CHAPTER ONE

**INTRODUCTION**

* 1. Overview

Text file encryption plays a vital role in securing the confidentiality of the information stored in the computer hard disk. This chapter takes us through the background information, problem statement, main objective, specific objectives, project justification and the scope of the project.

1.2 Background of the study

Nowadays Organizations, Corporate Companies, Government agencies, Schools, Hospitals, and enterprises - all of these organizations have the confidential data on their desktop or in the drive. To provide security to all these confidential data on the desktop by converting the plain text to cipher text when the file is encrypted, which is set to be read only and delete option is disabled for all the encrypted files. Many encryption algorithms are widely available and used in information security. They can be categorized into Symmetric (private) and asymmetric (public) keys encryption. In Symmetric keys encryption or secret key encryption, only one key is used to encrypt and decrypt data. In Asymmetric keys, two keys are used; private and public keys. Public key is used for encryption and private key is used for decryption (e.g. RSA). Public key encryption is based on mathematical functions, computationally intensive. This project proposes the use of the AES symmetric key encryption scheme.

The idea of using this algorithm is, AES is unbreakable when compared to DES and IDEA algorithm. Unlike DES can be attacked by a high-order differential attack requiring 264-252 chosen plain texts breaks 6 rounds with a complexity of 2126.8 encryptions. DES is insecure because, a brute force attack is possible. Other than this, there are few other attacks which proves DES algorithm insecure are one round attack, full 16-round attack and Meet-in-the-middle attack.

The Advanced Encryption Standard is the strong symmetric key cryptographic algorithm which has been introduced due to the limitations of other algorithms like DES (Data Encryption Standard). The security of AES will be high due to the presence of large number of rounds or blocks. The output of one block acts as the input of the next block in both encryption and decryption.

1.3 Problem Statement

Data in the personal computers are stored in form of the files. These files can either be text files, image files or even PDF files. Sometimes the intruder get access to our personal and confidential , get a copy of it, modify it or even delete it without the authorized permission of the user.

* 1. Main objective

The main aim of this project is to provide the confidentiality of the information stored in the computer hard drive through the use of the AES encryption algorithm.

* 1. Specific Objectives
* To analyze the previously developed encryption systems for securing the text files and their loopholes.
* To design the modern text file encryption system using AES algorithm scheme.
* To implement the design text file encryption system.
* To test its working capability using various software testing methods i.e. unit testing, integration testing, acceptance testing etc.
  1. Project Justification

The security of the personal computers has the great hole when it comes to confidentiality of the personal sensitive files. This project will fill this hole and further boosts the security especially the confidentiality of information in the PCs.

* 1. Scope of the project

This project covers the encryption of the static text files within the computer drive. It uses the symmetric key encryption called AES algorithm. This project won’t cover the image encryption.

CHAPTER TWO

LITERATURE REVIEW

2.1 Overview of the Literature review

Nowadays AES algorithm plays a vital role in securing the confidentiality and secrecy of both the static and dynamic information. Many computer scientists have greatly applied the AES in different ways ranging from organization, institutions, schools, banks, military etc. In order to understand this, this chapter gives more prospective and detailed ideas about the performance of the encryption algorithms, this subsection describes and examines previous work done in field of data encryption. This subsection also discusses the results obtained for some of the algorithms.

2.2 Definitions of the terms

The following are the terms used in this chapter;

**Encryption**- This is the process of converting the plain message into cipher text, which cannot be easily understood by anyone except authorized parties.

Decryption-This process of converting the encrypted message (cipher) into plain text, which can easily be understood.

WLAN- This is the wireless local area network.

AES- This is called Advanced encryption standard. It one of the modern symmetric key encryption commonly used for providing data confidentiality.

Blowfish- this is the symmetric key encryption that uses the feistel network to perform the encryption and decryption.

RC4- this is the Rivest Cipher 4, which is the stream cipher.

