

**Improving AES Algorithm using Multiple Secret Keys**

تحسين خوارزمية AES باستخدام المفاتيح السرية المتعددة

**Prepared by**

**Suhair Haseeb Al-Waith**

**Supervisor**

**Prof. Hamza Abbass Al-Sewadi**

**A Thesis Submitted in Partial Fulfillment of the  
Requirements for the Master Degree in Computer Science**

**Department of Computer Science**

**Faculty of Information Technology**

**Middle East University**

**Jan. 2019**

## Authorization

I, **Suhair Haseeb Al-Waith**, authorize Middle East University to provide hard copies or soft copies of my thesis to libraries, institutions and establishments upon request.

Name: Suhair Haseeb Al-Waith

Date: 26 / 01 / 2019

Signature:



## Thesis Committee Decision

This thesis titled "Improving AES Algorithm using Multiple Secret Keys" was successfully defended and approved on **26/1/2019**.

### Thesis Committee Members

### Signature

(Head of the Committee and Supervisor)

Prof. Hamza Abbass Al-Sewadi

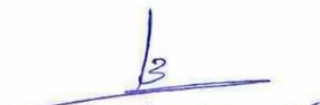
Middle East University

  
4 - 2 - 2019

(Internal Examiner)

Dr. Mohammed Abbas AL-Husainy


Middle East University

  
5/2/2019

(External Examiner)

Dr. Shadi almasadeh

Al-Israa University

  
4 - 2 - 2019

## **Acknowledgment**

I thank my God and thank him very much for giving me the patience and the will to continue my studies and help me to complete this thesis.

I would like to express my sincere appreciation to my supervisor Prof. Hamza Abbass Al-Sewadi for his advice and patient guidance throughout my Master's Thesis.

I thank all my respected lecturers at the Faculty of Information Technology, Middle East University.

**The Researcher  
Suhair AL-Waith**

## Dedication

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

( وَقُلْ رَبِّ زِدْنِي عِلْمًا )

(اية 114) {سورة طه}

قال رسول الله (صلى الله عليه وسلم)

اللهم لا سهل إلا ما جعلته سهلاً، وأنت تجعل الحزن إذا شئت سهلاً"

To all my family who gives me support, and to all my friends,  
And colleagues who wish me success and progress in my study

**The Researcher**  
**Suhair AL-Waith**

## Table of Contents

<b>Subject</b>	<b>Page</b>
Title .....	I
Authorization.....	II
Thesis Committee Decision .....	III
Acknowledgment.....	IV
Dedication.....	V
Table of Contents.....	VI
List of Tables.....	VIII
List of Figures.....	IX
List of Appendixes.....	X
Table of Abbreviations .....	XI
English Abstract .....	XII
Arabic Abstract.....	XIII
<b>Chapter One: Background of the Study and its importance.....</b>	<b>1</b>
1.1 Introduction .....	2
1.2 Problem Statement .....	2
1.3 Motivation.....	3
1.4 Questions of the Study.....	3
1.5. Objectives of Study.....	4
1.6. Scope and Limitations .....	4
1.7. Thesis Outlines.....	5
<b>Chapter Two: Theoretical Background and Literature Review.....</b>	<b>6</b>
2.1 Introduction.....	7
2.2. Cryptography.....	7
2.2.1 Data integrity, Confidentiality, and authentication.....	8
2.3. Classification of cryptographic algorithm.....	9
2.4. Encryption Methods.....	11
2.5. Decryption Methods.....	12
2.6 Comparison between DES, AES, and RSA.....	13
2.7 Symmetric algorithms.....	16
2.7.1 AES algorithm.....	16

2.7.1.1 AES Cryptanalysis.....	21
2.7.2 DES algorithm.....	22
2.7.3. Blowfish Algorithm.....	24
2.8 Asymmetric Encryption algorithms.....	28
2.8.1 RSA Algorithm.....	28
2.8.2 RSA Cryptanalysis.....	30
2.9. Review of Related work.....	31
<b>Chapter Three: Study Methodology and proposed work.....</b>	<b>34</b>
3.1 Introduction .....	35
3.2 proposed technique.....	35
3.2.1 The Encryption key (Ks).....	36
3.2. 2 Encryption process.....	38
3.2.3 Secret Key seed encryption.....	40
3.2.4 Decryption process.....	42
3.3 PRNG.....	45
<b>Chapter Four: Evaluation and Experimental Results .....</b>	<b>47</b>
4.1 Introduction.....	48
4.2. Resources used for the Implementation.....	48
4.3. The program main Screens.....	49
4.3.1 Encryption process screen.....	49
4.3.2 Decryption process screen.....	51
4.3.3 Encryption without PRNG.....	52
4.4 Results.....	54
4.4.1 Processing time versus message segment length.....	54
4.4.2 Processing time versus key seeds.....	56
4.4.3 Processing time versus secret keys.....	59
4.4.4 Processing time versus message length.....	61
4.4.5 Comparison with traditional AES algorithm.....	63
4.4.6 Key strength comparison.....	67
<b>Chapter Five: Conclusions and Future Work.....</b>	<b>68</b>
5.1 Conclusion.....	69
5.2 Future Work.....	71
References.....	72

## List of Tables

Chapter No. -Table No.	Table Contents	Page
Table 2.1.	Comparison between AES, DES and RSA	14
Table.2.2	AES, RSA algorithms functions	15
Table 2.4	Comparison of DES, AES and Blowfish.	27
Table 4.1	Encryption and Decryption processing time for 128-bit encryption key.	55
Table 4.2	Encryption and Decryption processing time for 192-bit encryption key.	55
Table 4.3	Encryption and Decryption processing time for 256-bit encryption key.	56
Table4.4	Encryption and Decryption Processing times for various seeds (50 characters segment length).	57
Table4.5	Encryption and Decryption Processing times for various seeds (1000 Characters segment length).	58
Table 4.6	Processing times for various secret keys (50 characters segment length).	60
Table 4.7	Processing times for various secret keys (1000 characters segment length)	60
Table 4. 8	Encryption and Decryption processing times versus message lengths.	61
Table 4.9	Processing times Comparison for 128-bit encryption sub-key length.	63
Table 4.10	Processing times Comparison for 192-bit encryption sub-key length.	64
Table 4. 11	Processing times Comparison for 256-bit encryption sub-key length.	64
Table 4.12.	Processing time for AES and proposed algorithm for different message lengths.	65
Table 4.13	Lists the key sizes, data block size, and key space for the most widely used cryptosystems.	67



## List of Figures

Fig. Number	Contents	Page
Fig. 2.1	Encryption / Decryption processes	8
Fig. 2.2	Types of ciphers	11
Fig. 2.3	Symmetric-key encryption Algorithm	13
Fig. 2.4	Asymmetric-key encryption Algorithm	13
Fig. 2.5	AES algorithm Structure	17
Fig. 2.6	AES algorithm Encryption	19
Fig. 2.7	AES algorithm Decryption	19
Fig.2.8	AES (Advanced Encryption Standard) processes	21
Fig. 2.9	flowchart DES Algorithm	24
Fig.2.10	Block diagram for Blowfish cryptosystem	26
Fig 2.11	RSA Algorithm implementation	29
Fig.3.1	The elements of the secret encryption key (Ks) for the proposed algorithm	37
Fig.3.2	Block diagram for the Encryption process	39
Fig.3.3	Message Splitting and Merging Mechanism	41
Fig.3.4	Flowchart of the encryption process	42
Fig.3.5	Block diagram of the proposed decryption Process	44
Fig.3.6	Cipher text splitting and merging	45
Fig.4.1	Example for the Encryption the process	51
Fig.4.2	Example for the decryption process	52
Fig.4.3	Example for encryption and decryption using NO PRNG process.	53
Fig. 4.4	An example screen shot for Encryption and Decryption	59
Fig. 4.5	results	62
Fig 4.6 (a)	Processing time for AES and proposed algo. For different message lengths.	66
Fig 4.6 (b)	Processing time for AES and proposed algo. for different message lengths	66

## List of Appendixes

<b>Appendix</b>	<b>Contents</b>	<b>Page</b>
A	The main screen parts for the implementation of the proposed alg.	75
A	The Encryption process screen	75
B	كتاب الاستلال	76

### Table of Abbreviations

<b>Abbreviations</b>	<b>Meaning</b>
AES	Advanced Encryption Standard
CAST-128	Carlrics Adams And Stafford Tavares
DES	Data Encryption Standard
DSA	Digital Signature Algorithm
DH	Diffie Hellman
D	Decryption
IV	Initialize vector
IDEA	International Data Encryption Algorithm
LH	Left half
MD5	Message Digest 5
M	Message
NIST	National Institute of Standards and Technology
PGP	Pretty Good Privacy
PRNG	Pseudo-Random Number Generator
RSA	Rivest, Shamir, and Adelman
RC2	Rivest Cipher 2
RH	Right half
SHA	Secure Hash Algorithm

## **Improving AES Algorithm using Multiple Secret Keys**

**Prepared by**

**Suhair Haseeb Al-Waith**

**Supervisor**

**Prof. Hamza A. Al Sewadi**

### **Abstract**

In today's IT world, Encryption and Decryption has become a key factor for the technological transformation achieving the business goals due to its extensive list of advantages. This thesis looks into various available security measures, then proposes an algorithm that is based on file segmentation and encryption scheme. It implements the concept of multiple elements key for the encryption of the message segments. It is intended to encrypt each message segment with different secret key by utilizing a pseudo-random number generator (PRNG) that generates a sequence of codes of any required length according to a selected input encryption key seed of certain length. The algorithm adopts the use of the strong advanced encryption standard (AES) algorithm for the message segments encryption, with the option of using any of the three key length options provided by the AES algorithm, namely the 128-bit, 192-bit, and the 256-bit key lengths. Therefore, the secret encryption key length for the proposed algorithm would consist of multiple elements, initial key, message segment length, PRNG seed, and one of the three AES key length choices. The secret key for this algorithm is then can be exchanged between the sender and receiver using one of the asymmetric cryptosystems, such as the widely used Rivet-Shamir-Adleman (RSA) algorithm, hence, the proposed algorithm takes advantage of both symmetric and asymmetric cryptographic techniques.

**Keywords: security, Cryptography, AES algorithm, Encryption, Decryption, RSA algorithm.**

## تحسين خوارزمية AES باستخدام المفاتيح السرية المتعددة

اعداد

سهير حسيب الواعظ

اشراف

الأستاذ الدكتور حمزة السوادي

الملخص

في عالم تكنولوجيا المعلومات اليوم، أصبح التشفير وفك التشفير عاملاً أساسياً للتحويل التكنولوجي الذي يحقق أهداف النشاط التجاري من خلال قائمة واسعة من المزايا. تبحث هذه الأطروحة في مختلف التدابير الأمنية المتاحة، ثم تقترح خوارزمية تستند إلى تجزئة الملفات ونظام التشفير. ينفذ مفهوم العناصر الرئيسية المتعددة لتشفير الأجزاء. الغرض منه هو تشفير كل مقطع رسالة بمفتاح سر مختلف عن طريق استخدام مولد أرقام عشوائية زائفة (PRNG) يقوم بإنشاء تسلسل من الرموز لأي طول مطلوب يعني وفقاً لبذرة مفتاح تشفير مدخلات محددة بطول معين. تعتمد الخوارزمية استخدام خوارزمية معيار التشفير المتقدم القوي (AES) لتشفير شرائح الرسالة، مع اختيار استخدام أي من اختيارات طول المفتاح الثلاثة التي توفرها خوارزمية AES أي 128بت، 192بت، وأطول 256بت، سيتكون طول مفتاح التشفير السري للخوارزمية المقترحة من عناصر متعددة، ومفتاح أولي، وطول جزء الرسالة، وبذور PRNG، وإحدى اختيارات طول مفتاح AES الثلاثة. يمكن بعد ذلك تبديل المفتاح السري لهذه الخوارزمية بين المرسل والمستقبل بواسطة أحد أنظمة التشفير غير المتماثلة، (RSA) Rivet-Shamir-Adleman المستخدمة على نطاق واسع الخوارزمية، وبالتالي، فإن الخوارزمية المقترحة تستفيد من تقنيات التشفير المتماثلة وغير المتماثلة.

الكلمات المفتاحية: الأمن، التشفير، الخوارزمية AES، التشفير، فك التشفير، خوارزمية RSA

## **Chapter one**

### **Background of the Study and its importance**

## **Chapter one**

### **Background of the Study and its importance**

#### **1.1. Introduction**

Due to the vast use of distributed data storage on computer networks and the ever-increasing communication all over the world in various applications (personal, social, and governmental) in all walks of life, the data and information safety and security on the computing environment are of great concern nowadays. So, a lot of digital data security algorithms, techniques, protocols, and measures were developed and implemented. For each security measure or cryptosystem some cryptanalysis efforts occur that tries to breach that cryptosystem. Hence still, there is a great deal of research efforts to counter the cryptanalysts by developing new algorithms and improving previous ones in order to enhance the data and computer security.

#### **1.2. Problem Statement**

As the use of computer for data and information storage is inevitable nowadays, the data and information security is a hot topic, as numerous security issues and challenges such data integrity, data theft, privacy issues, infected application, data loss, security on vendor level, and data location.

Protecting data and information is crucial and many cryptographic and hiding techniques are in use nowadays, but they are either time consuming or not secure enough, hence this research is going on to develop and produce more efficient systems. The time for breaking the security of any encryption algorithm is very important and it is usually influence by the algorithm sophistication and the used secret key strength. Brute force search is an exhaustive search that can be opposed by size of the key space,

hence this thesis looks for methods to strengthen the data security by looking into these two factors.

### **1.3 Motivation:**

Since the data security problem is under attacks and trial to breach, hence there always be a need for new or enhanced data security algorithms. As the larger the encryption key space, the harder for any cryptanalysis to break the security, therefore it is thought that it is possible to enhance the key space of any of the currently strong algorithms, such as AES. So a method is suggested and will be studied for benefiting from the strength of AES in addition to the implementation of other possible arrangement such as segmentation, variable key for each segment to increase the security.

### **1.4 Questions of the Study:**

Some questions that illustrate the problem discussed in this research:

1. What are the possible improvement in security strength if the secret encryption key length is improved using one of the secured algorithms, such as AES algorithm for encryption?
2. What would be the cost for the key enhancement proposed key space in this research work?
3. How would the use of pseudo random number generator (PRNG) improve the security of algorithm? And what are the benefits of using such PRNG for the security enhancement?
4. What is the effect of the private key length for different messages on the encryption and decryption processing time?



### **1.5. Objectives of Study**

The goal of this research is to improve the data security and provide requirements for an adoption of such system for data storage and information exchange over the communication networks between users. Therefore, it includes the following:

1. Suggest a new method to enhance the security in general using message segmentation technique for data encryption/decryption using the widely used AES algorithm with the aid of Pseudo – random – number generator (PRNG).
2. Implement a secret encryption/decryption key that consists of multiple elements in order to enhance the key space.
3. Investigate the encryption and decryption execution times for the proposed segmented message algorithm and compare them with those for the traditional AES algorithm (without segmentation).

### **1.6 Scope and Limitations**

This research will cover the investigation of the proposed segmented message algorithm for various values of important parameters, such as the secret key types, the PRNG seed values, the message segments lengths, and the different AES key lengths. Measurements will include the processing execution times of the proposed algorithm and its comparison with those for the Traditional AES algorithm.

This research will be limited to the point of designing an encryption/decryption scheme for storage and/or transmission of digital data or files between communicating parties, which can be client to client or client to server.

### **1.7. Thesis Outlines:**

The thesis is organized in five chapters. The contents of these chapters are given in the following:

**Chapter One:** An introduction to the thesis with the problem statement, research scope and limitations.

**Chapter two:** Theoretical Background and Literature review.

**Chapter three:** Methodology and proposed work.

**Chapter four:** This chapter presents the implementation of the proposed algorithm.

**Chapter five:** Conclusions and Future Work.

**Chapter Two**

**Theoretical Background and Literature**

**Review**

## Chapter Two

### Theoretical Background and Literature review

#### 2.1. Introduction

To protect important and confidential data from being leaked or exposed to any non-authorized body whether intentionally or unintentionally, there are two ways, either converting the data into unintelligible form, (i.e. encryption/decryption) or by hiding the data into another data. The first technique is called cryptography and the second is called steganography. This thesis is concerned with cryptography, hence this chapter will include some theoretical background and literature review on the related work in cryptography.

#### 2.2. Cryptography

Cryptography is a science of secret writing. It is the technique that converts readable, intelligible, and clear message into an unreadable, cipher, and supposedly secure message. Hence, it is used for making the confidential data safe when it stored or being transmitted over the communication devices. There are two types of encryption/decryption methods; symmetric and asymmetric way of communication depending on the type of key(s) and algorithms used, the first used single secret key for both encryption and decryption and it is called “secret key cryptography”, (e.g. DES, AES, IDEA), and the second use two different keys, one is private and the other is public, hence it is called “public key cryptography”, such as RSA, Diffie Hellman etc.

