

إقرار

أنا الموقع أدناه مقدم الرسالة التي تحمل العنوان:

Enhancing of CryptoBI Algorithm for Data Security in Local Area Network

أقر بأن ما اشتملت عليه هذه الرسالة إنما هو نتاج جهدي الخاص، باستثناء ما تمت الإشارة إليه
حيثما ورد، وإن هذه الرسالة ككل أو أي جزء منها لم يقدم من قبل لنيل درجة أو لقب علمي أو
بحثي لدى أي مؤسسة تعليمية أو بحثية أخرى.

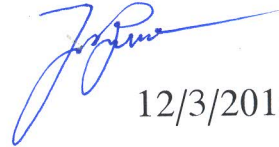
DECLARATION

The work provided in this thesis, unless otherwise referenced, is the
researcher's own work, and has not been submitted elsewhere for any
other degree or qualification

Student's name: Zakaria M. Abusilmiyeh

اسم الطالب: زكريا محمد أبو سلمية

Signature:

التوقيع: 

Date:

التاريخ: 12/3/2015

Islamic University of Gaza
Deanery of Graduate Studies
Faculty of Information Technology
Information Technology Program



Enhancing of CryptoBI Algorithm for Data Security in Local Area Network

By

Zakaria M. Abusilmiyeh

120103344

Supervised By

Dr. Tawfiq S. Barhoom

A Thesis Submitted in Partial Fulfillment of the Requirements for the

Degree of Master in Information Technology

March 2015



نتيجة الحكم على أطروحة ماجستير

بناءً على موافقة شئون البحث العلمي والدراسات العليا بالجامعة الإسلامية بغزة على تشكيل لجنة الحكم على أطروحة الباحث/ زكريا محمد محمد أبو سلمية لنيل درجة الماجستير في كلية تكنولوجيا المعلومات برنامج تكنولوجيا المعلومات وموضوعها:

تحسين خوارزمية CryptoBI لأمن البيانات في الشبكة المحلية

Enhancing OF CryptoBI Algorithm for Data Security in Local Area Network

وبعد المناقشة التي تمت اليوم الأحد 10 جماد أول 1436هـ، الموافق 2015/03/01م الساعة العاشرة صباحاً، اجتمعت لجنة الحكم على الأطروحة والمكونة من:

	مشرفاً ورئيساً	د. توفيق سليمان برهوم
	مناقشاً داخلياً	أ.د. أشرف محمد العطار
	مناقشاً خارجياً	د. سناء وفا الصايغ

وبعد المداولة أوصت اللجنة بمنح الباحث درجة الماجستير في كلية تكنولوجيا المعلومات / برنامج تكنولوجيا المعلومات.

واللجنة إذ تمنحه هذه الدرجة فإنها توصيه بتقوى الله وئزوم طاعته وأن يسخر علمه في خدمة دينه ووطنه.

والله ولي التوفيق،،،

مساعد نائب الرئيس للبحث العلمي والدراسات العليا



أ.د. فؤاد علي العاجز

Dedication

To my family for their love, endless support, and encouragement

Acknowledgements

I wish to express my deepest gratitude to my advisor **Dr. Tawfiq Barhoom** for his continued support, guidance and encouragement, without which it would not have been possible to succeed in my research. I would like to thank the **administration of Gaza Training Community College “GTC”** for providing me facilities and tools that helped me in my research study.

Special thanks go to all my roommates and friends for making my stay in Toledo, a memorable one. Above all, I wish to express my deep appreciation to my family and friends whose love and encouragement have made all this possible.

Zakaria M. M. Abusilmieh

2015

Enhancing of CryptoBI Algorithm for Data Security in Local Area Network

By Zakaria M. Abusilmiyeh

Abstract

The CryptoBI version1 “A Novel Cryptography Method Based on Image for Key Generation” algorithm is a new type of symmetric encryption algorithm that prevents the attacks and avoids key exchange between the two sides of communication by creating the key in home on the sender/receiver machine. In this research, we implemented the algorithm CryptoBI version1 and created a prototype for a specific scenario to measure the performance and the strength against the cryptanalysis attack. As a result of this measurement, we found that the algorithm was weak against the cryptanalysis attack because it can be broken in a short time in minutes. In addition, we found that the algorithm was less efficient than the Blowfish. According to unsatisfied results of these measurements, the performance and the strength of the CryptoBI version 1 are enhanced, the performance enhanced by removing the binarize step from the encryption/decryption algorithms but the strength against the cryptanalysis attack enhanced by randomize the scanning starting point of the key generation algorithm. The enhanced CryptoBI called (CryptoBI version2). The experiment results showed that the enhanced algorithm (CryptoBI version2) has higher data rate than CryptoBI version1, AES, Rijndael, DES, 3DES, RC2, RC6 and Blowfish algorithms. Furthermore, the CryptoBI version2 is stronger than the CryptoBI version1 against the cryptanalysis attack. This process is more flexible that any RGB image can be used for key generation as the key generation is directly based on the image content. The algorithm can generate a key of more than 512 bits according to data size.

Keywords: *Cryptography, Cryptanalysis, Key Generation Based on Image, Image Key.*

عنوان البحث

تحسين خوارزمية CryptoBI لأمن البيانات في الشبكة المحلية

الملخص

الخوارزمية CryptoBI version1 " طريقة جديدة للتشفير بالاعتماد على صورة لتوليد مفتاح التشفير " الذي بدوره هو عبارة عن نوع جديد من خوارزميات التشفير المتناظر الذي يمنع الهجمات ويلغي تبادل المفتاح بين جانبيين الاتصال عن طريق انتاج المفتاح محليا على جهاز المرسل / المستقبل. في هذا البحث، نفذنا الخوارزمية CryptoBI version1 وانشأنا نموذج لسيناريو محدد لقياس الأداء والقوة ضد هجوم تحليل الشيفرات. ونتيجة لهذا القياس، وجدنا أن الخوارزمية CryptoBI version1 كان بها ضعف ضد هجوم تحليل الشيفرات لأنه يمكن كسرها في وقت قصير في دقائق. وبالإضافة إلى ذلك، وجدنا أن الخوارزمية CryptoBI version1 كانت أقل كفاءة من الخوارزمية Blowfish. وفقا للنتائج الغير مرضية عن تلك القياسات، قمنا بتحسين وتعزيز الأداء وقوة الخوارزمية CryptoBI version1، وتم تحسين الأداء عن طريق إزالة خطوة تحويل البيانات الى النظام الثنائي في خوارزميات التشفير/ فك التشفير. وكذلك تم تحسينه ضد هجوم تحليل الشيفرات من خلال اضافة العشوائية على نقطة البدء للمسح لإيجاد المفتاح في خوارزمية توليد المفتاح. الخوارزمية المحسنة تدعى (CryptoBI version2). وأظهرت نتائج التجارب لبحثنا أن الخوارزمية المحسنة (CryptoBI version2) تمتلك معدل أداء أعلى من الخوارزميات CryptoBI version1، AES، Rijndael، DES، 3DES، RC2، RC6، Blowfish. وعلاوة على ذلك، فإن CryptoBI version2 أقوى من CryptoBI version1 ضد هجوم تحليل الشيفرات. هذه العملية هي أكثر مرونة بحيث يمكن استخدام أي صورة من نوع RGB لتوليد المفتاح بالاستناد مباشرة على المحتوى. ويمكن للخوارزمية توليد مفتاح بحجم أكثر من 512 بت وفقا لحجم البيانات.

الكلمات المفتاحية: الترميز، وتحليل الشيفرات، توليد المفتاح بالاعتماد على الصور، الصورة كمفتاح.

Table of Contents

Dedication.....	i
Acknowledgements.....	ii
Abstract.....	iii
List of Tables.....	viii
List of Figures.....	ix
List of Abbreviations.....	x
Chapter 1. Introduction.....	1
1.1 Background Information.....	1
1.2 Problem Statement.....	2
1.3 Objectives:.....	3
1.1. Specific objectives.....	3
1.4 Scope and limitation:.....	3
1.5 Methodology.....	4
1.6 Thesis Organization.....	4
Chapter 2. Theory Background.....	5
2.1 Introduction.....	5
2.1.1 Symmetric-key Cryptography.....	6
2.1.2 Asymmetric-key Cryptography.....	6
2.2 Cryptography Goals.....	7
2.3 Terminology of Symmetric Cryptography Algorithms.....	8
2.4 Symmetric cryptography Modes.....	11
2.5 Cryptanalysis.....	11
2.6 Conclusion.....	14
Chapter 3. Related works.....	15

3.1	Cryptography key generation based image	15
3.1.1	Key Generation Method Using Image Features.....	15
3.1.2	Key Generation Based on Face Features	16
3.1.3	Randomized Cryptographic Key Generation Using Image	18
3.2	Avoids Key Exchange.....	19
3.3	Conclusion.....	19
Chapter 4.	Design and System Model	21
4.1	Introduction	21
4.2	The System Model for CryptoBI version1	22
4.2.1	Database Creation for CryptoBI v1:	23
4.2.2	Binarize Message for CryptoBI v1:	24
4.2.3	LSB of RGB for CryptoBI v1:.....	24
4.2.4	Key Generation for CryptoBI v1:	25
4.2.5	Encryption/ Decryption Algorithm for CryptoBI v1:	27
4.3	The System Model for CryptoBI v2.....	30
4.3.1	Database Creation for CryptoBI v2:	31
4.3.2	LSB of RGB for CryptoBI v2:.....	32
4.3.3	Key Generation for CryptoBI v2:	32
4.3.4	Encryption/ Decryption Algorithm for CryptoBI v2:	35
4.4	Conclusion.....	38
Chapter 5.	Experiments and Results.....	39
5.1	Experimental Design	39
5.1.1	Simulation Setup.....	39
5.1.2	System Parameters	42
5.1.3	Experiment Factors	43

5.1.4	Simulation Procedure.....	44
5.2	Experimental Results.....	45
5.2.1	CryptoBI Key-Changing Principle	45
5.2.2	Differentiate Output Results of Encryption	46
5.2.3	Effect of Changing Data Size for Cryptographic Algorithms on Power Consumption.....	48
5.2.4	The Effect of Changing Key Size of CryptoBI on Power Consumption	51
5.2.5	The Effect of Cryptanalysis Attack against the Strength of CryptoBI Algorithms.	52
5.3	Conclusion.....	58
Chapter 6.	Conclusions and Future Directions	59
6.1.1	The conclusion	59
6.1.2	The Future Directions	61
	Bibliography	62

List of Tables

Table 2-1: Types of Attacks on Encrypted Messages.....	12
Table 2-2: Average Time Required for Exhaustive Key Search	14
Table 3-1: Algorithm Phases	15
Table 3-2: Feature selection from principal component analysis	16
Table 5-1: Algorithms settings.....	40
Table 5-2: Comparative execution times (in milliseconds) of the encryption algorithms.....	47
Table 5-3: Throughput “Average data rates” comparison at encryption stage.....	48
Table 5-4 : Throughput “Average data rate” comparison at decryption stage.	50
Table 5-5: Some of cryptanalysis result for CryptoBI v1.....	54
Table 5-6: Some of cryptanalysis result for CryptoBI v2.....	55
Table 5-7: Time Required for Exhaustive Key Search.....	58
Table 6-1: Most important differences between the CryptoBI version1 and version2.....	60

List of Figures

Figure 2-1: Cryptography System [11].	5
Figure 2-2 : Taxonomy of Cryptography [14].	6
Figure 2-3: A Typical Encryption Procedure [14].	9
Figure 2-4: The Encryption and Decryption Procedure of a Feistel Cipher [14].	10
Figure 3-1: The framework of proposed scheme [30]	17
Figure 4-1: the sender and the receiver block diagram	22
Figure 4-2: Binary Array of the Message “1Testing“	24
Figure 4-3: Key generation phase and the key array	26
Figure 4-4: the encryption process.	28
Figure 4-5: the sender and the receiver block diagram for CryptoBI v2.	31
Figure 4-6: the image scanning starting point for key generation algorithm.	33
Figure 4-7 : The Encryption Process steps for each data byte.	35
Figure 4-8: CryptoBI v2 encryption process.	36
Figure 5-1: The main form of the simulation program, works as receiver.	40
Figure 5-2: The sending form of the simulation program	41
Figure 5-3: The CryptoBI Cryptanalysis from.	42
Figure 5-4: Wired Local Area Network	43
Figure 5-5: The plain text.	45
Figure 5-6: The encryption of the plaintext during session 1.	45
Figure 5-7: The encryption of the plaintext during session 2.	46
Figure 5-8: The decryption of the plaintext.	46
Figure 5-9: Time consumption of encryption algorithm	47
Figure 5-10: Throughput of each encryption algorithm (MB/Sec).	49
Figure 5-11: Throughput of each decryption algorithm (MB/Sec).	51
Figure 5-12: Time consumption for different key size for CryptoBI v1.	52
Figure 5-13: Time consumption for different key size for CryptoBI v2.	52
Figure 5-14: CryptoBI v1 ciphertext for cryptanalysis	53
Figure 5-15: CryptoBI v2 ciphertext for cryptanalysis	55

List of Abbreviations

LSB _____ Least Significant Bit

RGB _____ Color image channels Red, Green and Blue

ECB _____ Electronic Code Book

CBC _____ Cipher Block Chaining

CFB _____ Cipher Feedback

OFB _____ Output Feedback

ECC _____ Elliptic Curve Cryptography

PGP _____ Pretty Good Privacy

PKI _____ Public key infrastructure

Chapter 1. Introduction

This chapter introduces the content of the thesis work “Enhancing of CryptoBI Algorithm for Data Security in Local Area Network”. The chapter is divided into several sections discussing the problem statement and research approach. In addition, the organization of this thesis report is presented.

1.1 Background Information

In the information age, sharing and transferring of data has increased tremendously and usually the information exchange is done using open channels which can make it vulnerable to interception. The threat of an intruder accessing secret information has been an ever existing concern for the data communication experts [1]. Steganography and cryptography are two important tools for protecting information.

Cryptography presents various methods for taking legible, readable data, and transforming it into unreadable data for the purpose of secure transmission, and then using a key to transform it back into readable data when it reaches its destination [2]. Cryptography is considered to be one of the fundamental building blocks of computer security [3]. The need of reliable and effective security mechanisms to protect information systems is getting increased due to the rising magnitude of identity theft in our society. The cryptography has been proposed to ensure the confidentiality and authenticity of the message. The message confidentiality means that the enterprise messages usually contain confidential data that should be kept secret and not known to third parties at all times, but authenticity of the message refers to the operation of checking message contents and assuring contents remain unchanged and authenticity of source [4].

Key exchange (also known as "key establishment") is any method in cryptography by which cryptographic keys are exchanged between sides of communication, allowing the use of a cryptographic algorithm. If sender and receiver wish to exchange encrypted messages, each of them must be equipped to encrypt messages to be sent and decrypt messages to be received. The nature of the equipping they require depends on the encryption technique they might use. If they use a code, both will require a copy of the same codebook. If they use a cipher, they will need appropriate keys. If the cipher is a symmetric key cipher, both will need a copy of the same key.

If using an asymmetric key cipher with the public/private key property, both will need the other's public key [5]. Although cryptography is a powerful tool to achieve information security, the security of cryptosystems relies on the fact that cryptographic keys are secret and known only to the legitimate user [6]. Current digital webcam is capable of capturing high quality image. Thus, user's image from the webcam can be a good source of random cryptographic key [7]. Generate a key based on image which is very cheap, cost effective, convenient and universal. The attacker could not derive the key from the image. And also it does not require any additional devices. Small change in the image should lead to a significant difference in the generated key [8].

