KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY, KUMASI
COLLEGE OF SCIENCE
INSTITUTE OF DISTANCE LEARNING
DEPARTMENT OF INFORMATION TECHNOLOGY


# AN ENHANCED ECC FOR SECURING DATA, COMPARATIVE STUDY

A THESIS SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY, KUMASI, IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF THE MASTER OF SCIENCE DEGREE IN INFORMATION TECHNOLOGY

BY
EDWARD KWADWO BOAHEN
(PG4508315)

[NOVEMBER 2017]

## DECLERATION

I, hereby declare that under supervision I have personally undertaken the study. All other works cited in this study have been acknowledged and discussed. To the best of my knowledge, it contains no material previously published by another person nor material which has been accepted for the award of any other degree of this and in any other university.

KNUST

**EDWARD KWADWO BOAHEN** (PG4508315)

(Student)

Date …… Signature ……

I declare that I have supervised the student in undertaking the study. I further confirm that, the student has had constant interaction with me for the guidance and direction and has my permission to present the work for assessment.

**Dr. J.B Hayfron-Acquah (supervisor)**

Date ……. Signature …….

## ABSTRACT

The purpose of this research is to enhance a cryptographic system called the Elliptic Curve Cryptographic System. Elliptic Curve cryptosystem is a technique of public-key encryption which is rooted on the arithmetical construction of elliptic curves over finite fields. Elliptic Curve Cryptographic System necessitates smaller keys compared to non-ECC cryptography to offer equal security. Elliptic curves are valid for digital signatures, key agreement, generators, pseudo-random and other related tasks.

The first phase of the project involves understanding the key exchanges of Diffie-Hellman and also applying the properties of the Elliptic Curves, and it is terminated with key facts that the

Elliptic Curve Cryptography has a shorter key length, saves bandwidth which facilitates key generation during the encryption/decryption of data, also the assurance of faster encryption and decryption, and notwithstanding its efficiency and efficacy in small devices.


KNUST

## TABLE OF CONTENTS

**DECLARATION**……..ii

**ABSTRACT** ……iii

**TABLE OF CONTENTS**……iv

**LIST OF TABLES**……..vii

**LIST OF FIGURES**……viii

**LIST OF ABBREVIATIONS**……..x

**CHAPTER 1**.....1

**INTRODUCTION**..... 1

1.0 Background..... 1

1.1 Cryptography..... 1

Symmetric Encryption..... 5

Digital signatures in SSL/TLS..... 7

1.2 Problem Statement:..... 8

1.3 Research Objective:..... 10

1.4 Research Question:..... 10

1.5 Scope..... 10

1.6 Justification of the study..... 10

1.7 Organization of Thesis..... 11

**CHAPTER 2**.....12

**LITERATURE REVIEW**..... 12

2.0 Introduction..... 12

2.1 Symmetric Encryption..... 13

2.2 Asymmetric Encryption..... 14

2.3 The Representation of Commonly Used Encryption Algorithms..... 15

2.3.0 Rivest-Shamir-Adleman (RSA)..... 16

2.3.1 Data Encryption Standard (DES)..... 18

2.3.2 Triple DES (3DES)..... 18

2.3.3 AES (Rijndael)..... 20

2.4 Notions of security....23
2.5 HTTP Secured Connection....24
2.6 ECC and RSA Comparison Parameters....26
2.7 Summary....30
**CHAPTER THREE....31**
**METHODOLOGY....31**
3.0 Introduction....31
Table 3.1 The Bouncy Castle in C # and Java currently support the following ECDSA curves 33
3.1 Proposed Procedure....36
3.2 Mathematics of Cryptography....41
3.2.1 Groups, Rings, and Fields....41
3.2.2 Properties of groups....42
3.3 Diffie-Hellman Key Exchange....42
3.3.1 Steps Used In the Diffie- Hellman Key Exchange Algorithm....43
3.4 Elliptic Curve Arithmetic....45
3.4.1 Elliptic Curve Over Real Numbers....45
3.4.2 Elliptic Curve over GF (2m )....45
3.4.3 Elliptic Curve Over GF (p) or $Z_p$....45
3.4.4 Operations In Elliptic Curves....46
3.4.5 Generating Point on the Elliptic Curve....47
3.4.6 Algorithm to Generate Points on the Elliptic Curve....48
3.5 The Steps Involved in the Elliptic Curve Cryptography....48
3.6 Materials and Tools....51
**CHAPTER 4....52**
**DATA ANALYSIS....52**
4.1 Comparative Analysis of Popular Secret Key Algorithms or Methods by Using Two Different Hardware Platforms....53
4.2 Key Length Comparison of RSA and ECC....58
4.3 Performance Evaluation of RSA and ECC....59
4.4 ECC and RSA Comparison....60
**CHAPTER FIVE....62**

**SUMMARY AND CONCLUSIONS** ..... **62**
5.1 Introduction ..... 62
5.2 Summary of Findings ..... 62
5.3 Conclusion ..... 63
**REFERENCES** ..... **64**

KNUST

## LIST OF TABLES

Table 3.0: A comparison of public-key cryptosystems (Vanstone, 2003).....32

Table 3.1 The Bouncy Castle in C # and Java currently support the following ECDSA curves:. 32

Table 3.2: Equivalent key sizes (in bits).....35

Table 4.0: Algorithm settings used in the experiment.....56

Table 4.1: Key Length Comparison of RSA and ECC.....57

Table 4.2: Key Generation Performance.....59

Table 4.3: Signature Generation Performance.....59

Table 4.4: Signature Verification Performance.....60

Table 4.5: ECC and RSA Overview.....60

KNUST

## LIST OF FIGURES

Figure 1.0 Encryption and Decryption..... 5

Figure 1.1 Symmetric Encryption..... 5

Figure 1.2 Asymmetric Cryptography.....7

Figure 1.3 The digital signature process.....8

Figure 2.0 Encryption Algorithm Types.....13

Figure 2.1 Symmetric Encryption..... 14

Figure 2.2 Diagram of Symmetric Encryption..... 15

Figure 2.3 RSA processing of Multiple Blocks.....17

Figure 2.4 General Depiction of DES..... 19

Figure 2.5 AES Encryption Standard..... 21

Figure 2.6 AES Process..... 23

Figure 2.7 Classification of Encryption.....26

Figure 2.8 Digital Signature and Verification Process..... 30

Figure 3.0 C++ Code showing the various procedure for the mathematical models..... 40

Figure 3.1 Diffie Hellman Key Exchange Algorithm..... 43

Figure 3.2 Diffie Hellman Key Exchange Algorithm..... 44

Figure 3.3 Identities of Elliptic Curves..... 47

Figure 4.0 Speed Benckmarks for some commonly used cryptographic methods.....51

Figure 4.1 Comparative execution times of encrypting algorithms..... 52

Figure 4.2 Average Execution time with respect to bytes processed..... 53

Figure 4.3 Execution times of encryption algorithms in ECB mode.....53

Figure 4.4 Average Execution Time.....54

KNUST

Figure 4.5 Comparative execution time in seconds of encryption in ECB mode..... 55

Figure 4.6 Comparative time in seconds of encryption algorithm in ECB mode..... 55

Figure 4.7 Perfomance Results with ECB Mode.....57

Figure 4.8 Key Length Comparison for RSA and ECC Cryptosystems..... 58

Figure 4.9 Perfomance of RSA and ECC..... 58

KNUST

## LIST OF ABBREVIATIONS

| | |
|---|---|
| 3DES | Triple DES |
| AES | Advanced Encryption standard |
| DES | Data Encryption Standard |
| DH | Diffie-Hellman |
| DSA | Digital Signature Algorithm |
| ECC | Elliptic Curve Cryptography |
| FIPS | National Institute of Standards and Technology |
| FTP | File Transfer Protocol |
| GB | Gigabyte |
| HTTP | Hypertext transfer protocol |
| IDEA | International encoding algorithmic rule |
| IETF | Internet Engineering Task Force |
| IMAP | Internet Message Access Protocol |
| LDAP | Lightweight Directory Access Protocol |
| NIST | National Institute of Standards and Technology |
| RAM | Random Access Memory |
| RSA | Rivest-Shamir-Adleman |
| S/MIME | Secure/Multipurpose Internet Mail Extensions |
| SET | Secured Encryption Technology |
| SSH | Secured Shell |
| SSL | Secured Socket Layer |
| TCP | Transport communication protocol |

KNUST

# CHAPTER 1

## INTRODUCTION

Cryptography is the technique that can be implemented to secure data on a network. It can also be considered as the science of securing information to be transmitted in a manner that the authorized recipient only is able to able to retrieve or access the information. As far as communication is concern, it is vital to encrypt the message to conceal it from intruders. Network security is primarily based on cryptography (Stallings, Data and Computer Communications, 2005). The art of protecting information (encryption) is usually done by converting it into associate undecipherable format (encrypted text), which is referred to as the cipher text.

### 1.0 Background

The objective of secure communications is based on confidentiality or secrecy (S), that is, to cover the contents of a publicly exposed message from unauthorized recipients or intruders. In modern-day business and communications, it is normally difficult for the receiver to know and confirm if the communication has not been tempered with at some stage during the transmission or whether an intruder has tampered with it. One major problem in message authentication is that, you cannot tell exactly if it is the actual message since there is usually no delay in transmission. In this research, HTTP communications securities are discussed with emphasis on applications that cannot be satisfactorily threatened by means of current cryptographic procedures. A completely new idea, the elliptic curve encryption/decryption channel solves the brand-new requirements in comfy transmissions. Cryptosystems are symmetric when the sender and recipient secretly possess the same key. The secret keys mentioned are used within the encoding procedure to present indecision, which can be detached in the procedure of decryption

by the certified receiver using his key. With this method if the key currently used by both parties is compromised, the communication between the two parties would not be secure and hence will create a barrier. The new cryptosystems are asymmetric in the sense that there are two keys being used by the parties in the communication. One of the keys is public to both and the other is private. Secure HTTP Communication is a prerequisite for today's online exchanges and communication. Whether trading financial, business or individual data, individuals need to be in the know of the parties involved in the conveying (verification) as well as the surety that the records will not be reformed (information trustworthiness) nor unveiled (confidentiality) during transmission. As such, the Secure Sockets Layer (SSL) protocol is considered as one of the best alternatives in attaining these objectives. The SSL convention is application free – theoretically, applications that keeps running over TCP will likewise keep running over SSL. Making it an imperative motivation why the spoken about transmissions have outperformed that of additional security conventions, for example S/MIME, SET and SSH, and. There are a number of cases of utilization conventions like LDAP, FTP, TELNET and IMAP and using Secure Shell Layer in various transmissions. In any case, the greatest use of Secure Shell Layer is for safeguarding Hypertext Transfer Protocol, the principle convention regarding the World Wide Web (Frier, Karlton, & Kocher).

Amid its origination with Netscape in 1990s, throughout its institutionalization inside the IETF (Internet Engineering Task Force) around the later years of the 20th century, the convention and its usage have been examined by a distinguished security expert in the world. Currently, the Secure

Shell Layer is trusted to protect exchanges for delicate systems going from financial transactions, to stock exchange, to electronic business. The utilization of Secure Shell Layer forces a

prominent execution price on web servers. Coarfa et al. (2006) having testified that web servers that are secured runs 3.4 to nine times slower contrasted with steady web servers on similar equipment stage. Moderate reaction time is a noteworthy reason for disappointment for online customers and regularly drives them to surrender their electrical spending baskets amid look at. As indicated by one gauge, the potential income misfortune from e-trade exchanges prematurely ended because of Web execution issues surpasses a few billion dollars every year (Wagner & Schneier, 1996).

SSL uses RSA mode of encoding to convey an arbitrarily picked mystery that is utilized to determine keys for information encoding and confirmation. The RSA decoding procedure is the most figure concentrated section of the Secure Shell Layer exchange for a safe web server. A few merchants, for example Rainbow, nCipher, Sun and Broadcom and now offer specific equipment to devolve RSA calculations to improve its server execution. This research sightsees the use of Elliptic Curve Cryptography (ECC), a resourceful option to RSA, as a means of refining the Secure Shell Layer performance without selecting posh superior purposeful hardware. ECC was first projected by Victor Miller and singularly by Neal Koblitz in the middle of the 20th century and has changed into an established public-key cryptosystem.

### 1.1 Cryptography

Cryptography refers to the means of translating plaintext (readable text) to cipher text (an unreadable form) (IBM, 2016).

This occurs as follows:

i. The sender is required to translate the plaintext or the information to be concealedinto a form which is not easily understood, known as a cipher text. This procedure of translating the original message into the cipher text is known as encoding or encryption.

ii. The cipher text is then sent to the receiver.

iii. The recipient then translates the cipher text message back to its original form, a processreferred to as decryption or decoding.

The conversion or translating process entails a series of mathematical operations that alter the message into a form which is difficult to understand or which has a different meaning from the original message during transmission but does not affect its content when it is converted back to its original form. Cryptographic procedures guarantee confidentiality and ensures security since it protects messages against unauthorized access. This is because an encoded message is not comprehensible or has a different meaning from the original message. Digital signatures for example which provide an assurance of message integrity, use encryption techniques.

Usually, cryptographic procedures implement a general algorithm that makes use of use of keys specified by the sender and receiver.

There are two groups of cryptographic algorithms:

Public Key System

Symmetric Key System

Though the encoding and decoding algorithms implemented can be publicly known, it is required that the shared key and the private key must be kept secret. Figure 1.1 shows the encoding and decoding process using the same symmetric key.

KNUST

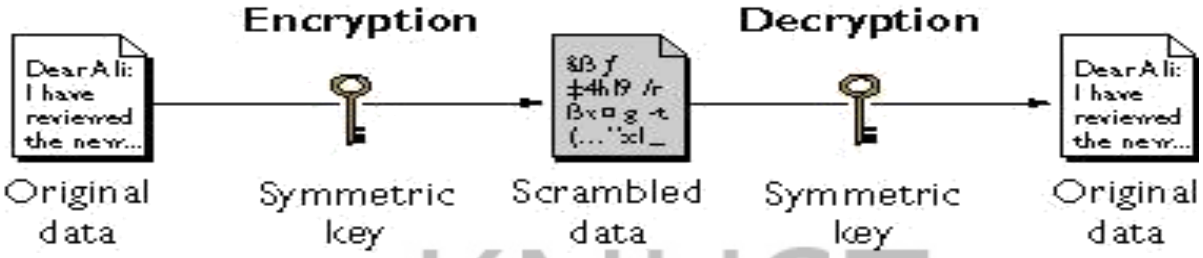


**Figure 1.1 Encryption and Decryption**

KNUST

**Symmetric Encryption**

This required the use of a common key by both parties. Here, the same key is used to encode and encode the message. Figure 1.2 shows the same used for encryption and decryption of data.