2.3 Related studies

The following are previous work done by different scientists;

**D. S. Abdul. Elminaam et.al., (2009)** presents a performance evaluation of selected symmetric encryption algorithms on power consumption for wireless devices in their paper named “Evaluating the Effects of Cryptography Algorithms on power consumption for wireless devices.” Following points are concluded by him from his experimental result. If packet size is changing with or without transmission of data using various WLANs protocols and different architectures. It was concluded form the result that Blowfish and AES has better performance than other common encryption algorithms used, followed by RC6. Worm holes are present in the security mechanism of DES and 3DES; Blowfish and AES do not have such worm holes any so far.

**Seyed Hossein Kamali, Reza Shakerian, Maysam Hedayati, Mohsen Rahmani**, “A New Modified Version of Advanced Encryption Standard (AES) Based Algorithm for Image Encryption” (2010). The authors proposed an enhanced model of Advanced Encryption Standard to possess good level of security and better range of image encryption. The modification process can be carried out by adjusting the Shift Row Transformation. As the result shown, that the comparison has been made in between the original AES encryption algorithm and the modified algorithm which produces very good encryption results focusing towards the security against statistical attacks.

**P.Prasithsangaree and his collegeue P. Krishnamurthy** 2003 analyze the Energy Consumption of RC4 and AES Algorithms in Wireless LANs. RC4 and AES encryption algorithms performance evaluation is made by their research. The matrices for such evaluation are as follows: CPU work load, encryption throughput, key size variation and energy cost. Experimental results conclude that for encrypting large packets the RC4 is energy efficient and fast. However, for a smaller packet size encryption AES was more efficient than RC4. Therefore it appears that by using a combination of RC4 and AES we can save energy to provide encryption for any packet size.

**Agrawal et al.** Present detailed study of the popular symmetric key encryption algorithms such as DES, TRIPLE DES, AES, and Blowfish. Symmetric Key algorithms run faster than Asymmetric Key algorithms such as RSA etc and the memory requirement of Symmetric algorithms is lesser than asymmetric encryption algorithms. Security of Symmetric key encryption is superior to Asymmetric key encryption. It was concluded that the supremacy of Blowfish algorithm over DES, AES and Triple DES on the basis of key size and security. The F function of Blowfish algorithm provides a high level of security to encrypt the 64 bit plaintext data.

**Seth et al.** made a comparative analysis of three algorithms, DES, AES and RSA considering certain parameters such as computation time, memory usages and output byte. It was concluded that RSA consumes longest encryption time and memory usage is also very high but output byte is least in case of RSA algorithm. Based on the text files used and the experimental result it was concluded that DES consume least encryption time and AES has least memory usage while encryption time difference is very minor in case of AES algorithm and DES algorithm.

**Marwaha et al.** discussed three algorithms DES, 3DES and RSA. DES and 3DES are symmetric key cryptographic algorithms and RSA is an asymmetric key cryptographic algorithm. Algorithms have been analyzed on their ability to secure data, time taken to encrypt data and throughput the algorithm requires. Performance of different algorithms was different according to the inputs. It was concluded that confidentiality and scalability provided by 3DES over DES and RSA is much higher and makes it suitable even through DES consumes less power memory and time to encrypt and decrypt the data but on security from DES can be easily broken by brute force technique as compared to 3DES and RSA, making it the last secure algorithm.

2.4 Objectives

* To understand how various algorithms were implemented to encrypted different types of file formats.
* To understand the different types of encryption algorithms used

CHAPTER THREE

SYSTEM ANALYSIS

3.1 Introduction

At this step the developers decide a roadmap of their plan and try to bring up the  
best software model suitable for the project. System analysis includes understanding of software product limitations, learning system related problems or changes to be done in existing systems beforehand, identifying and addressing the impact of project on organization and personnel etc. The project team analyzes the scope of the project and plans the schedule and resources accordingly.

Weakness of the previous systems

* The text file encryption system developed previously uses the outdated and old encryption scheme i.e. DES which had a small key space hence vulnerable for the attacks.
* Due to the day-to-day technological advances, the previously developed encryption system does not suit the modern technological advances.