1.2 Problem Statement

The protection of the confidentiality of encryption keys is one of the important issues to be dealt with. We need to protect the privacy of key encryption, when the key is long, it is difficult to remember it, storing it in database or in a file may be insecure, and exchanging the key between the two sides over public network is insecure, too.

The CryptoBI version1 [9] is a new Cryptography method based on the key that is generated directly from an image stored in the database and the process of key generation based on sessions. The CryptoBI algorithm avoids key exchange between the two sides of communication and solves this issue efficiently by generating the key before starting the process of encryption and decryption, rather than storing it. The CryptoBI version1 support key-updating technique where the length of Key varies according to the size of the message, as is vary every session according to the session type. The algorithm efficiency did not examine within the local network, where it use by a number of users in data transfer among them safely, and also, the algorithm strength did not examine against any attack to break the ciphertext and access to information.

Through in-depth analysis for the algorithm, it was conclude that the efficiency of the algorithm could be increase by replacing the bit operation by the byte operations, and the algorithm become weak and can be broken in case of knowing the images database by an attacker.

This work problem is to enhance the strength and the performance of the CryptoBI version1 in key generation and encrypting /decrypting the data, which have been transmitting between the two sides in the LAN.

1.3 Objectives:

The main objective of this research is to evaluate the performance and the strength of CryptoBI version1 algorithm to make a necessary updates on the algorithm to enhance its strength and performance.

1.1. Specific objectives

1. Explore more about cryptography key based on image methods to get more understanding of the problem.
2. Implement the CryptoBI version1 system model and create a prototype for a scenario to work in LAN by many users, the model include three sub models and they are:
 - 2.1. Implement algorithm to generate the key.
 - 2.2. Implement algorithm to encrypt the data.
 - 2.3. Implement algorithm to decrypt the data.
3. Evaluate the model by using methods such as performance, cryptanalysis and key updating. I will use the cryptanalysis according to a specific scenario that is can be used for this purpose, because it is hard to find a tool that can be used in general, all the tools created and published were to a specific algorithm.
4. Determine the strengths and weakness points in the model.
5. Analysis the weakness points in the model.
6. Find solutions to get over the weakness points of the model.
7. Update the model algorithms and create a new model.
8. Evaluate the new enhanced model using the methods as in the objective 4.

1.4 Scope and limitation:

The work can be applied with some limitations and assumption such as:

1. The users can be communicating with each other's through Ethernet "LAN" or private business network with small number of users. Why LAN not WAN? One way to think about the difference is not in terms of size or geographical location only, but control. LAN is a network under the control and jurisdiction of a single entity; the WAN is everything else. On the other hand, they are different in the security measurements, so we

need to test the model in a small controlled business area even business users route their traffic between them publicly, over the WAN.

2. The user can use the system to transmit data such as text message or files.
3. Synchronization between the both sides of communication, they have to agree on many setting before they can communicate with each other which they are:
 - a. Database of RGB images
 - b. Image RGB channel
 - c. Session type (for example hourly session)

1.5 Methodology

In this work, we are going to implement the CryptoBI algorithm by creating a prototype for a specific scenario to evaluate the performance and the power of the algorithm in securing the data which have been transmitting through the local area network in order to find the weakness of the algorithm, if there is any, and to enhance the algorithm as needed by making the necessary changes on the algorithm. Simplicity is the spirit of this work, we need to encrypt and decrypt the user and / or application data very easily. Also we need to create complications for the eavesdropper to decrypt the cipher text, but for the end-users this algorithm does not pose any complications to perform functional activities.

1.6 Thesis Organization

The rest of the work is organized as follows:

Chapter 2 is devoted to literature survey on cryptography, presenting the history and background knowledge about cryptography. Chapter 3 is about related works. Chapter 4 describes the system model and introduces the CryptoBI algorithms. Chapter 5 discusses the experiments and the results. Chapter 6 gives the conclusion of the work done and future work.

Chapter 2. Theory Background.

This chapter discusses the background concepts of this work. Basic cryptography theory and cryptography goals will be discussed. Terminology of symmetric cryptography algorithms, symmetric cryptography modes, cryptanalysis and other information are provided.

2.1 Introduction

Cryptography is the study of the mathematical techniques related to aspects of Information security such as confidentiality, data integrity and authentication. The basic idea of cryptography is to transmit information over an insecure channel and ensure that no entity in the middle can understand what is being transmitted. The history of cryptography goes back all the way to the Egyptians about 4000 years ago. It was also used extensively during the world wars [10].

As an illustration we can consider two people, say Alice and Bob, who wish to transmit information back and forth in such a way that a third person Oscar who is an opponent and can listen to the transmission but cannot decipher the information being transmitted. The information Alice intends to send is called the plaintext since it is not encrypted yet. Alice encrypts the plaintext with a predetermined key called the cipher key and transmits the ciphertext over the insecure channel to Bob. Bob knowing the cipher key can decipher the ciphertext to obtain the plaintext. Oscar, who does not have the cipher key, cannot decode the ciphertext [11]. Figure 2-1 illustrates this.

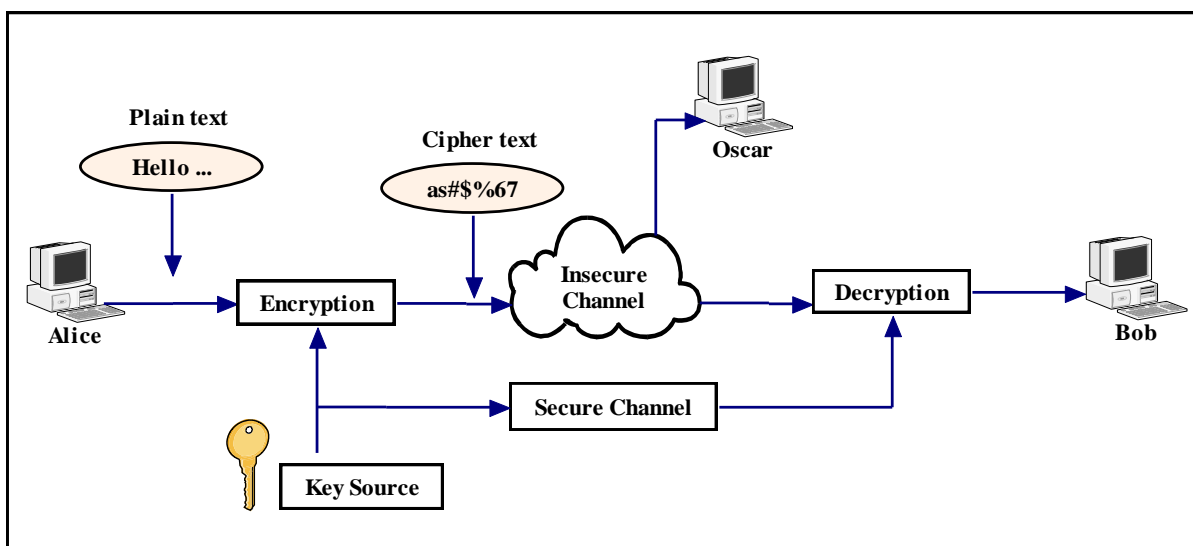


Figure 2-1: Cryptography System [11].

Cryptography is probably the most important aspect of communications security and is becoming increasingly important as a basic building block for computer security [12]. The increased use of computer and communications systems by industry has increased the risk of theft of proprietary information. Although these threats may require a variety of countermeasures, encryption is a primary method of protecting valuable electronic information [13].

Modern cryptography can be divided into two main subfields of study: Symmetric-key and Asymmetric-key cryptography. Symmetric-key can be divided into block ciphers and stream ciphers. Figure 2-2 depicts the taxonomy of cryptography.

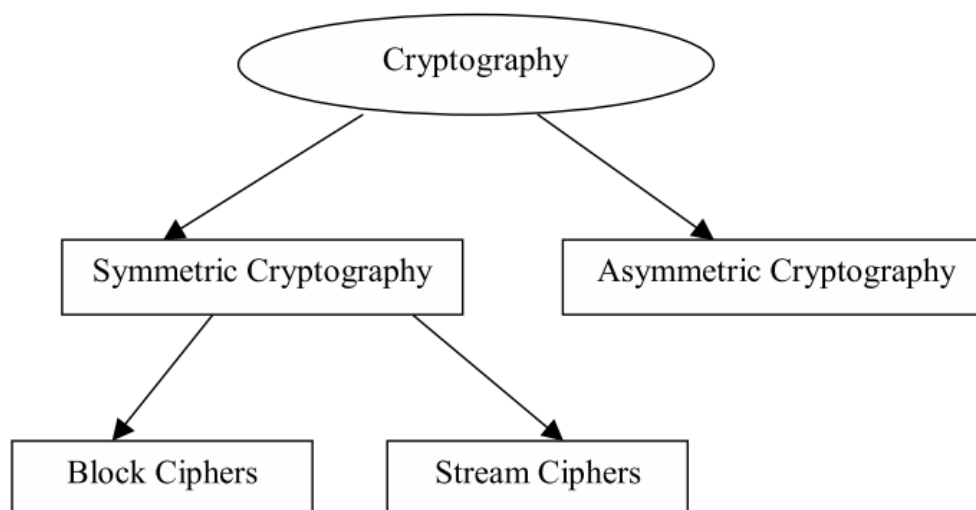


Figure 2-2 : Taxonomy of Cryptography [14].

2.1.1 Symmetric-key Cryptography

Symmetric-key cryptography refers to encryption methods in which both the sender and receiver share the same key (or, less commonly, in which their keys are different, but related in an easily computable way). This was the only kind of encryption publicly known until June 1976 [15]. The modern study of symmetric-key ciphers relates mainly to the study of block ciphers and stream ciphers and their applications.

We will discuss the terminology involved in symmetric-key cryptography in sections 2.3 and 2.4.

2.1.2 Asymmetric-key Cryptography

Asymmetric-key cryptography is also known as public-key cryptography where by two different but mathematically related keys are used a public key and a private key [16]. A public-key

system is so constructed that calculation of one key (the 'private key') from the other (the 'public key') is computationally infeasible, even though they are necessarily related. Instead, both keys are generated secretly, as an interrelated pair [17]. The public key may be freely distributed, while its paired private key must remain secret. The public key is typically used for encryption, while the private or secret key is used for decryption. The most famous applications of public-key cryptography are Elliptic-curve cryptography, PGP and the public-key infrastructure (PKI).

Elliptic Curve Cryptography (ECC) provides the highest strength-per-key-bit of any cryptography algorithm known to take. Compared with other public-key approaches, ECC not only has the higher security but also has lower computation overhead, shorter key size and narrower bandwidth. Therefore, the experts believe that ECC will become the next generation widely used public-key cryptography.

Pretty Good Privacy (PGP) is a computer program that provides cryptographic privacy and authentication. PGP is often used for signing, encrypting and decrypting e-mails to increase the security of e-mail communication.

A PKI (public key infrastructure) enables users of a basically unsecured public network such as the Internet to securely and privately exchange data and money through the use of a public and a private cryptographic key pair that is obtained and shared through a trusted authority. PKI provides single sign on where a single login to the infrastructure system enables them to use that authentication token or service transparently through the rest of the infrastructure [11].

2.2 Cryptography Goals

There are five main goals of cryptography. Every security system must provide a bundle of security functions that can assure the secrecy of the system. These functions are usually referred to as the goals of the security system. These goals can be listed under the following five main categories [18]:

- **Authentication:** The process of proving one's identity. This means that before sending and receiving data using the system, the receiver and sender identity should be verified.
- **Privacy/confidentiality:** Ensuring that no one can read the message except the intended receiver. Usually this function is how most people identify a secure system. It means that only the authenticated people are able to interpret the message content and no one else.

- **Integrity:** Assuring the receiver that the received message has not been altered in any way from the original. The basic form of integrity is packet check sum in IPv4 packets.
- **Non-repudiation:** A mechanism to prove that the sender really sent this message. Means that neither the sender nor the receiver can falsely deny that they have sent a certain message.
- **Service Reliability and Availability:** Since secure systems usually get attacked by intruders, which may affect their availability and type of service to their users. Such systems provide a way to grant their users the quality of service they expect.

2.3 Terminology of Symmetric Cryptography Algorithms

The terminology of symmetric cryptography algorithms mainly includes the following:

Plaintext: Plain text is the ordinary information which the sender wishes to transmit to the receiver(s).

Ciphertext: The encrypted text is called Ciphertext.

Encryption and Decryption: Encryption is the process of converting plain text into ciphertext as shown in Figure 2-3. Decryption is the reverse process, moving from ciphertext back to the original plaintext.

Cipher: A cipher is a pair of algorithms which ensure the encryption and the reversing decryption. The detailed operation of a cipher is controlled both by the algorithm and by a specific key.

Key: The key is a secret parameter for encrypting or decrypting a specific message exchange context. Keys are important, as ciphers without keys are trivially breakable and therefore less than useful for most purposes.

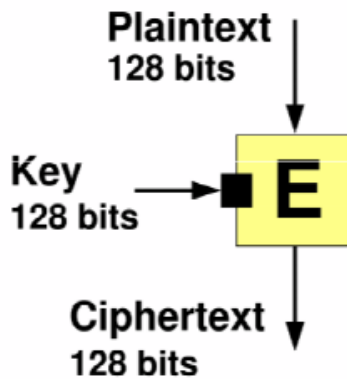


Figure 2-3: A Typical Encryption Procedure [14].

Block Ciphers: The block cipher is a type of symmetric-key encryption algorithm that transforms a fixed-length block of plain text data into a block of ciphertext data of the same length. This transformation takes place under the action of a user provided secret key. Decryption is performed by applying the reverse transformation to the ciphertext block using the same secret key. The fixed length is called the block size and, for many block ciphers, the block size is 64 bits. In the coming years, the block size will increase to 128 bits as processors become more sophisticated. Since messages are almost always longer than a single block, some method of knitting together successive blocks is required. The different ways of knitting together blocks are known as the modes of operation and must be carefully considered when using block ciphers. Mode of operation will be further elaborated in section 4. Block ciphers can be easier to implement in software than stream ciphers, because they often avoid time consuming bit manipulations and they operate on data in computer-sized blocks [19].

Stream Ciphers: A stream cipher is a symmetric-key cipher where plaintext bits are combined with a pseudo-random cipher bit-stream (key stream), typically by an exclusive-or (xor) operation. In a stream cipher the plaintext digits are encrypted one at a time, and the transformation of successive digits varies during the encryption. Stream ciphers typically execute at a higher speed than block ciphers and have lower hardware complexity [19]. Stream ciphers that only encrypt and decrypt data one bit at a time are not really suitable for software implementation [19]. This explains why stream ciphers can be better implemented in hardware than block ciphers.

Feistel - Cipher: A Feistel cipher is a symmetric structure used in the construction of block ciphers; it is also commonly known as a Feistel network. A large proportion of block ciphers use the scheme, including Blowfish, DES, MYSTY1, RC5, TWOFISH, XXTEA, GOST, etc. The Feistel structure has the advantage that encryption and decryption operations are very similar, even identical in some cases, requiring only a reversal of the key schedule. Therefore, the size of the code or the circuitry required to implement such a cipher is nearly halved [14]. Feistel networks combine multiple rounds of repeated operations, such as:

- Bit-shuffling functions (often called permutation boxes or P-boxes).
- Simple, non-linear functions (often called substitution boxes or S-boxes).
- Linear mixing (in the sense of modular algebra) using XOR operations.