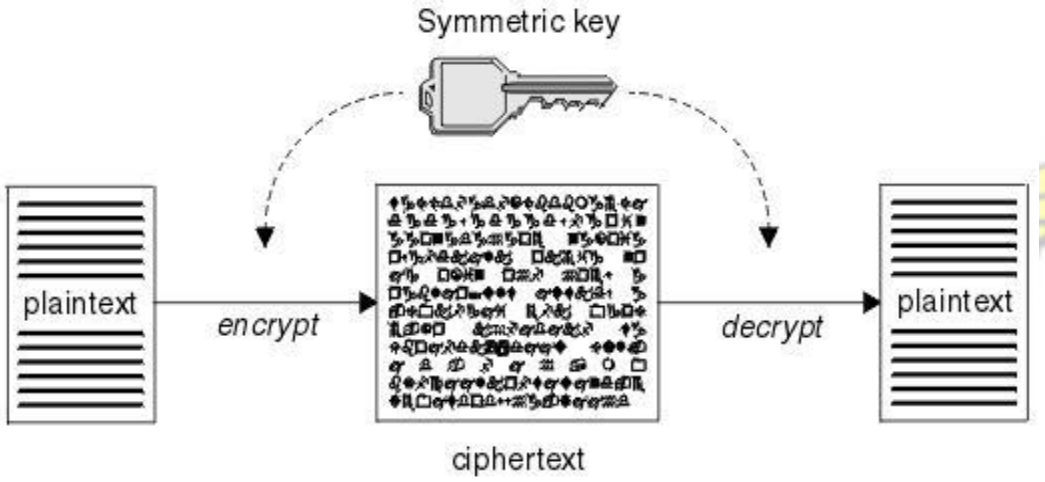


**Figure 1.2: Symmetric Encryption**

Ciphers of symmetric key are valued because:

- It is comparativelycheap to generate a robust key for these ciphers.
- The keys are shorter for the level of protection they afford.
- The algorithms are comparativelycheap to process.

Subsequently, representing symmetric cryptography can be very feasible in light of the fact that you don't face any noteworthy delays in time as an aftereffect of the encoding and decryption. Symmetric cryptography, moreover, gives a level of authentication since information twisted with one symmetric key cannot be deciphered with another symmetric key. In this manner, the length of the symmetric key is kept unknown by the two parties using it to encrypt correspondences; every meeting can ensure that it is speaking with another length of the deciphered messages keeps on making logic. Habitually, with a symmetric encryption, the key can be exchanged with a trusted member. You can be guaranteed that all messages that are exchanged, are encrypted with a specific key, between the parties and must be decoded by the other party that possesses that specified key. The key must be kept by every party in the exchange. Subsequently, these keys are likewise referred to as unknown key figures. In the event that another party gets a hold of the key, it breaches privacy. Anyone with an uncertified symmetric key cannot just decipher messages sent with that key yet can scramble new messages and send them as though they came from one of the two parties.

Symmetric-key structures are less difficult and enhance speed; however, their essential drawback is that the two gatherings should by a few means change the critical thing in a casual way. Open key encryption keeps away from this bother because of the reality people in general key might be administered in a non-comfortable manner, and the private key is in no way, shape or form transmitted.

**Asymmetric Cryptography**

Asymmetric cryptography, which is also known as public key cryptography, useboth public/open and private keys to encode and decode information. These keys are huge numbers that have been matched together yet are not indistinguishable (deviated). Here, the public key can be accessed by

everybody; it is known to people as the public key. On the other hand, the private key is personal to the parties communicating. Both keys can be used to encode a message; however, the inverse key is used for decoding. Figure 1.3 shows how the asymmetric cryptography works.

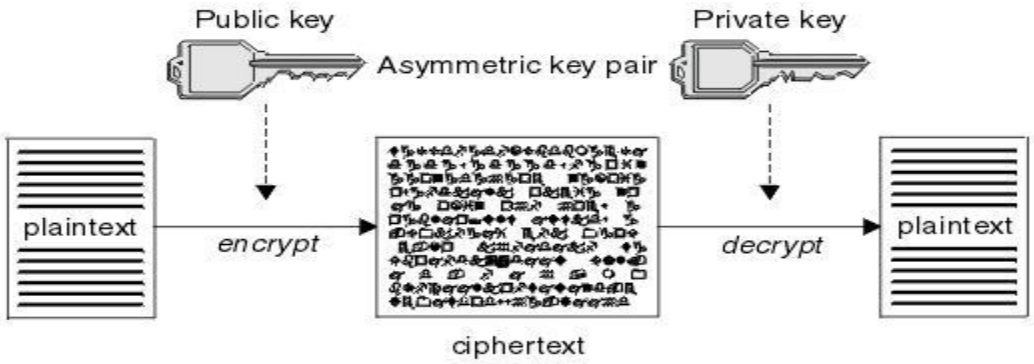


**Figure 1.3 Asymmetric Cryptography**

**Digital signatures in SSL/TLS**

This involves the use of a digital signature which is created by encrypting a representation of a message. Here, the encoding utilizes the private key of the party involved and, for effectiveness, for the most part, works on a message procedure as against the message itself.

For every piece of information that is marked, a different digital mark is produced, which is unlike all manually written marks, which do not depend upon the substance of the record being agreed upon. With instances where two exceptional messages are marked digitally by a similar substance, both marks vary, nevertheless both marks can be checked with a similar open key, that is, general society key of the element that marked the messages.

The steps involved in the formation of digital signature process are as follows:

i. The sender is required to work out a message digest and then encode the digest using the key in possession to form the digital signature.

ii. Together with the message, the sender then transmits the digital signature to the receiver.

iii. The recipient, on the other hand, decodes the digital signature with the sender's public key, by regenerating the sender's message digest.

iv. The recipient then develops a message digest from the message or data received and verifies that the two digests are the same.

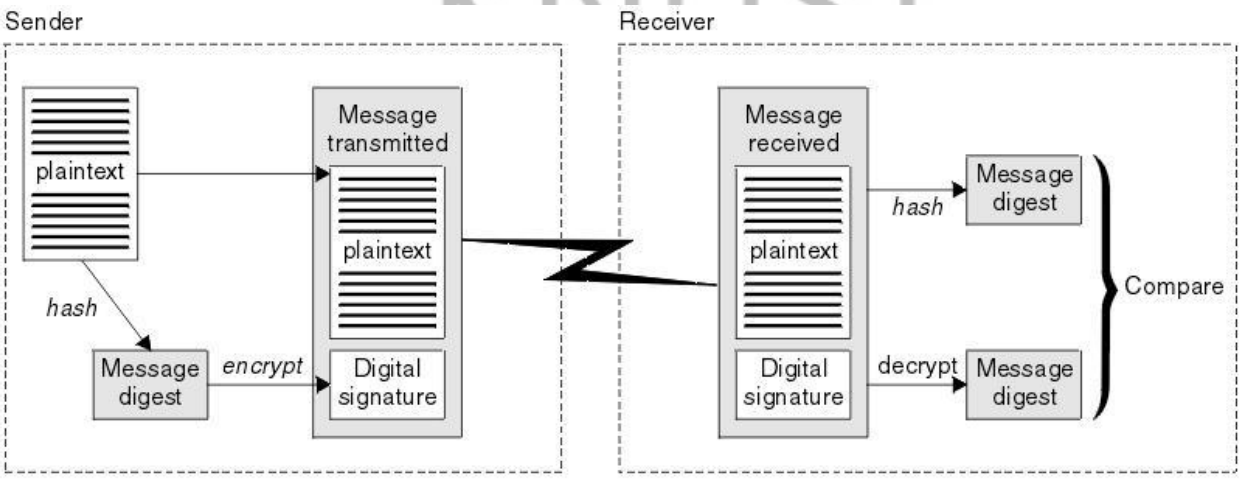


**Figure 1.4: The digital signature process**

## 1.2 Problem Statement:

It is evident that the security of RSA is based on the computational task of considering extensive numbers. As such, as figuring power increases, and more productive calculations are established, the capacity to consider more lengthy numbers increases. The quality of encryption by directly fixing to key size and multiplying key length conveys an exponential increment in quality, thoughit is considered to it impair execution. Basically, RSA keys are either 1024-bits or 2048-bits in length. With the increase in computing power, specialists trust that the 1024-piece keys could be eased up soon. That is why stakeholder is looking into the implementation of a base key length of 2048-bits.

Because the strength of the security in the HTTP-SSL transmission is dependent on the encryption key size, expects try to increase the key size to 4096 bit. The increase in the key size therefore requires the technical staff to face the following issues:

- There's an increase in encryption computation time.
- The SSL handshake at the start of each connection will be slower.
- There's an increase in CPU usage during handshakes.

Websites that have busy data exchange such as amazon.com, google.com, and the likes cannot afford to defray customers with such connectivity issues and the problems stated up here. Therefore, there should be a new way of solving these problems that is why ECC encryption is gaining popularity and attention.

KNUST

### 1.3 Research Objectives:

- To develop a cryptographic framework to protect data transmission with HTTP protocol.
- To develop a cryptographic framework to improve SSL security with HTTP protocol.

KNUST

### 1.4 Research Questions:

- How can we protect data transmission with HTTP protocol?
- How can we improve the SSL security with elliptic curve cryptography encryption?

### 1.5 Scope

The project reviews the existing encryption and decryption algorithms and finds the best asymmetric key encryption algorithms in terms of key length used, key length generation performance, the efficient security used, the computational overheads saved, bandwidth savings and its efficiency in small devices.

### 1.6 Justification of the study

Elliptic Curve Cryptography (ECC) is developing into a substitute for the traditional public-key cryptosystems. ECC provides commensurate security with reduced key sizes ensuing in quicker computation times, low consumption of power, notwithstanding, bandwidth and effective memory usage. While these features make ECC alluring for devices with limited resource such as mobile devices, they are likely to reduce the computational load on protected web servers. This research work on ECC will study the performance impact with regards to SSL, the mainly used Internet

security protocol. This project aims to create an enhanced version of elliptic curve cryptography and use it to compare other asymmetric cryptographies

### 1.7 Organization of Thesis

Chapter 2 has the literature review on the elliptic curve cryptography Encryption, as far as HTTP security is concerned, is examined. Programming-based answers from various literature are reviewed to point out the various techniques used. Chapter 3 has a general representation of the elliptic curve cryptography Encryption is illustrated through a web application. Our proposed method will be discussed into detailed. Chapter 4 Chapter 4 comprises the beneficial exploratory consequences of the investigation. By employing the conclusion of our procedure in the third chapter to execute analytical and conventional report of the work done. Chapter 5 has closing comments are exhibited in this part, and further augmentations that can be made with this anticipate are additionally analyzed.

.

KNUST

# CHAPTER 2

# LITERATURE REVIEW

## 2.0 Introduction

Cryptography, bearing Greek origins, meaning "secret writing" is the area of technology where the focus is on the study of designs of safeguarding secrecy and/or genuineness of messages and the protection of sent messages against security attacks. Cryptography has undoubtedly become one of the most principal areas of communication security and thus an imperative block of computer security. Cryptographic techniques provide a secure communication in the occurrence of adversaries to stay abreast with data securities such as data authentication, non-repudiation, confidentiality, and information integrity.

The method of converting plain text or ordinary information into incomprehensible text classified as cipher text in cryptography is termed encryption. This cipher text is perceivable exclusively to someone who is aware of a way to decode it, and any resister that may understand the cipher text should not be intelligent to confirm all-round the first encrypted message. Furthermore, any lawful party can decipher the cipher text using a secret writing algorithmic rule that sometimes requires a secret writing key. With that well stated, there are two main frameworks used for providing the absent protection by converting a message into a state, where if it were captured in transfer, the innards of the novel message could not be overtly exposed, and these techniques fall under two generic classes and these are show in Figure 2.0.

i. Public Key or Asymmetric Systems

ii. Secret Key or Symmetric Systems.

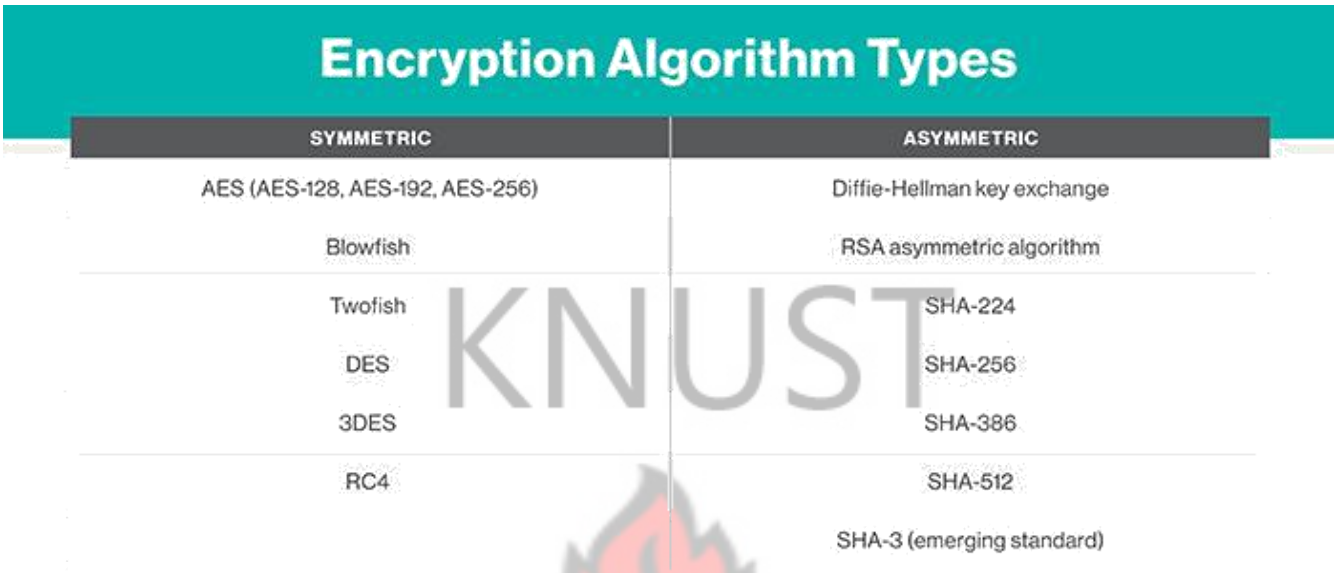


**Figure 2.1 Encryption Algorithm Types**

## 2.1 Symmetric Encryption

Per symmetric encryption, the undistinguishable key is used to decrypt and encrypt the information or data. Symmetric key algorithms are considerably faster systematic all than asymmetric algorithms. This is because; the process of encryption is not as much as complicated as asymmetric algorithms. The fact that the parties involved have contact to the secret key is one of the key downsides of encryption using symmetric key, compared to encryption using public-key, also termed asymmetric key encryption. The length of the size /length is imperative for the forte of the security at hand. There are intrinsic problems with symmetric key encryption which is that the key must be managed delicately and moreover the dispensing of the shared key is a chief security menace. Figure 2.2 gives a brief description of Symmetric Encryption


**Figure 2.2: Symmetric Encryption**

## 2.2 Asymmetric Encryption

Public-key cryptography is as well-known as asymmetric encryption because of the spatial property in the control of key data by the parties involved. Specifically, one of the parties features a secret key whereas the other party has the universal public key that equals this secret key. This is often in distinction to the evenness within the personal key setting, wherever each party has an identical key. Uneven secret writing is an alternative term for public-key secret writing. The study of uneven secret writing begun by looking for applicable philosophies of security, representations and formalizations through seized data. There is a tendency to think about constructions wherever there is a glance at the way to style and analyze numerous schemes. There is incredibly very little dissimilarity between bilateral and uneven encryption; not rather more than the very fact that within the latter the human gets the

general public key as data input. This is often important and encouraging to recollect. Figure 2.3 also gives a brief description of Symmetric Encryption and Symmetric Decryption.

