بسم الله الرحمن الرحيم

Palestine Polytechnic University
Deanship of Graduate Studies and Scientific Research
Master of Informatics

Exploring and Adapting AES Algorithm for Optimal Use as a Lightweight IoT Crypto Algorithm

Submitted By

Mohammed AbuJoodeh

Supervisors

Dr. Liana Al Tamimi
Dr. Radwan Tahboub

A Thesis submitted in partial fulfillment of requirements for the Master Degree in Informatics

Jun. 2022

DECLARATION

I declare that the Master Thesis entitled "**Exploring and Adapting AES Algorithm for Optimal Use as a Lightweight IoT Crypto Algorithm**" is my original work, and hereby certify that unless stated, all work contained within this thesis is my independent research and has not been submitted for the award of any other degree at any institution, except where due acknowledgment is made in the text.

Mohammed AbuJoodeh

Signature: _____ Date: _____

The undersigned hereby certify that they have read, examined, and recommended to the Deanship of Graduate Studies and Scientific Research at Palestine Polytechnic University the approval of a thesis entitled: **Exploring and Adapting AES Algorithm for Optimal Use as a Lightweight IoT Crypto Algorithm**, submitted by Mohammed Rashad Ahmad AbuJoodeh in partial fulfillment of the requirements for the degree of Master in Informatics.

Graduate Advisory Committee:

Dr. Liana Tamimi (**Supervisor**), Palestine Polytechnic University.

Signature:———————————— Date:————————————

Dr. Radwan Tahboub (**co-Supervisor**), Palestine Polytechnic University.

Signature:———————————— Date:————————————

Dr. Mohammed Abutaha (**Internal committee member**), Palestine Polytechnic University.

Signature:———————————— Date:————————————

Dr. Ahmad AlSa'deh (**External committee member**), Birzeit University

Signature:———————————— Date:————————————

Thesis Approved

Signature:———————————— Date:————————————

Exploring and Adapting AES Algorithm for Optimal Use as a Lightweight IoT Crypto Algorithm

Jun. 2022

DEDICATION

سلاماً على من صان عهد الشهداء وأقسم على الوفاء...

ولأننا أبناء العظيمة فلسطين، ولأننا نحبها نيابةً عن كل الذين باعوها

فإنني أهدي هذا العمل المتواضع إلى الأيادي القابضة على الزناد في كل شبرٍ من هذه الأرض...

إلى أرواح الشهداء الأشرف منّا جميعا...
إلى من غيبتهم غياهب المعتقلات ولم يغيبوا

إلى فلسطين كل فلسطين...

من بحرها إلى نهرها... ومن غزتها، إلى قدسها

كانت كلها واحدةً تُسمى فلسطين... وستبقى.

Exploring and Adapting AES Algorithm for Optimal Use as a Lightweight IoT Crypto Algorithm

Jun. 2022

ACKNOWLEDGMENT

I would like to express my gratitude to everyone who helped me during this work. First of all, my supervisors Dr. Liana Tamimi and Dr. Radwan Tahboub deserve my endless thanks for they have spared no effort in encouraging me to take on this challenge. They provided me with valuable information and guided me the whole time. Thank you for the continuous support and the kind communication that has made me so excited about what we are working on.

Many friends were supportive models; You have shared many precious and unforgettable moments. Thank you very much to them and for their support.

Last but not least, I owe a huge thank you to my family members for their support and encouragement throughout my life. Without their help and support, I would not have been able to complete my studies and achieve this success.

ACRONYMS

ACRONYM	FULL PHRASE
IoT	*Internet of Things*
LWC	*Lightweight Cryptography*
CPC	*Clock Per Cycle*
Kb	*Kilo bit*
RSA	*Rivest–Shamir–Adleman*
DH	*Diffie-Helman*
ECC	*Elliptic Curve Cryptography*
DSA	*Digital Signature Algorithm*
RC4	*Rivest Cipher 4*
DES	*Data Encryption Standard*
AES	*Advanced Encryption Standard*
ECB	*Electronic Code Book*
CBC	*Cipher Block Chaining*
CFB	*Cipher Feedback Mode*
OFB	*Output Feedback Mode*
CTR	*Counter Mode*
LFSR	*Left Feedback Shift Register*
NLSC	*New Lightweight Stream Cipher*
NFCR	*Nonlinear Feedback Shift Registers*
SPN	*Substitution-Permutation Network*
NIST	*National Institution of Standards and Technology*
UACI	*Unified Averaged Changed Intensity*
NPCR	*Number of Pixel Change Rate*
RP	*Raspberry Pi*
10H	*10-Rounds AES with Half MixColumns*
10N	*10-Rounds AES without MixColumns*
5H	*5-Rounds AES with Half MixColumns*
5N	*5-Rounds AES without MixColumns*
5F	*5-Rounds AES with MixColumns*
3F	*Rounds AES with MixColumns*
2F	*2-Rounds AES with MixColumns*
NLW-AES	*New Lightweight AES*

ABSTRACT

ABSTRACT

Cyber and Information security are among the most critical challenges facing nowadays technologies, especially with the discrepancy in devices' capabilities that have increased with the emergence of the Internet of Things (IoT) devices. The main problem in IoT security is how to find lightweight cryptosystems that are suitable for devices with limited capabilities. In this thesis, a comprehensive literature survey that discusses the most prominent encryption algorithms used in device security in general and IoT devices in specific has been conducted. Many studies related to this field have been discussed to identify the most important requirements of lightweight encryption systems to be compatible with variances in IoT devices. Also, we explored the results of security and performance of the AES algorithm by changing it's core parameters and functions including the MixColumns and the number of rounds. In case of changing MixColumns, the results showed the security has been adversely affected. While changing in the number of rounds provides a promising result to improve the algorithm performance while keeping an acceptable security level which makes it more adaptable to IoT devices. In general, the results showed that running three rounds of standard AES maintains the same level of security practically under specific criteria with a 386% improvement in performance indicators. Accordingly, we proposed a New Lightweight AES (NLW-AES) which maintain the standard AES-MixColumns with three rounds of AES. Finally, we tested both the Standard AES and the New Lightweight AES on Raspberry Pi as an IoT model which provide compatible results with the explored scenarios.

Keywords: IoT Security, Networks, Cryptography, AES, Lightweight cryptography.

ABSTRACT- AR

الملخص

يعد الأمن السيبراني وأمن المعلومات من بين أهم التحديات التي تواجه التقنيات في الوقت الحاضر، لا سيما مع التباين في قدرات الأجهزة التي زادت مع ظهور أجهزة إنترنت الأشياء. تكمن المشكلة الرئيسية في أمان إنترنت الأشياء في كيفية العثور على نظام أمان خفيف الوزن مناسب للأجهزة ذات القدرات المحدودة. نسعى في هذا البحث إلى مراجعة مجموعة من الدراسات السابقة التي ناقشت أبرز خوارزميات التشفير المستخدمة في أمن الأجهزة بشكل عام وأجهزة إنترنت الأشياء بشكل خاص. في هذه الدراسة، تمت دراسة العديد من الدراسات في هذا المجال لتحديد أبرز متطلبات أنظمة التشفير المخفف لتكون متوافقة مع الاختلافات في أجهزة إنترنت الأشياء.

في هذا البحث أيضا قمنا بتنفيذ واختبار كل من مستوى الأمان ومستوى الأداء لخوارزمية AES والتي كانت نتائج واعدة وشكلت نقطة انطلاق لهذه الدراسة بهدف تحسينها وجعلها أكثر قابلية للتكيف مع أجهزة إنترنت الأشياء. مما دفعنا الى دراسة التغير في النتائج عند تغيير بعض المكونات الرئيسية للخوارزمية المتمثلة في اقتران MixColumns وعدد الجولات.

أظهرت النتائج أن التغيير في اقتران MixColumns قد أدى إلى انحدار ملحوظ في مستوى أمان الخوارزمية، بينما التغيير في عدد الجولات قد قدم نتائج واعدة لامكانية تعديلها وتطويرها بما يتلائم مع قدرات أجهزة انترنت الأشياء.

بشكل عام، أظهرت النتائج أن تشغيل ثلاث جولات من خوارزمية AES يحافظ على نفس المستوى من الأمان مع تحسن بنسبة 386٪ في الأداء. بناءً على هذه النتائج، قمنا باقتراح خوارزمية مخففة A New Lightweight AES والتي حافظت على استخدام AES-MixColumns كما هو مع استخدام ثلاث جولات من AES. أخيراً، اختبرنا كلاً من خوارزمية AES والتعديل المقترح على Raspberry Pi كنموذج تطبيقي على إنترنت الأشياء والذي أظهر توافقا مع النتائج التي تم الوصول إليها خلال اختبار النماذج.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1 INTRODUCTION

This chapter introduces the direction of our work in this thesis. In *Section 1.1*, we introduce this work and give a general overview of the thesis. Our problem statement, discussed in *section 1.2.* Finally, we summarize the contributions of this study in *Section 1.3*.

1.1 OVERVIEW

An information system is a set of interconnected components that collect, process, store and transfer information. These components include the physical and software components, and the communication networks used [1].

Networks are based on connecting many devices and enabling communication among them in the best possible way. Moreover, networks are subject to many attacks due to many users and their different directions. The real challenge is maintaining the security of these networks with their resources and data while maintaining good performance results [1-3].

In its simplest sense, ***Internet of Things (IoT)*** is a system of various smart devices known in our daily lives. These things link and communicate between them and ensure the transfer of data between them independently via the networks without human interaction, which means that it is a self-control system [4-6]. *Smart Cities* played an essential role in highlighting IoT. Smart Cities express the concept that depends on the city's technology, as these cities are linked to each other electronically. Information is collected continuously from sensors, monitoring, and computers covering the whole city [5-6].***"Thing"*** *term in IoT;* represent any device that can take an IP address and able to interact through a network [5].

The application's ***security*** plays an essential role in judging the application strength. Any user wishes to have the software request that is secure in all respects. The application's security includes a secure transfer of data, protection from eavesdropping, and unauthorized access. So, that system security has become one of the essential critical requirements of the system's core functions [2][3]. Furthermore, the security aims to achieve what is known as *the Confidentiality, Integrity, and Availability (CIA)* triad. Finally, one of the most critical security goals is to control access through the *Authentication, Authorization, and Accounting (AAA)* framework [3][7].

1.2 PROBLEM STATEMENT

IoT has caused a massive increase in the volume of data. Securing this enormous amount of data requires hard work. Several technologies serve this purpose. But the devices used in the IoT vary among themselves in capabilities. Moreover, most of these devices have limited specifications and restrictions on use [5-6]. Hence the need to find new technologies that work on these variant capabilities and achieve an acceptable degree of security and performance. Furthermore, since the capabilities are limited, these technologies should be lightweight and rely on simple operations without consuming energy, storage, and processing capacity. Therefore, in this study, we look forward to finding a lightweight crypto algorithm that guarantees security and performance requirements and is compatible with IoT devices.

1.3 THESIS METHODOLOGY

The thesis methodology is as follows:

- Defining the problem statement clearly, and clarify the background terms in details.
- Surveying the related works in Lightweight Cryptography (LWC) and AES-based lightweight algorithms (*to be published*).
- Studying the security and performance of Advanced Standard Algorithm (AES).
- Exploring the performance and security through implementing different AES Scenarios.
- Proposing a new Lightweight AES based on the results of the explored scenarios.

1.4 CONTRIBUTION

In this study, our main contributions are:

- Surveying state-of-the-art literature review in the field of LWC for IoT and AES based researches.
- Comparing various techniques recently suggested to secure IoT and discuss their effectiveness in applications based on security and performance tests.
- Implementing and testing AES in term of security and performance. Furthermore, exploring many AES scenarios based on its core functions including changing the number of rounds and modifying MixColumns operation.
- Evaluating the explored scenarios in term of security and performance results.
- Implementing and testing the best scenario on Raspberry Pi (RP), which is a model for IoT device.

1.5 DOCUMENT ORGANIZATION

The rest of the thesis is organized as follows:

- **Chapter 2**: Provides clarifications for some sciences in this field.
- **Chapter 3**: Presents in detail some of the related studies carried out in the field of LWC.
- **Chapter 4**: Presents the evaluation mechanisms.
- **Chapter 5**: Evaluates the performance and security of AES.
- **Chapter 6**: Explores AES algorithm and evaluate each scenario.
- **Chapter 7**: introduce the new lightweight AES and its Results.
- **Chapter 8**: Concludes the thesis and clarify the outline for future work.

CHAPTER 2

BACKGROUND

2 BACKGROUND

This chapter introduces the concepts of IoT, Cryptography, and LWC in sections 2.1, 2.2, and 2.3 respectively.

2.1 INTERNET OF THINGS

IoT today is a hot topic in research. The importance of IoT comes because of keeping pace with the variables of life that call us to exploit everything new in technology, such as computers, cars, TV, refrigerators, and washing machines [5-6]. Figure 2.1 shows IoT Reference Architecture. The figure shows that the IoT system consists of data collector's devices as a sensor used to get the data and data analyzer device like a mobile phone used for data processing to make a decision. These two subsystems communicate and transfer data via a network [5-6][8].

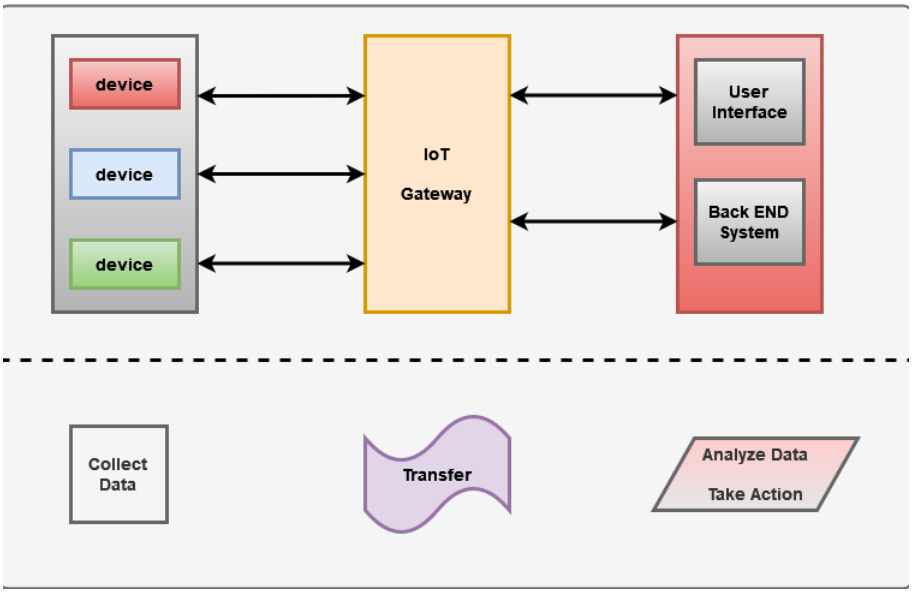


Figure 2.1. IoT System [8]

IoT has dramatically helped to increase the efficiency of work and operations. Since it relies on a system of self-interaction, that means reducing the waiting time for response. As a result, performance gains, and therefore the number of completed processes increases, giving users access to the best possible user services, enhancing the work's actual value [5][8]. In general, IoT provides a wide range of benefits at the enterprise and individual levels. *Table 2.1.1* present these benefits.

Table 2.1.1. IoT Benefits

Enterprise-level	***Human- level***
Monitor overall processes.	Safe life.
Enhance experiences.	Simple and beautiful life.
Reduce time and cost.	Easy access to information.
Increase productivity.	
Model and devices integration.	
Increase profits.	

Figure 2.1 presents the concept of IoT. There are many useful and valuable applications for IoT, such as Safe Houses, Health Care, and Farming systems.

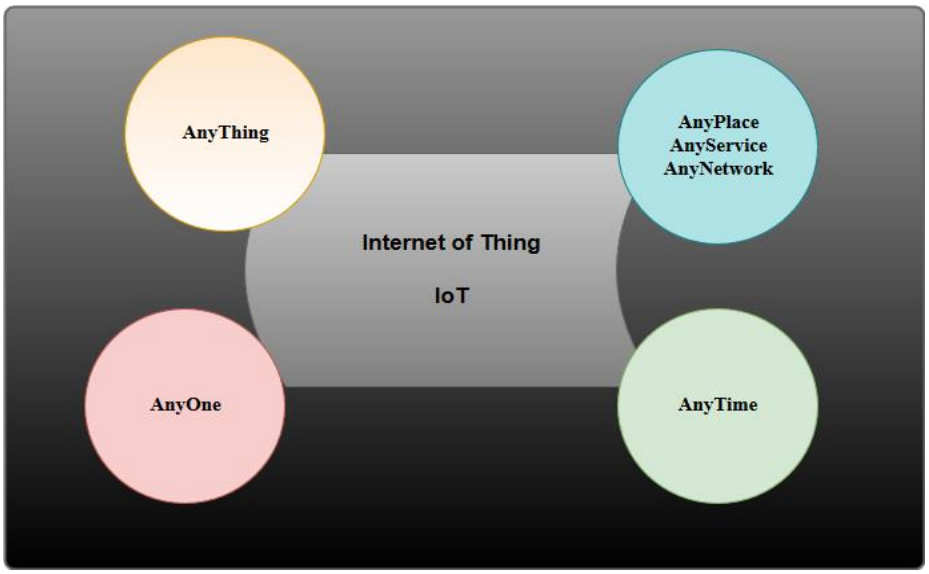


Figure 2.2. Concept of IoT [4]

Despite the significant benefits of IoT, the IoT suffers from a lack of standardization and is vulnerable to cyber-attacks, data theft, data fraud, botnet attacks, and physical compromises. The reason for this is that the IoT differs from traditional networks. There are two types of IoT devices: those rich in resources, like computers, and those with limited resources, like sensors. The real challenges are in the second type, which has low memory and computing power, short battery life, and Low bandwidth to connect [6][8]. So, we should be careful about security and privacy [8]. Hence, the challenge is how to design an IoT system efficiently and securely.

2.2 CRYPTOGRAPHY

Cryptography is a way to protect data and communications through coding operations to ensure that their reading and processing are withheld from others who are not authorized to access them [3][9]. The goals of the encryption process revolve around guaranteeing each of the following [3] [9-10]:

- ***Confidentiality****:* Using encryption to protect data from unauthorized access.
- ***Data Integrity:*** Ensures that the message remains the same as sent without changing it by using a unique message digest.
- ***Non-Repudiation****:* Ensures that the recipient does not deny the message's arrival by proving that the sender sent the message. Also, it prevents the sender from denying.
- ***Authentication:*** Proof of an entity identity, which confirms the user's right to access the system or data.
- ***Access Control:*** Ensures that access to the system or data is limited by preventing unauthorized access and checking their privileges.

Still, there some essential terms related to security worlds; they are:

- ***CIA Triad****:* In addition to confidentiality and integrity, we still have the concept of availability, which ensures that authorized users can access what they want at any time. Therefore, the CIA triad tries to achieve the three goals that have been emphasized [9].
- ***AAA Framework****:* is responsible for enforcing policies and controlling access over resources. In addition to the authentication previously mentioned, it ensures that the security methods used in the network guarantee [7][11]:
  - ***Authorization***: Not much different from access control. It works on the resources the user is allowed to access and used.
  - ***Accounting:*** Directly, it can be defined as a complete monitoring process and writing down all the operations that the user performs to be used further in the accounting, analysis, and planning process.

Figure 2.3 summarizes CIA triad and AAA framework.

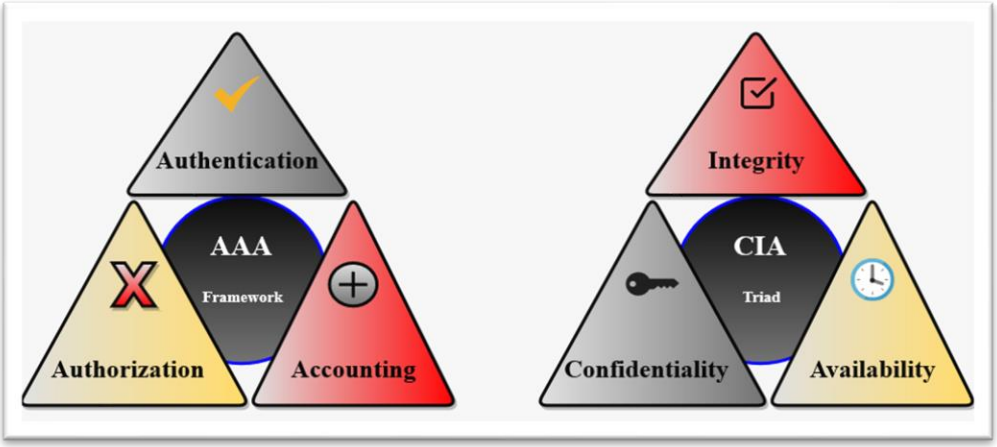


Figure 2.3. CIA Triad and AAA Framework

2.2.1 CRYPTOGRAPHY ALGORITHMS

In cryptography science, ***Encryption*** is transforming original messages (Plaintext) to non-readable data (Cipher Text) using an encryption algorithm. This Cipher Text cannot give anyone any information about the Plaintext except those with the encryption key [9, 12-17]. Therefore, we can perform a simple encryption example by replacing every character in the plaintext with its next character in aliphatic order.

$$P = \text{“}Thesis\text{”}$$
$$Alg.: substitution \qquad Pi = P_{i+1} \qquad (1)$$
$$C = \text{“}uiftjt\text{”}$$

There are two main types of encryptions: Asymmetric cipher, and Symmetric cipher, as shown in Figure 2.4 [9, 12-17].

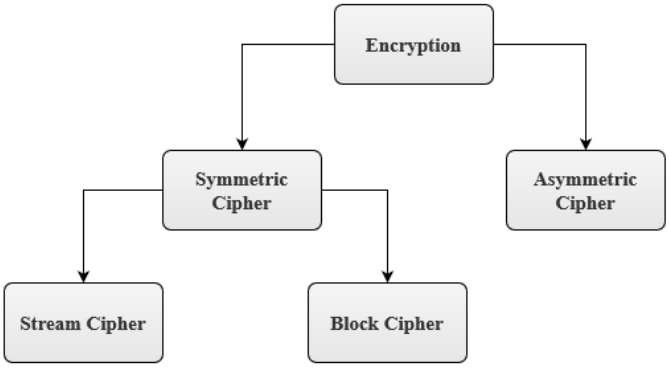


Figure 2.4. Encryption Models [9]

2.2.1.1 ASYMMETRIC CIPHER

Asymmetric cipher is conjointly referred to as public-key cryptography; the associate cryptography technique uses a mix of public and private keys. The sender has the receiver's public key, whereas the private key is not known. The receiver ought to produce his try of the general public and private key, publish his public key while not considering its security. The private key should be a procedure not possible to seek out through the general public key.

Asymmetric cryptography is employed in authentication and digital signatures. A signed message with the sender's private key proves the sender's identity, and it is scanned by anyone who has the sender's public key. Thus, the receiver sure that the message has not been changed or replaced by the other one that confirms the sender's identity [6, 12-13].

2.2.1.1.1 RIVEST SHAMIR ADLEMAN ALGORITHM

Rivest–Shamir–Adleman (RSA) algorithm is one of the most popular and widely used asymmetric encryption algorithms. It was developed in 1977 by Ron Rivest, Adi Shamir, Leonard Adleman and took its name from them. Besides encryption, RSA can be used for key exchange and digital signature [17-18].

RSA gained its strength by relying on the difficulty of parsing large integers in the formation of keys. Two prime numbers are manipulated to create the user's public and private keys. The message is encrypted using the recipient's public key and decrypted exclusively with the recipient's private key. Figure 2.5 shows the RSA Process [17].

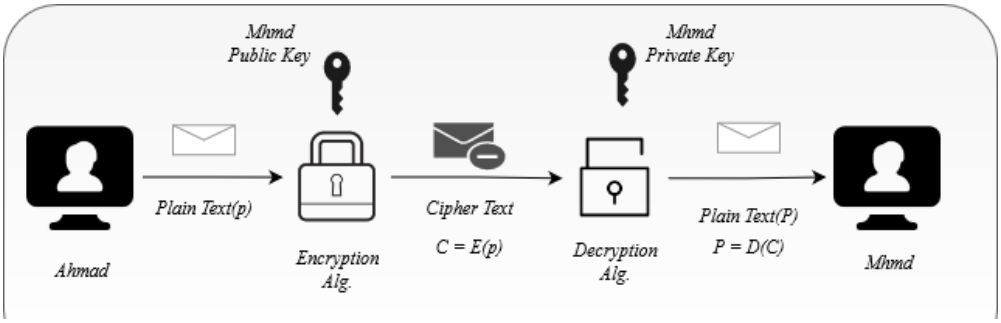


Figure 2.5. RSA Process [17]

Although RSA is one of the most popular and secure asymmetric encryption algorithms in terms of key difficulty, it takes a long time to encrypt and decrypt. Besides, a security flaw appears that encrypting the same message again produces the same encrypted message [18].

2.2.1.1.2 ELGAMAL

ElGamal is an asymmetric cipher based on Diffie–Hellman (DH) key exchange. This algorithm gains its strength through the difficulty of finding discrete logarithms. For example, even though we know G^x and G^y, it is challenging to find G^{xy}. This algorithm consists of the key generation process, encryption, and decryption process. Figure 2.6 shows each of them [19-20].

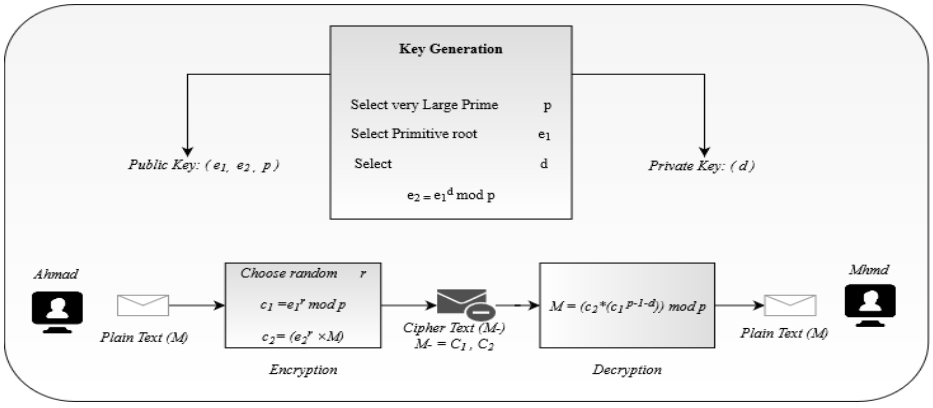


Figure 2.6. ElGamal Alg. [20]

2.2.1.1.3 DIFFIE-HELLMAN

DH is a protocol used to exchange keys for asymmetric cryptography, which establishes a connection between two parties to establish a mutual secret session for them. This protocol also provides the basis for a wide range of validated protocols [21]. Figure 2.7 shows the DH Process.

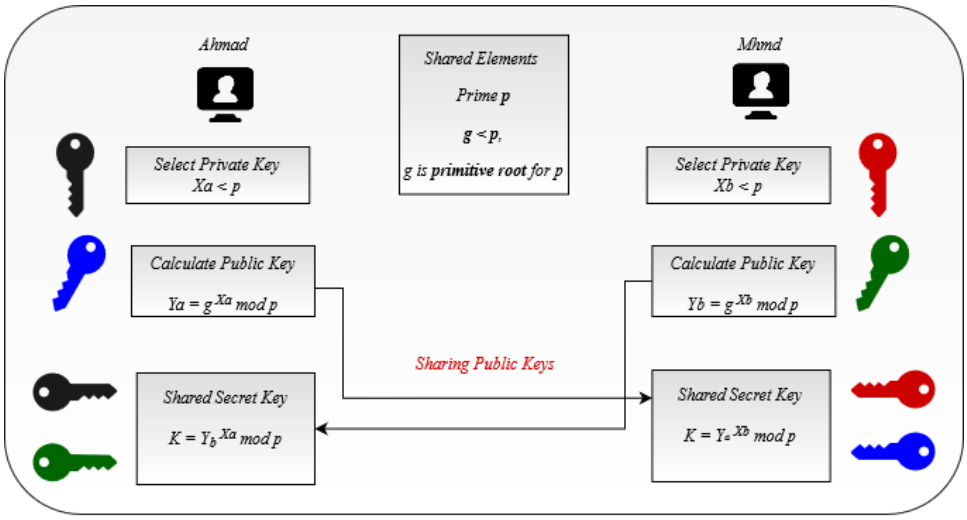


Figure 2.7. DH Keys [21]

2.2.1.1.4 ELLIPTIC-CURVE CRYPTOGRAPHY

Elliptic Curve Cryptography (ECC) it uses the mathematics on elliptic curves. ECC is widely used due to its high security and small size. The difficulty in cracking the elliptic curves that underpin key strength has made ECC more secured and considered as the next generation of RSA [22].

The main difference between ECC and RSA is the strength of the key. A 160-bit key in ECC is equivalent in power to a 1,024-bit key in RSA. Considering that there is no linear relationship, doubling the size of the RSA key does not mean that we need to double the size of the RSA key. ECC is characterized by the speed of obtaining the keys and less memory to store them. On the other hand, a challenge for ECC is that it cannot be implemented as efficiently as RSA [22]. Figure 2.8 presents the ECC basics.

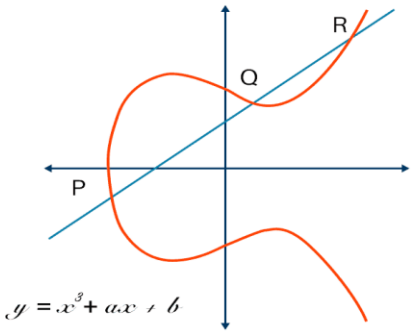


Figure 2.8. ECC Basics [22]

2.2.1.1.5 DIGITAL SIGNATURE ALGORITHM

Digital Signature Algorithm (DSA) was developed to use the discrete logarithm and standard bases to introduce and validate the concept of a digital signature. DSA is characterized by faster key generation compared to RSA. As a result, it is slower in the encryption process, but it offers better results in the decryption process. DSA is mainly used to verify the identity of the sender of a message since it bears his signature, which cannot be duplicated [23]. *Figure 2.9* presents the DSA mechanism.