The generic data encryption process can be summarized as shown in the block diagram of Fig 2.1. It consists of three components, encryption at the sender side, decryption at

the receiver side, and the Channel connecting the sender and the receiver. The plaintext input message  $M$ , is encrypted by the encryption algorithm using the key  $K_1$  producing the cipher text  $C$ . This cipher text  $C$  is transmitted over the channel which is usually unsecure, to be received and recovered by the decryption algorithm using the key  $K_2$ . If  $K_1 = K_2$ , the system is cryptosystem is symmetric, otherwise it asymmetric system.

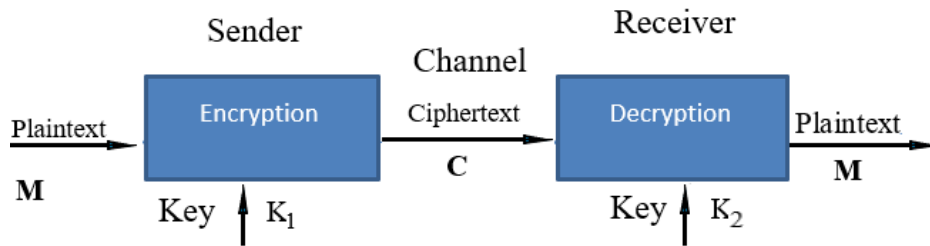


Fig (2.1) Encryption / Decryption processes

### 2.2.1 Data integrity, Confidentiality, and authentication: (Ayushi,2010)

The main security services for data and information storage and transmission are data integrity, confidentiality, non-repudiation, and authentication. These services will be defined below.

**1-Data Integrity:** Data integrity is to ensure that data is protected from accidental or intentional modification. Integrity is usually provided by message authentication hashes or code. A hash value is a fixed length numeric value derived from a sequence of data. Hash values are used to verify the integrity of data sent through insecure channels. The hash value of the received data is compared to the hash value of the data as it was sent in order to determine if the data was modified or not. Any information has a value only if it is correct, hence data integrity means maintaining and assuring the accuracy and consistency of data its implementation for computer systems that stores data, processes or retrieve that data.

**2-Authentication:** means for determining whether someone is in fact, who is claimed or declared to be. Authentication is proving you are who you ask to be. In a computer center or systems, a password check is conducted to check the authenticity at session start. In distributed systems, the use of authentication server, and is usually based on ability to encrypt/decrypt a message

**3- Non-Repudiation:** someone cannot refuse the authenticity of their signature and the sending a message that they originated. Non-repudiation use in a session is that in an e-commerce system, non-repudiation service is recall several times and to stages. Non-repudiation service is required in the implementing, and payment phases. we want to view a series of message transfers as a series of request-response messages between two parties A and B: A sends the first message; B processes the received message and formulates the answer and sends it back to A; A processes the reply and formulates the next message for B; and so on. The process continues for a sequence of messages until one side calls for termination of the session.

**4-Confidentiality:** is usually using Encryption algorithms (that use certain encryption keys) to convert plaintext into ciphertext. Symmetric encryption algorithms use the same key for encryption and decryption, while asymmetric algorithms use a public/private key pair. The loss of confidentiality is related to the loss of privacy, unauthorized access to information and identity theft.

### **2.3. Classification of cryptographic algorithm:**

Cryptographic algorithms can be classified into two types, based on the number of keys that are employed for encryption and decryption, and defined by their application and

use. The block diagram for generic cryptographic algorithm is illustrated in Figure 2.1.

These two types are as follows.

**Secret Key Cryptography (SKC):** Uses a single key for both encryption and decryption, i.e.  $K_1 = K_2$ , hence this type is called symmetric encryption system. It is used for privacy and confidentiality.

**Public Key Cryptography (PKC):** Two keys are used, one key for encryption and another for decryption, i.e.  $K_1$  and  $K_2$  are different but they mutually related to each other. This type is called asymmetric encryption system. It is usually used for authentication, non-repudiation, and key exchange. (Gary C. Kessler, 2018).

Another classification primitive used in symmetric cryptographic systems that depend on way a message is encrypted known as block ciphers and stream ciphers, and can be defined as follows.

- **Block Ciphers:** in this type, the plaintext messages are segmented into block of equal sizes or number of bits for example 16, 32 64, etc. then each block is encrypted using the used algorithm. If a block was shorter than the adopted block size, some padding is used to complete it. Then all the encrypted blocks are concatenated and sent to the receiving side, who will also segmented to the same sized and decrypt it block by block to finally recover the original message. DES and AES are examples of block ciphers.
- **Stream Ciphers:** Stream ciphers encrypt the messages elements one after the other, i.e. it is like the block ciphers but with very short blocks, or the message element. For example, the block might be single bit of byte only. It is usually faster than block cipher with fact that the encryption transformation can change for each symbol of plaintext being encrypted. RC4 is examples of stream cipher types.

Due to the fact that stream ciphers encrypt the messages one unit at a time, it is preferred over the block cipher encryption is applied in limited storage capability devices. Besides, stream ciphers do not have error propagation, which is very useful characteristic as compared with block cipher whenever the probability of transmission errors is high.

Figure 2.2 illustrate a summary for the cryptographic algorithm classification (Gurpreet Singh, Supriya,2013).

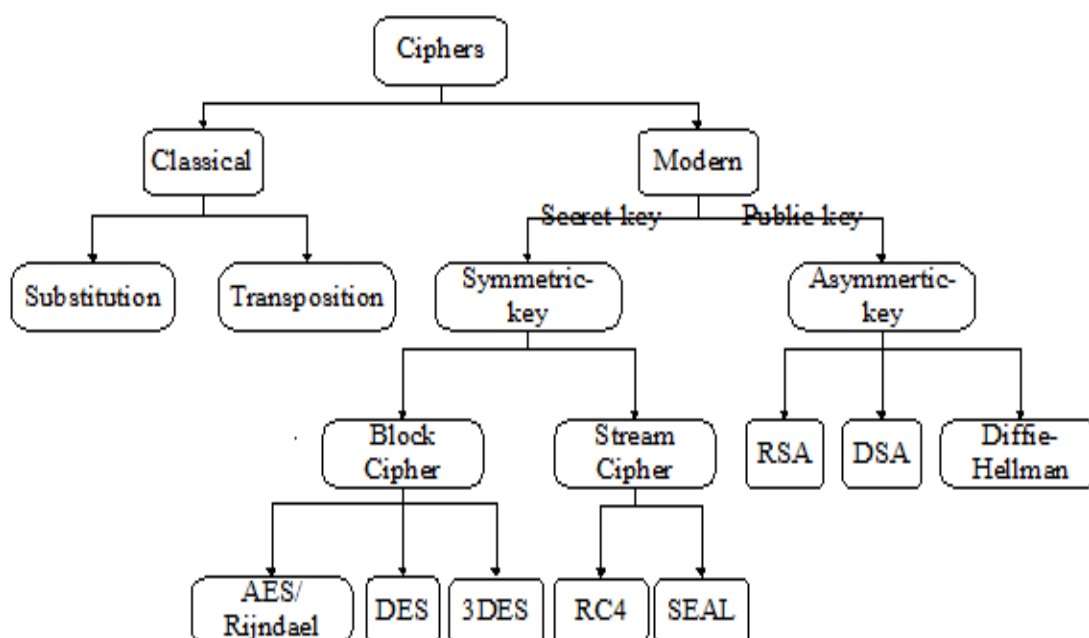


Fig 2.2 Types of ciphers (Gurpreet Singh, Supriya,2013)

## 2.4 Encryption Methods:

Process to convert the data into some another form that appears to be random, usually done by using a secret encryption key and a cryptographic cipher. It also means that encryption is the process of transforming plaintext into the cipher text where plaintext



[is the input to the encryption process and cipher text is the output of the encryption process.

Symmetric encryption is known as private-key cryptography, because the key used to encrypt and decrypt the data must be one secure key, a sender encrypts the data with one key, sends the data (the ciphertext) and then the receiver uses the same key to decrypt the data. There are many popular algorithms used now a days for symmetric encryption, such as DES, 3DES, AES, BLOWFISH, etc. (Ayushi, 2010 and Tanisha,2013).

On the other hand, asymmetric encryption algorithm uses one key for encryption and another different but related key for decryption. Examples of asymmetric algorithms in use now a days are RSA, Diffie-Hellman, DSA, etc.

## **2.5. Decryption Methods:**

Decryption process is the inverse of the encryption process in order to convert the message back to its original content. The receiver uses a decryption algorithm and a key to transform the ciphertext back to original plaintext, it is also known as decoding process.

A mathematical process used for decryption that generates original plaintext as a result of any offered ciphertext and decryption key is known as Decryption algorithm.

The keys used for encryption and decryption could be similar and dissimilar depending on the type of cryptosystems used (i.e., Symmetric key encryption and Asymmetric key encryption).

The block diagrams for the symmetric cryptographic algorithm is illustrated in Fig 2.3 and that for asymmetric cryptographic algorithms is illustrated in Fig 2.4 (Satyajee R. Shinge, Rahul Patil,2014).

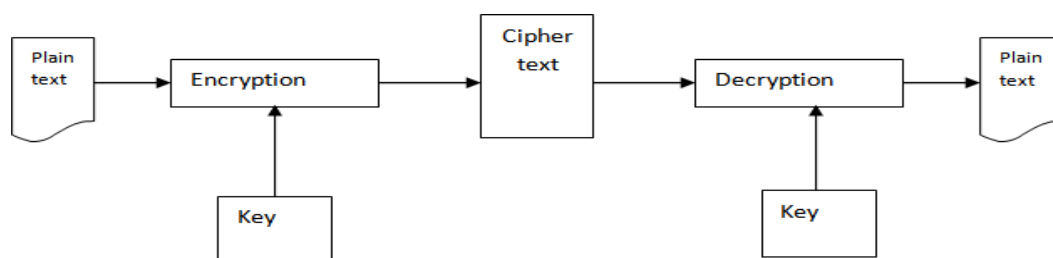


Fig 2. 3 Symmetric-key encryption Alg. (Satyajeet R. Shinge, Rahul Patil,2014)

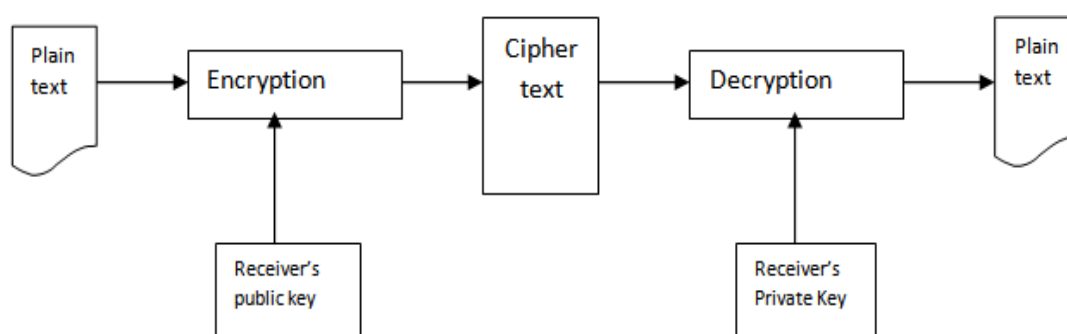


Fig 2. 4. Asymmetric-key encryption Algorithm (Satyajeet R. et, al.2014)

## 2.6 Comparison between DES, AES, and RSA

DES and AES algorithms are examples of symmetric algorithm, while RSA algorithm is an example of asymmetric algorithm., Table 2.1 present a useful comparison for their features and characteristics. These comparisons are selected from (Gurpreet Singh, Supriya, 2013 and Gambhir A. ,2014).

Table 2.1. Comparison between AES, DES and RSA (Singh G., et, al, 2013 and Gambhir A. ,2014)

Method	AES	RSA	DES
Developed Date	2000	1978	1977
Encryption speed	Faster	Slower	Moderate
Decryption speed	Faster	Slower	Moderate
Algorithm	Symmetric algorithm	Asymmetric algorithm	Symmetric Algorithm
Platform	Cloud Computing	Cloud Computing	Cloud computing
Key Size	128,192,256 bits	1024 bits	56 bits
Block Size	128 bits	1024 bits	64bits
Capacity of Encryption data	Large amount of data	Small data	Less than AES

More comparisons are included in Table 2.2, showing detailed features of AES and RSA algorithms. (Prerna Mahajan, Abhishek Sachdeva, 2013).

Table 2.2. AES, RSA algorithms functions (Prerna Mahajan, et, al, 2013)

<b>Item</b>	<b>AES</b>	<b>RSA</b>
<b>Nature of algorithm</b>		
Type of algorithm	Symmetric	Asymmetric
Block size	128,192, 256 bits	Minimum 512 bits
Data size	Big	Small
Rounds	10,12,14	1
Requirement of memory	Less	More
<b>Processing time</b>		
Encryption/Decryption	Fast	Slow
Speed of computation	Fast	Slow
Software Implementation	Fast	Slow
Hardware Implementation	Fast	Slow
<b>Data security level</b>		
Secure services	Confidentiality	Confidentiality, Integrity, nonrepudiation.

Table 2.1 and Table 2.2 have shown that in general the symmetric cryptographic algorithms are more secure and have high speed compared with asymmetric cryptographic algorithms, however, they suffer of very serious difficulty of key distribution. Besides, they do not lend themselves for authentication and digital signature. On the contrary, asymmetric algorithms provide good way for key transfer, but they are slower than symmetric algorithm if used for encryption / decryption purposes. In the following sections, some useful explanation of AES, DES, Blowfish, and RSA algorithms will included.

## 2.7 Symmetric algorithms

In the following, some theoretical descriptions will be included for three symmetric cryptographic systems, namely:

1-Advanced Encryption Standard (AES).

2-Data Encryption Standard (DES).

3-Blowfish algorithm.

### 2.7.1 AES algorithm

The Advanced Encryption Standard (AES) algorithm is a digital computer security standard that was published by NIST in Nov 2001, Based on a competition won by Rijmen and Daemen (Rijndael) from Belgium, Rijndael allows many data block sizes and key sizes, AES features are restricted to the following:

- Message Block size: 128 bits, all 128 bits are encrypted, and
- Key sizes: 128, 192, 256 (AES-128, AES-192, AES-256) .

For example, when we need to send sensitive data in an e-mail, we must know the right password for the decryption of the encrypted text. AES algorithm is an iterative rather than Feistel Cipher. It is also based on ‘substitution–permutation network .

It is a series of linked operations, some of which and others involve shuffling bits Around (i.e. permutations).

The important part is that the key length does not affect the block size but the number of Repetitions of transformation rounds varies with the key size, namely 128-bit key is 10 cycles, for 192 bit it is 12 cycles and for 256 bit is 14. And Key scheduling are 44, 52, or 60 sub keys respectively, having length =32 bit.

Figure 2.5 presents the general structure of AES algorithm showing only 1<sup>st</sup> round and the, 9<sup>th</sup> and 10<sup>th</sup> round . All rounds from 2<sup>nd</sup> to 8<sup>th</sup> round are exactly the same as round 9. Each round consists of four operations; substitute byte, shift rows, mix columns, and add round key as will be explained next.