3.2 Feasibility study

After requirement gathering, the team comes up with a rough plan of software process. At this step the team analyzes if software can be designed to fulfill all requirements of the user, and if there is any possibility of software being no more useful. It is also analyzed if the project is financially, practically, and technologically feasible for the organization to take up. There are many algorithms available, which help the developers to conclude the feasibility of a software  
project.

3.3 Development paradigm

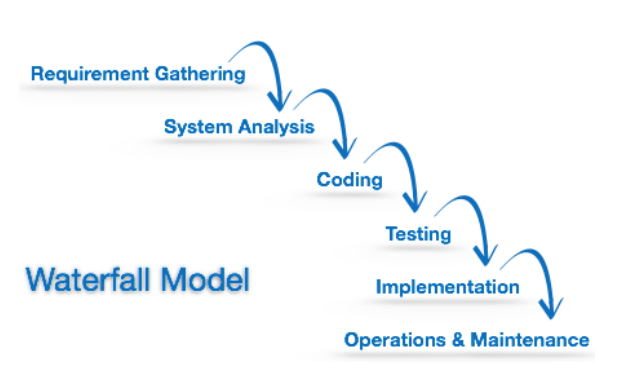
Software paradigm is often the steps and methods use while designing the software. It often falls into three categories;

* Software design paradigm
* Design
* Maintenance
* Programming
* Software development paradigm
* Requirement gathering
* Software design
* Programming
* Programming paradigm
* Coding
* Testing
* Integration

The software development paradigm helps a developer to select a strategy to  
develop the software. A software development paradigm has its own set of tools,  
methods, and procedures, which are expressed clearly and defines software  
development life cycle. In this project, I use waterfall model as my software development paradigm.

3.4 Waterfall Development model

Waterfall model is the simplest model of software development paradigm. All the  
phases of SDLC will function one after another in linear manner. That is, when the  
first phase is finished then only the second phase will start and so on.

Fig 1: Waterfall software model 

This model assumes that everything is carried out and taken place perfectly as planned in the previous stage and there is no need to think about the past issues that may arise in the next phase. This model does not work smoothly if there are some issues left at the previous step. The sequential nature of model does not allow us to go back and undo or redo our actions. This model is best suited when developers already have designed and developed similar software in the past and is aware of all its domains.

3.4 System requirement specifications

Software requirement specifications defines how the intended software will interact with hardware, external interfaces, speed of operation, response time of system, portability of software across various platforms, maintainability, speed of recovery after crashing, Security, Quality, Limitations etc

The system requirements are expressed in the following ways;

1. User requirements e.g. having the signup form, login form and meets the user experience.
2. Technical experience. This project needs the following hardware and software materials

* Eclipse IDE
* JDK version 8.1
* Computer with processor of 2.4GHZ, Harddisk 500gb and RAM of 2GB.
* USB flash disk
* Java books for referencing.
* Cryptographic and network security guide

CHAPTER FOUR

SYSTEM DESIGN

4.1 Overview of Advanced Encryption Standard (AES)

AES is the Advanced Encryption Standard, a United States government standard algorithm for changing the plain text to cipher text i.e. encrypting and decrypting the data. The National Institute of Standards and Technology (NIST) published a request for comments for the “Development of a Federal Information Processing Standard for Advanced Encryption Standard” on January 2, 1997. NIST searched for alternatives that have higher level of security than that offered by the other algorithms such as DES, IDEA and RSA. Data Encryption Standard (DES) which grew vulnerable to brute-force attacks due to its 56-bit effective key length. AES candidates were required to support a symmetric block cipher that supported multiple key lengths. The algorithm had to be publicly defined, free to use, and able to run efficiently in both hardware and software. The central design principle of the AES algorithm is the adoption of symmetry at different platforms and the efficiency of processing. After a 5-year standardization process, the NIST adopted the Rijndael algorithm as the Advance Encryption Standard (AES).

