CIPHER VAULT

A PROJECT SUBMITTED
IN FULFILMENT OF THE REQUIREMENT
FOR THE AWARD OF THE DEGREE

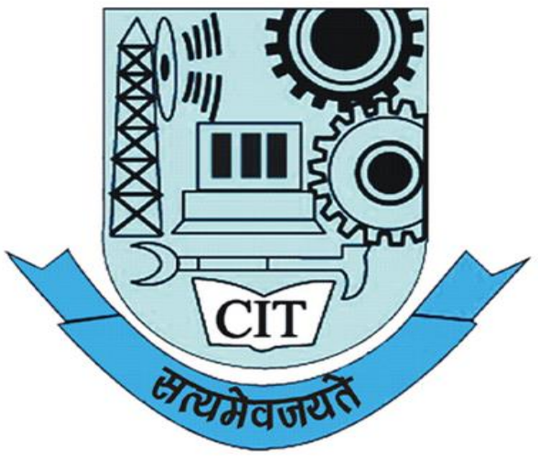
OF

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE & ENGINEERING

BY

Aman Kumar Singh 19050445001

Raghav Kumar Singh 19050445015



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
CAMBRIDGE INSTITUTE OF TECHNOLOGY
TATISILWAI, RANCHI – 835103.
(2022)

DECLARATION CERTIFICATE

This to certify that the work presented in the project entitled **Cipher Vault** in fulfillment of the requirement for award of the degree of Bachelor of Technology in Computer Science & Engineering of Cambridge Institute of Technology, Tatisilwai, Ranchi is an authentic work carried out under my supervision and guidance.

To the best of my knowledge, the content of this project is original.

Date:

Guide's Name
Prof. Deepak Kumar Verma
Dept. of Computer Science & Engineering
Cambridge Institute of Technology.
Tatisilwai, Ranchi – 835103

Head of Department
Prof. Arshad Usmani
Dept. of Computer Science &Engineering
Cambridge Institute of Technology.
Tatisilwai, Ranchi – 835103

CERTIFICATE OF APPROVAL

The foregoing project entitled **Cipher Vault** is hereby approved as a creditable work on the topic and has been presented in satisfactory manner to warrant its acceptance as prerequisite to the degree for which it has been submitted.

It is understood that by this approval, the undersigned do not necessarily endorse any conclusion drawn or opinion expressed therein, but approve the project for the purpose for which it is submitted.

(**Internal Examiner**) (**External Examiner**)

Head of Department

Department of Computer Science & Engineering

Cambridge Institute of Technology

Tatisilwai, Ranchi – 835103

Acknowledgement

I would like to express my heartfelt gratitude to everyone who supported me throughout the development of "Cipher Vault". This project wouldn't have been possible without the encouragement and guidance of my family, friends, and mentors.

I would also like to extend my thanks to the team members who worked tirelessly on this project with me, bringing their unique perspectives and expertise to the table. Together, we created a secure and user-friendly application that we can all be proud of.

Finally, I would like to thank the users who have downloaded and used "Cipher Vault". Your feedback and support have been instrumental in helping us improve and refines the application. We hope that Cipher Vault will continue to be a valuable tool for anyone seeking to keep their sensitive information safe and secure.

We take this opportunity to thank all who have contributed towards shaping this project. I would like to express my sincere thanks to my guide Prof. Deepak Kumar Verma, Department of Computer Science & Engineering, whose invaluable suggestions helped me in development of this project. As my supervisor, he has constantly encouraged me to remain focused on achieving my goal.

I would also like to thank all faculty and technical staff members of the Department who have been kind enough to advise and help in their respective roles. I have been fortunate to have wonderful support structure among the graduate students. I am really thankful to my friends. My sincere thanks to everyone who has provided me with kind words, new ideas, useful criticism, or their invaluable time, I am truly indebted.

Abstract

"Cipher Vault" is a secure and user-friendly application designed to help users keep their sensitive information safe and secure. The application employs advanced encryption algorithms to ensure that user data is protected at all times.

The application allows users to store passwords, credit card information, and other sensitive data securely in a digital "vault". Access to the vault is protected by a master password and multi-factor authentication, making it nearly impossible for unauthorized users to gain access.

In addition to providing a secure storage solution, Cipher Vault also includes features such as password generation and autofill, making it easy for users to create strong passwords and fill them in automatically when logging into websites or applications.

Overall, Cipher Vault offers a comprehensive solution for anyone seeking to protect their sensitive information from prying eyes. With its powerful encryption, user-friendly interface, and robust features, Cipher Vault is a valuable tool for anyone looking to stay safe and secure online.

LIST OF FIGURES

TABLE OF CONTENTS

1.Introduction

1.1Introduction

In today's digital age, information security is paramount. With increasing reliance on technology, companies and individuals are generating and storing vast amounts of sensitive data. To protect this data, they require secure and reliable methods for encryption, storage, and access control.

Digital communication witnesses a noticeable and continuous development in many applications in the Internet. Hence, secure communication sessions must be provided. The security of data transmitted across a global network has turned into a key factor on the network performance measures. So, the confidentiality and the integrity of data are needed to prevent eavesdroppers from accessing and using transmitted data. Steganography and Cryptography are two important techniques that are used to provide network security.

One such solution is Project Cipher Vault, a cutting-edge data encryption and storage platform that provides advanced security features to its users. The project is designed to ensure the confidentiality, integrity, and availability of data through a combination of cryptographic techniques, secure storage, and access control mechanisms.

At its core, Cipher Vault uses advanced encryption algorithms to protect data from unauthorized access. The system employs a multi-layered approach, with each layer providing an additional level of security. This includes encryption of data at rest, in transit, and in use. The encryption keys are managed by the platform, ensuring that they are secure and tamper-proof.

Moreover, Cipher Vault provides secure storage for the encrypted data. The platform uses distributed storage technology to store data across multiple servers, reducing the risk of data loss or corruption. The system is designed to be highly scalable, allowing it to handle large volumes of data.

In addition, Cipher Vault provides robust access control mechanisms to ensure that only authorized users can access the encrypted data. The system supports multi-factor authentication, which requires users to provide multiple forms of identification to access their data. It also provides role-based access control, which allows administrators to define access levels based on a user's role within an organization.

Overall, Project Cipher Vault is a state-of-the-art platform that provides advanced security features to protect sensitive data. Its multi-layered approach to encryption, secure storage, and access control ensures the confidentiality, integrity, and availability of data, making it an ideal solution for organizations and individuals who require the highest levels of security for their data.

1.1.1 Cryptography

Cryptography is one of the traditional methods used to guarantee the privacy of communication between parties. This method is the art of secret writing, which is used to encrypt the plaintext with a key into ciphertext to be transferred between parties on an insecure channel. Using a valid key, the ciphertext can be decrypted to the original plaintext. Without the knowledge of the key, nobody can retrieve the plaintext. Cryptography plays an essential role in many factors required for secure communication across an insecure channel, like confidentiality, privacy, non-repudiation, key exchange, and authentication.

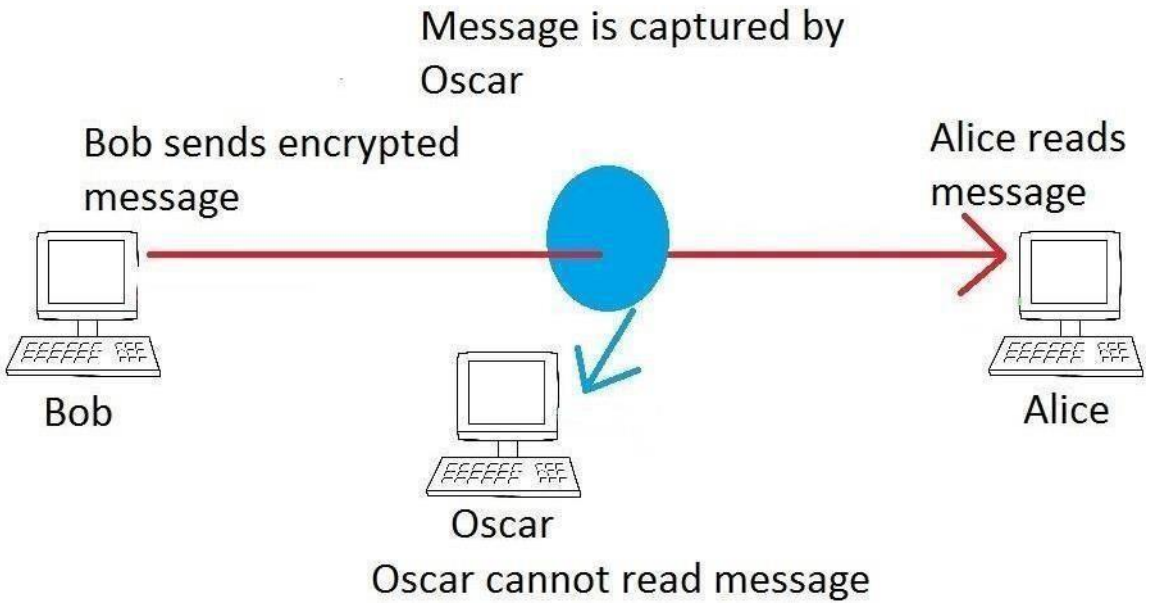


Fig 1.1.1 : Cryptography as a flow model.