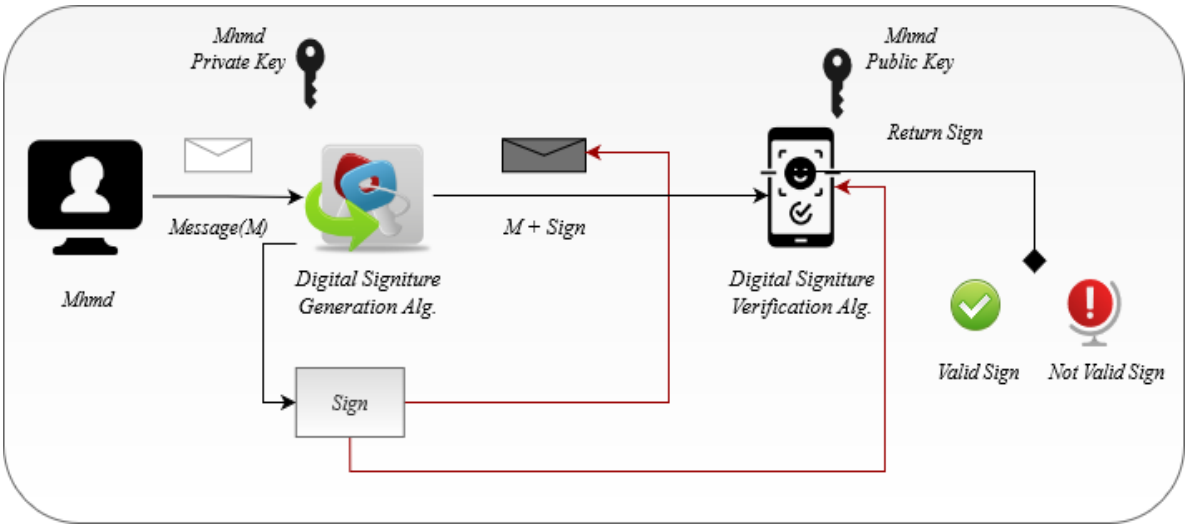


Figure 2.9. DSA Process [23]

2.2.1.1.6 ASYMMETRIC CIPHER SUMMARY

This section discussed various Asymmetric Cipher algorithms such as RSA, ElGamal, DH, DSA, and ECC. *Table 2.2.1* highlight the most comparison points between them.

Table 2.2.1. Asymmetric Cipher Comparison

Cipher	Key size *(bits)*				Strength	Weakness
RSA	1024	2048	3072	4096	• Low computational time. • Fast.	• Use same module for multi users. • For small messages. • Not Scalable.
ElGamal	1024				• Fast. • Very efficient in hardware imp. • Solve discrete logarithm. • Good Scalability. • Low Power Consumption.	• Require Random Number Generator. • Ciphertext is very Large. • Slow in Signing.
DH	1024	3072			• Solve discrete logarithm. • Sharing keys, not information.	• Expensive exponential operations. • Lack of Authentication.
DSA	512 – 1024 ***(multiple of 64)***				• Authentication. • Integrity. • Non-repudiation.	• Entropy. • Secrecy. • Uniqueness of random signature.
ECC	160	224	256		• Small Key size. • Low storage. • Low transmission time, and power consumption. • Very Fast.	• Ciphertext is large. • High Complexity

Although this type of algorithm offers high strength in terms of security, it requires a large amount of processing, which means low performance and draining resources. Therefore, based on the preceding, these algorithms are not compatible with the discrepancy in the capabilities of IoT devices and therefore cannot be used in building security systems in term of encryption. Hence, we find that symmetric encryption is more suitable for such systems. However, this does not detract from its value, as it cannot be dispensed with in verification, key exchange, and signature operations.

2.2.1.2 SYMMETRIC CIPHER

Each sender and receiver share the same secret key in this kind of encryption. It uses within the encryption and decryption processes. However, symmetric encryption has better speed but

provides a lower security level than asymmetric [9, 12, 16, 24]. Also, Symmetric ciphers can be used as a block cipher or stream cipher. *Figure 2.10* shows the general structure of this encryption model.

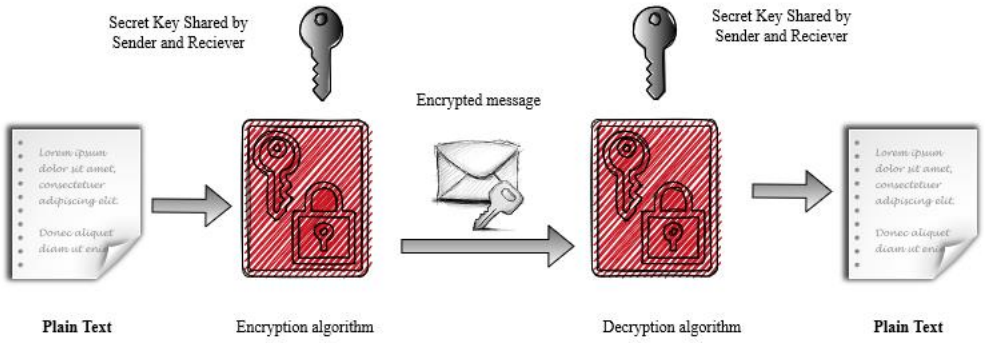


Figure 2.10. Simple Symmetric model [9]

2.2.1.2.1 STREAM CIPHER

In this type of encryption, the data are encrypted bit by bit. That's means every encrypted bit is independent of other bits, so diffusion and confusion properties are not achieved in this type. This encryption mainly uses as simple as possible operators. In most cases, it uses the XOR operation between the plaintext bits and the corresponding key bits. As a result, stream cipher throughput (encryption speed) is much *higher* than the block cipher but is considered less secure than Block Cipher [9, 12-16,24].

A. RIVEST CIPHER 4

Rivest Cipher 4 (RC4) is a stream cipher algorithm proposed by Ron Rivest in 1987. It later became a widely used algorithm from being a personal algorithm due to its speed and simplicity. As a result, RC4 has been frequently used to encrypt network traffic [25].

This algorithm uses byte-oriented operations with a variable key size. Simply put, RC4 relies on an XOR operation between each piece of plaintext with a small portion of the key to produce the ciphertext. And the decoding process is only a reflection of this process. However, with the development of computers, it became possible to break this algorithm

easily. However, RC4 can be considered secure if the initial bytes of the key are deprecated [25-27].

B. *SALSA20*

Salsa20 is a synchronous stream cipher suggested by Bernstein. The number 20 indicates the number of rounds, but this can be reduced to 12 or 8 as needed. *Salsa20* reliance on simple operations such as rotation, addition, and XOR makes it a high-speed algorithm, which makes it secure against timing attacks [27-28].

C. *SOSEMANUK*

Sosemanuk is a synchronous stream cipher with variable key length. It has good properties of confusion and diffusion for a low cost. Mux operation makes it secure against algebraic attacks and fast correlation attacks. Finally, Sosemanuk has good performance due to the internal static data [29].

D. *STREAM CIPHER SUMMARY*

After we have discussed them, and after reviewing the definition and specifications of each. *Table2.2.2* provides a brief comparison of these algorithms.

Table 2.2.2. Stream Cipher Comparison

Stream Cipher	**Key size** $_{bits}$		**Data size** $_{bits}$	**Rounds**	**Speed** $_{CPC}$
RC4	1 – 2048		2046	1	7
SALSA20	128	256	512	20	3.91
SOSEMANUK	128 – 256		32	20 - 32	5.6

From this comparison, we note that the RC4 algorithm is optimal for use, as it is more robust and available in more than one version to suit the system requirements. Since the stream cipher provides high speed but with a low level of security and requires the key size to be the same as the plaintext size, we cannot recommend any of these algorithms to be a basis for building a new system for the IoT.

2.2.1.2.2 BLOCK CIPHER

In Block Cipher, the plaintext is divided into blocks based on encryption algorithm structure [12]. This type of encryption has an execution time slower than the stream cipher. So, the encryption throughput of stream cipher is much higher than the block cipher [9][23]. In contrast, a block cipher provides better security than the stream cipher against some well-known attacks such as the Reused key attack and the Bit-flipping attack. Moreover, the essential properties of the secure ciphertext, which are the confusion and the diffusion properties, are included inside block ciphering algorithms. Based on these facts, we can nominate one block cipher algorithm to build our algorithm for the IoT after reviewing it and choosing the most appropriate based on its specification and results [9][12].

A. DATA ENCRYPTION STANDARD

Data Encryption Standard (DES) is a symmetric encryption algorithm that uses a seemingly 64-bit key, of which 56 bits are used as the effective key over 16 rounds of the 48-bit subkeys, to encrypt data of a fixed length of 64 bits. The apparent key's remaining 8 bits are utilized to verify for parity. In decryption, the same process is employed in reverse [30-31]. *Figure 2.11* shows the structure of DES encryption.

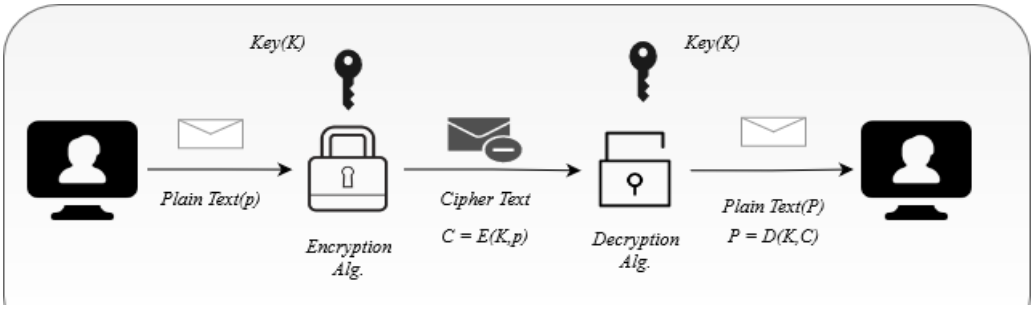


Figure 2.11. DES Algorithm [30]

Despite the spread of this algorithm due to its encryption speed and ease of application, it suffers from a noticeable security weakness in reality. The use of DES for a short key makes it very fragile, especially using a brute force attack, which is easy to use in this case, and there are many attacks such as offensive Linear and differential cryptanalysis, which are theoretical attacks, are called demonstrative vulnerabilities [30-31].

B. TRIPLE DES

An improved version of the encryption algorithm has been created to solve the security issues with DES. This method is as simple as applying the DES algorithm precisely three times. We now have three keys, each of which is 56 bits long. As a result, the implementation technique differed in terms of the keys utilized. Because the relationship of the three keys to each other affects the extent of the algorithm's power in the previously described, there were several versions. ***Triple DES (3DES)***, which used three distinct keys with a total of 156 actual bits, was thought to be very powerful [28]. However, 3DES will not be used at the end of 2023 as we move to more secure generations for encryption [32].

C. BLOWFISH

Blowfish is a symmetric cipher technique that uses a 64-bit block and a variable-length encryption key as needed. In terms of speed, Blowfish is a good algorithm, but the amount of security it provides varies depending on the length of the key employed. As a result, even though there have been no genuine threats detected, it has gotten less attention compared to other algorithms [33-34].

D. ADVANCED ENCRYPTION STANDARD

Advanced Encryption Standard (AES) is one of the most famous and prominent symmetric encryption algorithms that has been introduced to be a quantum leap in this field. AES has outstanding performance and an excellent security level compared to its peers. AES deals with data blocks with a fixed size of 128 bits in length, in addition to providing flexibility in choosing the size of the key according to the required degree of security. From here, it appears that AES has three versions according to the size of the key, namely AES-128, AES-192, and AES-256 with 10, 12, and 14 rounds, respectively. Each cycle uses several operations to encrypt a data block [34-35].

The working mechanism of AES is based on the use of the design principle known as the permutation and substitution network, and this mechanism is represented by using the following operations:

- ***SubBytes:*** Using a predefined lookup table known as Rijndael S-Box, each byte will be replaced with another one. Without any linear relation.

- ***ShiftRows:*** The row elements are swapped by shifting them cyclically to the left.

- ***MixColumns:*** Using a linear transformation relationship, the change of all column elements is combined so that they affect each other to increase the level of difficulty through the propagation property.

- ***AddRoundKey:*** The data cells are combined with the subkey cells generated for this round using XOR operation.

The need for key expansion is that each AES round needs a key of a specific length based on the criteria mentioned earlier. Therefore, when using AES-128, we need 11 keys depending on the number of rounds. Key derivation is done using the AES Key Schedule algorithm, which expands the key using a key schedule [33-35]. *Figure 2.12* represents the flow of the AES algorithm.

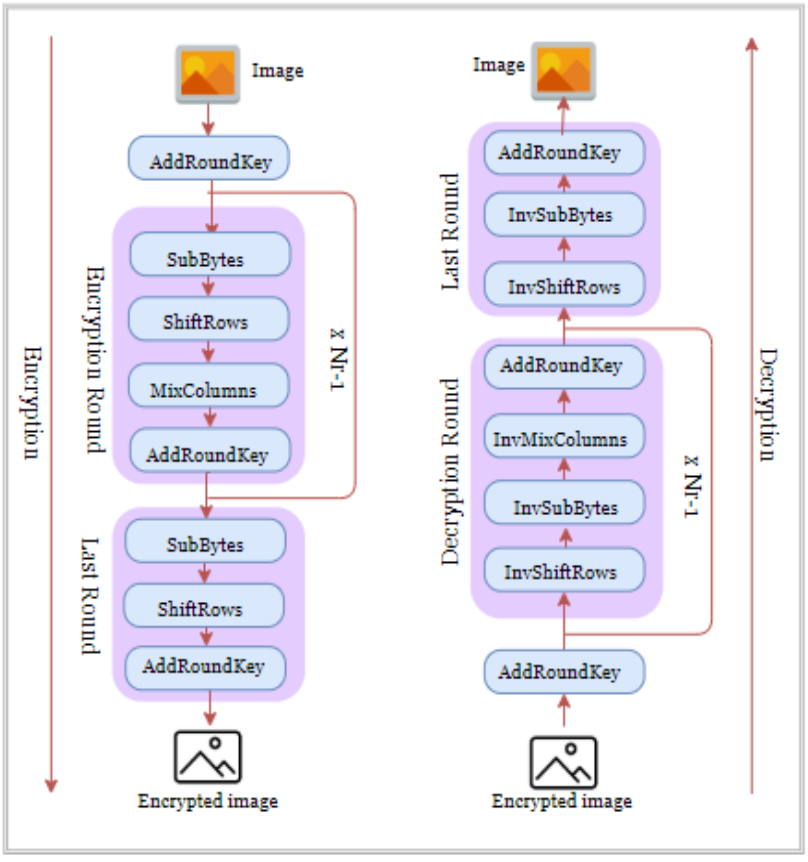


Figure 2.12. AES Flow [31]

AES distinguished itself from its peers in improving its performance for systems dealing with large amounts of data by integrating these steps and running them on a byte-oriented approach. This approach only converts its arithmetic operations into some lookup tables [35].

E. MODES OF OPERATION

In Block Cipher, a fixed size block is handled at a time. Usually, the data size is much larger than the block size. Hence, the data is divided into a set of blocks. Each block is encrypted as one unit, the relationship and dependency between encrypted blocks relaying on the encryption mode. Several modes have been developed to suit the type of target application. The process of selecting the required mode depends on many factors such as error propagation, the level of security, preprocessing, parallelization, and the speed of encryption and decryption [12][33]. These modes are as follows:

- ***Electronic Code Book (ECB):*** It is an explicit and imperative coding process. It is considered the simplest since the text is split and each block is encrypted independently [36].

- ***Cipher Block Chaining (CBC):*** This mode has constituted a development from the ECB. The block encryption process has become dependent on the result of the previous block encryption, which increased the data dependency on each other and made it possible non-deterministic. In this mode, the plaintext XOR-ed with the result of the previous block encryption before the encryption process [36].

- ***Cipher Feedback (CFB):*** Looking at the CBC mode, this mode also relies on the result of the previous block as feedback to the present block and some other variables to increase the resistance to attacks [36].

- ***Output Feedback (OFB):*** There is no difference between it and CFB except in some minor details that increased the resistance to bit errors and reduced the relationship of encryption to the plaintext [36].

- ***Counter (CTR):*** It is a counter-based CFB. This mode is mainly based on maintaining the synchronization of the counter between the sender and receiver [36].

Without going into the details of these modes. *Table 2.2.3* compares these modes.

Table 2.2.3. Encryption Modes Comparison

Mode	Padding Required	Error Propagation	Parallel		Pre-Comp	Speed$_{(5)}$		Security	As Stream
			Enc	Dec		Enc	Dec		
ECB	Yes	No	Yes	Yes	No	5	2	Low	No
CBC	Yes	All next Blocks	No	Yes	No	2	1	High	No
CFB	No	One block	No	Yes	No	1	4	High	Yes
OFB	No	No	No	No	Key stream	3	3	High	Yes
CTR	No	No	Yes	Yes	Yes	4	5	Medium	Yes

The term (Speed$_{(5)}$) in sixth column pointing to the level of mode speed from 1 to 5. From these results we found that CBC, CFB, and OFB provide better security. CBC and CFB outperform in case of Decryption parallelization and provide higher Error Propagation which lead to better confusion and diffusion.

F. BLOCK CIPHER SUMMARY

After discussing the previous block cipher algorithms such as DES, 3DES, Blowfish, and AES, after reviewing the definition and specifications of each. *Table 2.2.4* provides a brief comparison of these algorithms.

Table 2.2.4. Block Cipher Comparison

Cipher	Key size $_{bits}$	Block Size $_{bits}$	Rounds	Security Rate	Speed	Scalability
DES	56	64	16	Inadequate	Slow	No
3DES	112 168	64	48	Secure	Very Slow	Yes
Blowfish	32 - 448	64	16	Moderate	Fast	Yes
AES	128 192 256	128	10 12 14	Secure	Fast	Yes

From this comparison, we note that the AES algorithm is best symmetric cipher, as it is more robust and available in more than one version to adapt with the system requirements.

2.2.1.3 SYMMETRIC CIPHER SUMMARY

In this section, we summarize the symmetric cipher algorithms. *Table 2.2.5* compares stream and block cipher algorithms.

Table 2.2.5. Stream vs Block Cipher

	Stream	**Block**
Design	***Complex***	***Simple***
Data	***Stream of data***	***Split data*** into a set of blocks
Number of bits	***1 bit***	***Depending on Block size***
Complexity	***High***	***Low***
Speed	***Fast***	***Slow***
Resources	Require ***more*** resources	Require ***less*** resources
Confusion and Diffusion	***Confusion***	***Confusion*** and ***diffusion***
Reversing	***Simple***	***Hard***
	Cannot take block cipher properties	Can be as a stream, depending on mode of operation

From this comparison, we found that the stream has better performance and complexity, but it is not guaranteeing the diffusion, can be reversed easily, and it is providing less security. Because of that, we conclude that the block cipher is better solution since it provides more security in the case of text-based and image-based encryption.

2.2.2 CRYPTOGRAPHY SUMMARY

After we have discussed the cryptography algorithms and classified them into Asymmetric and Symmetric, we reviewed their definition and specifications of each type. *Table 2.2.6* provides a brief comparison of these algorithms.

Table 2.2.6. Asymmetric vs Symmetric

	Asymmetric	**Symmetric**
Keys	***2 keys***; one for encryption, and other for decryption	***Single Key*** for encryption and decryption
Key Exchange	***Not a problem***	***Big Problem***
Relation between number of keys and receivers	***# of Keys = (# of receivers) *2***	***# of Keys = # of receivers***
Cipher Size	Same or ***Larger*** than plaintext size	Same or ***Smaller*** than plaintext size
Speed	***Slow***	***Fast***
Data Size	Used for ***small data***	Used for ***Large data***
Provide	***Confidentiality***, ***authenticity***, and ***non-repudiation***	***Confidentiality***
Key Encryption And Recourses Utilization	***High***	**Low**
Examples	***RSA***, ***ElGamal***, ***DH***, ***ECC***, and ***DSA***	***RC4***, ***Salsa20***, ***Sosemanuk***, ***DES***, ***3DES***, ***Blowfish***, and ***AES***

In this section, we discussed Asymmetric and Symmetries cipher in comprehensive detail, and after compare many examples from each type, we found that the block symmetric is the suitable solution for this study goals.

2.3 LIGHTWEIGHT CRYPTOGRAPHY

National Institution of Standards and Technology (NIST) defined ***LWC*** as a cryptosystem whose features have been optimized to meet the requirements of devices of varying specifications, especially resource-constrained devices [37]. From this definition, we conclude that all cryptography terms can be LWC if it is possible to legalize its need for resources to ensure the desired effect. Thus, asymmetric encryption is an exception due to its complexity and demand for high resources. On the other hand, symmetric encryption can be used in these systems if it is properly exploited [37].

Depending on the critical challenges mentioned before, we found that the LWC algorithm should use little memory and power and provide good performance while maintaining the required level of security [38]. Therefore, the factors of LWC requirements can be explained as follows [40]:

- ***Key Size:*** Longer Key size is better for security, but it requires more complexity and power.
- ***Block Size:*** smaller block size is more familiar with IoT since the big block size requires more CPU, memory, and power.
- ***The number of rounds:*** Fewer rounds are better since the rounds require more computation and resources.
- ***Structure:*** The structure here is the way of managing the trade-off between all previous factors to find the optimal combination to ensure an acceptable level of performance and security.

Many LWC algorithms provide other performance and security strengths. And after studying many related studies, we find that there have been some trends in relying on stream cipher due to its high efficiency in terms of performance. Still, most of the algorithms were based on block cipher since it offers better security but with a significant performance improvement. [38]. We highlight some of these LWC algorithms in the following sections depending on its base as a stream or block.

2.3.1 STREAM LWC

In this section, we present some LWC algorithms based on stream cipher methodologies.

2.3.1.1 A4

A4 is a very efficient lightweight stream cipher that uses Left Feedback Shit Register (LFSR) and Feedback with carry shit register. The critical feature of A4 is the ease of implementation and high security. In addition, A4 has proven itself in resistance to brute-force and algebraic attacks [39].

2.3.1.2 NEW LIGHTWEIGHT STREAM CIPHER

New Lightweight Stream Cipher (NLSC) is a chaos-based algorithm that uses an 80-bit secret key, two Nonlinear Feedback Shift Registers (NFCR), and three multiplexers. NFCR has good security, making it resistant to statistical attacks and providing good performance [38], [39].

2.3.2 BLOCK LWC

This section presents some LWC algorithms based on block cipher methodologies.

2.3.2.1 PRESENT

PRESENT is an LCW algorithm that relies on Substitution-Permutation Network (SPN). It was suitable for limited hardware as it uses an 80-bit key. However, it was noted that it takes 32 rounds to encrypt 64 bits. Another version uses a 128-bit key, but it requires more computations [38].

2.3.2.2 GIFT

GIFT is an enhanced PRESENT version; it uses a lighter S-Box with minimal rounds and a faster key scheduling algorithm. These properties enable it to provide more throughput. It is also available in more than one version depending on the required throughput. These versions are; GIFT-64 and GIFT-128. With a 64-bit block size that requires 28 rounds and a 128-bit block size that requires 40 rounds, respectively, with a 128-bit key for both [40].

2.3.2.3 KATAN

KATAN outperforms PRESENT by saving 48% of the power. KATAN uses an 80-bit key and handles different text sizes such as 32, 48, and 64 bits. However, its downside is that it requires 254 rounds to complete this process [40].

2.3.2.4 SIMON AND SPECK

The National Security Agency developed ***SIMON*** as an improved algorithm that uses 22 rounds but uses a lot of arithmetic operations. It offers many different key sizes as 64-bit, 72-bit, 96-bit, 128-bit, 144-bit, 192-bit, and 256-bit that handle 32-bit, 48-bit, 64-bit, 96-bit, and 128-bit block size through 32, 36, 42, 44, 52, 54, 68, 69, and 72 rounds. While ***SPECK*** is the same as SIMON, it supports exact block sizes and keys, but 22, 23, 26-29, and 32-34 rounds [40].

2.3.2.5 RECTANGLE

RECTANGLE is a very fast LWC algorithm, which is different from PRESENT. It Relies on lighter SPN with 25 rounds. This reduced algorithm significantly speeded up the execution based on Bit-slice, as it relies on parallel swapping and replacement [40].

2.3.2.6 SIT

SIT combines Feistel and SP network and takes five rounds to handle 64 bits of text with 64 bits as a key. It mainly consists of two parts: the first is for key expansion, and the second is for the encryption section. Key expansion relies on simple operations such as *concatenation, shifting, addition,* and *XOR*. This algorithm achieves high throughput and low power consumption [40].

2.3.3 LWC SUMMARY

After we discussed various LWC algorithms such as LSC, A4, NLSC, PRESENT, GIFT, KATAN, SIMON and SPECK, RECTANGLE, and SIT in this section. *Table 2.3.1* highlight the most comparison points between them.

Table 2.3.1. LWC Summary

	Key $_{bits}$	**Block size** $_{bits}$	**Rounds**	**Security**	**Characteristics**
A4	128	-	-		Secure, high performance
NLSC	80	-	-		Secure, good performance
PRESENT	80 128	64	32	80%	Low memory, For small data
GIFT	128	64 128	28 40	85%	Simple, Fast Key Scheduling, High throughput
KATAN	80	32 48 64	256		Inefficient, Low throughput Energy consuming
SIMON	64-256	32-128	32-72	67%	High performance, Easy, Flexible
SPECK			22-34	58%	Same SIMON but optimized for software
RECTANGLE	80 128	64	25	60%	Fast, Hardware Friendly
SIT	64	64	5		Fast Key Scheduling, High throughput Need low energy

CHAPTER 3

LITERATURE REVIEW

3 LITERATURE REVIEW

In this chapter, we have discussed the most recent related researches. After studying these researches, we categorized them into two groups. The first group, including [40-52], reviewed LWC and defined its essential requirements. The second group discusses AES versions that are proposed to be compatible with LWC requirements [53-65]. *Figure 3.1* presents the flow of literature review.

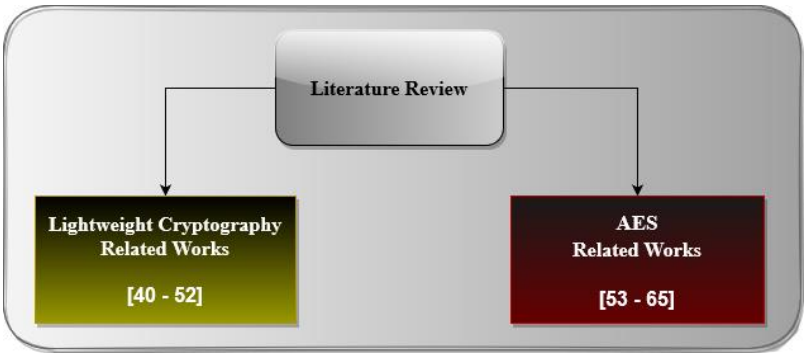


Figure 3.1. Literature Review

3.1 LIGHTWEIGHT CRYPTOGRAPHY RELATED WORKS

This section summarizes some researches that introduce the concept of LWC in terms of terminology, requirements, and how to implement them in line with the available capabilities.

Manifavas et al. [40] discussed lightweight encryption algorithms, focusing on streaming encryption, which provides high performance with simple operations, making it suitable for the capabilities of IoT devices, especially when the text length is unknown or continuous. The results showed the superiority of symmetric encryption in performance. Still, most of the streaming algorithms were not secure, as after analyzing 31 algorithms, it was found that only 6 were secure.

Buchanan et al. [41] emphasized the IoT's security and privacy challenges. Also, the researchers review the trends of designing lightweight algorithms after explaining alternatives to traditional cryptography methods that fit the composition of the IoT. Finally, after reviewing the challenges

in terms of physical and software implementation, the study recommended that when developing LWC solutions, the following should be noted:

- Resorting to small blocks and a short key constitutes a security weakness and leads to faster wear of CBC mode.

- The number of operations is directly proportional to the size of the inputs; in lightweight symmetric cipher almost twice.

- The algorithm architecture must be adapted to new applications and better integrate with existing protocols.

Based on the previously mentioned recommendations, the following are the methods presented by this study that can be included when designing a lightweight security system for the IoT [41]:

- ***Hashing***: is a mathematical algorithm that assigns data of arbitrary size (often called "message") to a fixed-size bit matrix (the "message summary"). It is a one-way function that is practically useless to reverse or reverse the account. Ideally, the only way to find a message that produces a particular hash is to forcibly search for potential inputs to see if they have a match or use a rainbow table of identical hash.

- ***Streaming***: it is a symmetric key cipher in which the plaintext is combined with a string of pseudorandom, keystream characters. In-stream cipher, each plaintext character is encoded with its corresponding character from the stream key to giving the ciphertext characters. An alternative name is state encoding, stream cipher, where the encoding of each character depends on the current state. The character is usually a bit and operation (XOR) or exclusive-or in practice.

- ***Block***: It's an encryption method that applies a deterministic algorithm along with a symmetric key to encrypt a block of text, rather than encrypting one bit at a time as in stream ciphers. For example, a typical block cipher, AES, encrypts 128-bit blocks with a key of a predefined length: 128, 192, or 256 bits. Block ciphers are Pseudo-Random-Permutation (PRP) families that operate on a fixed-size block of bits. PRPs are functions that are computationally indistinguishable from random permutations and, therefore, are considered reliable until their unreliability is proven.

- ***Signing***: Digital signature is a mathematical scheme to verify the authenticity of digital messages or documents. A valid digital signature, where the basic requirements are met, gives the recipient powerful reason to believe that the message was created by a known sender

(authentication) and that the message was not changed during the transfer (integration). Digital signatures are a standard element in most encryption protocols, commonly used for the distribution of software, financial transactions, contract management software, and in other cases where it is essential to detect fraud or manipulation.

Sehrawat et al. [42] presented a detailed comparison between several algorithms compatible with the IoT and after conducting cryptanalysis attacks. This study also showed that block ciphers had attracted the attention of many researchers as a basis for developing LWC algorithms. Finally, this study also recommended the requirements for the future of LWC algorithms.

Dutta et al. [43], reviewed the encryption solutions that can be used in the IoT by comparing some LWC that can fit with the nature of IoT devices. Researchers believe that symmetric encryption is the closest to suit the heart of the IoT. They also found that the modified AES algorithm provides a suitable security solution to the restrictions imposed by the capabilities of IoT devices after studying many algorithms like *DES, 3DES, Blowfish*, etc.

After choosing AES as a standard and reliable algorithm and achieving the desired goal, the researchers analyzed the performance of a set of versions of the algorithm implemented in previous studies by sorting them into two parts as follow [43]:

- **Recent Research Work on AES for IoT**: Many implementations achieved good results in high productivity, low energy, and minimal costs.