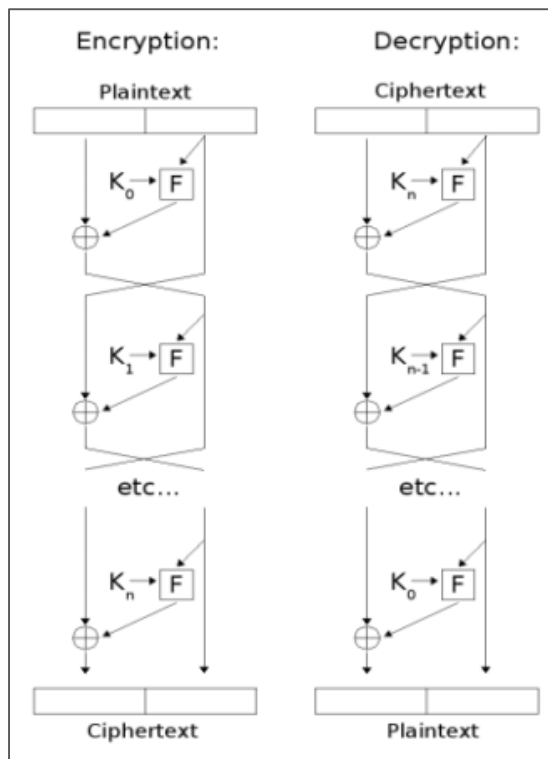


Figure 2-4: The Encryption and Decryption Procedure of a Feistel Cipher [14].

Key Size: key size or key length is the size of the key used in a given cryptographic algorithm. An algorithm's key length is distinct from its cryptographic security, which is a logarithmic measure of the fastest known computational attack on the algorithm, also measured in bits [20].

Rounds: Rounds is the number of iterations in a cipher system. From Figure 2-4 we can have a clear view that each repeated operations stand for a round. According to the crypto analysts, the

bigger the number of rounds, the more secure the algorithms will be. The downside is that the execution time of the algorithms increases enormously.

2.4 Symmetric cryptography Modes

In cryptography, a mode of operation is an algorithm that uses a block cipher to provide an information service such as confidentiality or authenticity [21]. A block cipher by itself is only suitable for the secure cryptographic transformation (encryption or decryption) of one fixed-length group of bits called a block [22]. A mode of operation describes how to repeatedly apply a cipher's single-block operation to securely transform amounts of data larger than a block [23] [24] [25].

Most modes require a unique binary sequence, often called an initialization vector (IV), for each encryption operation. The IV has to be non-repeating and, for some modes, random as well. The initialization vector is used to ensure distinct ciphertexts are produced even when the same plaintext is encrypted multiple times independently with the same key [26]. Block ciphers have one or more block size(s), but during transformation the block size is always fixed. Block cipher modes operate on whole blocks and require that the last part of the data be padded to a full block if it is smaller than the current block size [22]. There are, however, modes that do not require padding because they effectively use a block cipher as a stream cipher.

There are several different modes of symmetric cryptography. ECB, CBC, OFB, and CFB are the earliest modes of operation that have been defined where they provide confidentiality, but they do not protect against accidental modification or malicious tampering [27].

2.5 Cryptanalysis

Cryptanalysis refers to the study of ciphers, ciphertext, or cryptosystems (that is, to secret code systems) with a view to finding weaknesses in them that will permit retrieval of the plaintext from the ciphertext, without necessarily knowing the key or the algorithm. This is known as breaking the cipher, ciphertext, or cryptosystem [28].

Typically, the objective of attacking an encryption system is to recover the key in use rather than simply to recover the plaintext of a single ciphertext. There are two general approaches to attacking a conventional encryption scheme:

Cryptanalysis: Cryptanalytic attacks rely on the nature of the algorithm plus perhaps some knowledge of the general characteristics of the plaintext or even some sample plaintext-ciphertext pairs. This type of attack exploits the characteristics of the algorithm to attempt to deduce a specific plaintext or to deduce the key being used.

Brute-force attack: The attacker tries every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained. On average, half of all possible keys must be tried to achieve success.

If either type of attack succeeds in deducing the key, the effect is catastrophic: All future and past messages encrypted with that key are compromised.

We first consider cryptanalysis and then discuss brute-force attacks. Table 2-1 summarizes the various types of cryptanalytic attacks, based on the amount of information known to the cryptanalyst. The most difficult problem is presented when all that is available is the ciphertext only. In some cases, not even the encryption algorithm is known, but in general we can assume that the opponent does know the algorithm used for encryption. One possible attack under these circumstances is the brute-force approach of trying all possible keys. If the key space is very large, this becomes impractical. Thus, the opponent must rely on an analysis of the ciphertext itself, generally applying various statistical tests to it. To use this approach, the opponent must have some general idea of the type of plaintext that is concealed, such as English or French text, an EXE file, a Java source listing, an accounting file, and so on.

Table 2-1: Types of Attacks on Encrypted Messages

Type of Attack	Known of Cryptanalyst
Ciphertext only	<ul style="list-style-type: none"> • Encryption algorithm. • Ciphertext to be decoded.
Known plaintext	<ul style="list-style-type: none"> • Encryption algorithm. • Ciphertext to be decoded. • One or more plaintext-ciphertext pairs formed with the secret key.
Chosen plaintext	<ul style="list-style-type: none"> • Encryption algorithm. • Ciphertext to be decoded.

	<ul style="list-style-type: none"> • Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with secret key.
Chosen ciphertext	<ul style="list-style-type: none"> • Encryption algorithm • Ciphertext to be decoded • Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plain generated with the secret key
Chosen text	<ul style="list-style-type: none"> • Encryption algorithm • Ciphertext to be decoded • Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key. • Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key

The ciphertext-only attack is the easiest to defend against because the opponent has the least amount of information to work with. Table 2-1 lists two other types of attack: chosen ciphertext and chosen text. These are less commonly employed as cryptanalytic techniques but are nevertheless possible avenues of attack. Only relatively weak algorithms fail to withstand a ciphertext-only attack. Generally, an encryption algorithm is designed to withstand a known-plaintext attack.

Two more definitions are worthy of note. An encryption scheme is unconditionally secure if the ciphertext generated by the scheme does not contain enough information to determine uniquely the corresponding plaintext, no matter how much ciphertext is available. That is, no matter how much time an opponent has, it is impossible for him or her to decrypt the ciphertext, simply because the required information is not there. With the exception of a scheme known as the one-time pad, there is no encryption algorithm that is unconditionally secure. Therefore, all that the users of an encryption algorithm can strive for is an algorithm that meets one or both of the following criteria:

- The cost of breaking the cipher exceeds the value of the encrypted information.
- The time required to break the cipher exceeds the useful lifetime of the information.

An encryption scheme is said to be computationally secure if either of the foregoing two criteria are met. The rub is that it is very difficult to estimate the amount of effort required to cryptanalyze ciphertext successfully.

A brute-force attack involves trying every possible key until an intelligible translation of the ciphertext into plaintext is obtained. On average, half of all possible keys must be tried to achieve success. Table 2-2 shows how much time is involved for various key spaces. Results are shown for four binary key sizes.

Table 2-2: Average Time Required for Exhaustive Key Search

Key size (bits)	Number of alternative keys	Time required at 1 decryption/ms	Time required at 10^6 decryption/ms
32	$2^{32} = 4.3 \times 10^9$	$2^{31} \text{ ms} = 35.8 \text{ minutes}$	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	$2^{55} \text{ ms} = 1142 \text{ years}$	10.01 hours
128	$2^{128} = 3.4 \times 10^{38}$	$2^{127} \text{ ms} = 5.4 \times 10^{24} \text{ years}$	$5.4 \times 10^{18} \text{ years}$
168	$2^{168} = 3.7 \times 10^{50}$	$2^{167} \text{ ms} = 5.9 \times 10^{36} \text{ years}$	$5.9 \times 10^{30} \text{ years}$
26 characters (permutation)	$26! = 4 \times 10^{26}$	$2 \times 10^{26} \text{ ms} = 6.4 \times 10^{12} \text{ years}$	$6.4 \times 10^6 \text{ years}$

2.6 Conclusion

As this chapter has shown the background concepts of this work, not only does it give you the Basic cryptography theory and cryptography goals, but it also provides more information about symmetric cryptography such as terminology of symmetric cryptography algorithms, symmetric cryptography modes and cryptanalysis. Guiding you along the ways is used to hacking the cryptography key to decrypt the ciphertext that is transmitting between the sides of communication in the network.

The CryptoBI algorithm uses the streaming cipher method, which is symmetric-key cryptography. The CBC mode will used in the implementation of well-known symmetric algorithms, which we are going to use in the performance compression experiment. In the cryptanalysis experiment, we will consider that the attacker know the algorithm plus some knowledge of the algorithm setting and characteristics.

Chapter 3. Related works

In this chapter, we will review and study the previous related work and researches concerned with key generation based on image. We will study how they generate the cryptography key using an image.

3.1 Cryptography key generation based image

In this section, we will review and study the methods used to generate the cryptographic key from image. The generated key used by well-known symmetric cryptography such as DES to encrypt the data.

3.1.1 Key Generation Method Using Image Features

The author in [29], proposed a novel algorithm for key generation using image features. This study uses the Gray Level Co-occurrence matrix of an image to extract the Gray Level Co-occurrence properties of the image. A 56-bit sub-key is generated from the extracted Gray Level Co-occurrence properties. Then the key for encryption and decryption is generated using the sub-key generated from the image. The proposed algorithm is split up into five phases that is given in Table 3-1.

Table 3-1: Algorithm Phases

Phases	Process
Phase 1	Database creation
Phase 2	Key generation
Phase 3	Encryption
Phase 4	Secure communication
Phase 5	Decryption

There are 22 features in an image. They are extracted from the Gray level co-occurrence matrix. They are given as input to the principal component analysis for selecting the most dominant (top 4) features among them. Using GLCM concept, features are computed for an image. Based on this features and steps in phase 2 generates key for the secure communication. In the below table 3-2 the dominant features of the images and their values are specified. Both the sender and receiver should use same images and an image with both users should be of same name.

The experimental results for this study conclude that this method is more secured than traditional cryptographic processes and steganography. The process has an advantage of key generation based on sessions. This process is more flexible that any image can be used for key generation as the key generation is directly based on the image properties.

Table 3-2: Feature selection from principal component analysis

Image	Contrast	Correlation	Energy	Homogeneity
jpg	0.0845	0.9821	0.2263	0.9648
jpg	0.0756	0.9657	0.2565	0.9660
jpg	0.0960	0.9724	0.2262	0.9553
pg	0.2412	0.9497	0.1175	0.9013
jpg	0.1634	0.9616	0.1519	0.9343

3.1.2 Key Generation Based on Face Features

The author in [30], proposed a biometric cryptosystem based on face biometrics. At encryption stage, a 128-dimensional principal component analysis (PCA) feature vector is firstly extracted from the face image. And a 128 bit binary vector is obtained by thresholding. Then select the distinguishable bits to form biokey and the optimal bit order number is saved in a look-up table.

Furthermore, an error-correct-code (ECC) is generated using Reed-Solomon algorithm. The message is encrypted using symmetric DES with bio-key. In decryption phase, a 128 dimensional PCA features vector extracted from the query face image. Then a bio-key is generated using the look-up table generated at encryption stage. The final key is obtained using both bio-key and Error correct code (ECC). Finally, the symmetric DES decryption algorithm implemented to obtain message using final key.

In a symmetric cryptographic system, the decryption key should be exactly identical with the encryption key. However, the features extracted from the different face images of a subject are not definitely same. It usually is caused by noise of camera, pose or illumination variation and so on. To solve these problems, this study proposes a novel key generation scheme that based on error-correct-code (ECC). The framework of proposed scheme is shown in Figure 3-1.

In the Encryption phase, PCA features of a face image are extracted. Using these features, they generate a binary string by binarization. And then the stable bio-key is generated by statistical optimization and the optimal bits are saved in a lookup table. The bio-key is used to encrypt message. And the biokey is encoded using Reed – Solomon code. The optimal bit order, ECC code and encrypted message are transmitted to receiver.

In the decryption phase, the PCA features of query face image are extracted. And corresponding bio-key is generated by binarization and optimal bit order. And the bio-key is corrected using the Reed – Solomon error correction code (ECC). Then the corrected key is used for decryption. If the faces at the encryption side and decryption side are same, they could generate the correct key and the Face correct message could be obtained.

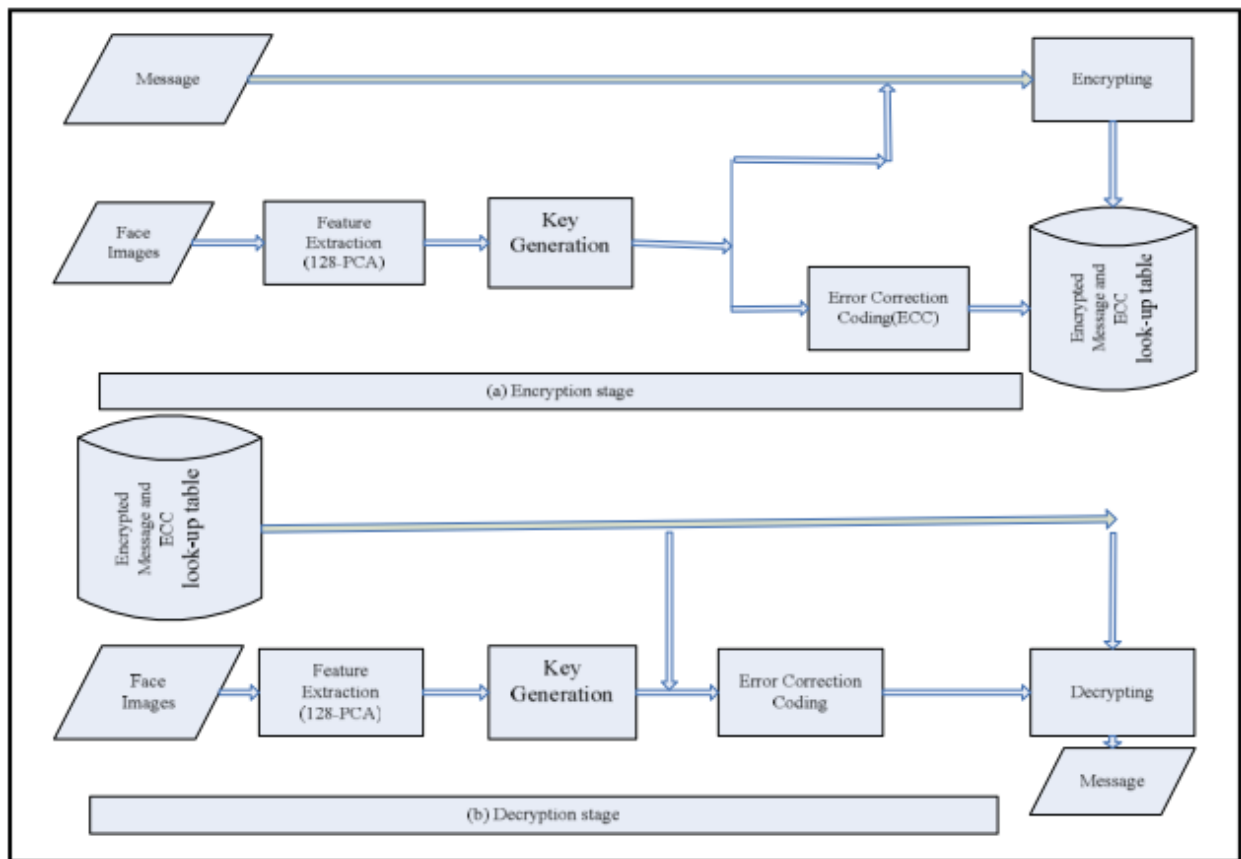


Figure 3-1: The framework of proposed scheme [30]

Symmetric DES algorithm is used for message encryption and decryption with generated bio-key. Experimental results show that their scheme can work effectively under the condition that allowed certain tolerate bits.