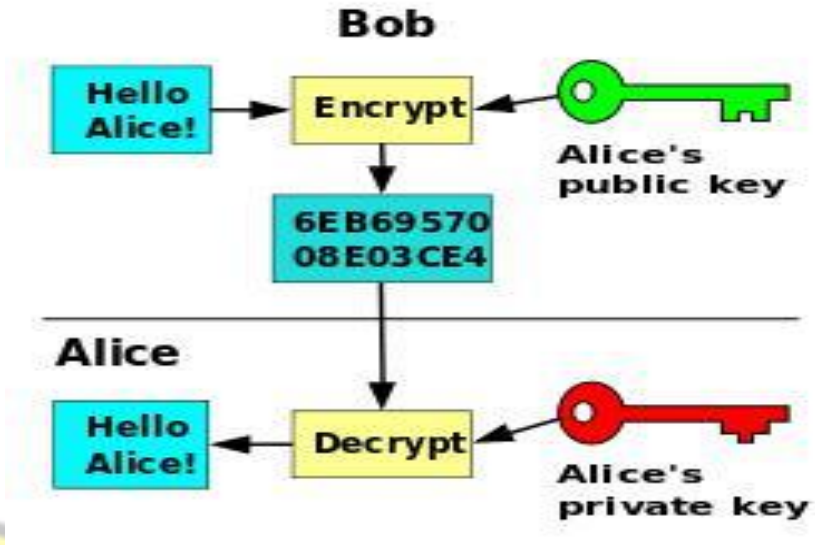


**Figure 2.3 Diagram of Symmetric Encryption**

## 2.3 The Representation of Commonly Used Encryption Algorithms

The transport, alteration and generation, of keys are completed by encryption rules. It is conjointly termed as the cryptographic rule. There are numerous cryptographic algorithms available for the encryption of data and information. The forte of the encryption rule deeply depends on the computing device used for key generation. The encryption algorithms are vividly declared below:

### 2.3.0 Rivest-Shamir-Adleman (RSA)

Rivest et al. (1978) designed RSA, known to be amongst the finest recognized public key cryptosystems for key exchange or digital signatures or coding of blocks of knowledge. RSA utilizes an inconstant size-coding block and an adjustable key size. It's associate degree public key cryptosystem, buttressed variety theory, that may be a system for block cipher. It deploys two prime numbers to produce the universal public and private keys. The two very unalike keys are used for coding and decipherment purposes. The sender then encrypts the message with public key of the recipients which is then sent through a reliable, but not principally a secret route and once the message gets conveyed to the recipient, then the recipient will use his/her own personal key to decrypt the message. RSA operations are rotten in four (4) stages;

i. Key generation
ii. Key distribution
iii. Encryption
iv. Decryption.

RSA has some disadvantages, and these are:

KNUST

- **The Need of Security Proof**. RSA is dependent on the drawback of the factorization of massive numbers but is identical to the factoring, which has not been verified hypothetically. In the mean time because there exists no evidence of broken or fragmented RSA factorization would be required. If there exists an algorithm that can rapidly crumble a huge number, the RSA algorithm's security would be jeopardized.

Figure 2.4 demonstrates the arrangement of the algorithmic rule that RSA follows aimed at the coding of several blocks.

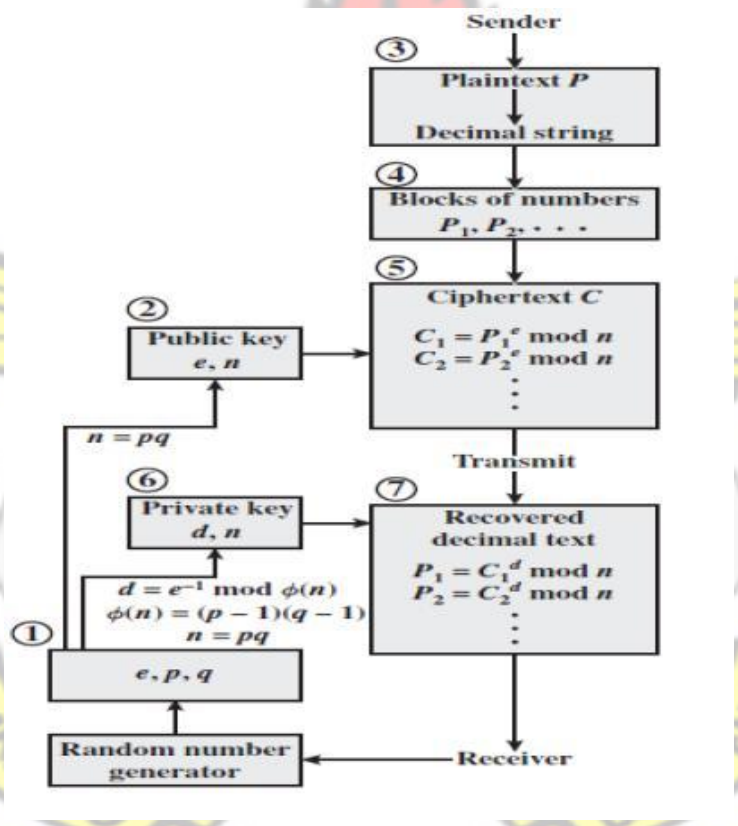


**Figure 2.4 RSA processing of Multiple Blocks**

**(Singh & Supriya, April 2013)**

### 2.3.1 Data Encryption Standard (DES)

DES is one of a kind with broadest acknowledged, openly attainable crypto graphical structures. It was established by the computer manufacturing company International Business Machines in the seventies but was far along espoused by the National Institute of Standards and Technology (NIST), as Federal data process commonplace forty-six (FIPS saloon 46). Data encryption standard is a block cipher that is intended to cipher. Though the input key for DES is just 64 bits long, the key used is unbiasedly 56 bits long. The least significant bit in all computer memory units could be a bit and might be set, so the area unit repeatedly associate in nursing odd range of 1's in every unit of the computer's memory. The algorithmic rule goes through 16 reiterations that intertwine blocks of plaintext with values gained from the key. The algorithmic rule changes 64-bit input into a sequence of stages into a 64-bit yield. A correspondent phase, with a comparable key area unit is used for deciphering. There are components of numerous bouts and habits chronicled that exploit the softness of DES in governing doubtful block ciphers. Notwithstanding the mounting matters regarding its defenselessness, DES remains extensively employed by financial facilities and substitute businesses internationally to guard delicate on-line applications. The algorithmic rule courses with subordinates in treatment preliminary variation, sixteen rounds block cipher and a concluding permutation.

### 2.3.2 Triple DES (3DES)

3DES algorithmic program was established to discourse the palpable errors in the Data encryption standard algorithm, while not developing an entire novel cryptosystem. Encoding of data with the normal DES utilizes a 56-bit key and isn't considered clad to engrave delicate data. 3DES simply encompasses the length of key of DES by registering the program algorithm

three (3) stints in series through three (3) wholly unalike key pairs. The mutual key length is 168 bits, which is three (3) times 56. TDEA comprises victimization 64-bit law enforcement agency keys (K1, K2, K3) in write-Decrypt- Encrypt (EDE) mode, that is, the plaintext is encoded with K1, then decoded with K1, and then encoded once more with K3 (3DES, n.d.). The standards outline three keying options and are carefully shown in Figure 2.5 below:

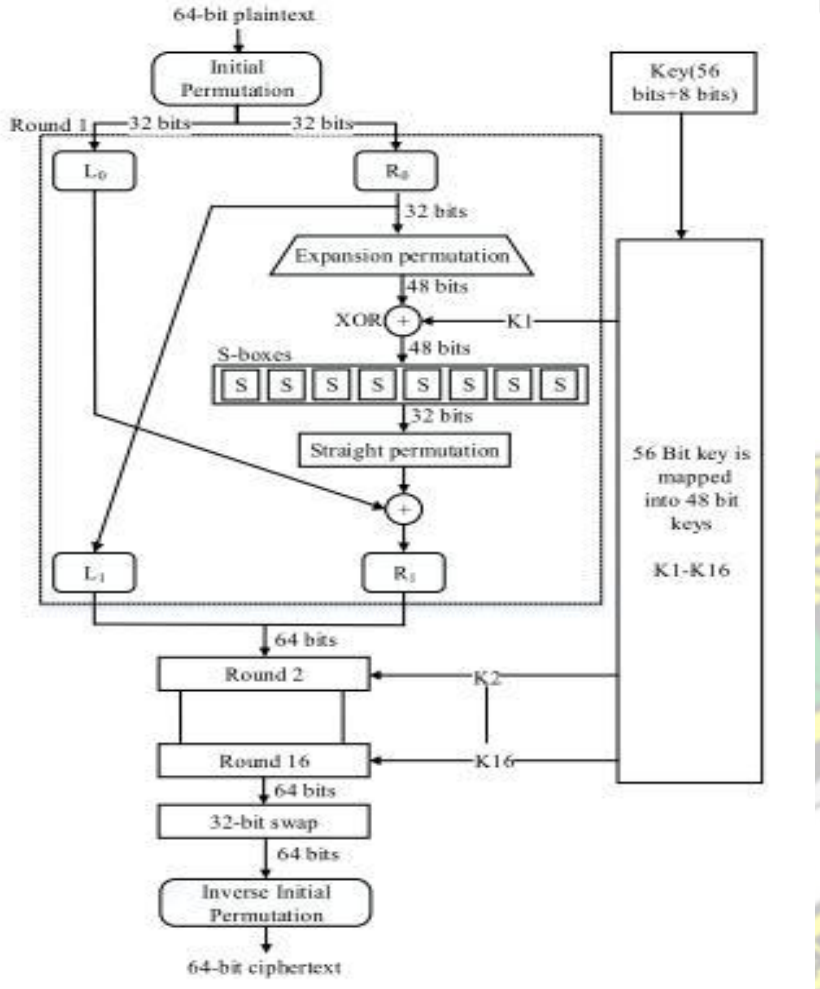


**Figure 2.5: General Depiction of DES**

**(Singh & Supriya, April 2013)**

### 2.3.3 AES

The Advanced Encryption Standard (AES), initially identified as Rijndael is a description for the encoding of electrical information and data setup by the United States National Institute of Standards and Technology (NIST) in 2001.

AES is a subsection of the Rijndael cipher industrialized by two Belgians, Vincent Rijmen and Joan Daemen, who pitched to the NIST throughout the AES choice process. Rijndael refers to a family of ciphers with disparate key and block dimensions.

For AES, out of the Rijndael family, NIST judiciously chose three members each with a block length of about 128 bits, but three (3) varying key lengths: 192, 128 and 256 bits.

AES has been accepted by the United States administration and is now used internationally. It succeeds DES which was initialized in 1977. The AES implements a symmetric-key algorithm, which presupposes that an identical key is utilized for both encoding and decoding of data and information.

AES is grounded on a structural design code called substitution–permutation network, a mixture of mutual permutation and substitution, with its swiftness in both hardware and software. Contrasting its forerunner DES, Feistel networks are not used. AES is a modification of Rijndael which has at least a static block length of 128 bits, and a key size of 256, 128 or 192 bits. By dissimilarity, the Rijndael requirement is stated with key sizes and blocks that might be slightly multiple of 32 bits, having 128 bits and 256 bits for the minimum and maximum length respectively. AES is available in numerous several undisclosed inscription packages, sub-degrees in that the first in public accessible cipher approved by the National Security Agency (NSA) for high secret data in an NSA approved science module as shown in Figure 2.5.

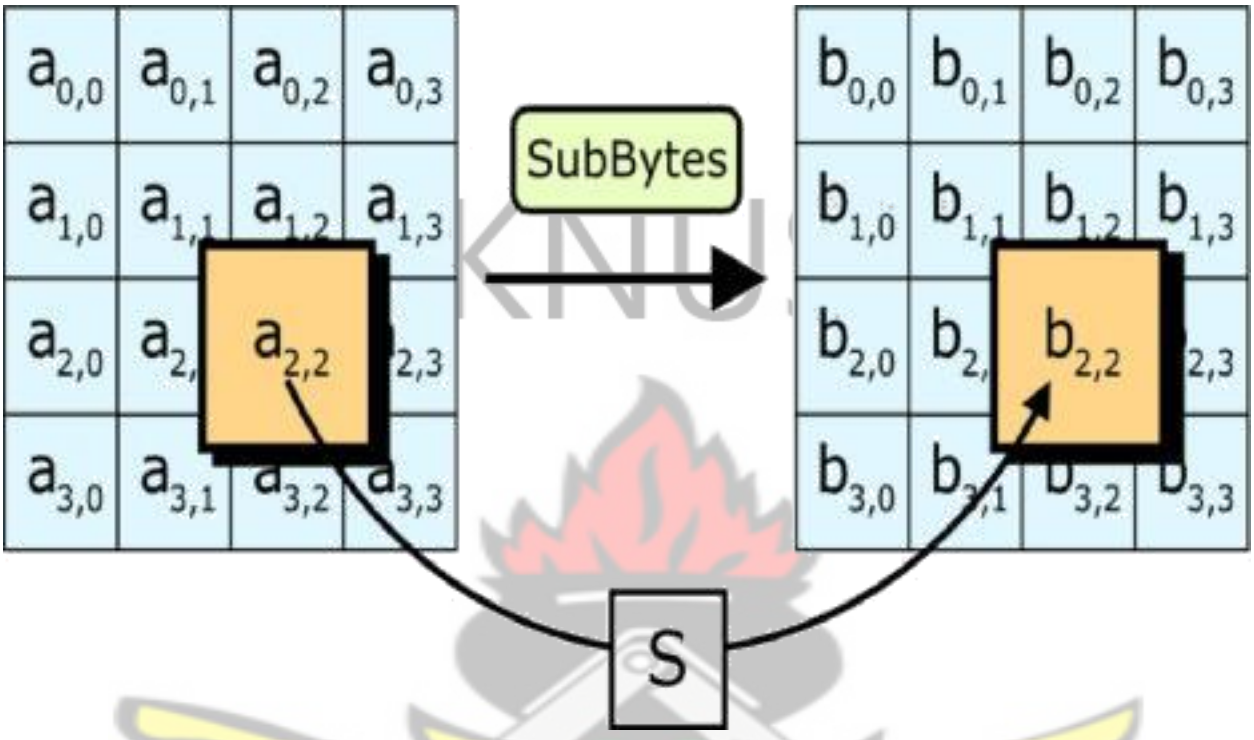


**Figure 2.6: AES encryption standard**

**AES (RC6)**

RC6 a derivative of RC5 was established to assemble the necessities of the Advanced Encryption Standard (AES) competition and it was designated to be a member of the five qualifiers plus being offered to the NESSIE and CRYPTREC assignments. RSA Security patents it. RC6 is highly efficient in relation to security and compatibility.

**AES (Serpent)**

Serpent, also a front-runner of the Advanced Encryption Standard (AES), contested and was considered second to the Rijndael. It is a symmetric key block cipher, which Lars Knudse, Eli Biham and Ross Anderson, designed. Security offered by Serpent is grounded on additional accepted methods as compared to the others. The Serpent is available in the community sphere and yet to be unproved.

KNUST

**AES (Two Fish)**

Bruce Schneider et al. (1993) covers the Blowfish squad to improve the previous block cipher Blowfish to its revised form called Two fish to meet the standards of AES for the making of the algorithm. AES (Two Fish) one of the five finalists of the Advanced Encryption Standard contest. However, due to standardization it was not selected. The Two fish is also open to civic domain yet to be patented.

**AES (MARS)**

MARS was wholly considered to resist forthcoming developments in crypto systems. MARS is also a block cipher that was submitted at IBM's Advanced Encryption Standard contest and it was chosen as one of the five finalists. MARS was also considered as the least of the five, in terms of the variation of elevated flexibility, speed and security brands MARS is an excellent substitute for the encryption needs of the data world.

