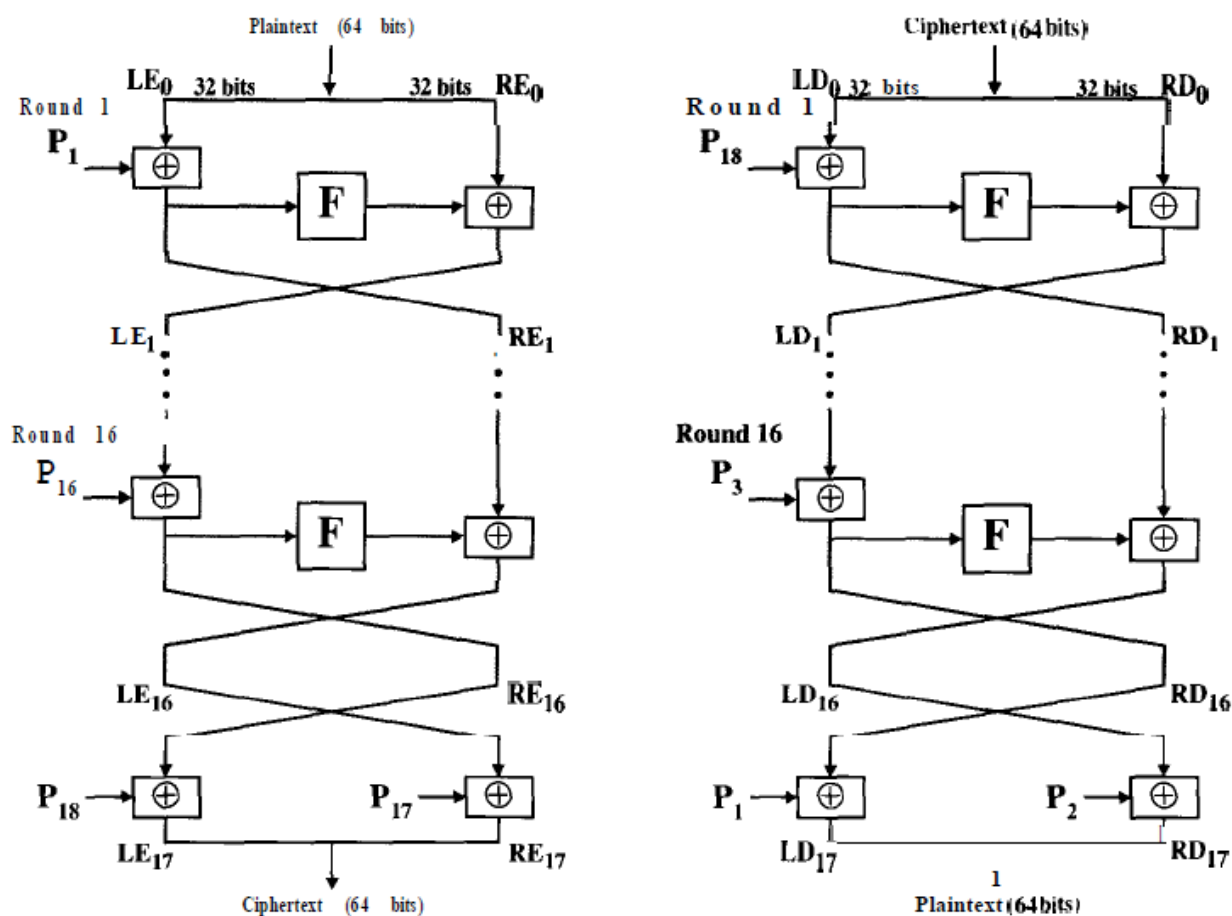


Fig 1.5.2 Blowfish Block Diagram

1.5.4 Triple-DES Encryption

The DES (Data Encryption Standard) In cryptography, Triple DES (3DES or TDES),officially the Triple Data Encryption Algorithm (TDEA or Triple DEA), is a symmetric-key block cipher, which applies the DES cipher algorithm three times to each data block. While The government and industry standards abbreviate the algorithm's name as TDES (TripleDES) and TDEA (Triple Data Encryption Algorithm), RFC 1851 referred to it as 3DES from the time it first promulgated the idea, and this namesake has since come into wide use by most vendors, users, and cryptographers. Before using 3DES, users first generate and distribute a 3DES key K , which consists of three different DES keys K_1 , K_2 and K_3 . This means that the actual 3DES key has length $3 \times 56 = 168$ bits.

- Encrypt the plaintext blocks using single DES with key K_1 .
- Now decrypt the output of step 1 using single DES with key K_2 .
- Finally, encrypt the output of step 2 using single DES with key K_3 .
- The output of step 3 is the cipher text

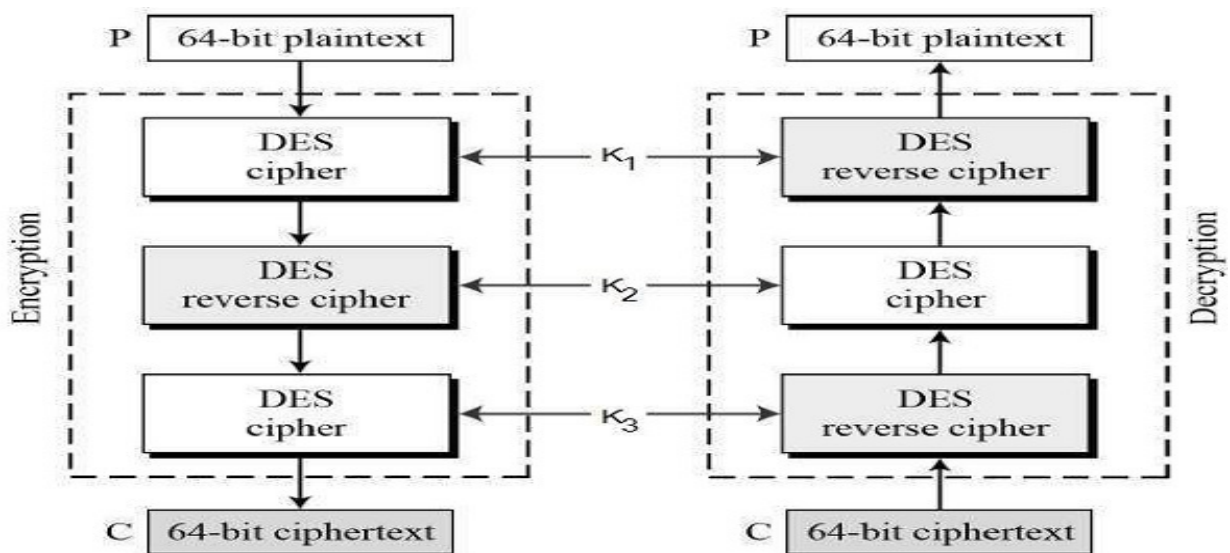
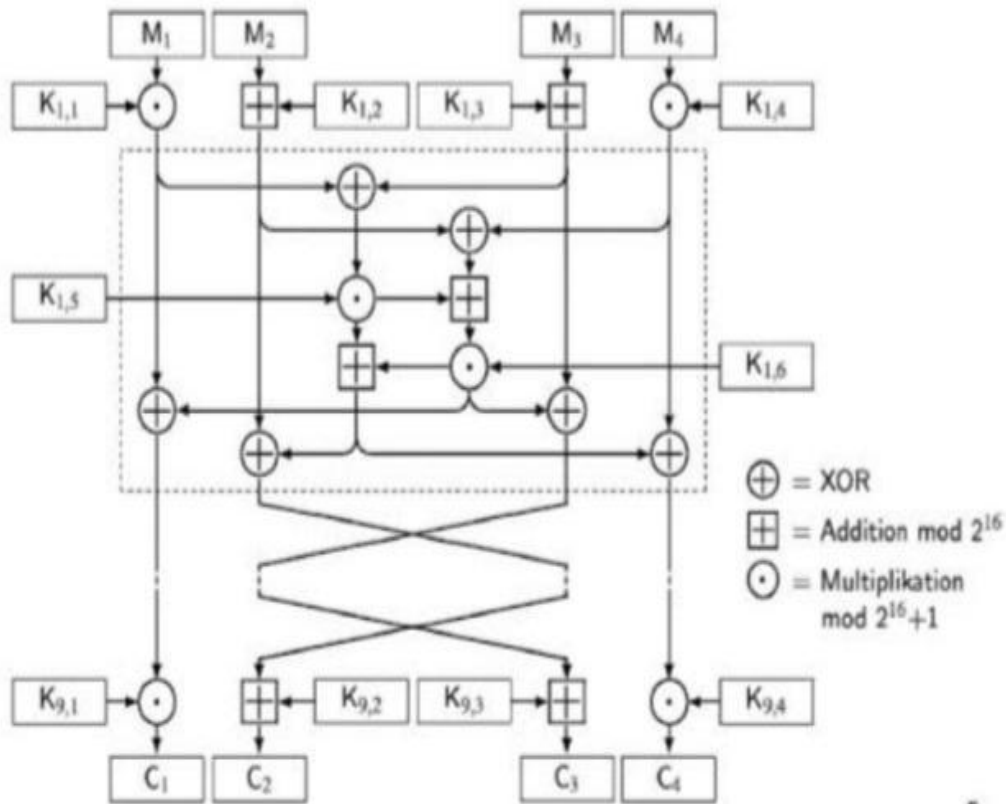


Fig.No:1.5.3 TripleDES Block Diagram

1.5.4 INTERNATIONAL DATA ENCRYPTION ALGORITHM (IDEA)

It is a symmetric key block cipher encryption algorithm designed to encrypt text to an unreadable format for transmission via the internet. IDEA uses a 128-bit key and operates on 64-bit blocks. Essentially, it encrypts a 64-bit block of plaintext into a 64-bit block of ciphertext. This input plaintext block is divided into four sub-blocks of 16 bits each.