3.1.3 Randomized Cryptographic Key Generation Using Image

The authors in [31], proposed system focuses on generating key based on images. The generated key need not have to be stored. It can be generated anywhere using the image and the session. To generate the original message from the cipher text one has to know the session on which the encryption is done, image considered for encryption, RGB values taken and processed for key generation. The key generation is based on the image stored in the database. Consider the pixel value of an image and extract one channel (RGB) at a time. It can be either a red channel or green channel or blue channel. The values of the components should be stored in an array. The array size should be $p \times q$ where p and q are the resolution of an image.

The key is generated based on the following steps. They considered all the channels for key generation.

1. Red channel: Consider only the diagonal values of the array whose indexes (N) are $N \% 8 = 0$. RMS(Root Mean Square) value for those diagonal values are calculated. The generated RMS value is considered as a part of the key and it is stored in a variable.

2. Green channel: Sum up the all the pixel values in a zigzag manner starting from 0×0 to $(n-1) \times (n-1)$, $n = \text{size of the image}$ and store the summed up value in a variable.

3. Blue channel: Repeat the steps of the red channel. Now, the value got using the red channel is appended with the value got using the green channel and the blue channel value is also appended to generate the key.

The sender encrypts the confidential message using the RC5 algorithm, once the encryption is done it is sent to the receiver along with the session log. The session log contains the time in which the encryption is done.

In reference to the session log the receiver will consider the image in the image database to generate key for decryption. The generated key along with the encrypted message is sent for decryption (RC5 Algorithm) and the original message is extracted.

3.2 Avoids Key Exchange

In [32], new type of Symmetric encryption algorithms is introduced. This new symmetrical encryption algorithm is proposed to prevent the outside attacks. The new algorithm avoids key exchange between users and reduces the time taken for the encryption and decryption. It operates at high data rate in comparison with The Data Encryption Standard (DES), Triple DES (TDES), Advanced Encryption Standard (AES-256), and RC6 algorithms.

The new algorithm avoids fixed-key exchange between sender and receiver with each authentication process in WLAN.

The new algorithm uses a key size of 512-bits to encrypt a plaintext of 512-bits during the 16-rounds. In this Algorithm, a series of transformations have been used depending on S-BOX, different shift processes, XOR-Gate, and AND-Gate. The S-Box is used to map the input code to another code at the output.

The key generation generates 16-keys during 16-rounds. One key of them is used in one round of the encryption or decryption process. In the first time, the initial key is divided into four parts a, b, c, and d, 128-bits each. In each round of the key generation, there are series of the transformation used to generate the round-key.

The Encryption process in the new algorithm is used to encrypt the plaintext of size of 512-bits by a key of size of 512-bits in each round during 16-rounds. Series of transformations are applied on the plaintext in each round to obtain a ciphertext finally.

The key-updating is a new approach to increase the difficulty to discover the key. The text and speech signals are used to prove the success of the proposed algorithm. The proposed algorithm has higher data rate than DES, TDES, and AES256 algorithms. The voice encryption and decryption is applied using wired and wireless connection.

3.3 Conclusion

We have explored the previous related work and researches, which are concerned with key generation based on image. The papers [29] , [30] and [31] proposed method to generate the cryptography key from an image only, where they used the generated key by one of well-known

symmetric cryptography algorithm such as DES or RC5 to encrypt the data. The authors in [32] proposed complete cryptography system; Key generation algorithm, encryption algorithm and decryption algorithm. In addition, their method supports new approach to increase the difficulty to discover the key that is key-updating approach.

We can sum up the previous related work problems in two points: the first, which is that the generated key is short, while the second, which is that the proposed algorithm not support key-updating approach. The CryptoBI version1 and version2 different from the others proposed algorithm that the key dos not have a fix length, the length of Key generated by CryptoBI algorithm varies according to the size of the message, as is vary every session according to the session type, and the key length can be more than 512 bits according to data size.

Chapter 4. Design and System Model

This chapter discusses the background concepts of the “CryptoBI” algorithm [9]; the structural model that will be implemented, the general steps of the algorithm, the key generation algorithm, the encryption algorithm, the decryption algorithm, and the enhancement on the CryptoBI algorithm.

4.1 Introduction

Our scenario is that two terminals need to exchange data through an unsecured channel. They need to ensure the confidentiality and authenticity of the messages transmitted between them, so they should use cryptography techniques to prohibit a third terminal from intercepting the exchanged data. They need to agree on the cryptography key before they can communicate with each other. Since the channel is unsecure to data exchange, it is unsecure to key exchange which should be secure and periodically changed to assure high security. Instead of exchanging session keys, we need a simple and secure method to generate the key in both sides.

A Novel Cryptography Method Based on Image for Key Generation “CryptoBI” algorithm is a new type of symmetric encryption algorithm that prevents the attacks and avoids key exchange between the two sides of communication by creating the key in home on the sender/receiver machine [9].

In this work, we have implemented the CryptoBI algorithm according to a specific scenario that will be detailed in this chapter in order to measure the strength against the cryptanalysis attack and the efficiency of the algorithm against well-known symmetric cryptographic algorithms which are AES, Rijndael, DES, 3DES, RC2, RC6 and Blowfish. As a result of this measurement, we found that the algorithm was weak against the cryptanalysis attack. We found that the algorithm was more efficient than most of the well-known algorithms, but less efficient than the Blowfish algorithm which was at the forefront of other algorithms in some published researches such as [18], [33], [34], [35] and [36]. Based on the foregoing, we decided to update this algorithm (CryptoBI) to increase the strength and efficiency to be in the forefront, so during this chapter, we will be explaining the previous algorithm which will be called "CryptoBI version1". We will also explain the new enhanced algorithm which is the modified form of the previous algorithm. The new enhanced algorithm will be called “CryptoBI version2”.

4.2 The System Model for CryptoBI version1

The structure model of CryptoBI version1 implementation approach is depicted in Figure 4-1 [9]. The block diagram show that the host A and host B are two hosts in the local area network, and they are communicating with each other by exchanging text messages and files. The model provides the two hosts a secure channel so they can exchange the data without doubting about the confidentiality and authenticity of the data transmitted over the channel.

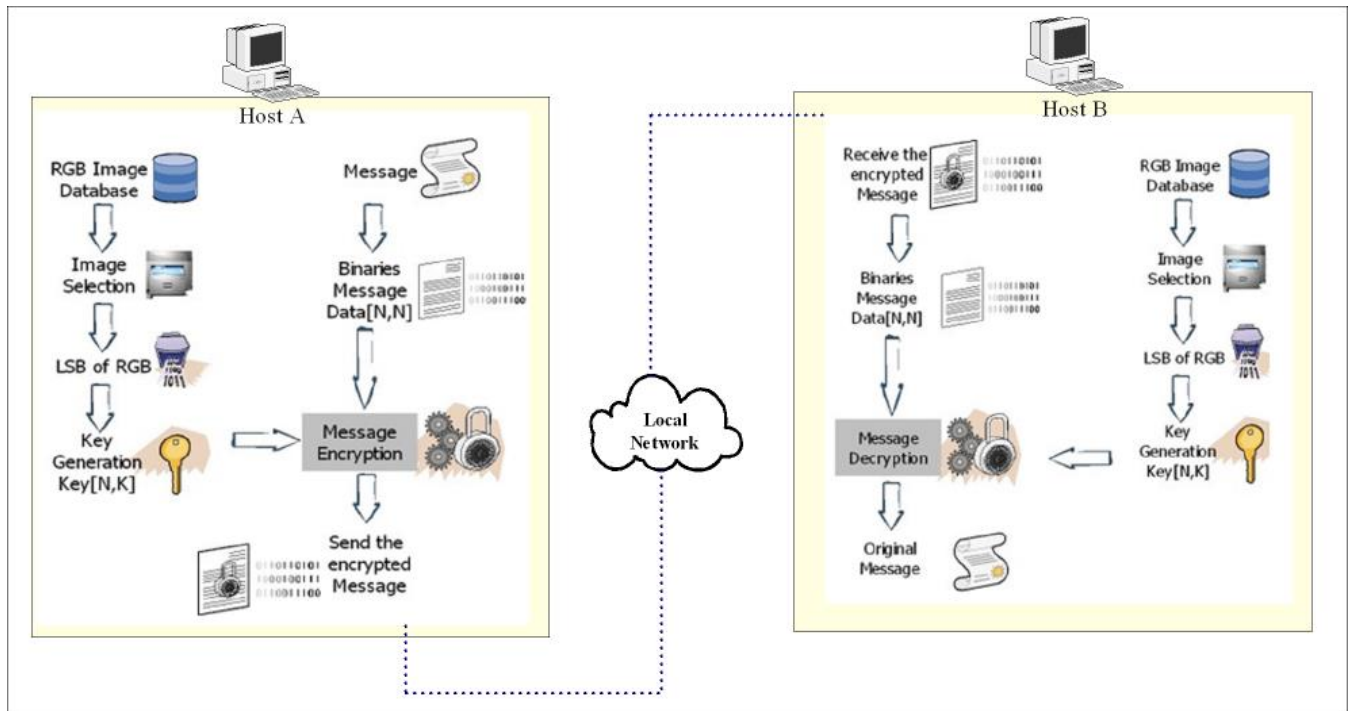


Figure 4-1: the sender and the receiver block diagram

This model provides a method to generate cryptography keys in home instead of exchanging them over the communication channel. By using this model the protection of the confidentiality of cryptography keys is guaranteed by avoiding the exchange of the cryptography keys between the two sides of communication. In short, the following steps describe what happens between the two hosts when they exchange data. For example when host A sends a text message to host B, the following operations are performed:

Host A (sender):

1. Host A binarize the text message
2. According to the session type, an image will be selected from images database.

3. Generate the key from one of the selected image channels R, G or B pixels according to the setting between the two hosts.
4. Encrypt the binary message bits using the generated key in step 3.
5. Transmit the encrypted message over the network to the host B

Host B (Receiver):

6. Host B binarize the received encrypted message
7. According to the session type an image will be selected from the images database which will be the same image selected on the Host A.
8. Generate the key from one of the selected image channels R, G or B pixels according to the setting.
9. Decrypt the binary message bits using the generated key in step 8.
10. Get the original text message

The model works under some limitations and assumption such as:

4. The users can be communicant with each other through Ethernet “LAN” or private business network with small number of users.
5. The user can use the system to transmit data such as text messages or files.
6. Synchronization between the two sides of communication, From Figure 4-1, we can conclude that both the sender and receiver, before starting to communicate with each other, should agree on many settings, which are:-
 - a) Database of RGB images
 - b) Image channel R, G or B that will be used to generate the key.
 - c) K value for the key array.
 - d) The session type daily, hourly or custom.

All the above-mentioned settings will be described in the next sections. These settings are very important to both sides, and must be the same on both sides to encrypt and decrypt the exchanged data successfully.

4.2.1 Database Creation for CryptoBI v1:

In this phase, creating database of color RGB images will be used for generating the key for the encryption/decryption process. The Key generation is based on sessions and according to this, our model will use one image per session, In case of using hourly session, the database should contain 24 images and 7 images for daily session. Both the sender and the receiver should use

the same database of images (24 or 7 images), as the names of the images should be the same on both sides. Custom session type can be used other than hourly and daily. Regardless of the session type that will be used, the key generation algorithm selects an image from the database according to the time of session then it will be used to generate the encryption key.

4.2.2 Binarize Message for CryptoBI v1:

It converts the values of the message data, which are in decimal to binary values. In this step, the binary values of the message are stored in a multidimensional array $Data[N,N]$ where the number of message bits will be less than or equal to $N*N$.

Assume the text message content is “1Testing”, the message contains 8 characters and each character is represented by 8 bits, the best value of N to store the binary bits of the message where $N*N \leq 64$ is 8, so the $N=8$. Figure 4-2 describe the array $Data[8,8]$ content after binarize the message.

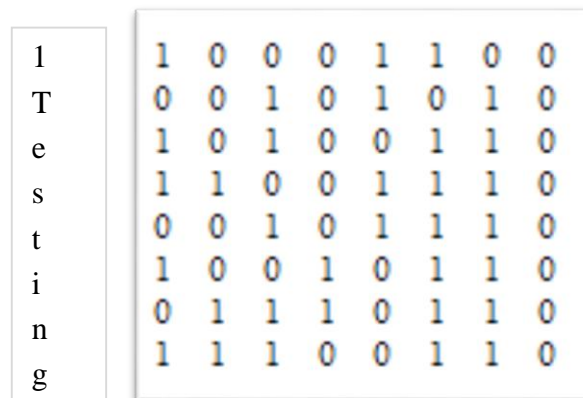


Figure 4-2: Binary Array of the Message “1Testing”.

The model can be used to transfer texts and files between two sides of communication. If the two sides need to transfer a file between each other, the same operation done on the text message is done on the file after reading the contents of the file as one dimensional array of bytes. The one dimensional array of bytes convert to multidimensional array $Data[N,N]$ containing the binary values of the file content.

4.2.3 LSB of RGB for CryptoBI v1:

Color digital images are made of pixels, and pixels are made of combinations of primary colors. A channel is made of just one of these primary colors, a grayscale image has just one channel,

but the RGB image has three channels: red, green, and blue, If the RGB image is 24-bit, each channel has 8 bits, for red, green, and blue in other words, the image is composed of three images (one for each channel), where each image can store discrete pixels with conventional brightness intensities between 0 and 255 [37]. In this thesis, we used one channel of RGB color image which are Red, Blue or Green, so we have to choose one of these channels where each pixel should donate only one bit, called the Least Significant Bit (LSB). This model step deals with the image channel as two dimensional array of binary values contains the LSB of the channel pixels as described in Figure 4-3.

4.2.4 Key Generation for CryptoBI v1:

The encryption/decryption process depends on the key to encrypt/decrypt the data, so this step generates the key automatically from the selected image from the database of images according to session type that the sender and receiver agree on. If the session type is hourly, the images database should be contain 24 images, and the images should be named serially from 1.jpg to 24.jpg. Suppose that the sender sends a message at 1 :00 am, according to that, the algorithm will select the image 1.jpg to generate the key.

The generated key is represented as binary array Key [N, K] where k value is even and between 2 and N, where N is the width of Data array that we mentioned in the previous section. For example, we need to generate the key to encrypt the message we mentioned above “1Testing” where the value of N was 8, and if we set the value of K to 6 so, we need array 8*6 size to store the key bits (Key [8, 6]).

The key generation algorithm use one of the selected image channels which are R, G or B according to what both sides have agreed on. The algorithm generates the key by scanning the pixels of the selected image channel to find two neighbor values where the absolute difference between them is one. Then the algorithm stores the LSB of neighbor values in the array Key [N, K]. The algorithm continues scanning and storing until the array Key [N, K] is full. The key generation process is described in the Figure 4-3.