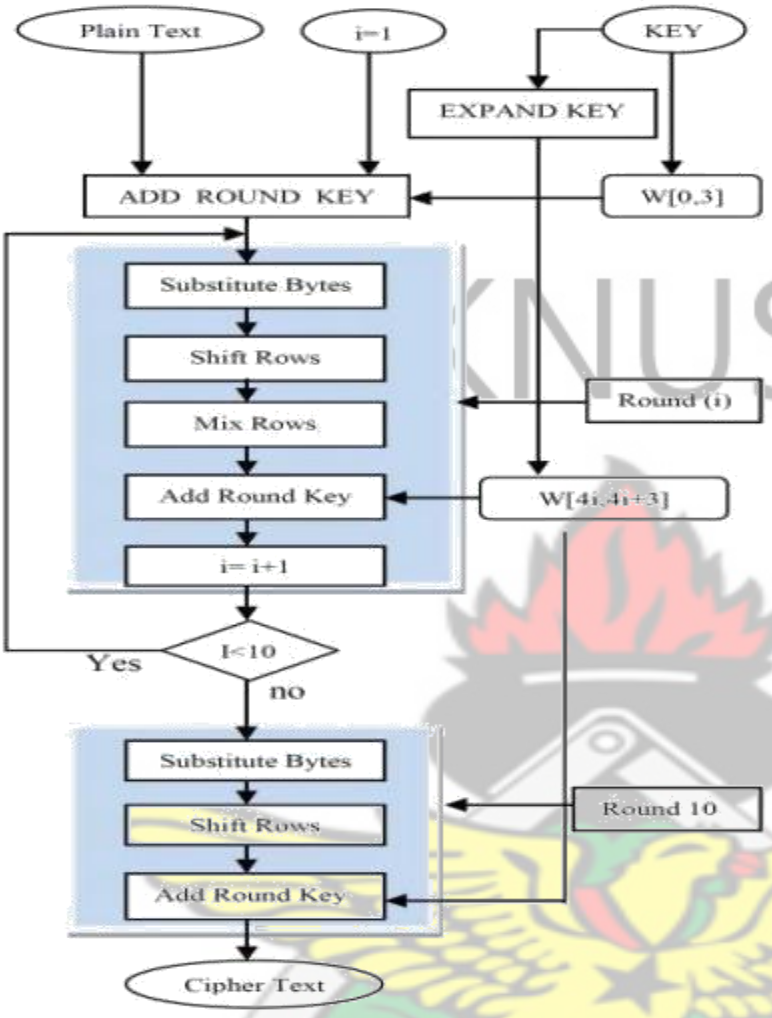


**Figure 2.7 AES (Advanced Encryption Standard) process**
**(Singh & Supriya, April 2013)**

## 2.4 Notions of security

Security of an encryption theme (whether parallel or asymmetric) is supposed to replicate the inability of given cipher texts (and any public info like a public key), to induce "non-trivial" info regarding the underlying plaintext. We tend to permit (having the goal of deciding some non-trivial info regarding plaintext from cipher texts) very different "attack" capabilities reflective different things. The foremost basic quite attack could be a chosen-plaintext attack, within which the resister

will acquire coding of messages of its selection. We tend to mention this kind of attack exhaustive within the context of parallel coding and argue that the definition of security within the sense of "left-or-right" captured security against these sorts of attacks during a robust sense. Within the uneven case, an equivalent is true, and that we can use an equivalent notion to capture security against chosen plaintext attack. Access to a "decryption oracle", this being a box that contains the key secret writing key and implements secret writing below this key.

KNUST

**2.5 HTTP Secured Connection**

Intrinsically, secured communication is highly demanded in this era of on-line transactions. In the exchange of information that may encompass the context of finance, trade or personal transactions, folks need to grasp with whom they are (authentication) and that they would like to confirm that the info is neither altered (data integrity) nor made known (confidentiality) in transit. The Secure Sockets Layer (SSL) protocol is considered the best alternative for achieving these goals. The SSL protocol is application freelance – abstractly, all applications that runs over TCP also can route over SSL. Thus making it a crucial reason why its preparation has outpaced that of alternative security protocols like SSH (Ylonen, Kivinen, Saarinen, Rinne, & Lehtinen, 2003) S/MIME and SET. The square measure several samples of application protocols such as TELNET, FTP, IMAP and LDAP which run evidently over SSL. Nevertheless, the prime purpose of SSL usage is to secure communication protocols (Fielding, 1999), the most used protocol for the World Wide Web.

The hypertext transfer protocol (Https) and the uniform resource identifier system has equal syntax which has the same quality of HTTP structure, except for the theme token. However, the HTTPS hints the browser in other to use a new secret writing layer of the SSL/TLS to shield the traffic

flow. SSL/TLS is particularly fitting to communications protocol, subsequently it will give a level of safety even though only one facet of communication is attested. This can be the instance with communications protocol transactions done on the web, wherever the consumer examining the server's certificate attests usually solely the server. HTTPS constructs a protected path over an insecure network. This provides shielding from attacks like eavesdropping and man-in-the-middle attacks, given that sufficient cipher suites area unit is used of which the server certificate being used is verified and convinced. Since communications protocol piggybacks HTTP is wholly on high of the TLS, primarily everything about communications protocols are often encrypted.

Probably this may include the requested universal resource locator (which specific online page that has been requested), question parameters, the headers, and the cookies (which typically contain identity info regarding the user). However, due to factors such as host (website) addresses and port numbers area unit which is an essential component of the TCP and IP protocols, thereby HTTPS cannot defend their revealing. In observing this, advise is that even with a properly organized internet server, intruders who eavesdrops will deduce the IP addresses and also the port variety of the online server (occasionally even the name e.g. www.example.org, however not the remainder of the URL) that one is to act with, likewise because the quantity which is data conveyed and length which is the length of period of the communication, that is not the content of the communication.

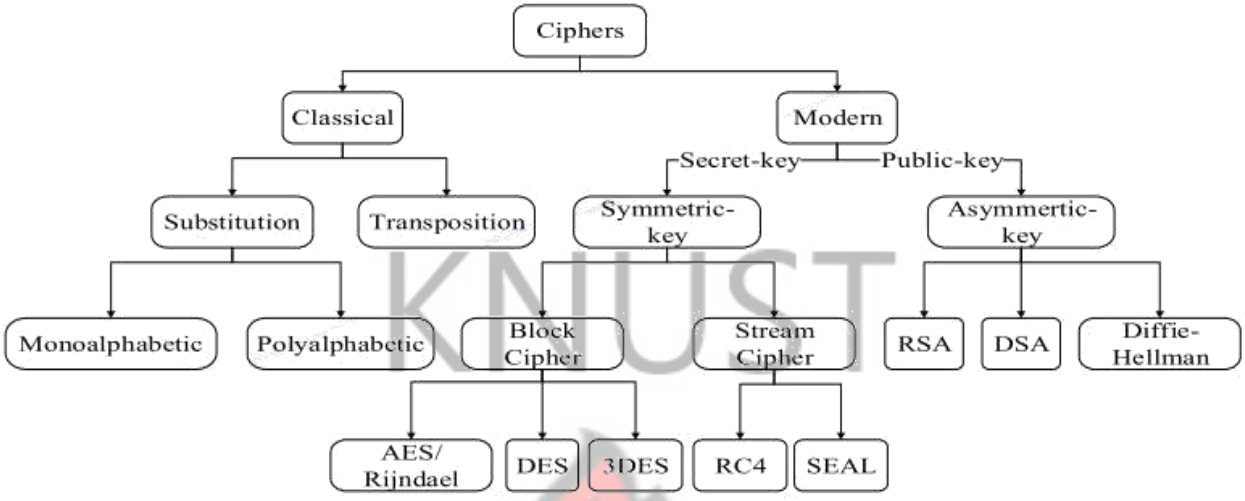


**Figure 2.8: Classification of Encryption Methods**

## 2.6 ECC and RSA Comparison Parameters

Three parameters would be discussed below and would be used extensively in comparing ECC and RSA to conclude our study and these are:

### 1) Key Size or Length

In most crypto systems, the security parameter which is the key length is of much importance. Key size of a crypto system algorithm is the number of bits in a key used. Key sizes determine the upper-bound of the algorithm's security. Since the safety of algorithms can be distorted by brute force attack. Preferably, key size should concur with the lower-bound of the algorithm's security. Undeniably, almost all symmetric-key crypto algorithms are designed to possess security equivalent to their key size. Nevertheless, prior to design, a new attack can be uncovered.

Triple DES was intended/structured to have a 168-bit key, but attacks of complexity $2^{112}$ is now known. Regardless, since the relation between security and key length is enough for a specific application, and then it would not matter if security and key length coincide. This is chief for asymmetric-key crypto algorithms, since no algorithm is familiar to serve this property; elliptic curve cryptography becomes evenly matched with an efficient and effective security of approximately half the length of the key.

KNUST

**2) Key Generation Performance**

Generally, key generation refers to the process of producing keys for cryptography. The key is then used to decrypt and encrypt data.

Symmetric-key algorithms comprising AES and DES are new cryptographic systems which include public-key algorithms such as RSA. They also implement a single common key that keeps data secret whilst protecting the key undisclosed. The Public-key crypto algorithms utilize a private key and a public key. A public key is made accessible to anyone normally with the use of a digital certificate. The person sending will then encrypt the data with the public key. The holder of the private key is the only one who would be able to decrypt the data.

Because symmetric-key crypto algorithms are much faster than public-key crypto algorithms in comparison, modern systems such as SSL and its successors TLS not excluding SSH utilize a combination of the two in which:

- A party receives someone's public key and encodes a minute amount of data and either some data or a symmetric key will be used in its generation).

- The rest of the communication (the remaining party) employs symmetric-key algorithm for encryption, which is typically faster for the encryption.

The easiest mode of reading data which has been encrypted is using the brute force. This attack– simply attempts all the numbers, up to its highest limit which is the length or size of the key. Hence, it is imperative to use a suitably long key length. Long keys take exponentially more time to attack, making a brute force attack imperceptible and unworkable. Presently, symmetric key algorithms implement 128-bits whilst public-key algorithms make use of 1024-bits.

### 3) Signature Generation and Signature Verification

A digital signature is a mathematical procedure used to confirm the authenticity and the integrity of a piece of data, software or digital document.

Digital signatures are a regular component of most cryptographic protocol groups, and are generally used for software circulations, financial businesses, forgery detection or tampering among others.

They implement asymmetric crypto systems. In many situations digital signatures provide a film of proof and secures the message that is sent through an unconfident channel: Accurately applied, a digital signature then gives the receiver motives to have confidence that the message was indeed sent by the demanded sender. Digital seals and signatures may be likened to the traditional handwritten signatures and seals. However, appropriately executed digital signature is more challenging to falsify than that which are handwritten. This is because, digital signature schemes are grounded in cryptography they must be applied properly to be very effective. They also offer non-repudiation, which means that the signer cannot efficaciously assert that they did not

sign a message, whilst also claiming the secrecy of their private key. Moreover, other non-repudiation schemes bid a time stamp for the digital signature, with the aim of exposing the private key, and even if that is successful, the signature is valid. Digitally signed messages can be denoted as a bit string. Figure 2.7 shows the definition of digital signature and verification.

There are three algorithms used in digital signature generation;

i. Key generation algorithm. This enables the selection of a private key homogenously and arbitrarily from a set of likely private keys. This algorithm yields the private key and a conforming public key.

ii. Signing algorithm. This algorithm, when provided a message and a private key, yields a signature.

iii. Signature verifying algorithm. This algorithm, when presented with a message, public key and signature, either takes or discards the message's privilege due to authenticity.

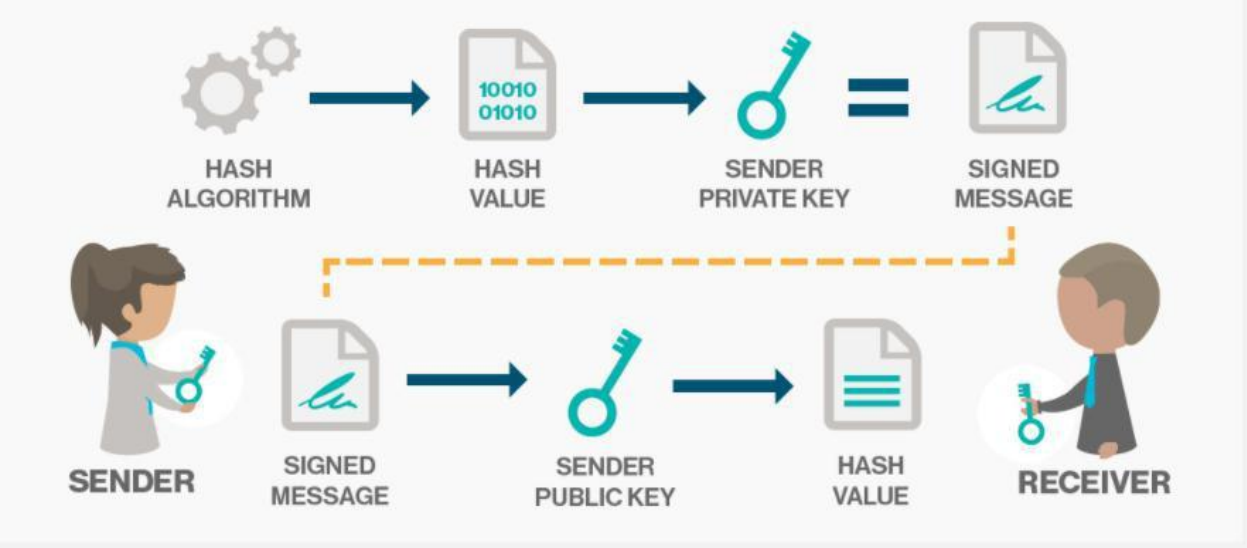


**Figure 2.9 Digital Signature and Verification Process**

## 2.7 Summary

To conclude, every techniques beneficial for real-time encryption and each technique is exclusively independent, which may be appropriate for diverse applications and has its peculiar ace's and rip-off's. Conferring to investigative study done not excluding literature survey, it can be instituted that AES algorithm is most effectual in terms of throughput, avalanche effect, time and speed. The Security offered by the stated algorithms can be improved additionally, when we myriad of the stated algorithms are applied to information and data. Our research work will explore and compare the various Asymmetric techniques such as Elliptic Curve encryption, RSA, SSL and DSA concept and a combination to identify the best encryption method for HTTP data transmission.

KNUST

# CHAPTER THREE

# METHODOLOGY

## 3.0 Introduction

The goal of this chapter is to present the rational and analytical assumptions corroborating this research, as well as introducing the research strategy and the experimental techniques applied. The chapter also shows the scope and limitations of the research design and puts the research amongst existing research in cryptography.

The proposed procedure which is based on the Diffie-Hellman algorithm would be described into detail in this chapter.

At the basis of every public-key cryptosystem is a difficulty that is systematically inflexible. The comparative challenge of resolving this issue regulates the security strength of the system. Table 3.0 reviews the three kinds of popular public-key cryptosystems. The table carefully shows that DSA, Diffie-Hellman and RSA can all be vulnerable to attacks using sub-exponential algorithms, but the popular attack on ECC demands exponential time. Public-key systems are normally used to convey or share keys for symmetric-key ciphers. The work factor needed to crack, a symmetric key should match that needed to break the public-key system used during key exchange since the security of the systems is as robust as its feeblest constituent.

KNUST

**Table 3.1 A comparison of public-key cryptosystems (Vanstone, 2003)**

| Public-key system | Examples | Mathematical Problem | Best known method for solving math problem (running time) |
|---|---|---|---|
| Integer factorization | RSA, Rabin-Williams | Given a number $n$, find its prime factors | Number field sieve: $\exp[1.923(\log n)^{1/3}(\log\log n)^{2/3}]$ (Sub-exponential) |
| Discrete logarithm | Diffie-Hellman (DH), DSA, ElGamal | Given a prime $n$, and numbers $g$ and $h$, find $x$ such that $h = g^x \bmod n$ | Number field sieve: $\exp[1.923(\log n)^{1/3}(\log\log n)^{2/3}]$ (Sub-exponential) |
| Elliptic curve discrete logarithm | ECDH, ECDSA | Given an elliptic curve $E$ and points $P$ and $Q$ on $E$, find $x$ such that $Q = xP$ | Pollard-rho algorithm: $\sqrt{n}$ (Fully exponential) |

In place of exposing the secret agreement directly, the EC DiffieHellmanCng class performs a number of post-processing on the agreement before the value is provided. The post-processing technique also known as the key derivation function; here, you are able to choose which KDF you wish to implement and set its parameters via a set of properties on the instance of the Diffie-Hellman object. Among most of the popular curves employed in ECC are shown in Table 3.2.