5

Fig.No:1.5.4 International Data Encryption Algorithm (IDEA) Block Diagram

1.5.5 FERNET ALGORITHM

Fernet guarantees that a message encrypted using it cannot be manipulated or read without the key. Fernet is an implementation of symmetric (also known as “secret key”) authenticated cryptography. Fernet also has support for implementing key rotation via Multi-Fernet .

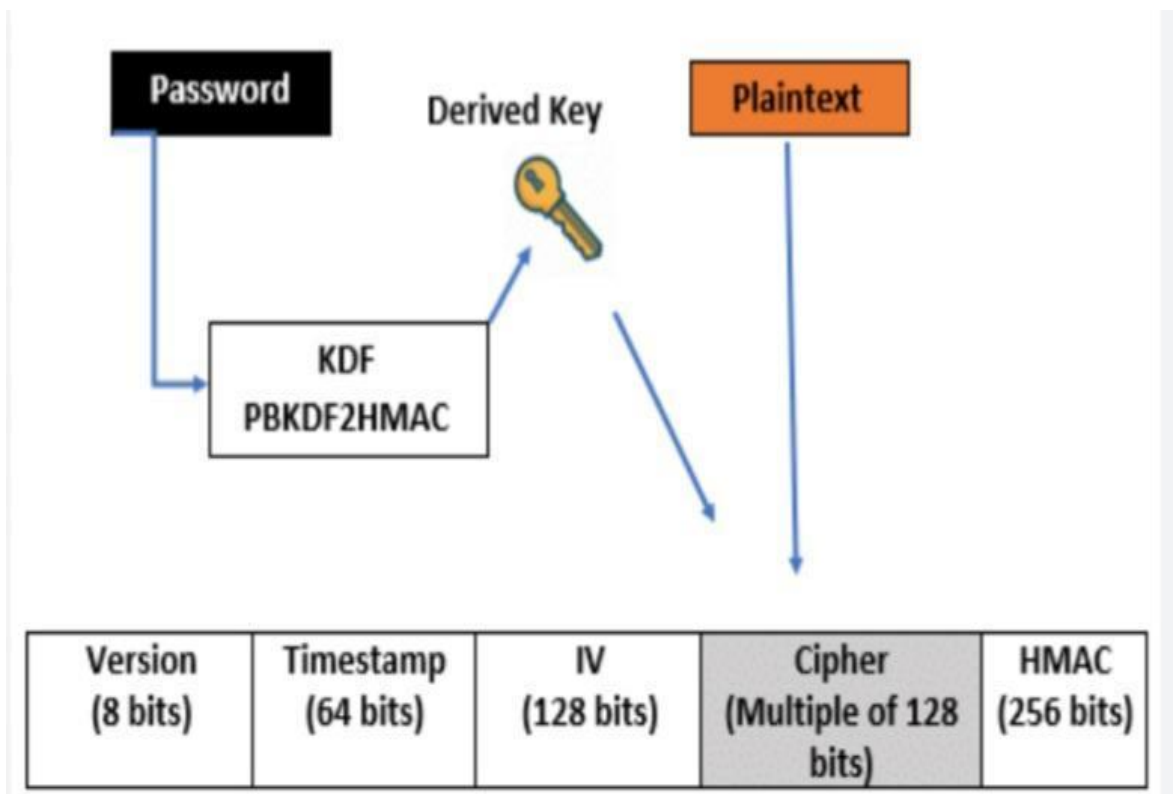
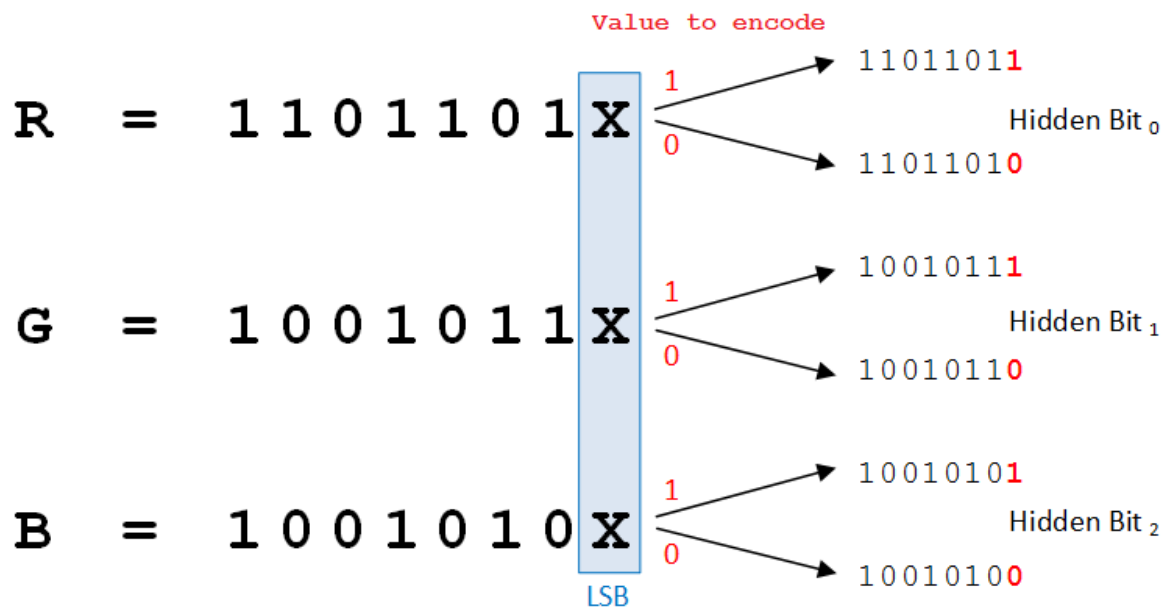


Fig.No:1.5.5 Fernet Block Diagram

1.5.4 LSB-Image Steganography

Least Significant Bit Steganography is a technique of hiding data within digital media, here, Image. Images are made up of pixels, and the value of each pixel usually refers to the color-code of that pixel. In a photo's gray-scale mode, these pixel values range from 0-255. In LSB Image Steganography, the least-significant bit of a pixel is changed, but that doesn't have much of a visible change in the image. A cover image is where the data is hidden. The cover image is converted to grey scale. The message is converted into binary. Each pixel of the image is traversed through, and for each pixel, initiate a temporary variable, temp. If the LSB of the Pixel Value and the message bit is the same, set temp as 0 and set temp as 1 otherwise. Update the output image pixel as image pixel value added with the temporary variable value, temp. This is done until the message is completely embedded.



CHAPTER - 2

2. LITERATURE SURVEY

2.1 Literature of Related Works

2.1.1 Design of a secure virtual file storage system on cloud using hybrid cryptography Bello A.Buhari, Aliyu Mubarak, Bello A.Bodinga, Muazu D.sifawa, Muaza D.sifawa int.j.advanced Networking and Application 2022 In this proposes system Secure file Storage, Hybrid Cryptography, AES, SHA-2 Algorithm are used to provide security.

2.1.2 A Comprehensive Study of Digital Image Steganography Techniques SHAHID RAHMAN¹, JAMAL UDDIN, MUHAMMAD ZAKARYA (Senior Member, IEEE), HAMEED HUSSAIN, AYAZ ALI KHAN , AFTAB AHMED AND MUHAMMAD HALEEM 6A Comprehensive Study of Digital Image Steganographic Techniques SHAHID RAHMAN¹, JAMAL UDDIN, MUHAMMAD ZAKARYA , (Senior Member, IEEE), HAMEED HUSSAIN⁴ IEEE 2023. In this proposes System steganography, data concealing, image quality assessment metrics.

2.1.3 Improving Security with Efficient Key Management in Public Cloud using Hybrid AES, ECC and LSB Steganography comparing with Novel Hybrid Cube Base Obfuscation K Vandhana, Dr.S.Krishna Kumari IEEE, 2022. This proposed system compares the Hybrid AES, ECC and LSB steganography and cube based Obfuscation. Based on the experimental and statistical results achieved it is concluded that cube based obfuscation seems to be secure.

2.1.4 Secure File Storage on Cloud using Hybrid Cryptography Vivek Sharma¹, Abhishek Chauhan², Harsh Saxena³, Shubham Mishra⁴, Sulabh Bansal

Assistant Professor, GLA University, Mathura(UP), India 2021. The proposed method Data Encryption Algorithm, Blowfish, Hybrid Cryptography, Cryptography.