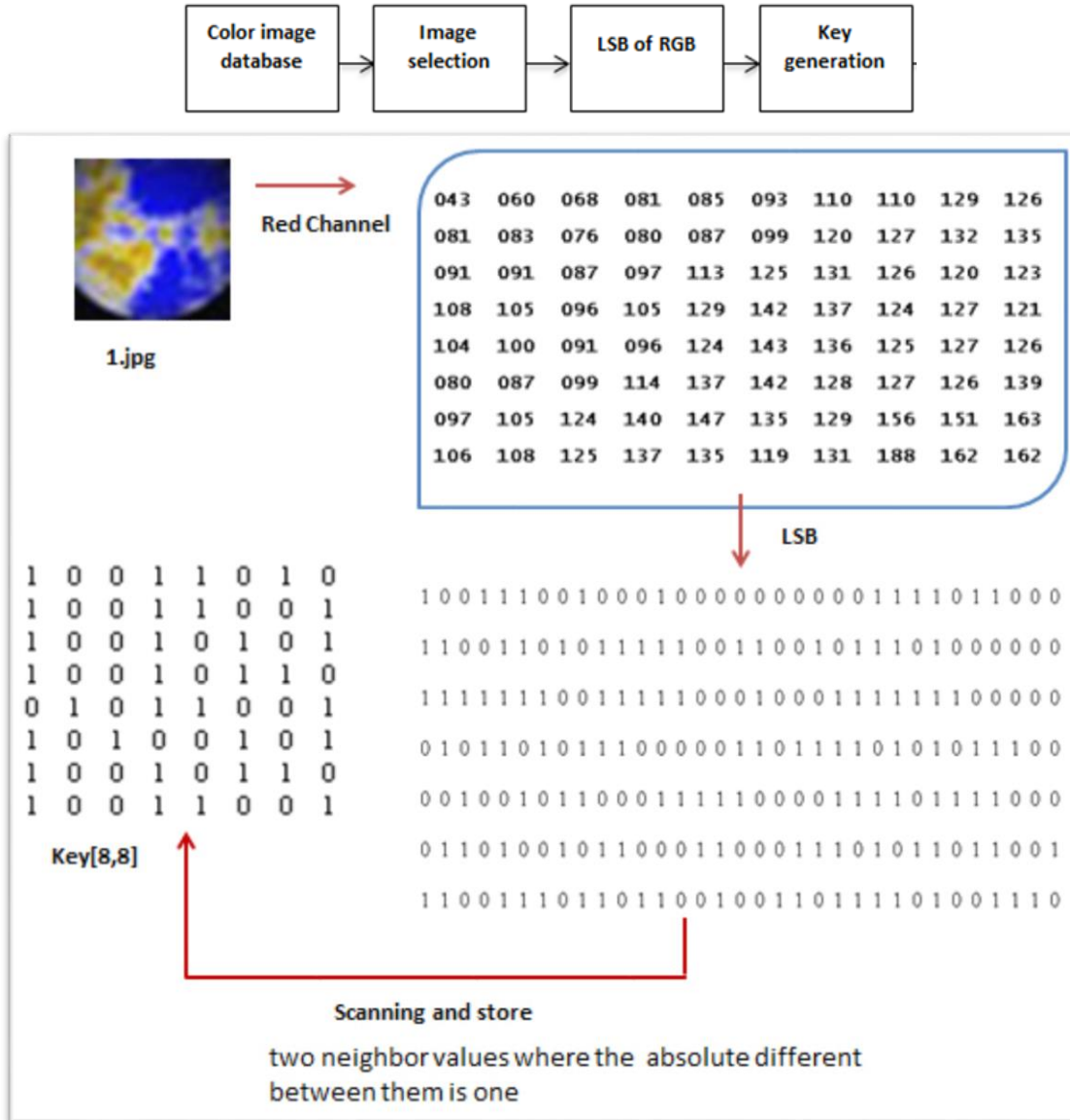


Figure 4-3: Key generation phase and the key array

The pseudo code for the Key generation algorithm is as follows:-

1. Get the value of K variable and get the value of N from binarize message step.
2. Select color image from the database according to session type and the current date or time.
3. Retrieve the image and create array Img [x, y] as LSB of the pixels for one channel (red, green or blue) according to the settings, where the image has x*y dimension.
4. Create array Key [N, k] to represent the encryption key.

```

5. Loop i=0 to N-1 // for each row in Key
6.     Loop c=0 to k/2-1 step 2 //fill array by two each step
7.         Scan the LSB Img[x,y] to find two neighbor values where
           the absolute difference between them is one.
8.         Key[i,c] = First LSB value.
9.         Key[i,c+1] = Second LSB value.
10.        End loop
11.        End loop

```

The scanning process for the key starts from step 5 of the algorithm, where the loop starts scan from the first pixel in the selected image. Every iteration of the inner loop fills the Key array by two bits.

4.2.5 Encryption/ Decryption Algorithm for CryptoBI v1:

The encryption process is to encrypt the data of the sender by using the key generated from the previous process which is key generation. We will complete the previous example as we mentioned in the previous sections where the message to encrypt was “1Testing” and the current session time is 1:00 AM. After we have generated the key and binarize the message described in Figure 4-2 and Figure 4-3. The implementation of the encryption phase is described in Figure 4-4. We need to create a new array to store the encrypted data EncData[8,8] as the same size of Data array. The first row of Data array is processed with the first two columns of Key array as we described in Figure 4-4 where the bits will be substituted to 0 or 1 according to key bits. If the Data array bit equals key bit in the first column, it is substitute to 0 else it is substitute to 1. For example Data[0,0] is compared with Key[0,0] and Key[0,1]. We found that the Data[0,0]=Key[0,0] then EncData[0,0]=0, where 0 represent the column index of the two columns of Key array. The index might be 0 or 1, next bit Data[0,1] is compared with Key[1,0] and Key[1,1], and we found that Data[0,1]=Key[1,1] then EncData[0,1]=1. This compression is repeated on the next bits of the first row of Data array. After that, the operation is repeated on the next Data row and next two Key columns and so on to all bits in Data array where the Key array has a specific number of columns, so when we reach the last two columns we return back to the first two columns. The result of this phase was a binary array EncData[8,8] as shown in Figure 4-4. Then the EncData array is transformed into text format. We have got the encrypted form of the message “1Testing” which is “Ptö:›Iý.”.

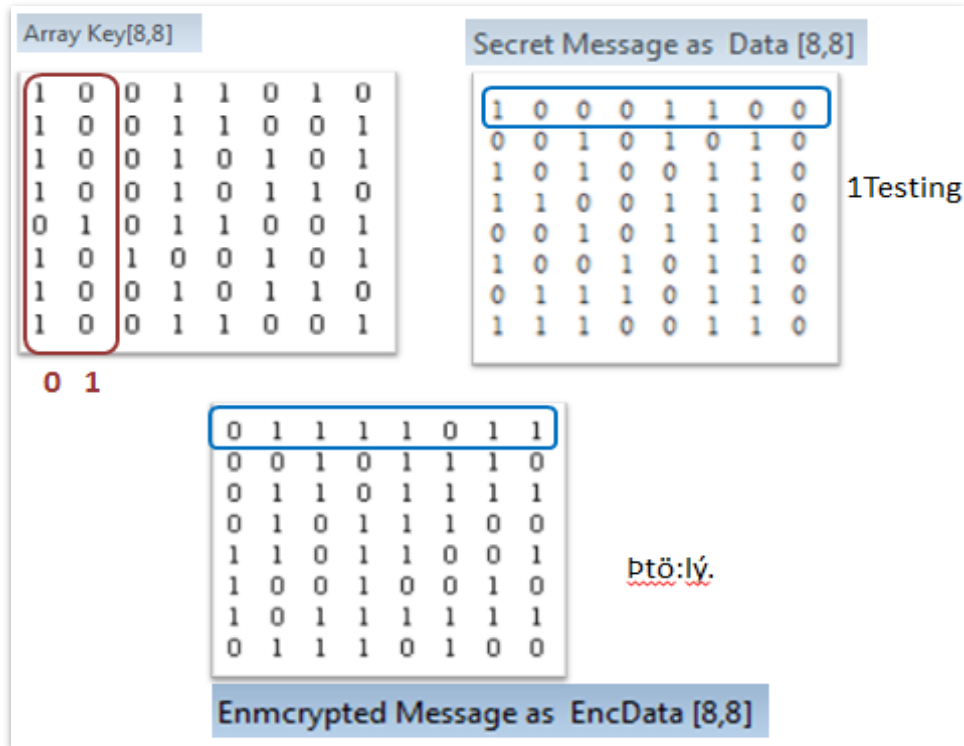


Figure 4-4: the encryption process.

Encryption Algorithm pseudo code

The pseudo code for the encryption algorithm and the algorithm steps are as follows:

1. Get the value of K variable.
2. Read a message to encrypt
3. Binarize the message and store the binary data in array Data[N,N] where message bits are $\leq N*N$.
4. Generate the key using key generation algorithm and create Key [N, k].
5. Create array EncData [N, N] to store the encrypted data
6. c=0
7. Loop i=0 to N-1 // for each row in Data
// the elements in the Data array row i compare with two columns of Key array
// start from c
8. Loop j=0 to N-1 // for each bit in the row i
9. If Data[i,j] = Key[j,c]
10. EncData[i,j]=0 // the value in the first column

```

11.      Else
12.          EncData[i,j]=1    // the value in the second column
13.      End loop
14.      c=c+2                // go to next two columns of Key array.
                            // return to the first two columns of Key array
15.      if c >= K then c=0
16.      End loop
17.      Convert the EncData from binary data to text.

```

The steps from 1 to 6 are preparation and initialization steps. The encryption process starts from the step 7, where each bit in the row i in the Data array will compare with two columns of Key array start from c , the bits will be substituted by the two columns indexes 0 or 1 and store in EncData array. When the c is exceed the Key array column boundary, will set to zero to start from the beginning again.

The decryption process to decrypt the received encrypted data on the receiver side after the key generation process that should produce the same key generated on the sender side. The decryption process does the opposite of the encryption process.

Decryption algorithm pseudo code

The pseudo code for the decryption algorithm and the algorithm steps are as follows:

```

1. Get the value of K variable.
2. Binarize the encrypted message and store the binary data in array
   EncData[N,N]
3. Generate the key using key generation algorithm and create Key [N, k].
4. Create Data[N, N] to store the original data
5. c=0
6.  Loop i=0 to N-1          // for each row in Data
   // the elements in the Data array row i compare with two
   // columns of Key array start from c
7.  Loop j=0 to N-1        // for each bit in the row i
8.  If Encdata[i,j]=0 then
9.      Data[i,j]= Key[j,c]    // get value from first column
10. Else

```

```

11.      Data[i,j]= Key[j,c+1] // get value from second column
12.      End loop
13.      c=c+2                // go to next two columns of Key array.
14.                        // return to the first two columns of Key array
15.      if c >= k then c=0
16.      End loop
17.      Convert the Data from binary data to text.

```

The steps from 1 to 5 are preparation and initialization steps. The decryption process starts from the step 6, where each bit in the row i in the EncData array will compare with two columns of Key array start from c . The bits of row i will be substituted by the two columns (Key[c] and Key[$c+1$]) bits values according to row i bits and store in Data array. When the c is exceed the Key array columns boundary, will set to zero to start from the beginning of Key array again.

4.3 The System Model for CryptoBI v2

The structure model of CryptoBI version2 implementation approach is depicted in Figure 4-5.

From Figure 4-5, we can conclude that the structure model of CryptoBI version2 is the same as the structure model of CryptoBI version1 except:

- The binarization message step is removed from CryptoBI version2 because the CryptoBI version1 performs the encryption/decryption operation bit by bit on the plaintext to obtain ciphertext, but the CryptoBI version2 did that using byte operation. Removing this step from the system model increased the performance of the algorithm because the byte operation faster than the bit operation.
- A new two bytes integer variable Kv is added to the synchronization setting that is a feedback value for encryption/decryption algorithm.

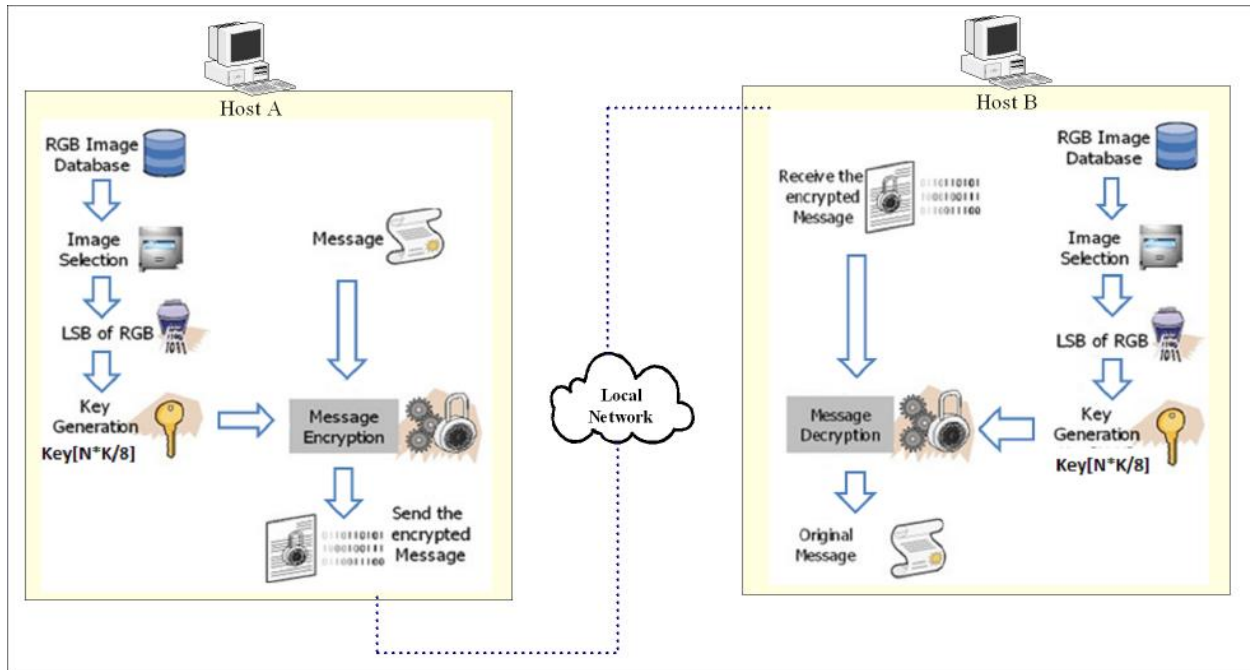


Figure 4-5: the sender and the receiver block diagram for CryptoBI v2.

- A different key generation algorithm. We will sum up the difference in two points: First, the key in CryptoBI version1 has been generated as two-dimensional array which contains binary values, but the key is generated in CryptoBI version2 as one-dimensional array of bytes. Second, the image scanning starting point for key generation in the CryptoBI version2 changes according to the algorithm settings. In the CryptoBI version1, the key generation starts from the first pixel of the image, so this facilitating to the attacker to get the cipher key when he knows the images database.
- Different encryption/decryption algorithm, as we mentioned in the previous point that the CryptoBI version2 encryption/decryption of the plaintext is done using byte operation not bit operation as the CryptoBI version1.

4.3.1 Database Creation for CryptoBI v2:

The operation of this step is similar to CryptoBI version1. In this phase, creating database of color RGB images will be used for generating the key for the encryption/decryption process. The Key generation is based on sessions and according to this, our model will use one image per session, In case of using hourly session, the database should contain 24 images and 7 images for daily session. Both the sender and the receiver should use the same database of images (24 or 7 images), as the names of the images should be the same on both sides. Custom session type can

be used other than hourly and daily. Regardless of the session type that will be used, the key generation algorithm selects an image from the database according to the time of session then it will be used to generate the encryption key.