**Table 3.2 The Bouncy Castle in C # and Java currently support the following ECDSA curves**

| Curve | Size (in bits) |
|---|---|
| **prime192v1** | 192 |
| **prime192v2** | 192 |
| **prime192v3** | 192 |
| **prime239v1** | 239 |
| **prime239v2** | 239 |
| **prime239v3** | 239 |
| **prime256v1** | 256 |
| | |
| **SEC** | |
| **Curve** | Size (in bits) |
| **secp192k1** | 192 |
| **secp192r1** | 192 |
| **secp224k1** | 224 |
| **secp224r1** | 224 |
| **secp256k1** | 256 |
| **secp256r1** | 256 |

W J SANE NO

| | |
|---|---|
| **secp384r1** | 384 |
| **secp521r1** | 521 |
| | |
| **NIST (aliases for SEC curves)** | |
| **Curve** | Size (in bits) |
| **P-224** | 224 |
| **P-256** | 256 |
| **P-384** | 384 |
| **P-521** | 521 |
| | |
| **X9.62** | |
| **Curve** | Size (in bits) |
| **c2pnb163v1** | 163 |
| **c2pnb163v2** | 163 |
| **c2pnb163v3** | 163 |
| **c2pnb176w1** | 176 |
| **c2tnb191v1** | 191 |

KNUST

| | |
|---|---|
| **c2tnb191v2** | 191 |
| **c2tnb191v3** | 191 |
| **c2pnb208w1** | 208 |
| **c2tnb239v1** | 239 |
| **c2tnb239v2** | 239 |
| **c2tnb239v3** | 239 |
| **c2pnb272w1** | 272 |
| **c2pnb304w1** | 304 |

| **SEC** | |
|---|---|
| **Curve** | Size (in bits) |
| **sect163k1** | 163 |
| **sect163r1** | 163 |
| **sect163r2** | 163 |
| **sect193r1** | 193 |
| **sect193r2** | 193 |
| **sect233k1** | 233 |
| **sect233r1** | 233 |
| **sect239k1** | 239 |
| **sect283k1** | 283 |
| **sect283r1** | 283 |
| **sect409k1** | 409 |
| **sect409r1** | 409 |
| **sect571k1** | 571 |
| **sect571r1** | 571 |

KNUST

**Table 3.3: Equivalent key sizes (in bits).**

| Sym-metric | ECC | RSA/ DH/DSA | MIPS Yrs to attack | Protection lifetime |
|---|---|---|---|---|
| 80 | 160 | 1024 | $10^{12}$ | until 2010 |
| 112 | 224 | 2048 | $10^{24}$ | until 2030 |
| 128 | 256 | 3072 | $10^{28}$ | beyond 2031 |
| 192 | 384 | 7680 | $10^{47}$ | |
| 256 | 512 | 15360 | $10^{66}$ | |

Table 3.3 shows how NIST advises on choosing equal computational symmetric and public-key sizes.

### 3.1 Proposed Procedure

The steps involved in the proposed procedure involves standard techniques such as the Diffie-Hellman key exchange algorithm, elliptic curve properties and computations, and mathematical principles in modular arithmetic and the Euclidean algorithm, mathematics of cryptography etc. All these algorithms and processes are described in details below.

**Modular Arithmetic**

The whole concept of modular arithmetic is similar to that of the analogue or wall clock. For instance, in converting a time from 24-hour format to that of a 12-hour format, one takes the time value in 24-hours and reduces the hour by 12. Example 15:00 becomes 15-12 or preferably 15mod12 which also give 3, hence an equivalence of 3:00 in the 12hours format. The symbol $Z_n$ means integer modulo n and will be used throughout the chapter.

Generally, a positive integer N is selected, the modulus then another positive integer r is also chosen such that $0 \leq r < N$ to be a valid number that can be used in the range, all numbers that are outside this range should be converted or changed to a corresponding one in the range given. For

any two integers a and N, if a is divided by N, the result is an integer quotient q and an integer remainder r that obeys the following relationship

a = q*N + r where $0 \leq r < N$; q = [a/N] and [x] is the largest integer, less than or equal to x. For example

- 19 (mod 7) = 5
- 21 (mod 7) = 7 = 0.
- −18 (mod 7) = 3.

KNUST

**Congruence**

The term congruence refers to a pair of points in a set if a number, a mod n is equal to another number b mod n, such as (a mod n) = (b mod n), that is any points with the same value in modulus n.

For example, the number 1,9,25 are all congruent modulus in mod 8 with a value of 1, since they all are 1 in mod 8.

**Inverse Modulo**

In modulo arithmetic, there is the quest to identify the inverse of a number, there are two types of inverses in modulo arithmetic they are,

- Additive Inverse

  In $Z_n$ two numbers a, b are said to be additive inverses of each other if the modulo of the sum of the two integers is 0 i.e. a + b = 0 (mod n).

, or the modulus number N minus one of the integers a, b gives the other number a, b i.e. N-a = b or N-b =a.

Since modulo arithmetic uses integers primarily, each integer has a corresponding inverse which is unique of that number, also it is worth noting that the additive inverse of a number can also be the number itself. For example the additive inverse pairs in $Z_8$ are (0,0) , (1,7) , (2,6) , (3,5) and (4, 4).

KNUST

- Multiplicative Inverse

  In $Z_n$ two numbers a, b are said to be multiplicative inverses of each other if the modulo of the product of the two integers is 1 i.e. a * b = 1 (mod n). In other words the product of an integer and its multiplicative inverse is congruent or equivalent to 1. Not all integers in a range have a multiplicative inverse in modulo arithmetic.

  For $Z_8$ some available multiplicative inverses are (1, 1), (3, 3), (5, 5), (7, 7). The integer 6 has no multiplicative inverse because we cannot find any number between 0 and 8 which when we multiply will yield a result congruent to 1 in mod 8.

**Euclidean Algorithm**

It is very easy to find the solution to basic modulo arithmetic, which involves primarily positive integers. Many programming languages will generate the answer with ease and as such there would not be any need for most of the concepts discussed earlier, the problem arises when we have to find the mod of complex numbers such as fractions, irrational numbers etc. The technique used to generate such values is derived from the Euclidean algorithm.

The Euclidean algorithm is utilized to identify the highest common divisor of two positive integers, gcd (a, b) = gcd (b, a mod b). This algorithm can be used in computing any equation of the form ab = x (mod N) where there is a solution.

KNUST

- Algorithm used in generating modulus assuming equation is of the form

  ab = x (mod N)

  Generate (a, b, N)

  i. Input a, b, n.
  ii. For all values of N i.e. $0 \leq x < N$;
  iii. Find bx such that bx (mod N ) is congruent modulo of a (mod N)
  iv. If a value x is found, return x
  v. End loop
  vi. If no value for x found return -1.

A return of -1 means no possible value is attained, else the result is returned as x.

- For example with this method the 5/7 (mod 8) is 3, thus 3*7 mod (8) = 5 mod(8)
- -2/5 (mod 8) is 6, thus 5*6 (mod 8) = -2 (mod 8) = 6.

- Algorithm used in generating modulus of equations of the form is similar to the one explained above.

  √a = x (mod N)

  Squaring both sides of the equation will yield an equation of form

a = x*x (mod N), similar to the one solved above

- For example, √3 = 9 (mod 13) since 9*9 mod (13) = 3(mod) 13.

  It is also important to note that not all equations will yield an integer value and as such are not defined in the range. For example, √2 = x (mod 13) is not valid since there is no integer value $0 \leq x < 13$; where x*x (mod 13) = 2 mod 13.

  Also -41/7 mod 91 is invalid since -41 (mod 13) = 50 but there is no integer x, such that $0 \leq x < 13$ multiplied by 7 which gives 50. 7*x (mod 13) $\neq$ 50 (mod 13) for all $0 \leq x < 13$.

  This is shown in the C++ code provided below in Figure 3.0

KNUST

```
#include <iostream>
#include <cstdlib>
#include <cmath>
using namespace std;
int innermodulo(int mod, int value){
    if(value>0)
    return value%mod;
    else{
    while(value<0){
        value = (value)+mod;
    }
    return value;
    }
}
int modulomain(int mod, int num, int denom){
    for(int i=0;i<mod;i++){
        if(innermodulo(mod,num)== innermodulo(mod,i*denom))
        return i;
    }
    return -1;
}
int main() {
    int mod, num,denom;int loop=0;
    do{
    cout<<"Please enter the limit";
    cin>>mod;
    cout<<"Please enter the numerator";
    cin>>num;
    cout<<"Please enter the denominator";
    cin>>denom;
    cout<<modulomain(mod,num,denom);
    }while(loop==0);
   system("pause");
    return 0;
}
```

**Figure 3.0 C++ Code showing the various procedure used for the mathematical models**

## 3.2 Mathematics of Cryptography

### 3.2.1 Rings, Groups, and Fields

Groups, rings, and fields are basic elements in abstract algebra, in considering abstract algebra we consider sets on whose items we can perform some mathematical operations and the result is also a member of the set. The operations and computations are subjected to some rules and principles, which shows the characteristics of that particular set.

### 3.2.2Properties of groups

These are the properties of groups and they are adhered to throughout the process of elliptic curve cryptography, to generate finite fields in which we will operate.

- *Closure*: This property implies that if a, b are members of the set S, then a●b is also a member of the set S.
- *Associative*: this property implies that (a●b) ●c = a● (b●c) for all elements a, b, c in the set S.
- *Identity*: There is an element in S such where a●e = e●a = a for all a in S.
- *Inverse Element*: For each element a in set S, there is an element a" = in S such that a" ●e =e● a" = e.

  In elliptic curve cryptography a special group known as Abelian group is used. An abelian group is one which satisfies an extra condition apart from the ones stated earlier.

- *Commutative*: a●b = b●a for all a, b in S.

A field is defined as a set of elements with two binary operations called addition and multiplication, such that for all elements a, b, c in the field many other axioms are obeyed. A Finite field has a finite number of elements and it is used in this cryptography, to generate values to encrypt and decrypt.

### 3.3 Diffie-Hellman Key Exchange

This algorithm enables two users to securely exchange a key that can then be subsequently used for the encryption of messages. The algorithm itself is limited to the exchange of secret values.

### 3.3.1 Steps Used In the Diffie- Hellman Key Exchange Algorithm

a) User A and User B both agree on a finite field, G = {g} say q.

b) User A and User B also agree on a point within the field α, an integer which is a primitive root of q.

c) User A selects a random integer $X_a$, such that $0 \leq a < q$ and computes $Y_A = \alpha^{Xa} \pmod q$

d) User B also selects a random integer $X_b$, such that $0 \leq b < q$ and computes $Y_b = \alpha^{Xb} \pmod q$

e) Each user a, b keeps their X values $X_a$, $X_b$ private but makes the Y values $Y_A$, $Y_b$ public to the other side.

f) User A and User B can then compute a shared secret from the public keys $Y_A$, $Y_b$ such that shared secret $K = (Y_A)^{Xb} \pmod q = (Y_B)^{Xa} \pmod q$.

g) Although both user A and B keep their X values private, they both obtain the same shared secret K after the computation. They can then use the K for encryption and decryption of messages. This is shown in Figure 3.1 and 3.2.

| Global Public Elements | |
|---|---|
| $q$ | prime number |
| $\alpha$ | $\alpha < q$ *and* $\alpha$ a primitive root of $q$ |

**User A Key Generation**

Select private $X_A$ $\quad X_A < q$

Calculate public $Y_A$ $\quad Y_A = \alpha^{X_A} \bmod q$

**User B Key Generation**

Select private $X_B$ $\quad X_B < q$

Calculate public $Y_B$ $\quad Y_B = \alpha^{X_B} \bmod q$

**Calculation of Secret Key by User A**

$K = (Y_B)^{X_A} \bmod q$

$K = (Y_B)^{X_A} \bmod q$

**Calculation of Secret Key by User B**

$K = (Y_A)^{X_B} \bmod q$

**Figure 3.1 Diffie Hellman Key Exchange Algorithm**

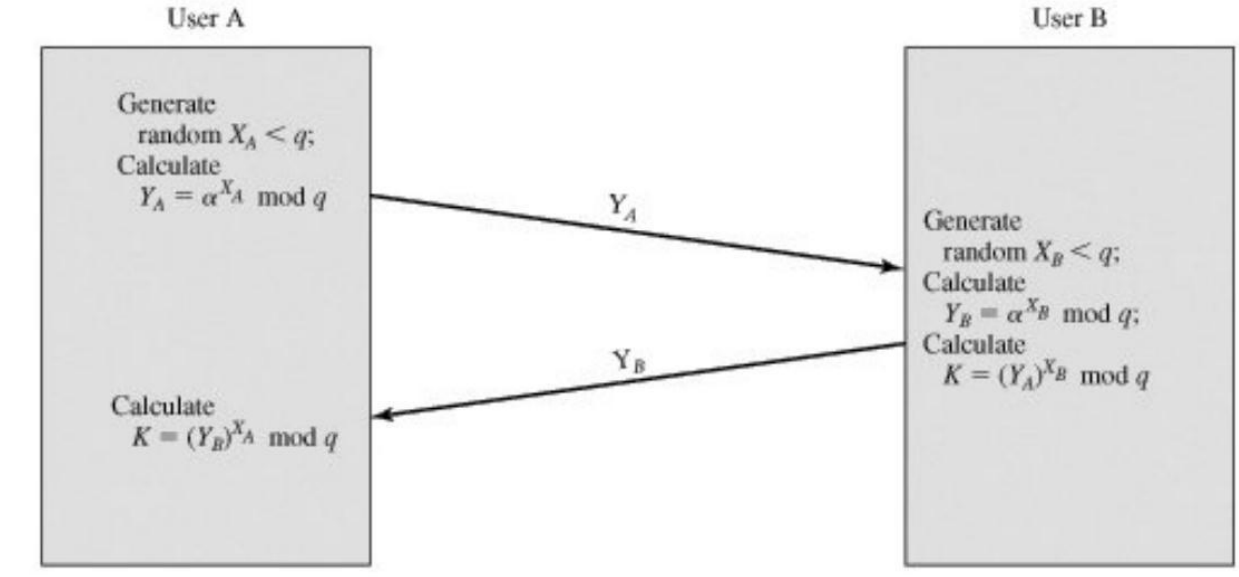


**Figure 3.2 Diffie Hellman Key Exchange Algorithm**

### 3.4 Elliptic Curve Arithmetic

The elliptic curve is a cubic equation in two variables. The general equation of an elliptic curve is $y^{2} + b_1 xy + b_2 y = x^3 + a_1 x^2 + a_2 x + a_3$. Since we are using values generated from the equation for encryption and decryption of data, the variables, coefficients and even values generated must be restricted to only elements in a specified finite field.

KNUST

#### 3.4.1 Elliptic Curve Over Real Numbers

Elliptic curve over real numbers is the easiest case to visualize and it gives us a thorough understanding operations in an elliptic curve. The equation of the form $y^2 = x^3 + ax + b$ generally used. For any given values of a and b, a graph can be plotted, which consists of positive and negative values of y for each value of x.

In cryptography, two groups of elliptic curves are used, prime curves over $Z_p$ or elliptic curves over GF (p) and binary curves over GF ($2^m$).