- **Recent Research Work on AES for IoT Focusing MixColumns and S-box**: The researchers focused on this aspect of the hardware implementation. Delay and reducing the area are the main goals of algorithm development, so the main challenge that exists to date is to improve Mix-column round and S-box operations. There are many implementations as the Serpent Algorithm that were previously developed to meet the challenges mentioned [43]. With these designs, we can provide good results to achieve these goals.

The researchers also presented a study of attacks on AES that should be monitored and found solutions such as Differential Fault Analysis Attacks and wireless interceptive side-channel attack techniques. These attacks can be resisted through the use of dummy keys and XOR operations [43].

Rajesh et al. [44] presented the Novel Tiny Symmetric encryption Algorithm (NTSA), which provides better confusion for each round which leads to better security level. The comparison

centered with the TEA algorithm is considered one of the most attractive algorithms because of its ease of implementation and less memory usage. Its main problem is to use the same key for all rounds, which reduces the level of security and its poor performance. The results show that NTSA outperforms many other security algorithms and achieves better performance, making it more suitable for IoT and embedded devices.

Noura et al. [45], the researchers proposed a lightweight stream cipher. This design derives its security strength from the dynamic key. This design is based on simple operations that do not require as high capabilities as XOR and LFSR in resource consumption. The dynamic key of this algorithm is a function of a secret key and nouns. The S-boxes depend on this key, making it more difficult to detect.

Based on what the researchers put forward, this algorithm is characterized by low overhead by relying on simple operations and low sequencing, variable cipher primitive that changes after each time, low error propagation, a simple implementation that does not require large memory.

Gunathilake et al. [46] discussed the future applications of LWC, how to implement it, and the challenges it faces. The study also touched on the existing LWC algorithms previously mentioned in our research and confirmed the effectiveness of the modified AES algorithm in this field.

Usman et al. [47] reviews the light encryption algorithms that fit the nature of the IoT after identifying the obstacles to using traditional algorithms, such as the low power capacity of the devices. Researchers believe that the security of big data flowing through the IoT is the main problem, as this weakness may overwhelm the advantages of IoT applications. Therefore, considering the capabilities of these devices represented in the low capacities, it was necessary to think of new methods that require simpler arithmetic operations and less memory while providing an acceptable degree of security. In addition to what has been mentioned, these methods must consider the diversity of devices, their different capabilities, and the protocols used to have the ability to integrate and adapt to this diversity. And now we still have the issue of privacy, as the IoT, with the vast amounts of data circulating, must provide the user with the possibility of appropriate control over his data [47].

The researchers considered that symmetric encryption is best suited for the IoT because asymmetric encryption requires higher capabilities. And the following are some of the symmetric encryption algorithms that have been reviewed [47].

Abutair et al. [48] believe that despite their importance, smart cities still face the challenge of balancing the quality of service and maintaining the privacy and security of information. This study summarized the difficulty of achieving this balance as follows:

- Design limitations and limited capabilities make this environment an easy target for hacking.
- The truth of the data may be injected to cause damage, leading to great disasters.
- Difficulty of building a standardized system due to different manufacturers.

The researchers studied many lightweight algorithms used in the IoT. Based on this study, an infrastructure has been proposed that provides a specific degree of privacy and security for the IoT. This study concluded that some modern algorithms such as *CLEFIA* and *TRIVIUM* achieved terrible results compared to the old algorithms, especially *TRIVIUM*, which gave disastrous results [48].

The study explains the structure of smart cities. Without going into details here, the aspect that concerns us is the necessity of providing IoT devices with algorithms that meet the guarantee of authentication, integration, and confidentiality to protect the network from threats. Such as *Corrupted Data, Replay Attacks, IP Spoofing, Identity Usurpation, DoS/DDoS Attacks, and, Data Leakage [48].*

This study presented a new design that depends on the capabilities of the device that will be added. Based on these capabilities, the appropriate lightweight algorithm is selected for it. The mechanism of this design can be summarized as follows:

- *Input:* Device specifications
- *Knowledge Base:* minimal requirements for each lightweight algorithm.
- *Output:* The appropriate algorithm for this device.

After testing many algorithms by changing some factors, the researchers found that the algorithm closest to adapting to the majority of IoT devices is the *AES* algorithm, with the need to reduce its resources [48].

Khalifa et al. [49], after discussing the challenge resulting from changing objects places, a new method has been proposed to protect the IoT system from address modification attack and heap penetration by encrypting the object during runtime using a cryptographic hash function. The results proved that this method is powerful, effective, and provides a good level of security. After discussing the challenge caused by object relocations, this study proposes a way to protect the IoT system from address modification attack and heap penetration by encrypting the object during runtime using the ECC besides cryptographic hash function. The results prove that this method is powerful, effective, and provides a good level of security.

The researchers also proposed an authentication system intending to verify devices and collect their behavior dynamically to detect any unusual activity in the system using machine learning algorithms.

Ramadan et al. [50] introduced a LWC algorithm called LBC-IoT that handles 32-bit blocks with a key of up to 80 bits. This algorithm is based mainly on the Feistel structure, along with simple operations such as XOR that do not consume power and 4-bit S-boxes. The results indicate the strength of this algorithm against attacks in addition to its acceptable performance, and it is considered a promising algorithm for implementation on small and very restricted devices.

Periasamy et al. [51] proposed a lightweight block cipher mechanism that works on 8-bit processing, as their study indicates that this algorithm is superior to its counterparts. According to the researchers, this algorithm derives its strength from the strength of the encryption in the compensation boxes. In terms of performance, the design of the compensation boxes played marginally using the Multi sequence Linear Feedback Shift Register and reliance on simple operations such as XOR, shifting, and registers to reduce space required and optimization in power consumption and speed.

Thabit et al. [52], researchers introduced a New LWC Algorithm (NLCA) to secure cloud computing applications. This algorithm uses a 16-byte key based on Feistel and substitution permutation. This algorithm succeeded in achieving confusion and diffusion by introducing some logical operations into the algorithm's formula, such as Shifting, Swapping, and XOR. One of the advantages of this algorithm is the flexibility, such as AES, where the number of rounds and the length of the key are variable according to the application's needs. The results also indicate that

this algorithm provides a good level of security and performance, which makes it suitable for these applications.

3.1.1 LIGHTWEIGHT CRYPTOGRAPHY RELATED WORKS SUMMARY

In this section, we focus on the key points that have been noticed upon studying LWC related studies. *Table 3.1.1* summarizes these studies.

Table 3.1.1. LWC Related Works summary

Study	Key Points
[40] 2015	▪ Discuss many LWC algorithm. ▪ It shows that the symmetric encryption is very good in performance, but most symmetric algorithms are not secure.
[41] 2018	▪ Discuss IoT security challenges. ▪ Recommendations to be followed when developing LWC.
[42] 2018	▪ Compare many security algorithms that compatible with IoT. ▪ It shows that the block cipher algorithms are more suitable to be used. ▪ It also recommended the requirements for the future of LWC algorithms.
[43] 2019	▪ Discuss some encryption techniques that can be used in IoT. ▪ Discuss AES algorithm.
[44] 2019	▪ Propose NTSA which provide good security level.
[45] 2019	▪ Propose a new lightweight stream cipher. ▪ The proposed model based on simple operation as XOR. ▪ The proposed model does not require many resources.
[46] 2019	▪ Discuss the future of LWC and its challenges.
[47] 2020	▪ Discuss some LWC algorithms. ▪ It recommended the adoption of symmetric encryption because asymmetric encryption requires powerful resources, and this is what IoT devices lack.
[48] 2020	▪ Discuss many LWC algorithms. ▪ The study found that modern algorithms did not meet the requirements due to poor results. ▪ The study recommended the use of AES due to its strength, provided that it is configured to improve performance.
[49] 2020	▪ Propose a new LWC which provide good level of security. ▪ Propose a new authentication method.
[50] 2021	▪ Propose LBC-IoT which provide very good performance with low power consumption.
[51] 2021	▪ Propose a new lightweight block cipher which provide a good security and performance.
[52] 2021	▪ Propose NLCA which used to secure the cloud, and it provide very good security with accepted performance.

3.2 AES RELATED WORKS

In this section, we summarize some researches that present some AES-based system, discuss these systems and highlight the differences in these AES versions to reach the best possible ways to improve the performance and strength of this algorithm more.

Javed et al. [53], presented a new design for the AES algorithm to make it suitable for mobile devices and speed it up despite the limitations of the hardware specifications. After reviewing the mechanism of the standard AES algorithm, the researchers discuss the improvement that was made to AES implementation and the motives that were relied upon in this optimization as follows:

- This optimization used a 10-byte look-up table for round constant and two 256-bytes look-up tables for S-box and InvS-box. The constant round means that the three rightmost bytes are always 0. Thus, XOR performed only on the leftmost byte of the word. The round constant differs from one round to the other.

- ***In MixColumns***, the multiplication with 02 can be performed by a left shift and bitwise XOR with 1b.

- ***In ShiftRow***, Using the row index as a specific number (i), each row is rotated to the left by i. This implies that the first row will not be rotated.

- ***In RoundKey***, a rounded key is added to the State matrix by a simple bitwise XOR operation: a sum in the field GF (2^8). Each round key is obtained from the key schedule.

- There are two ways to implement Key scheduling: (1) key unrolling (2) On the fly key generation. This study implements key unrolling because that *On the fly key generation* approach is costly in clock rounds and need 16 bytes of additional memory to store the last round keys for the decryption [53].

The results of this study showed that the performance of the proposed method gives better results, as it provides 3 times better encryption speed and is about 20 times better in round keys calculations. This design outperformed its predecessor by 20 times while reading data from the hard disk and encrypting it if the data was greater or equal to 1 MB [53].

Abhijith et al. [54], presented an improved model for implementing the AES algorithm by slicing and integrating the internal processes of the algorithm. This new version used Block-Ram and 10 levels of pipelines to improve efficiency and productivity. The results indicate that this enhanced version significantly enhances performance and the possibility of integrating it with other systems.

Bui et al. [55] worked on finding an improved version of AES in several ways. First, reduce the combinational logic and number of records by organizing the data path. Second, the clock gateway strategy, key expansion, and minimization of data activities contributed to reducing the algorithm's energy use. Here are the modifications that have been implemented to achieve the above improvements:

- By using the Low Power S-Box, power consumption is reduced.
- Logic relationships were reduced by manipulating data by columns after eliminating ShiftRow.
- Using a special mechanism to load data and encryption keys limits the number of records.
- Finally, the clock gate scheme worked in reducing energy consumption.

These modifications were additions that can be used without modifying the algorithm. As for the fundamental alterations in the algorithm, they were represented as follows [54]:

- ***Thirty-Two-Bit Datapath Optimizations:*** The Advanced Low Power Encryption Standard (AES) can be used in smaller applications such as small-scale IoT devices. Proposed 32-bit AES data paths to meet low energy consumption and small space requirements. We only use the 32-bit data path in MixColumns.

- ***Substitution Box:*** The S-box takes several input bits (m) and converts them into a certain number of output bits (n), where n is not necessarily equal to m. m × n S-box can be performed as a search table with 2 million words each n bits. Fixed tables are usually used.

- ***Key Expansion Optimizations:*** The expansion was implemented in VHDL, resulting in ascending design and test methodology. This choice also ensures that the code can be transferred to different vendors' devices. The code and simulation were manufactured using Altera MAX + PLUS II version 7.21 Student Edition. The FPGA family was selected for execution from Altera Flex 10K. It's part of an 8-bit execution with a 128-bit block and a 128-bit key. Because the goal of improvement is to reduce consumption, to suit it for mobile applications, the structure is directed to minimize space.

The results show that the proposed version offers the same PRESENT algorithm in energy use. Also, the proposed system is resistant to the attack of power correlation analysis with less than 20,000 traces, which seeks to expose the data path. Also, the data path in case of parallelism provides it with more robustness. Finally, this design uses different key sizes, which contributes to providing various levels of security as needed [55].

Mamun et al. [56] provided a comprehensive explanation of the AES algorithm. The study presented a new model for the AES algorithm to enhance its security level by adding an XOR operation to an extra byte of s-box and using an additional random key. The results indicate that this modification contributed to improving the level of AES security variably due to the randomness of the added key. The results also showed that this modification improved confusion and increased time security.

Farooq et al. [57] tested AES using different techniques depending on the resources of the target devices, the results were characterized by varying in nature according to the techniques used. Among these techniques were used; Parallelization and storage of s-box and key expansion, as it has been noted that the introduction of such technologies helps in optimizing the exploitation of resources to provide better results.

Daoud et al. [58], the researchers present an optimization of the AES algorithm using Vivado High-Level Synthesis (HLS), and their results show significant progress in increasing the throughput of the proposed algorithm, which was implemented on the FPGA only using flip flops and look-up tables.

Since optimizing commands in Hardware Description Languages (HDL) is not easy and time-consuming, HLS improves the algorithm with less effort. HLS is an automated process that deals with high-level programming languages such as C that is used to ease the struggles that HDL requires in the development process, debugging, and provide flexibility in meeting system requirements. HLS tool synthesized compiled core AES functions in an RTL block, and sub-functions were divided into sub-blocks at higher system levels. Below is a review of the improvements that this study made to the AES algorithm [58]:

- ***Key Expansion-based Implementation:*** key expansion process combined with the encryption process so that the two processes will run simultaneously during each round.

- ***SW-based Implementation:*** Key extension process is performed before the encryption process to obtain 11 different 128-bit keys based on AES-128 design.

- ***High Throughput-based Optimization****:* The algorithm has made some special optimizations to increase the encryption throughput.

The main objective of this study was to achieve the maximum throughput in encryption. The process that most positively affected the results is integrating key expansion with encryption. By comparing the effects of frequency, productivity, and area utilization, it appears to us that the proposed design in this study has outperformed the previous strategies [58].

Proceeding from the fact that the AES algorithm is considered the best secure algorithm currently available and can be adapted to IoT devices. ***Rokan*** et al. [59] provided an integrated security system for the IoT called *Modified Lightweight AES* (***MLAES***) that includes two integrated systems; The first one is a *Secure Encryption* based on a lightweight version AES integrated with Chaos Maps. The second is a *Secure Authentication* using a chaotic hash function based on SHA3-256-bits. The following is a review of the three main phases of this system:

- ***Lightweight Modified AES:*** The goal of mitigating and optimizing AES is to reduce computational complexity, execution time and reduce required iterations and memory used. One of the most important modifications is the use of 4 chaos keys, which increases the randomness of results, which means enhancing system security. The first modification in the algorithm uses *shifting operations*, *data blocks*, and *logical functions*. MLAES uses two sub-boxes, each dealing with 64 bits of data. The second modification is to make the number of times of rounds and ShiftRow are executed dynamically based on a dynamic number. This number is generated depending on some chaos keys that change with each iteration. Finally, the last modification is to eliminate the MixColumns operation due it its complexity and high execution time by replacing it with some XOR operations, SHA3-128, and shift operations.

- ***Modified Sub-Bytes(S-Box):*** The s-box represents one of the complex operations in MLAES and is directly related to the degree of security of the design; S-Box takes 128 bits of data and divides it into 16-bit blocks. Every 64 bits of data is sent to a sub-S-box, where the system contains two S-Boxes. The S-Box shifted after each iteration using K to change its values.

- ***The Proposed IoT Security System:*** As mentioned earlier, besides the MLAES encryption process described in the previous points, the proposed system includes a hashing stage using SHA3-256.

The study results indicate that despite the modification to AES, the level of security remained strong, in addition to the significant improvement in its performance and the specifications required for its operation. Perhaps the most prominent result was that this system passed the NIST tests, which means that the system is resistant to linear differential attacks and brute force attacks [59].

Farooq et al. [60], given the discrepancy between the capabilities of IoT devices, explored five implementations of the AES algorithm. These applications use modifications and improvements to the AES algorithm. The applications indicate the disparity in the results, as each of these applications fits a specific category of IoT devices. Therefore, the study recommended moving away from comprehensiveness and not limiting encryption to one algorithm for all devices, but instead relying on the device's capabilities to choose the optimal AES version for use.

Nagalakshmi et al. [61], given the discrepancy between the capabilities of IoT devices, presented some strategies for implementing AES with a set of other systems to suit these devices of varying powers, and the study also touched on the use of LFSR. The results indicate a security improvement, the ability to check signatures, and random checks without significantly affecting performance.

Pablo et al [62], this study provided an understanding of the dynamic AES algorithm, by changing the core AES operations represented by: SubBytes, ShiftRows, MixColumns, and AddRoundKey which were compensated by random key-dependent transformations of RandomSubBytes, RandomShiftRows, RandomMixColumns, RandomAffineTransfKey respectively. The contribution of this study represented by these fundamental changes helps to provide better security with more random properties.

Salim et al. [63] presented the development of an AES algorithm called multi-key AES. The name came concerning the fact that this proposal uses the AES algorithm but uses several keys as the

secret key is used to configure a variable number of keys using ECC. The study specialized in implementing this algorithm in the IoT, provided that it is used on devices capable of running this algorithm. The results indicated that this modification did not affect the algorithm's performance, but it contributed to improving its security.

Amandeep et al. [64], tested both a strict crash rate and a bit independence criterion of eight rounds AES algorithm with a dynamic S-Box used to verify that it maintains its security level. The results showed an improvement in the level of performance due to the reduction in the number of rounds, in addition to the improvement in the level of security compared to its reliance on this improved S-Box.

Lihang et al [65], proposed a new model for representing AES by introducing some extended operations such as using DRAM, which reduces runtime by providing parallelism between encryption and access to the system. The results indicate a significant improvement in the effectiveness of the algorithm by reducing the encryption time. However, on the contrary, this has resulted in an increase in energy consumption.

3.2.1 AES RELATED WORKS SUMMARY

In this section, we focus on the key points that have been noticed upon studying AES based algorithm related studies. *Table 3.2.1* summarizes these studies.

Table 3.2.1. AES Related Works summary

Study	Key Points
[53], 2010	▪ Use new look-up tables for S-box and InvS-box. ▪ Optimize MixColumn, ShiftRow, and RoundKey. ▪ These optimizations enhance AES performance.
[54], 2014	▪ Improve AES by use slicing and integrating processes, block-RAM, and 10 level pipelines. ▪ These modifications enhance the performance of AES.
[55], 2017	▪ Reduce combinational logic and number of records. ▪ Use Low- Power S-Box, and clock gate scheme. ▪ Eliminate ShiftRow. ▪ These optimizations enhance AES performance.
[56], 2017	▪ Provide a comprehensive study of AES. ▪ Enhance AES by adding an XOR operation to an extra byte which enhance the security of AES. ▪ This enhancement improves the time security and confusion.
[57], 2017	▪ By testing AES, this study shows that the improvement in AES performance can be done by parallelization storage of S-Box, and key expansion.
[58], 2019	▪ Using Vivado HLS which enhance the throughput of AES.
[59], 2019	▪ Propose MLAES which provide a secure encryption AES-based algorithm and secure authentication used chaotic hash function. ▪ Enhance AES by use 4 chaos keys to improve security. And use two sub-boxes.
[60], 2020	▪ Using 5 modified AES models and studying their results, the study recommended not relying on the same algorithm on all devices, but rather choosing the appropriate algorithm for the capabilities of each device.
[61], 2020	▪ Modifying AES by using LFSR. ▪ This modification enhances the security of AES, but it didn't improve its performance.
[62], 2020	▪ Modifying AES by changing all its core operations to a new version to enhance AES security and give it more randomness.
[63], 2021	▪ Use AES but with several keys based on ECC. ▪ This optimization improves AES security, but it didn't enhance its performance.
[64], 2021	▪ Using 8 rounds of AES and tested in both strict crash rate and bit-independence criterion. ▪ 8 rounds maintain same level in these tests, and provide better performance compared with AES. ▪ By using improved S-Box, AES became more secure.
[65], 2021	▪ Using DRAM to reduce runtime and provide penalization which leads to enhancement in performance.

CHAPTER 4

EVALUATION CRITERIA

4 EVALUATION CRITERIA

This chapter provides a definition and clarification of the mechanisms for evaluating the algorithm in terms of performance and security. All implementation and evaluation processes were carried out 3 times to get more accurate results, and the average has been taken.

4.1 DEVICE SPECIFICATIONS

In this section, we present the environmental elements used in this thesis.

4.1.1 HARDWARE SPECIFICATION

The specification of the devices that used, are as follows:

- Computer Test Environment:
    - Intel Core i7 7700hq, 2.4 GHz.
    - 16 GB DDR4 RAM with 2400 MHZ frequency.
    - SSD Hard Disk.
- IoT Environment:
    - RP 3 model B 2017 with 1.4GHz processor and 1GB SD RAM.

4.1.2 SOFTWARE SPECIFICATION

In this thesis, many software's and programming languages that have been used to implement and test algorithms and tests. The used software's are as follows:

- Linux Ubuntu with G++ compiler for NIST Tests.
    - PowerTop to monitor power usage on Ubuntu.
- Windows 10 Professional for algorithm code and other Tests:
    - Python 3.10 for algorithm implementation, confusion and diffusion, and performance test. The Hardware Usage calculated using Python Profiling.
    - MATLAB 2016 for histogram, mapping, chi-square, and correlation tests.

- Intel(R) PowerGadget to monitor the power usage on windows.
- Python Profiling to measure the Hardware Usage.

4.2 TESTED DATA

All evaluation used different data sizes including 10 KB to 500 KB of texts as a small data size, and 1 MB to 100 MB texts as a large data size. Emphasizing that what applies to texts applies to images, but in the case of audio and video, we will need some compression mechanisms before encryption.

4.3 PERFORMANCE EVALUATION METRICES

The performance evaluation process indicates the time and resources the algorithm needs to complete its work. Below, we review the necessary performance tests.

4.3.1 EXECUTION TIME

Execution time is one of the essential parameters for evaluation performance. It measures the time needed to encrypt or decrypt a specific data size [66].

4.3.2 THROUGHPUT

Throughput reflects how much data can be processed during a time. It presents the average of data in Kb divided by the average Encryption or Decryption time.

4.3.3 HARDWARE USAGE

We calculate the Processes, CPU and RAM used by the algorithm to measure the effect of optimization using application profiling in python.

4.3.4 Power Consumption

We calculate the Average Power Consumption (APC) used by the algorithm to measure the effect of optimization.

4.4 Security Evaluation Metrics

The process of evaluating the security shows the strength of the security level of the algorithm. Below we review the necessary security tests.

4.4.1 Mapping

The phase space trajectory is one of the characteristics of the generated sequence that reflects the system's dynamic behavior. A system is secure if the relationship between x(n) and x(n+1) is unknown [66].

4.4.2 Histogram

Another critical property of any robust algorithm is to provide a uniform distribution in the whole phase space. This test evaluates the uniformity of data in a graphical representation [9][66].

4.4.3 Chi-Square Test

The Chi-Square test is used to assert the uniformity of generated sequences. The following formula calculates the statistical Chi-Square test $\chi 2$:

$$X_{exp}^2 = \sum_{i=0}^{K-1} \frac{(O_i - E_i)^2}{E_i} \quad (2)$$

with K being the number of classes (sub-intervals) equal to 256, Oi is the number of observed (calculated) samples in the i-th class, and Ei is the expected number of samples of a uniform distribution, $E_i = 10^7/k$. We compare the experimental value calculated above with a theoretical value obtained for a threshold α=0.05 and a degree of freedom K-1=999. To prove the uniformity of a generated sequence, the experimental value of chi2 must be smaller than the theoretical value $\boldsymbol{X_{exp}^2 < X_{th}^2}$. Therefore, the generated sequence is uniform distribution if it has lower chi2 value. [66].

4.4.4 KEY SECURITY

The first way to judge an algorithm's strength is its ability to withstand a brute force attack. The power of the algorithm to counter this attack depends mainly on the length of the key. As a relative secondary effect, the difficulty of the algorithm in terms of execution time is added. However, since the increase in the execution time means the increase in the time of breaking the algorithm, this effect is almost negligible beside the impact of the length of the keys [9][66].

Suppose the key length is (n) bits and the average execution time is (x) unit. then the breaking formula can be written as:

$$Br = 2^n * x \quad (3)$$

That's means we need Br time to break the algorithm. In addition, if n is much greater than x, we can suppose that

$$Br \approx 2^n \quad (4)$$

4.4.5 AUTO AND CROSS-CORRELATION

The properties of a sequence are that the values in the sequences are not repeated or correlated, and the cross-correlation of two sequences x and y (generated with slightly different keys) is ***close to zero***. The correlation coefficient ρ_{xy} of the two sequences x and y is calculated by the following mathematical equations [9][66][68]:

$$\rho_{xy} = \frac{cov(x.y)}{\sqrt{D(x)}\sqrt{D(y)}} \tag{5}$$

Where,

$$cov(x.y) = \frac{1}{N}\sum_{i=1}^{N}([x_i - E(x)][y_i - E(y)]) \tag{6}$$

$$D(x) = \frac{1}{N}\sum_{i=1}^{N}\left(x_i - E(x)\right)^2 \tag{7}$$

$$E(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i) \tag{8}$$

4.4.6 CONFUSION AND DIFFUSION

For security analysis, ***confusion*** study the relationship between ***ciphertext*** *and* ***key***. This relation should be robust. In simple words, the changing of 1-bit in secret key should lead to a significant change in ciphertext. On the other hand, ***diffusion*** study the relationship between ***ciphertext*** *and* ***plaintext***. In simple words, changing 1-bit in plaintext should affect the ciphertext highly [67].

The optimal security result for this effect should be **50%**. It can measure using Hamming Distance (HD). ***HD*** is a metric used to compare two binary data with the same length, which shows the similarity between these two data sets. **50%** Ensure that the probability that the bit value is 0 or 1 is equal.

4.4.7 NIST TESTS

To evaluate the statistical performances of the generated data, we also use one of the most popular tests for investigating the randomness of binary data, namely the NIST statistical test. NIST released a suite for testing PRNGs that contains 188 tests, including 15 main tests. These tests attempt to test the randomness of binary sequences produced by an algorithm. In addition, these tests focus on different types of non-randomness that could exist in a binary sequence [9][68].

We used the NIST test on all of these entities. A set of ***100 P-values*** is produced for each test, and a sequence passes a test whenever the ***P-values*** $\geq \alpha = 0.01$, where α is the level of significance of the test. A value of $\alpha = 0.01$ means that **1%** of the ***100 sequences*** are ***expected to fail***. The proportion of sequences passing a test equals the number of P-values ≥ α divided by 100. The minimum pass rate for the random excursion (variant) test is 95. The number of sequences that need to pass each test (except the random excursion test) should be ***over than or equal to 96***, given that the total number of generated sequences is 100. The general tests of this package are as follows [9][67]:

- ***Frequency (Monobit) Test:*** This test determines whether the number of zeroes and ones produced by the pseudo-random number generator is approximately equal to those found in a truly random sequence.

- ***Frequency Test within a Block:*** This test determines whether the number of zeroes and ones within M-bit blocks produced by the generator is approximately equal to those found in a truly random sequence. For block size M=1, this test degenerates to the Frequency test.

- ***Runs Test:*** This test focus on the total number of runs (uninterrupted sequence of identical bits) in the binary sequence. The purpose is to determine whether the calculated number of runs of zeroes and ones of different lengths is as expected for a truly random sequence.

- ***Test for the Longest Runs of Ones in a Block:*** This test focus on the longest run of ones within M-bit blocks. The purpose is to determine whether the length of the longest run of ones found in the sequence is approximately equal to that found in a truly random sequence.

- ***Binary Matrix Rank Test:*** The purpose of this test is to check if there exists a linear dependence among fixed-length substrings of the original sequence. This test focus on the rank of disjoint sub-matrices of the entire sequence.

- ***Discrete Fourier Transform (Spectral Test):*** This test focus on the peak heights in the DFS of the sequence. The purpose is to detect periodic properties and features in the sequence of bits. If there are frequent features, that would indicate a deviation from the assumption of randomness. The main intention of this test is to detect if the number of peaks exceeding the 95% threshold is different than 5%.

- ***Non-Overlapping Template Matching Test:*** This test focus on the number of occurrences of specific strings. The purpose is to calculate the number of events of a given pattern and detect generators that produce too many occurrences of that pattern. An *m*-bit window is used to search for that specific pattern. If the pattern is found, the window is reset to the first bit after the successfully found pattern, and the search continues. However, if the pattern is not found, the window slides a one-bit position.

- ***Overlapping Template Matching Test:*** This test focus on the number of occurrences of specific strings. It is the same as the Non-Overlapping Template Matching test as it uses an *m*-bit window. The only difference between this test and the Non-Overlapping Template Matching test is that when the pattern is found, the window slides only one bit instead of being reset to the bit after the found pattern.

- ***Maurer's Universal Statistical Test:*** The purpose of the test is to check whether the sequence from the generator is compressible or not by measuring the number of bits between matching patterns. It fits if the generator's sequence can be significantly compressed without data loss. The sequence is considered to be random if it is not considerably compressible.

- ***Linear Complexity Test:*** The purpose of this test is to check complexity through the length of a linear feedback shift register (LFSR). A sequence considered random if it is complex enough. The longer LFSR in a sequence, the more complicated it is.

- ***Serial Test:*** This test is similar to the frequency test for m-bit patterns. This test measures the frequency of all overlapping possible m-bit patterns in the sequence; 3-bit overlapping pattern: 000, 001, 010, 100, 011, 110, 101, 111. The frequencies of overlapping patterns in a sequence should be approximately the same as those in a random sequence. All m-bit patterns have the same chance of appearing in a random sequence.

- ***Approximate Entropy Test:*** An advancement on the serial test, this test measures the frequency of all possible overlapping m-bit blocks of two adjacent lengths (m, m+1) in a sequence with the expected results from a random sequence.

- ***Cumulative Sums Test:*** The purpose of this test is to determine whether the cumulative sums (cusums)* in partial sequences in the tested sequence is too large or too small relative to the

behavior+ of the cusums in a random sequence. The Cusum should be considered as a random walk. The random walk should be near zero in a random sequence.

*Cusum: a sequence of partial sums of a given sequence.

- ***Random Excursions Test:*** This test checks if the number of visits to a selected state within a cycle deviate from what is expected in a random sequence. The test determines the number of cycles having exactly K visits in a cumulative sum random walk. This test is a series of eight tests made with these states: -4, -3, -2, -1, +1, +2, +3, +4.

- ***Random Excursions Variant Test*** detects deviations from the expected number of visits to various states in the random walk. The test determines the number of times a known state is visited in a cumulative random walk. This is a series of eighteen tests made with these states: -9, -8, …, +8, +9.