2.1.5 Secure File Storage using Hybrid Cryptography Putta, Bharathi, Gayathri Annam, Anjali T and Vamsi Krishna Duggana 2021 The proposed method AES, RSA, DES, Hybrid cryptography, LSB, Security Algorithm are used to provide security.

2.1.6 Hiding Data Using Efficient Combination of RSA Cryptography, and Compression Steganography Techniques OSAMA FOUAD ABDEL WAHAB, ASHRAF A. M. KHALAF, AZIZA I. HUSSEIN, AND HESHAM F. A. HAMED IEEE, 2021. The proposed system deals about the efficient way of hiding data using a combination of RSA Cryptography and Compression Steganography Techniques with its types (lossy and lossless).

2.1.7 Hybrid Cryptography for Cloud Computing Heena Kausar Khan, Rubika Pradhan, B. R. Chandavarkar 2021 The proposed method Security, Confidentiality, Integrity, Authentication, RSA, SHA-256, Blowfish Algorithm are used to provide security.

2.2 Issues Identified

Splitting the given information into 3 parts and applying both symmetric and Asymmetric algorithms lead to confusion and time consuming so we are using only symmetric encryption algorithms. Placing the encrypted file in the cloud depends on internet facility and cloud server availability. There are combinations of cryptography algorithms that can be applied but some combinations tend to be time consuming and less secure.

2.3 Summary of Related works

In the proposed system, the main goal is to Encrypt data using Cryptographic Algorithms like Blowfish, AES and Triple Des in a cascading manner using the least resources and with least complexity. The use of multiple encryptions makes the data even more secure without the requirement of Higher Key lengths. The System consists of three Encryption Layers, a Key Generator and a List of Keys. The Key generates the random n-bits Key depending on the Encryption Algorithm, while the List of Keys stores the Key Generated in each layer. From the List of Keys, an AES Encryption Block encrypts it and a LSB Steganography Block to embed the Keys into a Cover Image. And send it to the receiver via mail, where he uses the decryption process to obtain the confidential data.

CHAPTER - 3

3.1 SYSTEM ARCHITECTURE

3.1.1 SYSTEM ARCHITECTURE FOR ENCRYPTION

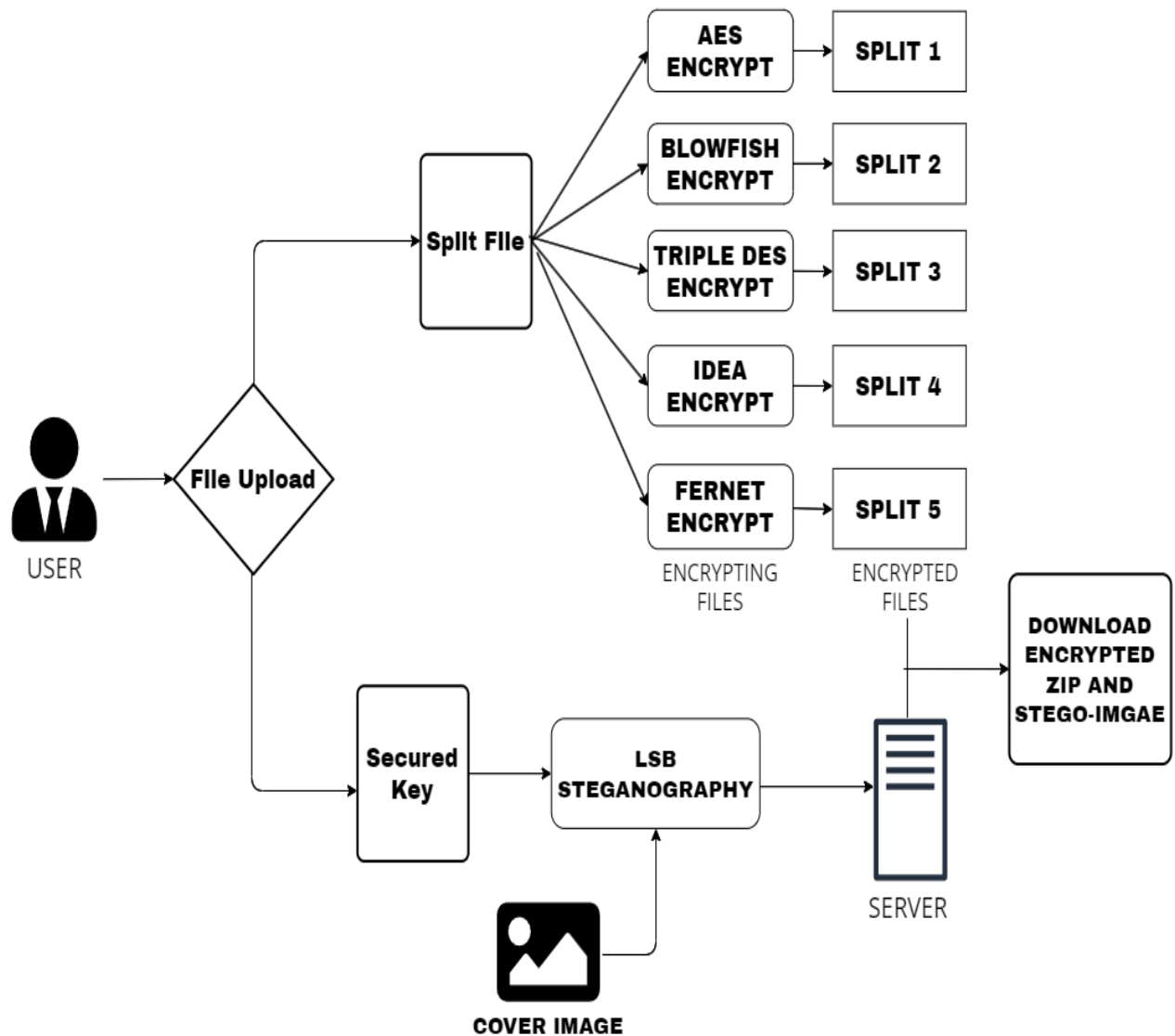


Fig.No.3.1.1 System architecture - Encryption

3.1.2 SYSTEM ARCHITECTURE FOR DECRYPTION

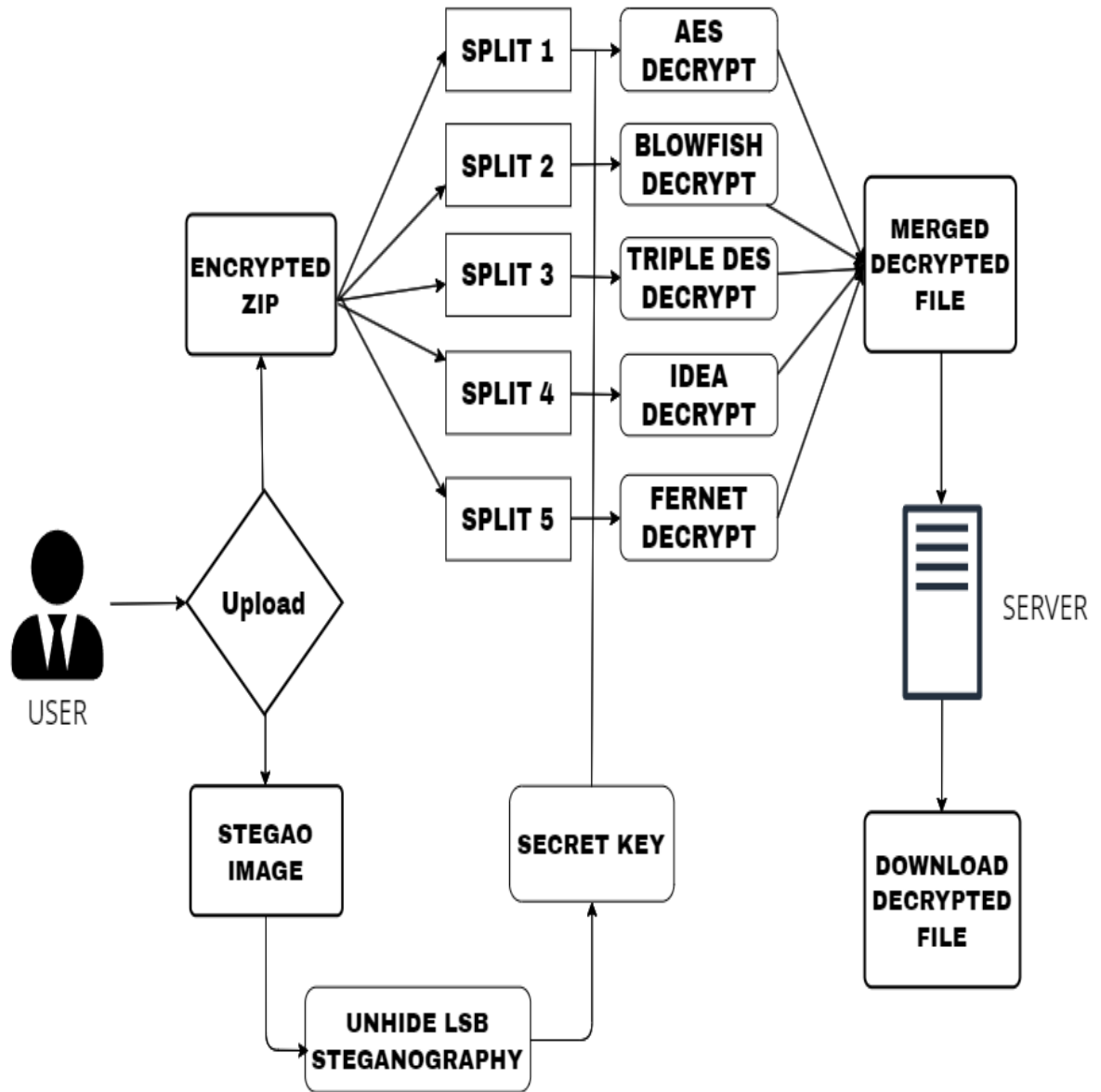


Fig.No.3.1.2 System architecture - Decryption

3.2 SYSTEM REQUIREMENTS

3.2.1 HARDWARE SPECIFICATION

The System doesn't require any specialized hardware as it doesn't carry out a large amount of processing.