#### 3.4.2 Elliptic Curve over GF (2m )

For a binary curve defined over GF ($2^m$), the variables and coefficients all take on values in GF ($2^m$) and in calculations are performed over GF ($2^m$).

#### 3.4.3 Elliptic Curve Over GF (p) or $Z_p$

For prime curves over GF (p), we use a cubic equation in which the variables and coefficients all take on values in the set of integers ranging from 0 through p1 and in which calculations are done in modulo p. In this project elliptic curves over GF (p) is used.

### 3.4.4 Operations In Elliptic Curves

Operations in elliptic curves varies from that of normal integers. The operations is the addition of two point on a curve, to get another point which must also be on the curve.

Assume $P(X_1,Y_1)$, $Q(X_2,Y_2)$ and $R(X_3,Y_3)$ if $R = P + Q$ , to find R a point on the curve we consider some cases .

KNUST

**Case 1**

O is the additive identity or identity element as such for any point chosen on the curve P + O = P.

**Case 2**

The negative form of any point on the curve is the point with same x coordinate value but negative of y coordinate value. P + ‘P = PP’ = O.

Also, as stated earlier, the negative of any value is its additive inverse.

Example P(4,5) if the limit of the field is 13 then the inverse of point P(x,y) = ‘P(x,-y) is (4,8).

This is because the additive inverse of the point 5 is 8 in the field.

Also, the points P and ‘P are additive inverse of each other and hence their sum is O.

P + ‘P = PP’ = 0.

As such it is worth noting that if two point on the curve have same x-coordinates and different y-coordinates which are additive inverse of each other, their sum is 0.

**Case 3**

As shown in Figure 3.3a, If two point have different x-coordinates and different y-coordinates, if a straight line is drawn between them a third point is intersected. The third

point which is intersected is the inverse of the sum of the two points, as such finding the inverse of that point gives the sum of the two points.

In finding the point (the third point) the slope of the individual points is found first as

$\lambda = (y_2 - y_1)/ (x_2 - x_1)$,

The third point can then be found as $X = \lambda^2 - x_2 - x_1$,

and $y = \lambda(x_1 - X) - y_1$.

**Case 4**

If two point have same x-coordinates and same y-coordinates, such as P(x,y) then 2P can be found as follows.

$\lambda = (3(x_1)^2+a)/ (2y_1)$, $X = \lambda^2 - x_2 - x_1$, and $y = \lambda(x_1 - X) - y_1$.

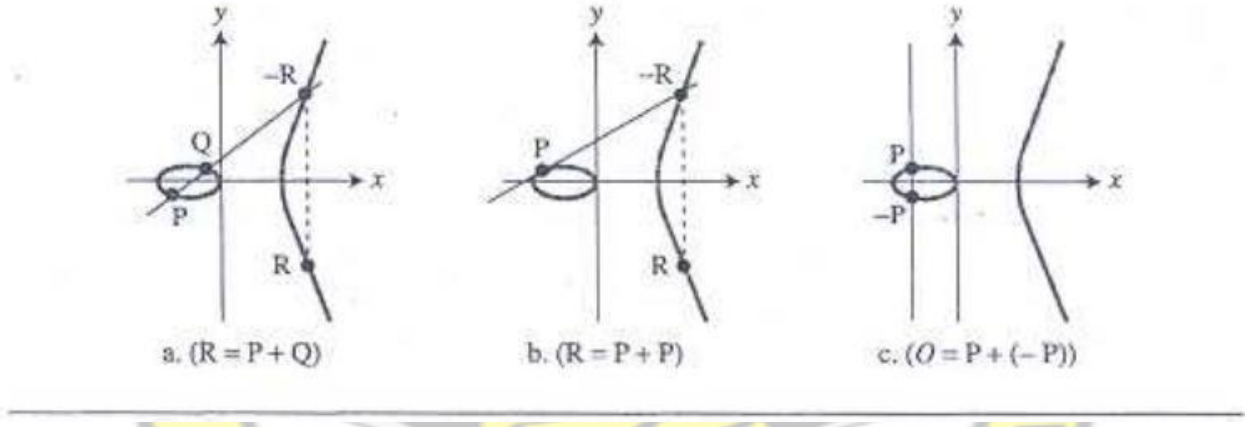


**Figure 3.3 Identities of elliptic curves**

### 3.4.5 Generating Point on the Elliptic Curve

It is very important to generate all valid points on the elliptic curve before beginning the process of elliptic curve cryptography, this is because as stated earlier all valid points selected for any operations must produce another valid point in the selected field. In generating points for the elliptic curve of the form $y^2 = x^3 + ax + b$

P is the range or limit of the field, a and b are constants for the elliptic curve equation

### .4.6 Algorithm to Generate Points on the Elliptic Curve

```
ellipticCurve_generatePoints (p, a, b)          //p is the modulus and limit
{
        x = 0
        while(x< p)
        {
        testpoint = x^3 + ax + b (mod p)
        if (testpoint is a perfect square in mod p){
                output (x, testpoint)           //as point 1
                output (x,'testpoint)           //as point 2
                 // where 'testpoint is the additive inverse of testpoint.
                x = x + 1
        }
}
```

### 3.5 The Steps Involved in the Elliptic Curve Cryptography

The encryption system which is used in this project is the ElGamal, which is built on the Diffie-Hellman Key exchange.This can be defined over any cyclic group. The level of security depends on a certain problem relating to discrete logarithms

The steps used in this encryption process have been explained above, like the Diffie-Hellman key exchange algorithm, modular arithmetic etc.

The only difference between this algorithm and the Diffie-Hellman key exchange algorithm is the coordinates and computations in the elliptic curve.

The encryption and decryption process can be grouped into three main parts which are:

1. **Generating Public and Private Keys**

   The equation for the elliptic curve is $y^2 = x^3 + ax + b$, where a and b are constants that determine the type of equation, some curves are vulnerable to attacks whereas others are highly secured, that is the constants a, b can determine how secured the elliptic curve is, the security of elliptic curve cryptography depends on the difficulty in solving the elliptic curve logarithm problem.

   Both parties A and B agree on the limit or range to work in, as well as the constants a, b these parameters determine an elliptic group E(a, b) to work in.

   With the agreed parameters and limit set, the elliptic group of points can be determined or generated. The algorithm used to generate the points is explained above.

   The parties can then select a point G, in the elliptic group, this point can be made public to both parties and even a third person.

   With the point G selected, both parties can now generate a public key for the other party. This is done by selecting a private key $n_x$, by both parties such that $n_x$ is only known to the party that selected it. The public key is then generated by multiplying the private key with the point G agreed by the two parties.

   For example if party A has to generate a public key $P_A$ for the other party we have $P_A = n_A * G$.

   Both parties can then announce their public keys to the other parties for encryption and decryption of data, but the private key is kept secret.

2. **Encryption**

To encrypt and send any message, $P_m$ from one party to the other, the sender takes the public key of the receiver and multiplies it with his own private key $n_x$, to generate a shared secret between the two parties. This shared secret is not public and only known to the two parties involved.

The shared secret is added to the message to get an encrypted message $P_E$, which can be sent to the other parties.

In sending the encrypted message, a cipher text $C_m = \{P_x, P_S\}$ is sent to the receiver for decryption.

For example if party B has to encrypt a message to person A, he first multiplies his private key $n_B$ with the public key of person A, $P_A$ from the example above to generate a shared secret, $P_s$. The message $P_m$ is then added to the shared secret $P_S$ to get the encrypted message $P_E$, such that $P_E = P_m + n_B * P_A$ . Party B then send the cipher text $C_B = \{P_B, P_S\}$ to party A for decryption.

3. **Decryption**

To decrypt the message the receiving party, takes the public key of the sending party Px and multiplies it with his private key $n_x$ the result is the shared secret which is known to only the two parties A and B. With the secret key the receiver can then subtract it from the encrypted message, to get back the original message Pm.

For example if party A has to decrypt a message from person B, he first multiplies his private key $n_A$ with the public key of person B, $P_B$ to generate a shared secret, $P_S$. The shared secret $P_s$ is then subtracted from encrypted message $P_E$, to get back the original messagesuch that $P_m = P_E - n_A * P_B$ .

It is very difficult for a third part to decrypt the message given the cipher text alone. For instance, for a third-party C to decrypt the message given the cipher text he needs to know the private key $n_x$ of the receiving party to generate the shared secret. This means given the public key of the receiving party and the point G, the third party must find the multiplier that creates public key. The method used for this known as the elliptic curve logarithm problem and the only method available to solve it is the Polard rho algorithm, which is infeasible if the private key or the elliptic group GF (p) is large.

KNUST

### 3.6 Materials and Tools

The Dev C++ IDE was used for the development of the application. C++ was the main programming language used. The application can run on a network of computers with a minimum processor type Pentium IV or better, 10GB Hard drive capacity or better and 1GB RAM or better.

# CHAPTER 4

# DATA ANALYSIS

## 4.0 SPEED BENCHMARKS FOR SOME OF THE MOST COMMONLY USED CRYPTOGRAHIC METHODS OR ALGORITHMS

**Figure 4.0 shows the speed standards for some of the most commonly used cryptographic methods**

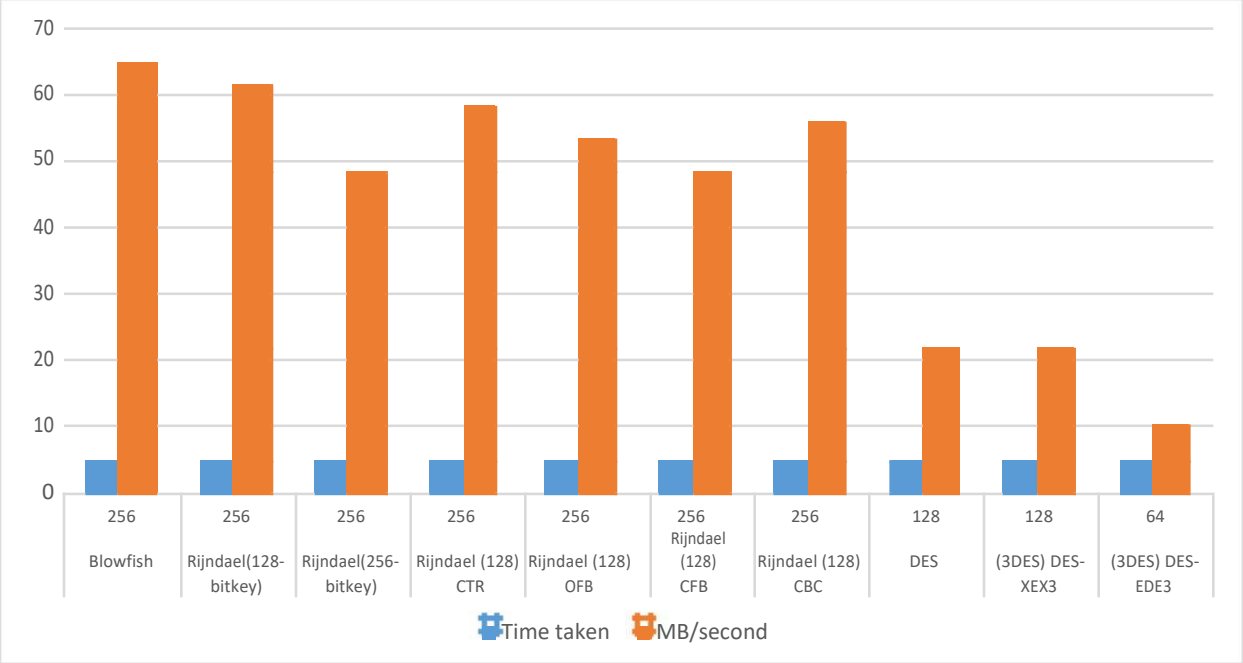


**Figure 4.0 – Speed benchmarks for commonly used cryptographic methods**

From Figure 4.0, it can be realized that not every mode has been tried for all the algorithms. The comparison results show that AES and Blowfish performbetter as compared to the others. DES and 3DES are prone to attacks as compared to AES and Blowfish.

### 4.1 Comparative Analysis of Popular Secret Key Algorithms or Methods by Using Two Different Hardware Platforms

Blowfish, AES, 3DES and DES were executed, and their output was likened by encoding input files of comparable sizes and contents. The algorithms were executed in Java using their typical conditions, and were verified on two separate hardware platforms, to evaluate their performance.

**Comparing the execution times in Seconds of Encryption Algorithms in ECB Mode onA P-Ii 266 MHz Machine. This Is as Shown in Figures 4.1 And 4.2 Respectively.**

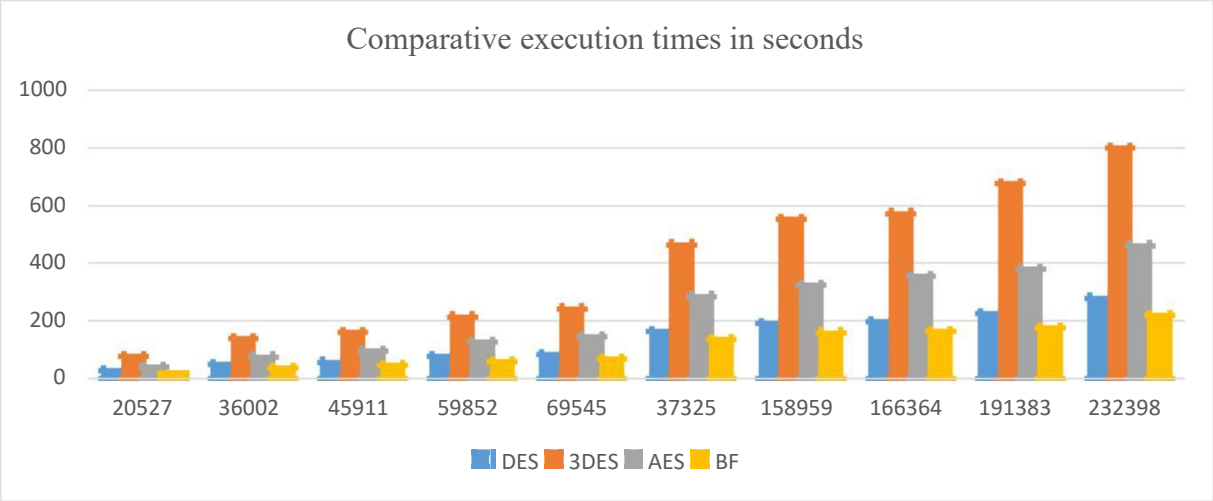


**Figure 4.1 – Comparative execution times of encrypting algorithms**

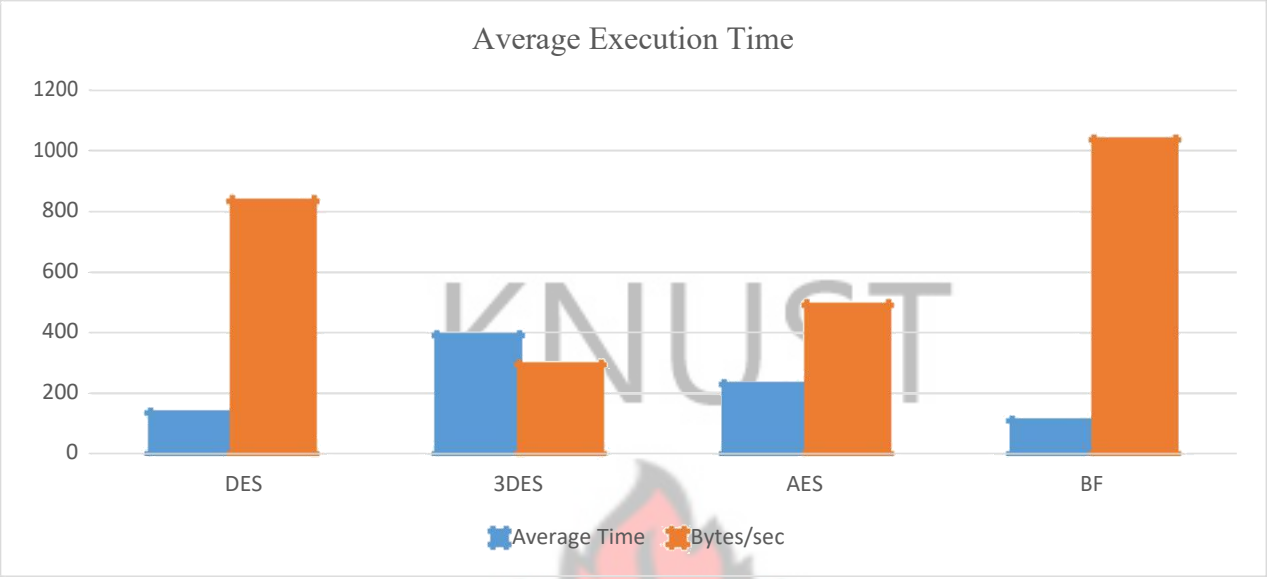


**Figure 4.2 Average Execution time with respect to bytes processed**

**Comparative Execution Times In Seconds Of Encryption Algorithms In Ecb Mode On A P-4 2.4 Ghz Machine. This Is Shown In Figure 4.3 and 4.4 Respectively**