4.3.2 LSB of RGB for CryptoBI v2:

The operation of this step is similar to CryptoBI version1. Color digital images are made of pixels, and pixels are made of combinations of primary colors. A channel is made of just one of these primary colors, a grayscale image has just one channel, but the RGB image has three channels: red, green, and blue, If the RGB image is 24-bit, each channel has 8 bits, for red, green, and blue in other words, the image is composed of three images (one for each channel), where each image can store discrete pixels with conventional brightness intensities between 0 and 255 [37]. In this work, we used one channel of RGB color image which are Red, Blue or Green, so we have to choose one of these channels where each pixel should donate only one bit, called the Least Significant Bit (LSB). This model step deals with the image channel as two dimensional array of binary values contains the LSB of the channel pixels as described in Figure 4-3.

4.3.3 Key Generation for CryptoBI v2:

The scanning for the key in the previous algorithm CryptoBI version1 always starts from the beginning of the image, and this is probably considered as a point of weakness for the algorithm, because the key can be generated in a short time when the attacker have images database. In this case, all the attacker has to do is guessing the other synchronization setting variables which they are the K variable and the image channel. To solve this problem, we suggested changing the scanning starting point for the key generation algorithm by the time of the session for example. The value of a variable is used as a key for this purpose. In other words, we can use the synchronization setting between both sides of communication to determine the scanning starting point, where the starting point is the first pixel of image that from it the key generation algorithm starts scanning for the key.

The key idea is to divide the image into four segments and apply a different starting point of the scanning process in one of those segments. The proposed approach considers the choice of one of the segments then making a jump from the selected segment corner to certain pixel as shown in Figure 4-6.

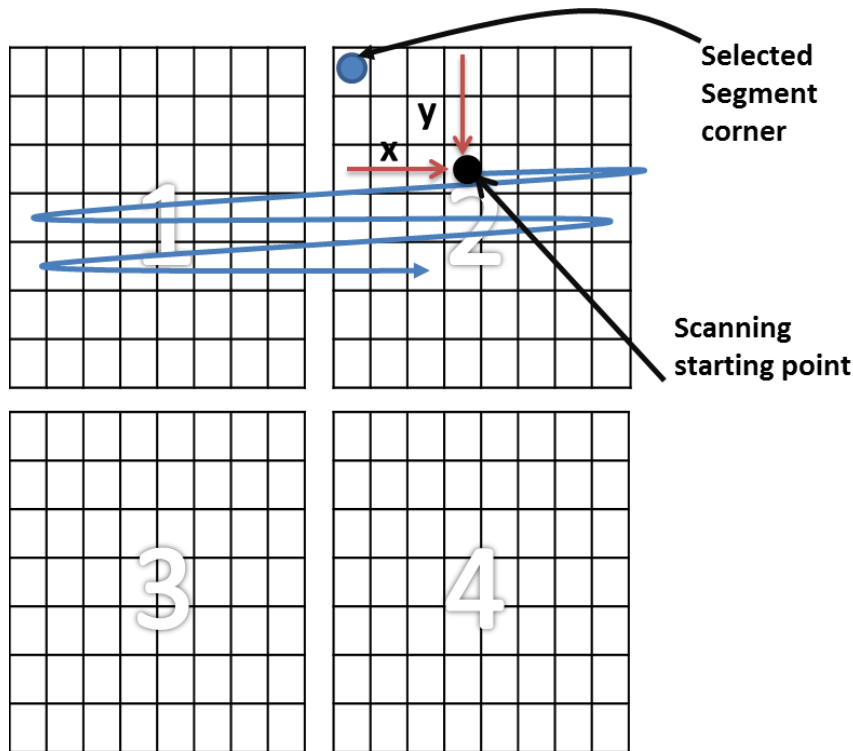


Figure 4-6: the image scanning starting point for key generation algorithm.

We used the session time to determine the segment from the following equation:

$$\text{Segment} = (\text{Session time} \% 4) + 1 \text{ ----- } 4.1$$

The equation 4.1 result is a number between 1 and 4 that represents the image segment to start the scanning process. After the segment is determined, we need to jump to the certain pixel(x, y) from left corner of this segment, and the location of that pixel can be calculated from the following two equations:

$$X = \text{decimal value of image pixel channel at position [0, the value of the first byte of Kv]} \text{ ----- } 4.2$$

$$Y = \text{decimal value of image pixel channel at position [0, the value of the second byte of Kv]} \text{ ----- } 4.3$$

Where the Kv value used as feedback in encryption/decryption algorithm and the decimal value of image pixel is the pixel value of a specific image channel R, G or B according to synchronization setting. Figure 4-6 shows that the starting scanning point for the key starts at the pixel(x, y) from the segment 2 at the left top corner. If the value of X or Y is greater than the dimension of the image, the scanning will start after jumping over the absolute of subtract X from Y pixels from the left top corner of the segment.

For example, suppose that the sender sends a message at 1:00 am, the session type is hourly, using red channel of the image and the Kv is 128, according to that, the algorithm will select the image 1.jpg to generate the key, the segment = $1\%4+1=2$, x =decimal value of image[0,128], and y =decimal value of image[0,0]. The key generation algorithm should start scanning for the key from the pixel at Segment2[x, y].

According to the previous notes, the CryptoBI version2 key generation algorithm different from the CryptoBI version1 in terms of two points. First, the key search starting point changes every session. This makes difficult to the cryptanalysis attacker to the find the key even if the attacker know the images database. Second, the output of the key generation algorithm is a one-dimensional array of bytes.

By default the algorithm generates keys that have length between 128 bits and 512 bits. It can generate keys of more than 512 bits length by changing the algorithm settings.

The pseudo code for the Key generation algorithm is as follows:

```
1. Get the value of K variable and get the value of N
2. Get the value of Kv variable
3. Select color image from the database according to session type and the
   current date or time.
4. Retrieve the image and create array Img [x, y] as LSB of the pixels for
   one channel (red, green or blue) according to the setting, where the
   image has x*y dimension.
5. Create array Key [N* k/8] to represent the encryption key bytes.
   // key length=N*K bytes
6. Create array B [8] to store the bits of one byte.
7. Calculate the scanning starting point
8. Loop i=0 to N*K/8 // for each byte in Key array
9. k=0
10. Loop c=0 to 4
11. Scan the LSB Img[x,y] to find two neighbor values
    where the absolute difference between them is one. The
    scanning start from the calculated scanning starting point in
    step 7.
12. B[k++] = First LSB value
```

```

13.          B[k++] = Second LSB value
14.          End loop
15.    Key[i] = result of converting B array to byte.
16.    End loop

```

After determine the scanning starting in step 7 which is the first pixel of image that from it the key generation algorithm starts scanning for the key. The scanning process for the key starts from step 8 of the algorithm. Every iteration of the inner loop produce a byte as 8 bits array B which is will be converted to a byte and stored in the Key array in step 15.

4.3.4 Encryption/ Decryption Algorithm for CryptoBI v2:

The Encryption process in the new algorithm CryptoBI version2 is used to encrypt the data (text message or file) using a key of a size between 128 bits and 512 bits as default, but the key size can be more than 512 bits. The transformations are shown in Figure 4-7 to obtain a cipher text. The CryptoBI version2 encryption of the plaintext is done by using byte operation instead of bit operation as in CryptoBI version1.

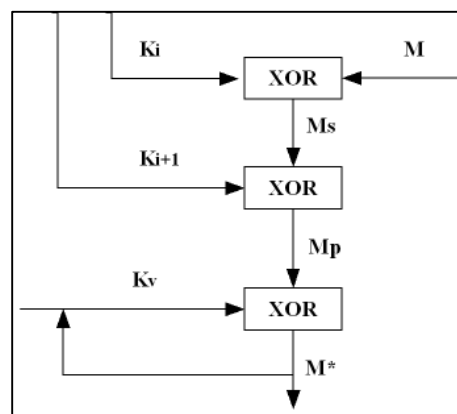


Figure 4-7 : The Encryption Process steps for each data byte.

Figure 4-8 describes the encryption process for CryptoBI v2. The plaintext is the hexadecimal code for the message “CryptoBI Testing” which we need to encrypt. The key is a result of key generation process that has generated 16 byte key from the selected image from the images database according to the session time. The ciphertext is the result of the encryption process as shown in Figure 4-7.

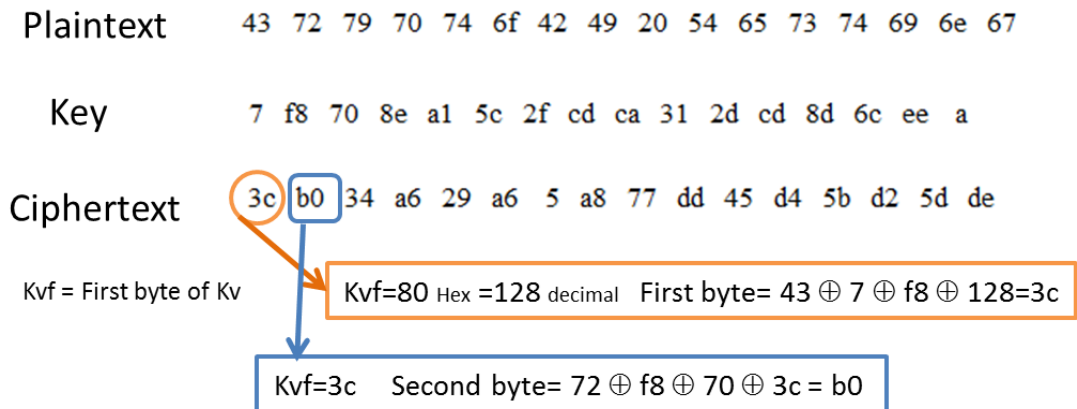


Figure 4-8: CryptoBI v2 encryption process.

The first byte of the ciphertext is calculated as a result of the following equation

$$Ct[i]= Pt[i] \oplus K[i] \oplus K[i+1] \oplus Kvf \text{ ----- } 4.4$$

Where kvf is the first byte of the Kv, which is the feedback value. The default value of the Kv is 128, so the value of Kvf will be 128 in the first time which is equivalent to 80 in hexadecimal. For the next byte Ct[i+1] operation the value of Kvf is set to Ct[i], repeat the previous steps for each byte in the plaintext to obtain ciphertext of all the plaintext.

Encryption Algorithm pseudo code

The pseudo code for the encryption Algorithm and the algorithm steps are as follows:

1. Read and store the data to encrypt in array Data [M] of bytes where M is the data length.
2. Get the value of Kv variable. Kv is the value of the feedback, and its value in the first time is 128.
3. Get the value of K variable, where the default value is 8.
4. Calculate the value of N variable.
 - Set Max_N=64 and Min_N=16. // the key size=K*N bits
 - Set N=n, where M <= n*n.
 - If N > Max_N then set N=Max_N.
 - If N < Min_N then set N=Min_N.
5. Generate the key using key generation algorithm and create Key [N*K/8] of bytes.
6. Create array EncData [M] to store the encrypted data

```

7. c=0
8. Kvf= first byte of Kv
9.   Loop i=0 to M-1    // for each byte in Data array
10.      EncData[i] = Data[i] ⊕ Key[c] ⊕ Key[c+1] ⊕ Kvf.
11.      Kvf = EncData[i]
12.      c = c+2
13.      if c >= length of K then c=0
14.   End loop

```

The steps from 1 to 8 are preparation and initialization steps. The encryption process starts from the step 9, where each byte in the Data array will encrypt by XOR with two bytes of the Key (at c and c+1) and with feedback byte Kvf, as it described in step 10. The new calculated byte store in EncData array which is represents the ciphertext. When the c is exceed the Key array boundary, will set to zero to start from the beginning again.

Decryption algorithm pseudo code

The decryption of the new algorithm is the same as the encryption except of the calculation of the feedback value. The pseudo code for the decryption algorithm and the algorithm steps are as follows:

```

1. Read and store the encrypted data in array EncData[M] of bytes, where M
   is the data length.
2. Get the value of Kv variable, Kv is the value of the feedback, and its
   value in the first time is 128.
3. Get the value of K variable, where the default value is 8.
4. Calculate the value of N variable.
   - Set Max_N=64 and Min_N=16.
   - Set N=n, where M <= n*n.
   - If N > Max_N then set N=Max_N.
   - If N < Min_N then set N=Min_N.
5. Generate the key using key generation algorithm and create Key [N*K/8]
   of bytes.
6. Create array Data [M] to store the encrypted data
7. c=0
8. Kvf= first byte of Kv

```

```

9.      Loop i=0 to M-1           // for each byte in Data array
10.      xdata = EncData[i] ⊕ Key[c] ⊕ Key[c+1] ⊕ Kvf.
11.      Kvf = EncData[i]
12.      Data[i]=xdata
13.      c = c+2
14.      if c >= length of K then c=0
15.      End loop

```

The steps from 1 to 8 are preparation and initialization steps. The decryption is the same as the encryption except of the calculation of the feedback value. The decryption process starts from the step 9, where each byte in the EncData array will decrypt by XOR with two bytes of the Key (at c and c+1) and with feedback byte Kvf, as it described in step 10. The new calculated byte store in Data array which is represents the plaintext. When the c is exceed the Key array boundary, will set to zero to start from the beginning again.

4.4 Conclusion

This chapter discussed the background concepts of the “CryptoBI version1” algorithm [9]; the structural model that will be implemented, the general steps of the algorithm, the key generation algorithm, the encryption algorithm and the decryption algorithm. We also discussed the enhancement on the CryptoBI algorithm version1.

All the updates that have been made on the CryptoBI version1 algorithm will enhance the performance and the strength against the cryptanalysis attack that is what we are going to prove in the next chapter. The performance enhanced by removing the binarize step from the CryptoBI version1 algorithm, the bit operation replaced by byte operation which is faster than bit operation according to the CPU delay time. The strength enhanced by adding some randomization for the key generation scanning starting point that is make the scanning start at different pixel according to the session time and the synchronization setting, this makes difficult to the cryptanalysis attacker to the find the key even if the attacker know the images database.

Chapter 5. Experiments and Results

In this chapter, we will discuss the experimental design for this work and the experiments results

5.1 Experimental Design

This section describes the techniques and simulation choices made to evaluate the performance and strength of the CryptoBI algorithms (version1 and version2). In addition to that, this section will discuss the methodology related parameters such as: system parameters, experiment factors, and experiment initial settings.

5.1.1 Simulation Setup

All the algorithms of the CryptoBI version1 and version2 system models have been implemented using C# programming language. We created a graphical user interface (GUI) application to simulate the performance of CryptoBI version1 and version2 and to apply a compression with the performance of other symmetric cryptography algorithms such as: Blowfish, AES, DES, 3DES, RC2, RC6 and Rijndael. We also created GUI interface to simulate the strength of CryptoBI version1 and version2 agents the cryptanalysis attack that tries to get plaintext from the ciphertext.

This application implementation uses the provided classes in .NET environment to simulate the performance of AES, DES, 3DES, RC2 and Rijndael. Blowfish and RC6 implementation being used here is the one provided by Chilkat Software [Chilkat Encryption .NET Component] under the name of Crypt2 [38]. This implementation is thoroughly tested and is optimized to give the maximum performance for the algorithm. The implementation uses managed wrappers for AES, DES, 3DES, RC2 and Rijndael available in System.Security.Cryptography that wraps unmanaged implementations available in CryptoAPI. Table 5-1 shows the algorithms settings used in this experiment.