The basic requirements are:

- Processors: Intel's Atom® processor or Intel's Core™ processor i3 or above
- Disk space: Recommended disk space is 1 GB
- RAM: 2GB or more

3.2.2 SOFTWARE SPECIFICATION

The project requires an operating system and other software needed for the execution of this application. Operating System provides the underlying environment for the execution of the application. Additional software is needed for the development.

- Operating systems: Microsoft Windows 7 or above, Apple's macOS, & Linux
- Python versions: Python 3.6.X and above
- Libraries: pycrypto, stegano, random, flask and other general purpose libraries

The project is implemented in Python with the help of a few libraries as mentioned next.

3.2.2.1 Python Libraries

The Python Libraries used for the project are:

- PyCrypto: It contains several containers and functions in Asymmetric and symmetric cryptography algorithms and hash functions.
- Hashlib: It contains multiple hash functions and operations. The constructors in the library provide some of the hashing functions used in this project.
- Stegano: This library is used to facilitate steganography in the project.
- Random: The random library is used to generate random numbers and strings for encryption.
- Flask: Flask is a micro Web framework written in python. It is a web framework that provides libraries to build lightweight web applications in python.
- Zipfile: Used to Zip the encrypted files into a single file transferable.

CHAPTER - 4

DETAILED ARCHITECTURE

4.1 List of Modules

- 1) Data Preprocessing
- 2) Cryptography Algorithm
- 3) Steganography and Key Transfer
- 4) Decryption and Extraction

4.1.1 Data Preprocessing

Data to be sent in an encrypted manner is received from the user along the key/password going to be used for encryption. The data given to the user in any format such as pdf or word is converted to a text sequence enabling us to make encryption. Key generation block is used to generate random keys for the cryptography algorithm , which is generally 16 bit lon Fig.No.4.1.1 Module-1 Block Diagram

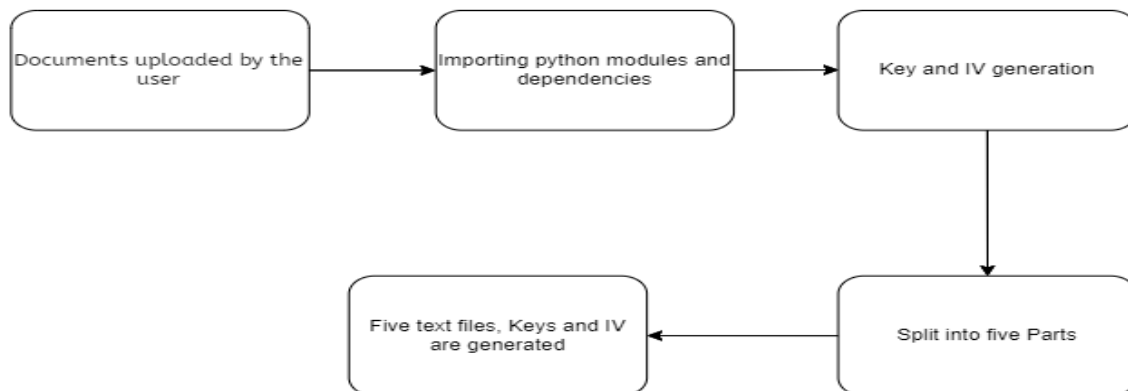


Fig.No.4.1.1 Module-1 Block Diagram

Input: File to be encrypted

Output: Processed text sequence and Keys for cryptography algorithm.

Pseudo code-Key generation:

```
def key_generator(size, case="default", punctuations="required"):
    if case=="default" and punctuations=="required":
        return ".join(random.choices(string.ascii_uppercase + string.ascii_lowercase +
string.digits + string.punctuation, k = size))
    elif case=="upper-case-only" and punctuations=="required":
        return ".join(random.choices(string.ascii_uppercase + string.digits +
string.punctuation, k = size))
```

Document segmentation:

```
for i in range(0,5):
    name=str(i+1)+".txt"
    f=open(name,'w')
    ctr=0
    for j in range(k,count):
        k+=1
        f.write(con[j])
        ctr +=1
        if(ctr==limit and i!=4):
            f.close()
            break
    f.close()
Segment()
```

4.1.2 Cryptography Algorithm

Five Cryptography algorithm such as blow fish, AES, Triple DES are implemented in this module in a for each part of the document separately. First the text sequence is applied to BlowFish algorithm along with the generated key and Initialization vector to obtain a cipher text. Similarly AES, TripleDES, IDEA and Fernet applied along with their keys and IV's and corresponding Encrypted files are obtained. All algorithms use CBC mode since plain text size is larger than that which an encryption algorithm generally accepts.

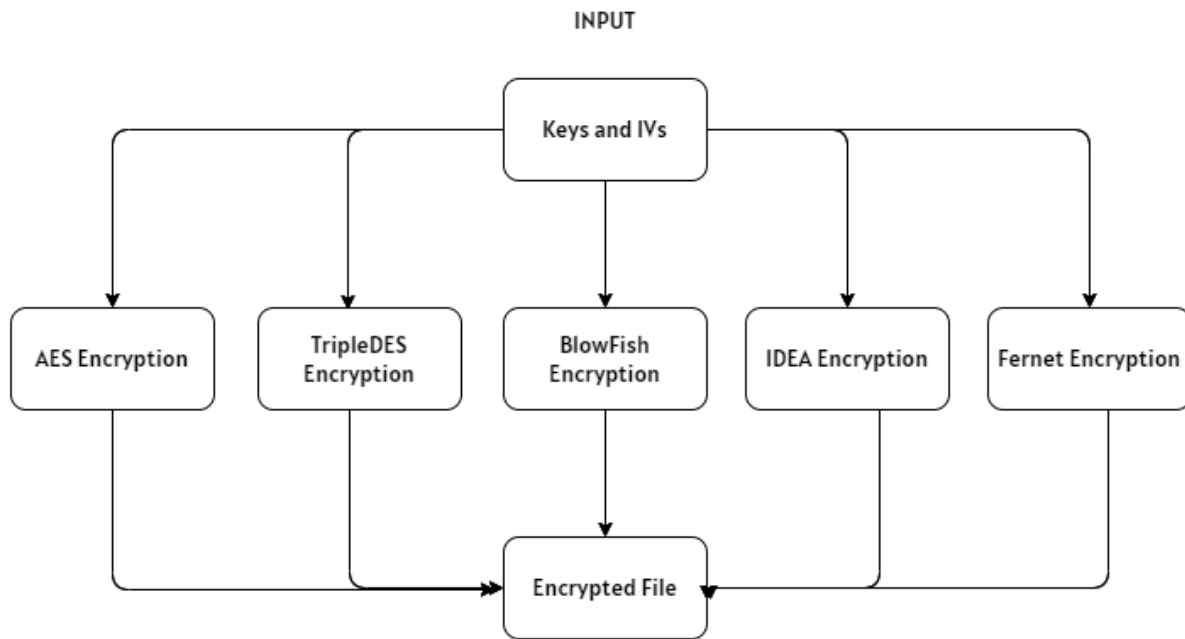


Fig.No.4.1.2 Module-2 Block diagram

Input: Text sequence of 5 parts, keys and Initialization vectors

Output: Encrypted files-cipher text.