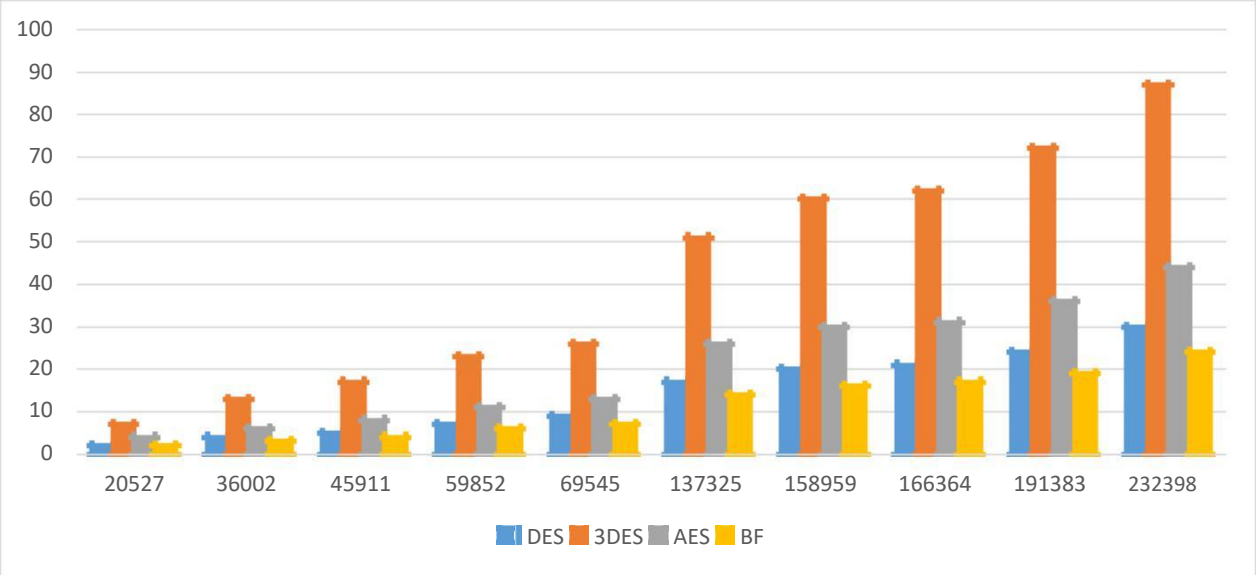


**Figure 4.3 Execution times of encryption algorithms in ECB mode**

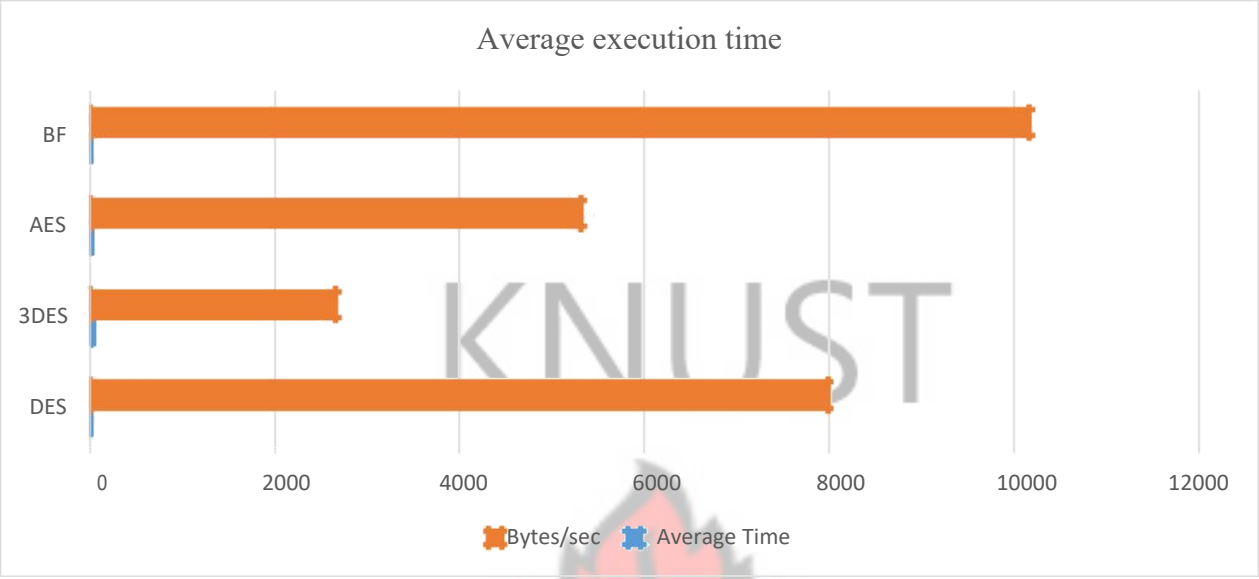


**Figure 4.4 Average Execution time**

From Figure 4.3 and table 4.4, it can be concluded with regards to throughput and performance that Blowfish has an upper hand over other algorithms. It also showed that DES and 3DES had a minimal performance as compared to AES. Astonishingly it depicts that 3DES has nearly one-third the throughput of DES, or it requires three times than DES to generate an equal amount of data.

KNUST

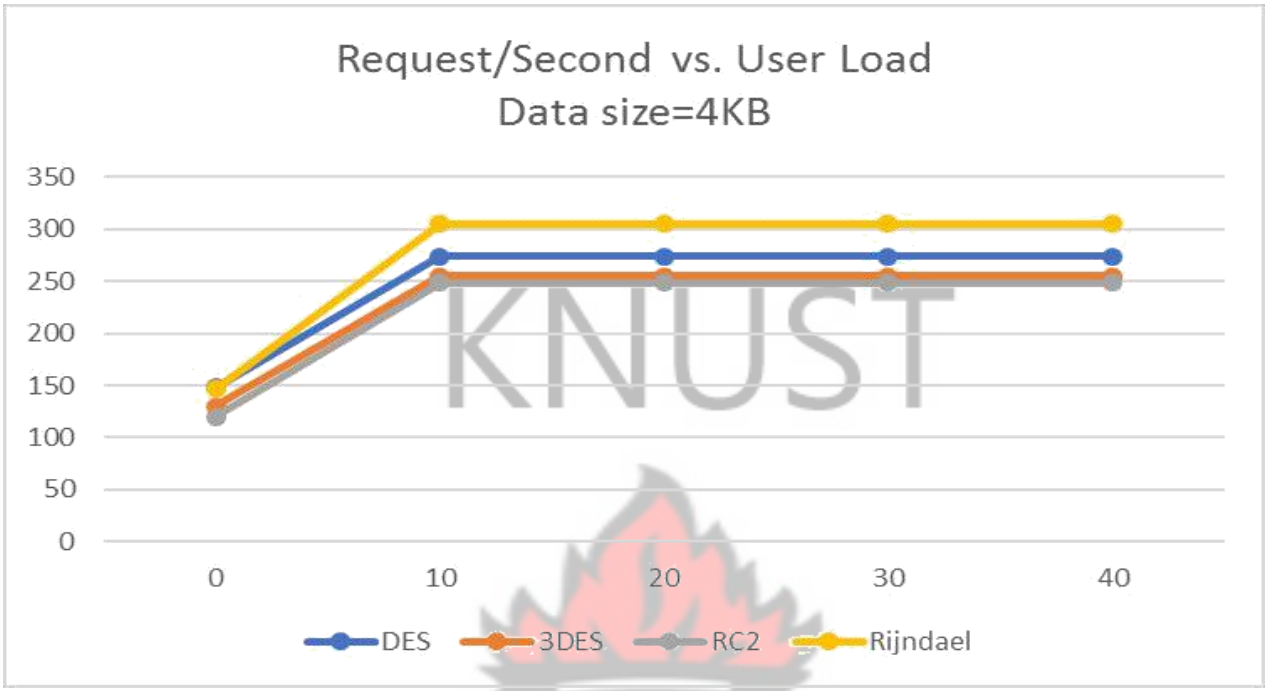


**Figure 4.5: Comparison of the execution time in seconds of encryption algorithm in ECB mode on a P-ii machine with speed of 266Mhz**

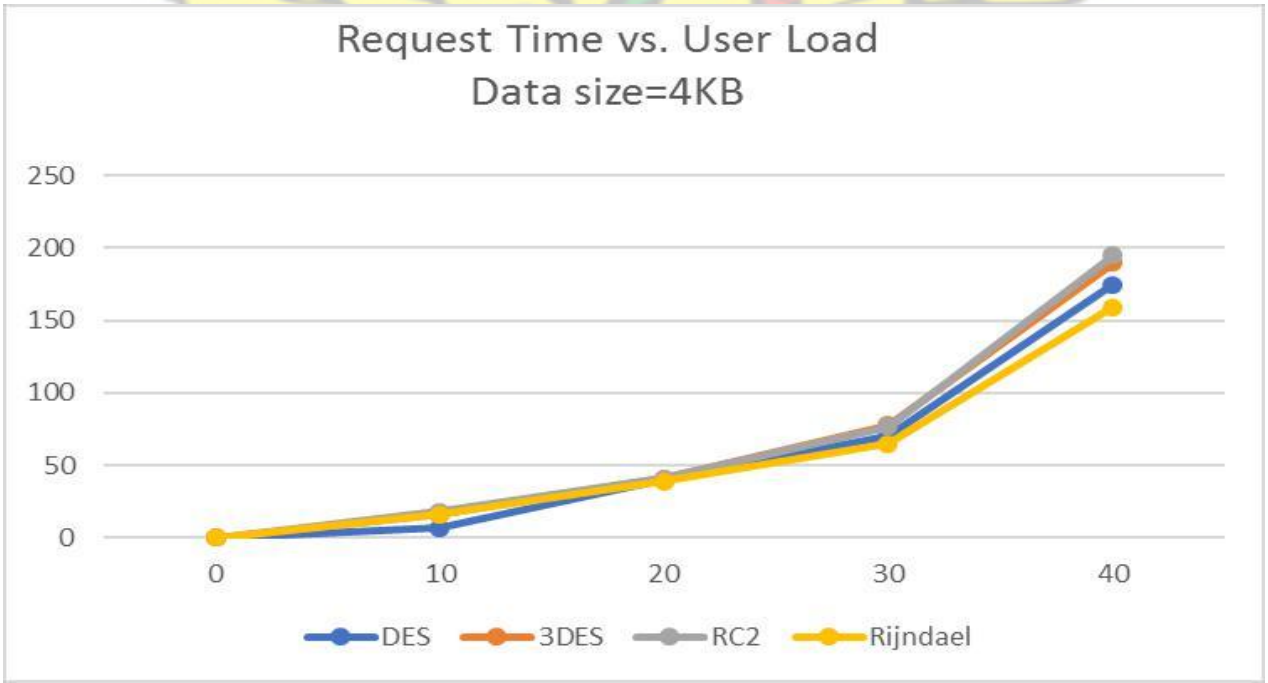


**Figure 4.6: Comparison of the execution time in seconds of encryption algorithm in ECB mode on a P-4 machine with speed of 2.4 GHZ.**

The results showed that AES compared to the other algorithms out performed excelled in both the number of requests executed per second in different user loads, and in the time required to get a response in different user-load situations.

**Table 4.0: Algorithm settings used in the experiment**

| Algorithm | Key Size (Bits) | Block Size (Bits) |
|---|---|---|
| DES | 64 | 64 |
| 3DES | 192 | 64 |
| Rijndael (AES) | 256 | 128 |
| Blowfish | 448 | 64 |

AES and3DES back other algorithm settings, but these algorithm settings signify the maximum-security settings they can portray. Lengthier key lengths mean more effortsare required to disrupt the security of the encrypted data. The load data are separated into the data blocks and they are formed using the Random Number Generator class which is accessible in the system.

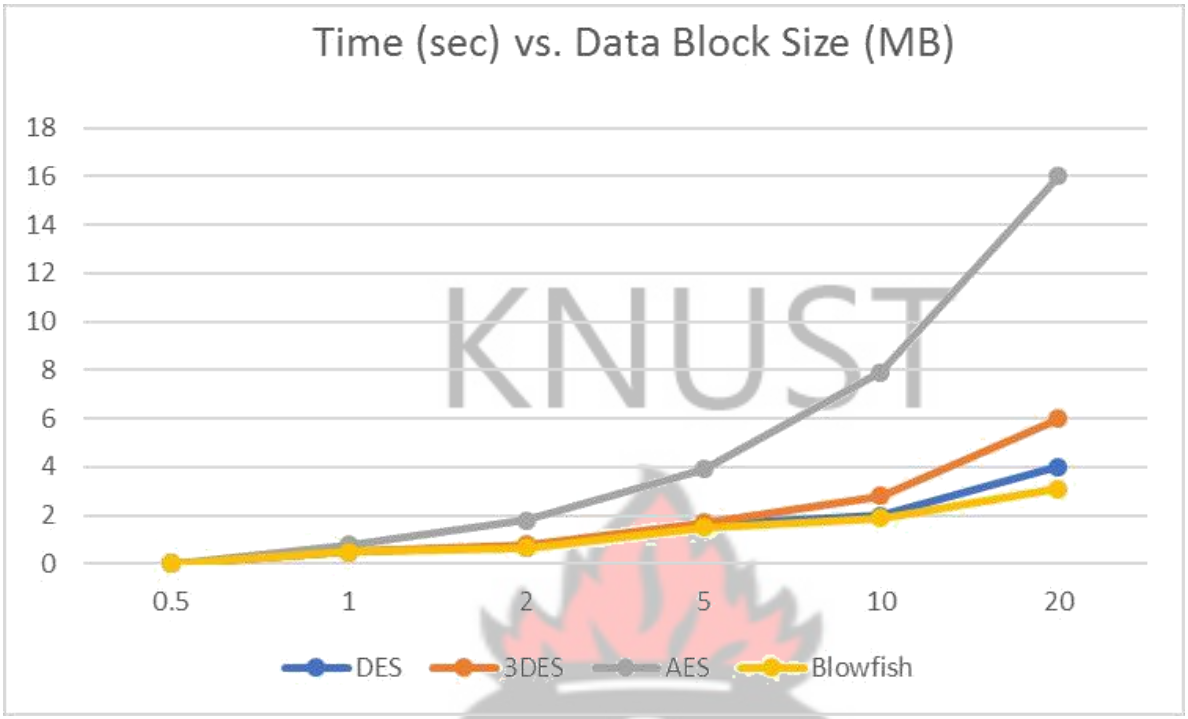


**Figure 4.7: Performance Results with ECB Mode**

## 4.2 Key Length Comparison of RSA and ECC

The length of the key used in encryption has a direct relationship with the computer resources needed and also how fast the encryption or decryption would be. From Table 4.1 and Figure 4.9, it is observed that ECC key length size is very small as compared to RSA providing the same level of security. This shows that using ECC to encrypt data would be faster as compared to RSA.