1.1.1.1 Symmetric / Secret Key Cryptography

The technique of Secret key encryption can also be known as the symmetric-key, shared key, single-key, and eventually private-key encryption. The technique of private key uses for all side's encryption and decryption of secret data. The original information or plaintext is encrypted with a key by the sender side also the

similarly key is used by the receiver to decrypt a message to obtain the plaintext. the key will be known only by a people who are authorized to the encryption/decryption. However, the technique affords the good security for transmission but there is a difficulty with the distribution of the key. If one stole or explore the key he can get whole data without any difficulty. An example of Symmetric-Key is DES Algorithm.

1.1.1.2 Asymmetric / Public Key Cryptography

We can call this technique as asymmetric cryptosystem or public key cryptosystem, this technique uses two keys which are mathematically associated, use separately for encrypting and decrypting the information. In this technique, when we use the private key, there are no possibilities to obtain the data or simply discover the other key. The key used for encryption is stored public therefore it's called public key, and the decryption key is stored secret and called private key. An example of Asymmetric-Key Algorithm is RSA.

1.1.2 Steganography

It can be defined as the science of hiding and communicating data through apparently reliable carriers in attempt to hide the existence of the data. So, there is no knowledge of the existence of the message in the first place. If a person views the cover which the information is hidden inside, he or she will have no clue that there is any covering data, in this way the individual won't endeavor to decode the data. The secret information can be inserted into the cover media by the stego system encoder with using certain algorithm. A secret message can be plaintext, an image, ciphertext, or anything which can be represented in form of a bitstream. after the secret data is embedded in the cover object, the cover object will be called as a stego object also the stego object sends to the receiver by selecting the suitable channel, where decoder system is used with the same stego method for obtaining original information as the sender would like to transfer .

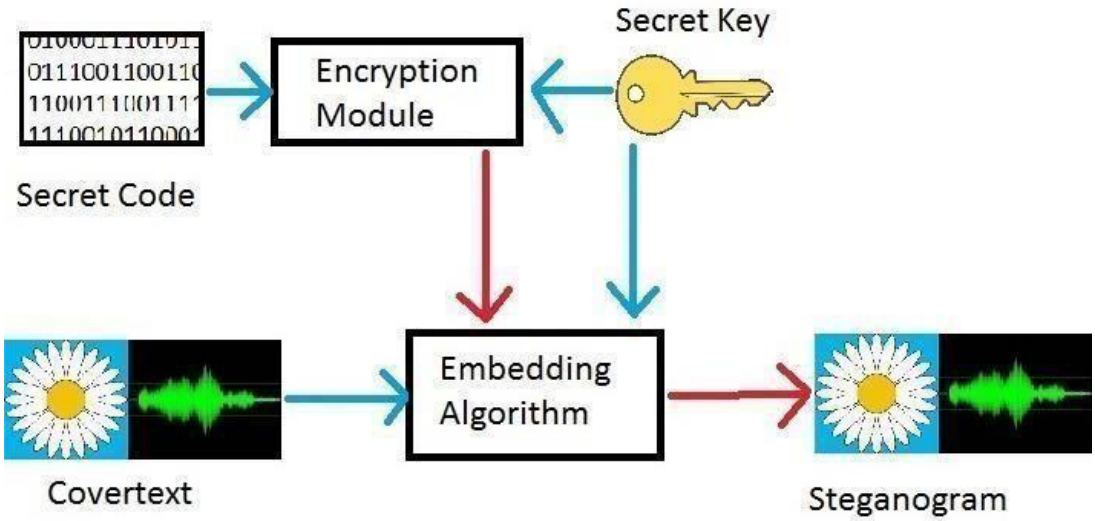


Fig 1.1.2 : Stegonography as a flow model.

1.1.3 Types of Steganography

There are various types of steganography.

A. Text Files

The technique of embedding secret data inside a text is identified as text stego. Text steganography needs a low memory because this type of file can only store text files. It affords fast transfer or communication of files from a sender to receiver.

B. Image Files

It is the procedure in which we embed the information inside the pixels of image. So, that the attackers cannot observe any change in the cover image. LSB approach is a common image steganography algorithm.

C. Audio Files

It is the process in which we hide the information inside an audio. There are many approaches to hide secret information in an audio files for examples Phase Coding, LSB .

D. Video Files

It is the process of hiding some secret data inside the frames of a video.

1.1.4 Steganography versus Cryptography

Steganography and cryptography are used for the purpose of data transmission over an insecure network without the data being exposed to any unauthorized persons. Steganography embeds the data in a cover image while cryptography encrypts the data. The advantage of Steganography is that, the look of the file isn't changed and this it will not raise any doubt for the attacker to suspect that there may be some data hidden unlike cryptography that encrypts the data and sends it to network.

1.1.5 Benefits of Steganography and Cryptography

It is noted that steganography and cryptography alone is insufficient for the security of information, therefore if we combine these systems, we can generate more reliable and strong approach. The combination of these two strategies will improve the security of the information. This combined will fulfill the prerequisites, for example, memory space, security, and strength for important information transmission across an open channel. Also, it will be a powerful mechanism which enables people to communicate without interferes of eavesdroppers even knowing there is a style of communication in the first place.

1.1.6 Applications of Steganography

(i)**Secret Communication :** Steganography does not advertise secret communication and therefore avoids scrutiny of the sender message. A trade secret, blueprint, or other sensitive information can be transmitted without alerting potential attackers.

(ii) **Feature Tagging :** Elements can be embedded inside an image, such as the names of the individuals in a photo or location in a map. Copying the stego image also copies all of the embedded features and only parties who possess the decode stego key will be able to extract and view the features.

(iii) **Copyright Protection :** Copy protection mechanisms that prevent the data, usually digital data from being copied. The insertion and analysis of water marks to protect copyrighted material is responsible for the percent rise of interest digital

steganography and data embedding.

1.2 MOTIVATION FOR THE WORK

Motivation is very important function for any project. It is one of the methods to induce the man on the job to get the work done effectively to have the best results towards the common objectives. It is necessary for the better performance.

Motivation can be seen as the inner drive, which prompts people to act in a way either towards achieving their personal goals or organizational goals. To a large extent, motivation is "leadership" as it involves getting the whole staff to learn to work willingly and well in the interest of the business. A leader can influence his subordinate only when they are convinced.

Conviction can only come when the entire subordinate accepts those factors that propel actions of individuals, which are referred to as motivation. They may be highly paid, prestigious titles promotion, praises, bonus, etc. The word is an abstract noun applying to the entire class of desired need wishes and similar forces. Motivation has to do with action which results, to satisfaction closely associated with motivation is the word "miracle" it is injecting of moral and loyalty into the working team so that they will carry their duties properly and effectively with maximum economy.

The main reason and motivation for choosing this project is, Due to recent developments in stego analysis, providing security to personal contents, messages, or digital images using steganography has become difficult. By using stego analysis, one can easily reveal existence of hidden information in carrier files. So, after been exposed to such problems it motivated us to do this project where the complete process of transferring of information is done using two different techniques. All that is required is to select a cover image and transfer the information using that image.

1.3 PROBLEM STATEMENT

The purpose of this project is to provide the correct data with security to the users. For some of the users the data might be lost during the transmission process in the network and for some, the data might be changed by the unauthorized person in the network and there are some other security problems in the network. Our application will give you more Security to the data present in the network and there will be able to reduce the loss of data in the network which will be transmitted from the sender to the receiver using the latest technologies. Only the Authorized persons i.e., who are using our application will be there in the Network. The proposed algorithm is to hide the audio data effectively in an image

without any suspicion of the data being hidden in the image. It is to work against the attacks by using a distinct new image that isn't possible to compare.

The aim of the project is to hide the data in an image using steganography and ensure that the quality of concealing data must not be lost.

We used a method for hiding the data in a distinct image file in order to securely send over the network without any suspicion the data being hidden. This algorithm, though requires a distinct image which we can use as a carrier and hide the data which is well within the limits of the threshold that the image can hide, that will secure the data.

2.LITERATURE SURVEY

As we said the significance of network security is increased day by day as the size of data being transferred across the Internet. This issue pushes the researchers to do many studies to increase the ability to solve security issues. A solution for this issue is using the advantage of cryptography and steganography combined in one system. Many studies propose methods to combine cryptography with steganography systems in one system. This Project has been implemented on the basis of the requirements of security i.e. authentication, confidentiality, and robustness.

There has been a continuous rise in the number of data security threats in the recent past and it has become a matter of concern for the security experts. Cryptography and steganography are the best techniques to nullify this threat. The researchers today are proposing a blended approach of both techniques because a higher level of security is achieved when both techniques are used together.