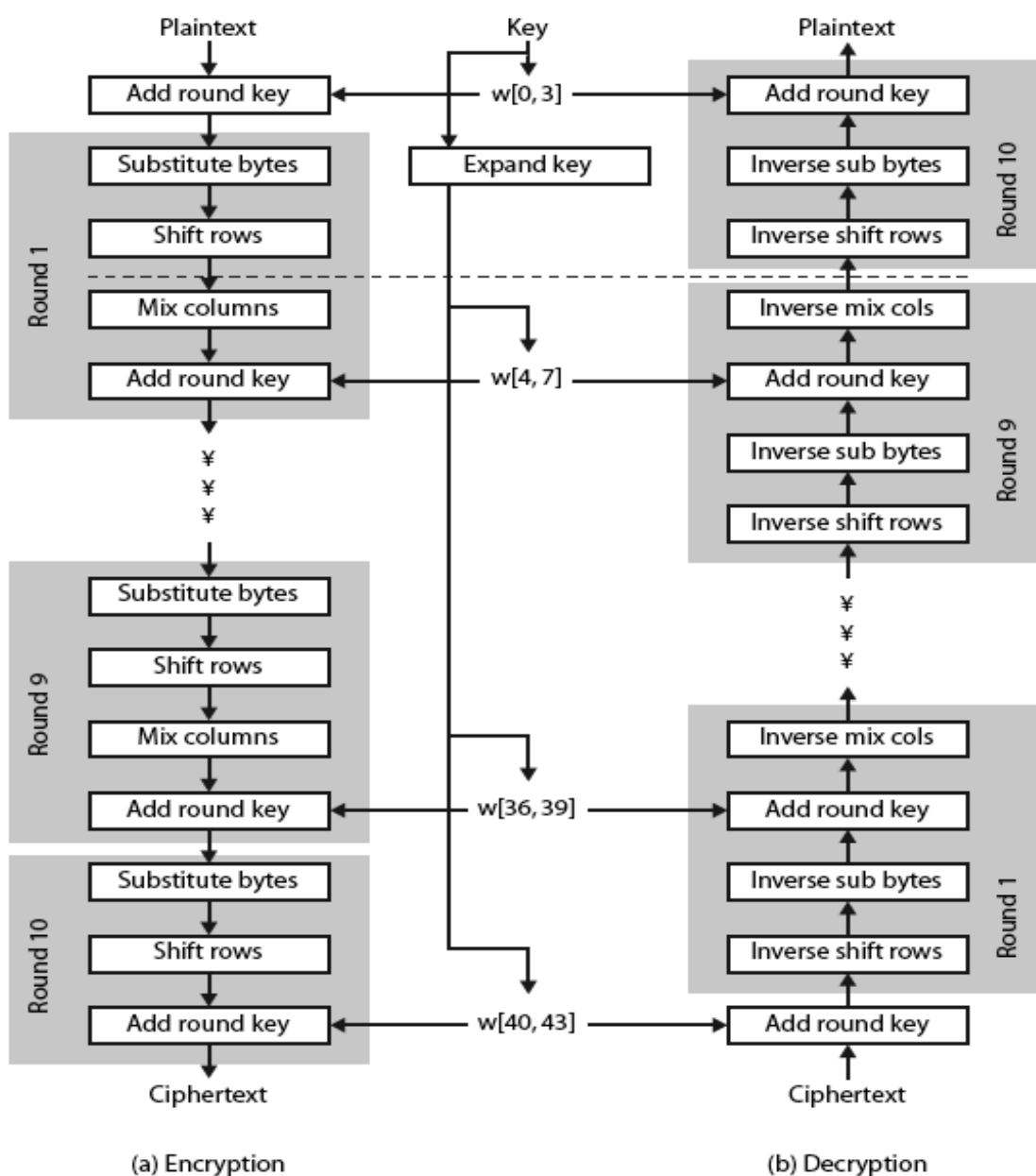


Fig 2.5. AES algorithm Structure [ Ali E. Taki El\_Deen , 2013]

## AES Structure

Four different stages are used in AES algorithm, as shown Fig 2.5:

Substitution bytes: Use S-box to perform byte-to-byte substitution of the block.

1. Shift rows : A simple permutation
2. Mix columns: A substitution that makes use of arithmetic operations.
3. Add round key: A simple bit wise XOR of the current block with the portion of the expanded key.

The algorithm has the following characteristics :

- Resistance against all known attacks.
- Input to the encryption algorithm ,decryption algorithm is a single 128-bit block .
- Speed and code compactness on a wide range of platforms
- Design simplicity

An overall block diagrams for encryption and decryption processes are shown in Fig 2.6 and Fig 2.7. The detailed description of AES operation and implementation can be seen in many good references such as: (Ali E. Taki El\_Deen,2013) and (Diaa Salama AbdElminaam, 2018).

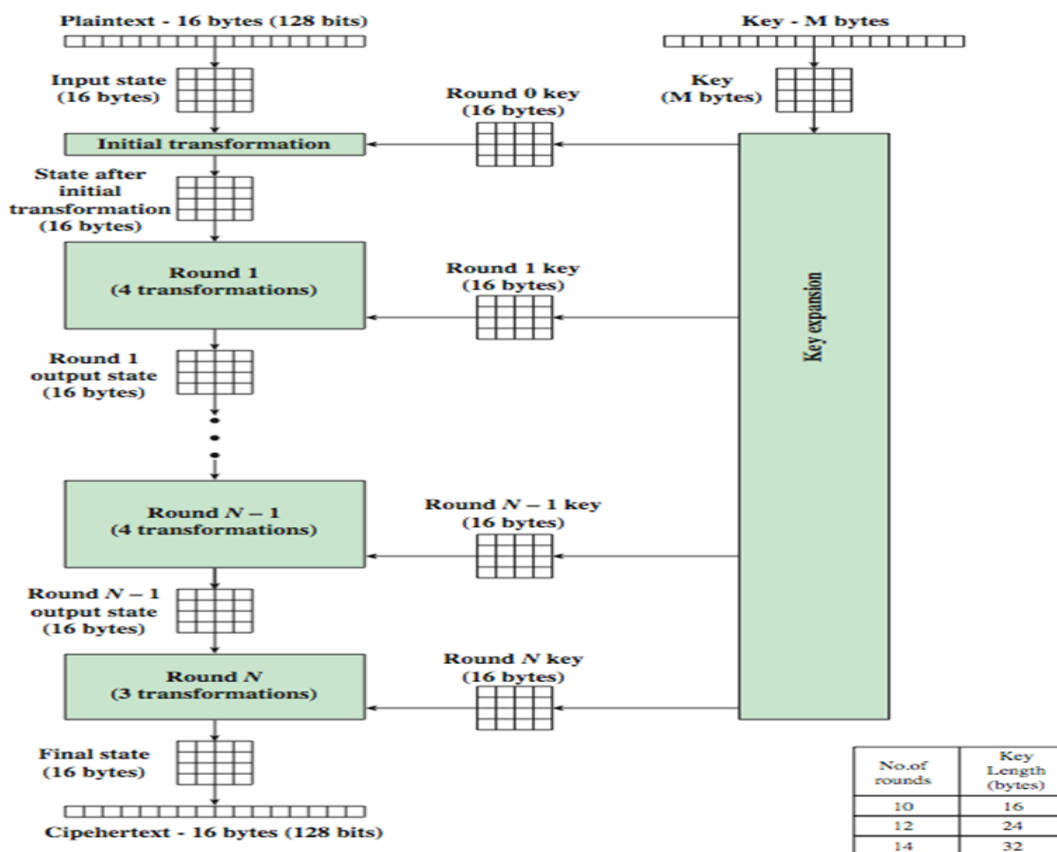


Fig 2.6. AES algorithm Encryption [Uma Naik, V. C. Kotak, 2014]

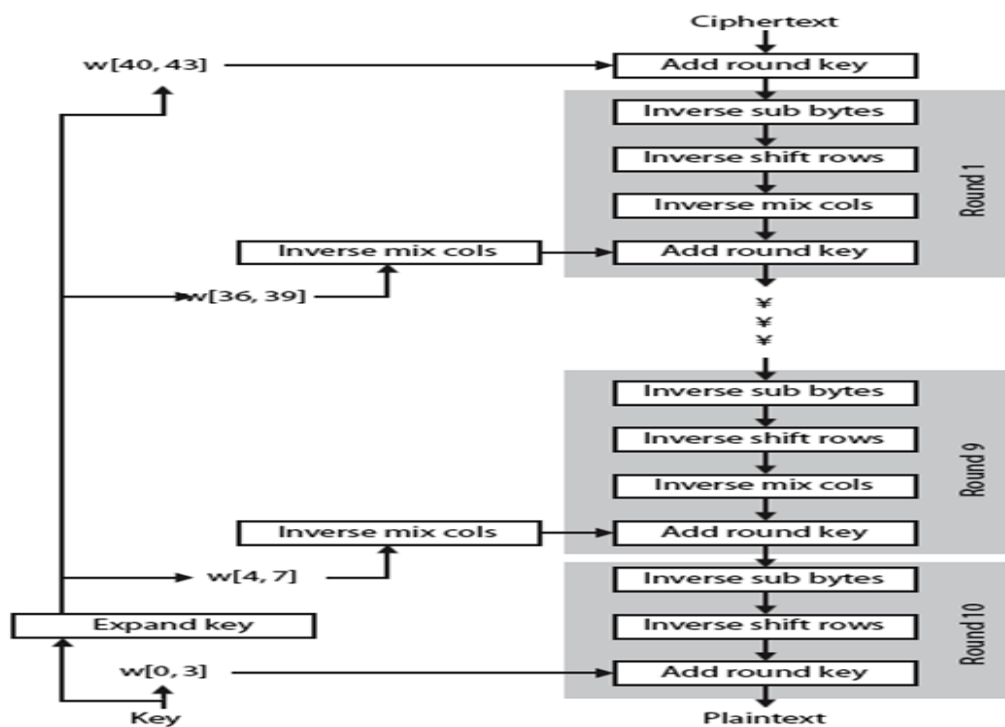


Fig 2.7. AES algorithm Decryption [Raj Jain, 2011]



AES operates faster and more efficient than many other symmetric encryption algorithms. The transmission of data is considered there is insignificant difference in performance of different symmetric key schemes. For data transfer it would be advisable to use AES scheme in case the encrypted data is stored at the other end and decrypted multiple times. Reducing the number of rounds leads to power savings but it makes the protocol comparatively insecure for AES and should be avoided. It is claimed that seven or more rounds can be secure enough, however, ten rounds were taken in the 128-bit key length version of the standard AES (Anand Kumar and S. Karthikeyan,2012).

AES is largely considered secured to all attacks, with the exception of brute force, which attempts to decipher messages using all possible types in the 128, 192, or 256-bit cipher. But security experts believe that AES will eventually be standard for encrypting data in the private sector.

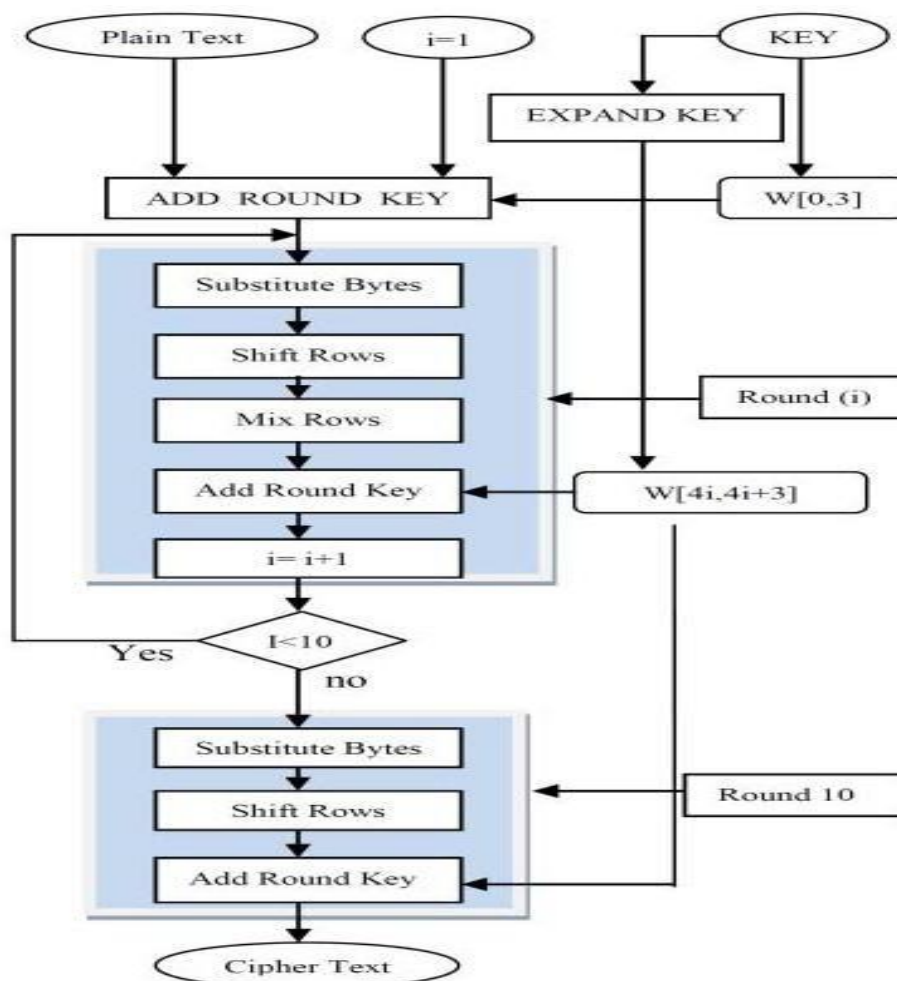


Fig 2. 8. AES (Advanced Encryption Standard) processes,

(Gurpreet Singh, Supriya, 2013)

### 2.7.1.1 AES Cryptanalysis

To use any system without fear, its vulnerability and threat are researched regularly. As AES is used for classified and unclassified security purposes in important applications, it has shown that it is secure for many reasons such as;

- No symmetry properties are noticed in its round mapping.
- No key recovery attacks exist that are faster than exhaustive search.
- No semi-weak keys or weak keys are identified like what was noticed in the DES system.

### 2.7.2 DES algorithm:

DES algorithm was adopted by the National Bureau of Standard NBS in the year 1977, and since then it was the widely used symmetric system internationally in most data storage and communication applications. It is a block cypher encrypting a 64-bit data with a 56-bit key in multi-round manipulation, substitutions and transpositions. DES system processes can be summarized as in the following and illustrated in

Fig 2.9.

1. An initial Permutation.
2. 16 rounds of a complex key dependent computations.
3. A final permutation, being the inverse of the initial permutation.

DES algorithm includes an input of 64-bit long plaintext (or a stream of 64 bits) data block and a secret key of 56-bit length (in addition to 8 bits used as parity) and generates output of 64-bit block of cipher text. If the input data was less than or greater than 64 bits, it pads the last block of such input data with some type of zeros, ones, or multiple ones and zeros, to make it a complete block (64-bit block or a multiple of 64 bits). The plaintext block was then entered as input to an Initial Permutation (IP) to shift the bits around.

A key generating algorithm is used to generate 16 keys, each of 48 bits, to be used for the 16 rounds of the encryption algorithm. The 8 parity bits of the entered key is removed from the key using an initial permutation table step, reducing the 64-bit key to 56- bits. After this step, the remaining 56 bits of the key are processed in 16 rounds operations to produce 16 keys, each of 48 bits length, as in the following operations.

1. The key is divided into two 28-bit halves.
2. Every half of the key is round-shifted (rotated) by one or two bits, depending on the round.

3. The halves were recombined after each shifting and input to a compression permutation in order to reduce the key from 56 bits to 48 bits.

4. The rotated key halves from, were used in next round to generate another 48 bit key to be used for the next encryption round and so on.

The message data block of 64 bits is encrypted in the following steps:

1. First the scrambled according to an initial permutation process.
2. The 64 bits are divided into two halves, left half (LH) and right half (RH), with both halves having a 32-bit data block.
3. The right half (RH) is processed by an expansion permutation to increase its size to from 32 bits to 48 bits.
4. The output of step 3 is XOR'ed with the 48-bit key obtained from the key generation algorithm.
5. Now this output goes through a substitution process that uses 8 substitution tables in such a way that produce 32 bits block, i.e. compressed to reduce the 48-bit block down to 32-bits.
6. The output of step 5 is XOR with left half (LH) of the input data block, and the two data halves were swapped prior to the next round's input.

This process is repeated for (16) times in (16) rounds using 16 keys based on operations.

7. The result of the 16<sup>th</sup> round goes into an inverse permutation process (the inverse of the initial permutation).

Then the output is the cipher output block which will sent to the receiving side (Adedeji Kazeem B. and Ponnle Akinlolu A.2014).

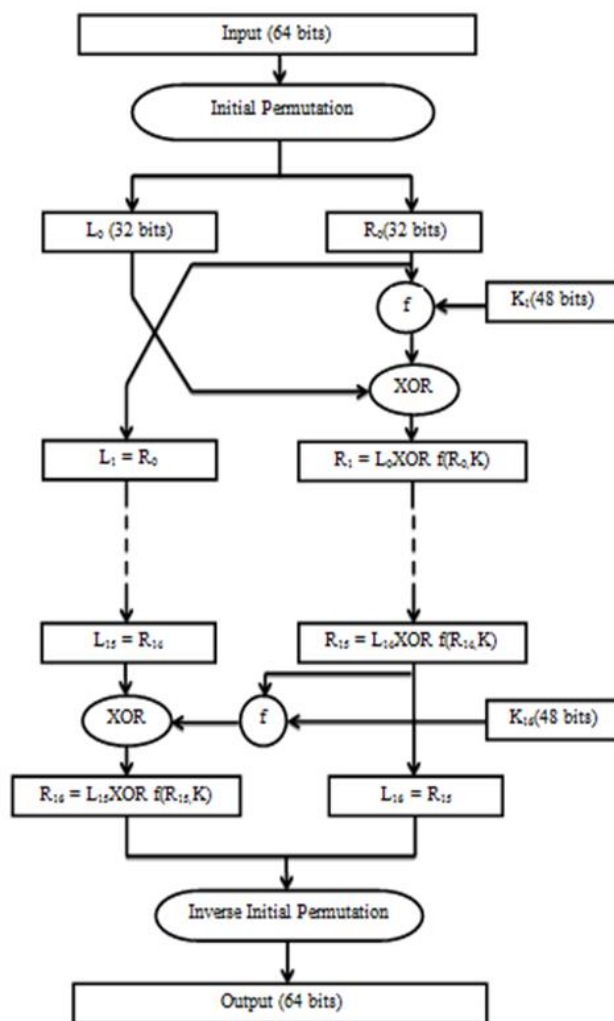


Fig 2.9. flowchart DES Algorithm. (Prerna Mahajan & Abhishek Sachdeva,2013)

### 2.7.3. Blowfish Algorithm:

It is a 64-bit symmetric block cipher with variable length key. The algorithm operates with two parts: a key expansion part and a data- encryption part. The role of key expansion part is to converts a key of at most 448 bits into several sub key arrays to 4168 bytes.