Pseudo code:**Blowfish:**

```
cipher = Cipher(algorithms.Blowfish(key), modes.CBC(iv), backend=backend)
encryptor = cipher.encryptor()
cont = encryptor.update(content) + encryptor.finalize()
open(os.path.join("1.txt"), "w").close()
f=open(os.path.join("1e.txt"), "wb")
f.write(cont)
f.close()
```

AES

```
cipher = Cipher(algorithms.AES(key), modes.CBC(iv), backend=backend)
encryptor = cipher.encryptor()
cont = encryptor.update(content) + encryptor.finalize()
open("2.txt", "wb").close()
f=open("2e.txt", "wb")
f.write(cont)
f.close()
```

TripleDES

```
cipher = Cipher(algorithms.TripleDES(key), modes.CBC(iv), backend=backend);
encryptor = cipher.encryptor();
cont = encryptor.update(content) + encryptor.finalize();
open("3.txt", "w").close();
f=open("3e.txt", "wb");
f.write(cont);
f.close();
```

IDEA:

```
content=content.encode()
b=len(content)
if(b%8!=0):
while(b%8!=0):
content+=" ".encode()
b=len(content)
backend = default_backend()
cipher = Cipher(algorithms.IDEA(key), modes.CBC(iv), backend=backend)
encryptor = cipher.encryptor()
cont = encryptor.update(content) + encryptor.finalize()
open(os.path.join(os.getcwd()+"/Parts","3.txt"),"w").close()
f=open(os.path.join(os.getcwd()+"/Parts","3.txt"),"wb")
f.write(cont)
f.close()
```

FERNET:

```
content=f.read()
f.close()
content=content.encode('utf-8')
fer = Fernet(key)
content=fer.encrypt(content)
open(os.path.join(os.getcwd()+"/Parts","4.txt"),'w').close()
f=open(os.path.join(os.getcwd()+"/Parts","4.txt"),"wb")
f.write(content)
f.close()
```


4.1.3 Steganography and key Transfer

Least significant bit steganography technique is used with a selected cover image. The keys and initialization vectors used in crypto-algorithms are combined and hidden in the image using a cover image PNG file. The stegano-image where the keys are hidden in the least significant bit of the image along with the encrypted file is compressed into a single file and sent to the receiver in email.

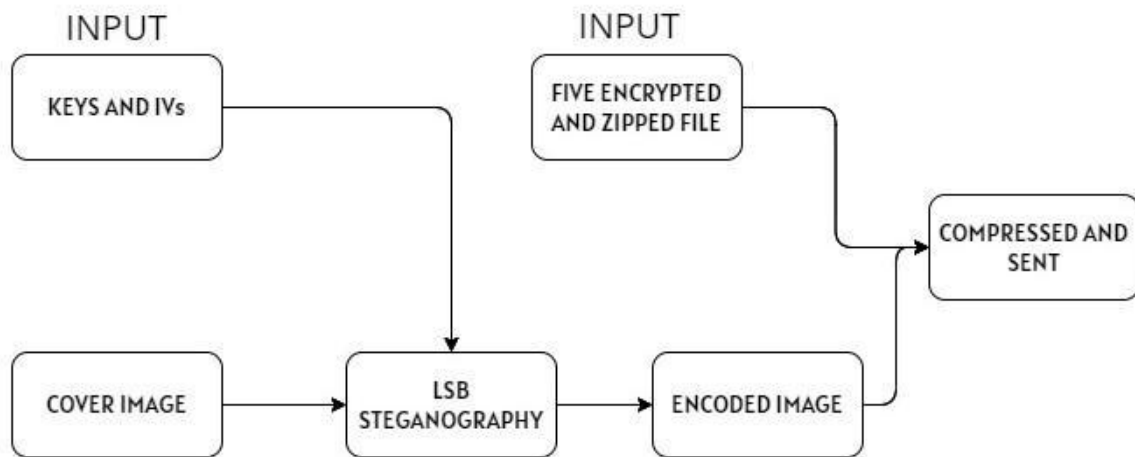


Fig.No.4.1.3 Module-3 Block diagram

Input: Encrypted files, keys, IV vector, Image.

Output: Compressed ZIP file-Enc files and encoded image.

Pseudo code:

Encryption of keys

```
hash = hashlib.sha1()
```

```
hash.update(password.encode())
```

```
password_encryption_cipher = AES.new( hash.hexdigest()[:16].encode() ,
AES.MODE_CBC, iv='16bitAESInitVect'.encode())
encrypted_keys_and_iv=hexlify(password_encryption_cipher.encrypt(pad(json.dumps(keys_iv).encode(), AES.block_size))))
```

LSB Steganography:

```
    r, g, b = to_binary(pixel)
if data_index < data_len:
    pixel[0] = int(r[:-1] + binary_secret_data[data_index], 2)
    data_index += 1
if data_index < data_len:
    pixel[1] = int(g[:-1] + binary_secret_data[data_index], 2)
    data_index += 1
if data_index < data_len:
    pixel[2] = int(b[:-1] + binary_secret_data[data_index], 2)
    data_index +=
if data_index >= data_len:
    break
```

4.1.4 Decryption and Extraction

The compressed File is unzipped to obtain Encrypted files. The Stegano-image received is decrypted to obtain the combined keys. The keys are separated and back to their form of original Key and IV. Then the keys and IV are given to encrypted files of Blowfish, AES, TripleDES, BlowFish and Fernet to decrypt the file into 5 parts.

The unzipped file contains five parts of encrypted file, these files are decrypted back using the keys and then the decrypted files are merged together in the appropriate order to get back the original file.

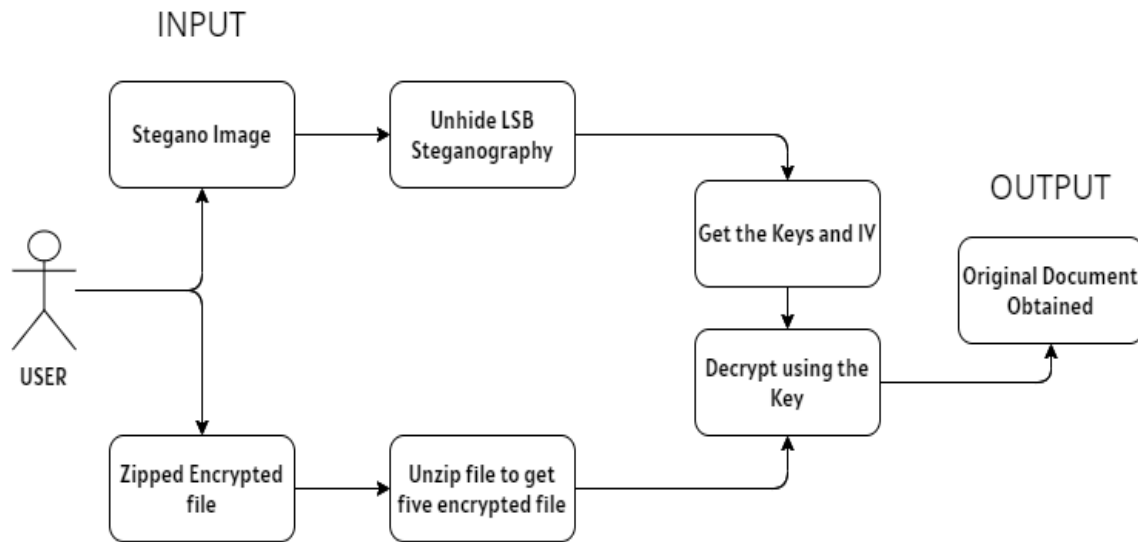


Fig.No.4.1.4 Module-4 Block diagram

Input: Compressed File-containing encoded image and encrypted file.

Output: Decrypted original file

Pseudo code:

Decode from Stegano-image:

```

for row in image:
    for pixel in row:
        r, g, b = to_bin(pixel)
        binary_data += r[-1]
        binary_data += g[-1]
        binary_data += b[-1]
  
```

```

all_bytes = [ binary_data[i: i+8] for i in range(0, len(binary_data), 8)
decoded_data = ""
for byte in all_bytes:
    decoded_data += chr(int(byte, 2))
    if decoded_data[-5:] == "=====":
        break

```

Decryption of keys:

```

hash = hashlib.sha1()
hash.update(password.encode())
password_decryption_cipher = AES.new( hash.hexdigest()[:16].encode() ,
AES.MODE_CBC, iv= '16bitAESInitVect'.encode())
decrypted_keys_iv=json.loads(unpad(password_decryption_cipher.decrypt(unhexli
fy(unhide_encrypted_keys_and_iv)), AES.block_size))

```

Decryption of Files:

```

def DAES(key,iv):
    f=open(os.path.join(os.getcwd()+"/zip","0.txt"),"rb")
    content=f.read()
    f.close()
    backend = default_backend()
    cipher = Cipher(algorithms.AES(key), modes.CBC(iv), backend=backend)
    decryptor = cipher.decryptor()
    content=decryptor.update(content) + decryptor.finalize()
    f=open(os.path.join(os.getcwd()+"/zip","0.txt"),"wb")
    f.write(content)
    f.close()

```

```

def DBlowFish(key,iv):
    f=open(os.path.join(os.getcwd()+"/zip","1.txt"),"rb")
    content=f.read()
    f.close()
    backend = default_backend()
    cipher = Cipher(algorithms.Blowfish(key), modes.CBC(iv), backend=backend)
    decryptor = cipher.decryptor()
    content=decryptor.update(content) + decryptor.finalize()
    f=open(os.path.join(os.getcwd()+"/zip","1.txt"),"wb")
    f.write(content)
    f.close()

```

```

def DTripleDES(key,iv):
    f=open(os.path.join(os.getcwd()+"/zip","2.txt"),"rb")
    content=f.read()
    f.close()
    backend = default_backend()
    cipher = Cipher(algorithms.TripleDES(key), modes.CBC(iv), backend=backend)
    decryptor = cipher.decryptor()
    content=decryptor.update(content) + decryptor.finalize()
    f=open(os.path.join(os.getcwd()+"/zip","2.txt"),"wb")
    f.write(content)
    f.close()

```

```

def DIDEA(key,iv):
    f=open(os.path.join(os.getcwd()+"/zip","3.txt"),"rb")
    content=f.read()

```

```

f.close()
backend = default_backend()
cipher = Cipher(algorithms.IDEA(key), modes.CBC(iv), backend=backend)
decryptor = cipher.decryptor()
content=decryptor.update(content) + decryptor.finalize()
open(os.path.join(os.getcwd()+"/zip","3.txt"),"wb").close()
f=open(os.path.join(os.getcwd()+"/zip","3.txt"),"wb")
f.write(content)
f.close()

```

```

def decrypt(token):
    return base64.decode(cipher_suite.decrypt(token, encoding="ascii"))""

```

```

def DFernet(key):
    f=open(os.path.join(os.getcwd()+"/zip","4.txt"),"rb")
    content=f.read()
    f.close()
    fer = Fernet(key)
    content=fer.decrypt(content)
    open(os.path.join(os.getcwd()+"/zip","4.txt"),"w").close()
    f=open(os.path.join(os.getcwd()+"/zip","4.txt"),"wb")
    f.write(content)
    f.close()

```

CHAPTER-5

IMPLEMENTATION AND RESULTS

5.1 Data Preprocessing

Random function is used to generate keys of 8 and 16 bits. It generates different set of keys of required length everytime.