In proposed an encrypting technique by combining cryptography and steganography techniques to hide the data. In cryptography process, we proposed an effective technique for data encryption using one's complement method. It used an Asymmetric key method where both sender and receiver share the Secret key for encryption and decryption. In steganography part, we used the LSB method that is used and mostly preferred.

We present a method based on combining both the strong encrypting algorithm and steganographic technique to make the communication of confidential information safe, secure and extremely hard to decode. An encryption technique is employed for encrypting a secret message into a Cipher text using the Senders Private Key and receiver public key. The Cipher Text is finally embedded in a suitable cover

image and transferred securely to deliver the secret information. They utilized a least significant bit method to accomplish the digital image steganography.

At the receiver's side, the secret data is retrieved through the decoding process. Thus, a three-level security has been rendered for them a secret message to be transferred.

1. METHODOLOGY

3.1 SYSTEM ARCHITECTURE

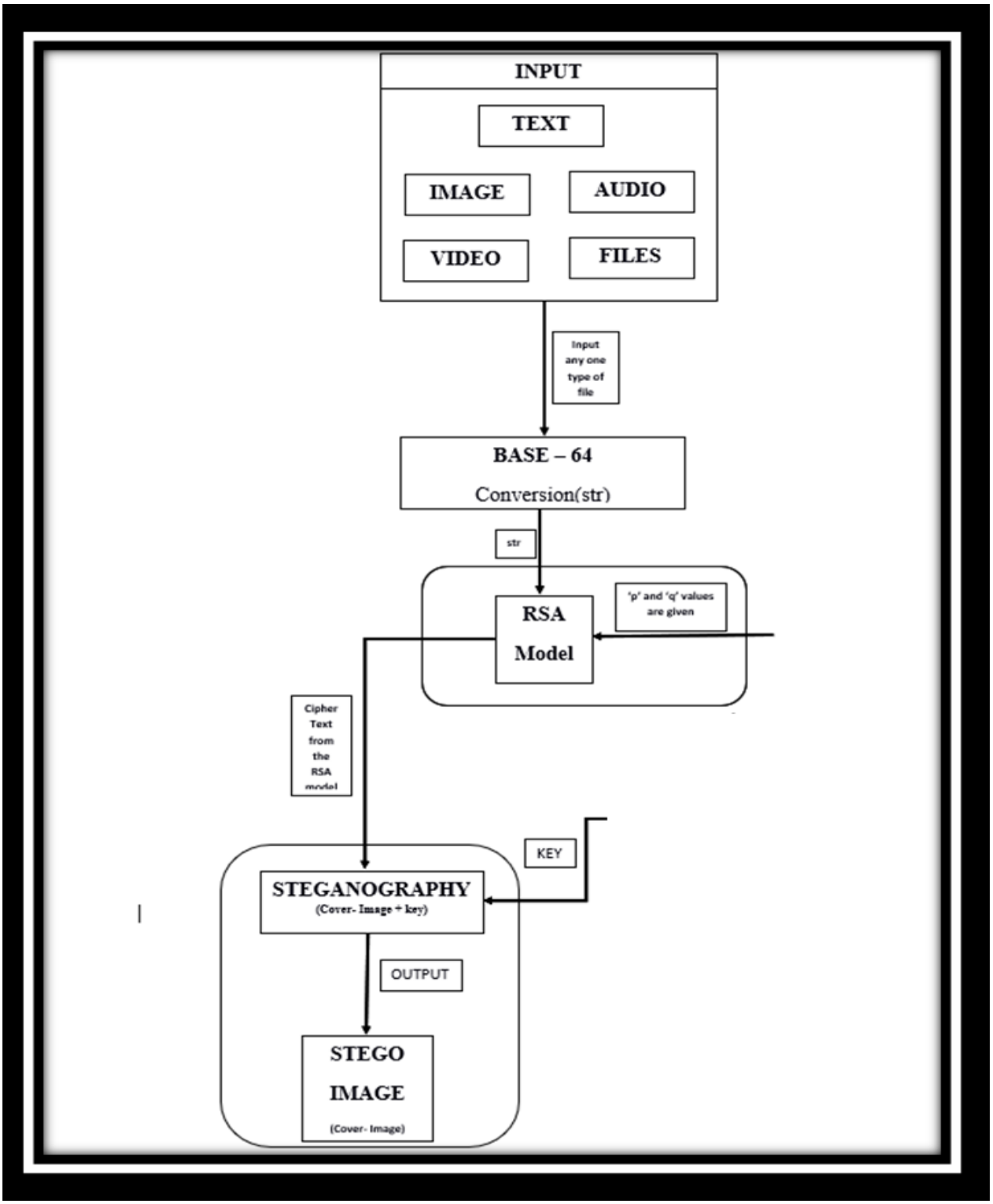
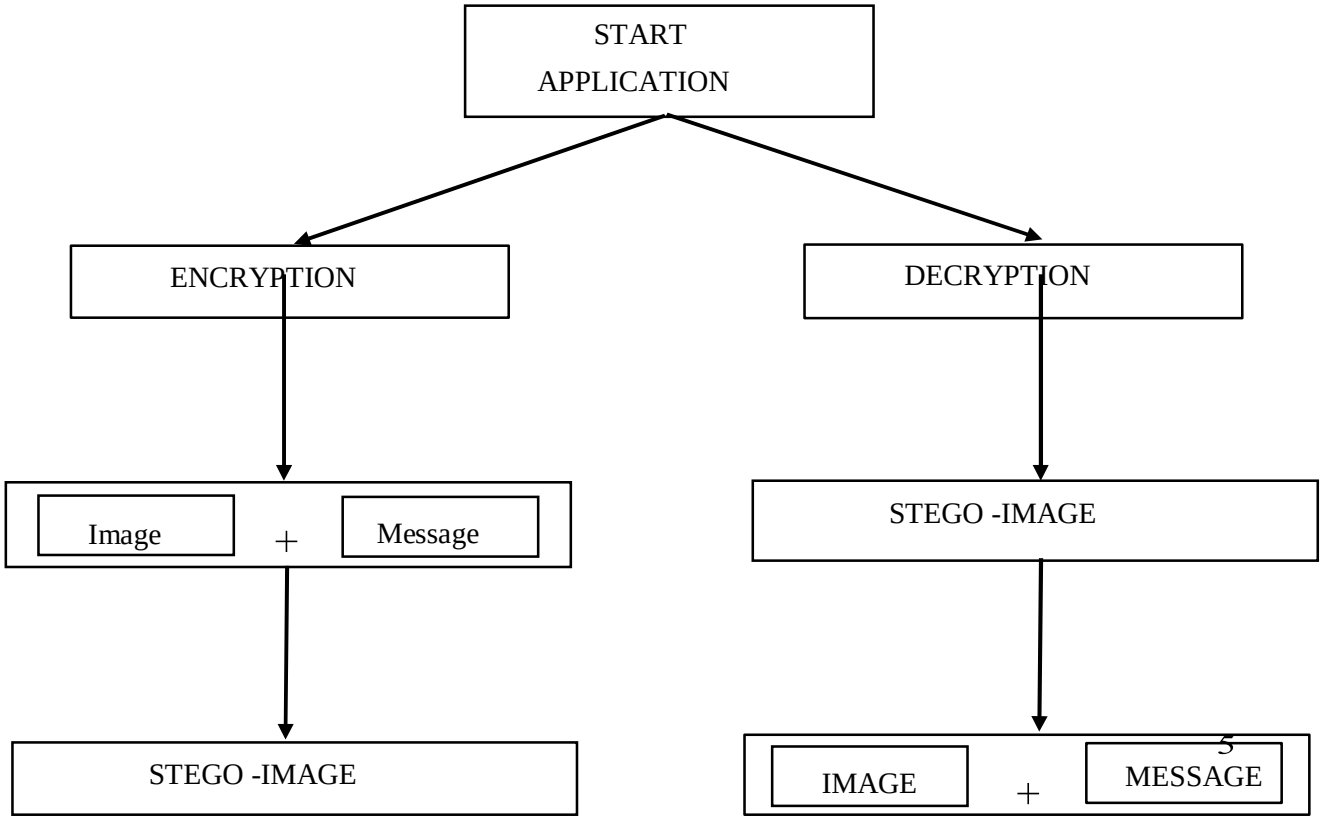


Fig 3.1 System Architecture

3.2 Proposed System

In this section, we will discuss proposed method which combines two different hiding techniques, which are Cryptography and Steganography. In this proposed method first, the message is encrypted by use RSA algorithm. After that, we use the modified LSB technique to embed the encrypted information in image. So, this technique combines the features of both cryptography and steganography and provides a high level of security. It is better than either of the technique used separately. There will be an agreement between the sender and the receiver about the key for the concealment algorithm as well as the key for the encryption algorithm or these keys may be exchanged by a secure communication method. Our method starts by encryption first then hide encrypted data.



Before applying the cryptography and steganography, initially we convert our input to Base-64. And we save the obtained text in a text file. Then we proceed to cryptography and steganography.

3.2.1 Sender Side

The Sender side consists of cryptographic and steganography stages. This method starts with cryptographic then steganography.