The data encryption is done in a 16-round Feistel network. Each round content of a key based on permutation, a key and data-based on substitution. All operations are EX-ORs and additions on 32-bit words Blowfish is after to Two fish.

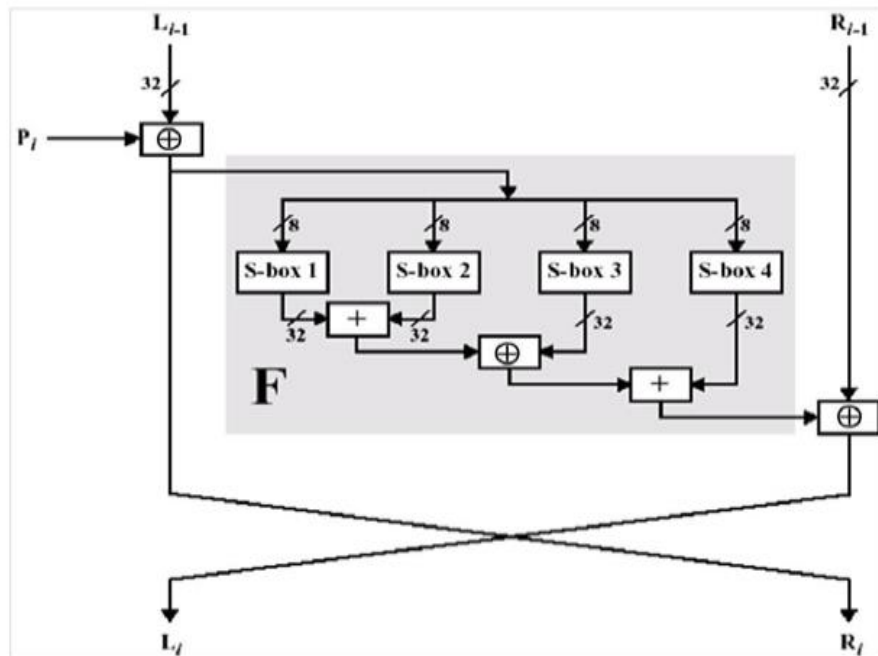
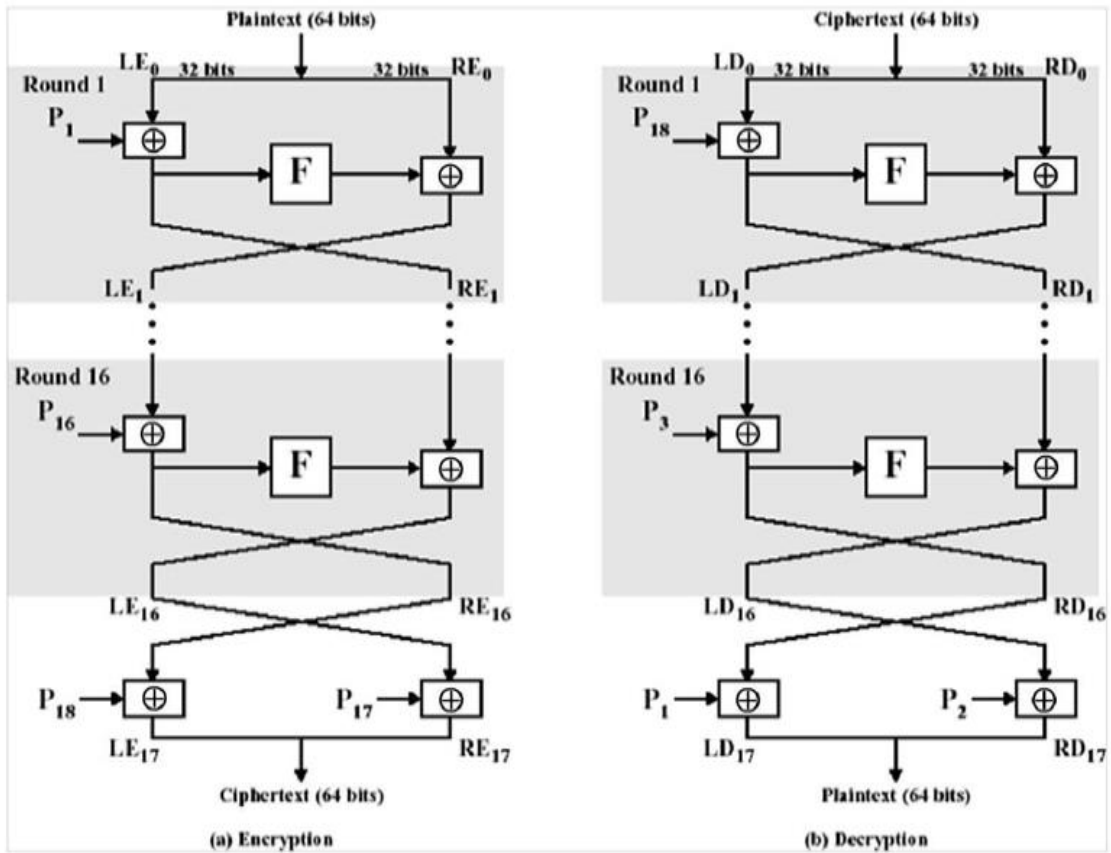
Two commonly used symmetric encryption algorithms such as Blowfish and Rijindael. Blowfish was designed in 1993 by Bruce Schneier as a fast encryption algorithms compared to other encryption algorithms. It is suitable for applications if the key is not change multiple.

Blowfish is divided into two types: a key expansion part used to convert a key to maximum of 448 bits into various sub-keys array and another is data encryption part that take place in 16 round Feistel network. where each round contains key and data dependent substitution and key dependent permutation. It provides a good level of security even performing less number of iteration rounds of encryption. Its process of creating keys is based upon the current XOR operation and that feature provides more security to its structure. All the keys must be computed before any encryption and decryption process start. It is very difficult to crack the security of this algorithm because, having long key size it will be more strong security (Shikha Rani, Shanky Rani, 2016).

Blowfish uses two primitive operations, Addition: - of words, denoted by +, is performed modulo 232 Bitwise exclusive-OR: This operation is denoted by the structure is a slight variant of classic Feistel network.

Left (L) and right (R) sides are both processed in each round, 16 rounds, with two extra XORs at the end.

The plaintext is splitted into two 32-bit halves LE0 and RE0. The resulting ciphertext is contained in the two variables LE17 and RE17. The function F details are shown in Fig 2.10, which outlines the block diagram of encryption and decryption processes of Blowfish algorithm (Stallings, W.).



(C) The F function of Blowfish

Fig 2.10. Block diagram for Blowfish cryptosystem (Stallings, W.)

Explanation method:

Initialize the P-array and S-boxes. XOR P-array with the key bits. For example, P1 XOR (first 32 bits of key), P2 XOR (second 32 bits of key). This new output is now P1 and P2. Encrypt the new P1 and P2 with the modified sub keys, This new output is now P3 and P4. (Ali E. Taki El\_Deen,2013).

This comparison of DES, AES and Blowfish in Table 2.4.

Table 2.4. comparison of DES, AES and Blowfish, (Aarti Devi1, Ankush Sharma2, Anamika Rangra3,2015).

<b>Factor</b>	<b>DES</b>	<b>AES</b>	<b>Blowfish</b>
Encryption / Decryption	Slow	Fast	Fast
Block Size	64 bits	128 bits	64 bits
Key size	64 bits (56 bits are actually used)	128 bits, 192 bits, and 256 bits	From 32 bits to 448 bits
Round	16	10,12,14	16
Speed depends on Key	Yes	Yes	No
Security	Insecure	Secure	secured, but less aim cryptanalysis than other algorithms.
Hardware and software implementation	Designed for hardware and quite slow in software	Fast in both in hardware and software.	Designed for software.



## 2.8 Asymmetric Encryption algorithms:

They are called asymmetric algorithms, because they use two keys; one for encryption and the other for decryption. They are referred to in most cases as public-key cryptography. They differ from symmetric algorithms in speed of operation, security strength, and also in suitability for certain applications.

The asymmetric algorithms were only invented in the last quarter of the twentieth century after the discovery of the one-way mathematic relation reported by Diffie and Hellman. As it is found that some computation and evaluations works in one way but does not work in opposite way. One of the early and widely used algorithm that implements this principle is RSA algorithm. This algorithm will be explained in the following section.

### 2.8.1 RSA Algorithm

RSA algorithm is a public key encryption algorithm developed by Ron **Rivest**, Adi **Shamir** and Len **Adleman** in 1977 (at MIT). It can used for encryption/decryption of messages, digital signature, and secret key distribution for the symmetric systems. It is asymmetric key cryptographic algorithm uses the prime numbers to generate the public and private key based on mathematical factoring and multiplying large numbers together, and the use of modular exponentiations.

The RSA algorithm steps are: (Ankit Gambhir ,2014).

- Choose two large prime numbers  $p$  and  $q$  that  $p$  is not equal to  $q$  .
- Calculate  $n$ , by multiplying  $p$  and  $q$ ;  $n=p*q$
- Now calculate  $\Phi(n)$  by formula  $\Phi(n)= (p-1) *(q-1)$
- Select a public key  $e$  such that  $e$  is not the factor of  $\Phi(n)$  .
- Next is to select the private key  $p$  such that  $(D*E) \bmod \Phi(n)=1$  .

- To calculate ciphertext (C) for a message M:  $C = M^e \bmod n$  .
- To calculate plaintext (M):  $C^d \bmod n \Rightarrow M$  .
- The ciphertext C is sent to receiver and at receiver decrypts it into plaintext.

For implementing RSA algorithm, all required operations and processes are summarized in the text boxes shown in Fig 2.11.

#### Key Generation

- Select random and secret two large primes p, q and check that  $p \neq q$ .
- Compute modulus  $n = pq$ .
- Compute phi,  $\phi(n) = (p - 1)(q - 1)$  .
- Select random public exponent: e such as  $1 < e < n$  and  $\gcd(e, \phi) = 1$
- Compute d such as  $e \cdot d \equiv 1 \pmod{\phi(n)}$  and  $1 < d < \phi(n)$  .
- Public Key: (n, e)
- Private Key: (n, d)

#### Encryption Algorithm

- Obtain the recipient's public key (n, e).
- Represents the plaintext message as positive integer m
- Compute the ciphertext  $c = m^e \bmod n$

#### Decryption Algorithm

- Uses his private key (n, d) to compute integer  $m = c^d \bmod n$
- Extract the plaintext from the integer representative m

Fig 2.11. RSA Algorithm implementation

### 2.8.2 RSA Cryptanalysis

There are four types possible attacks on RSA algorithm (Puneeth M. Jasmine shafi Farha, N. Sandhya, M. Yamini, 2015), as shown in the following.

1. **Brute-force attack:** Brute-force attack is an attack to trying use all the possible private keys.
2. **Mathematical attacks:** the attack which defines several type in effort for factorizing product of the two prime numbers
3. **Timing attacks:** depend upon the time is taken to decrypt the algorithm .
4. **chosen cipher text attacks:** only RSA algorithm

## 2.9. Review of Related work :

1. Ramaraj, Karthikeyan and Hemalatha, (2009) designed the new security type using hybrid encryption technique. The hybrid encryption technique is a combination of both private key and public key techniques. It provides all the three cryptographic services; integrity, confidentiality and authentication. The new design used Symmetric cipher system (AES Rijndael) and public key cryptography (RSA) with hash function.
2. Ali E. Taki El\_Deen (2013), implemented Advanced Encryption Standard (AES) algorithm to work on the Substitution Permutation network. AES has many mathematical formulas, and only requires one pass to encrypt data. and designed to be fast, unbreakable and able to support the smallest computing devices imaginable. The big difference between AES and Triple-DES are not only strength of security, but performance and better use of resources. Also Advanced Encryption Standard is not only assures security but also improves the performance in a variety of settings such as smartcards, hardware implementations etc. There are currently no known, non-brute-force direct attacks against AES.
3. Veerpal Kaur, Aman Singh (2013) defined some privacy and security-related issues that has been providing the study of past 20 years in the search for hybrid cryptographic algorithms that may help researchers to orientate their study areas and to choose various cryptographic algorithms for their studies. AES and DES have limited scope of use because of the problem of key management. Diffie-Hellman being very secure is the prior choice for eliminating various limitations

of cryptographic algorithms. Sample of papers that demonstrates the advantages and limitations of the widely used cryptographic algorithms.

4. Rachna Arora and Anshu Parashar, (2013) explained the symmetric Advanced Encryption Standard (AES), Data Encryption Standard (DES), and Asymmetric cryptographic algorithm (RSA). Algorithms like: RSA , AES , DES , Blowfish have been used and compared with each other for security of data on cloud computing environment. DES, AES, Blowfish are symmetric key algorithms.
5. Purna Mahajan & Abhishek Sachdeva, (2013), used AES and DES algorithm show very minor difference in time taken for encryption and decryption process. Encryption algorithm has very important role in communication security. Their work tested the performance of existing encryption techniques like AES, DES and RSA algorithms, and also studied by them and were based on the text files used and the experimental result it was concluded that AES algorithm consumes least encryption and RSA consume longest encryption time. They also observed that decryption of AES algorithm is better than other algorithms, i.e. much better than DES and RSA algorithm
6. Shakeeba S. Khan, Prof.R.R. Tuteja, (2015) explained Encryption Standard (DES) and the asymmetric cryptographic algorithm (RSA). And they explained how to merge such as DES and RSA in two different algorithms, and they also eliminated the security challenges of Cloud Storage.

7. Hardik Gandhi, Vinit Gupta, Indra Rajput, (2015) presented a Modified RSA Encryption Technique based on Multiple Public Keys. They Proposed an efficient implementation of RSA algorithm using two public key pairs and applying mathematical logics rather than sending the encryption key (e) value directly as a public key. If an attacker can get the value, he can directly find decryption key (d) value and decrypt the message. They approached for the newly created way to encrypt the given message converted to other form and then it is converted to cipher text and then it is sent to the communication channel so that cryptanalyst cannot attack to the encryption algorithm for getting the prime numbers. This approach makes RSA more secure.
  
8. Vinita Keer, Dr.Syed Imran Ali, Neeraj Sharma, (2016) looked for cryptographic algorithms used in cloud computing: They developed a definition of hybrid encryption as combining two or more cryptographic methods to take advantage of each method to protect data on the cloud. Modern Cryptography are implemented (e.g. AES, RC6, and DES).

## **Chapter Three**

### **Methodology and proposed work**

## Chapter Three

### Methodology and proposed work

#### 3.1 Introduction

This chapter summarizes the proposed technique for enhancing the data security and encryption/decryption algorithm that involve message segmentation. It also investigate the encryption/decryption processes using the widely used cryptosystems, namely Advanced Eryption Standards, AES, which implemented in the proposed algorithm and RSA. This chapter will briefly outline the suggested modified cryptosystem that enhances the file security storage and transfer over the network. A cryptosystem is an implementation of cryptographic techniques and their accompanying infrastructure to provide information security services. In this model, encryption of the segmented message is done by AES algorithm with involvement of a pseudo random number generator (PRNG) to supply various encryption keys for each segment, and (RSA) is used for the secured communication between users and/or servers. (Darshan Upadhyay, Prof. Chirag Pate, 2012).

#### 3.2 proposed technique:

The objective of the proposed work is to develop, design and test a new cryptographic algorithm that would be suitable for secured data storage and secure data transmit on network communication as well as cloud computing environment.

The proposed technique is a method of encryption/decryption that combines more than one processing algorithm. It's design includes a combination of file slicing technique for segmenting the file or the message to be encrypted, a powerful encryption algorithm



(such as AES). a Pseudo Random number generator (PRNG) to provide different key for each message segment from one secret key seed, and one choice of encryption key length. In addition to the use of public key cryptosystem.

This design is adopted in order to take benefit of the strengths of some types of cryptographic system in order to achieve the four security principle services, namely “Confidentiality”, “Integrity”, “Authentication”, and “Non-Repudiation”.

The following sections will describe the Encryption Key for the Proposed Algorithm, the Encryption Process, and the decryption Process.

### 3.2.1 The Encryption key (Ks)

The File slicing technique for the message to be stored or transmitted as well as integrity security principle concept are adopted to provide better security by making the length of secret encryption key long enough and more difficult to be compromised.