As we mentioned before about CryptoBI algorithms, the sender and the receiver have to agree on many setting before they can communicate with each other. Both the sender and receiver should use the same images database, the same color image channel, the same value for Kv variable, and the same value for K variable. In our implementation, we have created a database of color images which contains 24 images. The images are named as 1, 2...24 to specify the day hours because we used hourly session. We used red color image channel, Kv value equal to 128 and K

value equal to 8. As for the length of the key, we have used the value of N between 16 and 64. Accordingly the biggest key that can be obtained is 512 bits whereas the smallest key is 128 bits.

Table 5-1: Algorithms settings

Algorithm	Key Size
AES	256
Blowfish	256
DES	64
TDES	192
RC2	128
RC6	256
Rijndael	256
CryptoBI version1	512
CryptoBI version2	512

The forms of simulation program used in the experiments are shown in figures 5-1, 5-2 and 5-3. The simulation program main form shown below in Figure 5-1 which views the text messages and files that are received from the other host on the Local Area Network, the form accepts one input the cryptography algorithm.

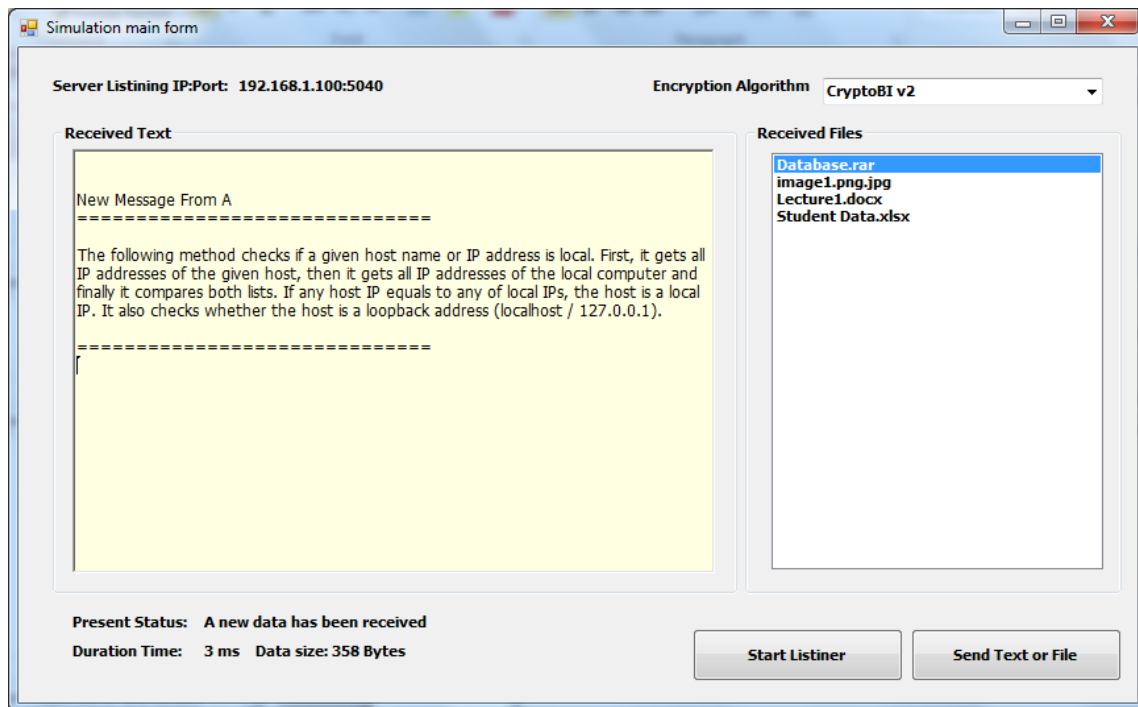


Figure 5-1: The main form of the simulation program, works as receiver.

Figure 5-2 shows the sending form that is used to send the text messages or files to the other host. The form accepts four inputs: the receiver IP address, the receiver port, the cryptography algorithm, and the text or the file to send. This form shows up when we click on the button “Send text or File” on the main form.

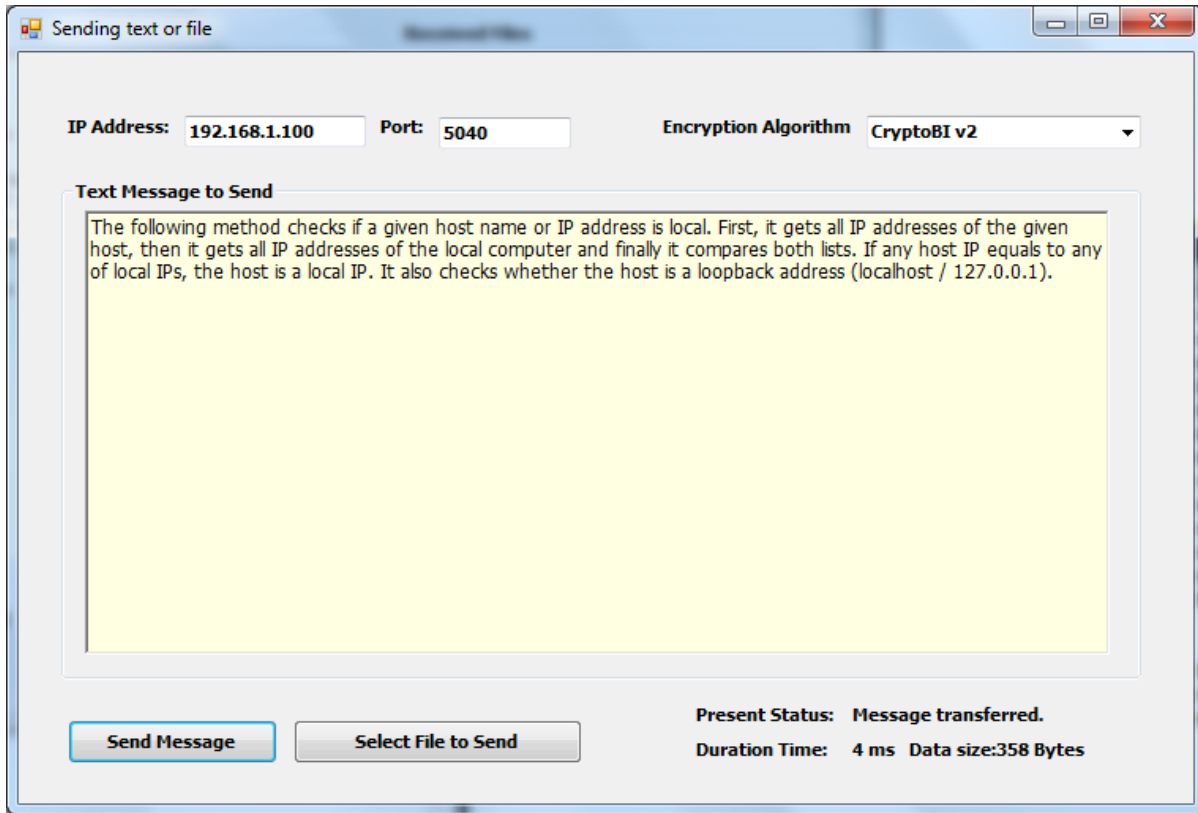


Figure 5-2: The sending form of the simulation program

After a successful execution, the file or text message gets encrypted then sent through the network using the sending form on the sender side “host 2” and on the receiver side “host1” the received data are decrypted then viewed in case a text message was sent or saved in a specific folder as file in case a file was sent.

Finally, Figure 5-3 shows the cryptanalysis form that is used to find the plaintext from the ciphertext. The cryptanalysis scenario assumes that the attacker knows the images database of the CryptoBI algorithm, and he should guess the other synchronization settings of the CryptoBI algorithm. According to that scenario, this form has been created. The form accepts as inputs the cryptography algorithm CryptoBI version1 or version2, an image from the database, the image channel, the K value and the Kv value, which they are the synchronization setting of the CryptoBI algorithm, by clicking on the next try button the form generates and shows the key

from the inputs using the generation algorithm of specified CryptoBI algorithm. Finally, the form shows the result of decrypting the ciphertext using the generated key as the time of this operation in the text boxes designated for that. The button “Next try” used when we want to check the result for every combination of the inputs values manually, but the button “Do all” create a loop to try all the combination for every possible value of the inputs and the save the results in a file.

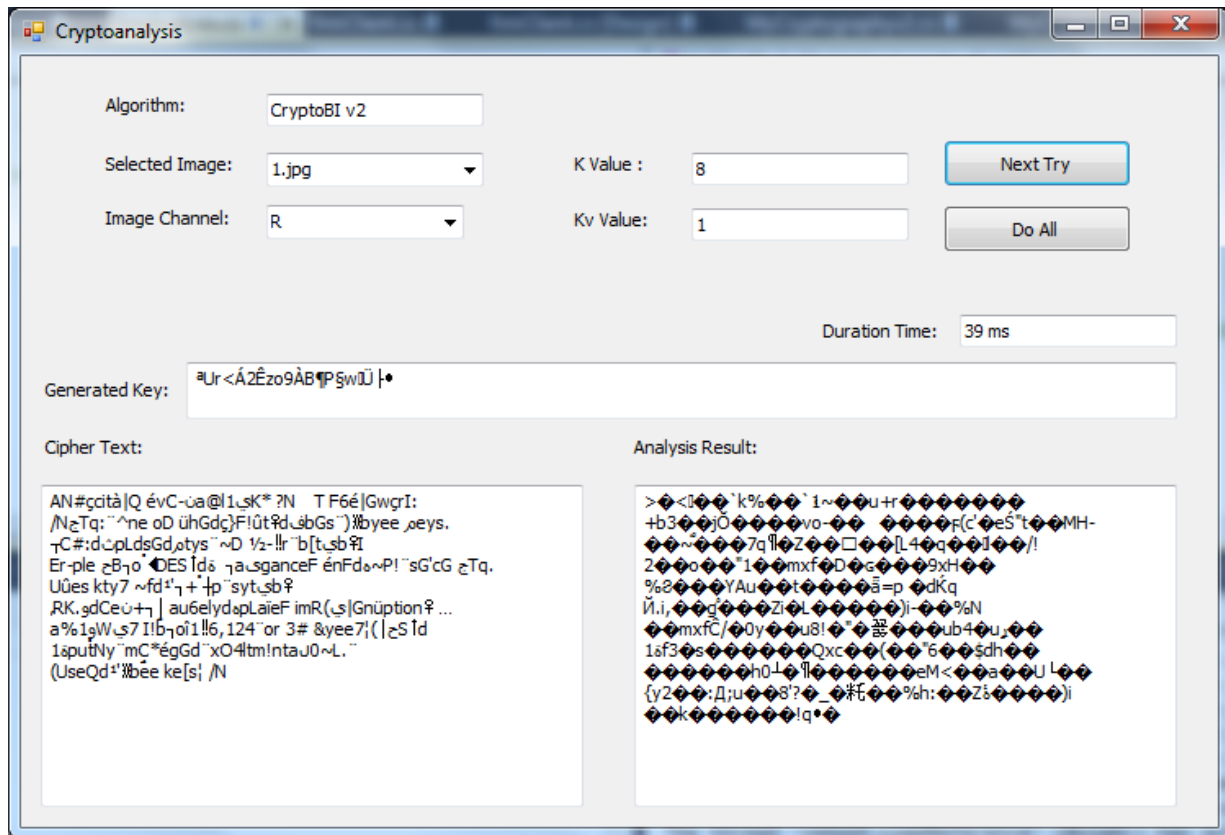


Figure 5-3: The CryptoBI Cryptanalysis from.

5.1.2 System Parameters

The experiments conducted use the following:

i. Software

- The simulation program is compiled using the default settings in .NET 2010 visual studio for C# windows applications.
- Windows 7.

ii. Hardware

- LAPTOP: Intel® Core™i5 2410M CPU N270 @2.30GHz 4GB of RAM.
- Intel® 82579V Gigabit Network Connection.

The experiments will be performed more than once to assure that the results are consistent and valid to compare the different algorithms. The simulation program is run inside the LAN environment; see Figure 5-4.

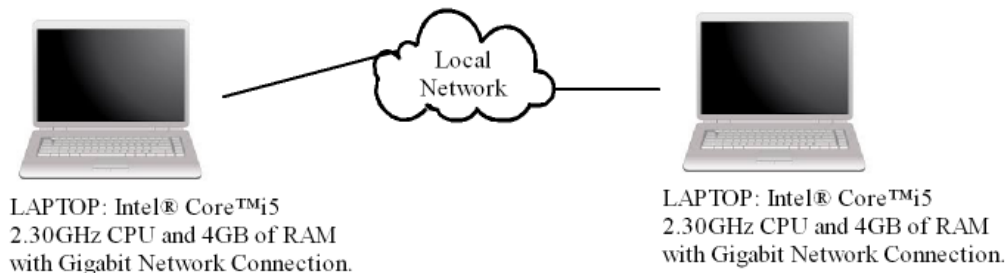


Figure 5-4: Wired Local Area Network

5.1.3 Experiment Factors

The evaluation of the CryptoBI algorithms is done by using methods such as:

- 1) **Performance**, Several performance metrics are collected, such as: encryption time and CPU process time. The encryption time is considered the time that an encryption algorithm takes to produce a cipher text from a plaintext. Encryption time is used to calculate the throughput of an encryption scheme. It indicates the speed of encryption. The throughput of the encryption scheme is calculated as the total plaintext in bytes encrypted divided by the encryption time [39]. The CPU process time is the time that a CPU is committed only to the particular process of calculations, It reflects the load of the CPU. The more CPU time is used in the encryption process, the higher load of the CPU is.

The chosen factor here to determine the performance is the algorithm's speed to encrypt/decrypt data blocks of various sizes

- 2) **Cryptanalysis**. we will use the cryptanalysis according to a specific scenario that can be used for this purpose. Because it is hard to find a tool that can be used in general, all the tools created and published were to a specific algorithm.

To break the CryptoBI algorithms and to decrypt the cipher text, we need to know the images database, color image channel, the key value, the Kv value, and the session type which are the synchronization setting between the two sides of communication, without

knowing those settings, one possible attack under these circumstances is the brute-force approach of trying all possible keys. It's impractical to break the algorithm because the algorithm encrypts the data by using a key length between 128 bits and 512 bits if the algorithm is run on the default setting. However, the key length can be more than 512 bits by changing the default setting.

The attacker needs to know some of the algorithm synchronization settings especially the images database to get the cryptography key by using the cryptanalysis attack. In this evaluation, we consider that the attacker has the images database only. In this case, the attacker should guess the selected image from the database that is used to encrypt the data, and he should guess the other synchronization settings.

The chosen factor here to determine the strength against the cryptanalysis attacks is trying to generate the key when the images database is only known to the attacker.

5.1.4 Simulation Procedure

In the experiments, the following tasks that will be performed are shown as follows:

- Proving the key-updating principle.
- A comparison is conducted between the results of the selected different encryption and decryption schemes in terms of the encryption time.
- A study is performed on the effect of changing packet size at power consumption during throughput for each selected cryptography algorithm. The simulation program used to encrypt different data (text messages or files) size ranges from 300 Kbyte to 10 Megabyte. All the implementations were exact to make sure that the results will be relatively fair and accurate.
- A study is performed on the effect of changing key size for cryptography CryptoBI version1 and verison2 algorithms on power consumption.
- A study is performed on the effect of cryptanalysis attack against the strength of CryptoBI version1 and version2.

5.2 Experimental Results

This section will show the results which are obtained by running the simulation program using different data loads. The results show the impact of changing data load on each algorithm and the impact of cryptanalysis attack against the CryptoBI algorithms version1 and version 2.