Cryptography Stage :

In encryption stage, we use RSA (Rivest Shamir Adelson) algorithm. This technique takes two prime numbers. The Encryption can be done using the Plain Text and with "e" values which was generated using the two prime numbers. Then we will get a cipher text, which is communicated to the receiving end for decryption. This encrypted data will be used in steganography stage.

Input = Message + Two Prime No.

Output = Encrypted Message

Steganography Stage :

In stenography stage, we use LSB (Least Significant Bit) algorithm with some modification to hide information (encrypted data from cryptography stage) inside a cover. In our experiment, we use the image as cover to present our method, but this method can be applied to other files such as audio, and video. The general LSB method used to hide secret information into a file; the last bit in each pixel or sample or frame used sequentially to hide one of the binary stream bits Encryption of the cover image.

Input= Encrypted Message + Secret key+ cover image.

Output= Stego-Image.

3.2.2 Receiver side

Receiver side consists of steganography and cryptography stages. In receiver side we will first extract embedded data then decrypt it.

Steganography Stage :

In the receiver side, we start with steganography then cryptography. We will use the same steps which are used in sender side.

Input= Stego-Image+ Secret Key.
Output= Encrypted Message.

Cryptography Stage :

In cryptography stage, we use the data which is extracted from stego file and use RSA. We will use the same steps which are used in sender side. The Decryption can be done using the Encrypted message, receivers private key and senders public key.

Input= Encrypted Message + 2 Prime Numbers.
Output= Plain Text.

Now the Plain Text is in the form of Base-64. After getting the plain text apply Base-64 conversion to change the Plain-text to given input, which can be Text, Image, Video, Audio.

3.3 MODULE DIVISION

3.3.1 Base-64

Base 64 is an encoding scheme that converts binary data into text format so that encoded textual data can be easily transported over network un-corrupted and without any data loss. Base64 is used commonly in a number of applications including email via MIME, and storing complex data in XML. Problem with sending normal binary data to a network is that bits can be misinterpreted by underlying protocols, produce incorrect data at receiving node and that is why we use this method. The term Base64 is taken from the Multipurpose Internet Mail Extension (MIME) standard, which is widely used for HTTP and XML, and was originally developed for encoding email attachments for transmission.

3.3.1.1 Why Do We Use Base64 ?

Base64 is very important for binary data representation, such that it allows binary data to be represented in a way that looks and acts as plain text, which makes it more reliable to be stored in databases, sent in emails, or used in text-based format such as XML. Base64 is basically used for representing data in an ASCII string format.

3.3.1.2 Base64 Encoding

Base64 encoding is the process of converting binary data into a limited character set of 64 characters. The characters are A-Z, a-z, 0-9, +, and / . This character set is considered the most common character set, and is referred to as MIME's Base64. It uses A-Z, a-z, 0-9, +, and / for the first 62 values, and +, and / for the last two values. The Base64 encoded data ends up being longer than the original data, so that, for every 3 bytes of binary data, there are at least 4 bytes of Base64 encoded data. This is due to the fact that we are squeezing the data into a smaller set of characters.

3.3.1.3 Base64 Decoding

Base64 decoding is the opposite of Base64 encoding. In other words, it is carried out by reversing the steps described in the Encoding. So, the Each character in the string is changed to its Base64 decimal value. The decimal values obtained are converted into their binary equivalents. The first two bits of the binary numbers are truncated from each of the binary numbers obtained, and the sets of 6 bits are combined, forming one large string of binary digits. The large string of binary digits obtained in the previous step is split into groups of 8 bits. The 8-bit binary numbers are converted into their decimal equivalents. Finally, the decimal values obtained are converted into their ASCII equivalent.

3.3.1.4 Usage

Base64 is most commonly used to encode binary data (for example images, or sound files) for embedding into HTML, CSS, EML, and other text documents. In addition, Base64 is used to encode data that may be unsupported or damaged during transfer, storage, or output. Some of the applications of the algorithm:

- Attach files when sending emails
- Embed images in HTML or CSS via data URI
- Preserve raw bytes of cryptographic functions
- Output binary data as XML or JSON in API responses
- Save binary files to database when BLOB is unavailable

3.3.2 RSA

The RSA algorithm is the basis of a cryptosystem a suite of cryptographic algorithms that are used for specific security services which enables public key encryption and is widely used to secure sensitive data, particularly when it is being sent over an insecure network such as the internet. RSA was first publicly described in 1977 by Ron Rivest, Adi Shamir and Leonard Adleman of the Massachusetts Institute of Technology, though the 1973 creation of a public key algorithm by British mathematician Clifford Cocks was kept classified by the U.K.'s GCHQ until 1997. In RSA cryptography, both the public and the private keys can encrypt a message; the opposite key from the one used to encrypt a message is used to decrypt it. This attribute is one reason why RSA has become the most widely used asymmetric algorithm, it provides a method to assure the confidentiality, integrity, authenticity, and non-repudiation of electronic communications and data storage.

RSA derives its security from the difficulty of factoring large integers that are the product of two large prime numbers. Multiplying these two numbers is easy, but determining the original prime numbers from the total or factoring is considered infeasible due to the time it would take using even today's supercomputers.

3.3.2.1 Why RSA Algorithm is used ?

The public and private key generation algorithm is the most complex part of RSA cryptography. Two large prime numbers, p and q, are selected. N is calculated by multiplying p and q. This number is used by both the public and private keys and provides the link between them. Its length, usually expressed in bits, is called the key length.

The public key consists of the modulus n and a public exponent e. The e doesn't have to be a secretly selected prime number, as the public key is shared with everyone.

The private key consists of the modulus n and the private exponent d, which is calculated using the Extended Euclidean algorithm to find the multiplicative inverse with respect to the totient of n.

3.3.2.2 RSA Security

RSA security relies on the computational difficulty of factoring large integers. As computing power increases and more efficient factoring algorithms are discovered, the ability to factor larger and larger numbers also increases. Encryption strength is directly to key size, and doubling key length can deliver an exponential increase in strength, although it does impair performance. RSA keys are typically 1024-bits or 2048-bits long, but experts believe that 1024-bit keys are no longer fully secure against all attacks. This is why the government and some industries are moving to a minimum key length of 2048-bits.

Barring an unforeseen breakthrough in quantum computing, it will be many years before longer keys are required, but elliptic curve cryptography (ECC) is gaining with many security experts as an alternative to RSA to implement public key cryptography. It can create faster, smaller and more efficient cryptographic keys.

Modern hardware and software are ECC-ready, and its popularity is likely to grow, as it can deliver equivalent security with lower computing power and battery resource usage, making it more suitable for mobile apps than RSA. Finally, a team of researchers, which included Adi Shamir, a co-inventor of RSA, has successfully created a 4096-bit RSA key using acoustic cryptanalysis; however, any encryption algorithm is vulnerable to attack.

3.3.2.3 Description of Algorithm

- Plaintext is taken from a specified file and then encrypted using RSA Algorithm.
- Encryption and decryption are of following form for same plaintext M and ciphertext C.
- C=(M^e) mod n
- M=(C^d) mod n
- M=((M^e)^d) mod n
- M=(M^ed)modn
- Both sender and receiver must know the value of n.
- The sender knows the value of e, and the receiver knows the value of d.
- Thus, this is a public key encryption algorithm with a public key of PU = {c, n} and private key of PR= {d, n}.

3.3.2.3 RSA algorithm

a) Key Generation :

- Select p and q such that both are the prime numbers, $p \neq q$.
- Calculate $n=p \times q$
- Calculate $q(n) = (p-1)(q-1)$
- Select an integer e such that : $g(d((n), e)) = 1$ & $1 < e < (n)$
- Calculate d; $de = 1 \bmod (q(n))$
- Public Key, PU= {e, n}
- Private Key, PR ={d,n}

b) Encryption :

- Plaintext : M
- Ciphertext: $C = (M^e) \bmod n$

c) Decryption:

- Ciphertext: C
- Plaintext : $M= (C^d) \bmod n$
- Note 1 : (n) -> Euler's totient function
- Note 2: Relationship between C and d is expressed as:

$$ed \pmod{(n)} = 1$$

$$ed = 1 \bmod (n)$$

$$d = e^{-1} \bmod (n)$$

3.3.3 STEGANOGRAPHY

Data hiding is of important in many applications. For hobbyists, secretive data transmission, privacy of users etc. the basic methods are: Steganography and Cryptography. Steganography is a simple security method. Generally, there are three different methods used for hiding information: steganography, cryptography, watermarking. In cryptography, the information to be hidden is encoded using certain techniques; this information is generally understood to be coded as the data appears nonsensical. Steganography is hiding information; this generally cannot be identified because the coded information doesn't appear to be abnormal i.e. its presence is undetectable by sight. Detection of steganography is called Stego analysis. Steganography is of 4 different types:

- Text steganography
- Image steganography
- Audio steganography
- Video steganography

In all of these methods, the basic principle of steganography is that a secret message is to be embedded in another cover object. So, it cannot be detected easily to be containing hidden information unless proper decryption is used.
Every steganography consists of three components:

- Cover object
- Message object
- Resulting Steganographic object

Image Steganography

Image Steganography deals with the hiding of data within the image, data can be any file, such as an image, audio, text or another file. We have to wrap up the data using an image. We have chosen to bind data in an image. This kind of embedding an audio in an image helps to authenticate the sender, verify whether valid user is receiving the data or not and to find whether a third-party attacker is present in the channel of communication or not. Since, image is used as a cover file, we have to make sure that the image must be accountable for the data that is being embedded. Hence a 24-bit image format proved to be the best solution for hiding the data, since it holds a large memory space and convenient to hide a considerable amount of data. Furthermore, the threshold sure of the image must be calculated for the given image size which will be explained in the layer parts.