The proposed algorithm is designed such that the secret encryption key  $K_s$  for the proposed system comprises of four elements, namely the initial vector key which is the usual 128 bits required for the AES algorithm, referred to in this thesis as IV the key seed which, in the proposed algorithm can vary from 1 to 9999 (i.e. four decimal digits), the selected AES Key length choice (which is one of three choices; 128-bit, 192-bit, and 256-bit length),

Finally, the chosen message segment length for the encryption session  $L_m$  (which can take any value from one character up to the complete message length). These key elements of the secret encryption key are concatenated as illustrated in Fig 3.1. Hence, the secret encryption key (Ks) has the following length:

128bit +4des.digits+3choices+message segment length

So the engryption key space will equal to:

$$2^{128} * 10^4 * 3 * L_M$$

Where  $L_M$  is the selected message segment length.

The secret encryption key ( $K_S$ ) consists of four parts :

- IV: initial vector of length 128bits.
- Key seeds: 4 digits.
- AES key length choice (3 types).
- selected message segment length  $L_M$

Where  $L_M = M$  characters

$$= 8M \text{ bits (each character is 8bit long)}$$

Hence the key space of  $K_S$  is:

$$\begin{aligned} \text{Key space} &= 2^{128} * 10^4 * 3 * 2^8 * M \\ &= 2^{128} * (2^{10} * 10) * 3 * 2^8 * M \end{aligned}$$

- now if 10 is approximated to  $2^8$  and 3 is taken as 2 , then the key space can be treated as :

$$\begin{aligned} \text{Key space} &> 2^{128} * 2^{10} * 2^3 * 2 * 2^8 * M \\ &> 2^{150} * M \end{aligned}$$

Then, we can say:

$$\text{Key space} > 2^{150+}$$

The initial vector secret key, 128 bits (key space = $2^{128}$ )	The key seed, 4 decimal digits (key space > $2^{13}$ )	AES key length choice (key space = 2)	Message segment length. (key space = $M!$ ) M: message length (8 M bits)
------------------------------------------------------------------------	--------------------------------------------------------------	---------------------------------------------	-----------------------------------------------------------------------------------

Fig. 3.1. The elements of the secret encryption key ( $K_S$ ) for the proposed algorithm

The secret encryption key,  $K_s$  for each session is exchanged between communicating parties using RSA algorithm or any available key distribution technique.

### **3.2.2 Encryption process:**

The block diagram for the encryption process of the proposed algorithm is illustrated in Fig 3.2. The inputs to the algorithm are the message or file  $\mathbf{M}$  of any length (characters), the key seed  $\mathbf{K}$  which can have any numeric value from 1 to 9999 (i.e. four decimal digits), and the initial vector (IV), which is a secret key element that consists of 128 bits. Also AES key length selected for the encryption session, which is a choice of one of the three available key lengths; 128-bit, 192-bit, and 256-bit. The encryption process can be summarized by the following steps:

#### **Step 1: file slicing:**

The file  $\mathbf{M}$  is sliced (segmented into **blocks** ( $M_1, M_2 \dots M_n$ ) of equal size.

(The length of  $M_i$  is left to the communicating parties for extra security as part of the key choice).

#### **Step 2: Key generation:**

The key seed  $\mathbf{K}$  is used to generate sequence of randomly generated sub keys as will be described later in this chapter ( $K_1, K_2, \dots, K_n$ ) using the PRNG.

#### **Step 3: Ciphering the slices:**

These keys  $\mathbf{K}_i$  are used to encrypt the file slices  $\mathbf{M}_i$  producing cipher blocks or (encryption file slices)  $\mathbf{C}_i$  by using in the application of this thesis, AES cryptosystem is selected because its trusted and secure for the time being and possibly for so many years to come. symmetric encryption algorithm  $\mathbf{E}$ , using equation 3.1.

**Step 4: Merging ciphertext:**

The ciphered file slices are merged together in order to produce the complete file cipher text file C, which is to be sent for storage in the cloud storage or to the possible recipient. The message encryption process is illustrated in the block diagram of Fig 3.2, and the flow chart or the road map is summarized in Fig 3.3

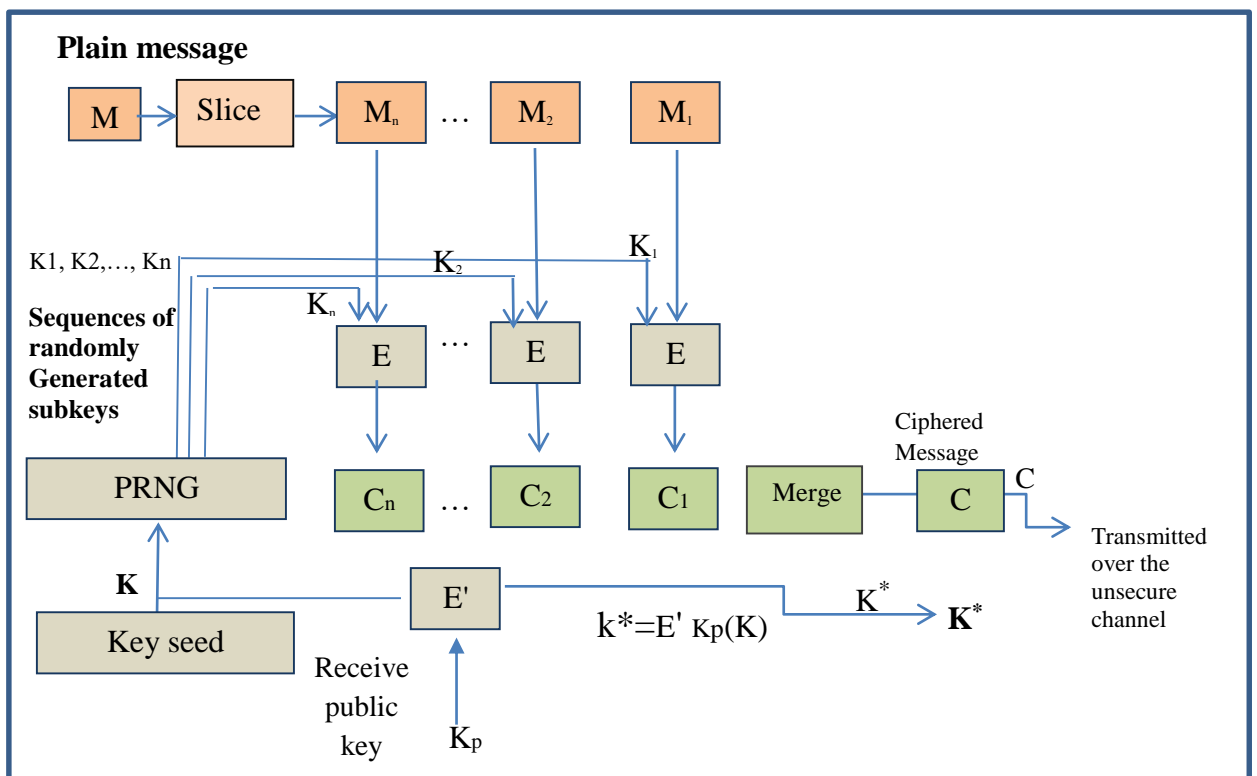


Fig 3.2. Block diagram for the Encryption process

Mathematically, the encryption and decryption processes are explained as follows:

The encryption process of each slice  $C_i$  is done using equation (3.1).

$$C_i = E_{K_i}(M_i) \dots\dots\dots (3.1)$$

Where  $E$  is the symmetric encryption technique, and  $K_i$  is the  $i$ th generated sub- key by the PRNG.

It is worth mentioning that **E** can be any symmetric cryptographic algorithm, however, In this thesis, AES system is used for its high security, efficiency, and strength.

In order to send the secret encryption key **Ks** to the receiver privately, an asymmetric system is used such as RSA for example, and will be achieved by the following process.

### 3.2.3 Secret Key seed encryption:

The secret key **Ks** is encrypted using the RSA of the receiver **Kp** for an asymmetric system. The encrypted secret key **Ks\*** is computed using

Equation (3.2)

$$\mathbf{Ks}^* = \mathbf{E}'_{\mathbf{Kp}} (\mathbf{Ks}) \dots \dots \dots (3.2)$$

Where **Kp** is the public key of the intended receiver, and **E'** is the asymmetric Encryption function of the asymmetric algorithm used. For this thesis, the RSA public key system is used.

Then all, encrypted slices (**C1, C2,..... Cn**) are merged together to get the complete message cipher text **C**,

Finally, **C** and **Ks\*** are send (concatenated) to be used and / or stored to be used at the receiver side, where **Ks\*** will be used for the decryption of **C** in order to recover **M**.

The processes of segmenting (or slicing) the message, encrypting, and merging are illustrated in the block diagram of Fig (3.3).

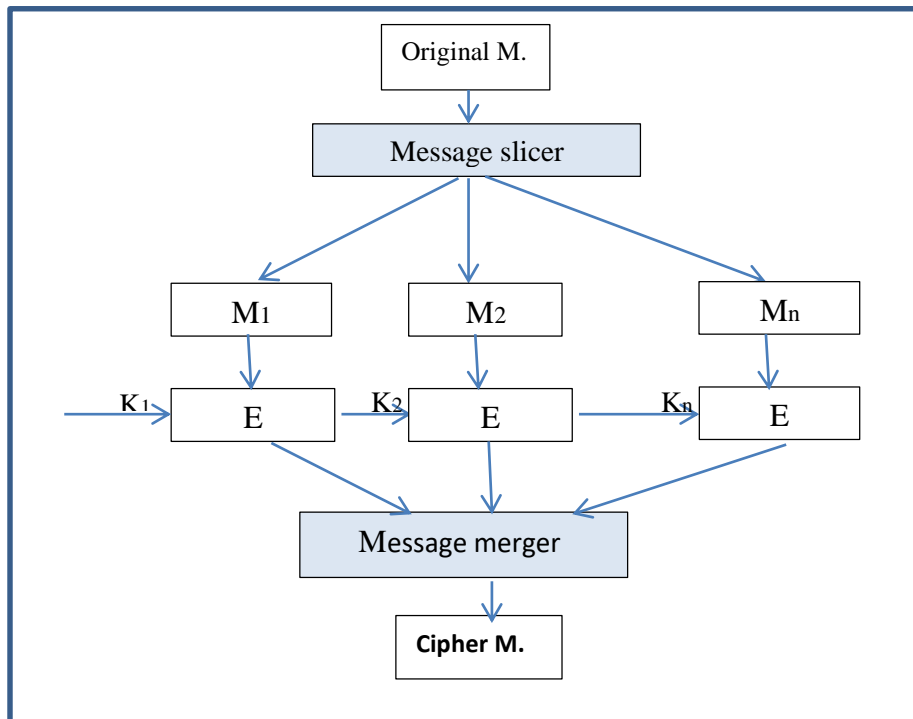


Fig 3.3 Message Splitting and Merging Mechanism

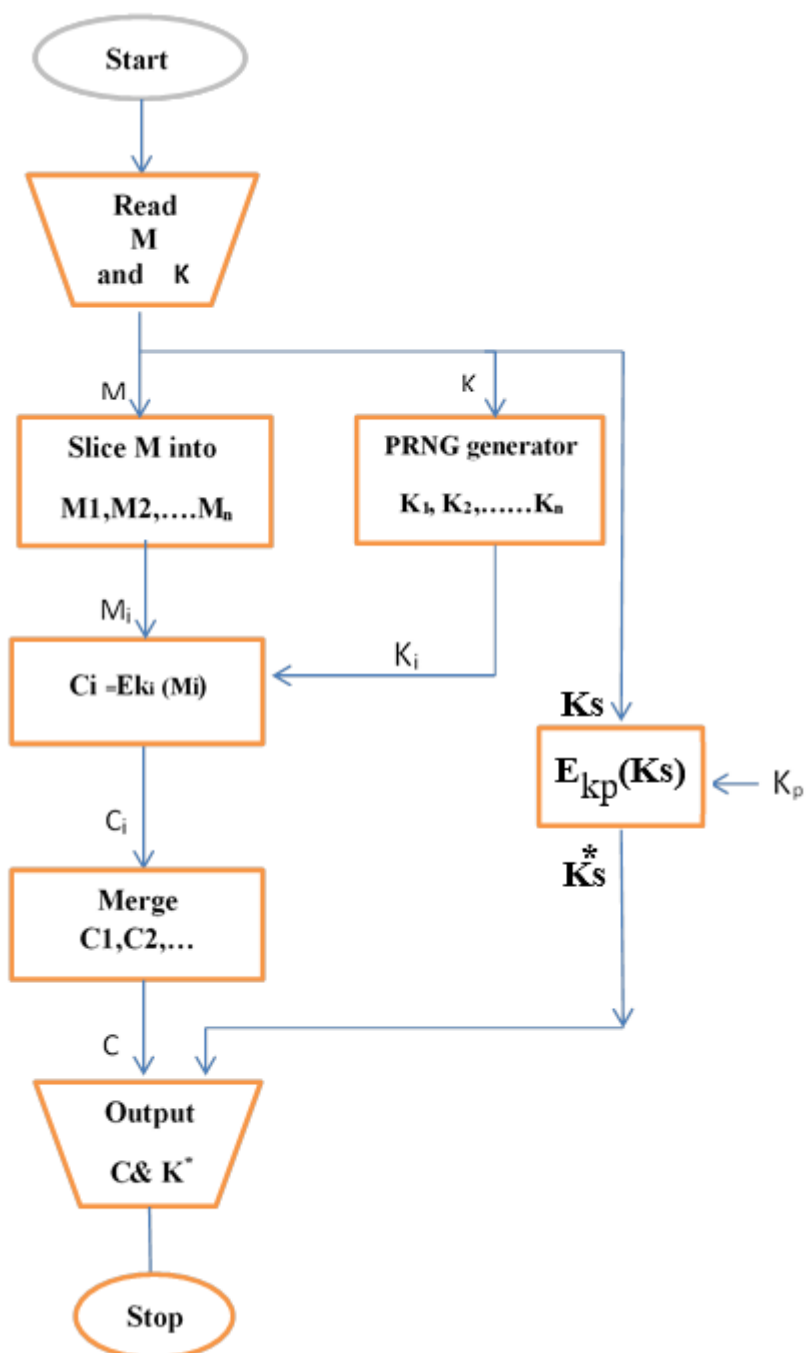


Fig 3.4. Flowchart of the encryption process

### 3.2.4 Decryption process

The decryption process is exactly the inverse of the encryption process. It is performed by the same owner on the retrieval by the intended receiver who has the private key  $K_r$  in order to recover the secret encryption key  $K_s$  first and then extract the key seed  $K$ .

The key seed  $K$  part is used to generate the encryption sub keys  $K_1, K_2, \dots, K_n$  in order to recover the original file.

The decryption process can be summarized by the following steps and illustrated in Fig.3.5. The received cipher message and the encrypted secret key are  $C$  and  $K_s^*$  respectively.

**Step 1: Recover the key seed:**

Now from  $K_s^*$ , the secret key can be recovered as follows:

$$D'_{K_r}(K_s^*) \longrightarrow K_s \dots \dots \dots (3.3)$$

Where  $D'$  is the asymmetric decryption algorithm used and  $K_r$  is the RSA of the receiver.

**Step 2: decryption sub- keys generation:**

The recovered key seed  $K$  is fed to the pseudo random number generator (PRNG) in order to generate the same sub - keys sequence,  $K_1, K_2, \dots, K_n$ , that were used at the encryption process.

**Step 3: Slicing:**

The received ciphered message  $C$  is sliced into  $C_1, C_2, \dots, C_n$ .

**Step 4: Decryption:**

A decryption algorithm  $D$ , that is the inverse of  $E$ , is used to recover  $M_1, M_2, \dots, M_n$ , as given by equation (3.4),

$$D_{k_i}(C_i) \longrightarrow M_i \dots \dots \dots (3- 4)$$



Now the recovered message slices are merged to gether to produce the original message  $M$  as shown in the block diagram of Fig (3.4).