4.2 Rounds in AES algorithm

Length of the input output block and the State is 128 for AES algorithm. This is represented by Nb = 4, which reflects the number of 32-bit words (number of columns) in the State. For the AES algorithm 128, 192, or 256 bits is the length of the Cipher Key, K. The key length of the block is denoted by Nk and its value is 4, 6, or 8. This value reflects the number of 32-bit words (number of columns) in the Cipher Key.

**Table 1**: Key-Block-Round Combinations

|  |  |  |  |
| --- | --- | --- | --- |
| **Algorithm Block Size** | **Key Length (Nb words)** | **Block size**  **(Nk words)** | **Number of** **Rounds (Nr)** |
| AES-128 | 4 | 4 | 10 |
| AES-192 | 4 | 6 | 12 |
| AES-256 | 4 | 8 | 14 |

For the AES algorithm, during the execution of algorithm the numbers of rounds to be performed are dependent on the key size. Nr is used to represent the number of rounds. The combinations of Key-Block-Round that conform to this standard are given in table 1 above. AES algorithm uses a round function for both its Cipher and Inverse Cipher. This function is composed of four different byte-oriented transformations: A) Using a substitution table (S-box) byte substitution, B) By different offsets shifting rows of the State array, C) mixing the data within each column of the State array, and D) adding a Round Key to the State. No. of round keys generated by key-expansion algorithm is always one more than the actual no. of rounds present in the algorithm. Therefore the equation can be made as follows:  
Number of round keys = Nr+1 (1)  
We refer to the round keys as K0, K1, K2……….KNr

4.3 Algorithm

The input (block size Nb, also known as plaintext) of the AES algorithm is converted into a 4 x 4 array, called a state. Four transformations, *AddRoundKey, SubBytes,* *ShiftRows and MixColumns*, perform various operations on the state to calculate the output state (the final cipher  
text). Except for *AddRoundKey* each of these operations are invertible.  
InvMethod( Method( a ) ) = a (2).

If *AddRoundKey* operates on a variable twice, the variable itself is returned.

4.4 Transformation in AES

To perform all these transformations above, some mathematical operations are needed to understand which are given as below.

1. AddRoundKey Routine

The *AddRoundKey* routine is simple XOR addition of round key and a portion of expanded key into plaintext

Table2: XOR adding of the RoundKey to all states

|  |  |  |  |
| --- | --- | --- | --- |
| K0,0 | K0,1 | K0,2 | K0,3 |
| K1,0 | K1,1 | K1,2 | K1,3 |
| K2,0 | K2,1 | K2,2 | K2,3 |
| K3,0 | K3,1 | K3,2 | K3,3 |

XOR

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | a0,0 | a0,1 | a0,2 | a0,3 | | a1,0 | a1,1 | a1,2 | a1,3 | | a2,0 | a2,1 | a2,2 | a2,3 | | a3,0 | a3,1 | a3,2 | a3,3 | |

=

|  |  |  |  |
| --- | --- | --- | --- |
| B0,0 | B0,1 | B0,2 | B0,3 |
| B1,0 | B1,1 | B1,2 | B1,3 |
| B2,0 | B2,1 | B2,2 | B2,3 |
| B3,0 | B3,1 | B3,2 | B3,3 |

1. SubBytes

*Subbyte is* the SBOX for AES. It operates on each byte in the state and performs a non-linear substitution in the GF (28) field, which is what makes AES a non-linear cryptographic system. In order to be invertible each value of b’ must be generated from a unique value of b. A look up table can also be implemented for *SubBytes*. SubByte operation performs an affine transformation on the inverse of byte b, and adds it to 0xC6.

1. ShiftRows

*ShiftRows operates* on individual rows of the state. It provides diffusion throughout the AES algorithm. The first row is not changed. The second row is shifted one byte to the left, with the left most byte wrapping around. The third row shifts two bytes to the left, and the fourth row shifts three bytes to the left with appropriate wrapping to the right. This description is for AES-128, the number of shifts for each row changes based on the key size.