5.1.1 Keys for 3 Algorithms:

```
b'?\9),-_<I{V/9,l@'  
b'?zfzpS:2PY0o4rCxn\\z^St79'
```

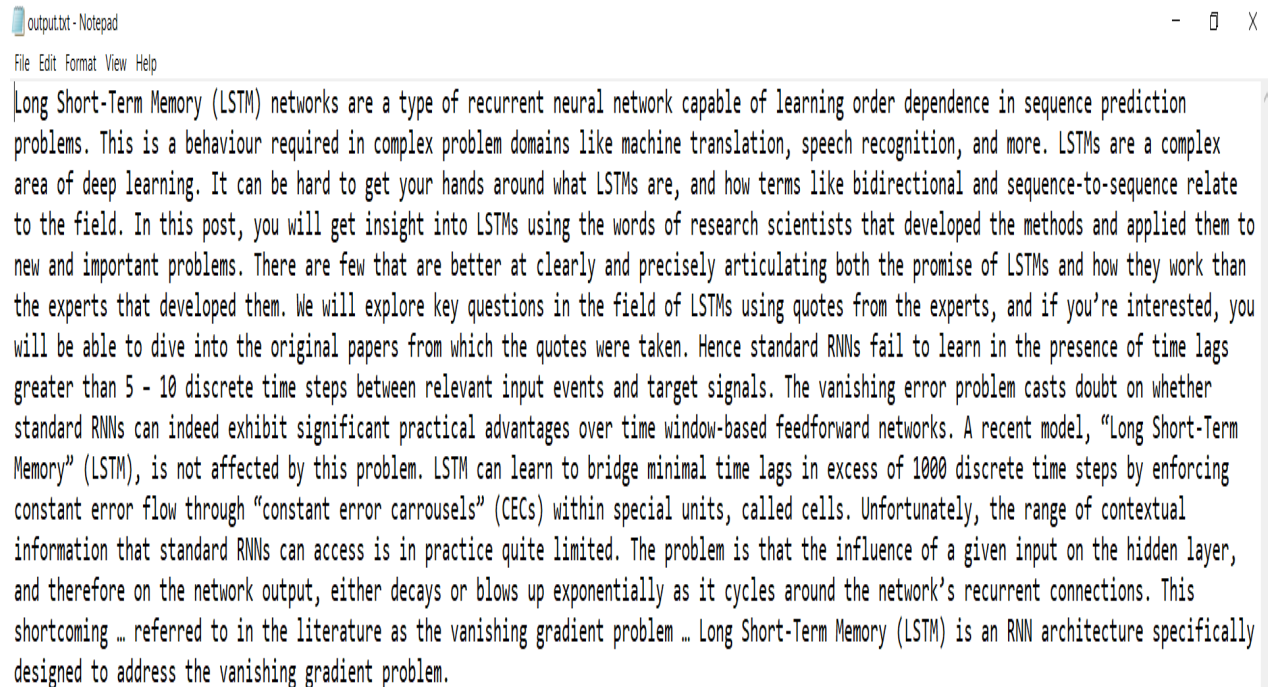
5.1.2 Initialization vectors generated:

```
b'\xe1\xf2z\xb0O\xfc\x10\x81t\x0\xafK\x1F'  
b'\xc4\xc2\x1d\x1c\xddAd'
```

5.1.3 Text document segmentation:

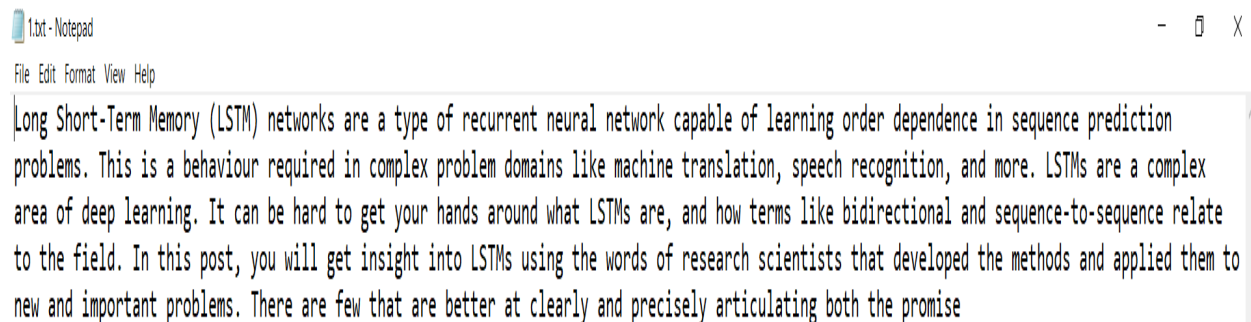
In order to perform hybrid cryptography (BLOWFISH, AES, TripleDES, IDEA and Fernet) we have split the document into 5 parts based on the character count. Below are the original document and the 5 parts namely “0.txt, 1.txt, 2.txt, 3.txt, 4.txt”.

Original Text file:



Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems. This is a behaviour required in complex problem domains like machine translation, speech recognition, and more. LSTMs are a complex area of deep learning. It can be hard to get your hands around what LSTMs are, and how terms like bidirectional and sequence-to-sequence relate to the field. In this post, you will get insight into LSTMs using the words of research scientists that developed the methods and applied them to new and important problems. There are few that are better at clearly and precisely articulating both the promise of LSTMs and how they work than the experts that developed them. We will explore key questions in the field of LSTMs using quotes from the experts, and if you're interested, you will be able to dive into the original papers from which the quotes were taken. Hence standard RNNs fail to learn in the presence of time lags greater than 5 - 10 discrete time steps between relevant input events and target signals. The vanishing error problem casts doubt on whether standard RNNs can indeed exhibit significant practical advantages over time window-based feedforward networks. A recent model, "Long Short-Term Memory" (LSTM), is not affected by this problem. LSTM can learn to bridge minimal time lags in excess of 1000 discrete time steps by enforcing constant error flow through "constant error carousels" (CECs) within special units, called cells. Unfortunately, the range of contextual information that standard RNNs can access is in practice quite limited. The problem is that the influence of a given input on the hidden layer, and therefore on the network output, either decays or blows up exponentially as it cycles around the network's recurrent connections. This shortcoming ... referred to in the literature as the vanishing gradient problem ... Long Short-Term Memory (LSTM) is an RNN architecture specifically designed to address the vanishing gradient problem.

Segmented File-1:



Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems. This is a behaviour required in complex problem domains like machine translation, speech recognition, and more. LSTMs are a complex area of deep learning. It can be hard to get your hands around what LSTMs are, and how terms like bidirectional and sequence-to-sequence relate to the field. In this post, you will get insight into LSTMs using the words of research scientists that developed the methods and applied them to new and important problems. There are few that are better at clearly and precisely articulating both the promise

Segmented File-2 :

2.txt - Notepad

- □ X

File Edit Format View Help

of LSTMs and how they work than the experts that developed them. We will explore key questions in the field of LSTMs using quotes from the experts, and if you're interested, you will be able to dive into the original papers from which the quotes were taken. Hence standard RNNs fail to learn in the presence of time lags greater than 5 - 10 discrete time steps between relevant input events and target signals. The vanishing error problem casts doubt on whether standard RNNs can indeed exhibit significant practical advantages over time window-based feedforward networks. A recent model, "Long Short-Term Memory" (LSTM), is not affected by this problem. LSTM can learn to bridge min

Segmented File-3:

3.txt - Notepad

- □ X

File Edit Format View Help

imal time lags in excess of 1000 discrete time steps by enforcing constant error flow through "constant error carrouseis" (CECs) within special units, called cells. Unfortunately, the range of contextual information that standard RNNs can access is in practice quite limited. The problem is that the influence of a given input on the hidden layer, and therefore on the network output, either decays or blows up exponentially as it cycles around the network's recurrent connections. This shortcoming ... referred to in the literature as the vanishing gradient problem ... Long Short-Term Memory (LSTM) is an RNN architecture specifically designed to address the vanishing gradient problem.