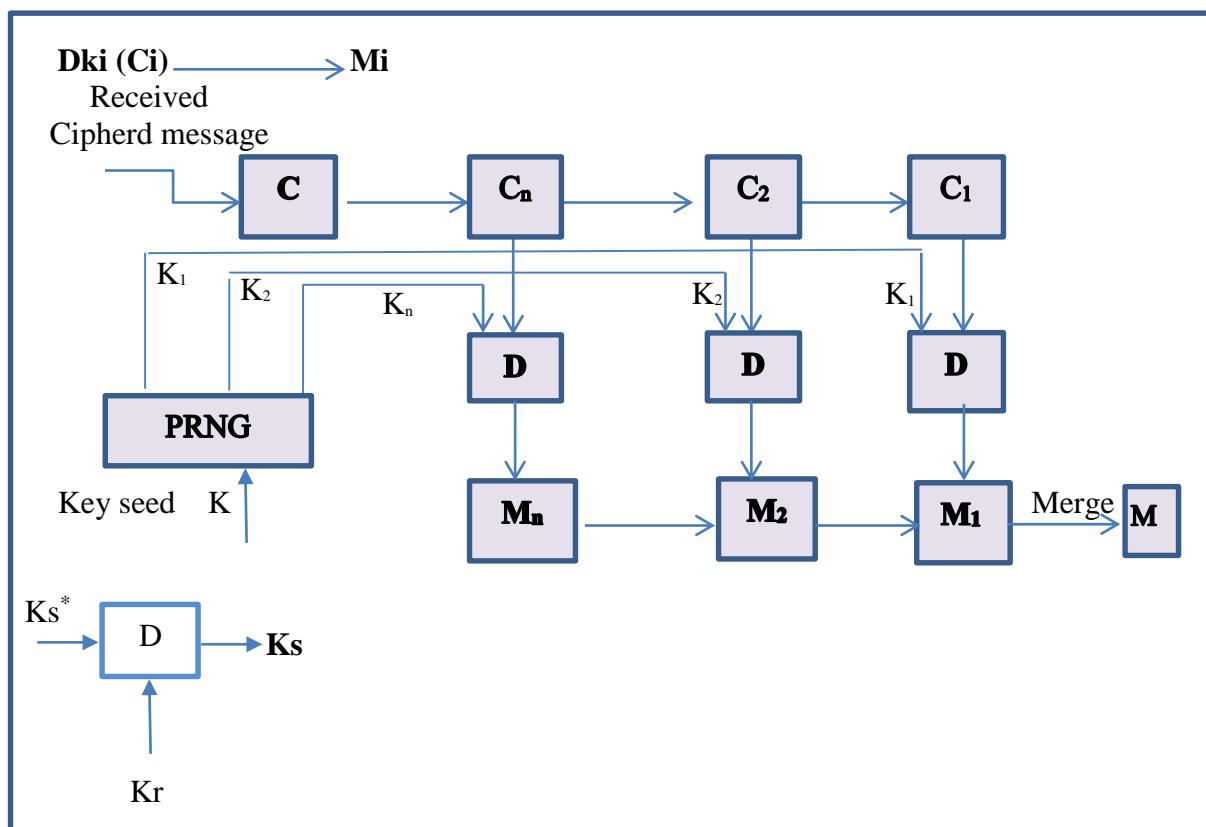


Fig.3.5. Block diagram of the proposed decryption Process

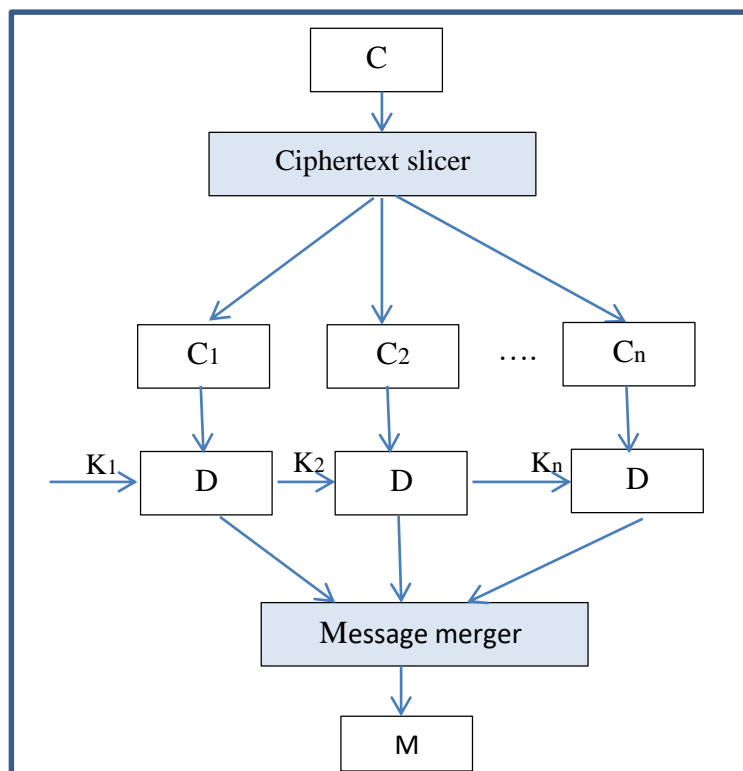


Fig 3.6 Cipher text splitting and merging

### 3.3 Pseudo-Random Number Generator (PRNG)

As PRNG is used in the proposed algorithm, it must be noted here that in this thesis an existing PRNG of the C# language is customized and modified to be used for the needed sub-keys generation. The pseudo-random number generator (PRNG) is an algorithm that accept a key seed  $K$  of certain length  $d$  bits and produce a sequence of sub-keys  $K_1, K_2, \dots$  of the required length. Random number generators have applications in statistical sampling, computer simulation, cryptography, completely randomized design, and generally, such as in security applications, hardware generators are preferred where ever feasible.

In general, Random numbers can be generated by two methods; Pure random number generator (RNG) and pseudo-random number generator (PRNG). They are described below.

- **Random number generators (RNG):** A random number generator is a natural process that produces pure random numbers. Any random process can be used to generate random numbers. Truly random numbers (RNs) or keys are changing and can be generated by measuring random physical or natural event such as radioactive decay, or atmospheric noise, and then the output is sampled to get a series of bits which are statistically independent or random, and then feeding that seed data to a computer program which generates a random value.
- **Pseudo-Random number generators (PRNG):**  
Various applications of randomness have led to the development of several different methods for generating random data, Because of the mechanical nature of these techniques, generating large numbers of sufficiently random numbers, several computational methods for random-number generation exist. PRNG is also known as a deterministic random bit generator (DRBG) is an algorithm for generating a sequence of numbers whose properties approximate of sequences of random numbers.

The PRNG-generated sequence is not truly random, because it is completely determined by an initial value, called the PRNG's seed (which may include truly random values). They are very useful in developing a debugging work facilitated by the ability to run the same sequence of random numbers again by starting with the same random seed. They are used in cryptography as long as the key seed is secret. Sender and receiver can generate the same set of keys automatically to be used as keys.

## **Chapter Four**

### **Evaluation and Experimental Results**

## Chapter Four

### Evaluation and Experimental Results

#### 4.1 Introduction:

This chapter includes the implementation and examination of the proposed segmented outlined in chapter 3. The examination included encryption and verification execution time calculated for different message lengths and contents. For the coding and testing purpose, C# language has been utilized in this study. Finally, the execution speed for the AES algorithms are calculated and compared with that for the proposed segmented method for different segment lengths, for different secret keys and for different key seeds. This chapter includes the results obtained for three different key lengths, namely; 128bit, 192bit, 256bit.

#### 4.2. Resources used for the Implementation:

To perform the processes and implementation we use C# language and personal computer laptop with the following features:

Operating system: Windows 10, 64-bit

.Memory (RAM): 4.00 GB

Process: core i3 -6006U, 2.00 GHz

### 4.3. The program main Screens:

In order to apply and investigate the proposed segmented AES algorithm, the program is coded in C# language and it equipped with the main screen. This computer interface consists of five parts and defined in the following:

**part 1:** is the text encryption section that produce the cipher text.

**Part 2:** is the decryption section to recovers the original plaintext.

**Part 3:** This section is included to show the measured encryption execution time for the message to be sent to the receiving side.

**Part 4:** This section is included to show the decryption execution time of the received cipher text.

**Part 5:** This section is included to show the segmented cipher text sub-blocks together with the corresponding generated encryption sub-keys as the encryption process is performed.

#### 4.3.1 Encryption process screen

The most important parts are the encryption part (section 1) and the decryption part (section 2). They will be explained in more details and with be used to show the intended process in the following sections.

The encryption process screen It consists of the following fields :

**Initialize vector (IV)** must be 16 characters (numbers, characters, special characters), which will be considered as part of the secret encryption key.

**PRNG Seed:** A number in the rang 0-9999 can be selected as the seed for generating the sub-keys for the message segments using the PRNG used in the algorithm.

This seed will be considered as the second part of the secret key where the first part is the IV value.

**Segments sub-keys:** three types of sub-keys lengths can be used, namely 128-bits, 192-bits, or 256-bits. This is achieved in the program by selecting 16, 24, or 32 options through the field shown next to the PRNG seed field, (note: these numbers represent the lengths of the encryption sub-keys in bytes).

**NO PRNG:** An option field is also added in order to give the possibility of running the algorithm using only one encryption key for all the message segments.

To show the implementation for encryption process, an example is shown in the following:

- (1) The message to be encrypted is placed in the box “Text to be Encrypted”.
- (2) An encryption key of 16 bytes (i.e. 128 bits) is placed in the “initialize vector” field.
- (3) A seed is placed in the field “PRNG Seed” of up to four decimal digit.
- (4) The selected segment length is placed in the field “Length”.
- (5) Now pressing “Generate by length” will run the program to encipher the supplied message by the segment algorithm and place the resulting encrypted text segments together with the corresponding sub-keys that were generated by the PRNG.

for example, we write a text of 100 character and select the field “Generate by the length”, using 20 characters the result for encryption is 5 block, each of 20 characters, as shown in Fig 4.1. The encryption key is shown on the figure which consists of few parts (namely; IV. seed. Sub-keys length. Segment length).





The screenshot displays a software interface for decryption. On the left, the 'Decryption' section includes an 'Enter Seed for PRNG' field with '9999' and a dropdown set to '16'. Below it is an 'IV' field containing '1234abcdef1234ss' and a 'Generate Password with PRNG' button. A text area shows a long alphanumeric string: '4569598471079777', '9793851612971123', '5018104754862427', '0242357896753754', and '5466611038645086'. A 'Decrypt (on the other side)' button is positioned to the right. Below this is a text box containing the message: 'Information Security plays a major role in today's scenario. Cloud computing is a technology that'. A small '5' is centered below the text box.

On the right, the 'Time Calculations' panel is divided into two sections. The top section, 'Encryption Time / Encrypted String', lists five time values: '00:00:00.0005490', '00:00:00.0003315', '00:00:00.0002981', '00:00:00.0003002', and '00:00:00.0002806'. A 'Calculate' button is to the right, and a list of corresponding values is shown: '0005490', '0003315', '0002981', '0003002', and '0002806'. An 'Average=' field shows the value '3518.8'.

The bottom section, 'Decryption Time / Decrypted String', lists five time values: '00:00:00.0004595', '00:00:00.0003084', '00:00:00.0003475', '00:00:00.0002837', and '00:00:00.0002996'. A 'Calculate' button is to the right, and a list of corresponding values is shown: '0004595', '0003084', '0003475', '0002837', and '0002996'. An 'Average=' field shows the value '3397.4'.

Fig (4.2) Example for the decryption process

As in the encryption process, we put the same secret key which includes all different characters, that we have used in the encryption process for the field.

### 4.3.3 Encryption without PRNG

An example for encryption and decryption by using “NO PRNG” field is shown in Fig (4.5).

The message blocks to be enciphered should be of certain length (namely 128 bits), however, if the message segment is shorter than 128 bits, the AES algorithm adds padding “\$” in order to complete it to the required length.

The results of encrypting a text with password in this example appears as shown in Fig 4.3.

To isolate ciphered blocks from each other, they are enclosed by the special characters “<” and “>”, as shown in the lower part of main screen shown in Fig 4.3.

It must be noted that all message segments will be encrypted using the same secret key as no Pseudo random numbers are generated.

The screenshot displays a web application for encryption and decryption. The main interface is titled "Encrypt and Decrypt" and includes a "Batch Encryption and Decryption" tab. On the left, the "Encrypt" section shows an "Initialize Vector (IV)" field with the value "aaaaagghh123555" and a length of "16" (128 bit). Below this is a "Provide PASSWORD" field with "ssssssshh7777993" and a "NO PRNG" checkbox checked. A text area contains a sample paragraph about information security. There are buttons for "Generate By Words" and "Generate By Length" (set to 30). An "Encrypt -->" button is also present. On the right, the "Decryption" section has an "Enter Seed for PRNG" field set to "16" and a "Provide PASSWORD" field with "ssssssshh7777993". A "Decrypt (on the other side)" button is located below. To the right of the main interface is a "Time Calculations" panel. It contains two sections: "Encryption Time / Encrypted String" and "Decryption Time / Decrypted String". Each section has a list of time values and a "Calculate" button. The encryption average is 2776.6 and the decryption average is 24669.7. At the bottom, there are "Results" and "Password Text" sections. The "Results" section shows "Ciphered Text" as a long string of characters enclosed in angle brackets. The "Password Text" section shows a long string of characters. A "Clear All Fields" button is located at the bottom right of the main interface.

Fig (4.3) Example for encryption and decryption using NO PRNG process.

## 4.4 Results

In this section , the implementation and observed results of the proposed algorithm are listed. It includes the calculated encryption and decryption processing time for various component variations, such as the use of different message segmentation length , key seed values , encryption key lengths , and the comparison of the proposed algorithm with the traditional AES algorithm (i.e.with no message segmentation) processing time.

### 4.4.1 Processing time versus message segment length

For the investigation of the proposed segmented message algorithm, the encryption and decryption processing time are calculated for different message segment lengths using the following input data:

- Message length = 10000 character, i.e. 8000 bits.
- Secret key that consists of three parts, namely
  - 1- Fixed IV part of 128 bits
  - 2- Fixed PRNG seed of four decimal digits (e.g. used here 0139).
  - 3- Sub-keys generated by the PRNG, (where three encryption key length options are available, namely 128-bit, 192-bit, 256-bit).

The message segments length selected was 10, 50, 100, 500, and 1000 characters.

The results are listed in Table4.1. Table4.2. and Table 4.3. for the encryption sub-keys 128-bit, 192-bit, 256-bit, respectively.

Table 4.1. Encryption and Decryption processing time for 128-bit encryption key

<b>Segment length</b> <b>Chr.</b>	<b>Encryption time</b> <b>(Microsecond)</b>	<b>Decryption time</b> <b>(Microsecond)</b>
10	3722.03	3434.97
50	3949.54	2943.895
100	3592.33	2761.32
500	3683.95	3001.8
1000	3503.9	2997.9

It can be notice from Table 4.1 that if the length of the segment is small there is a big difference between Encryption and Decryption processing time but as the length of the segment approaches 100 characters and higher the encoding time is still higher than decoding time but with comparatively small difference. This observed difference can be attributed to the need for padding when the message segments lengths are less than 128-bits.

Table4.2. Encryption and Decryption processing time for 192-bit encryption key

<b>Segment length</b> <b>Chr.</b>	<b>Encryption time</b> <b>(Microsecond)</b>	<b>Decryption time</b> <b>(Microsecond)</b>
10	3781.16	2748.10
50	3715.78	2808.29
100	3725.68	2813.04
500	3573.55	3203.5
1000	3933.1	2996.3

**Table.4.3.** Encryption and Decryption processing time for 256-bit encryption key

<b>Segment length Chr.</b>	<b>Encryption time (Microsecond)</b>	<b>Decryption time (Microsecond)</b>
10	3758.81	2746.75
50	3700.88	2746.75
100	3658.33	2928.74
500	3633.6	2995.05
1000	3793.6	3058.1

It is also noticed that as the segment length increases, processing time for the same message decreases, which is expected as the number of segments decreases. It is also noticed from Table4.2. and Table4.3. that the encryption time is higher than the decryption time with all lengths ranging from small to large lengths when using a 24bytes key and a 32-bytes encryption keys, that is 192-bits and 256-bits, respectively.

#### **4.4.2 Processing time versus key seeds**

The encryption and decryption processing time for the proposed algorithm is calculated for different seed values, using three sub-key lengths 128-bits, 192-bits and 256-bits for the segment's encryption.

This test was done using the same secret key (i.e. IV) and message length of (10000 character) but for different key seeds. Also, the calculations are done for the three sub-keys lengths, 128-bits, 192-bits, 256-bits. The experiment is repeated for many selected segment lengths, but only two cases are listed as examples, namely, 50 characters and 1000 characters. The obtained experimental results for the 50 characters segment length is listed in Table 4.4. (The secret key used in these table is fixed and has the value IV=

suhair7539hhhhhh). Due to the difficulty of calculating the time complexity, (or the time for execution of any program on its own), the average calculated time is considered after many runs. Moreover, the first few runs are ignored, as they always give some odd values.

Table4.4. Encryption and Decryption Processing times for various seeds  
(50 characters segment length)

Seed	Encryption processing time (Microsecond)			Decryption processing time (Microsecond)		
	Key length used (bits)			Key length used (bits)		
	128	192	256	128	192	256
25	3964.54	3822.85	3848.12	3057.81	2918.33	2976.35
77	3832.59	3781.27	3764.1	2891.25	2843.985	5264.77
159	3906.32	3858.98	3780.29	2853.29	2866.5	2875.78
578	3754.3	3851.39	3851.39	2959.58	2876.4	2876.4
9876	3806.14	3779.29	3754.3	2963.9	2960.58	2895.54
Average	3852.778	3818.756	3799.64	2945.166	2893.159	3377.768

It is noticed from Table4.4. that the average encryption time for different secret key lengths are very close to each other regardless of the different key seeds used, Besides, it is noticed, too that the differences for the different encryption key lengths (i.e.128 bits, 192 bits, and 256 bits) were small.