4.4.8 OTHER RANDOMNESS TESTS

The differential attack is one of the attacks that threatens encryption algorithms. This attack can be resistance by relying on the confusion that can be produced by making a small change (such as one pixel in an image) that leads to a large change in the result of the encryption process [69].

Unified averaged changed intensity (UACI) and Number of Pixel Change Rate (NPCR) represent two important criteria in studying this change, which study the change in the encrypted image when changing one pixel in the original image. NPCR measures the percentage of different pixels and it calculated using formula 9.

$$NPCR = \frac{\sum_{i,j} D_{(i,\ j)}}{W \times H} \times 100\% \qquad (9)$$

If ($C_{1(i,\ j)} = C_{2(i,\ j)}$) then $D_{(i,\ j)} = 1$ else $D_{(i,\ j)} = 0$. Where C1 is an encrypted image and C2 is the same encrypted image, but with one pixel change.

UACI measures the average intensity of the difference between the two encoded images and it calculated using formula 10 [69].

$$UACI = \frac{1}{W \times H}\left[\frac{C_1(i,j) - C_2(i,j)}{255}\right] \times 100\% \qquad (10)$$

The encryption algorithm has an acceptable degree of differential attack resistance if it has an **UACI score >=33**% and **NPCR score >=99%.**

CHAPTER 5

AES EVALUATION

5 AES EVALUATION

In this chapter, we discuss the results of testing AES performance and security for all modes of operation. These results enable us to specify which suitable mode can satisfy acceptable performance and security levels.

5.1 PERFORMANCE RESULTS

This section presents the results of encryption/decryption time, and encryption/decryption throughput founds according to testing the AES algorithm on different data sets.

5.1.1 ENCRYPTION

Table 5.1.1 and *Table 5.1.2* show the results of encryption time and encryption throughput of the AES algorithm on 10-500 KB data files.

Table 5.1.1. AES Encryption Time for 10 - 500 KB data

	Data	**Encryption Time**				
	KB	**S**				
		ECB	**CBC**	**CFB**	**OFB**	**CTR**
	10	0.067	0.076	0.108	0.082	0.102
	25	0.182	0.191	0.419	0.287	0.21
	50	0.379	0.388	0.584	0.776	0.377
	100	0.889	0.717	0.734	0.839	0.82
	150	1.039	1.136	1.067	1.222	1.601
	250	1.814	1.866	1.791	2.217	2.224
	500	3.581	3.547	3.865	4.196	4.153
Total	**1085**	**7.951**	**7.921**	**8.568**	**9.619**	**9.487**
Average	**155**	**1.136**	**1.132**	**1.224**	**1.374**	**1.355**

Table 5.1.2. AES Encryption Throughput for 10 - 500 KB data

	Data	Encryption Throughput				
	KB	Kb/sec				
		ECB	CBC	CFB	OFB	CTR
	10	1263.644	868.354	1176.640	1084.595	1165.104
	25	1288.462	954.407	1113.167	1237.955	1104.940
	50	1263.449	1031.131	1157.534	1218.134	1029.373
	100	1276.440	1192.963	1023.265	1203.448	1061.415
	150	1309.005	1257.885	1075.722	1223.603	1200.876
	250	1293.176	1107.662	1075.975	1209.065	1173.725
	500	1253.503	1034.715	1065.098	1173.584	1194.182
Total	**1085**	**8947.680**	**7447.117**	**7687.402**	**8350.385**	**7929.615**
Average	**155**	**1278.240**	**1063.874**	**1098.200**	**1192.912**	**1132.802**

In *Table 5.1.3* and *Table 5.1.4*, we present the results of encryption time and encryption throughput of AES algorithm on 1-100 MB data files.

Table 5.1.3. AES Encryption Time for 1 - 100 MB data

	Data	Encryption Time				
	MB	S				
		ECB	CBC	CFB	OFB	CTR
	1	6.705	7.137	6.273	6.741	7.648
	5	33.593	40.409	40.240	41.779	39.317
	10	74.706	82.259	82.954	77.852	80.929
	20	142.978	162.870	166.557	160.638	155.909
	50	377.366	402.444	395.498	380.934	382.516
	100	738.048	763.620	789.345	787.772	782.514
Total	**186**	**1373.396**	**1458.739**	**1480.867**	**1455.716**	**1448.833**
Average	**31**	**228.899**	**243.123**	**246.811**	**242.619**	**241.472**

Table 5.1.4. AES Encryption Throughput for 1 - 100 MB data

	Data	Throughput				
	MB	Kb/sec				
		ECB	CBC	CFB	OFB	CTR
	1	1162.184	1123.932	1104.572	1141.404	1061.968
	5	1217.422	1011.826	1018.928	979.558	1042.075
	10	1094.543	996.084	989.717	1054.715	1011.893
	20	1143.571	1004.784	986.417	1021.759	1051.006
	50	1082.688	1017.868	1033.744	1072.379	1067.951
	100	1106.917	1069.792	1035.343	1037.063	1043.971
Total	**186**	**6807.325**	**6224.286**	**6168.721**	**6306.878**	**6278.864**
Average	**31**	**1134.554**	**1037.381**	**1028.120**	**1051.146**	**1046.477**

5.1.2 DECRYPTION

Table 5.1.5 and *Table 5.1.6* show the results of decryption time and decryption throughput of AES algorithm on 10-500 KB data files.

Table 5.1.5. AES Decryption Time for 10 - 500 KB data

	Data	Decryption Time				
	KB	S				
		ECB	CBC	CFB	OFB	CTR
	10	0.080	0.112	0.071	0.079	0.066
	25	0.199	0.303	0.173	0.167	0.214
	50	0.408	0.441	0.385	0.361	0.342
	100	0.798	0.795	0.781	0.670	0.758
	150	1.157	1.243	1.340	0.982	1.242
	250	2.029	2.395	1.784	1.675	1.787
	500	4.584	4.608	3.454	3.447	3.432
Total	**1085**	**9.255**	**9.898**	**7.989**	**7.380**	**7.841**
Average	**155**	**1.322**	**1.414**	**1.141**	**1.054**	**1.120**

Table 5.1.6. AES Decryption Throughput for 10 - 500 KB data

	Data	Decryption Throughput				
	KB	Kb/sec				
		ECB	CBC	CFB	OFB	CTR
	10	1000.742	749.206	1126.194	1087.014	1221.472
	25	1003.749	697.342	1155.123	1203.845	997.732
	50	982.277	914.747	1044.149	1118.267	1169.637
	100	1004.035	1006.853	1064.523	1195.268	1095.062
	150	1036.983	968.365	948.521	1222.835	1023.358
	250	986.958	851.328	1139.415	1195.783	1122.551
	500	895.030	877.323	1161.710	1160.616	1165.517
Total	**1085**	**6909.774**	**6065.164**	**7639.634**	**8183.628**	**7795.331**
Average	**155**	**987.111**	**866.452**	**1091.376**	**1169.090**	**1113.619**

In *Table 5.1.7* and *Table 5.1.8*, we present the results of decryption time and decryption throughput of AES algorithm on 1-100 MB data files.

Table 5.1.7. AES Decryption Time for 1 - 100 MB data

	Data	Decryption Time				
	MB	S				
		ECB	CBC	CFB	OFB	CTR
	1	8.207	9.702	7.752	7.578	8.107
	5	46.602	51.621	40.682	38.544	37.347
	10	92.924	99.092	80.733	78.250	78.131
	20	190.093	195.674	158.200	159.774	162.334
	50	486.682	501.812	390.692	401.869	411.919
	100	971.067	983.469	778.475	804.833	769.967
Total	**186**	**1795.575**	**1841.370**	**1456.534**	**1490.848**	**1467.806**
Average	**31**	**299.262**	**306.895**	**242.756**	**248.475**	**244.634**

Table 5.1.8. AES Decryption Throughput for 1 - 100 MB data

	Data	Decryption Throughput				
	MB	Kb/sec				
		ECB	CBC	CFB	OFB	CTR
	1	1001.036	848.673	1059.446	1095.357	1015.754
	5	882.369	796.848	1007.632	1067.809	1103.774
	10	881.728	827.575	1025.281	1048.462	1048.728
	20	863.098	838.779	1036.913	1033.824	1013.385
	50	842.437	816.287	1048.477	1020.513	994.934
	100	843.702	833.194	1052.355	1019.877	1064.412
Total	**186**	**5314.372**	**4961.356**	**6230.104**	**6285.842**	**6240.987**
Average	**31**	**885.729**	**826.893**	**1038.351**	**1047.640**	**1040.165**

5.1.3 HARDWARE USAGE

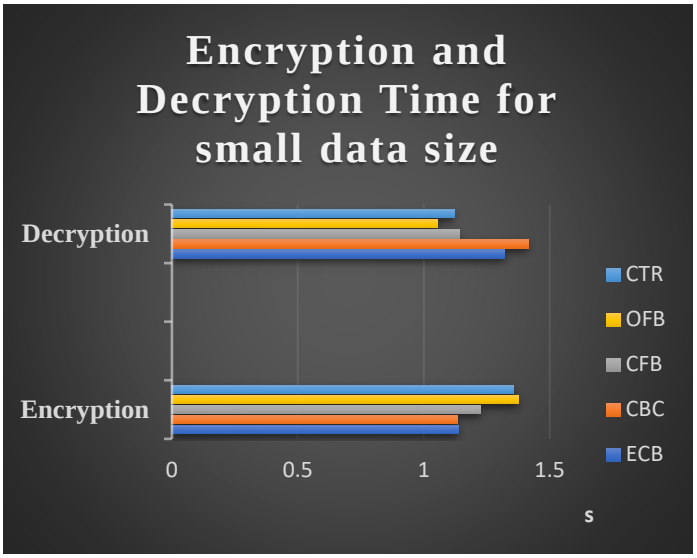
Table 5.1.9 shows the processes of algorithm and calculate the CPU and RAM that needed to encrypt 100KB data.

Table 5.1.9. AES Hardware Usage

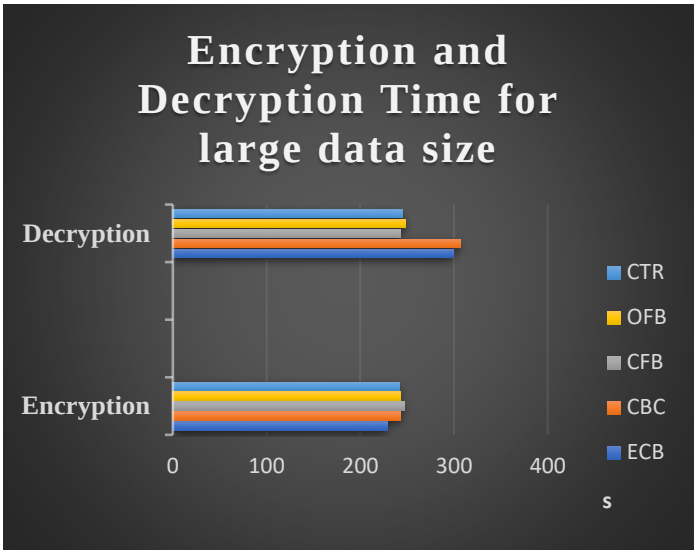
	Processes	CPU	MEMORY
AES	32	16.50%	1.4629

5.1.4 PERFORMANCE RESULTS SUMMARY

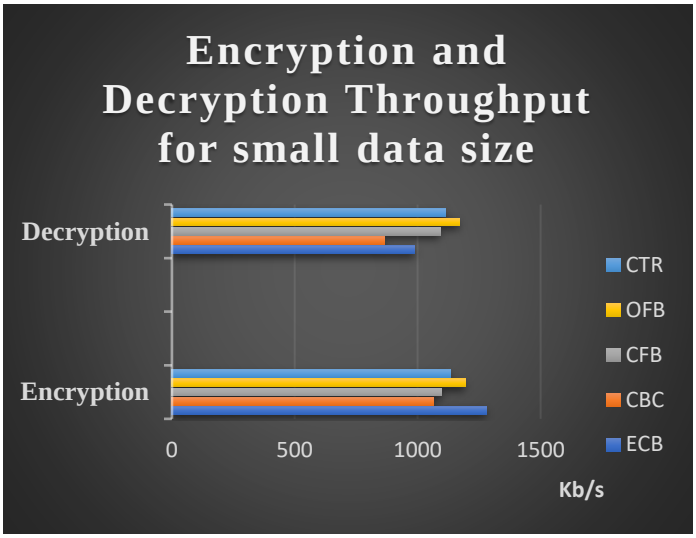
After presenting the encryption time, encryption throughput, decryption time, and decryption throughput. *Figure 5.1* summarize these results.



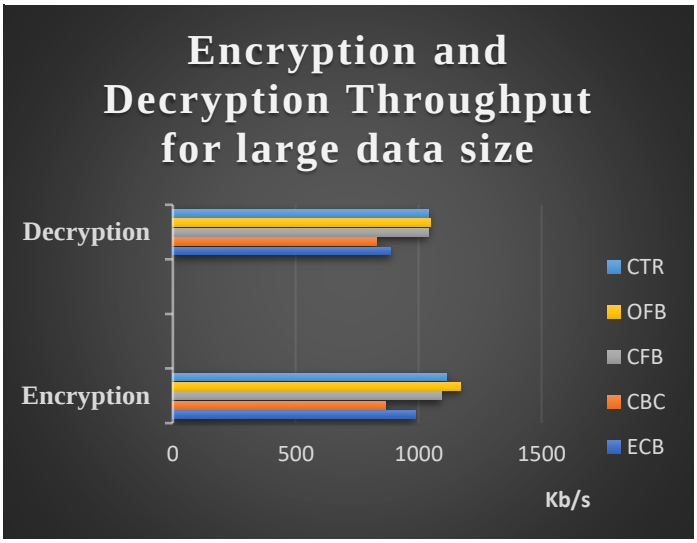
(a) Small Encryption and Decryption Time Summary



(b) Large Encryption and Decryption Time Summary



(c) Small Encryption and Decryption Throughput Summary



(d) Large Encryption and Decryption Throughput Summary

Figure 5.1. AES Performance Results Summary

- ECB mode provides the best encryption time, followed by OFB, CTR, CBC, and CFB mode, respectively, for small data size.

- ECB mode provides the best encryption time, followed by CTR, OFB, CBC, and CFB mode, respectively, for large data sizes.

- OFB mode provides the best decryption time, followed by CTR, CFB, ECB, and CBC mode, respectively, for small data size.

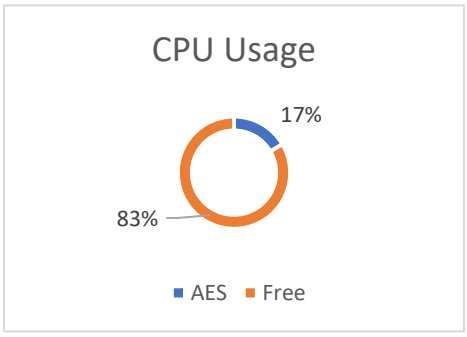
- CFB mode provides the best decryption time, followed by CTR, OFB, ECB, and CBC mode, respectively, for large data sizes.

- ECB mode provides the best encryption throughput, followed by OFB, CTR, CFB, and CBC mode, respectively, for small data size.

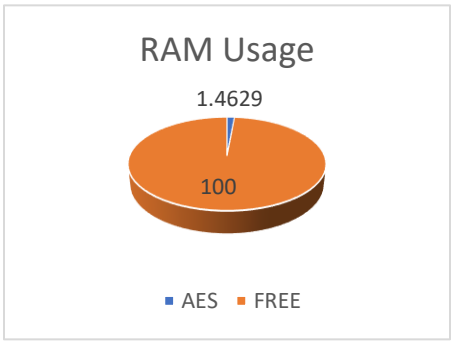
- ECB mode provides the best encryption throughput, followed by OFB, CTR, CBC, and CFB mode, respectively, for small data size.

- OFB mode provides the best decryption throughput followed by CTR, CFB, ECB, and CBC mode, respectively, for all data sizes.

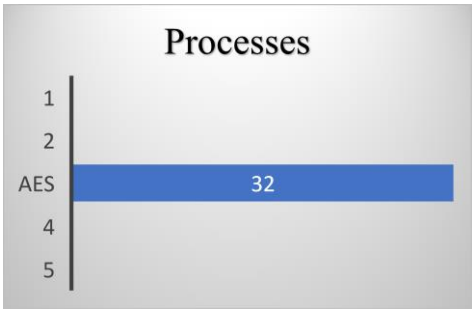
Figure 5.2 reflects the hardware used by AES to encrypt 100KB data.



(a). CPU Usage



(b) RAM Usage



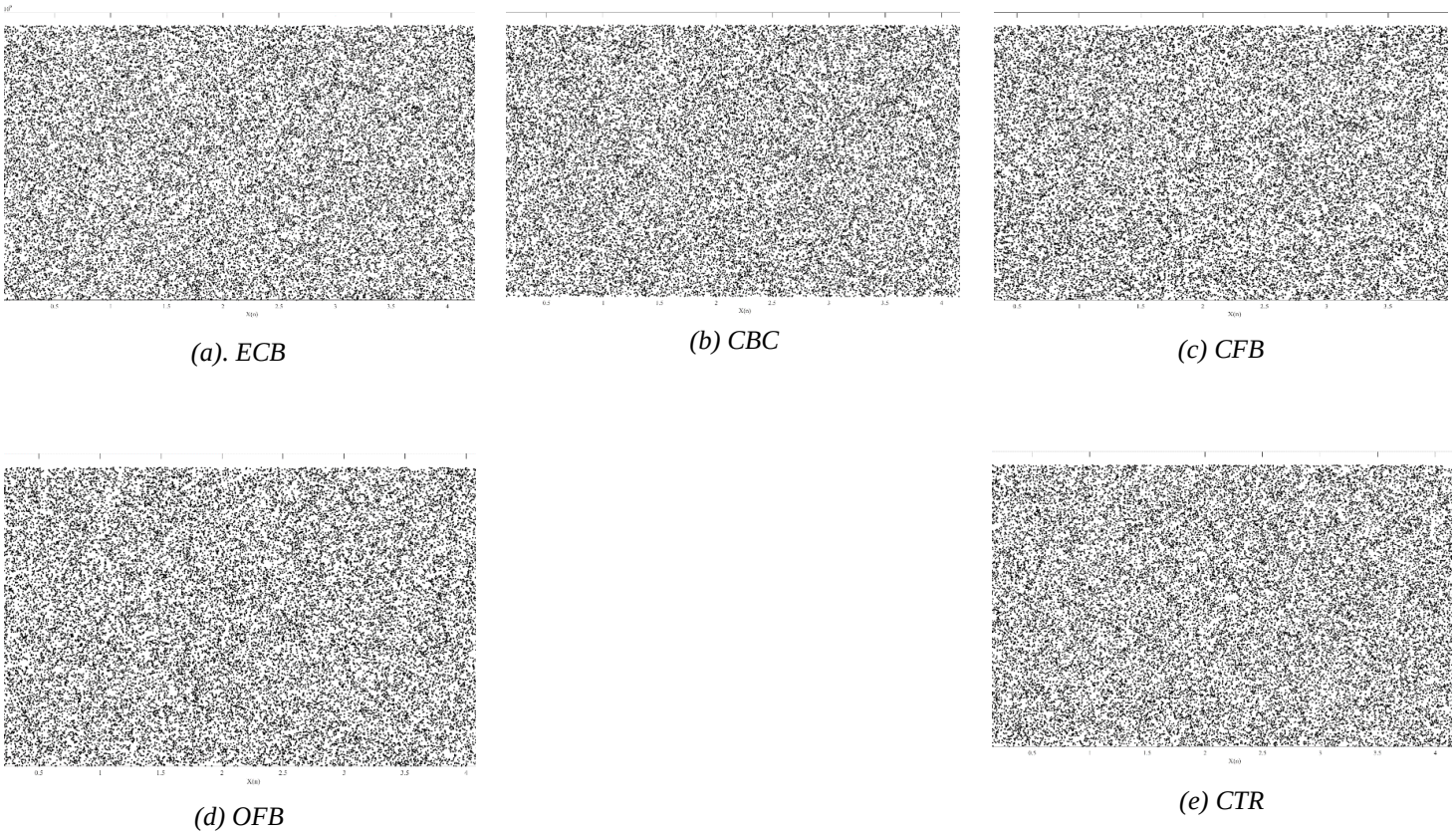
(c) # of Processes

Figure 5.2. AES Hardware Usage

5.2 SECURITY RESULTS

This section presents the results of security tests that tested the AES algorithm on different data sets.

5.2.1 MAPPING



(a). ECB *(b) CBC* *(c) CFB*

(d) OFB *(e) CTR*

Figure 5.3. AES Mapping Results

Figure 5.3 shows that all modes achieved good mapping results.

5.2.2 *HISTOGRAM AND CHI-SQUARE*

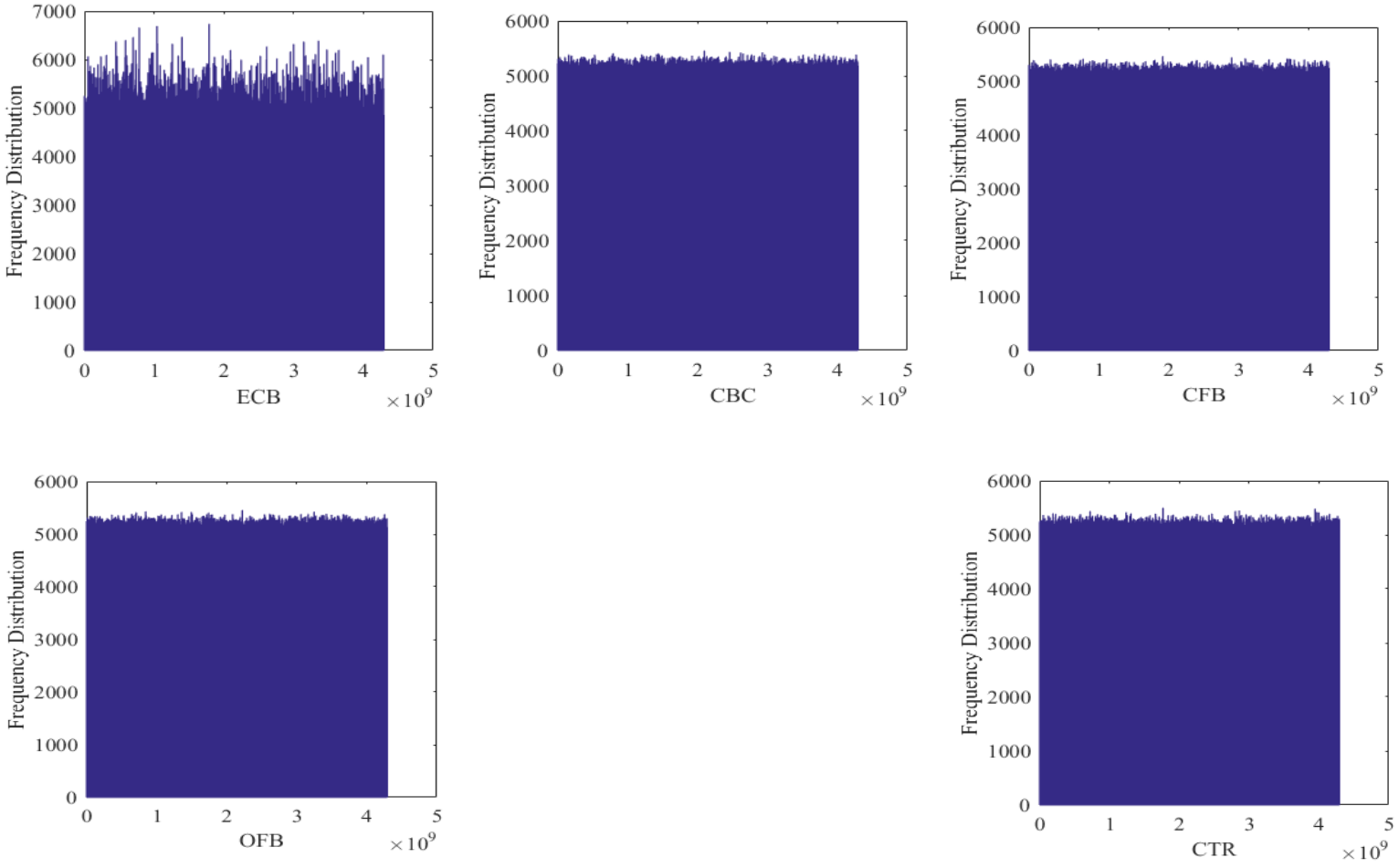


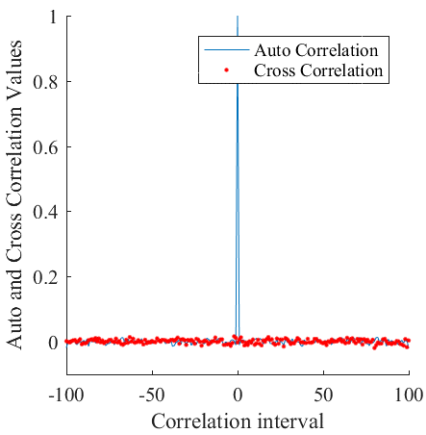
Figure 5.4. AES Histogram Results

Table 5.2.1. AES Chi-Square Test Results

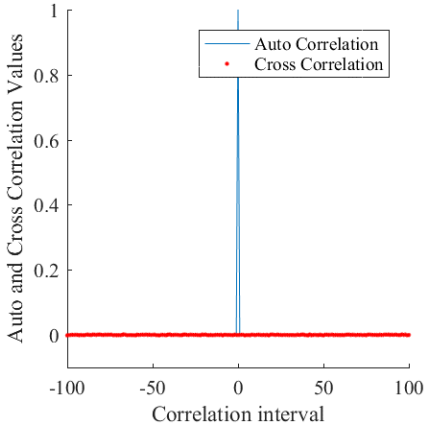
No of Classes = 256			**Alpha = 0.05**		
	ECB	**CBC**	**CFB**	**OFB**	**CTR**
Experimental Value	8194.444	240.912	251.804	274.401	257.436
Critical Value	293.248				
Chi-Square	Not Uniform	Uniform	Uniform	Uniform	Uniform

Figure 5.4 show the results of histogram of all modes, *Table 5.2.1* presents chi-square results, which reflect the distribution of data. From results we note that all modes have a uniform data distribution except ECB mode.

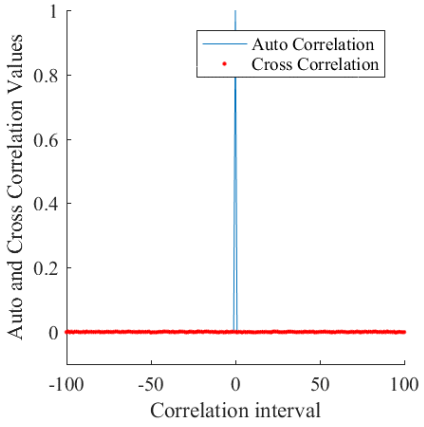
5.2.3 *CORRELATION*



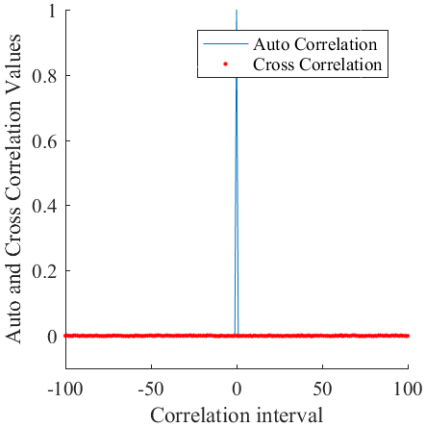
(a) ECB



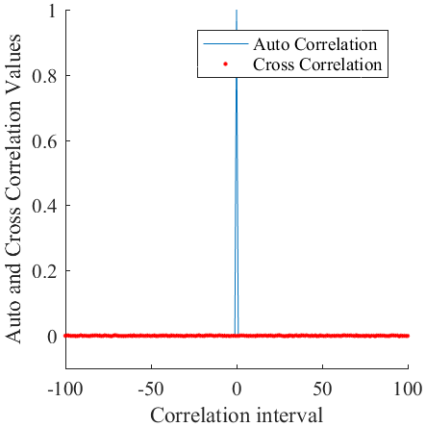
(b) CBC



(c) CFB



(d) OFB



(e) CTR

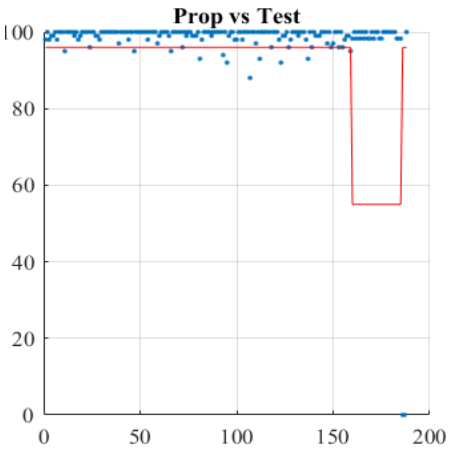
Figure 5.5. AES Auto and Cross Correlation Results

Table 5.2.2. AES Correlation Test Result

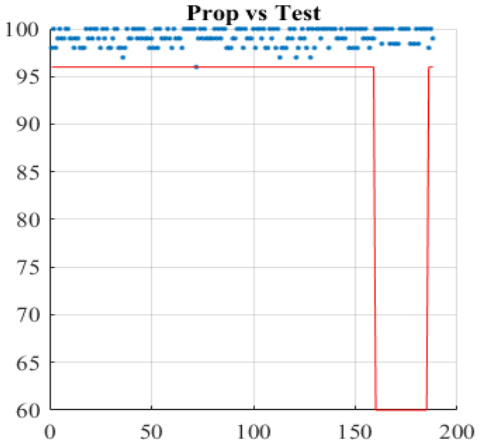
	ECB	**CBC**	**CFB**	**OFB**	**CTR**
Correlation Coefficient	-0.00155	-0.00010	0.00056	0.00050	-0.00029

Figure 5.5 shows the auto and cross correlation, based on results in *Table 5.2.2* we found that CBC provide best correlation result followed by CTR, OFB, and CFB mode respectively. ECB mode achieve result that is relatively far from other modes.