**Table3. Number of shifts in ShiftRows.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nb | Row 1 | Row 2 | Row 3 | Row 4 |
| 4 | 0 | 1 | 2 | 3 |
| 6 | 0 | 1 | 2 | 3 |
| 8 | 0 | 1 | 3 | 4 |

1. MixColumns

*MixColumns* operates on individual columns of the state. It provides diffusion throughout the AES algorithm. The columns are considered polynomials over GF(28) and multiplied modulo x4+1 with a(x) where a(x) = {03}x3 + {01}x2 + {01}x + {02} NOTE: x4+1 is relatively prime to a(x). This can be represented as a matrix equation:

This can be converted to a system of equations which can be computed per the addition and multiplication rules described in section 2. *InvMixColumns* can be described by the equation:

The AES algorithm loops through certain sections Nr times. The  
AddRoundKey is performed at the beginning and at the end of the cipher in order to provide initial and final randomness to the algorithm. Without this operation, anyone can easily deduct the first or last portion of the cipher, and therefore it would be irrelevant to the security of the cipher. In order to make the encryption and decryption routines more similar the last round in the cipher is completely different from the other rounds. It reduces the complexity in hardware, and software, implementations.

CHAPTER FIVE

SYSTEM IMPLEMENTATION

5.1 Overview of system implementation

I have successfully implemented the AES text encryption using **JAVA code and itel-Pentium** Processor using **Eclipse IDE as a compiler**. I got all the desired results of encryption as well as correct decryption of text message and produces original message. The maximum positions in the input message as well as in key should not exceed 16 positions in length. Because I use the 128 bit size of data and key. Therefore each position whether it is alphabet, number, symbol or blank space is encrypted into 8 bit sequence.

**Number of positions = 128/8 = 16 (3)**

One block comprises total sixteen bytes of 8 bit sequence format. From this block first four bytes forms first column elements. In such a way next four bytes forms second, third and fourth column elements. One row contains total four letters and 32 bits. Four such rows are placed one  
below another. It forms 4\*4 sizes square of blocks. This block is then further encrypted as well as decrypted.

5.2 Flowchart of the system

The following pseudo code summarizes the implementation of the proposed system

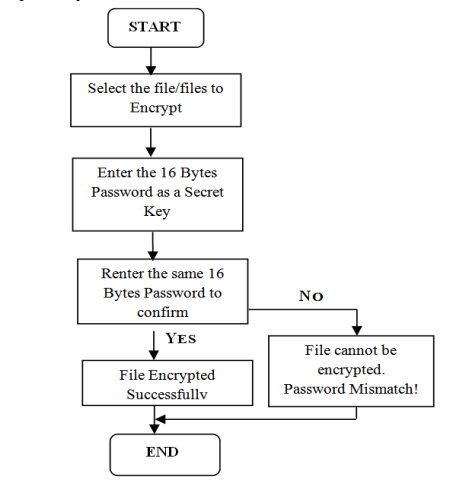


FIG2: Encryption Process

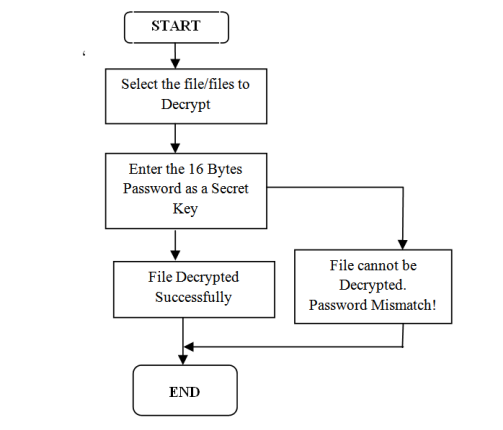


FIG3. Decryption process.