The same process used for getting Table4.4. as well as using the same message, the same secret key, the same seeds, and the same encryption sub-keys are repeated for long message segment (i.e. 1000 character) using the same parameters and message used there. The results are listed in Table 4.5.

Table4.5. Encryption and Decryption Processing times for various seeds  
(1000 characters segment length)

Seed	Encryption processing time (Microsecond)			Decryption processing time (Microsecond)		
	Key length used (bits)			Key length used (bits)		
	128	192	256	128	192	256
25	3649.8	3462.9	3521.4	3399.7	3223.5	3356.9
77	3319.6	3593.2	3426	4258.6	3190.1	3612.5
159	3314.5	3366.8	3447.9	3126	3189	3296
578	3429.3	3261	3519.2	3175.2	3356.4	3303.6
9876	3396.3	3407.3	3459.9	3202.2	3256.9	3313.6
Average	3421.9	3418.24	3474.88	3432.34	3243.18	3376.52

In Table 4.5. it can be seen that the encryption processing time is fluctuating within few percentages around the average, but no relation can be deducted from these observations. Hence it can be concluded that the internal running of the cpu and the random number generation dictated this fluctuating effect.

In both, Table4.4 and 4.5. it is noticed that the average processing time for the encryption is a bit higher than the decryption time. This is attributed to the time required for the generation of the encryption sub-keys by the PRNG. Hence the fluctuation in the measured time can be explained as due to the internal time complexity feature and hidden central processing unit activities.

As an example, one screen shot for one of the results of Table 4.5. It shows all the field were filled with either plaintext (input data), used secret key, selected seed, the message segments length, the ciphertext, the chosen encryption sub-key length, and the calculated processing time.

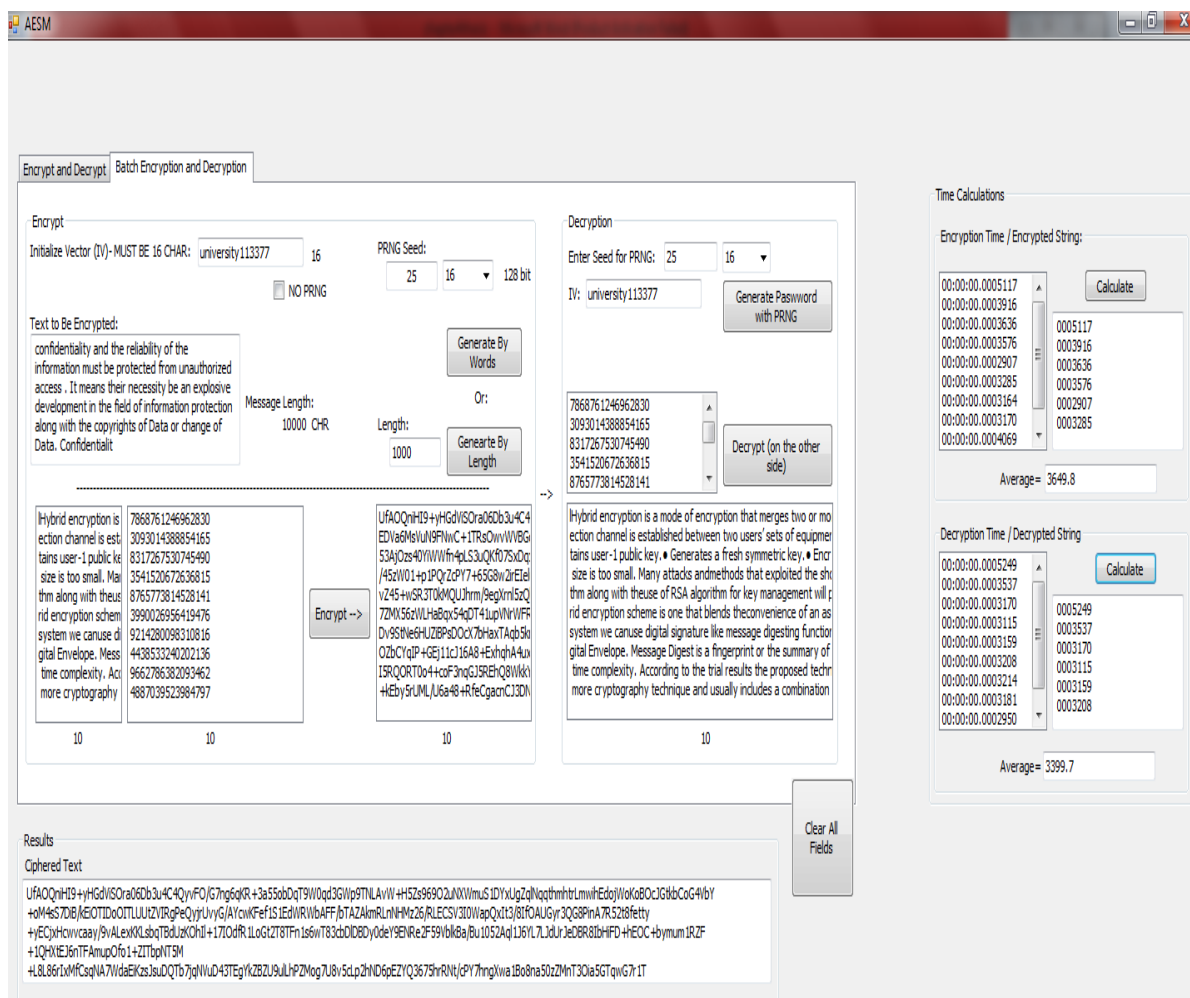


Fig 4.4. An example screen shot for Encryption and Decryption processes of Table4.5

#### 4.4.3 Processing time versus secret keys.

The encryption and decryption processing time are calculated for various input secret keys that comprises of either numeral only, alphabets only, and mixed alphanumeric characters. The calculations are done for the three encryption sub-key lengths under consideration, ( i.e. 128-bit, 192-bit, and 256-bit), and the experiment is performed twice; one for message segment length of 50 characters and the other for 1000 characters. The results are listed in Table4.6 and Table4.7. A message of 10000 characters is used as input and the seed value is 1379 for both cases.



**Table 4.6.** Processing times for various secret keys (50 characters segment length)

IV	Encryption processing time (Microsecond)			Decryption processing time (Microsecond)		
	Key length used (bits)			Key length used (bits)		
	128	192	256	128	192	256
9754325721498764	3775.44	3740.31	3626.44	2929.63	2787.77	2769.10
AAaahHmMmmsSsgGL	3330.13	3428.90	3376.18	2328.43	2431.42	2435.13
AAaahss3777799SS	3998.66	3380.07	3422.46	2394.96	2394.96	2394.96
Average	3701.41	3516.42	3475.02	2551.006	2538.05	2533.06

**Table 4.7.** Processing times for various secret keys (1000 characters segment length)

IV	Encryption processing time (Microsecond)			Decryption processing time (Microsecond)		
	Key length used			Key length used		
	128-bit	192-bit	256-bit	128-bit	192-bit	256-bit
AAaahss3777799SS	3281.1	3305.2	3473	3292.4	4093	3354.7
AAaahHmMmmsSsgGL	3672.3	3947	3495	3221.4	3182.4	3231.3
9754325721498764	3871.9	3879.1	3742.9	2963	3344	3635.5
Average	3608.43	3710.43	3570.3	3158.93	3539.8	3407.16

It is noticed that there is only slight difference in time for different key types, and on the average, the calculated time for 128-bit key length is the lowest while that for the 256-bit is the highest. It is also noticed from both Tables 4.6 and 4.7 that the time of the encryption for the three sub-key lengths are the higher than decoding execution.

#### 4.4.4 Processing time versus message length.

Another experimentation is conducted to illustrate the differences in processing time as related to the message size. For a fixed key seed (value 3355) and a selected secret key (hhhssmmm3377999), different message lengths were encrypted and the processing time calculations were conducted for the three encryption sub-keys lengths under consideration, the results are listed in Table 4.8. and plotted in Fig 4.7.

**Table 4. 8.** Encryption and Decryption processing times versus message lengths

Message lengths (char.)	Encryption processing time (Microsecond)			Decryption processing time (Microsecond)		
	Key length used (bits)			Key length used (bits)		
	128	192	256	128	192	256
100	3078.46	3004.53	2917.86	2796.66	2852	2772.6
300	3254.4	3207.2	3006.6	3021.8	3159.8	2907.8
500	3601.33	3285	3440.33	3767.66	3347	3343.66
700	3608.66	3418.33	3725.6	3325.33	3288.66	3347
900	3733	3444	3552	3738	3486	3557.33
1000	3480.3	3542.3	3634.8	3208.2	3216.4	3380.3
5000	4856.5	4988	5306	6066	6131.5	6573
10000	6911	6850	7728	9126	9280	9648

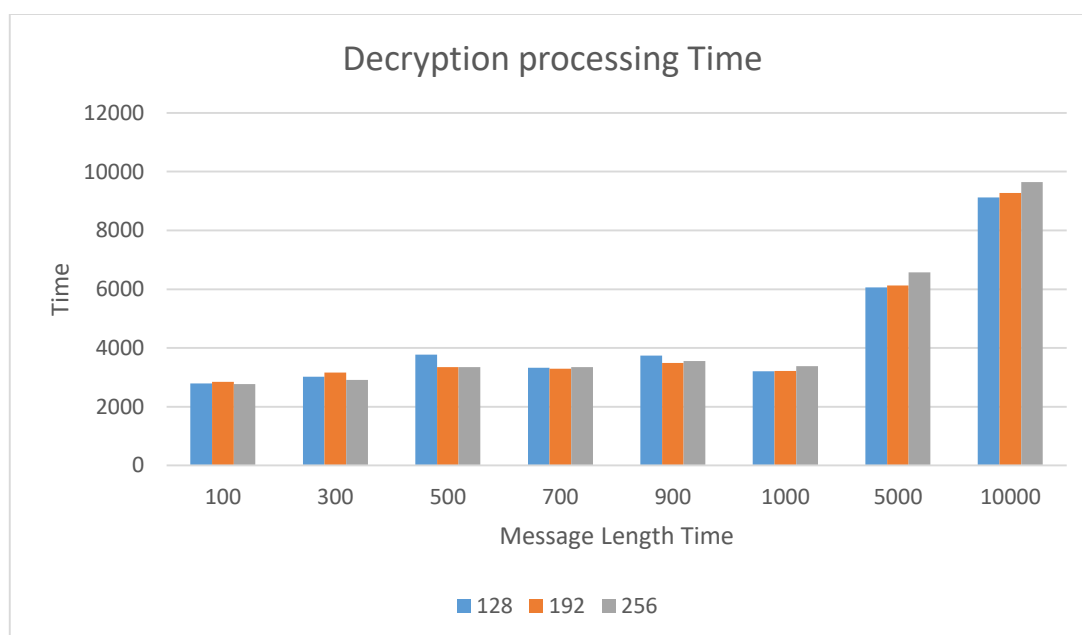
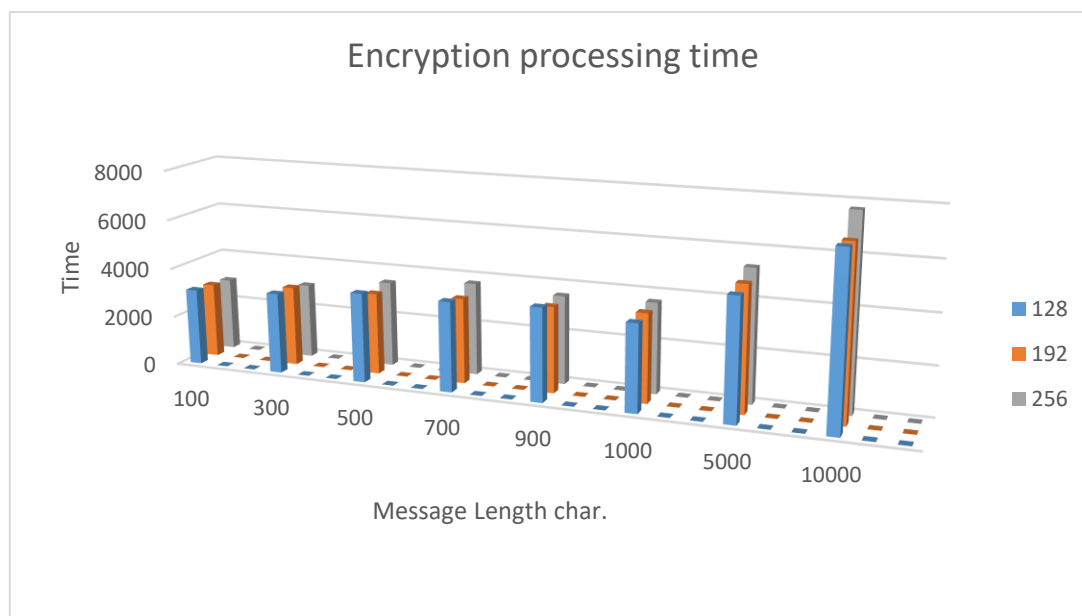


Fig 4.5 results listed in Table 4.8

It is noticed from Table 4.8 and Fig 4.7 how the processing time increases as the message size increases. It can be deduced that the internal CPU time complexity affect the actual processing time. That is why time measurements for all cases are approximate and it is always better to refer to the average time rather than the individual processing time.

#### 4.4.5 Comparison with traditional AES algorithm:

The encryption and decryption processing time for the proposed segmented algorithm are calculated for different segmentation lengths and compared with those for the traditional AES algorithm using the following data and the same computing environment for both algorithms.

- Message length = 10000 character (i.e. 8000 bits)
- Secret key consists of:
  1. Fixed IV part of 128 bits.
  2. Fixed seed of four decimal digits (used 1235).
  3. Variable message segment length (e.g. 25,35,75,125,250,500, 1000 characters).
  4. Sub- keys of three options (namely; 128-bits, 192 bits, and 256 bits).

The results are listed in three tables, each for one length for encryption sub-keys; Table4.9 for 128-bit, Table4.10 for 192-bit, and Table4.11 for 256-bit encryption sub-keys length.

**Table 4.9** Processing times Comparison for 128-bit encryption sub-key length.

Segment length (number of chr.)	Encryption time Microsecond		Decryption time Microsecond	
	Original AES Algorithm	Proposed Algorithm	Original AES Algorithm	Proposed Algorithm
25	7906	8819	9957	8096
35		8043		9591
75		6928		9679
125		8313		7175
250		8811		7761
500		8214		8069
1000		8777		8412
Average		8272.14		8397.57

**Table 4.10.** Processing times Comparison for 192-bit encryption sub-key length.

Segment length (number of chr.)	Encryption time (Microsecond)		Decryption time (Microsecond)	
	Original AES Algorithm	Proposed Algorithm	Original AES Algorithm	Proposed Algorithm
25	8001	7609	8073	8834
35		8537		7426
75		8225		7073
125		8119		9918
250		9081		8518
500		9104		7868
1000		9393		8853
Average				8581.143

**Table 4. 11.** Processing times Comparison for 256-bit encryption sub-key length

. Segment length (number of chr.)	Encryption time ( Microsecond)		Decryption time (Microsecond)	
	Original AES Algorithm	Proposed Algorithm	Original AES Algorithm	Proposed Algorithm
25	7423	6802	8754	7008
35		6586		7160
75		6951		8674
125		8149		6943
250		7164		7419
500		9024		8309
1000		6487		6856
Average				7309

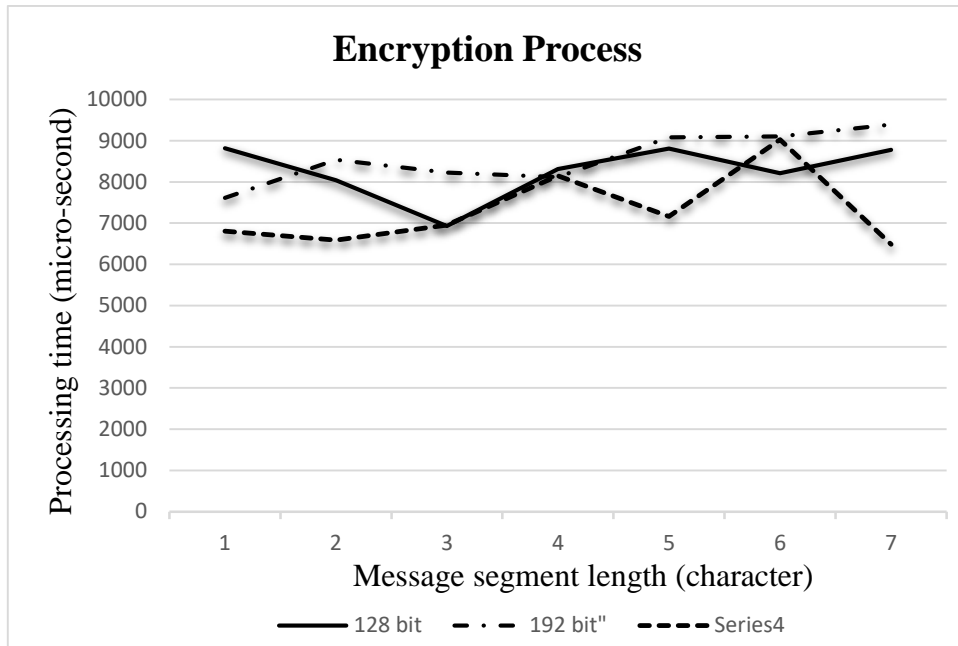
From the results listed in Tables 9-11, a combined graph can be drawn as shown

An example comparison of the use of traditional AES algorithm and the proposed segmented message algorithm with different message lengths is listed in Table 4.13 for the case of 128-bit encryption key. It illustrates the time difference resulted when segmenting process is used.