LSB Positioning Method

This method is the simplest method of hiding data within in the given image. We utilize the LSB bits of the pixels within the given image. When converting the image to digital format, we usually choose between three different ways of representing colors :

- 24-bit color : every pixel can have one in 2^24 colors, and these are represented as different quantities of three basic colors : red(R), green(G), blue(b) given by 8 bits (256) each.
- 8-bit color : every pixel can have one in 256 (2^28) colors, chosen from a palette, or a table of colors.
- 8-bit gray-scale : every pixel can have one in 256 (2^28) shades of gray.

LSB insertion modifies the LSBs of each color in 24-bit images, or the LSBs of the 8-bit value for 8-bit images. The most basic of LSBs insertion for 24-bit pictures inserts 3 bits/pixel.

For image steganography we are using Spatial methods. In spatial method, the most common method used is LSB substitution method. Least significant bit (LSB) method is a common, simple approach to embedding information in a cover file. In steganography, LSB substitution method is used. I.e. since every image has three components (RGB). This pixel information is stored in encoded format in one byte. The first bits containing this information for every pixel can be modified to store the hidden text. For this, the preliminary condition is that the text to be stored has to be smaller or of equal size to the image used to hide the text. LSB based method is a spatial domain method. But this is vulnerable to cropping and noise. In this method, the MSB (most significant bits) of the message image to be hidden are stored in the LSB (least significant bits) of the image used as the cover image.

The Human visual system (HVS) cannot detect changes in the colour or intensity of a pixel when the LSB bit is modified. This is psycho-visual redundancy since this can be used as an advantage to store information in these bits and yet notice no major difference in the image.

3.3.3.1 Algorithm

Inputs : Image, Message, Key.

1) Initially Sender consider a Cover Image.

2) Hide the Encrypted message in the image.

3) Hiding can be done using a secret key for confidentiality.

4) After Hiding the Image is considered as a Stego-Image. Which consists of image and data which is encrypted.

5) The Receiver will receive the Stego-Image.

6) Using the Secret Key, the receiver can view the data hidden in the image.

7) Thus, the receiver can receive the message safely.

3.4 ALGORITHM ILLUSTARTION

Encryption :

Inputs : Message, 2 Prime Numbers, Image, Secret Key

Step 1 : Consider an Input , it can be :

a) Text

b) age

c) Audio

d) Video

Step 2 : Convert the input to Base-64 using Base-64 conversion Algorithm.

Step 3 : After converting into Base-64 we will be getting a String.

Step 4 : Store the entire string in a Text File and save the file.
Step 5 : From that file consider each character and apply RSA.
Step 6 : By Using RSA we will be getting Cipher Text (cm).
Step 7 : Let the Cipher Text (cm) be encrypted message.

Step 8 : Consider an image, and hide the encrypted message(cm) in the given image with the secret key Using Steganography Algorithm.

Step 9: Now send the Stego-Image to the Receiver.

DECRYPTION :

Inputs : Cipher Text, 2 Prime Numbers, Image, Secret Key
Step 1 : Consider the input be Stego-Image.

Step 2 : Using the Secret Key , Obtain the hidden message from the Stego-Image.
Step 4 : And the obtained message is a Cipher Text. We must decrypt the message.
Step 5 : The Decryption of the message can be done using RSA Algorithm.
Step 6: By Using RSA we will be getting Plain Text.

Step 7 : And thus, the receiver will decrypt the message and it is in the form of Base-64.

Step 8 : Finally, by using Base-64 algorithm the Base-64 text is converted into the original input, which can be Text, Image, Audio, Video.

4.DESIGN

Project design is a major step towards a successful project. A project design is a strategic organization of ideas, materials and processes for the purpose of achieving a goal. Project managers rely on a good design to avoid pitfalls and provide parameters to maintain crucial aspects of the project. Project design is an early phase of the project where a project's key features, structure, criteria for success, and major deliverables are all planned out. The point is to develop one or more designs which can be used to achieve the desired project goals. Stakeholders can then choose the best design to use for the actual execution of the project. The project design phase might generate a variety of different outputs, including sketches, flowcharts, HTML screen designs, and more.

So, the design can be implemented using Unified Modeling Language. diagrams such as class diagram, use case diagram, sequence diagram, activity diagrams. UMLoffers a way to visualize a system's architectural blueprints in a diagram, including elements such as :

- Any activates
- Individual components of the system
- How the system will run
- How entities interact with others
- External user interface

UML is a common language for business analysts, software architects and developers used to describe, specify, design, and document existing or new business processes, structure and behaviour of artifacts of software systems. The key to making a UML diagram is connecting shapes that represent an object or class with other shapes to illustrate relationships and the flow of information and data.

4.1 Class Diagram

A class diagram in the Unified Modelling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for

visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modelling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

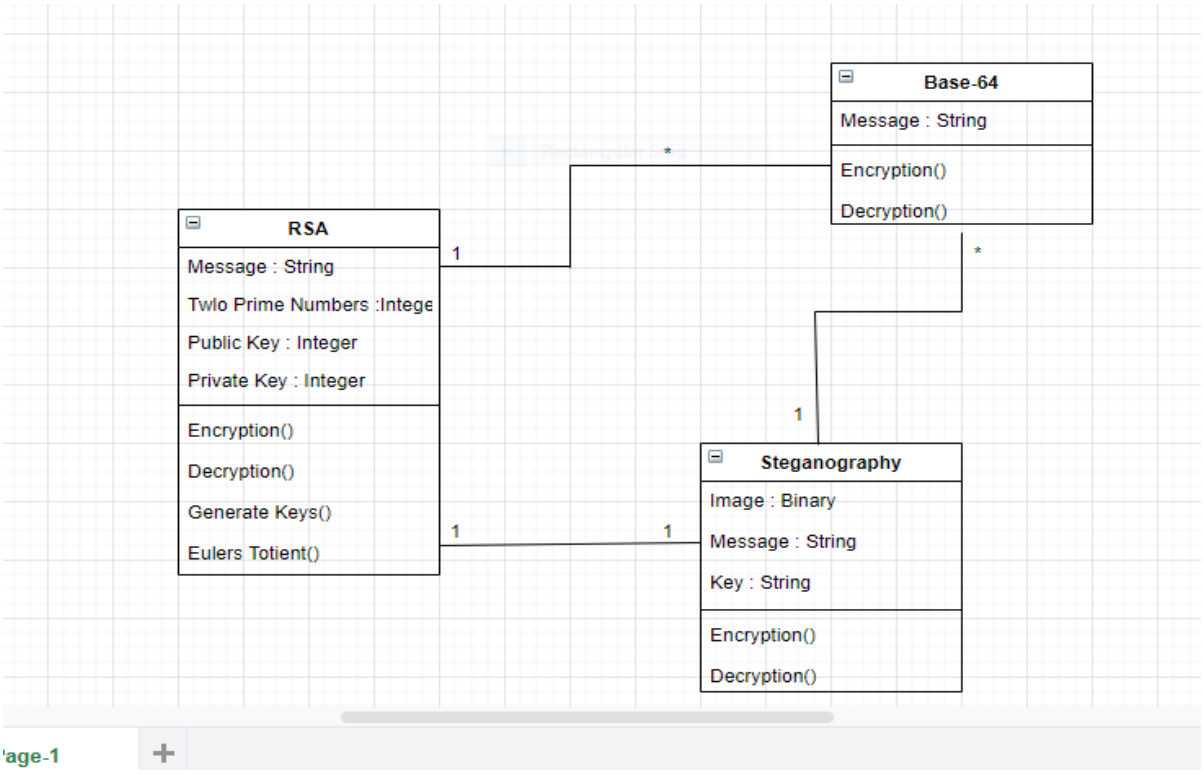


Fig 4.1 Class Diagram

4.2 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.