5.3 Algorithm implement in the proposed system

Algorithm Start (FileName, SecretKey) {  
Browse files  
Enter 16 byte SecretKey  
}  
Algorithm Encrypt() {  
Convert to State Array  
AddRoundKey()  
SubBytes()  
ShiftRows()  
MixColumns()  
Key Expansion  
}

Algorithm Decrypt {  
Convert to State Array  
AddRoundKey()  
InvSubBytes()  
InvShiftRows()  
InvMixColumns()  
Key Expansion  
}

5.4 The working of the proposed system  
1. Initially the user selects the file from the disk for the encryption process.  
2. On selecting the files, the user must enter 16 bytes SecretKey as a password.  
3. After entering the 16 bytes SecretKey, the user must re-enter to confirm password and Click on encrypt.  
4. The encrypted file is created with FilenameEncryption.txt in a disk. The encrypted files are set as read only.  
5. To decrypt the encrypted text files, go to the decrypt screen. Select the encrypted files and enter the same SecretKey which is used for encryption process.  
6. In this system, the encrypted files cannot be deleted and delete  
option in right click menu is disabled for all encrypted files. This  
feature provides security.

CHAPTER SIX

SYSTEM TESTING AND INTERGRATION

6.1. Unit testing

I carried out this type of software testing during the development of this system. This form of testing ensures that each detail of the implementation is logically correct. This testing I carried out using the Eclipse IDE which was my major development tool.

6.2. Functional testing

I carried out the functional testing on every module of this system to ensure each of the modules performs and functions correctly according to the design specifications. During the functional testing I checked the core applications functions i.e. plain text encryption and cipher text decryption.

Apart from that, I also checked text input including the use of the extended characters, string handing and accurate hot-key shortcuts without any duplication.

6.3. System testing

This testing is also called end-to-end testing, tests a complete integrated system to verify that the system meets its requirements.

In this text file encryption system, I tested the sign up and login interfaces to verify that the right data format was only allowed. Apart from that, I tested that the text file was encrypted and decrypted using the same secret key such that no other key was allowed. Lastly I tested the logoff interface, allowing the user to exit his/her account when clicked logout button.

6.4 Database testing

This is the essential type of the software testing. I tested the database during the software development process to verify that only accepted data values were allowed to the tables fields of the database.

CHAPTER SEVEN

RESULTS AND OBSERVATIONS

7.1 Results of the encryption and Decryption process

The program is compiled using the eclipse IDE of windows operating system to compile java programs. In the resultant screenshot we have performed both encryption and decryption. For the encryption the following was the user GUI where the user was expected to choose the file which he or she wants to encrypt.

First create the text in notepad text-editor and save as the myfile.txt

Fig4. Example of the plain text

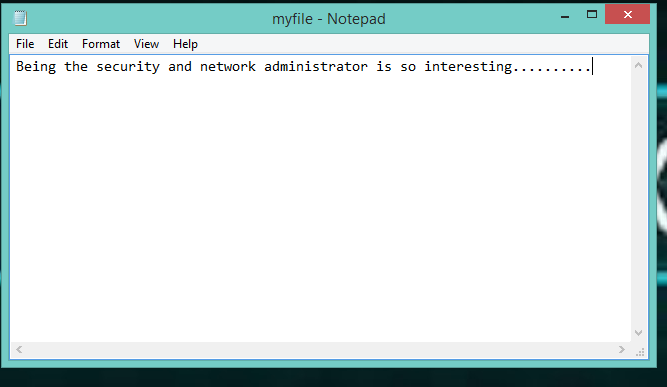
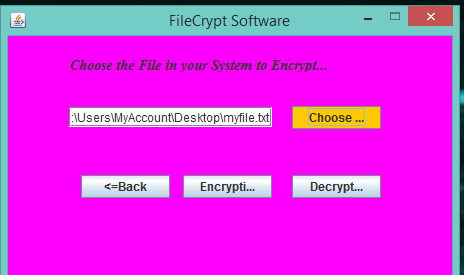


Fig5. GUI for choosing the file from the PC.