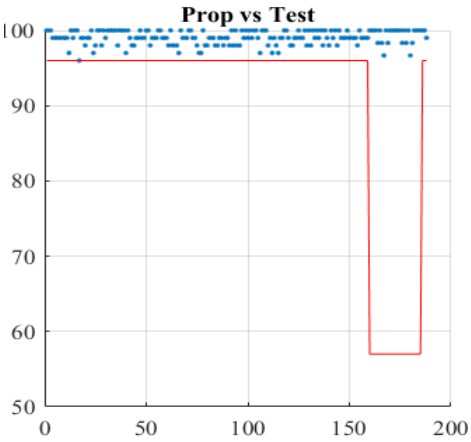
5.2.4 NIST



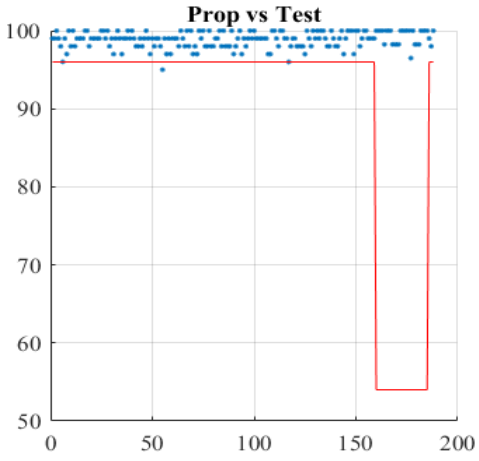
(a) ECB



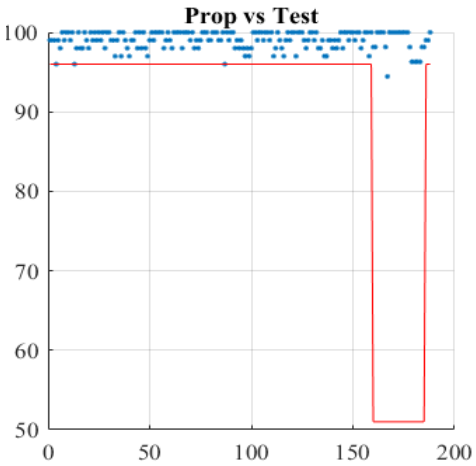
(b) CBC



(c) CFB



(d) OFB



(e) CTR

Figure 5.6. AES NIST Results

Table 5.2.3. AES P-Value for NIST Tests

Test		P - Value			
	ECB	**CBC**	**CFB**	**OFB**	**CTR**
Frequency	0.087	0.389	0.119	0.503	0.503
Block-Frequency	0.401	0.309	0.249	0.438	0.438
Cumulative-Sums	0.162	0.624	0.469	0.547	0.547
Runs	0.209	0.297	0.723	0.721	0.721
Longest-Run	0.276	0.556	0.468	0.682	0.682
Rank	0.225	0.596	0.757	0.664	0.664
FFT	0.678	0.627	0.493	0.558	0.558
Nonperiodic-Templates	0.141	0.463	0.529	0.490	0.490
Overlapping-Templates	0.837	0.730	0.310	0.507	0.507
Universal	0.423	0.149	0.340	0.336	0.336
Approximate Entropy	0.000	0.765	0.455	0.316	0.316
Random-Excursions	0.366	0.479	0.422	0.397	0.397
Random-Excursions-Variant	0.370	0.426	0.491	0.442	0.442
Serial	0.000	0.628	0.425	0.361	0.361
Linear-Complexity	0.512	0.076	0.517	0.724	0.724

Table 5.2.4. AES NIST Tests Results

Test		Probability			
	ECB	**CBC**	**CFB**	**OFB**	**CTR**
Frequency	99.333	99.000	97.667	99.000	99.000
Block-Frequency	99.667	98.333	99.667	99.667	99.667
Cumulative-Sums	99.500	99.000	97.333	99.333	99.333
Runs	99.000	99.667	99.000	99.000	99.000
Longest-Run	99.333	99.333	99.000	97.667	97.667
Rank	99.000	99.000	99.000	98.667	98.667
FFT	99.667	98.667	99.333	97.667	97.667
Nonperiodic-Templates	98.833	99.000	98.872	98.982	98.982
Overlapping-Templates	99.000	98.667	98.667	98.667	98.667
Universal	99.333	98.000	98.667	98.000	98.000
Approximate Entropy	87.667	98.667	99.333	98.667	98.667
Random-Excursions	98.706	98.839	99.124	99.505	99.505
Random-Excursions-Variant	98.816	99.596	99.461	99.256	99.256
Serial	0.667	99.333	99.667	98.500	98.500
Linear-Complexity	100.000	99.667	99.333	99.667	99.667
Pass Rate except RE-Variant	**96**				
Pass Rate of RE-Variant	**54.667**	**56.667**	**54.667**	**53.667**	**58.667**

Figure 5.6 shows the visual results of all 188 NIST tests and subtests. While *Table 5.2.3* presents the P-Value of main 15 Test and *Table 5.2.4* presents the Results of main 15 NIST tests.

- From these results, we conclude that All modes have pass all NIST tests except ECB mode, which failed in Entropy and Serial tests.

5.2.5 CONFUSION

5.2.5.1 ONE BIT CHANGE

In *Table 5.2.5*, we present the results of hamming distance according to change one bit in the Key for small data size, while *Table 5.2.6* represents it for large data sizes.

Table 5.2.5. AES Confusion Results using 1 Bit Change for 10-500 KB Data

Data Size KB		ECB (%)	CBC (%)	CFB (%)	OFB (%)	CTR (%)
10	First	49.987	49.977	50.005	50.245	49.942
	Middle	50.104	50.091	49.948	49.897	49.938
	Last	50.156	50.036	49.830	50.178	49.963
Average		50.082	50.035	49.928	50.106	49.948
100	First	50.029	50.056	49.930	49.949	50.006
	Middle	50.072	49.928	50.060	49.926	49.976
	Last	49.919	49.992	49.980	50.077	49.967
Average		50.007	49.992	49.990	49.984	49.983
500	First	49.971	49.980	49.972	49.945	50.017
	Middle	50.061	49.985	49.990	49.973	49.991
	Last	49.926	49.968	49.986	49.982	49.974
Average		49.986	49.978	49.983	49.967	49.994

Table 5.2.6. AES Confusion Results using 1 Bit Change for 1-20 MB Data

Data Size MB		ECB (%)	CBC (%)	CFB (%)	OFB (%)	CTR (%)
1	First	49.957	49.992	49.976	49.996	50.008
	Middle	50.074	49.984	50.008	49.981	50.006
	Last	49.953	49.990	49.982	49.985	49.995
Average		49.995	49.989	49.989	49.987	50.003
5	First	49.976	50.001	49.992	50.008	50.011
	Middle	50.108	50.013	50.015	49.994	50.008
	Last	49.930	49.996	49.998	49.981	50.002
Average		50.005	50.003	50.002	49.995	50.007
20	First	49.968	50.002	49.999	50.008	50.001
	Middle	50.112	50.005	50.009	49.993	50.005
	Last	49.945	50.001	50.008	49.996	50.004
Average		50.008	50.003	50.005	49.999	50.003

5.2.5.2 TWO BIT CHANGE

Table 5.2.7 presents the results of hamming distance according to change two bits in the Key for small data size, while *Table 5.2.8* represents it for large data sizes.

Table 5.2.7. AES Confusion Results using 2 Bit Change for 10-500 KB Data

Data Size KB		ECB (%)	CBC (%)	CFB (%)	OFB (%)	CTR (%)
10	First	49.826	49.477	49.535	49.560	49.979
	Middle	49.659	50.008	49.788	50.060	49.946
	Last	49.477	50.126	49.880	49.832	50.249
Average		49.654	49.870	49.734	49.817	50.058
100	First	49.927	49.893	50.002	49.906	49.942
	Middle	49.990	49.955	49.920	50.058	49.991
	Last	49.831	50.029	49.990	49.952	50.003
Average		49.916	49.959	49.971	49.972	49.979
500	First	49.986	49.945	49.997	49.993	49.985
	Middle	50.023	50.018	49.973	50.002	49.991
	Last	49.789	49.996	49.991	50.051	50.043
Average		49.933	49.986	49.987	50.015	50.006

Table 5.2.8. AES Confusion Results using 2 Bit Change for 1-20 MB Data

Data Size MB		ECB (%)	CBC (%)	CFB (%)	OFB (%)	CTR (%)
1	First	49.961	49.965	49.980	49.993	50.012
	Middle	50.040	50.011	49.993	50.007	50.007
	Last	49.783	50.028	49.984	50.041	50.026
Average		49.928	50.001	49.986	50.014	50.015
5	First	49.936	50.004	49.997	49.997	49.991
	Middle	50.056	49.997	49.996	50.010	50.013
	Last	49.852	50.002	49.992	50.014	50.010
Average		49.948	50.001	49.995	50.007	50.005
20	First	49.934	50.000	50.000	50.002	49.997
	Middle	50.071	49.996	50.004	50.004	50.001
	Last	49.867	49.997	49.999	49.996	50.001
Average		49.958	49.998	50.001	50.001	50.000

5.2.5.3 THREE BIT CHANGE

In *Table 5.2.9*, we present the results of hamming distance according to change one bit in the Key for small data size, while *Table 5.2.10* represents it for large data sizes.

Table 5.2.9. AES Confusion Results using 3 Bit Change for 10-500 KB Data

Data Size KB		ECB (%)	CBC (%)	CFB (%)	OFB (%)	CTR (%)
10	First	49.925	49.970	49.941	50.163	49.524
	Middle	49.780	49.861	50.067	49.788	49.998
	Last	49.717	49.954	49.929	49.843	49.930
Average		49.807	49.929	49.979	49.931	49.817
100	First	50.072	50.138	49.937	50.005	49.917
	Middle	49.926	50.018	50.003	49.973	50.006
	Last	49.932	49.936	50.101	49.985	50.157
Average		49.976	50.031	50.014	49.988	50.027
500	First	50.023	50.007	49.970	49.959	49.941
	Middle	49.963	49.990	49.998	49.987	50.015
	Last	49.942	49.939	50.006	49.990	50.041
Average		49.976	49.979	49.991	49.979	49.999

Table 5.2.10. AES Confusion Results using 3 Bit Change for 1-20 MB Data

Data Size MB		ECB (%)	CBC (%)	CFB (%)	OFB (%)	CTR (%)
1	First	50.035	49.994	49.977	49.980	49.976
	Middle	49.967	49.991	50.016	49.987	50.002
	Last	49.942	49.955	50.010	49.997	50.022
Average		49.981	49.980	50.001	49.988	50.000
5	First	50.047	49.996	49.993	50.000	49.997
	Middle	49.985	50.006	49.997	49.989	50.002
	Last	49.921	49.994	49.999	50.005	49.998
Average		49.984	49.999	49.997	49.998	49.999
20	First	50.042	49.994	49.999	49.999	50.001
	Middle	49.994	50.001	49.996	50.000	49.998
	Last	49.907	49.996	50.000	49.996	49.997
Average		49.981	49.997	49.998	49.998	49.998

5.2.6 DIFFUSION

In *Table 5.2.11*, we present the results of hamming distance according to change one bit in the plaintext for small data size, while *Table 5.2.12* represents it for large data sizes.

Table 5.2.11. AES Diffusion Results using 1 Bit Change for 10-500 KB Data

Data Size KB		ECB (%)	CBC (%)	CFB (%)	OFB (%)	CTR (%)
10	First	0.08179	50.15808	49.82225	0.00367	0.00367
	Middle	0.06714	32.76285	32.42685	0.00611	0.00611
	Last	0.07202	0.08667	0.00489	0.00489	0.00489
Average		0.07365	27.66920	27.41800	0.00489	0.00489
100	First	0.00832	49.99920	49.89309	0.00024	0.00024
	Middle	0.00698	25.00370	24.98118	0.00049	0.00049
	Last	0.00759	0.00771	0.00061	0.00061	0.00061
Average		0.00763	25.00354	24.95829	0.00045	0.00045
500	First	0.00159	50.00765	49.98292	0.00007	0.00007
	Middle	0.00147	23.20192	23.17237	0.00005	0.00005
	Last	0.00147	0.00147	0.00005	0.00005	0.00005
Average		0.00151	24.40368	24.38511	0.00006	0.00006

Table 5.2.12. AES Diffusion Results using 1 Bit Change for 1-20 MB Data

Data Size MB		ECB (%)	CBC (%)	CFB (%)	OFB (%)	CTR (%)
1	First	0.0008129	50.0176744	49.9812855	0.0000239	0.0000239
	Middle	0.0008248	25.0883453	25.0884533	0.0000239	0.0000239
	Last	0.0006694	0.0007531	0.0000239	0.0000239	0.0000239
Average		0.0007691	25.0355910	25.0232542	0.0000239	0.0000239
5	First	0.0001435	50.0034345	49.9875757	0.0000072	0.0000072
	Middle	0.0001698	24.8697684	24.8806801	0.0000120	0.0000120
	Last	0.0001482	0.0001387	0.0000072	0.0000072	0.0000072
Average		0.0001538	24.9577805	24.9560877	0.0000088	0.0000088
20	First	0.0000389	49.9983231	49.9975909	0.0000024	0.0000024
	Middle	0.0000353	25.0350126	25.0351059	0.0000018	0.0000018
	Last	0.0000389	0.0000371	0.0000018	0.0000018	0.0000018
Average		0.0000377	25.0111243	25.0108995	0.0000020	0.0000020

5.2.6.1 CONFUSION AND DIFFUSION SUMMARY

In confusion and diffusion tests, we made the changes of bits in three different places, for confusion we change 1-3 bits in key. But, in diffusion we just change one bit since the size of plaintext is much greater than the key length. From the previous tables we conclude that:

- Confusion and Diffusion results does not affect by file size.
- Diffusion results affected by the place of bit that changed in plaintext, the results show that change the first bit give best result.
- The place of bit changed in key does not affect the confusion results.
- The number of bit changes has no mean in diffusion. Changing two, three, or ten bit is the same since the plaintext is much larger than the changed bits.
- All modes provide excellent confusion results.
- Just CBC and CFB modes give perfect diffusion results.

5.2.7 KEY SECURITY

AES has a key security with complexity 2^{128}, which make AES is secure against brute force attack.

5.2.8 SECURITY RESULTS SUMMARY

After presenting the results of Mapping, Histogram and Chi-Square, Correlation, NIST, Confusion and Diffusion, and Key security test. These results summarized as follow:

- AES pass Mapping test in all modes.
- AES pass Histogram test in all modes except ECB.
- AES pass Correlation test in all modes except ECB.
- AES pass NIST test in all modes except ECB.
- AES pass Confusion test in all modes.
- AES pass Diffusion test only in CBC and CFB modes.
- AES Key is secure against brute force attack with complexity $\approx 2^{128}$.

5.3 AES EVALUATION SUMMARY

In this section, we summarize the result of testing the standard AES. We define some variable to facilitate the comparison process as follow:

- a, b, X_i: it indicates to the size of data. Where *(a) = 155KB*, *(b) = 31MB*, and *(x)* is the *encryption time*.
- W: it is a score take a value from a specific range.
    - In performance, the range from 1 to 8 according to the number of AES versions that implemented. The values generated by changing the range to which the values belong in fixed proportions.
- G: the score of each test in the range of W.

- The improvement percentage calculated by measure the gain based on the following formula:

$$Gain = \pm \frac{Scenario\ Score - AES\ Score}{AES\ Score} * 100\% \qquad (11)$$

  - (+) indicate that we have an improvement in results, while (–) indicate that the results have been drawn back.

- The Security Score in Security Summary for each mode is Calculated based on formula 12:

$$Mode\ Security\ Score = \sum_{Mapping}^{NIST} G \qquad (12)$$

- The total Security Score for each Scenario is calculated based on formula 13:

$$Security\ score = \frac{(CBC\&CFB\ score) + ECB\ score + OFB\ score + CTR\ core}{4} \qquad (13)$$

Where,

$$CBC\&CFB\ score = \frac{CBC\ score + CFB\ score}{2} \qquad (14)$$

Table 5.3.1 summarizes the security tests results obtained from this implementation, while *Table 5.3.2* summarize the performance tests results.

Table 5.3.1. AES Security Tests Results Summary

	W		ECB	G	CBC	G	CFB	G	OFB	G	CTR	G
Mapping	1		uniform	1	uniform	1	uniform	1	uniform	1	uniform	1
Histogram	1		8194.444	0	240.912	1	251.804	1	274.401	1	257.439	1
Correlation	1		0.00155	0	0.00010	1	0.00056	1	0.00050	1	0.00029	1
Confusion	1	a	50.082	1	50.035	1	49.928	1	50.106	1	49.948	1
	1	b	50.008	1	50.003	1	50.005	1	49.999	1	50.003	1
Diffusion	1	a	0.082	0	50.158	1	49.822	1	0.004	0	0.004	0
	1	b	0.00003	0	49.998	1	49.998	1	0.000002	0	0.000002	0
Key Security	1	$2^{128} * x_{AES}$										1
NIST	15	13			15		15		15		15	
Security Score	**23**	**17**			**23**		**23**		**21**		**21**	

Table 5.3.2. AES Performance Tests Results Summary

	G		ECB	CBC	CFB	OFB	CTR	G
Enc. Time	8	a	1.136	1.132	1.224	1.374	1.355	1
	8	b	228.899	243.123	246.811	242.619	241.472	1
Dec. Time	8	a	1.321	1.414	1.141	1.054	1.120	1
	8	b	299.262	306.895	242.756	248.475	244.634	1
Enc. Throughput	8	a	1278.240	1063.874	1098.200	1192.912	1132.802	1
	8	b	1134.554	1037.381	1028.120	1051.146	1046.477	1
Dec. Throughput	8	a	987.111	866.452	1091.376	1169.090	1113.619	1
	8	b	885.729	826.893	1038.351	1047.640	1040.165	1
# Of Processes	8	a	32					1
CPU Usage	8	a	16.5%					4
RAM Usage	8	a	1.463					1
Performance Score	**88**		**14**					

CHAPTER 6

AES EXPLORING

6 AES EXPLORING

Based on the time-consuming analysis of the AES algorithm, we found that a large proportion of the time is consumed in the number of rounds and MixColumns operations.

By running AES, we found that MixColumns operation take about 50% of the running time, while rounds take about 77% of running time. Therefore, these two factors have been chosen as critical points to be manipulated for the performance improvement to be of good value and noticeable. *Figure 6.1* present the time analysis of AES algorithm.

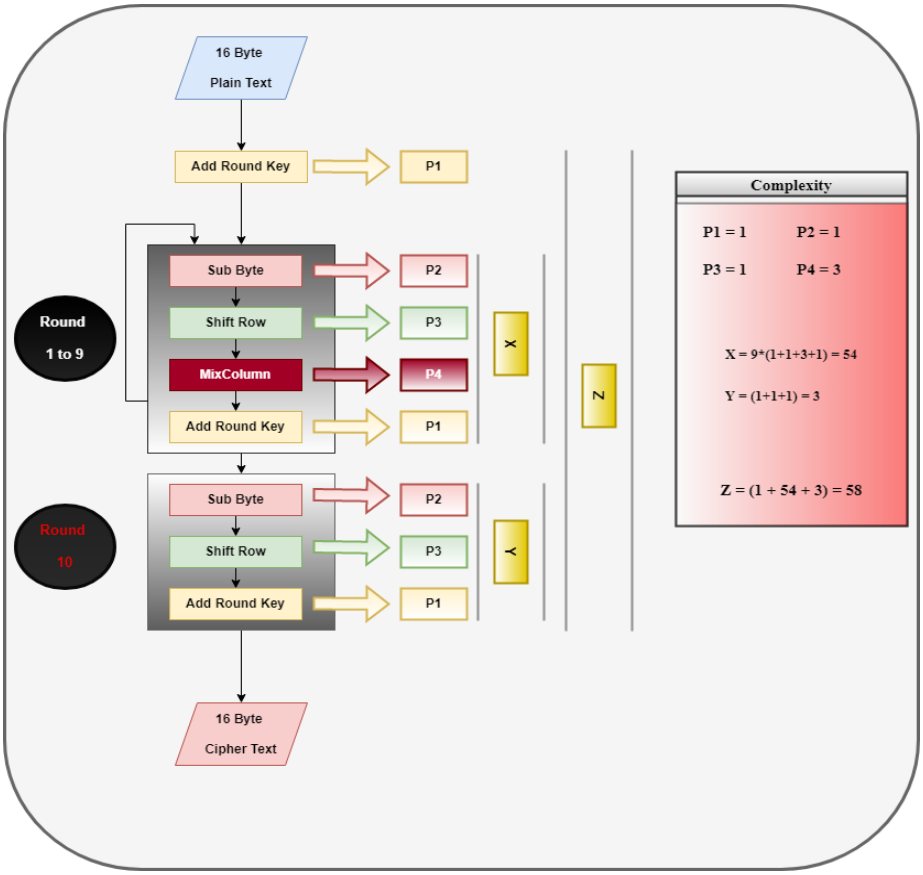


Figure 6.1. AES Algorithm Time Analysis

In this chapter, we implement some AES scenarios and discuss their results in term of performance and security in section 6.1. While in section 6.2 we test the standard AES and the best case of explored scenarios on RP in term of performance.

6.1 EXPLORED SCENARIOS

In this section we discuss the results of testing some AES scenarios. These scenarios focus on changing the number of rounds, MixColumns operation, or both. This chapter presents the result of performance and security for all modes of operation. These Scenarios are:

- 10-Rounds AES with Half MixColumns(10H).
- 10-Rounds AES without MixColumns(10N).
- 5-Rounds AES with Half MixColumns(5H).
- 5-Rounds AES without MixColumns(5N).
- 5-Rounds AES with MixColumns(5F).
- 3-Rounds AES with MixColumns(3F).
- 2-Rounds AES with MixColumns(2F).

The Data proceed as a 4*4 Matrix (M) in each operation. After the ShiftRow operation, each column of the Matrix is multiplied by a fixed matrix in the MixColumn operation to generate a new Matrix (M'). *Figure 6.2* presents the process of MixColumn operation.

Without MixColumns means completely eliminating MixColumns, while Half-MixColumns means that the MixColumns are applied only on two columns of ciphertext matrix instead of the four columns.

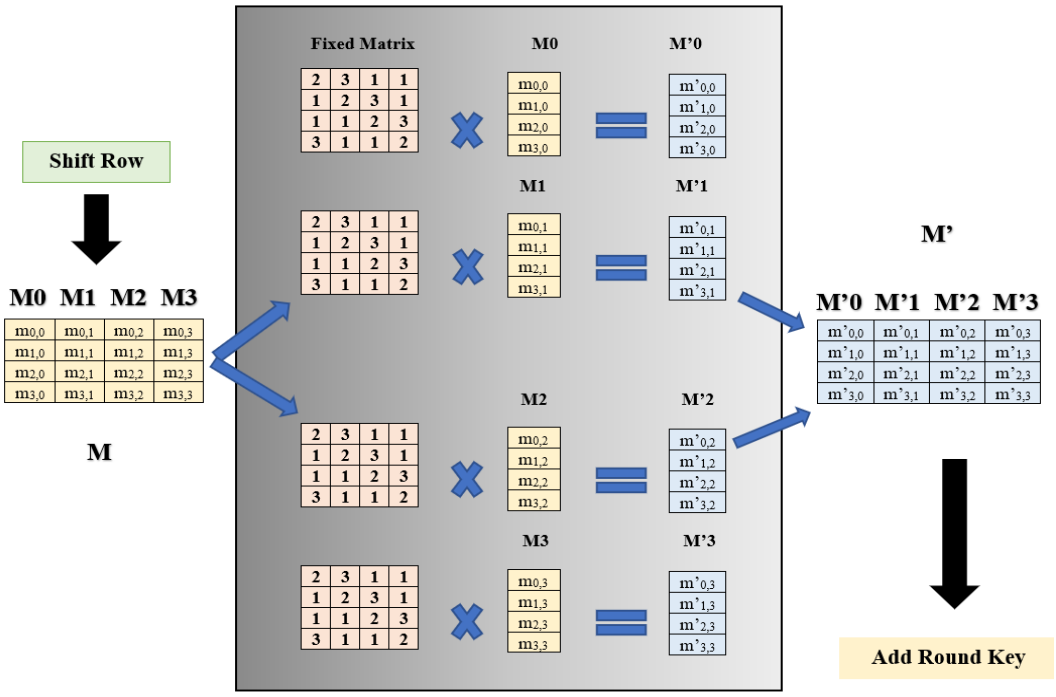


Figure 6.2. MixColumn Operation

In Half-MixColumn, we mean that not all M columns will be multiplied by the fixed matrix. Only two columns will proceed in the MixColumn operation, and the other two columns will transform to the M' without any changes. We explored all possible possibilities of the chosen column. *Figure 6.3* presents the process of Half-MixColumn operation.

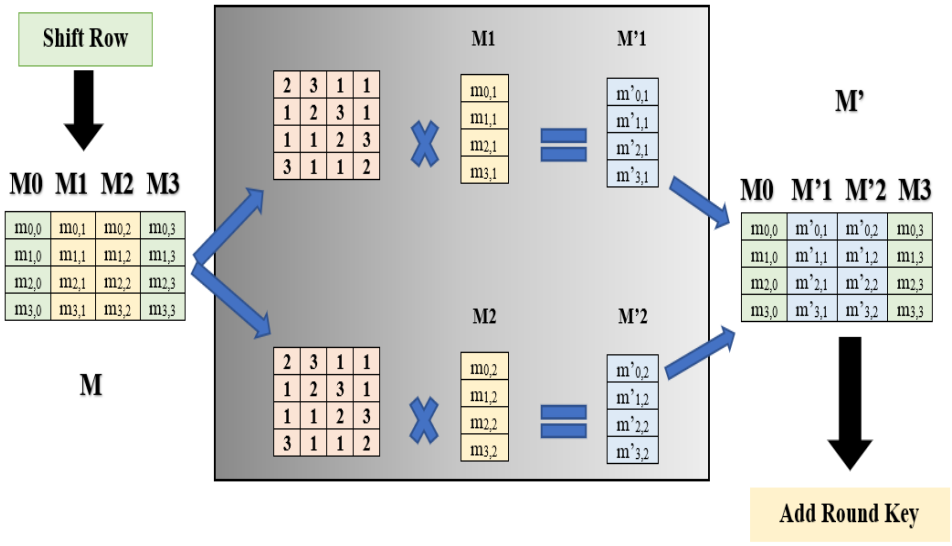


Figure 6.3. MixColumn Operation

6.1.1 10-ROUNDS AES WITH HALF MIXCOLUMNS

In this section, we implement the AES by modifying the MixColumns to operate on two columns instead of four with same rounds of Standard AES. *Table 6.1.1* presents the security tests results obtained from this implementation, while *Table 6.1.2* presents the performance tests results.

Table 6.1.1. 10H Security Tests Results Summary

	G		ECB	G	CBC	G	CFB	G	OFB	G	CTR	G
Mapping	1		Not uniform	0	uniform	1	uniform	1	uniform	1	Not uniform	1
Histogram	1		4509320	0	273.755	1	252.391	1	28428	0	3865284	0
Correlation	1		0.00562	0	0.00028	1	-0.00022	1	-0.00253	0	0.15493	0
Confusion	1	a	49.684	1	49.976	1	50.000	1	49.977	1	50.004	1
	1	b	49.746	1	49.997	1	50.000	1	50.007	1	49.996	1
Diffusion	1	a	0.00049	0	3.15021	0	3.15003	0	0.00024	0	0.00024	0
	1	b	0.000010	0	3.137280	0	3.136115	0	0.000005	0	0.000005	0
Key Security	5	$2^{128} * x_{10H}$										1
NIST	15	7		15		15		14		9		
Security Score	**23**	**10**		**21**		**21**		**18**		**13**		

Table 6.1.2. 10H Performance Tests Results Summary

	G		ECB	CBC	CFB	OFB	CTR	G
Enc. Time	8	a	1.002	0.886	0.916	0.904	0.989	2
	8	b	170.881	179.541	186.285	175.967	182.550	3
Dec. Time	8	a	1.077	1.035	0.903	0.849	0.944	3
	8	b	205.945	215.934	177.390	182.761	180.659	4
Enc. Throughput	8	a	1266.963	1372.820	1378.628	1405.832	1379.977	1
	8	b	1425.770	1392.339	1351.712	1449.816	1390.555	2
Dec. Throughput	8	a	1110.221	1197.534	1370.828	1442.922	1415.183	2
	8	b	1238.786	1160.071	1414.753	1407.890	1400.626	2
# Of Processes	8	a	26					3
CPU Usage	8	a	17.45%					1
RAM Usage	8	a	1.392					6
Performance Score	**88**	**29**						

Compared to AES, in the case of studying security, from *Table 6.1.1* we found that:

- In addition to ECB mode, 10H fails in the CTR mode mapping test, while it passes the mapping test in all other modes.
- 10H passes Histogram test only in CBC and CFB modes.
- Only CBC and CFB pass the correlation test with accepted results.

- 10H passes the confusion test in All modes.
- 10H failed in the diffusion test in all modes.
- Only CBC and CFB modes pass all NIST tests.
- OFB mode has good NIST results.
- 10H maintains the same level of key security compared with AES.

In the case of studying performance, from *Table 6.1.2*, we found that:

- 10H provides better encryption and decryption time with a 44.25% improvement.
- 10H provides better encryption and decryption throughput with a 79.75% improvement.
- 10H requires a smaller number of processes with an 18.75% improvement.
- 10H requires less RAM with a 4.81% improvement but requires more a 5.74% CPU.

6.1.2 10-ROUNDS AES WITHOUT MIXCOLUMNS

In this section, we implement the AES by ignoring the MixColumns operation with same rounds of Standard AES. *Table 6.1.3* presents the security tests results obtained from this implementation, while *Table 6.1.4* presents the performance tests results.