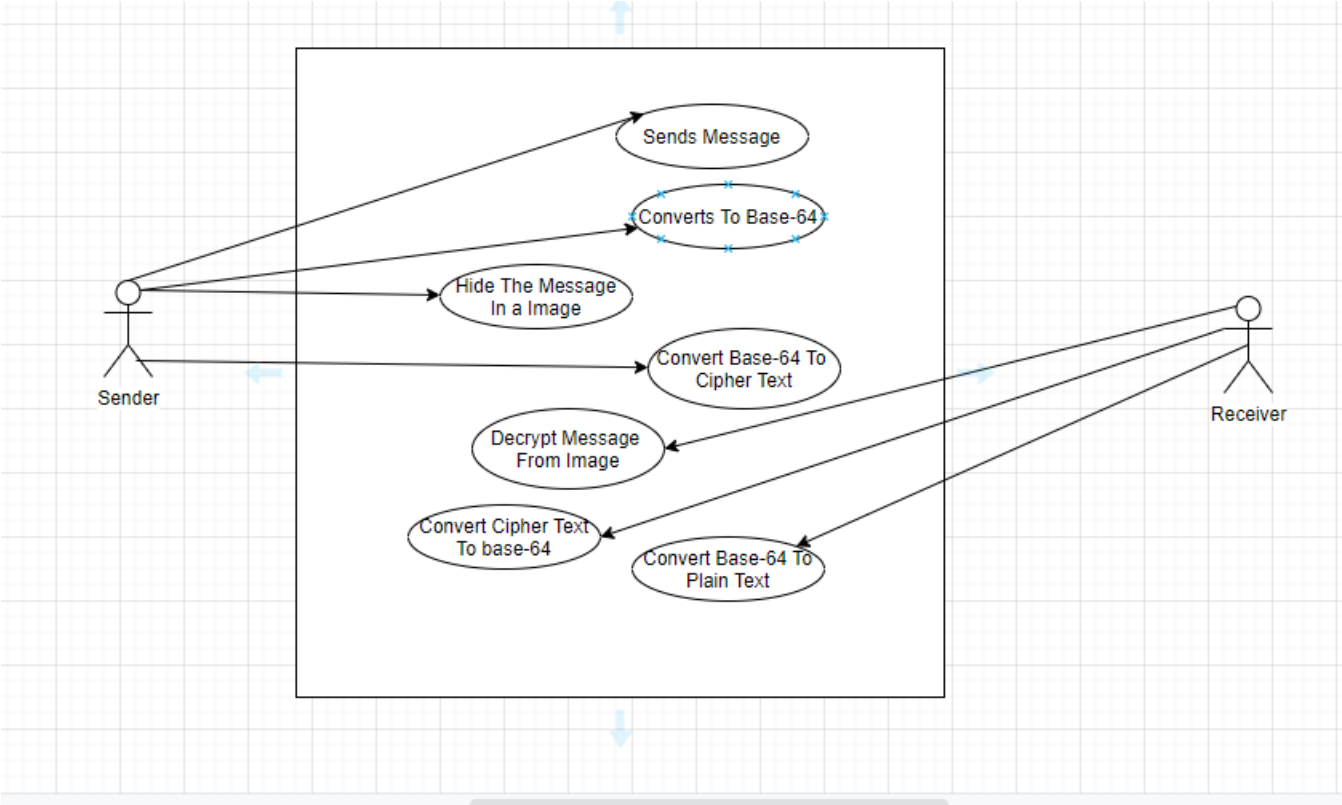


Fig 4.2 Use Case Diagram

4.3 Sequence Diagram

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram shows, as parallel vertical lines, different processes or objects that live simultaneously and as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

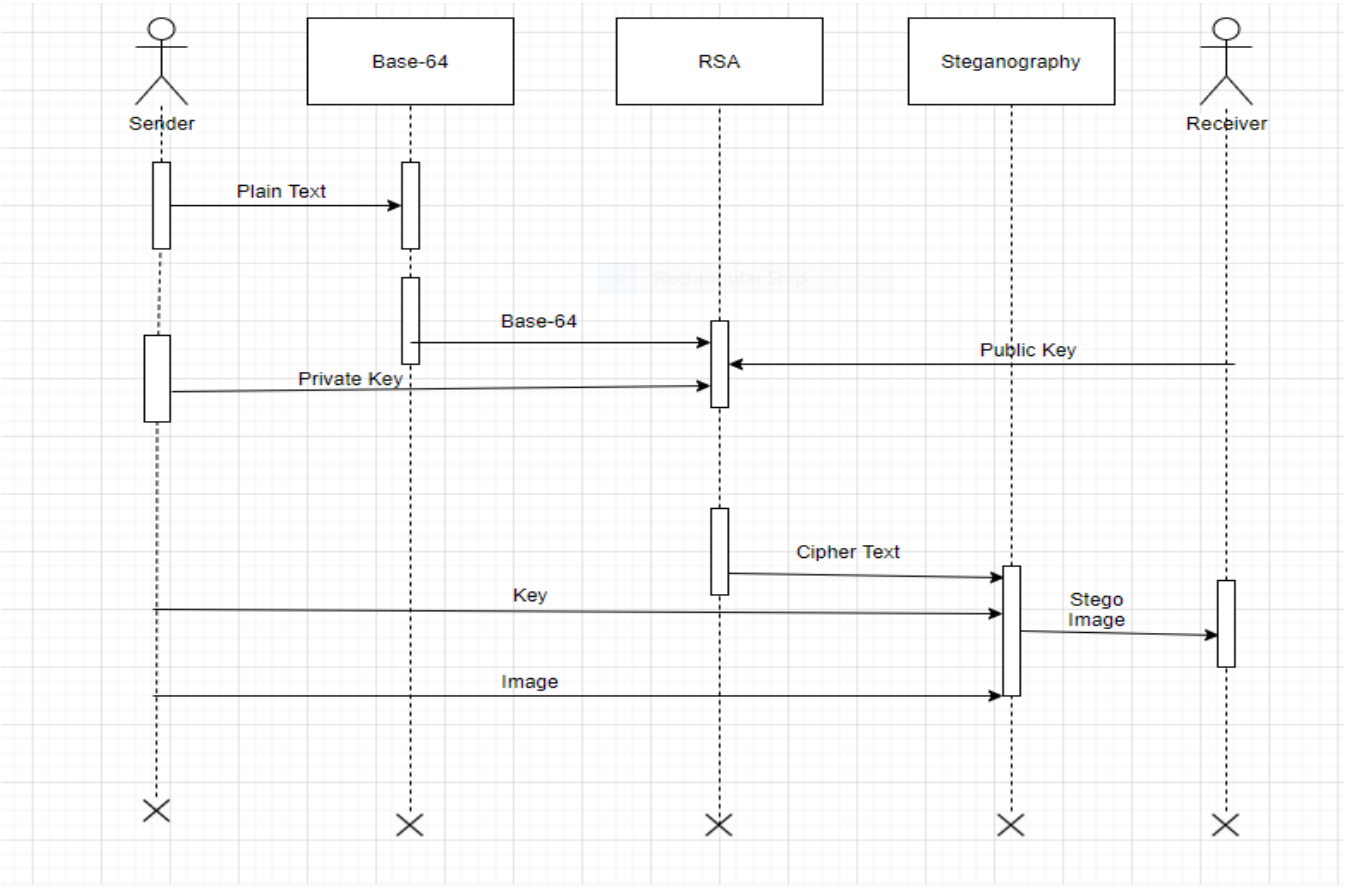


Fig 4.3 Sequential Diagram

4.4 Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows) as well as the data flows intersecting with the related activities. Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores.