**Table 4.1 Key Length Comparison of RSA and ECC**

| Security Level (bits) | RSA key length (bits) | ECC key length (bits) | Approximate ratio |
|---|---|---|---|
| 80 | 1024 | 160 – 223 | 5-6:1 |
| 112 | 2048 | 224 – 255 | 8-9:1 |
| 128 | 3072 | 256 – 283 | 11-12:1 |
| 192 | 7680 | 384 – 511 | 15-20:1 |
| 256 | 15360 | 512 – 571 | 27-30:1 |

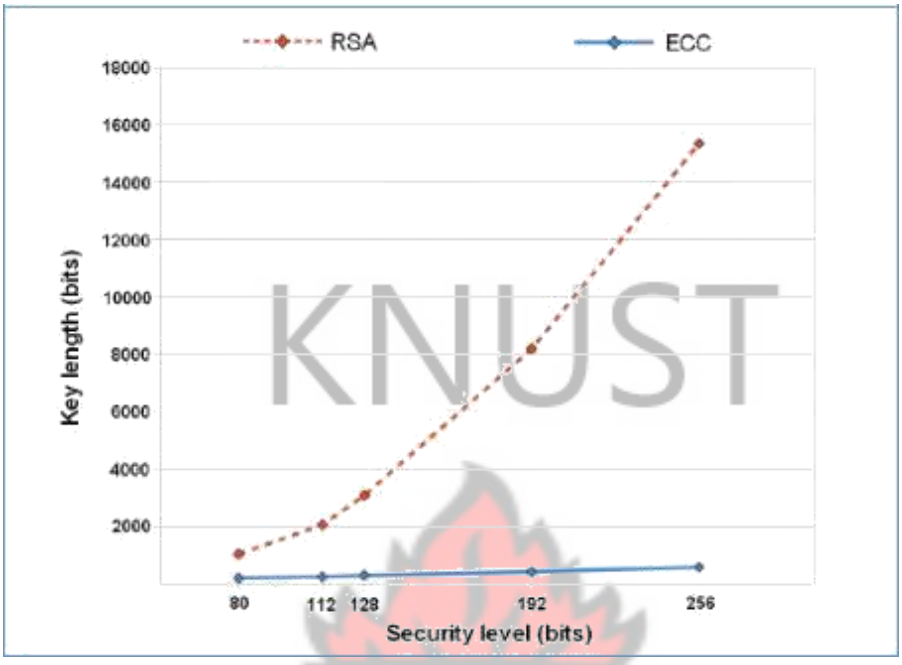


**Figure 4.8 Key Length Comparison for RSA and ECC Cryptosystems**

### 4.3 Performance Evaluation of RSA and ECC

The ECC performs better as show in Figure 4.10 due to its key length being relatively smaller and also ensures much security due to the algorithm used for the encryption.

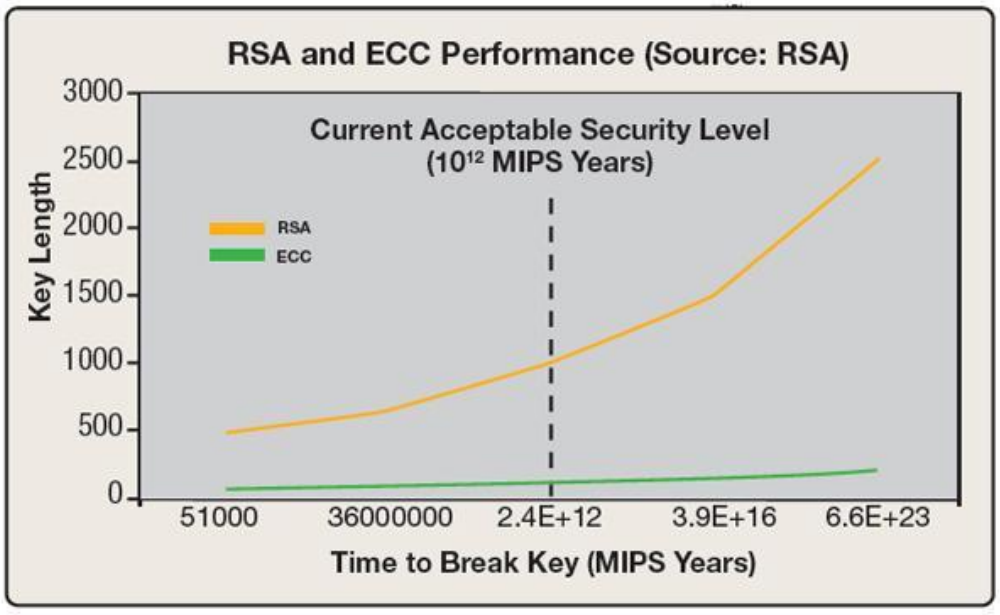


**Figure 4.9 Performance of RSA and ECC**

### 4.4 ECC and RSA Comparison

The various parameters used in verifying the performance are illustrated in the Tables 4.2, 4.3, 4.4 and 4.5. The performance indicators used include Key Generation, Signature Generation and Signature Verification.

KNUST

**Table 4.2 Key Generation Performance**

| Key Length (bits) | | Time (s) | |
|---|---|---|---|
| RSA | ECC | RSA | ECC |
| 1024 | 163 | 0.16 | 0.08 |
| 2240 | 233 | 7.74 | 0.18 |
| 3072 | 283 | 9.80 | 0.27 |
| 7680 | 409 | 133.90 | 0.64 |
| 15360 | 571 | 679.06 | 1.44 |

**Table 4.3 Signature Generation Performance**

| Key Length (bits) | | Time (s) | |
|---|---|---|---|
| RSA | ECC | RSA | ECC |
| 1024 | 163 | 0.01 | 0.15 |
| 2240 | 233 | 0.15 | 0.34 |
| 3072 | 283 | 0.21 | 0.59 |
| 7680 | 409 | 1.53 | 1.18 |
| 15360 | 571 | 9.20 | 3.07 |

**Table 4.4 Signature Verification Performance**

| Key Length (bits) | | Time (s) | |
|---|---|---|---|
| RSA | ECC | RSA | ECC |
| 1024 | 163 | 0.01 | 0.23 |
| 2240 | 233 | 0.01 | 0.51 |
| 3072 | 283 | 0.01 | 0.86 |
| 7680 | 409 | 0.01 | 1.80 |
| 15360 | 571 | 0.03 | 4.53 |

**Table 4.5 ECC and RSA overview**

| **Parameters** | **ECC** | **RSA** |
|---|---|---|
| Computational Overheads | Roughly 10 times that of RSA can be saved | More than ECC |
| Key Sizes | System parameters and key are shorter for the ECC. | System parameters and key pairs are larger for the RSA |
| Bandwidth saving | ECC offers considerable bandwidth savings over RSA | Much less bandwidth saving than ECC |
| Key Generation | Faster | Slower |
| Encryption | Much faster than RSA | At good speed but slower than ECC |
| Decryption | Much Faster than RSA | Faster than ECC |
| Small Devices efficiency | Much more efficient | Less efficient than ECC |

# CHAPTER FIVE

# SUMMARY AND CONCLUSIONS

## 5.1 Introduction

This chapter outlines the summary and conclusions of the project researched into, which seek to achieve the main objectives and solve the problem outlined in the research.

KNUST

## 5.2 Summary of Findings

The study was focused on the comparative analysis of an enhanced elliptic curve cryptography and other asymmetric methods of cryptography used in securing data. A review of the existing encrypting and decrypting algorithms or methods were analyzed. We analyzed the existing Asymmetric and Symmetric Key encryption systems. Systems including the RSA, DES, 3DES, AES and Diffie-Hellman approach. The Elliptic Curve Cryptography takes its basics form the Diffie-Hellman key exchange and improves upon it using the properties of the elliptic curve. The Elliptic Curve Cryptography poses to have much efficacy and efficiency as compared to the other encryption and decryption algorithms. A lot of findings where concluded through the entire research.

The findings of the study revealed the following;

- The Elliptic Curve Cryptography (ECC) uses a shorter key length for encryption and decryption than that of RSA which ensures the optimum and effective use of system resources.
- The time taken for key generation in Elliptic Curve Cryptography (ECC) is shorter as compared to that of Rivest-Shamir-Adleman (RSA) Cryptography.

- The Elliptic Curve Cryptography (ECC) offers better security in comparison the other encryption algorithms due to the algorithm it uses.
- Computational overheads saved using Elliptic Curve Cryptography (ECC) is roughly ten (10) times using Rivest-Shamir-Adleman (RSA) Cryptography.
- Elliptic Curve Cryptography (ECC) offers considerable bandwidth savings
- Elliptic Curve Cryptography (ECC) is efficient even small devices are used.

KNUST

### 5.3 Conclusion

The results provided during the analysis of Elliptic Curve Cryptography (ECC) as against the Rivest-Shamir-Adleman (RSA) and the other types of encryption algorithms, it is seen that, ECC is much more efficient than the other encryption algorithms. Elliptic Curve Cryptography has a shorter key length and size which is an important security parameter. It also saves bandwidth which facilitates key generation in the encryption and decryption of data hence enhancing performance. Moreover, ECC ensure faster encryption and decryption, and efficient even in small devices. From all the explicitly stated results, it can be stated without an iota of doubt that Elliptic Curve Cryptography is a better option to use in securing data.

## REFERENCES


*3DES*. (n.d.). Retrieved February 18, 2017, from http://www.cryptosys.net/3des.html

Alanazi, H. O., Zaidan, B. B., Zaidan, A. A., Jalab, H. A., Shabbir, M., & Al-Nabhani, Y. (MARCH 2010). New Comparative Study Between DES, 3DES and AES within Nine Factors. *Journal Of Computing, 2*(3), 152-157.

Apostolopoulos, G., Peris, V., & Saha, D. (1999). Transport Layer Security: How much does it really cost? *In Proc. of IEEE Infocom*.

Arora, P., Singh, A., & Tiyagi, H. (2012). Evaluation and Comparison of Security Issues on Cloud Computing Environment. *World of Computer Science and Information Technology Journal (WCSIT), 2*(5), 179-183.

Badia, L. (2001, September). *Rain bow Technologies Whitepaper* . Retrieved January 10, 2017, from Real World SSL Benchmarking: http://www.rainbow.com/insights/whitePDF/RealWorldSSLBenchmarking.pdf

Daemen, J., & Rijmen, V. (2003). AES Proposal: Rijndael. *National Institute of Standards and Technology., 21*, 1.

*DES* . (n.d.). Retrieved 2 18, 2017, from http://www.tropsoft.com/strongenc/des.htm

Elminaam, Abdul, D. S., Kader, Abdul, H. M., Hadhoud, & Mohamed, M. (December, 2008). Performance Evaluation of Symmetric Encryption Algorithms. *IJCSNS International Journal of Computer Science and Network Security, 8* (12), 280-286.

Federal Information. (2001). *Announcing the ADVANCED ENCRYPTION STANDARD (AES), 197.* Federal Information Department.

Fielding, R. (1999). *Hypertext Transfer Protocol – HTTP/1.1.* RFC 2616.

Forouzan, B. A. (2007). *Cryptography and Network Security.* New York: Tata McGraw-Hill Publishing Company.

Frier, A., Karlton, P., & Kocher, P. (n.d.). The SSL3.0 Protocol Version 3.0. Retrieved from http://home.netscape.com

Goldberg A.; Buff R.; Schmitt A. (1998). Secure Web Server Performance Dramatically Improved by Caching SSL Session Keys. In *In Proc. of Workshop on Internet Server Performance.* SIGMETRICS.

Haiyong Xie; Li Zhou; Laxmi Bhuyan. (n.d.). *"Architectural Analysis of Cryptographic Applications for Network Processors.* Riverside: Department of Computer Science & Engineering, University of California.

*Https Everywhere*. (n.d.). (Electronic Frontier foundation) Retrieved January 24, 2017, from https://www.eff.org/https-everywhere/faq

IBM. (2016, Septamber). *IBM Knowledge Center*. (IBM) Retrieved November 2016, from http://www.ibm.com/support/knowledgecenter/en/SSFKSJ_9.0.0/com.ibm.mq.sec.doc/q009800_.htm

Kakkar, A., Singh, M. L., & Bansal, P. (January 2012). Comparison of Various Encryption Algorithms and Techniques for Secured Data Communication in Multinode Network. *International Journal of Engineering and Technology, 2*(1), 87-92.

Kumar, A., Jakhar, D. S., & Makkar, M. S. (2012). Comparative Analysis between DES and RSA Algorithm's. *International Journal of Advanced Research in Computer Science and Software Engineering, 2*(7), 386-391.

Laboratories, R. (2012). RC6 Block Ciphe. *Historical: RSA Algorithm: Recent Results on OAEP* .

Mandal, A. K., Parakash, C., & Tiwari, M. A. (2012). Performance Evaluation of Cryptographic Algorithms: DES and AES. *IEEE Students' Conference on Electrical, Electronics and Computer Science*, 1-5.

Meghna, V. (2014). Comparative Analysis of Cryptographic Algorithms and Advanced Cryptographic Algorithms. *International Journal of Computer Engineering and Science*, 1.

Pavithra, S., & Ramadevi, M. E. (2012). Performance Evaluation of Symmetric Algorithms. *Journal of Global Research in Computer Science, 3*(8), 43-45.

Qi, N., Wei, W., Zhang, J., Wang, W., Zhao, J., Li, J., . . . Hu, J. (2013). Analysis and Research of the RSA Algorithm. *Analysis and Research of the RSA Algorithm*, 1818-1824.

Rouse, M. (2015). *TechTarget*. Retrieved November 5, 2016, from http://searchsoftwarequality.techtarget.com/definition/HTTPS

S, G. J. (n.d.). Symmetric and Asymmetric Encryption . *Computer Survey, 11*, 4.

Schwartz, J. (2000). *U.S. Selects a New Encryption Technique.* New York: New York Times.

Seth, S. M., & R. I. (2011). Comparative Analysis of Encryption Algorithms for Data Communication. *International Journal of Computer Science and Technology, 2*(2), 292-294.

Singh, G., & Supriya. (April 2013). A Study of Encryption Algorithms (RSA, DES, 3DES and AES) for Information Security. *International Journal of Computer Applications, 67*(19), 0975 – 8887.

Stallings, W. (2005). *Data and Computer Communications.* 6eWilliam.

Stallings, W. (2005). *Cryptography and Network Security Principles and Practices.* Prentice Hall.

Team, A. (2013, March 11). *Atmel Bit and Pieces*. (Atmel) Retrieved December 20, 2016, from http://blog.atmel.com/2013/03/11/symmetric-vs-asymmetric-encryption-which-way-is-better/

university, c. (n.d.). Asymmetric Encryption. In *Encryption.*

Vanstone, S. A. (2003). Next generation security for wireless elliptic curve cryptography. *Computers and Security, 22*(5).

Wagner, D., & Schneier, B. (1996). Analysis of the SSL 3.0 protocol. *2nd USENIX Workshop on Electronic*.

Ylonen, T., Kivinen, T., Saarinen, M., Rinne, T., & Lehtinen, S. (2003). *SSH Protocol Architecture.*

KNUST