Table 6.1.3. 10N Security Tests Results Summary

	G		ECB	G	CBC	G	CFB	G	OFB	G	CTR	G
Mapping	1		Not uniform	0	uniform	1	uniform	1	uniform	1	Not uniform	0
Histogram	1		11393615	0	234.330	1	246.861	1	226787	0	15697125	0
Correlation	5		-0.11289	0	0.00038	1	0.00071	1	0.00198	0	-0.30794	0
Confusion	1	a	49.925	1	49.969	1	50.016	1	49.973	1	52.847	0.75
	1	b	49.949	1	49.995	1	49.991	1	49.984	1	52.615	0.75
Diffusion	1	a	0.00049	0	3.15021	0	3.15003	0	0.00024	0	0.00024	0
	1	b	0.000010	0	3.137280	0	3.136115	0	0.000005	0	0.000005	0
Key Security	5	$2^{128} * x_{10N}$										1
NIST	15	5		15		15		8		5		
Security Score	**23**	**8**		**21**		**21**		**12**		**7.5**		

Table 6.1.4. 10N Performance Tests Results Summary

	G		ECB	CBC	CFB	OFB	CTR	G
Enc. Time	8	a	0.673	0.686	0.665	0.701	0.733	4
	8	b	125.727	135.063	131.104	131.788	129.354	5
Dec. Time	8	a	0.637	0.645	0.651	0.683	0.785	5
	8	b	131.038	134.813	132.223	129.853	133.546	5
Enc. Throughput	8	a	1893.364	1952.142	1868.477	1844.248	1763.085	3
	8	b	1992.391	1799.677	1908.993	1864.707	1906.061	2
Dec. Throughput	8	a	1925.548	1963.261	1958.766	1772.659	1766.501	3
	8	b	2020.852	1858.472	1873.850	1924.696	1880.749	3
# Of Processes	8	a	21					5
CPU Usage	8	a	16.39%					4
RAM Usage	8	a	1.368					8
Performance Score	**88**		**47**					

Compared to AES, in the case of studying security, from *Table 6.1.3* we found that:

- In addition to ECB mode, 10N fails in the CTR mode mapping test, while it passes the mapping test in all other modes.
- 10N passes Histogram test only in CBC and CFB modes.
- Only CBC and CFB pass the correlation test with accepted results.
- 10N passes the confusion test in All modes.
- 10N failed in the diffusion test in all modes.
- Only CBC and CFB modes pass all NIST tests.
- OFB mode has good NIST results.
- 10N maintains the same level of key security compared with AES.

In the case of studying performance, from *Table 6.1.4*, we found that:

- 10N provides better encryption and decryption time with a 22% improvement.
- 10N provides better encryption and decryption throughput with 28% improvement.
- 10N requires a smaller number of processes with a 34.38% improvement.
- 10N requires less CPU and RAM usage with a 0.69% and 6.46% improvement, respectively.

6.1.3 5-ROUNDS AES WITH HALF MIXCOLUMNS

In this section, we implement the AES by modifying the MixColumns to operate on two columns instead of four with five rounds of Standard AES. *Table 6.1.5* presents the security tests results obtained from this implementation, while *Table 6.1.6* presents the performance tests results.

Table 6.1.5. 5H Security Tests Results Summary

	G		ECB	G	CBC	G	CFB	G	OFB	G	CTR	G
Mapping	1		Not uniform	0	uniform	1	uniform	1	uniform	1	Not uniform	0
Histogram	1		5619513	0	256.552	1	247.859	1	101262	0	3623425	0
Correlation	5		-0.03932	0	-0.00036	1	0.00025	1	-0.00017	1	0.38144	0
Confusion	1	a	50.223	1	49.986	1	50.053	1	49.989	1	50.403	1
	1	b	50.215	1	50.001	1	50.008	1	49.963	1	50.488	1
Diffusion	1	a	0.00049	0	3.13565	0	3.10854	0	0.00024	0	0.00024	0
	1	b	0.000010	0	3.137445	0	3.135157	0	0.000005	0	0.000005	0
Key Security	5	$2^{128} * x_{5H}$										1
NIST	15	5		15		15		10		4		
Security Score	**23**	**8**		**21**		**21**		**15**		**7**		

Table 6.1.6. 5H Performance Tests Results Summary

	G		ECB	CBC	CFB	OFB	CTR	G
Enc. Time	8	a	0.460	0.470	0.464	0.439	0.467	6
	8	b	88.213	93.950	90.322	89.795	92.225	7
Dec. Time	8	a	0.572	0.538	0.453	0.484	0.496	6
	8	b	99.810	103.367	89.799	91.383	92.630	7
Enc. Throughput	8	a	2870.042	2672.889	2765.881	2841.733	2545.044	4
	8	b	2978.902	2719.274	2727.845	2849.535	2746.532	4
Dec. Throughput	8	a	2391.089	2279.549	2825.004	2728.629	2388.556	4
	8	b	2570.906	2436.613	2836.754	2804.326	2688.366	4
# Of Processes	8	a	17					6
CPU Usage	8	a	15.99%					5
RAM Usage	8	a	1.443					2
Performance Score	**88**	**55**						

Compared to AES, in the case of studying security, from *Table 6.1.5*, we found that:

- In addition to ECB mode, 5H fails in the CTR mode mapping test, while it passes the mapping test in all other modes.
- 5H passes Histogram test only in CBC and CFB modes.
- Only ECB and CTR modes failed in the correlation test.

- 5H passes the confusion test in All modes.
- 5H failed in the diffusion test in all modes.
- Only CBC and CFB modes pass all NIST tests.
- 5H maintains the same level of key security compared with AES.

In the case of studying performance, from *Table 6.1.6*, we found that:

- 5H provides better encryption and decryption time with a 60.25% improvement.
- 5H provides better encryption and decryption throughput with a 155% improvement.
- 5H requires a smaller number of processes with a 46.88% improvement.
- 5H requires less CPU and RAM usage with a 3.13% and 1.34% improvement, respectively.

6.1.4 5-ROUNDS AES WITHOUT MIXCOLUMNS

In this section, we implement the AES by ignoring the MixColumns operation with five rounds of Standard AES. *Table 6.1.7* presents the security tests results obtained from this implementation, while *Table 6.1.8* presents the performance tests results.

Table 6.1.7. 5N Security Tests Results Summary

	G		ECB	G	CBC	G	CFB	G	OFB	G	CTR	G
Mapping	1		Not uniform	0	uniform	1	uniform	1	uniform	1	Not uniform	0
Histogram	1		11440429	0	273.188	1	245.514	1	52580	0	18297198	0
Correlation	5		0.05409	0	0.00003	1	0.00047	1	-0.00097	1	0.10062	0
Confusion	1	a	51.619	0.75	49.970	1	49.967	1	49.896	1	49.860	1
	1	b	51.639	0.75	49.999	1	49.993	1	49.861	1	50.361	1
Diffusion	1	a	0.00049	0	3.13565	0	3.10854	0	0.00024	0	0.00024	0
	1	b	0.000010	0	3.137445	0	3.135157	0	0.000005	0	0.000005	0
Key Security	5	$2^{128} * x_{5N}$										1
NIST	15	6			15		15		12		6	
Security Score	**23**	**8.5**			**21**		**21**		**17**		**9**	

Table 6.1.8. 5N Performance Tests Results Summary

	G		ECB	CBC	CFB	OFB	CTR	G
Enc. Time	8	a	0.312	0.380	0.394	0.418	0.384	7
	8	b	68.117	75.668	80.376	78.506	74.925	7
Dec. Time	8	a	0.306	0.394	0.433	0.392	0.421	7
	8	b	68.421	75.672	79.756	76.743	75.955	7
Enc. Throughput	8	a	4090.061	3461.791	2927.564	3058.649	3315.866	6
	8	b	3812.354	3463.220	3315.897	3340.638	3395.339	5
Dec. Throughput	8	a	4082.724	3451.837	2898.837	3278.540	3151.139	6
	8	b	3822.637	3475.678	3264.334	3284.193	3244.035	5
# Of Processes	8	a	17					6
CPU Usage	8	a	16%					5
RAM Usage	8	a	1.427					4
Performance Score	**88**		**65**					

Compared to AES, in the case of studying security, from *Table 6.1.7*, we found that:

- In addition to ECB mode, 5N fails in the CTR mode mapping test, while it passes the mapping test in all other modes.
- 5N passes Histogram test only in CBC and CFB modes.
- Only CBC and CFB pass the correlation test with accepted results.
- All modes except ECB pass the confusion test.
- 5N failed in the diffusion test in all modes.
- Only CBC and CFB modes pass all NIST tests.
- OFB mode has good NIST results.
- 5N maintains the same level of key security compared with AES.

In the case of studying performance, from *Table 6.1.8*, we found that:

- 5N provides better encryption and decryption time with a 68.5% improvement.
- 5N provides better encryption and decryption throughput with a 226% improvement.
- 5N requires a smaller number of processes with a 46.88% improvement.
- 5N requires less CPU and RAM usage with a 3.05% and 2.45% improvement, respectively.

6.1.5 5 ROUNDS AES WITH FULL MIXCOLUMNS

In this section, we implement the AES by using five rounds of Standard AES. *Table 6.1.9* presents the security tests results obtained from this implementation, while *Table 6.1.10* presents the performance tests results.

Table 6.1.9. 5F Security Tests Results Summary

	G		ECB	G	CBC	G	CFB	G	OFB	G	CTR	G
Mapping	1		uniform	1	uniform	1	uniform	1	uniform	1	uniform	1
Histogram	1		7715	0	245.690	1	250.180	1	265.218	1	265.896	1
Correlation	5		0.00087	0	-0.00079	1	-0.00042	1	0.00014	1	0.00032	1
Confusion	1	a	50.010	1	49.980	1	50.039	1	49.962	1	50.020	1
	1	b	50.011	1	49.999	1	49.997	1	49.987	1	50.001	1
Diffusion	1	a	0.00906	0	49.97240	1	50.02466	1	0.00024	0	0.00024	0
	1	b	0.000177	0	49.994340	1	49.986120	1	0.000005	0	0.000005	0
Key Security	5	$2^{128} * x_{5F}$										1
NIST	15	13		15		15		15		15		
Security Score	**23**	**17**		**23**		**23**		**21**		**21**		

Table 6.1.10. 5F Performance Tests Results Summary

	G		ECB	CBC	CFB	OFB	CTR	G
Enc. Time	8	a	0.560	0.521	0.541	0.538	0.553	6
	8	b	113.776	127.844	119.227	121.439	124.436	5
Dec. Time	8	a	0.670	0.631	0.552	0.510	0.547	6
	8	b	140.059	159.454	124.909	120.795	125.360	5
Enc. Throughput	8	a	2151.024	2412.277	2353.803	2374.444	2323.249	3
	8	b	2331.612	2056.271	2108.205	2113.681	2064.834	3
Dec. Throughput	8	a	1793.508	1963.316	2316.115	2412.069	2318.531	3
	8	b	1897.996	1674.323	2049.804	2177.650	2056.723	3
# Of Processes	8	a	20					5
CPU Usage	8	a	17.17%					2
RAM Usage	8	a	1.445					2
Performance Score	**88**	**43**						

Compared to AES, in the case of studying security, from *Table 6.1.9*, we found that:

- 5F maintains same level of security compared with AES practically under specific criteria.

In the case of studying performance, from *Table 6.1.10*, we found that:

- 5F provides better encryption and decryption time with a 50.5% improvement.
- 5F provides better encryption and decryption throughput with a 103.25% improvement.
- 5F requires a smaller number of processes with a 37.5% improvement.

- 5F requires less RAM with a 1.22% improvement but requires more an 4.04% CPU.
- 5F has a 207% improvement in performance compared with AES.

6.1.6 3-ROUNDS AES WITH FULL MIXCOLUMNS

In this section, we implement the AES by using three rounds of Standard AES. T*able 6.1.11* presents the security tests results obtained from this implementation, while *Table 6.1.12* presents the performance tests results.

Table 6.1.11. 3F Security Tests Results Summary

	G		ECB	G	CBC	G	CFB	G	OFB	G	CTR	G
Mapping	1		uniform	1	uniform	1	uniform	1	uniform	1	uniform	1
Histogram	1		63946	0	256.531	1	256.922	1	247.026	1	253.986	1
Correlation	5		-0.00262	0	0.00002	1	-0.00055	1	-0.00075	1	0.00017	1
Confusion	1	a	50.081	1	50.034	1	49.955	1	49.995	1	50.190	1
	1	b	50.044	1	50.000	1	49.991	1	50.003	1	50.168	1
Diffusion	1	a	0.00820	0	50.01866	1	50.00373	1	0.00024	0	0.00024	0
	1	b	0.000160	0	50.018526	1	49.992503	1	0.000005	0	0.000005	0
Key Security	5	$2^{128} * x_{3F}$										1
NIST	15	13		15		15		15		15		
Security Score	**23**	**17**		**23**		**23**		**21**		**21**		

Table 6.1.12. 3F Performance Tests Results Summary

	G		ECB	CBC	CFB	OFB	CTR	G
Enc. Time	8	a	0.312	0.322	0.326	0.381	0.342	7
	8	b	70.500	74.601	75.950	75.309	79.786	7
Dec. Time	8	a	0.364	0.382	0.365	0.417	0.350	7
	8	b	84.030	89.429	75.789	76.506	78.163	7
Enc. Throughput	8	a	4121.117	3812.127	3830.001	3232.544	3603.957	6
	8	b	3713.571	3245.844	3278.260	3418.138	3160.745	5
Dec. Throughput	8	a	3490.198	3246.598	3700.335	3139.058	3495.501	6
	8	b	3098.007	2762.357	3382.161	3421.635	3236.305	6
# Of Processes	8	a	16					7
CPU Usage	8	a	15.81%					6
RAM Usage	8	a	1.426					4
Performance Score	**88**	**68**						

Compared to AES, in the case of studying security, from *Table 6.1.11*, we found that:

- 3F maintains same level of security compared with AES practically under specific criteria.

In the case of studying performance, from *Table 6.1.12*, we found that:

- 3F provides better encryption and decryption time with a 69.25% improvement.
- 3F provides better encryption and decryption throughput with a 224% improvement.
- 3F requires a smaller number of processes with a 50% improvement.
- 3F requires less CPU and RAM usage with a 4.2% and 2.54% improvement, respectively.
- 3F has a 386% improvement in performance compared with AES.

6.1.7 2-ROUNDS AES WITH FULL MIXCOLUMNS

In this section, we implement the AES by using two rounds of Standard AES. *Table 6.1.13* presents the security tests results obtained from this implementation, while *Table 6.1.14* presents the performance tests results.

Table 6.1.13. 2F Security Tests Results Summary

	G		ECB	G	CBC	G	CFB	G	OFB	G	CTR	G
Mapping	1		uniform	1	uniform	1	uniform	1	uniform	1	Not uniform	0
Histogram	1		71760	0	271.097	1	267.193	1	254.307	1	2743916	0
Correlation	5		0.00139	0	0.00014	1	-0.00049	1	0.00029	1	0.01912	0
Confusion	1	a	32.008	0	49.993	1	50.009	1	49.967	1	33.780	0
	1	b	31.992	0	49.997	1	50.007	1	49.995	1	33.158	0
Diffusion	1	a	0.001101	0	24.955730	0	24.979958	0	0.000245	0	0.000245	0
	1	b	0.000022	0	24.999807	0	24.993844	0	0.000005	0	0.000005	0
Key Security	5	$2^{128} + x_{2F}$										1
NIST	15	13		15		15		15		4		
Security Score	**23**	**15**		**21**		**21**		**21**		**5**		

Table 6.1.14. 2F Performance Tests Results Summary

	G		ECB	CBC	CFB	OFB	CTR	G
Enc. Time	8	a	0.221	0.308	0.260	0.240	0.271	8
	8	b	44.149	49.841	50.131	50.969	52.105	8
Dec. Time	8	a	0.221	0.308	0.253	0.272	0.277	8
	8	b	52.827	56.459	49.524	48.500	53.571	8
Enc. Throughput	8	a	5674.642	4170.364	4939.190	5170.860	2323.249	8
	8	b	5848.731	5259.072	5082.957	5113.115	4914.126	8
Dec. Throughput	8	a	4874.683	3553.473	5059.652	5008.225	4313.223	8
	8	b	5070.784	4562.062	5194.253	5117.648	4868.957	8
# Of Processes	8	a	12					8
CPU Usage	8	a	11.35%					8
RAM Usage	8	a	1.414					5
Performance Score	**88**	**85**						

Compared to AES, in the case of studying security, from *Table 6.1.13*, we found that:

- 2F fails in the CTR mode mapping test, while it passes the mapping test in all other modes.
- In addition to ECB mode, 2F fails in the CTR mode histogram test, while it passes the histogram test in all other modes.
- In addition to ECB mode, 2F fails in the CTR mode correlation test, while it passes the correlation test in all other modes.
- ECB and CTR modes failed in the confusion test.
- 2F failed in the diffusion test in all modes.
- CBC, CFB, and OFB modes pass all NIST tests.
- 2F maintains the same level of key security compared with AES.

In the case of studying performance, from *Table 6.1.14*, we found that:

- 2F provides better encryption and decryption time with a 78.5% improvement.
- 2F provides better encryption and decryption throughput with a 357.5% improvement.
- 2F require a smaller number of processes with a 62.5% improvement.
- 2F requires less CPU and RAM usage with a 9% and 3.34% improvement, respectively.
- 2F has a 507% improvement in performance compared with AES.

6.1.8 EXPLORED SCENARIOS SUMMARY

In this section we discussed the results of AES, 10H, 10N, 5H, 5N, 5F, 3F, and 2F scenarios in term of performance and security. While 3F provides best performance scenarios with maintaining the level of security practically, *Table 6.1.15* compares it with AES in term of APC. While *Table 6.1.16* compares it with AES in term of UACI and NPCR tests.

Table 6.1.15. APC

	AES	**3F**	**Gain**
Average Power consumption (Watt)	13.31	12.446	6%

- From *Table 6.1.15*, we found that 3F provides better APC with a 6% improvement.

Table 6.1.16. UACI and NPCR for 512 × 512 Lena image

	AES	3F	Similarity
UACI	33.465	33.477	Almost, a perfect match in the results
NPCR	99.609	99.612	

From *Table 6.1.16*, we found that AES has good UACI and NCPR values, since the changing of 1 pixel has infected the entire image, and hence the encryption algorithm has a good avalanche effect, so the algorithm is resistant against differential attack.

- There is no real change between AES and 3F in UACI and NPCR results. In other words, 3F maintains the same level of UACI and NPCR practically.

6.2 RASPBERRY PI RESULTS

This section compares the result of AES and 3F in term of performance when run them on RP that have been chosen as an IoT model.

6.2.1 RP AES PERFORMANCE RESULT

Table 6.2.1 and *Table 6.2.2* show the results of encryption time and encryption throughput of the AES algorithm with 10-500 KB data files on RP. While *Table 6.2.3* and *Table 6.2.4* show the results of decryption time and decryption throughput of AES algorithm with 10-500 KB data files on RP.

Table 6.2.1. AES Encryption Time for 10 - 500 KB data on RP

	Data	Encryption Time				
	KB	S				
		ECB	CBC	CFB	OFB	CTR
	10	1.149	1.173	1.176	1.175	1.191
	25	2.833	2.920	2.918	2.919	2.947
	50	5.657	5.869	5.829	5.840	5.885
	100	11.294	11.651	11.657	11.664	11.895
	150	17.091	17.510	17.473	17.491	17.653
	250	28.364	29.169	29.138	29.302	29.407
	500	56.654	58.296	58.501	58.756	58.924
Total	**1085**	**123.043**	**126.587**	**126.693**	**127.148**	**127.902**
Average	**155**	**17.578**	**18.084**	**18.099**	**18.164**	**18.272**

Table 6.2.2. AES Encryption Throughput for 10 - 500 KB data on RP

	Data	Encryption Throughput				
	KB	Kb/sec				
		ECB	CBC	CFB	OFB	CTR
	10	69.611	68.222	68.049	68.088	67.172
	25	70.589	68.496	68.535	68.511	67.868
	50	70.709	68.166	68.621	68.499	67.972
	100	70.834	68.663	68.630	68.588	67.264
	150	70.216	68.535	68.678	68.608	67.978
	250	70.513	68.568	68.641	68.257	68.015
	500	70.464	68.480	68.240	67.947	67.749
Total	**1085**	**492.936**	**479.130**	**479.393**	**478.498**	**474.018**
Average	**155**	**70.419**	**68.447**	**68.485**	**68.357**	**67.717**

Table 6.2.3. AES Decryption Time for 10 - 500 KB data on RP

	Data	Decryption Time				
	KB	S				
		ECB	CBC	CFB	OFB	CTR
	10	1.416	1.173	1.172	1.173	1.183
	25	3.561	2.920	2.916	2.920	2.944
	50	7.056	5.869	5.827	5.835	5.931
	100	14.105	11.651	11.662	11.661	11.874
	150	21.535	17.510	17.474	17.493	17.630
	250	35.422	29.169	29.135	29.368	29.561
	500	70.910	58.296	58.511	58.379	59.160
Total	**1085**	**154.005**	**126.587**	**126.697**	**126.830**	**128.283**
Average	**155**	**22.001**	**18.084**	**18.100**	**18.119**	**18.326**

Table 6.2.4. AES Decryption Throughput for 10 - 500 KB data on RP

	Data	Decryption Throughput				
	KB	Kb/sec				
		ECB	CBC	CFB	OFB	CTR
	10	56.512	55.086	68.261	68.203	67.608
	25	56.178	55.067	68.590	68.503	67.944
	50	56.690	54.986	68.644	68.551	67.447
	100	56.716	55.321	68.601	68.606	67.381
	150	55.734	55.295	68.677	68.600	68.068
	250	56.463	55.157	68.648	68.107	67.660
	500	56.298	55.134	68.229	68.382	67.482
Total	**1085**	**394.590**	**386.046**	**479.650**	**478.952**	**473.590**
Average	**155**	**56.370**	**55.149**	**68.521**	**68.422**	**67.656**

6.2.1.1 RP AES PERFORMANCE RESULTS SUMMARY

In this section, we implement the standard AES on RP. *Table 6.2.5* presents the summary of performance results.

Table 6.2.5. AES Performance Results Summary on RP

	G	ECB		CBC	CFB	OFB	CTR	G
Enc. Time	2	a	17.578	18.084	18.099	18.164	18.272	1
Dec. Time	2	a	22.001	18.084	18.100	18.119	18.326	1
Enc. Throughput	2	a	70.419	68.447	68.485	68.357	67.717	1
Dec. Throughput	2	a	56.370	55.149	68.521	68.422	67.656	1
Performance Score	**8**	**4**						

6.2.2 RP 3F PERFORMANCE RESULT

Table 6.2.6 and *Table 6.2.7* show the results of encryption time and encryption throughput of the 3F algorithm with 10-500 KB data files on RP. While *Table 6.2.8* and *Table 6.2.9* show the results of decryption time and decryption throughput of 3F algorithm with 10-500 KB data files on RP.

Table 6.2.6. 3F Encryption Time for 10 - 500 KB data on RP

	Data	**Encryption Time**				
	KB	**S**				
		ECB	**CBC**	**CFB**	**OFB**	**CTR**
	10	0.362	0.386	0.380	0.391	0.394
	25	0.870	0.950	0.944	0.946	0.991
	50	1.739	1.900	1.886	1.892	1.950
	100	3.474	3.774	3.772	3.804	3.889
	150	5.213	5.685	5.660	5.705	5.832
	250	8.687	9.451	9.513	9.429	9.727
	500	17.383	18.978	18.984	18.877	19.464
Total	**1085**	**37.728**	**41.124**	**41.138**	**41.044**	**42.246**
Average	**155**	**5.390**	**5.875**	**5.877**	**5.863**	**6.035**

Table 6.2.7. 3F Encryption Throughput for 10 - 500 KB data on RP

	Data	Encryption Throughput				
	KB	Kb/sec				
		ECB	CBC	CFB	OFB	CTR
	10	221.142	207.481	210.529	204.723	203.219
	25	229.889	210.604	211.940	211.344	201.806
	50	230.017	210.545	212.127	211.383	205.128
	100	230.282	211.977	212.089	210.308	205.726
	150	230.195	211.085	212.027	210.376	205.773
	250	230.230	211.618	210.271	212.112	205.613
	500	229.654	210.350	210.303	211.478	205.100
Total	**1085**	**1601.409**	**1473.661**	**1479.285**	**1471.725**	**1432.365**
Average	**155**	**228.773**	**210.523**	**211.326**	**210.246**	**204.624**

Table 6.2.8. 3F Decryption Time for 10 - 500 KB data on RP

	Data	Decryption Time				
	KB	S				
		ECB	CBC	CFB	OFB	CTR
	10	0.416	0.448	0.378	0.379	0.391
	25	1.023	1.105	0.945	0.950	0.978
	50	2.044	2.201	1.887	1.918	1.945
	100	4.089	4.390	3.772	3.811	3.892
	150	6.131	6.608	5.704	5.692	5.843
	250	10.230	10.974	9.529	9.427	9.723
	500	20.430	21.957	18.987	18.873	19.458
Total	**1085**	**44.364**	**47.683**	**41.203**	**41.049**	**42.231**
Average	**155**	**6.338**	**6.812**	**5.886**	**5.864**	**6.033**

Table 6.2.9. 3F Decryption Throughput for 10 - 500 KB data on RP

	Data	Decryption Throughput				
	KB	Kb/sec				
		ECB	CBC	CFB	OFB	CTR
	10	192.484	178.546	211.454	210.897	204.604
	25	195.440	180.957	211.641	210.546	204.437
	50	195.695	181.766	211.977	208.689	205.621
	100	195.631	182.219	212.089	209.966	205.568
	150	195.716	181.610	210.395	210.836	205.363
	250	195.498	182.255	209.927	212.164	205.705
	500	195.399	181.807	210.269	211.523	205.156
Total	**1085**	**1365.864**	**1269.160**	**1477.752**	**1474.620**	**1436.454**
Average	**155**	**195.123**	**181.309**	**211.107**	**210.660**	**205.208**

6.2.2.1 RP 3F PERFORMANCE RESULTS SUMMARY

In this section, we implement the 3F on RP. *Table 6.2.10* presents the summary of performance results.

Table 6.2.10. 3F Performance Results Summary on RP

	G		ECB	CBC	CFB	OFB	CTR	G
Enc. Time	2	a	5.390	5.875	5.877	5.863	6.035	2
Dec. Time	2	a	6.338	6.812	5.886	5.864	6.033	2
Enc. Throughput	2	a	228.773	210.523	211.326	210.246	204.624	2
Dec. Throughput	2	a	195.123	181.309	211.107	210.660	205.208	2
Performance Score	**8**	**8**						

6.2.3 RASPBERRY PI RESULTS SUMMARY

Compared to AES on RP, in the case of studying performance, from *Table 6.2.10*, we found that:

- 3F provides better encryption and decryption time with a 67.5% improvement.
- 3F provides better encryption and decryption throughput with a 213.5% improvement.
    - **These two results are relatively consistent with the results that have been shown in section 6.1.8.**
- 3F has a 100% improvement in performance with AES. **The change in total improvement occurred because not all factors were taken into account in the calculation.**
- *Table 6.2.11*, compare between AES APC and 3F APC. The results show that 3F has a better APC with a 7.38% improvement.

Table 6.2.11. RP APC

	AES	**3F**	**Gain**
Average Power consumption (Watt)	1.49	1.38	7.38%

6.3 AES EXPLORING RESULTS SUMMARY

In this section, we present a brief summary of the results of the explored scenarios and discuss these results in a scientific manner. *Table 6.3.1* summarizes the results in a scientific and mathematical way to facilitate the process of extrapolating recommendations.

Table 6.3.1. AES Exploring Results Summary

	AES	10H	10N	5H	5N	5F	3F	2F
Key Security	**Pass**	**No changes**						
NIST	**Pass**	**Moderate**	**Fail**	**Moderate**	**Fail**	**No Changes**		
Diffusion	**Pass**	**Fail**				**No Changes**	**Acceptable**	
Confusion	**Pass**	**Acceptable**				**No Changes**		
Correlation	**Pass**	**Moderate**	**Fail**	**Moderate**	**Fail**	**No Changes**		
Histogram	**Pass**	**Moderate**	**Fail**	**Moderate**	**Fail**	**No Changes**		
Mapping	**Pass**	**Moderate**	**Fail**	**Moderate**	**Fail**	**No Changes**		
Dec. Throughput	**Good**	**+89 %**	**31 %**	**+158.5 %**	**+242.5 %**	**+105.5 %**	**+228 %**	**+374.5 %**
Enc. Throughput	**Good**	**+70.5%**	**25%**	**+151.5 %**	**+209.5 %**	**+101%**	**+220 %**	**+340.5%**
Dec. Time	**Good**	**+47.5 %**	**24.5%**	**+61 %**	**+70 %**	**+51%**	**+69.5 %**	**+79.5 %**
Enc. Time	**Good**	**+41 %**	**+19.5%**	**+59.5 %**	**+67%**	**+50%**	**+69%**	**+77.5%**
Security Score	**20.5**	**17.25**	**15.5**	**12.375**	**13.875**	**20.5**	**20.5**	**15.5**
Performance Score	**14**	**29**	**47**	**55**	**65**	**37**	**68**	**85**

From *Table 6.3.1*, the results of explored scenarios can be summarized as follow:

- The ECB mode fails in most security test due to the way of data handling in encryption process. Since each block in plaintext is isolated from other block, that mean each block in plaintext has an identical block in ciphertext.

- The CBC and CFB modes are the best modes to be used during their results in security tests.

- MixColumn operation improve the security of algorithm, especially in Diffusion. Since it provides dynamic changes to the results.

- 10H has a 24% loss in security compared with AES. While it provides a 93% improvement in performance.

- 10N has a 40% loss in security compared with AES. While it has a 220% improvement in performance.

- 5H has a 38% loss in security compared with AES. While it provides a 28% improvement in performance with AES.

- 5N has a 32% loss in security compared with AES. While it has a 347% improvement in performance.

- 5F maintains same level of security practically under specific criteria compared with AES and provides a 147% improvement in performance with AES.

- 3F maintains the same level of security compared with AES practically under specific criteria and provides a 386% improvement in performance compared with AES.

- 2F has a 24% loss in security compared with AES. While it has a 467% improvement in performance compared with AES.

CHAPTER 7

LIGHTWEIGHT AES ALGORITHM

7 LIGHTWEIGHT AES ALGORITHM

In this chapter, we highlight the most important points from previous the chapter, and by reviewing all the results of the best scenario among the previously mentioned scenarios to prove the superiority of this scenario to come up with our recommendations related to this study.

Based on the result and comparison that have been done in chapter 6, we found:

- The results indicate that running three rounds of AES at least is sufficient to maintain the level of security, and that this algorithm optimization achieves a significant performance advantage by 386%. Section 7.1 discusses the detailed result of 3F.

- The standard AES algorithm loses some security level when run at one or two rounds due to the lack of changes to the original text during encryption. Thus, it is important to devise a mechanism to complicate and increase this effect, provided it does not require a lot of time.

- AES loses more security when reducing the MixColumns operation.

- AES loses much more security when ignoring the MixColumns operation.

- Based on the results, we choose 3F as the New Lightweight AES Algorithm (NLW-AES).