Segmented File-4 :

2.txt - Notepad

- □ X

File Edit Format View Help

of LSTMs and how they work than the experts that developed them. We will explore key questions in the field of LSTMs using quotes from the experts, and if you're interested, you will be able to dive into the original papers from which the quotes were taken. Hence standard RNNs fail to learn in the presence of time lags greater than 5 - 10 discrete time steps between relevant input events and target signals. The vanishing error problem casts doubt on whether standard RNNs can indeed exhibit significant practical advantages over time window-based feedforward networks. A recent model, "Long Short-Term Memory" (LSTM), is not affected by this problem. LSTM can learn to bridge min

Segmented File-5:

3.txt - Notepad

- □ X

File Edit Format View Help

imal time lags in excess of 1000 discrete time steps by enforcing constant error flow through "constant error carrouseles" (CECs) within special units, called cells. Unfortunately, the range of contextual information that standard RNNs can access is in practice quite limited. The problem is that the influence of a given input on the hidden layer, and therefore on the network output, either decays or blows up exponentially as it cycles around the network's recurrent connections. This shortcoming ... referred to in the literature as the vanishing gradient problem ... Long Short-Term Memory (LSTM) is an RNN architecture specifically designed to address the vanishing gradient problem.

5.2 Cryptography Algorithm

5.2.1 AES Encryption:

Second part “0.txt” is encrypted using Advanced encryption standard cryptography algorithm into “2e.txt”.

5.2.2 TripleDES Encryption

Third part “1.txt” is encrypted using Triple Data encryption Standard or Triple DES algorithm to get encrypted file “3e.txt”.

5.2.3 Blowfish Encryption:

1.txt is encrypted to “2.txt” file using blowfish. Keys are generated using key generator module.

5.2.2 IDEA Encryption:

Third part “3.txt” is encrypted using Triple Data encryption Standard or Triple DES algorithm to get encrypted file “3e.txt”.

5.2.3 Fernet:

1.txt is encrypted to “4.txt” file using blowfish. Keys are generated using key generator module.

5.3 Steganography and Key Transfer

5.3.1 Merge Keys

The Keys used for encryption at the various layers can be securely stored. The List of keys, L stores all the keys generated throughout the Data Encryption Process. Whenever the key for a particular Encryption Layer is generated, it is appended to the List of Keys, L.

In the system, the encryption layers are Blowfish, Triple-Des and AES, IDEA and Fernet respectively, so the Keys used, are as: List of Keys,

$L = [K_{\text{Blowfish}} \wedge \text{blowfish-iv} \wedge K_{\text{TripleDES}} \wedge \text{TripleDES-iv} \wedge K_{\text{AES}} \wedge \text{aes_iv} \wedge K_{\text{IDEA}} \wedge \text{IDEA-iv} \wedge K_{\text{FERNET}}]$

This List, L is then passed into a function that converts the list into a single string of keys separated by separators (x , * , /)

$LS = \text{hexlify}(L, \text{separator} = 0 \times 0)$

The String, LSB is then encrypted using the AES Encryption Algorithm with a Key generated from user-input password. The user inputs a password, PW which is hashed using SHA1, & the first 16 Bits of the Hash is used as the key KPassword . The Key, KPassword is used for the Encryption, generating the encrypted string LS-Encrypted.

HashedPassword, HP = SHA(PW) Key,

$K_{\text{Password}} = \text{HP} [0 : 16]$

$LS\text{-Encrypted} = \text{AES}(LS, K_{\text{Password}})$

It prompts you to enter a password to encrypt the keys using AES.

Keys encrypted by AES for authentication SHA is used.

5.3.2 LSB Steganography

The keys encrypted are converted to bits so that they can be placed in lsb of the image.



Fig 5.3.2 LSB Steganography Images

5.3.3 Encrypted data and Key Transfer

In order to transfer the encrypted files and steganography image, the files are compressed into a zip folder. Named Final.zip file. Later the receiver unzips the compressed file into a folder called temp where he gets 0.txt, 1.txt, 2.txt, 3.txt, 4.txt and the encoded image

Zip file contains all the encrypted five parts of the original file.

5.4 Decryption and Extraction of Data

5.4.1 Key Extraction from Stegano encoded Image

We have obtained the keys from the encoded image using reverse LSB steganography. They have been encrypted by us using AES encryption using a password.

We perform AES Decryption and extract our keys used for decrypting our data files. We have obtained the keys and Initialization vector used for encryption, since its symmetric cryptography algorithm is used to decrypt the data.

Keys for Algorithms obtained:

```
b'?\9),-_<I{V/9,l@'  
b'?zfzpS:2PY0o4rCxn\\z^St79'
```

Initialization vectors Obtained:

```
b'\xe1\xf2z\xb0O\xfc\r\x10\x81tI\xc0\xafK\xc1F'  
b'\xc4\xc2\x1d\x1c\xddAd'
```

5.4.2 Decryption of Data

The extracted keys and the encrypted files unzipped from final.zip which is placed folder temp is used to extract the original data.

5.4.2.1 Blowfish Decryption

The keys and iv vector used for encryption is used here since Blowfish is a Symmetric key cryptography technology same keys are used for both encryption and decryption. The decrypted plain file obtained from Cipher text is stored in Parts folder

5.4.2.2 AES Decryption

The keys and iv vector used for encryption is used here since Advanced Encryption standard is a Symmetric key cryptography technology same keys are used for both encryption and decryption. The decrypted plain file obtained from Cipher text is stored in “Parts” folder

5.4.2.3 TripleDES Decryption

The keys and iv vector used for encryption is used here since Triple Data Encryption standard is a Symmetric key cryptography technology same keys are used for both encryption and decryption. The decrypted plain file obtained from Cipher text is stored in “Parts” folder.

5.4.2.4 IDEA Decryption

The Keys and iv vector used for used for encryption is used here since IDEA is a Symmetric key cryptography technology same keys are used for both encryption and decryption. The decrypted plain file obtained from Cipher text is stored in “Parts” folder.

5.2.2.5 Fernet

The Keys and iv vector used for encryption is used here since Fernet is a Symmetric key cryptography technology same keys are used for both encryption and decryption. The decrypted plain file obtained from Cipher text is stored in “Parts” folder.

5.4.3 Output Data

Data decrypted from three cryptography algorithms such as Blowfish, Advanced Encryption standard, Triple Data encryption standard, Idea and Fernet into Five parts separately since we have applied hybrid cryptography each part is encrypted by separate algorithms now we have to combine those decrypted files so that we can obtain the original text back

Original Data :

Merged file is stored into a single text file called final.txt

CHAPTER-6

6.1 SYSTEM TESTING

In a generalized way, we can say that the system testing is a type of testing in which the main aim is to make sure that the system performs efficiently and seamlessly. The process of testing is applied to a program with the main aim to discover an unprecedented error, an error which otherwise could have damaged the future of the software. Test cases which bring up a high possibility of discovering and error are considered successful. This successful test helps to answer the still unknown errors.

6.1.1 TEST CASE

Testing, as already explained earlier, is the process of discovering all possible weak-points in the finalized software product. Testing helps to counter the working of sub-assemblies, components, assembly and the complete result. The software is taken through different exercises with the main aim of making sure that software meets the business requirement and user-expectations and doesn't fail abruptly. Several types of tests are used today. Each test type addresses a specific testing requirement.

6.1.2 Testing Techniques

A test plan is a document which describes the approach, its scope, its resources and the schedule of aimed testing exercises. It helps to identify almost other test item, the features which are to be tested, its tasks, how will everyone do each task, how much the tester is independent, the environment in which the test is taking place,

its technique of design plus the both the end criteria which is used, also rational of choice of theirs, and whatever kind of risk which requires emergency planning. It can be also referred to as the record of the process of test planning. Test plans are usually prepared with significant input from test engineers.

(I) UNIT TESTING

In unit testing, the design of the test cases is involved that helps in the validation of the internal program logic. The validation of all the decision branches and internal code takes place. After the individual unit is completed it takes place. Plus it is taken into account after the individual unit is completed before integration. The unit test thus performs the basic level test at its component stage and tests the particular business process, system configurations etc. The unit test ensures that the particular unique path of the process gets performed precisely to the documented specifications and contains clearly defined inputs with the results which are expected.

(II) INTEGRATION TESTING

These tests are designed to test the integrated software items to determine whether they really execute as a single program or application. The testing is event driven and thus is concerned with the basic outcome of the field. The Integration tests demonstrate that the components were individually satisfied, as already represented by successful unit testing, the components are apt and fine. This type of testing is specially aimed to expose the issues that come-up by the components combination.

(III) FUNCTIONAL TESTING

The functional tests help in providing the systematic representation that functions tested are available and specified by technical requirement, documentation of the system and the user manual.

(IV) SYSTEM TESTING

System testing, as the name suggests, is the type of testing which ensures that the software system meets the business requirements and aims. Testing of the configuration is taken place here to ensure predictable result and thus analysis of it. System testing relies on the description of process and its flow, stressing on pre driven process and the points of integration.

(V) WHITE BOX TESTING

The white box testing is the type of testing in which the internal components of the system software are open and can be processed by the tester. It is therefore a complex type of testing process. All the data structure, components etc. are tested by the tester himself to find out a possible bug or error. It is used in situations in which the black box is incapable of finding out a bug. It is a complex type of testing which takes more time to get applied.