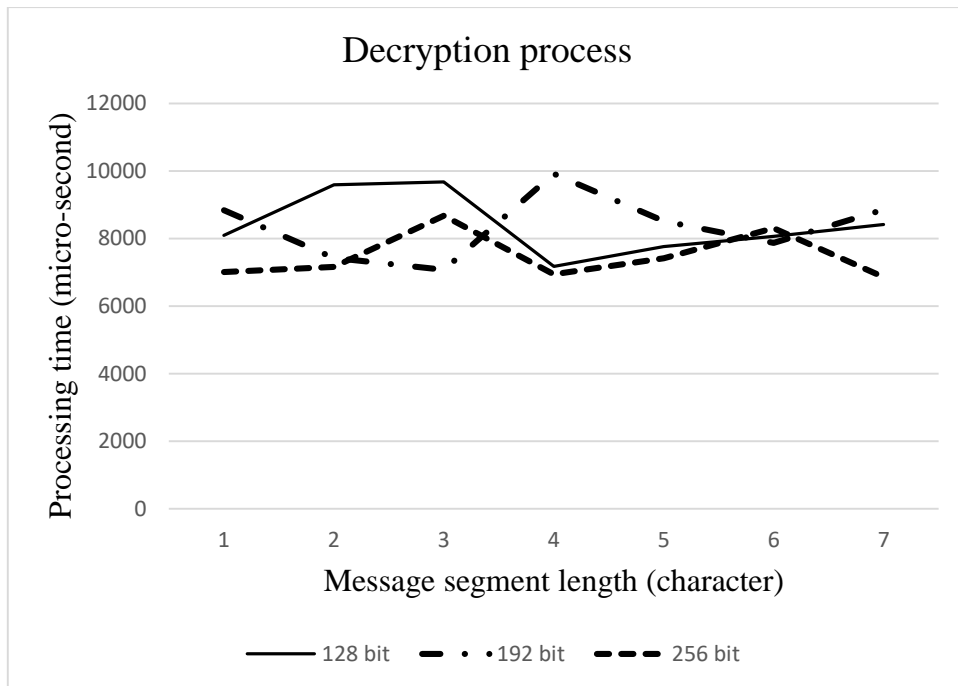
Table 4.12. Processing time for AES and proposed algo. for different message lengths.

Message Segment Lengths (Char.)	Encryption processing time (Microsecond) Key length used (bits)			Decryption processing time (Microsecond) Key length used (bits)		
	128	192	256	128	192	256
25	8819	7609	6802	8096	8834	7008
35	8043	8537	6586	9591	7426	7160
75	6928	8225	6951	9679	7073	8674
125	8313	8119	8149	7175	9918	6943
250	8811	9081	7164	7761	8518	7419
500	8214	9104	9024	8069	7868	8309
1000	8777	9393	6487	8412	8853	6856

The results displayed in Table 4.12 are plotted for the encryption and decryption processing time versus the message segment length, as shown in Fig 4.8. The processing time for the three AES key lengths close enough to each other. The fluctuation in the processing time can be attributed to the time complexity of the internal activities in the computation environment.



**Fig 4.6- (a)** Processing time for AES and proposed algo. for different message lengths for Table 4.12.



**Fig 4.8- (b)** Processing time for AES and proposed algo. for different message lengths for Table 4.12.

#### 4.4.6 Key strength comparison

To demonstrate the key enhancement achieved by the segmented message algorithm proposed in this thesis, a comparison of the key size and the key space for other cryptosystems is of great importance. Table 4.14 lists the key sizes, data block size, and key space for the most widely used cryptosystems nowadays, such as DES, 3DES, AES, Blowfish, IDEA, PGP, etc. Table 4.12. Key length and key space comparison.

Table 4.13 lists the key sizes, data block size, and key space for the most widely used cryptosystems

Algorithm	Data block size	Key size	Key space
Proposed	128 bits	150 + bits	$2^{150+}$
DES	64 bits	56 bits	$2^{56}$
3DES	64 bits	56, 112, & 168 bits	$2^{56}, 2^{112}, \& 2^{168}$
AES	128 bits	128, 192, & 256 bits	$2^{128}, 2^{192}, \& 2^{256}$
Blowfish	64 – 448	1 - 448 bits	Up to $2^{448}$
IDEA	64 bits	128 bits	$2^{128}$
PGP	64 bits	128 bits	$2^{128}$
RC2	64bits	8-1024 bits	$2^8 - 2^{1024}$
RC4	2064bits, 1684 effective	40-2048 bits	$2^{40} - 2^{2048}$
RC5	32, 64, & 128bits	0-2040 bits	$0-2^{2040}$
RC6	128bits	128, 192, 256, ..., 2040 bit	Up to $2^{2040}$
Two fish	128bits	Up to 256 bits	$2^{128} - 2^{256}$
Serpent	128bits	128,192, & 256bits	$2^{128}, 2^{192}, 2^{256}$
Nimbus	64bits	128 bits	$2^{128}$
CAST-128	64 bits	40 to 128 bits	$2^{40} - 2^{128}$



## **Chapter Five**

### **Conclusions and Future Work**

## **Chapter Five**

### **Conclusions and Future Work**

#### **5.1 Conclusions**

The purpose of this thesis is to propose an algorithm to enhance encryption and decryption processes by using AES algorithm. The proposed algorithm includes a segmented message process that is supported by a Pseudorandom number generator (PRNG) in order to encrypt each message segment with different key, as explained in chapter 3. The proposed algorithm is investigated for various parameter values and compared with AES algorithm. The achievement in the research work of this thesis can be summarized as follows.

- 1- The encryption key space for the proposed algorithm is enhanced due to increasing the encryption secret key by adding many parameters, namely message segment length, the PRNG seed, the AES key length choice (which are all varied according to user's choice).
- 2- The encryption of each message segment with different key that is generated by the PRNG will make extremely difficult for any hacker or cryptanalyst to break the security, because even if one segment is compromised, each of the other segments need equal efforts to be compromised.
- 3- For the coding and testing purpose, C# language has been utilized in this thesis. the execution Time for the AES algorithms are calculated and compared with that for the proposed segmented method for different segment lengths, for different secret keys and for different key seeds. This chapter includes the results obtained for three different key lengths, namely; 128bit, 192bit, 256bit.

- 4- AES is very strong algorithm but its secret key length is limited to 128-bit, 192-bit, and 256-bits. The proposed algorithm provides a flexible and strong key for encryption, as it consists of 4 parts: IV+ seed for PRNG + Key type + M. segment length. It includes the calculated encryption and decryption processing time for various component types, such as the use of different message segmentation length, key seed values, encryption key lengths, and the comparison of the proposed algorithm with the traditional AES algorithm (i.e. with no message segmentation) processing time.
- 5- It must be stated that both encryption and decryption processing time for the proposed segmented message algorithm were more than the traditional AES algorithm processing time, which is considered as the cost for the achieved strength in the key space.

## 5.2 Future Work

The work presented in this thesis can be further extended to be either improved or implemented in some useful applications. Here are some suggestions for future research works and implementations.

1. To increase the security, the encrypted message segments can be arranged to be transmitted in random sequence according to a certain PRNG.
2. A PRNG may be designed specifically for the proposed algorithm instead of the one used now. This PRNG may be based on the concept of neural networks, NN.
3. Study the application of the proposed segmented message encryption algorithm on a real cloud environment.
4. Use the proposed algorithm in concatenation with other cryptographic algorithms.

## References

Aarti Devi, Ankush Sharma, Anamika Rangra. (2015). A Review on DES, AES and Blowfish for Image Encryption & Decryption, Vol. 6 (3), pp3034-3036.

Adedeji Kazeem B. and Ponnle Akinlolu A. (2014). A New Hybrid Data Encryption and Decryption Technique to Enhance Data Security in Communication Networks: Algorithm Development, Vol.5, Issue 10, pp804-811.

Ali E. Taki El\_Deen. (2013). Design and Implementation of Hybrid Encryption Algorithm, Vol. 4, Issue 12, pp.669-673.

Ankit Gambhir. (2014). RSA Algorithm or DES Algorithm? Vol. 3, No.4, pp 26-29.

Anand Kumar and S. Karthikeya. (2012). Investigating the Efficiency of Blowfish and Rejindael (AES) Algorithms, pp22-28.

Ayushi. (2010). A Symmetric Key Cryptographic Algorithm, Vol.1, No. 15, pp.2-4.

Avi Kak.( 2018). Lecture 8: AES: The Advanced Encryption Standard Lecture Notes on “Computer and Network Security,pp2-92.

Diaa Salama AbdElminaam.(2018). Improving the Security of Cloud Computing by Building New Hybrid Cryptography Algorithms, Vol.8, No.1, pp.40-48.

Gary C. Kessler .(2018). An Overview of Cryptography.

Gurpreet Singh, Supriya, (2013). A Study of Encryption Algorithms (RSA, DES, 3DES and AES) for Information Security. Vol. 7, No.19, pp33-38.

Hardik Gandhi, Vinit Gupta, Indra Rajput. (2015). A Research on Enhancing Public Key Cryptography by The Use of MRGA with RSA, and N-Prime RSA, Vol , Issue 12, pp.91-98.

Puneeth M. Jasmine shafi Farha, N. Sandhya, M. Yamini. (2015). Vol.2, Issue.2, pp .15-20.

Prerna Mahajan & Abhishek Sachdeva. (2013). A Study of Encryption Algorithms AES, DES and RSA for Security ,Vol- 13 Issue 15, pp 1-9.

Ramaraj, Karthikeya and Hemalatha. (2009). A Design of Security Protocol using Hybrid Encryption Technique (AES- Rijndael and RSA), Vol. 17. No.1, pp.78-86.

Rachna Arora, Anshu Parashar. (2013). Secure User Data in Cloud Computing Using Encryption Algorithms, Vol. 3, Issue 4, pp.1922-1926 .

Raj Jain. (2011). Advanced Encryption Standard (AES).

<http://www.cse.wustl.edu/~jain/cse571-11>.

Satyajeet R. Shinge, Rahul Patil. (2014). An Encryption Algorithm Based on ASCII Value of Data, Vol. 5 (6), pp.7232-7234.

Shikha Rani, Shanky Rani. (2016). Data Security in Cloud Computing Using Various Encryption Techniques, Vol.4, Issue 3, pp.163-166.

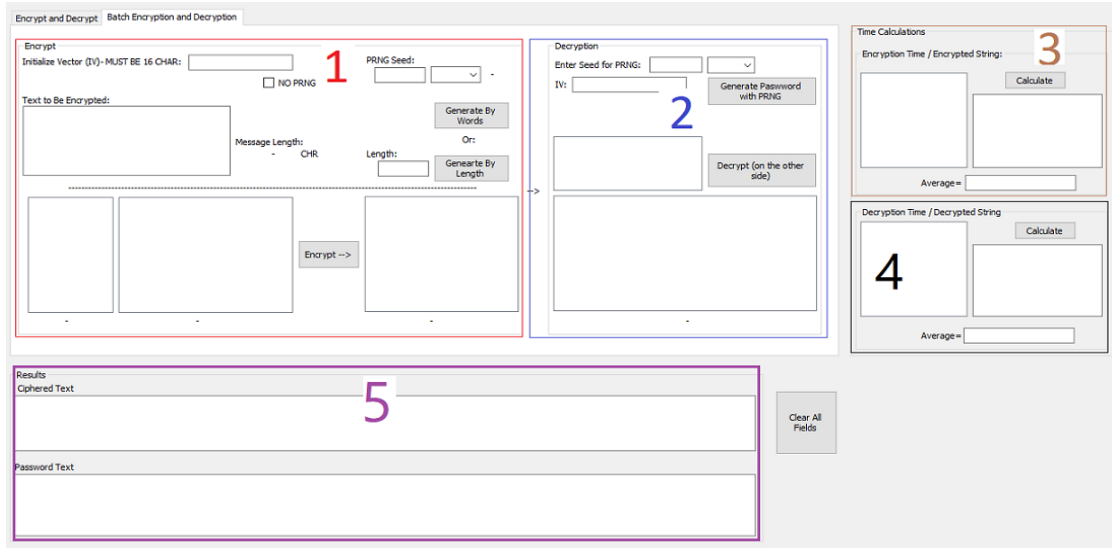
Stallings, W. (1999). Cryptography and Network Security: Principles and Practice.

Uma Naik, V. C. Kotak. (2014). Security Issues with Implementation of RSA and Proposed Dual Security Algorithm for Cloud Computing ,Vol.9, Issue.1,pp. 43-47

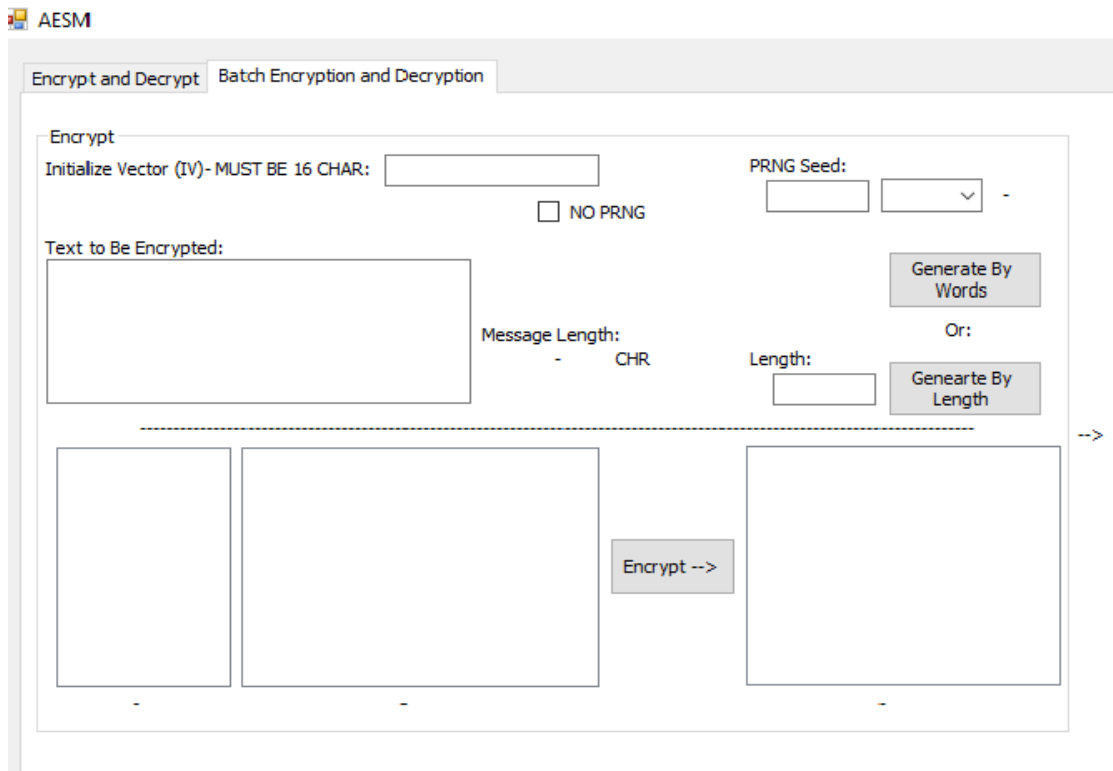
Veerpal Kaur, Aman Singh. (2013). Review of Various Algorithms Used in Hybrid, Vol. 2, Issue. 6, pp.157-173.

# Appendix A

The main screen parts for the implementation of the proposed algorithm



The Encryption process screen





## Appendix B



### Improving AES Algorithm using Multiple Secret Keys

تحسين خوارزمية AES باستخدام المفاتيح السرية المتعددة

Prepared by

Suhair Haseeb Al-Waith

401610094

Supervisor

Prof. Hamza A. Al-Sewadi

Thesis Submitted in Partial Fulfillment of the Requirements  
for the

Degree of Master in Computer Science

Department of Computer Science

Faculty of Information Technology

Middle East University

نسبة الاستلام: 12%.

عميد الدراسات العليا والبحث العلمي

Handwritten signature and date: 2012.05.12