7.1 NEW LIGHTWEIGHT AES EVALUATION

In this section, we discuss the results of the New Lightweight AES (NLW-AES) in term of performance and security for all modes of operation. These results enable us to specify which suitable mode can satisfy acceptable performance and security levels. *Figure 7.1* present the time analysis of NLW-AES algorithm.

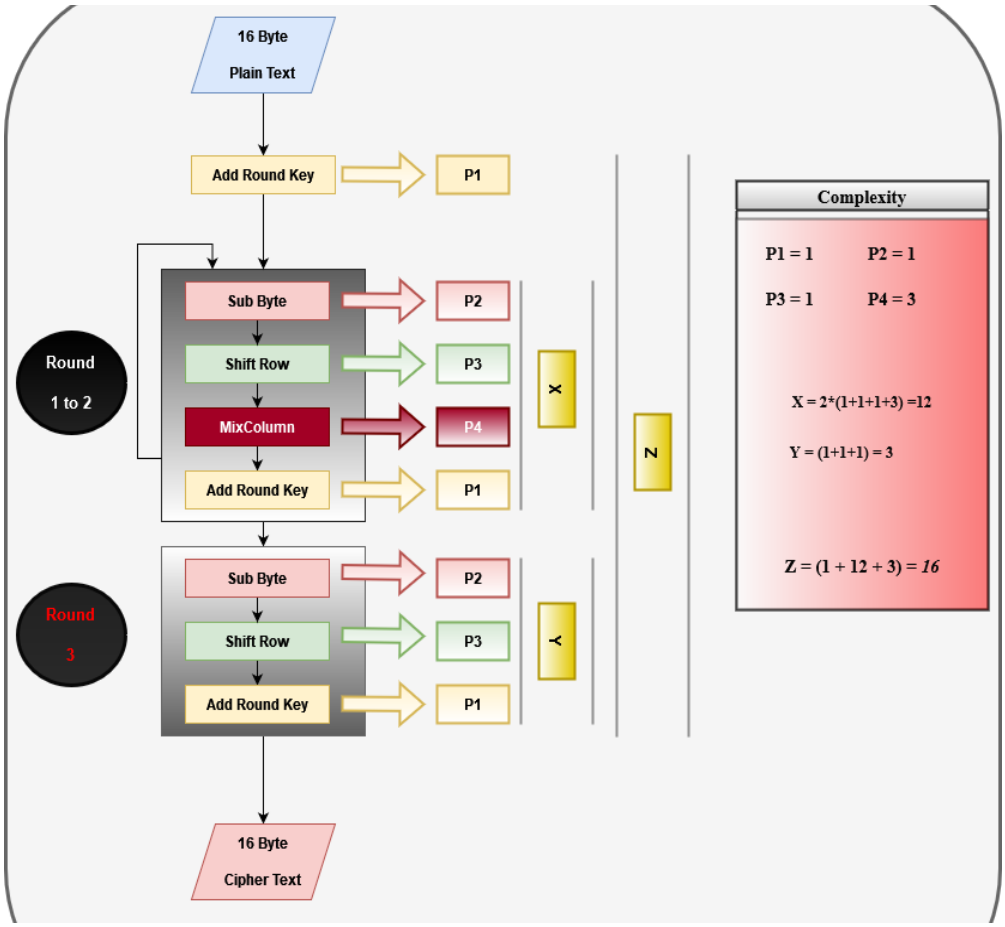


Figure 7.1. NLW-AES Time Analysis

From Equation 11, we can find that:

$$improvment = \frac{16 - 58}{58} * 100\% \quad = \mathbf{72.4\%}$$

which is close to the encryption time improvment that calculated practically

7.1.1 PERFORMANCE RESULTS

This section presents the results of encryption/decryption time, and encryption/decryption throughput founds according to testing the NLW-AES on different data sets.

7.1.1.1 ENCRYPTION

Table 7.1.1 and *Table 7.1.2* show the results of encryption time and encryption throughput of the NLW-AES algorithm on 10-500 KB data files.

Table 7.1.1. NLW-AES Encryption Time for 10 - 500 KB data

	Data	Encryption Time				
	KB	S				
		ECB	CBC	CFB	OFB	CTR
	10	0.020	0.024	0.021	0.025	0.022
	25	0.046	0.050	0.052	0.066	0.055
	50	0.090	0.106	0.102	0.152	0.110
	100	0.183	0.201	0.229	0.303	0.238
	150	0.323	0.307	0.312	0.329	0.354
	250	0.517	0.544	0.502	0.630	0.534
	500	1.008	1.020	1.068	1.161	1.078
Total	**1085**	**2.186**	**2.251**	**2.285**	**2.666**	**2.391**
Average	**155**	**0.312**	**0.322**	**0.326**	**0.381**	**0.342**

Table 7.1.2. NLW-AES Encryption Throughput for 10 - 500 KB data

	Data	Encryption Throughput				
	KB	Kb/sec				
		ECB	CBC	CFB	OFB	CTR
	10	4000.000	3385.507	3896.262	3365.079	3641.383
	25	4380.032	4000.000	3887.576	3205.128	3641.855
	50	4429.946	3802.891	3934.343	2842.152	3655.324
	100	4389.275	3973.662	3518.655	2777.498	3398.578
	150	3808.580	3910.805	3850.968	3681.493	3436.606
	250	3871.987	3695.208	3981.487	3271.399	3748.280
	500	3968.000	3916.815	3740.716	3485.060	3705.674
Total	**1085**	**28847.821**	**26684.888**	**26810.007**	**22627.810**	**25227.701**
Average	**155**	**4121.117**	**3812.127**	**3830.001**	**3232.544**	**3603.957**

In *Table 7.1.3* and *Table 7.1.4*, we present the results of encryption time and encryption throughput of NLW-AES algorithm on 1-100 MB data files.

Table 7.1.3. NLW-AES Encryption Time for 1 - 100 MB data

	Data	**Encryption Time**				
	MB	**S**				
		ECB	**CBC**	**CFB**	**OFB**	**CTR**
	1	2.122	2.741	2.946	2.342	2.554
	5	10.935	12.610	12.198	12.064	12.308
	10	21.990	25.499	23.816	23.179	29.136
	20	44.365	50.563	52.538	49.865	51.498
	50	110.541	126.356	120.793	120.918	128.033
	100	233.050	229.836	243.408	243.483	255.187
Total	**186**	**423.003**	**447.604**	**455.699**	**451.852**	**478.715**
Average	**31**	**70.500**	**74.601**	**75.950**	**75.309**	**79.786**

Table 7.1.4. NLW-AES Encryption Throughput for 1 - 100 MB data

	Data	**Throughput**				
	MB	**Kb/sec**				
		ECB	**CBC**	**CFB**	**OFB**	**CTR**
	1	3855.500	2983.899	2877.091	3499.226	3213.453
	5	3767.128	3240.006	3389.081	3405.807	3324.376
	10	3744.167	3205.472	3509.703	3561.017	2827.118
	20	3705.826	3232.553	3139.565	3285.008	3198.274
	50	3701.114	3237.022	3395.541	3402.053	3198.620
	100	3507.689	3576.111	3358.579	3355.717	3202.629
Total	**186**	**22281.424**	**19475.063**	**19669.560**	**20508.827**	**18964.471**
Average	**31**	**3713.571**	**3245.844**	**3278.260**	**3418.138**	**3160.745**

7.1.1.2 DECRYPTION

Table 7.1.5 and *Table 7.1.6* shows the results of decryption time and decryption throughput of NLW-AES algorithm on 10-500 KB data files.

Table 7.1.5. NLW-AES Decryption Time for 10 - 500 KB data

	Data	Decryption Time				
	KB	S				
		ECB	CBC	CFB	OFB	CTR
	10	0.023	0.024	0.021	0.026	0.021
	25	0.055	0.062	0.051	0.067	0.055
	50	0.106	0.122	0.128	0.151	0.129
	100	0.230	0.248	0.200	0.294	0.300
	150	0.380	0.366	0.313	0.399	0.344
	250	0.599	0.648	0.502	0.533	0.522
	500	1.152	1.204	1.343	1.447	1.081
Total	**1085**	**2.546**	**2.674**	**2.558**	**2.916**	**2.453**
Average	**155**	**0.364**	**0.382**	**0.365**	**0.417**	**0.350**

Table 7.1.6. NLW-AES Decryption Throughput for 10 - 500 KB data

	Data	Decryption Throughput				
	KB	Kb/sec				
		ECB	CBC	CFB	OFB	CTR
	10	3530.962	3288.889	3826.087	3387.387	3751.804
	25	3637.165	3226.071	3934.343	3150.194	3624.116
	50	3763.846	3275.121	3215.811	2853.393	3226.757
	100	3474.169	3231.034	3997.006	2789.984	2797.518
	150	3195.654	3285.300	3834.518	3166.177	3532.306
	250	3352.800	3094.622	3985.452	3763.184	3834.373
	500	3476.792	3325.152	3109.132	2863.084	3701.632
Total	**1085**	**24431.389**	**22726.189**	**25902.348**	**21973.403**	**24468.506**
Average	**155**	**3490.198**	**3246.598**	**3700.335**	**3139.058**	**3495.501**

In *Table 7.1.7* and *Table 7.1.8*, we present the results of decryption time and decryption throughput of NLW-AES algorithm on 1-100 MB data files.

Table 7.1.7. NLW-AES Decryption Time for 1 - 100 MB data

	Data	**Decryption Time**				
	MB	**S**				
		ECB	**CBC**	**CFB**	**OFB**	**CTR**
	1	2.610	3.253	2.829	2.444	2.534
	5	12.780	14.992	11.050	11.203	11.990
	10	26.691	30.216	22.288	23.776	27.478
	20	52.888	58.101	52.819	48.714	51.227
	50	134.063	139.363	118.406	125.108	128.435
	100	275.148	290.646	247.339	247.792	247.312
Total	**186**	**504.181**	**536.572**	**454.731**	**459.036**	**468.975**
Average	**31**	**84.030**	**89.429**	**75.789**	**76.506**	**78.163**

Table 7.1.8. NLW-AES Decryption Throughput for 1 - 100 MB data

	Data	**Decryption Throughput**				
	MB	**Kb/sec**				
		ECB	**CBC**	**CFB**	**OFB**	**CTR**
	1	3148.371	2519.066	2977.317	3386.263	3243.810
	5	3214.076	2735.974	3715.431	3695.364	3443.540
	10	3074.177	2714.724	3681.102	3470.730	2981.607
	20	3111.710	2823.087	3127.476	3384.646	3220.841
	50	3060.719	2941.712	3477.975	3285.386	3201.649
	100	2978.988	2839.576	3313.662	3307.421	3326.385
Total	**186**	**18588.040**	**16574.139**	**20292.963**	**20529.810**	**19417.833**
Average	**31**	**3098.007**	**2762.357**	**3382.161**	**3421.635**	**3236.305**

7.1.1.3 HARDWARE USAGE

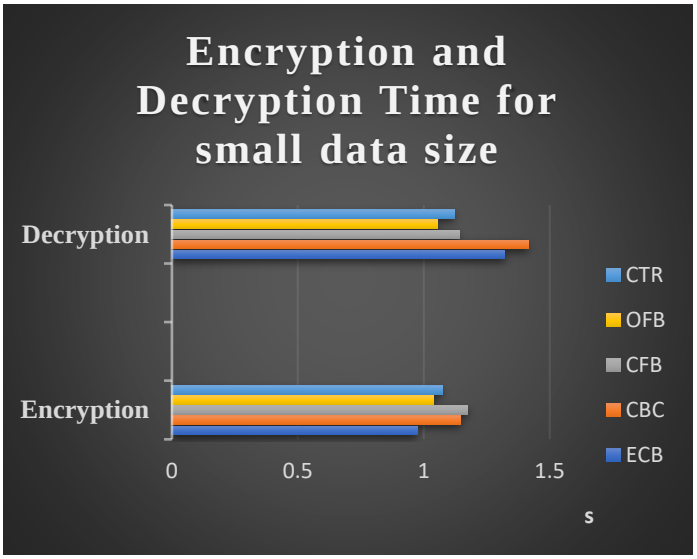
Table 7.1.9 shows the processes of algorithm and calculate the CPU and RAM that needed to encrypt 100KB data with LW-AES.

Table 7.1.9. NLW-AES Hardware Usage

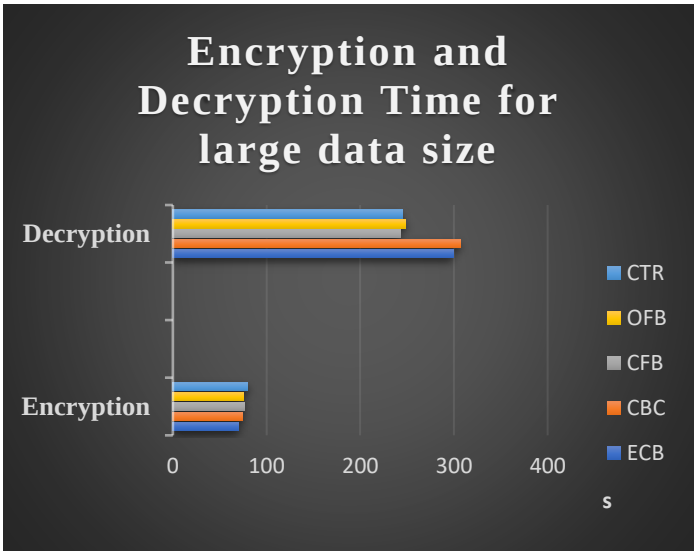
	Processes	CPU	MEMORY
AES	32	16.50%	1.4629

7.1.1.4 PERFORMANCE RESULTS SUMMARY

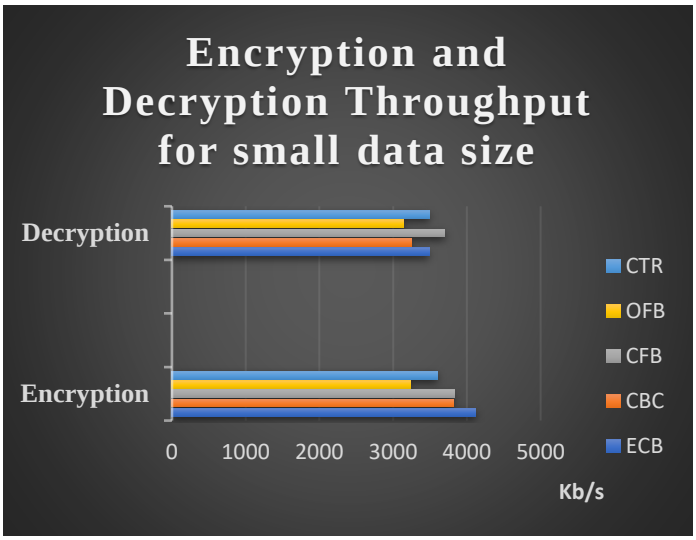
After presenting the encryption time, encryption throughput, decryption time, and decryption throughput. *Figure 7.1* summarize these results.



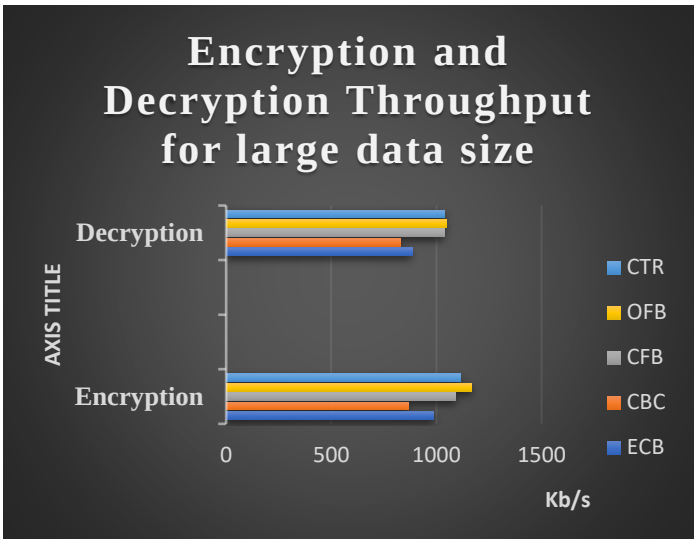
(a) Small Encryption and Decryption Time Summary



(b) Large Encryption and Decryption Time Summary



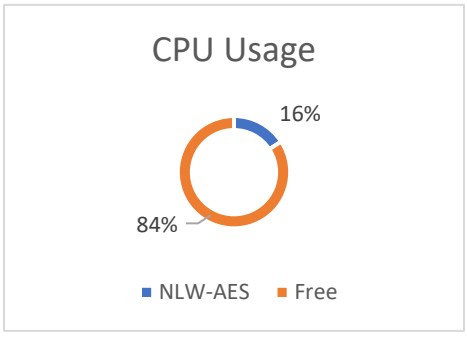
(c) Small Encryption and Decryption Throughput Summary



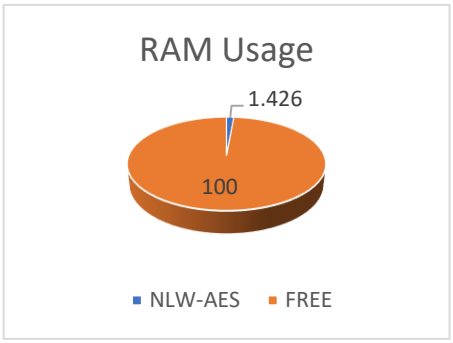
(d) Large Encryption and Decryption Throughput Summary

Figure 7.2. NLW-AES Performance Results Summary

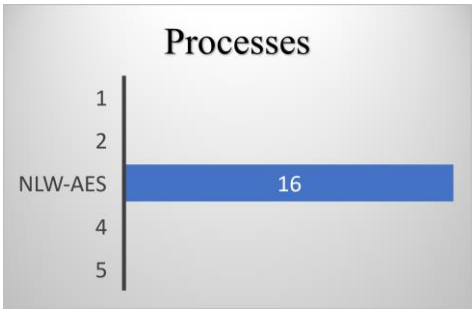
Figure 7.2 reflects the hardware used by NLW-AES to encrypt 100KB data.



(a) CPU Usage



(b) RAM Usage



(c) # of Processes

Figure 7.3. NLW-AES Hardware Usage

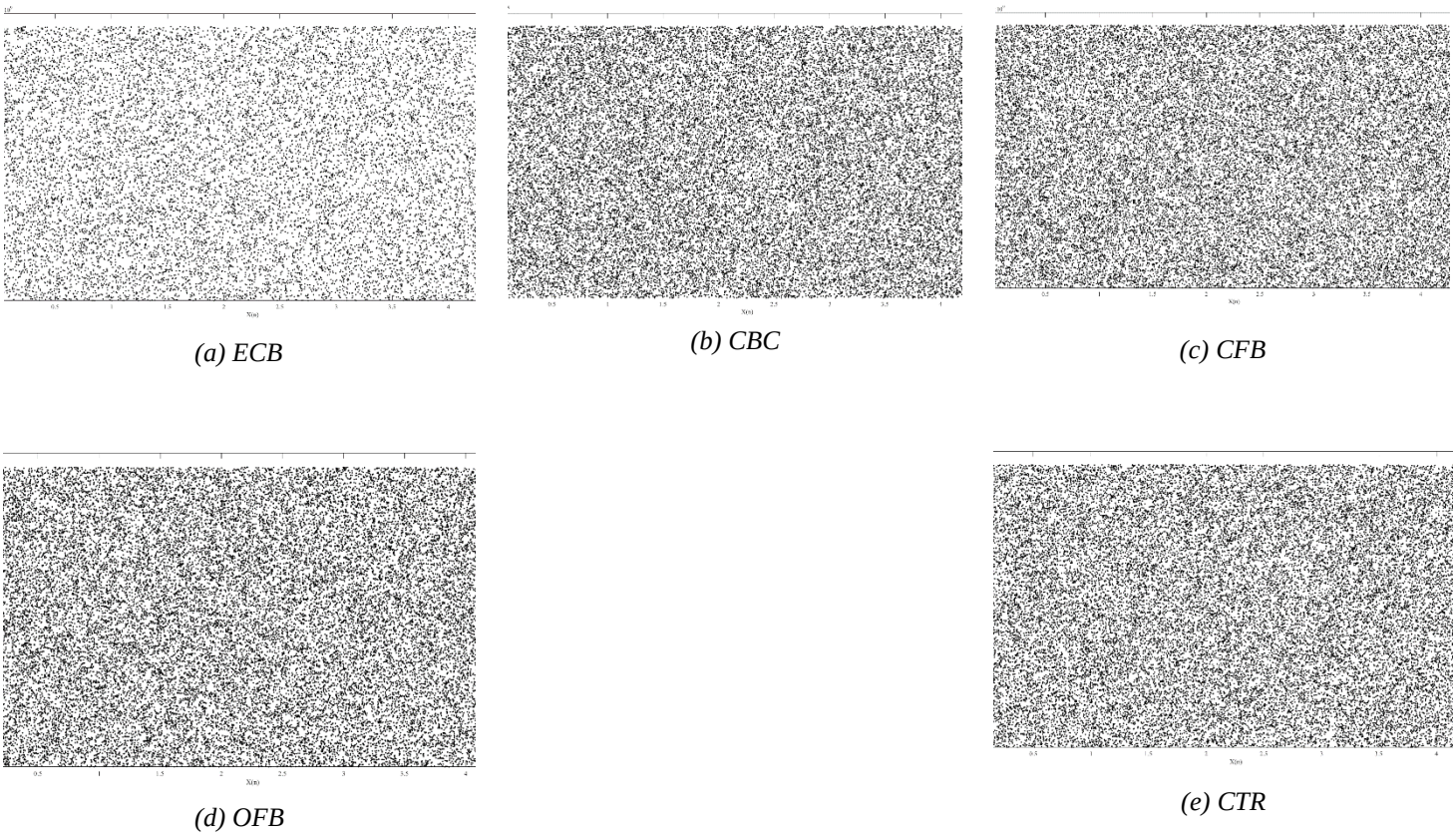
After presenting the results of Encryption and Decryption time, Encryption and Decryption Throughput, and Hardware Usage. These results can be summarized as follow:

- NLW-AES provides better encryption and decryption time with a 69.25% improvement.
- NLW-AES provides better encryption and decryption throughput with a 224% improvement.
- NLW-AES requires a smaller number of processes with a 50% improvement.
- NLW-AES requires less CPU and RAM usage with a 4.2% and 2.54% improvement, respectively.
- NLW-AES has a 386% improvement in performance compared with AES.

7.1.2 SECURITY RESULTS

This section presents the results of security tests that tested the NLW-AES algorithm on different data sets.

7.1.2.1 MAPPING



(a) ECB *(b) CBC* *(c) CFB*

(d) OFB *(e) CTR*

Figure 7.4. NLW-AES Mapping Results

Figure 7.3 shows that all modes achieved good mapping results.

7.1.2.2 HISTOGRAM AND CHI-SQUARE

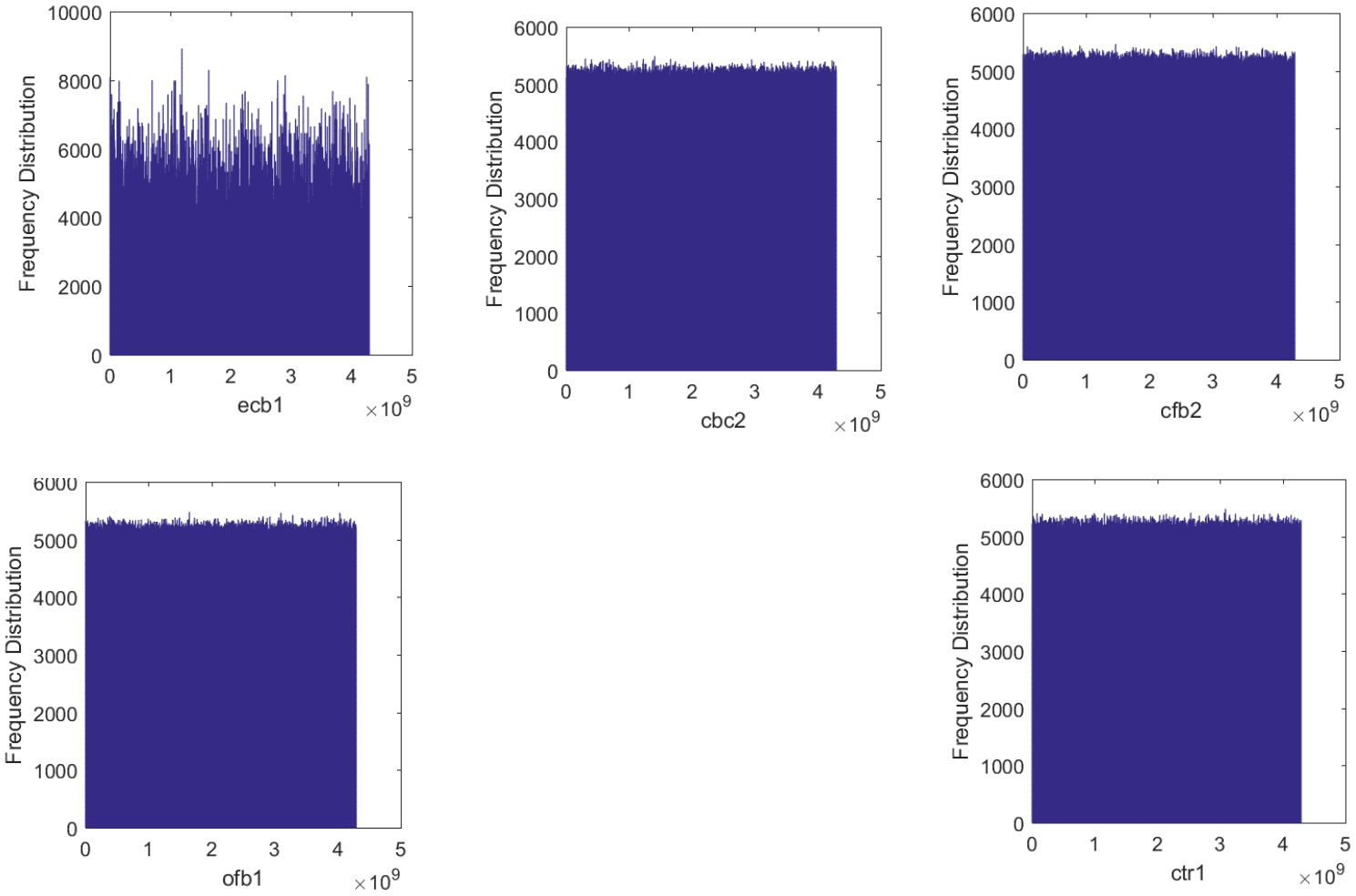


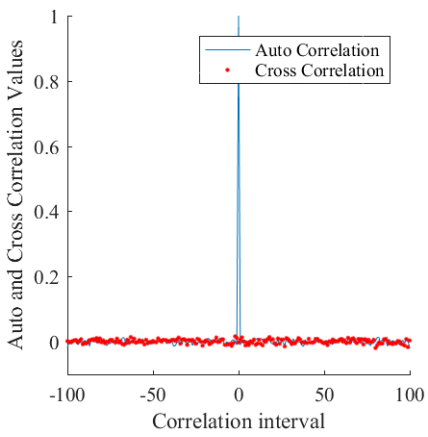
Figure 7.5. NLW-AES Histogram Results

Table 7.1.10. NLW-AES Chi-Square Test Results

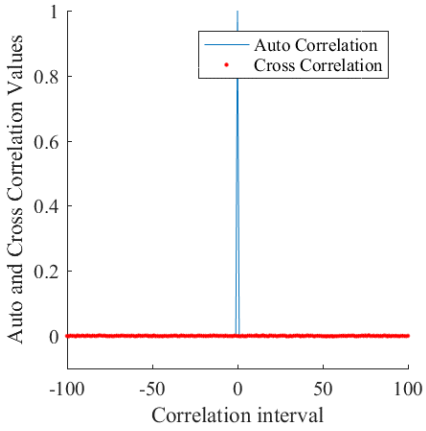
No of Classes = 256			**Alpha = 0.05**		
	ECB	**CBC**	**CFB**	**OFB**	**CTR**
Experimental Value	63946.770	256.531	256.922	247.026	253.986
Critical Value	293.248				
Chi-Square	Not Uniform	Uniform	Uniform	Uniform	Uniform

Figure 7.4 show the results of histogram of all modes, *Table 7.1.10* presents chi-square results, which reflect the distribution of data. From results we note that all modes have a uniform data distribution except ECB mode.