(VI) BLACK BOX TESTING

The black box testing is the type of testing in which the internal components of the software are hidden and only the input and output of the system is the key for the tester to find out a bug. It is therefore a simple type of testing. A programmer with basic knowledge can also process this type of testing. It is less time consuming as compared to the white box testing.

(V) ACCEPTANCE TESTING

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

(VI) ALPHA TESTING

Alpha testing is performed at the developer's site by the customer in a closed environment. This is done after the system testing. Alpha testing is one of the most common software testing strategies used in software development. It's specially used by product development organizations. Alpha testing is typically performed by a group that is independent of the design team, but still within the company, e.g. in-house software test engineers, or software QA engineers. Alpha testing is final testing before the software is released to the general public.

(VII) BETA TESTING

It is also known as field testing. It takes place at customer's sites. It sends the system to users who install it and use it under real-world working conditions. A beta test is the second phase of software testing in which a sampling of the intended audience tries the product out.

(VIII) SECURITY TESTING

Security testing is basically a type of software testing that's done to check whether the application or the product is secured or not. It checks to see if the application is vulnerable to attacks, if anyone hack the system or login to the application without any authorization. It is a type of non functional testing.

CHAPTER-7

7.1 Conclusion

The proposed cryptosystem uses a combination of symmetric cryptography to secure data. The system also introduces a sub-process to encrypt the keys used for encryption before embedding them in an image. The combination of Blowfish, AES, TripleDES, IDEA and Fernet has significantly improved the security and also ensured that the drawbacks of the standalone systems are addressed. The system also helps in improving security without the use of keys of larger lengths. We have also seen from the test results that the system is less susceptible to brute force attacks as the decryption time is significantly high. The manyfold expansion of plaintext into ciphertext also helps in ensuring a high level of security. While the system successfully does its intended work, it still required minor improvements for larger adoption. The system also helps in improving security without the use of keys of larger lengths.

7.2 Future Enhancement

The proposed system is very secure and robust. It has proven to encrypt data and ensure key security. While it is efficient and secure, it was also seen that the encrypted files are generally 2-3 times the size of the original file, hence the encrypted file takes up a significant amount of space to store. This drawback can be addressed by studying it further and making changes to the proposed system. Another improvement that can be made is by making the encryption and decryption time lesser. Further research on the proposed system can also be done by analyzing different order of combinations of the three algorithms used. a slightly different combination can also be studied by replacing one of the algorithms for improved performance. System is now designed only for text based files. Future work is to make the system capable of supporting audio, video, many other file formats.

7.3 Performance Evaluation

The performance of the Proposed system and the Existing system is calculated based on the time taken for the execution, block size of the given data, complexity of algorithm to withstand Brute-force attack, etc.,

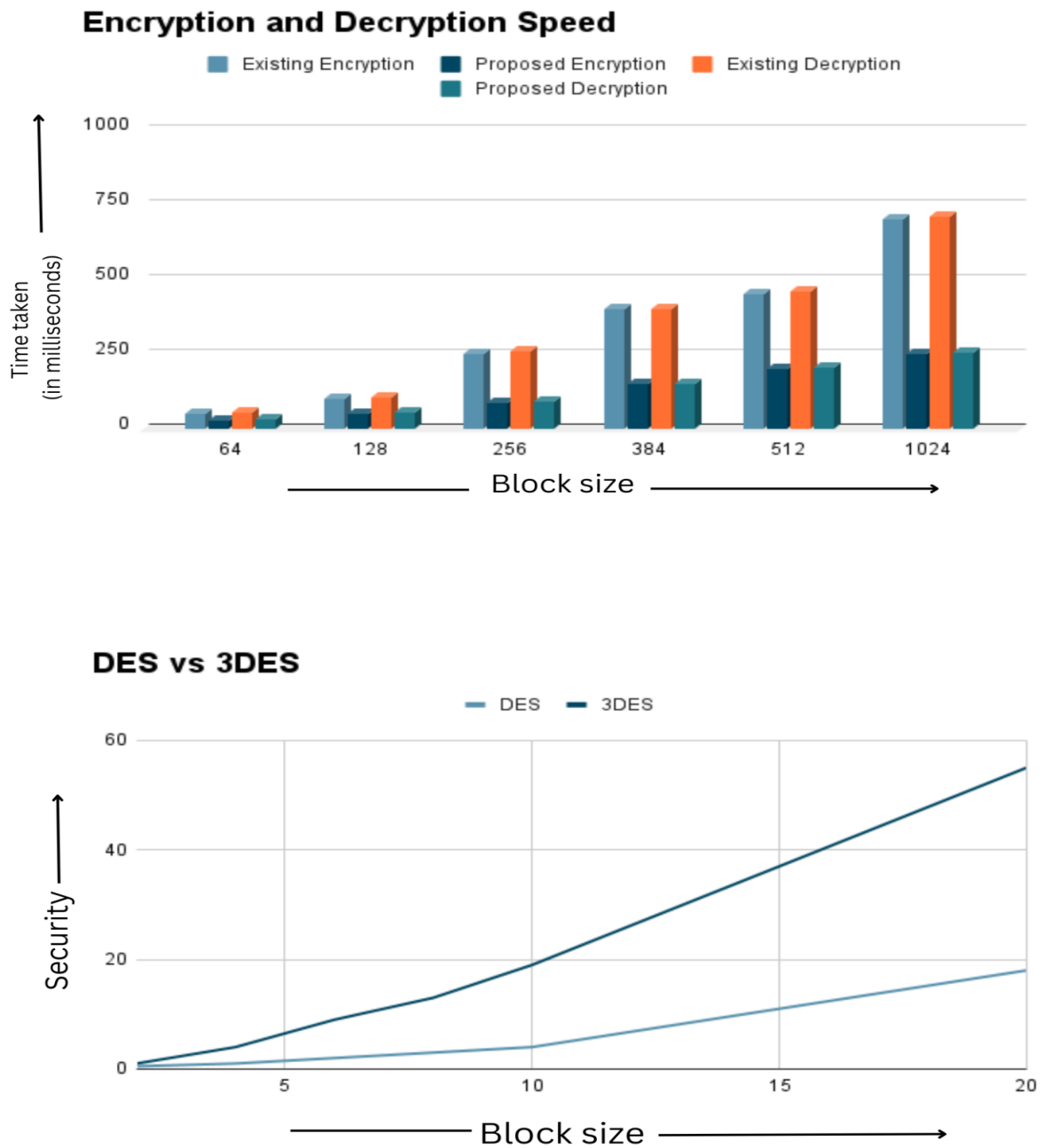


Fig 7.3.1 Performance Evaluation Graph

The above graph represents the Encryption and Decryption Speed based on the Block size and the time taken to execute and the hardness to decrypt the DES and the TripleDES.

7.4 OUTPUT



Fig 7.4.1 Output

REFERENCES

- [1] Chinnasamy, P., Padmavathi, S., Swathy, R., & Rakesh, S. (2021).
Efficient Data Security Using Hybrid Cryptography on Cloud Computing.
In Inventive Communication and Computational Technologies
(pp. 537-547). Springer, Singapore.

- [2] Rose Adeel and Haralambos Mouratidis (2022) “A Dynamic Four-Step
Data Security Model for Data in Cloud Computing Based on Cryptography
and Steganography”Sensors 1109. <https://doi.org/10.3390/s22031109>.

- [3] Secure File Storage Using Hybrid Cryptography: 2021 6th
International Conference on Communication and Electronics Systems
(ICCES) Authors:Putta Bharathi, Gayathri Annam, Jaya Bindu,
Kandi Vamsi,Krishna Duggana and Anjali.T.

- [4] Exploring LSB Steganography Possibilities in RGB Images:
2021 12th International Conference on Computing Communication
and Networking Technologies Authors: Rutvik Dumre and Aashka Dave.

- [5] Combined Cryptography and Steganography for Enhanced Security
in Suboptimal Images: 2020 International Conference on
Artificial Intelligence and Signal Processing (AISP)Authors:S. Joseph
Gladwin and Pasumarthi Lakshmi Gowthami

- [6] Suresh, S. R. (2021). An Electronic Digital Library Using Integrated Security Methods and Cloud Storages. *International Journal of Advanced Networking and Applications*, 13(1), 4839-4844.
- [7] Karati, A., Amin, R., Mohit, P., Suresh Kumar, V., & Biswas, G. P. (2021). Design of a secure file storage and access protocol for cloud-enabled Internet of Things environment. *Computers & Electrical Engineering*, 94, 107298.
- [8] Viswanath, G., & Krishna, P. V. (2021). Hybrid encryption framework for securing big data storage in multi-cloud environments. *Evolutionary Intelligence*, 14(2), 691-698.
- [9] Hossein Abroshan, “A Hybrid Encryption Solution to Improve Cloud Computing Security using Symmetric and Asymmetric Cryptography Algorithms”, *International Journal of Advanced Computer Science and Applications*, Vol. 12, No. 6, 2021.
- [10] Pravin Soni, Rahul Malik (2021) “A Hybrid Cloud Security Model for Securing Data on Cloud” *WCNC-2021: Workshop on Computer Networks & Communications*, Chennai, India.