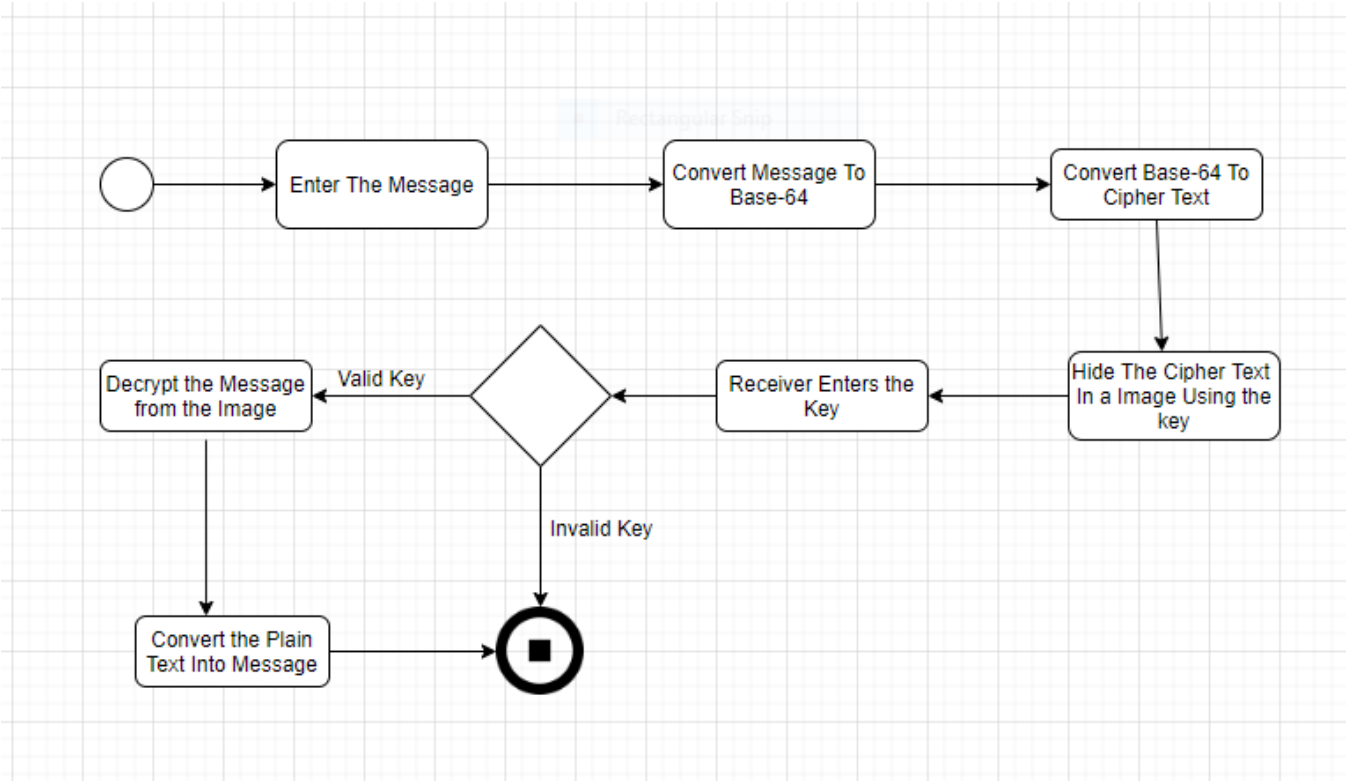


Fig 4.4 Activity Diagram

5.Requirements, Analysis & Result

5.1 SYSTEM CONFIGURATION

5.1.1 Software Requirements:

- Operating System : windows10.
- Programming Language: Java.
- Idle: Eclipse IDE.

5.1.2 Hardware Requirements:

- Processor : 1GHz or faster.
- RAM : 512MB or Higher.
- Hard Disk :10 GB or Higher.

5.2 Source Code

5.2.1 Main.java

```
package in.Raghavsingh.EncryptionSystem;

import java.awt.*;
import java.awt.image.BufferedImage;
import java.io.*;
import javax.imageio.ImageIO;
import javax.swing.JFrame;
import javax.swing.*;

public class Main {
ImageIcon image;
Dimension d = null;
  public Main() {
        JFrame frame = new JFrame("Cipher Vault");
        d=Toolkit.getDefaultToolkit().getScreenSize();
        frame.setSize(d.width/2, d.height/2);
        frame.setLocation(d.width/4,d.height/4);
        frame.setResizable(false);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
JPanel panel1=new JPanel();
JPanel panel2=new JPanel();
JPanel panel3=new JPanel();
panel1.setPreferredSize(new Dimension(100,100));
panel2.setPreferredSize(new Dimension(100,100));
panel3.setPreferredSize(new Dimension(100,100));
panel1.setOpaque(true);
panel2.setOpaque(true);
panel3.setOpaque(true);
panel1.setBackground(Color.yellow);
panel1.setForeground(Color.red);
panel2.setBackground(Color.green);
panel2.setForeground(Color.red);
panel3.setBackground(Color.yellow);
panel3.setForeground(Color.red);
image=new ImageIcon("wait.GIF");
```

```
JLabel im= new JLabel(image);
JLabel msg=new JLabel("LOADING.....");
panel1.add(msg);
panel2.add(im);
frame.add(panel1,BorderLayout.NORTH);
frame.add(panel2,BorderLayout.CENTER);
frame.add(panel3,BorderLayout.SOUTH);
//frame.setVisible(true);
//try
//{
        //Thread.sleep(4000);
        new Cryptography();
        //frame.setVisible(false);
        //frame.dispose();
//}
//catch(InterruptedException ec){}
}
public static void main(String args[]) throws Exception {
        new Main();
  }
}
```

5.2.2 Cryptography.java

```
package in.Raghavsingh.EncryptionSystem;


import java.nio.*;
import java.nio.channels.*;
import java.io.*;
import java.awt.*;
import java.awt.event.*;
import java.awt.Color.*;
import javax.swing.*;
import javax.swing.filechooser.FileFilter;
import java.util.*;
import java.awt.image.BufferedImage;
import javax.imageio.ImageIO;
class JavaFileFilter extends FileFilter{
                public boolean accept(File file)
```

```
                    {
                      if(file.getName().endsWith(".txt")) return true;
                      if(file.getName().endsWith(".java")) return true;
                      if(file.isDirectory()) return true;
                      return false;
                    }
                    public String getDescription()
                    {
                      return "Java and Text file for Encryption and Dencryption";
                    }
}

class SaveJavaFileFilter extends FileFilter
 {
   public boolean accept(File f)
   {
       if (f.isDirectory())
                    return true;

        String s = f.getName();

       return s.endsWith(".java");
   }

   public String getDescription()
   {
      return "*.java";
   }

}

class SaveTextFileFilter extends FileFilter
 {
   public boolean accept(File f)
   {
       if (f.isDirectory())
                    return true;
        String s = f.getName();
```

```
        return s.endsWith(".txt");
    }

    public String getDescription()
    {
        return "*.txt";
    }

}
public class Cryptography extends JFrame implements ActionListener
{
                        public JButton browse,enc,denc,cancel;
    private JLabel label;
    private JTextField filename;
    private JFileChooser jfc;
    private File file;
    Dimension d = null;
FileInputStream fin;
FileOutputStream fout;
FileChannel fichan,fochan;
long fsize;
ByteBuffer mbuf,ombuf;
long key=0;
String ext="";
JScrollPane displayScrollPane;
ImageIcon image;
int wdth,hight;

    public Cryptography() {
                        super("Cipher Vault");
                        d=Toolkit.getDefaultToolkit().getScreenSize();
                        setBackground(Color.yellow);
                           setForeground(Color.red);
                        wdth=d.width/2;
                        hight=d.height/2;
        setSize(wdth, hight);
        setLocation(d.width/4, d.height/4);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setResizable(false);
```

```
        label = new JLabel("CHOOSE THE TEXT OR JAVA FILE FOR
ENCRYPTION OR DECRYPTION");
        filename = new JTextField(50);
        filename.setEditable(false);
        browse=new JButton("Browse");
        browse.setForeground(Color.BLUE);
        enc=new JButton("Encrypt");
        enc.setForeground(Color.BLUE);
        enc.setEnabled(false);
        denc=new JButton("Dencrypt");
        denc.setForeground(Color.BLUE);
        denc.setEnabled(false);
        cancel=new JButton("Cancel");
        cancel.setForeground(Color.RED);
        jfc=new JFileChooser();
        jfc.setFileFilter(new JavaFileFilter());
        JPanel buttonPanel1 = new JPanel();
        JPanel buttonPanel2 = new JPanel();
        JPanel loadPanel = new JPanel();
        buttonPanel1.setPreferredSize(new Dimension(100,100));
        buttonPanel2.setPreferredSize(new Dimension(100,100));
        loadPanel.setPreferredSize(new Dimension(100,100));
        buttonPanel1.setOpaque(true);
                        buttonPanel2.setOpaque(true);
                        loadPanel.setOpaque(true);
                        loadPanel.setBackground(Color.yellow);
                        buttonPanel1.setBackground(Color.green);
                        buttonPanel1.setForeground(Color.red);
                        buttonPanel2.setBackground(Color.green);
                        buttonPanel2.setForeground(Color.red);

                    buttonPanel1.setBorder(BorderFactory.createLineBorder(Color.BL
UE));

                    buttonPanel2.setBorder(BorderFactory.createLineBorder(Color.BL
UE));
                        image=new ImageIcon("logo.GIF");
JLabel im= new JLabel(image);
loadPanel.add(im);
```

```
                    buttonPanel1.add(label);
        buttonPanel1.add(filename);
        buttonPanel1.add(browse);
        buttonPanel2.add(enc);
        buttonPanel1.add(denc);
        buttonPanel2.add(cancel);
        label.setBounds(105, 10, 400, 25);
                    browse.addActionListener(this);
                    enc.addActionListener(this);
                    denc.addActionListener(this);
                    cancel.addActionListener(this);
                    add(buttonPanel1, BorderLayout.NORTH);
        add(loadPanel, BorderLayout.CENTER);
        add(buttonPanel2, BorderLayout.SOUTH);
        setVisible(true);
     }
    public void actionPerformed(ActionEvent e) {
                    if(e.getSource() == browse)         {
                      int result = jfc.showOpenDialog(null);
                            if(result==JFileChooser.APPROVE_OPTION)
                            {
                                    label.setText("Selected file is :
"+jfc.getSelectedFile().getName());
                                    filename.setText(jfc.getSelectedFile().getPath());
                                    file=jfc.getSelectedFile();
                                    denc.setEnabled(true);
                                    enc.setEnabled(true);
                            }
                            else
                            {
                                    filename.setText("No file selected");
                                    denc.setEnabled(false);
                                    enc.setEnabled(false);
                            }
                    }
                    if(e.getSource()==enc)
                    {
                      try
```