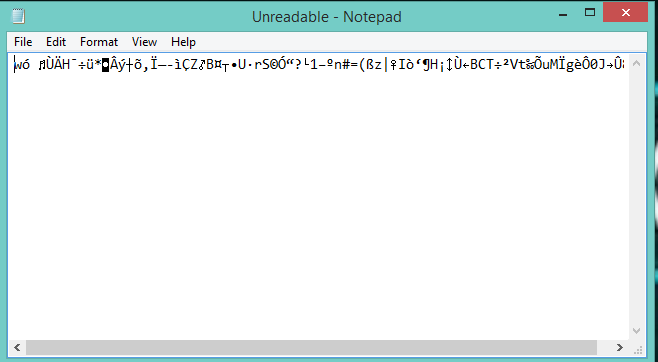
After choosing the file, the key to be used in encryption is input as shown in the figure below

Fig6. Java GU for entering the encryption key



On clicking the submit button the figure above the selected text is encrypted as shown in the figure below

Fig7. The encrypted text using AES Password-Based encryption algorithm.



The process of the decryption is the reverse of the encryption process. A single unique key is use for both encryption and decryption.

* 1. Observations

**Observation 1** concludes that any normal text along with symbol in both key as well as input can be easily encrypted and decrypted.  
**Observation 2** concludes that any same number sequence written in key as well as input produces the decrypted message which contains encrypted message also.  
**Observation 3** concludes that any number different types of number sequence written as a key and input produce decrypted output along with encrypted message. It means that if we write numbers in key and input message at that time decrypted message contains original message along with input message.  
**Observation 4** concludes that if we write same text message in input as well as key it will perform normal encryption and decryption.  
**Observation 5, 6 and 7** concludes that AES is case sensitive algorithm if case of input message only or key only or both input message and key are change then it will **produces a different encrypted message** and expected decrypted message.  
**Observation 8** if we write letters along with symbols in place of key and if input provided is in number form it will perform normal encryption and decryption and provide input message as a decrypted message.

CHAPTER EIGHT

CONCLUSION AND RECOMMENDATIONS

8.1 Conclusions

This paper gives the successful implementation of text encryption as well as the decryption. Due to combination of Java code, Eclipse IDE, JDK tool and Pentium Intel processor PC a unique solution for text encryption was able to be achieved. This text encryption uses 128 bit size of key as well as plaintext. Each word or space is converted into 8 bit sequence. Therefore maximum total 16 positions are recognized by this code. Letters, spaces, numbers or symbols can be written to these 16 positions. Results showed here uses different numbers, symbols and letters for text encryption. All the observations clearly conclude the behavior of text encryption code.

8.2 Recommendations

The proposed system can be extended to standard video coding systems such as those using MPEG and other video formats. All the existing costly encryption products will have no use in future if the video encryption also invented with royalty free open source software. Therefore it will be the most flexible and cheaper solution. In this world of internet encryption of every multimedia data is the basic need of all communication systems. Tremendous change in technology will appear if this solution is free of cost. Today mobile phones are most important and habitual thing for each human being. In the same way this encryption technology will become the basic need of everyone.

APPENDICES

User guide

The following are steps to be followed by the user when using the fileEncrypt software;

* Create an account by inserting the username, first name, last name, valid email address and password.

Note: One cannot encrypt or decrypt any text file without having an account.

* Secondly, login into your account by inserting the valid username and password as per the signup credentials.
* After that, interface for choosing the file either to encrypt or decrypt will appear. Choose the file from your computer and then click **Submit** button.
* On clicking the submit button, another window will pop up where the user is expected to insert the secret key for either encrypting or decrypting the file.

Note: Secret key should be kept secure to prevent the intruder from tampering with the file which was once encrypted.

* Then click submit button and the file will be either encrypted or decrypted.