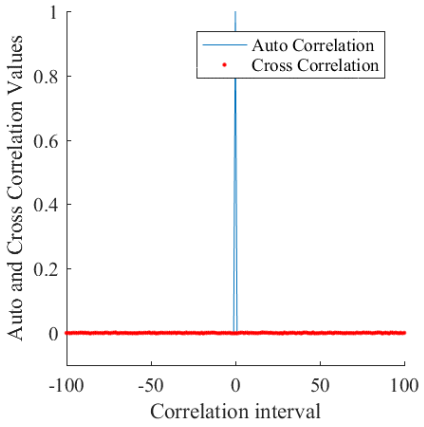
7.1.2.3 CORRELATION



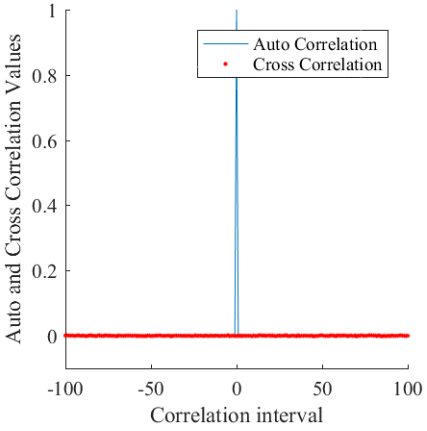
(a) ECB



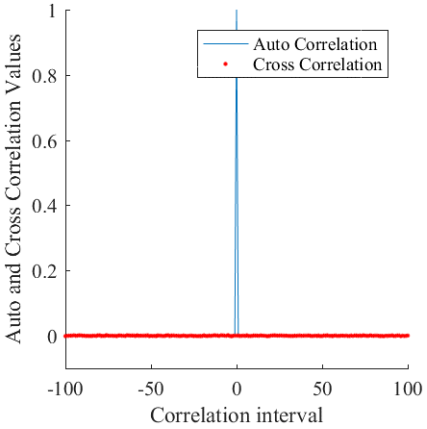
(b) CBC



(c) CFB



(d) OFB



(e) CTR

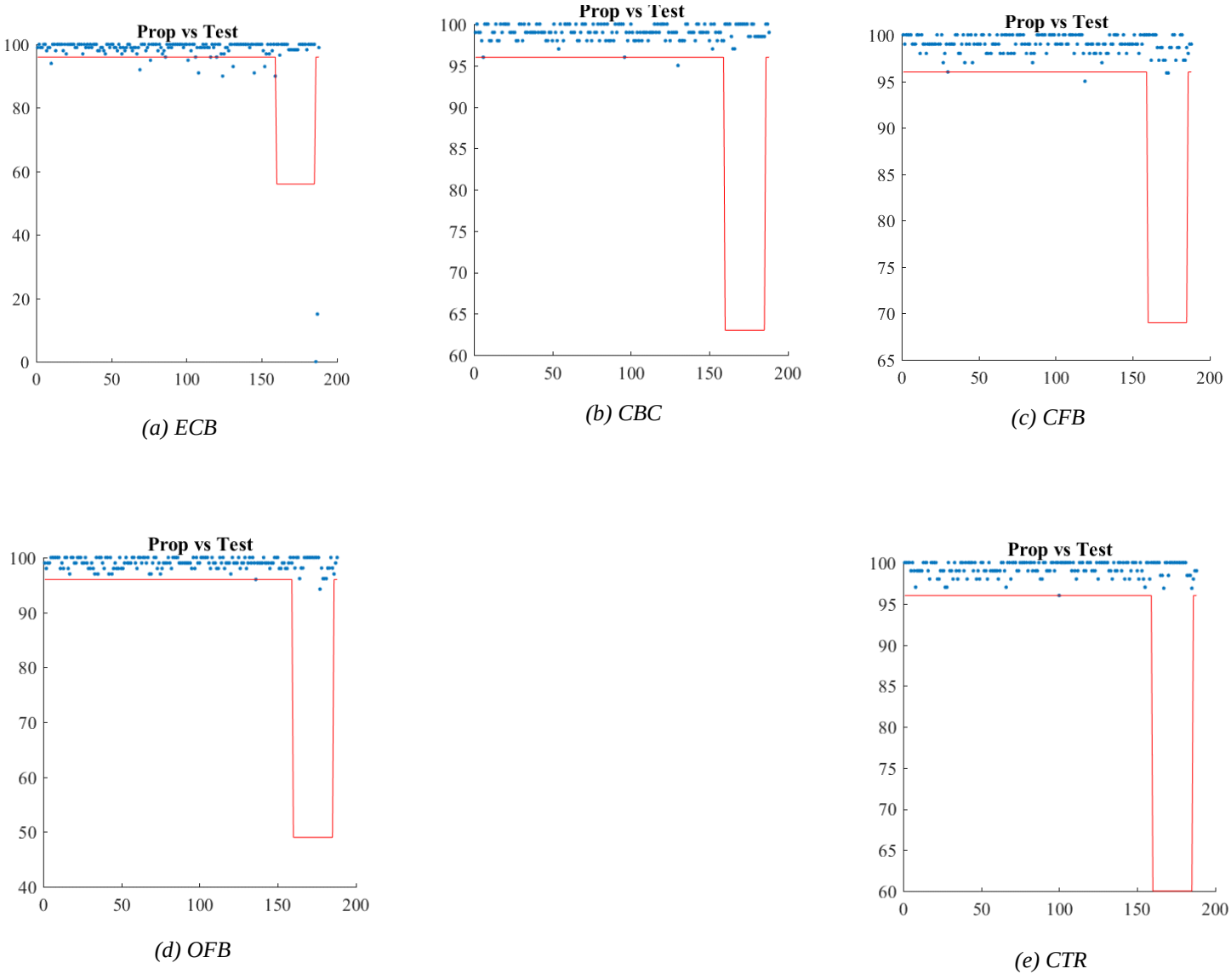
Figure 7.6. NLW-AES Auto and Cross Correlation Results

Table 7.1.11. NLW-AES Correlation Test Result

	ECB	CBC	CFB	OFB	CTR
Correlation Coefficient	-0.00262	0.00002	-0.00055	-0.00075	0.00017

Figure 7.5 shows the auto and cross correlation for NLWAES, based on results in *Table 7.1.11* we found that CBC provide best correlation result followed by CTR, CFB, and OFB mode respectively. ECB mode achieve result that is relatively far from other modes.

7.1.2.4 NIST



(a) ECB

(b) CBC

(c) CFB

(d) OFB

(e) CTR

Figure 7.7. NLW-AES NIST Results

Table 7.1.12. NLW-AES P-Value for NIST Tests

Test		P - Value			
	ECB	**CBC**	**CFB**	**OFB**	**CTR**
Frequency	0.169	0.352	0.501	0.235	0.453
Block-Frequency	0.036	0.182	0.550	0.825	0.220
Cumulative-Sums	0.132	0.677	0.380	0.321	0.072
Runs	0.095	0.475	0.556	0.359	0.398
Longest-Run	0.121	0.519	0.794	0.658	0.483
Rank	0.894	0.586	0.305	0.572	0.447
FFT	0.610	0.583	0.495	0.632	0.595
Nonperiodic-Templates	0.147	0.518	0.499	0.520	0.509
Overlapping-Templates	0.439	0.650	0.323	0.651	0.438
Universal	0.729	0.642	0.265	0.197	0.446
Approximate Entropy	0.000	0.687	0.390	0.443	0.395
Random-Excursions	0.302	0.442	0.440	0.431	0.478
Random-Excursions-Variant	0.396	0.416	0.427	0.407	0.490
Serial	0.000	0.333	0.489	0.646	0.497
Linear-Complexity	0.454	0.507	0.699	0.392	0.968

Table 7.1.13. NLW-AES NIST Tests Results

Test		Probability			
	ECB	**CBC**	**CFB**	**OFB**	**CTR**
Frequency	98.000	99.000	99.667	99.333	99.667
Block-Frequency	98.667	99.333	99.333	98.667	99.333
Cumulative-Sums	97.000	98.667	100.000	99.167	99.667
Runs	99.000	97.667	98.667	100.000	99.667
Longest-Run	99.333	97.000	99.333	99.667	99.333
Rank	99.333	99.667	99.667	99.333	99.667
FFT	99.667	99.000	98.667	99.333	98.333
Nonperiodic-Templates	98.953	99.002	98.998	99.050	99.115
Overlapping-Templates	97.667	98.333	100.000	99.667	99.667
Universal	99.667	98.667	99.000	99.667	98.333
Approximate Entropy	86.667	98.667	99.333	99.333	99.000
Random-Excursions	99.106	98.962	98.858	99.182	99.147
Random-Excursions-Variant	99.252	99.159	99.134	98.903	99.387
Serial	3.000	99.667	98.500	99.000	98.333
Linear-Complexity	99.000	98.667	99.333	99.667	99.000
Pass Rate except RE-Variant	**96**				
Pass Rate of RE-Variant	**60.333**	**57.333**	**61.667**	**53.333**	**59.333**

Figure 7.6 shows the visual results of all 188 NIST tests and subtests. *Table 7.1.12* presents the P-Value of main 15 Test and *Table 7.1.13* presents the Results of main 15 NIST tests.

- From these results, we conclude that All modes have pass all NIST tests except ECB mode, which failed in Entropy and Serial tests.

7.1.2.5 CONFUSION

In *Table 7.1.14*, we present the results of hamming distance according to change one bit in the Key for small data size, while *Table 7.1.15* represents it for large data sizes.

Table 7.1.14. NLW-AES Confusion Results using 1 Bit Change for 10-500 KB Data

Data Size KB		ECB (%)	CBC (%)	CFB (%)	OFB (%)	CTR (%)
10	First	49.982	50.167	49.878	50.333	50.060
	Middle	50.192	50.070	49.940	49.775	50.531
	Last	50.103	50.291	50.113	49.850	50.025
Average		50.093	50.176	49.977	49.986	50.205
100	First	49.984	50.078	50.051	50.049	50.082
	Middle	50.158	49.942	49.920	49.966	50.484
	Last	50.101	50.082	49.894	49.970	50.003
Average		50.081	50.034	49.955	49.995	50.190
500	First	50.008	49.995	50.027	49.991	50.029
	Middle	50.104	49.987	49.980	49.997	50.407
	Last	50.143	49.983	49.979	49.951	50.023
Average		50.085	49.988	49.995	49.980	50.153

Table 7.1.15. NLW-AES Confusion Results using 1 Bit Change for 1-20 MB Data

Data Size MB		ECB (%)	CBC (%)	CFB (%)	OFB (%)	CTR (%)
1	First	50.040	50.000	50.008	49.985	50.030
	Middle	50.064	49.979	50.002	50.003	50.499
	Last	50.120	50.014	49.971	49.964	50.006
Average		50.075	49.997	49.993	49.984	50.178
5	First	50.049	50.008	49.988	49.999	50.011
	Middle	50.045	49.982	50.000	50.021	50.498
	Last	50.039	50.011	49.986	49.988	49.996
Average		50.044	50.000	49.991	50.003	50.168
20	First	50.052	50.007	49.998	50.001	50.035
	Middle	50.041	49.994	50.000	50.008	50.481
	Last	50.030	50.012	49.999	49.999	49.988
Average		50.041	50.004	49.999	50.003	50.168

7.1.2.6 DIFFUSION

In *Table 7.1.16*, we present the results of hamming distance according to change one bit in the plaintext for small data size, while *Table 7.1.17* represents it for large data sizes.

Table 7.1.16. NLW-AES Diffusion Results using 1 Bit Change for 10-500 KB Data

Data Size KB		ECB (%)	CBC (%)	CFB (%)	OFB (%)	CTR (%)
10	First	0.06836	49.90295	49.86867	0.00367	0.00367
	Middle	0.08545	32.49185	32.75793	0.00611	0.00611
	Last	0.07935	0.07080	0.00489	0.00489	0.00489
Average		0.07772	27.48854	27.54383	0.00489	0.00489
100	First	.00820	50.01866	50.00373	0.00024	0.00024
	Middle	0.00685	26.04268	26.03511	0.00049	0.00049
	Last	0.00795	0.00746	0.00049	0.00049	0.00049
Average		0.00767	25.35627	25.34644	0.00041	0.00041
500	First	0.00164	50.01380	49.99252	0.00005	0.00005
	Middle	0.00147	25.48508	25.50221	0.00010	0.00010
	Last	0.00164	0.00291	0.00020	0.00007	0.00007
Average		0.00158	25.16726	25.16497	0.00007	0.00007

Table 7.1.17. NLW-AES Diffusion Results using 1 Bit Change for 1-20 MB Data

Data Size MB		ECB (%)	CBC (%)	CFB (%)	OFB (%)	CTR (%)
1	First	0.000801	50.001680	49.966785	0.000024	0.000024
	Middle	0.000741	27.664559	27.647534	0.000048	0.000048
	Last	0.000705	0.000909	0.000036	0.000036	0.000036
Average		0.000749	25.889049	25.871452	0.000036	0.000036
5	First	0.000160	50.018526	49.992503	0.000005	0.000005
	Middle	0.000151	26.265391	26.249602	0.000010	0.000010
	Last	0.000158	0.000136	0.000007	0.000007	0.000007
Average		0.000156	25.428018	25.414037	0.000007	0.000007
20	First	0.000040	50.001929	49.999128	0.000001	0.000001
	Middle	0.000038	24.081796	24.080209	0.000002	0.000002
	Last	0.000035	0.000033	0.000003	0.000003	0.000003
Average		0.000038	24.694586	24.693113	0.000002	0.000002

7.1.2.7 *KEY SECURITY*

- NLW-AES does not affect key security significantly and it is still secure against the brute force attack.

7.1.2.8 *SECURITY RESULTS SUMMARY*

After presenting the results of Mapping, Histogram and Chi-Square, Correlation, NIST, Confusion and Diffusion, and Key security test. These results can be summarized as follow:

- NLW-AES passes the mapping test in all modes.
- NLW-AES passes the histogram test in all modes except ECB.
- NLW-AES passes the correlation test in all modes except ECB.
- NLW-AES passes the NIST tests in all modes except ECB.
- NLW-AES pass the confusion test in all modes.
- NLW-AES passes the diffusion in CBC and CFB modes.
- NLW-AES is secure against the brute force attack with complexity 2^{128}.

7.2 SUMMARY

In light of the previously discussed results, in terms of improving performance by 386%, which is a significant improvement while maintaining the same level of security practically under specific criteria, we found that NLW-AES provided a qualitative development in the field of IoT security.

CHAPTER 8

CONCLUSION AND FUTURE WORK

8 CONCLUSION AND FUTURE WORK

In this chapter, we conclude the thesis, and present the next step for future work.

8.1 CONCLUSION

In this thesis, a detailed study of computer security has been conducted. After clarifying different kinds of cryptography, LWC has been addressed considering its basics and requirements. Some of the presented algorithms highlight the essential needs for LWC algorithms and the importance of making them compatible with the resources of IoT devices. This study also discussed the latest studies related to each LWC, Lightweight AES-based algorithms, and the most prominent evaluation criteria used to judge the suitability of an algorithm.

Furthermore, this study presented the results of testing the AES algorithm according to the specified criteria. Based on these promising results, we have undertaken an extensive exploration of a set of AES scenarios to study their results. This scenario is generated by changing some AES core such as the change in the number of rounds and in the MixColumns operation.

The results of the explored scenarios testing process varied in terms of performance enhancement along with decreasing the level of security, or improving performance while maintaining the same level of security. We found that the change in the MixColumns operation led to a significant decline in the level of security, while the change in the number of rounds preserved the core function of AES maintaining the same level of security considering three to ten rounds practically. Upon testing two rounds and one round of AES, the level of security has been declined significantly.

Based on the need of maintaining the level of security, the explored scenarios have been limited to 5F and 3F. These two scenarios maintain the same level of security as AES practically under specific criteria with different performance levels. Finally, through a detailed comparison process based on the level of performance, we found that the results showed that running three rounds of standard AES lead us to propose the NLW-AES which maintains the same level of security as AES practically under specific criteria with a 386% improvement in performance.

Upon our exploration and depending on the conducted experiments and results, we can say that “NLW-AES is a suitable choice for securing real-time IoT applications”.

8.2 FUTURE WORK

The field of security, IoT, and IoT security is still an attractive topic for research due to the great development in devices and their capabilities, in addition to the principle of the tradeoff between security and performance. As future work, in light of the obtained promising results and confirming the consequences of what was discussed in the previous studies, our next step is to work on improving the AES algorithm security when run over one or two rounds by combining AES with another lightweight operation to compensate for the level of security and increase the effectiveness of the one and two rounds on the text being processed. One of the suggested enhancements that should be taken into account, is to replace the s-box with a more efficient one. Furthermore, we should look if there any other security tests can be studied to evaluate AES and the NLW-AES in term of security. Also, the attacks such as Meet-in-the-Middle-attack, and Quantum attack should be applied on AES and NLW-AES to compare the effect of reducing number of rounds from ten to three. Finally, we recommend a new work including the study of a complete security system based on NLW-AES with suitable hashing, digital signature algorithms.

9 REFERENCES

[1]	Y. Alfred, *Network Security*; Malaysia, 2019, pp. 5-11; 10.13140/RG.2.2.19900.59526
[2]	A. Manuj, *Network security with pfSense: Architect, deploy, and operate enterprise-grade firewalls*, 1st ed. PACKT Publishing, 2018.
[3]	S. William, *Cryptography and network security: Principles and practice*, 8th ed. Pearson, 2017.
[4]	Z. Mohammed, E. Ahmed, "Internet of Things Applications, Challenges and Related Future Technologies", *Journal of world scientific news*, pp126-148, 2017.
[5]	https://internetofthingsagenda.techtarget.com/Ultimate-IoT-implementation-guide-for-businesses accessed on Feb 15, 2022.
[6]	S. Mista, , C. Roy, and A. Mukherjee, *Introduction to industrial internet of things and industry 4.0*, 1st ed. CRC Press, 2021.
[7]	L. Qabajeh, "A more secure and scalable routing protocol for mobile ad hoc networks". *Security and Communication Networks*, vol.6, pp. 286-308, 2013.
[8]	I, Makhdoom, M. Abolhasan, and W, Ni, "Blockchain for IoT: The Challenges and a Way Forward", *International Council for Evangelical Theological Education*, pp594-605, 2018.
[9]	O. Salhab, N. Jweihan, M. abuJoodeh, M. Abutaha and M. Farajallah "Survey paper: Pseudo-random number generators and security tests", *Journal of Theoretical and Applied Information Technology*. vol.96, pp. 1951-1970, 2018.
[10]	M. Abutaha, M. Farajallah, R. Tahboub. And M, "Survey Paper: Cryptography Is The Science Of Information Security", I*nternational Journal of Computer Science and Security*, Vol.5, pp. 475, 2011.
[11]	https://geek-university.com/ccna-security/aaa-explained accessed on Feb 15, 2022.
[12]	D. abd Elminaam, HM. Abdual-Kader and MM. Hadhoud, "Evaluating the performance of symmetric encryption algorithms", *IJ Network Security*, vol. 10, pp. 216-222, 2010.
[13]	C. Guanrong, M. Ybin and C. Charles. A symmetric image encryption scheme based on 3D chaotic cat maps. *Chaos, Solitons & Fractals*. vol.21, pp.749-61, 2004.
[14]	S. Zhu, "Algorithm Design Of Secure Data Message Transmission Based On Openssl And Vpn", *Journal Of Theoretical & Applied Information Technology*. vol. 48, 2013.
[15]	M. Bellare, and P. Rogaway, Optimal asymmetric encryption. In *Workshop on the Theory and Application of Cryptographic Techniques*, Springer, Berlin, Heidelberg, pp. 92-111, 1994.
[16]	G. Simmons, "Symmetric and asymmetric Encryption*". ACM Computing Surveys (CSUR)*", vol.1, pp. 305-330, 1979.
[17]	A. Shamir, L. Adleman, and R. Rivest, "A method for obtaining digital signatures and public-key cryptosystems*". Communications of the ACM*, vol. 21, pp. 120–126, 1978.
[18]	A. Alsadeh, and A. Karakra, "A-RSA: Augmented RSA". *In 40th conf of SAI Computing*, 2016, pp. 1016–1023.
[19]	T. Gamal, "A public-key cryptosystem and a signature scheme based on discrete logarithms". *IEEE Transactions on Information Theory*, vol. 31, pp. 469-472, 1985.

[20]	https://www.ques10.com/p/33937/el-gamal-cryptography-algorithm-1/ accessed on Feb 15, 2022.
[21]	U. Maurer, and S. Wolf, "The Diffie–Hellman Protocol", *Designs Codes and Cryptography*, vol.19, pp. 147-171, 2000.
[22]	M. Mihailescu, and S. Nita, "Elliptic-Curve Cryptography", Elliptic-Curve Cryptography. *In: Pro Cryptography and Cryptanalysis. Apress, Berkeley*, CA. https://doi.org/10.1007/978-1-4842-6367-9_1, 2021.
[23]	M. Al-Absi, A. Abdullaev, A. Absi, M. Sain, and H. Lee, "Cryptography Survey of DSS and DSA". *(eds) Advances in Materials and Manufacturing Engineering*, Lecture Notes in Mechanical Engineering. Springer, Singapore, pp 661-669, 2020.
[24]	M. Agrawal, and P. Mishra, "A comparative survey on symmetric key encryption techniques*". International Journal on Computer Science and Engineering*. vol. 4, pp. 877-882, 2012.
[25]	W. Stallings, "*The RC4 Stream Encryption Algorithm*", in Cryptography and network security, 2005.
[26]	S. Mister, and S. Tavares, "*Cryptanalysis of RC4-like Ciphers*", In *Selected areas in cryptography*, vol. 1556, pp. 131-143, 1998.
[27]	M. Robshaw, and O. Billet O, "*New stream cipher designs: the eSTREAM finalists*", Springer; 2008.
[28]	D. Bernstein, "*The Salsa20 Family of Stream Ciphers*", In: Robshaw M., Billet O. (eds) New Stream Cipher Designs. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, vol 4986, 2008.
[29]	C. Berbain, O. Billet, A. Canteaut, N. Courtois, H. Gilbert, Henri, L. Goubin, A. Gouget, L. Granboulan, C. Lauradoux, M. Minier, T. Pornin and H. Sibert, "*SOSEMANUK: a fast software-oriented stream cipher*", Springer, pp. 98-118, 2008.
[30]	D. Coppersmith, C. Holloway, S. Matyas, and N. Zunic, "The data encryption standard". *Information Security Technical Report*, vol. 2, pp. 22-24, 1997.
[31]	J. Daemen, Joan and V. Rijmen, (2002). "The Data Encryption Standard", in The Design of Rijndael, Springer, pp. 81-87, 2002.
[32]	https://www.cryptomathic.com/news-events/blog/3des-is-officially-being-retired. Accessed on Feb 15, 2022.
[33]	P. Bhat, and Deepthi. "Comparison of MD5 and Blowfish Algorithm", *International Journal of Innovative Research in Science, Engineering and Technology*, vol. 5, pp. 506-511, 2016.
[34]	R. C. Kalaiselvi, and M. Vennila, "An Analysis of AES, RSA, and Blowfish-A Review", *The International journal of analytical and experimental modal analysis*, vol. XII, pp. 568-588 ,2020.
[35]	U. Blumenthal, M. Fabio, and M. Keith, "*The advanced encryption standard (AES) cipher algorithm in the SNMP user-based security model*", Bell Labs, USA, No. RFC 3826. 2004.
[36]	M. Dworkin, "*Recommendation for Block Cipher Modes of Operation: Methods and Techniques*", NIST, SP 800-38A, 2001.
[37]	M. Cruz-Cunha, and I. Portela, "*Handbook of Research on Digital Crime, Cyberspace Security, and Information Assurance*", 1st ed. IGI Global, 2014.
[38]	V. Thakor, M.A. Razzaque, and M. Khandaker, "Lightweight Cryptography Algorithms for Resource-Constrained IoT Devices: A Review, Comparison, and Research Opportunities*", IEEE Access*, vol. 9, pp. 28177-28193, 2021.

[39]	R. Muhammad, M. Quazi, and I. Rafiqul, “Current Lightweight Cryptography Protocols in Smart City IoT Networks: A Survey”, A*rXiv*, vol. 2010.00852, 2020.
[40]	H. Manifavas, G. Hatzivasilis, K. Fysarakis, and J. Papaefstathiou, “A survey of lightweight stream ciphers for embedded systems”. Security and Communication Networks, vol. 9, pp. 1226-1246, 2015.
[41]	W. Buchanan, S. Li, Shancang, and R. Asif, “Lightweight cryptography methods”, *Journal of Cyber Security Technology*”, vol.1, pp.187-201, 2018.
[42]	D. Sehrawat, and N. Gill, “Lightweight Block Ciphers for IoT based applications: A Review”, *International Journal of Applied Engineering Research*, vol.13, pp. 2258-2270, 2018.
[43]	I. Dutta, B. Ghosh, and N. Bayoumi, “Lightweight Cryptography for Internet of Insecure Things: A Survey”, *2019 IEEE 9th Annual Computing and Communication Workshop and Conference*, pp. 0475-0481, 2019.
[44]	S. Rajesh, V. Paul, V. Menon, and M. Khosravi, Mohammad, “A Secure and Efficient Lightweight Symmetric Encryption Scheme for Transfer of Text Files between Embedded IoT Devices”, *Symmetry*”. vol. 11, 2019
[45]	N. Hassan, C. Raphaël, P. Congduc, and C. Ali, “Lightweight Stream Cipher Scheme for Resource-Constrained IoT Devices”, in *15th IEEE WiMob 2019At conf*, Barcelona, Spain, 2019.
[46]	N. Gunathilake, W. Buchanan, and R. Asif, Rameez, “Next Generation Lightweight Cryptography for Smart IoT Devices: Implementation, Challenges and Applications”, *IEEE 5th World Forum on Internet of Things (WF-IoT)*, pp.707-710, 2019.
[47]	M. Usman, “Lightweight Encryption for the Low Powered IoT Devices”, *arXiv*, vol. 2012.00193, 2020.
[48]	M. Abu-tair, S. Djahel, P. Perry, B. Scotney, U. Zia, J. Carracedo, and A. Sajjad, “Towards Secure and Privacy-Preserving IoT Enabled Smart Home: Architecture and Experimental Study”, *Sensor*, vol. 20, 2020.
[49]	M. Khalifa, F. Algarni, M. Khan, A. Ullah, and K. Aloufi, “A lightweight cryptography (LWC) framework to secure memory heap in Internet of Things”, *Alexandria Engineering Journal*, vol. 60, pp. 1489-1497, 2020.
[50]	R. Ramadan, B. Aboshosha, K. Yadav, I. Alseadoon, M. Kashout, and M. Elhoseny, “LBC-IoT: Lightweight Block Cipher for IoT Constraint Devices”, *Cmc-computers Materials \& Continua*, vol. 67, pp.3563-3579, 2021.
[51]	P. Prakasam, M. Madheswaran, K.P. Sujith, and S. Shohel, “An Enhanced Energy Efficient Lightweight Cryptography Method for various IoT devices”, *ICT Express*, vol. 7, pp. 487-492, 2021.
[52]	F. Thabit, S. Alhomdy, A. Al-ahdal, Abdulrazzaq, and P. Jagtap, “A New Lightweight Cryptographic Algorithm for Enhancing Data Security In Cloud Computing”, *Global Transitions*, vol.2, pp. 91-99, 2021.
[53]	A. Javed, "Fast implementation of AES on mobile devices", *Proc. 8th Int. Netw. Conf.*, pp. 133-142, 2010.
[54]	P. Abhijith, M. Goswami, S. Tadi, and K. Pandey, “Optimized Architecture for AES. *Cryptology ePrint Archive: Report 2014/540*, 2014.
[55]	D. Bui, D. Puschini, S. Bacles-Min, E. Beigné, and X.-T. Tran “AES Datapath Optimization Strategies for Low-Power Low-Energy Multisecurity-Level Internet-of-Thing Applications”, *IEEE Transactions on Very Large-Scale Integration (VLSI) Systems*, vol. 25, pp. 3281-3290., 2017.

[56]	A. Mamun, S. Rahman, T. Shaon, and A. Hossain, "Security Analysis of AES and Enhancing its Security by Modifying S-Box with an Additional Byte", *International Journal of Computer Networks &Communications*, vol.9, pp.69-88, 2017.
[57]	U. Farooq, and F. Aslam, "Comparative analysis of different AES implementation techniques for efficient resource usage and better performance of an FPGA", *Journal of King Saud University - Computer and Information Sciences*, vol. 29, pp. 295-302, 2017.
[58]	L. Daoud, F. Hussein, and N. Rafla, "Optimization of Advanced Encryption Standard (AES) Using Vivado High Level Synthesis (HLS)", *Proceedings of 34th International Conference on Computers and Their Applications*, pp. 36-44, 2019.
[59]	R. Naif, G. H. Abdul-Majeed and A. K. Farhan, "Secure IOT System Based on Chaos-Modified Lightweight AES", *2019 International Conference on Advanced Science and Engineering (ICOASE)*, pp.1-6, 2019.
[60]	U. Farooq, M. Mushtaq, and M. Bhatti, "Efficient AES Implementation for Better Resource Usage and Performance of IoTs" in *CYBER 2020 - 5th International Conference on Cyber-Technologies and Cyber-Systems*, Nice, France, 2020.
[61]	E. Nagalakshmi, V. Mohan, and D. Kumar, "AES DATAPATH OPTIMIZATION STRATEGIES FOR LOW-POWER LOW-ENERGY MULTI SECURITY-LEVEL INTERNET-OF-THING APPLICATIONS", *International Journal of Advanced Research in Science, Engineering and Technology*, vol. 2, pp. 347-355. 2020.
[62]	P. Freyre, O. Cuellar, N. Díaz, and A. Alfonso, "From AES to Dynamic AES", *Journal of Science and Technology on Information security*, vol. 1, pp. 11-22, 2020.
[63]	K. Salim, S. Alalak, and M. Jawad, "Improved Image Security in Internet of Thing (IoT) Using Multiple Key AES", *Baghdad Science Journal*, vol. 18, pp. 417-429, 2021.
[64]	A. Singh," Comparative Analysis of Reduced Round Dynamic AES with Standard AES Algorithm", *International Journal of Computer Applications*, vol. 183, pp. 41-49, 2021.
[65]	L. Pan, G. Tu, S. Liu, Z. Cai, and X. Xiong, "A Lightweight AES Coprocessor Based on RISC-V Custom Instructions", *Security and Communication Networks*, vol. 2021, pp.1-13, 2021.
[66]	S. Jagtap, F. Thabit, and S. Alhomdy, "Security Analysis and Performance Evaluation of a New Lightweight Cryptographic Algorithm for Cloud Computing Environment", *Global Transitions Proceedings*, vol 2, pp. 100-110, 2021.
[67]	B. Coskun and N. Memon, "Confusion/Diffusion Capabilities of Some Robust Hash Functions". *40th conf. Information Sciences and Systems*, pp.1188 – 1193, 2006.
[68]	M. Farajallah, M. Abutaha, M. Abujoodeh, O. Salhab and N. Jweihan, "PSEUDO-RANDOM NUMBER GENERATOR BASED ON LOOK-UP TABLE AND CHAOTIC MAPS", *Journal of Theoretical and Applied Information Technology*, Vol 98. 3130, 2020.
[69]	G. Chen, Y. Mao and Charles K. Chui "A symmetric image encryption scheme based on 3D chaotic cat maps", *Chaos, Solitons & Fractals*, Vol. 21, pp. 749-761, 2004.

10 APPENDIX

In this chapter, we provide numerical calculation examples for the results that have been discussed in this thesis.

- **386% improvement in NLW-AES performance.**

Using equation 11 to calculate the Gain:

$$Gain = \pm \frac{Scenario\ Score - AES\ Score}{AES\ Score} * 100\% \quad (11)$$

$$Gain = \pm \frac{68 - 14}{14} * 100\%$$

$$Gain = +385.7$$

- **20.5 score in NLW-AES security.**

Using equation 13:

$$Security\ score = \frac{(CBC\&CFB\ score) + ECB\ score + OFB\ score + CTR\ core}{4} \quad (13)$$

Where,

$$Security\ score = \frac{(23) + 17 + 21 + 21}{4} = 20.5$$

$$Mode\ Security\ Score = \sum_{Mapping}^{NIST} G \qquad (12)$$

$$Mode\ Security\ Score\ (CBC) = 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 15 = 23$$

$$CBC\&CFB\ score = \frac{CBC\ score + CFB\ score}{2} \qquad (14)$$

$$CBC\&CFB\ score = \frac{23 + 23}{2} = 23$$