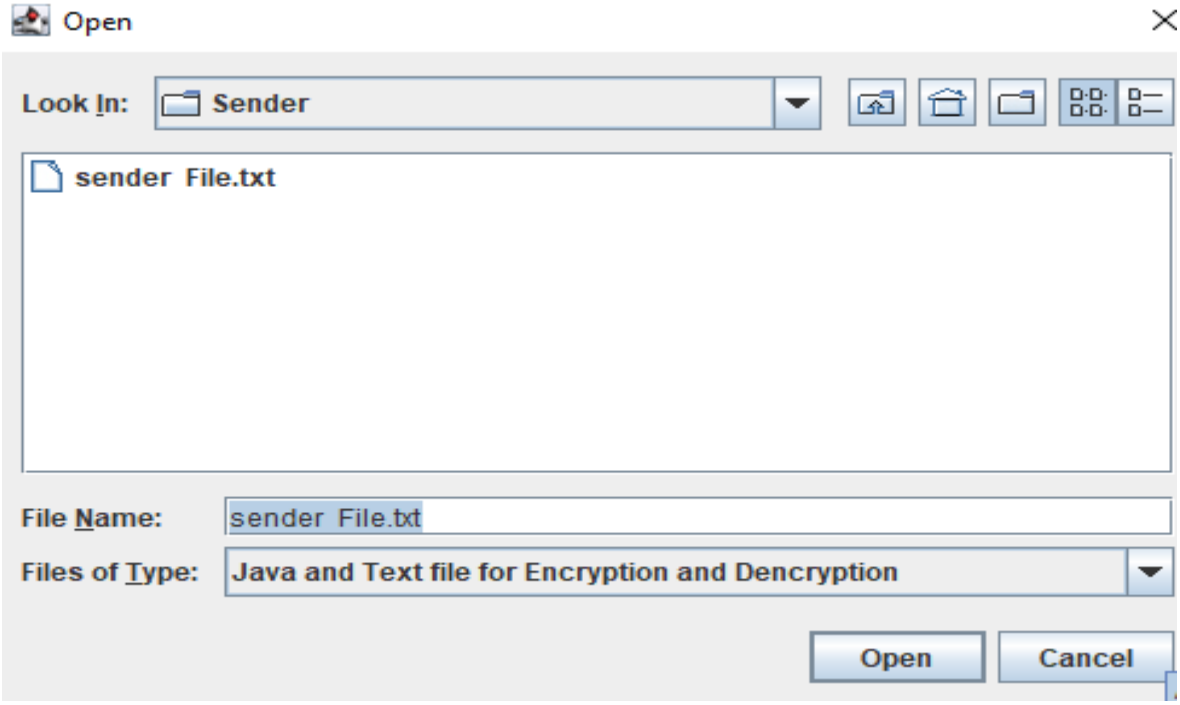
```
                    {
                    key=enterKey();
                    if(key>=1)
                    {
                    JOptionPane.showMessageDialog(null,"Save the Encrypted
file","ENCRYPTION COMPLETED",JOptionPane.INFORMATION_MESSAGE);
                convert(-key);
                    }
                    else
                                JOptionPane.showMessageDialog(null,"Please enter
a nonzero key","WARNING",JOptionPane.WARNING_MESSAGE);
                    }catch(Exception ioee){
                    }
                }
                if(e.getSource()==denc)
                {try
                {
                        key=enterKey();
                        if(key>=1)
                        {
                        JOptionPane.showMessageDialog(null,"Save the Decrypted
file","DECRYPTION COMPLETED",JOptionPane.INFORMATION_MESSAGE);
                        convert(key);
                        }
                        else
                                JOptionPane.showMessageDialog(null,"Please enter
a nonzero key","WARNING",JOptionPane.WARNING_MESSAGE);
                }catch(Exception ex){}
                }
   if(e.getSource() == cancel)    {
                    System.exit(1);
                }
   }
private long enterKey() throws IOException
{
                long k=0;
                String key=JOptionPane.showInputDialog(null,"Enter the
Key","SECURE KEYS",JOptionPane.QUESTION_MESSAGE);
                long intKey=(long)Integer.parseInt(key);
```

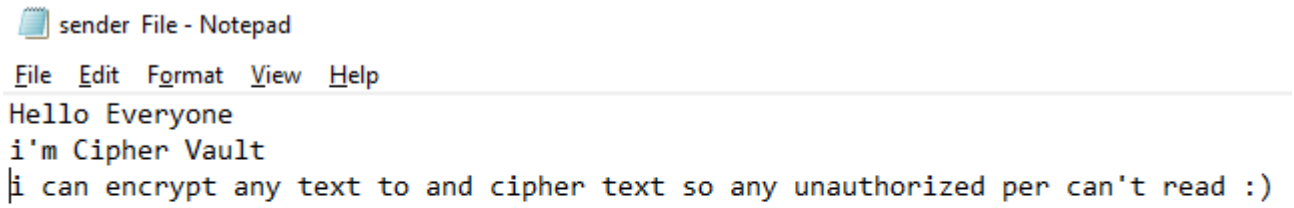
```
                    long temp=intKey;
                    checking:
                    {
                    do
                    {
                    k=k+(intKey%10);
                    intKey=intKey/10;
                    }while(intKey>=10);
                    k=k+intKey;
                    if(k>32)
                    {
                      intKey=temp/10;
                      break checking;
                    }
                    }
                    return k;
}
private void convert(long secureKey)
{
                    long Key=secureKey;
try
{
JFileChooser jFileChooser = new JFileChooser();
jFileChooser.addChoosableFileFilter(new SaveJavaFileFilter());
jFileChooser.addChoosableFileFilter(new SaveTextFileFilter());
jFileChooser.setSelectedFile(new File("fileToSave.txt"));
int responce = jFileChooser.showSaveDialog(null);
if(responce==JFileChooser.APPROVE_OPTION)
{
   String extension=jFileChooser.getFileFilter().getDescription();
   if(extension.equals("*.java"))
     {
       ext=".java";
     }
   if(extension.equals("*.txt"))
     {
       ext=".txt";
     }
}
```

```
fin=new FileInputStream(file);
fout=new FileOutputStream(jFileChooser.getSelectedFile()+ext);
fichan=fin.getChannel();
fochan=fout.getChannel();
fsize=fichan.size();
mbuf=ByteBuffer.allocate((int)fsize);
ombuf=ByteBuffer.allocate((int)fsize);
fichan.read(mbuf);
mbuf.rewind();
for(int i=0;i<fsize;i++)
{
long data=((long) mbuf.get());
ombuf.put((byte)(data+Key));
}
ombuf.rewind();
fochan.write(ombuf);
fichan.close();
fin.close();
fochan.close();
fout.close();
}
catch(IOException e)
{
System.out.println(e);
System.exit(1);
}
catch(BufferUnderflowException uf)
{
                System.out.println(uf);
}
 }
 public static void main(String args[]){
                SwingUtilities.invokeLater(new Runnable()
                  {
                        public void run()
                        {
                              new Cryptography();
                        }
```

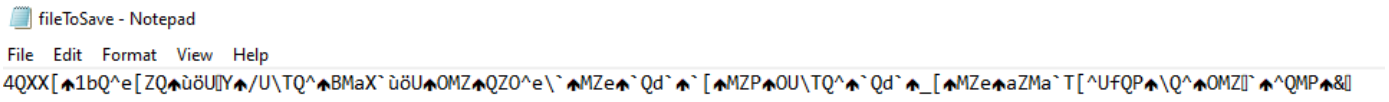
```
            });
        }

    }
```

5.3 Output Screen:





TEXT BEFORE ENCRYPTION



TEXT AFTER ENCRYPTION

6.FUTURE SCOPE AND CONCLUSION

In this project, we deal with the concepts of security of digital data communication across the network. This project is designed by using the steganography and cryptography features factors for better performance. We performed a new encryption method and combined it with RSA Encryption algorithm. We performed our method on text by implementing a program written in Java language. The method proposed has proved successful in hiding various types of text, images, audio and videos . We concluded that in our method the text files and RSA Cryptography are better . Because of their high capacity. Results achieved indicate that our proposed method is encouraging in terms of security and robustness.

7.REFERENCES:

[1]H.Abdulzahra, R. AHMAD, and N. M. NOOR, "Security enhancement; Combining cryptography and steganography for data hiding in images," ACACOS, Applied Computational Science,pp.978-960,2014.

[2]P. R. Ekatpure and R. N.Benkar, "A comparative study of steganography & cryptography,"2013.

[3]M. H. Rajyaguru, "Cryptography-combination of cryptography and steganography with rapidly changing keys ,"Interntional Journal of Emerging Technology and Advanced Engineering,ISSN ,pp.2250-2459,2012.

[4]D. Seth. L. Ramanathan, and A.Pandey, "Security enhancement; Combining cryptography and steganography," International Journal of Computer Applications(0975-8887)Volume,2010.

[5] J. V. Karthik and B. V. Reddy, "Authentication of secret information in image steganography," International Journal of Computer Science and Network Security(IJCSNS), vol. 14, no. 6. P. 58. 2014.