Test Results

The following are the tested results of the fileEncrypt software

1. 128 bit key: mmustpass@123456 (16 positions)

INPUT MESSAGE: computing is everywhere (16 positions)

Encrypted Message: ²ƒñ;Ù²PTnž·D¯WËÞ-ƒjÆ5¤F0Ñ]³šYŒò

Decrypted Message: Same as the input

1. 128 bit key: computerscience1 (16 positions)

INPUT MESSAGE: computer science has great ideas (16 positions)

Encrypted Message: Õžg]Ög98Xº“zªtÒ! ­XÔð?ýž¤šÊÄŸ‑¯#ÿ™<n“`Ë6Œ¸

Decrypted Message: Same as the input

1. 128 bit key: ilovecoding0000 (16 positions)

INPUT MESSAGE: Java is the most important programming language (16 positions)

Encrypted Message: È(³„êfmlNA20Šu×ÕËìÀugü±ZâòûŽ«C¹æ< ‘ÆÍ0­…54?)àC\*2»T

Decrypted Message: Same as the input

1. 128 bit key: computersecurity (16 positions)

INPUT MESSAGE: Mmust computer science is nowadays a school by itself (16 positions).

EncryptedMessage::kËM)[ÏSéê÷\_M#Ö›„x0•n)^ílØØÀƒšæª\*Zþ}dîòMCÓã”›V dp]õ

Decrypted Message: Same as the input

Code Listings

1. Sample of Filechooser code

FileInputStream file=**new** FileInputStream (textField.getText ());

FileOutputStream output=**new**FileOutputStream("C:\\users\\public\\project\\Readable.txt");

1. Sample of encryption code

**try**{

**byte** k[]="FiZi1701NuLI5252".getBytes();

SecretKeySpec key=**new** SecretKeySpec(k,"AES");

Cipher enc=Cipher.*getInstance*("AES");

enc.init(Cipher.***ENCRYPT\_MODE***, key);

CipherOutputStream cos=**new** CipherOutputStream(output,enc);

**byte**[] buf=**new** **byte**[1024];

**int** read;

**while**((read=file.read(buf))!=-1){

cos.write(buf,0,read);

}file.close();

output.flush();

cos.close();

JOptionPane.*showMessageDialog*(**null**, "Successfully Encrypted the Text file");

}**catch**(Exception e){

JOptionPane.*showMessageDialog*(**null**, e);

}

}

1. Sample of decryption code

**try**{

**byte** k[]="FiZi1701NuLI5252".getBytes();

SecretKeySpec key=**new** SecretKeySpec(k,"AES");

Cipher enc=Cipher.*getInstance*("AES");

enc.init(Cipher.***DECRYPT\_MODE***, key);

CipherOutputStream cos=**new** CipherOutputStream(output,enc);

**byte**[] buf=**new** **byte**[1024];

**int** read;

**while**((read=file.read(buf))!=-1){

cos.write(buf,0,read);

}

file.close();

output.flush();

cos.close();

JOptionPane.*showMessageDialog*(**null**, "Successfully Decrypted the Encrypted Text File");

}

**catch**(Exception ex)

{

JOptionPane.*showMessageDialog*(**null**, ex);

}

}

**REFERENCES**

1. **Neumann P (1994).** Computer Security. Issues in Science and Technology, pp.50-54.
2. **Schneider C (1996).** Applied Cryptography. John Wiley & Sons, Inc., New York pp.20-250.
3. **Yao Feng, He Chengwan, Hu Hongyin**,”A File Encryption Method Using Multiple Encryption Algorithms”, School of Computer Science and engineering, Wuhan Institute of Technology University, 2010.
4. Advance Encryption Standard, [Online], Available: <http://en.wikipedia.org/wiki/Advanced_Encryption_Standard>.
5. **U.S. DEPARTMENT OF COMMERCE, NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY**: ADVANCE ENCRYPTION STANDARD (AES).William Roche, “The Advanced Encryption Standard, The Process, Its Strengths and Weaknesses”, University of Colorado, Denver, Spring 2006 Computer Security Class, CSC 7002, Final Paper May 6, 2006.
6. **Network Security**, “Cryptography” , University of Houston, USA Website: http://www2.cs.uh.edu/~jhyang/network/module18.ppt
7. **Devin N. Shah**:”Information Security: Principles and Practice”.