5.2.1 CryptoBI Key-Changing Principle

Figure 5-5 shows a plain text. Its encrypted version and the decrypted version using CryptoBI version2 algorithm are shown in figure 5-6 and 5-8. Figure 5-7 shows the other encrypted version for the same plain text, but it's encrypted during different sessions. The difference between the two encrypted texts is clarified in figure 5-6 and 5-7 which proves the principle of key-updating. The encryption and decryption processes are applied between two computers using LAN.

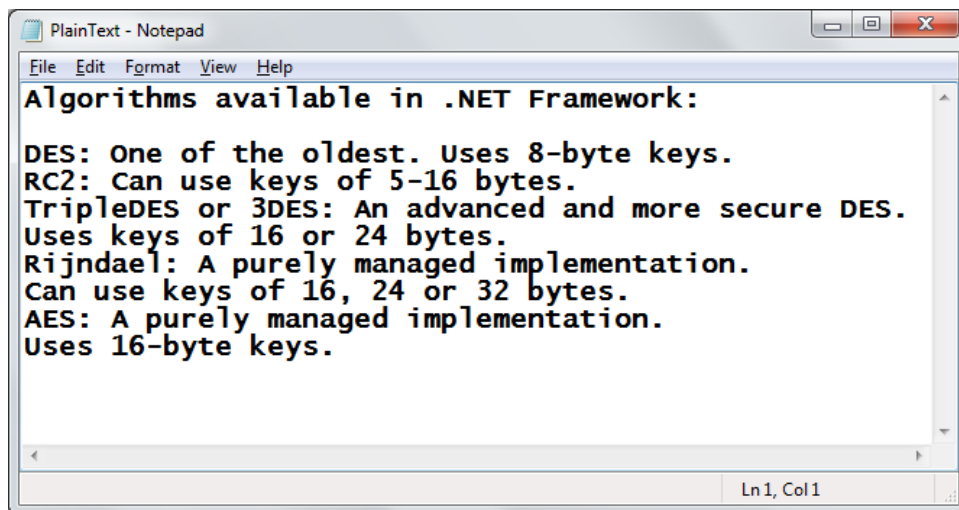


Figure 5-5: The plain text.

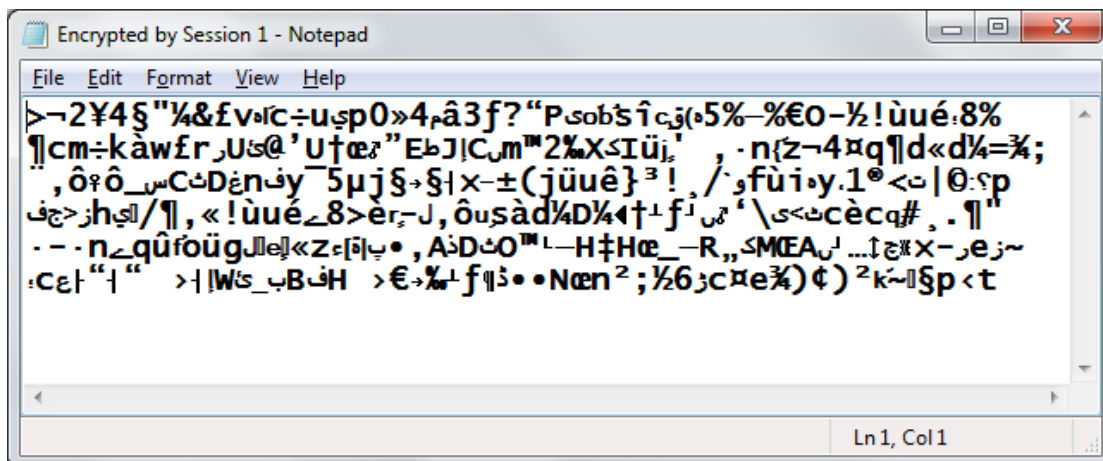


Figure 5-6: The encryption of the plaintext during session 1.

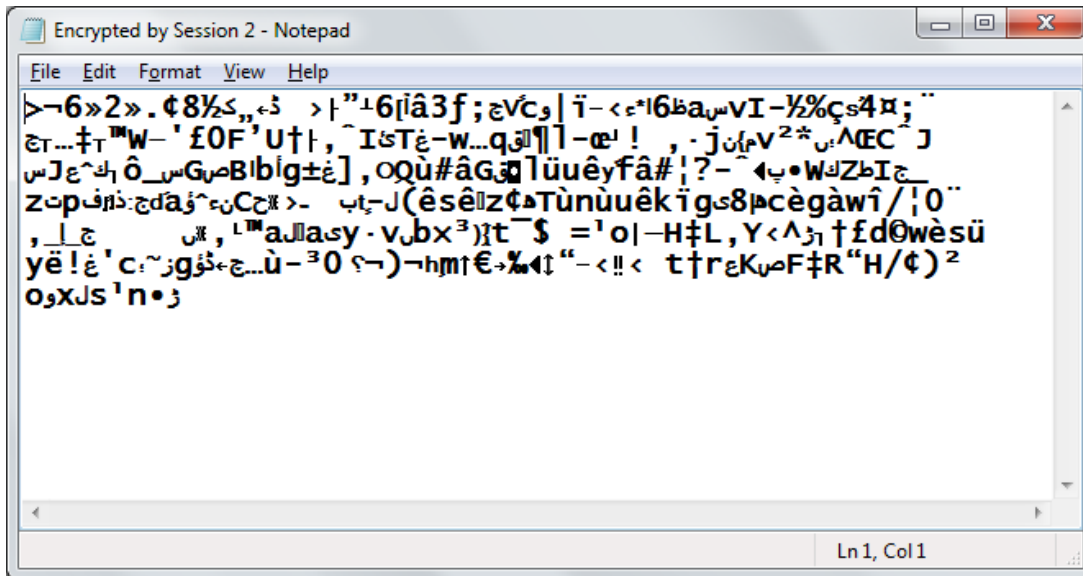


Figure 5-7: The encryption of the plaintext during session 2.

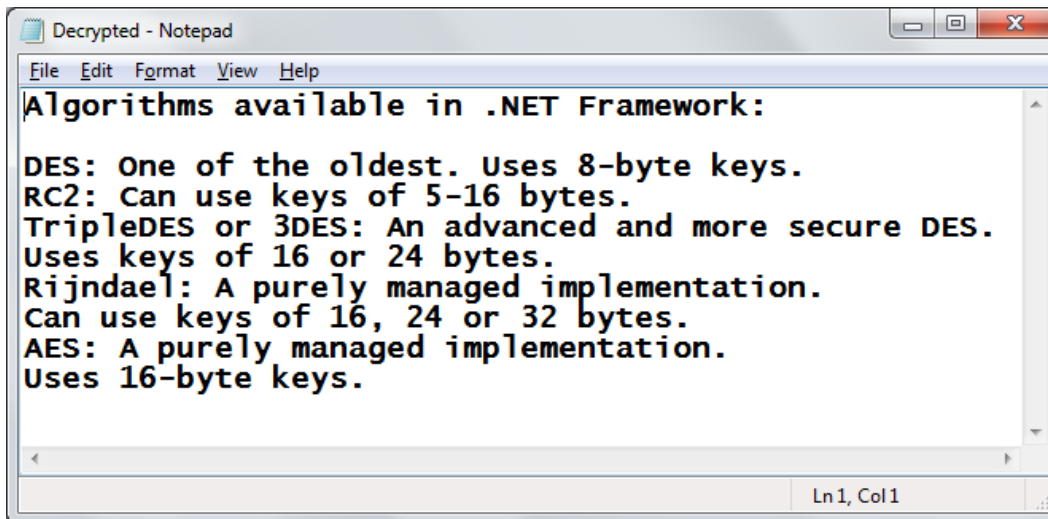


Figure 5-8: The decryption of the plaintext.

5.2.2 Differentiate Output Results of Encryption

The encryption times of CryptoBI version2 and version1 are compared with Blowfish, AES, DES, 3DES, RC2, RC6 and Rijndael using different data sizes ranging from 300 KB to 10 MB, see Figure 5-9. The delay time taken for the encryption process of the different algorithms is measured inside the simulation program between one and four times, therefore, the average of delay time is calculated to reach algorithm at different data sizes, see table 5-2.

Figure 5-8 shows the results of CryptoBI version2 and CryptoBI version1 against selected seven encryption algorithms. We can notice that there is a difference at both methods. The same data is

encrypted by all algorithms. We can realize that the curve of CryptoBI version2 algorithm shows superiority over CryptoBI version1 and all other algorithms because the curves of CryptoBI version2 algorithm have less time comparing to others.

Table 5-2: Comparative execution times (in milliseconds) of the encryption algorithms.

Data Size (KB)	CryptoBI v2	Blowfish	CryptoBI v1	Rijndael	AES	RC6	DES	RC2	TDES
314	7.80	15.60	23.40	35.10	39.00	54.60	50.70	54.60	62.40
403	7.80	15.60	23.40	42.90	46.80	62.40	62.40	62.40	78.00
726	7.80	15.60	54.60	74.10	97.50	109.20	109.20	120.90	140.40
1013	10.68	23.40	74.10	93.60	136.50	140.40	148.20	159.90	195.00
2123	31.20	46.80	148.20	202.80	280.80	300.30	315.90	331.50	397.80
4191	58.50	81.90	292.50	405.60	561.60	585.00	620.10	659.10	791.70
6112	85.80	113.10	436.80	588.90	811.20	838.50	900.90	951.60	1142.70
10561	144.30	187.20	756.60	1010.10	1415.70	1443.00	1536.60	1645.80	1977.30
Average Time (ms)	43.88	62.40	226.20	306.64	423.64	441.68	468.00	498.23	598.16

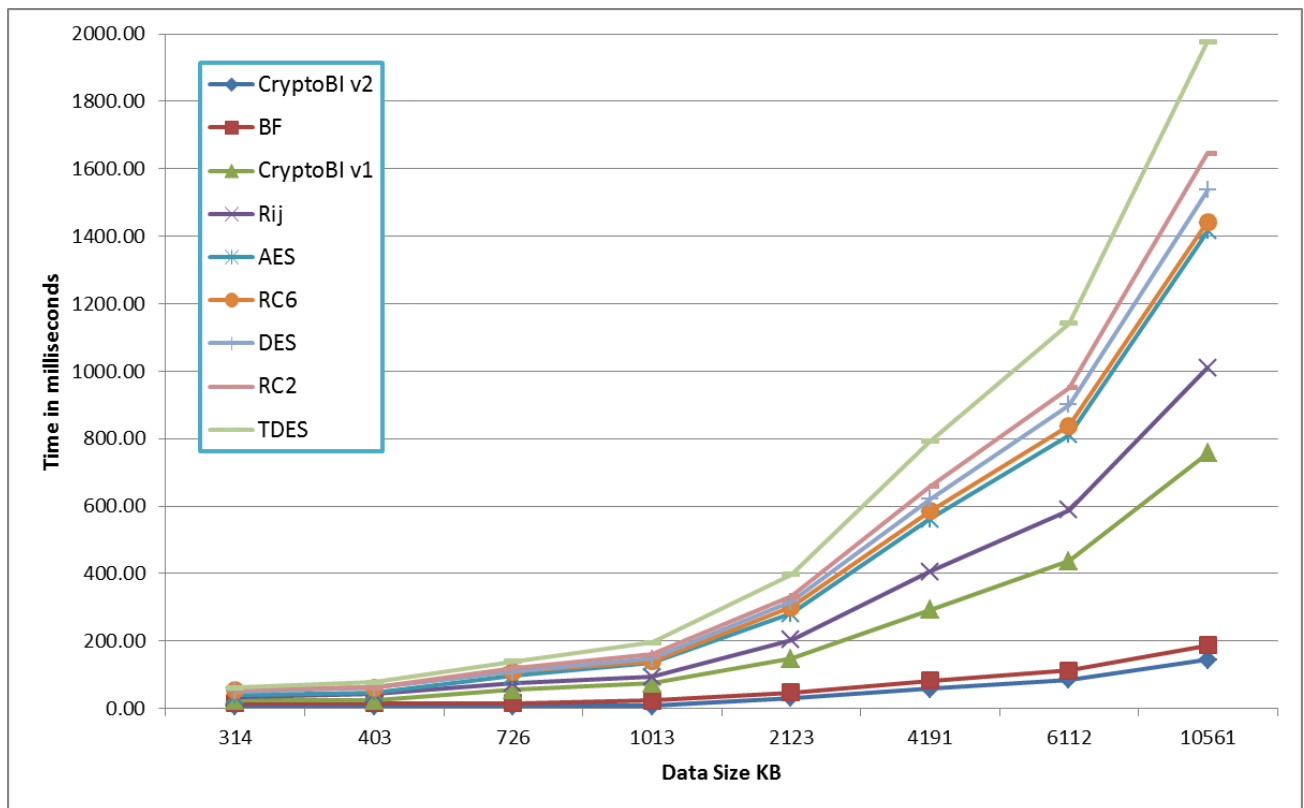


Figure 5-9: Time consumption of encryption algorithm.

5.2.3 Effect of Changing Data Size for Cryptographic Algorithms on Power Consumption

This section discusses the effect of changing data size for cryptographic algorithms on power consumption.

5.2.3.1 Encryption of Different Data Size

Encryption time is used to calculate the throughput of an encryption scheme. The throughput of the encryption scheme is calculated by equation (5.1), as we mention in the previous section (5.2.2), The encryption time of CryptoBI version2 and version1 are compared with of Blowfish, AES, DES, 3DES, RC2, RC6 and Rijndael using different data sizes ranging from 300 KB to 10 MB, therefor, the average data rate “throughput” of the different algorithm is calculated from equation (5.1). The measured values of the average date rate for different algorithms are shown in table 5-3.

$$Dr_{avg} = \frac{1}{Nm} \sum_{i=1}^{Nm} \frac{M_i}{t_i} \text{ (KB/s)} \text{ ----- (5.1)}$$

Where:

- Dr_{avg} is the average data rate (KB /s).
- Nm is Number of messages or files with different sizes from 300KB to 10MB.
- M_i is the message or files size (KB).
- t_i is the time taken to encrypt the message M_i .

Table 5-3: Throughput “Average data rates” comparison at encryption stage.

Encryption algorithm	Throughput or average data rate (KB /s)
CryptoBI v2	70,500.37
Blowfish	42,847.72
CryptoBI v1	14,276.61
Rijndael	10,074.40
AES	7,693.41
RC6	6,864.29
DES	6,658.95
RC2	6,269.00
TDES	5,235.60

As the throughput value is increased, the power consumption of this encryption technique is decreased.

Experimental results for this comparison point are shown in Figure 5-10 at the encryption stage. The results show the superiority of CryptoBI version2 algorithm over other algorithms in terms of processing time. Another point can be noticed here is that Blowfish requires less time than all other algorithms except CryptoBI version2. A third point, CryptoBI version1 has an advantage over other Rijndael, AES, RC6, DES, RC2 and TDES in terms of time consumption and throughput. A fourth point, 3DES has low performance in terms of power consumption and throughput when compared with DES. It always requires more time than DES because of its triple phase encryption characteristics. Finally, it is found that RC2 has low performance and low throughput when compared with other algorithms in spite of the small key size used.

The Figure 5-10 showed that the enhanced algorithm (CryptoBI version2) has higher data rate than CryptoBI version1, AES, Rijndael, DES, 3DES, RC2, RC6 and Blowfish algorithms.

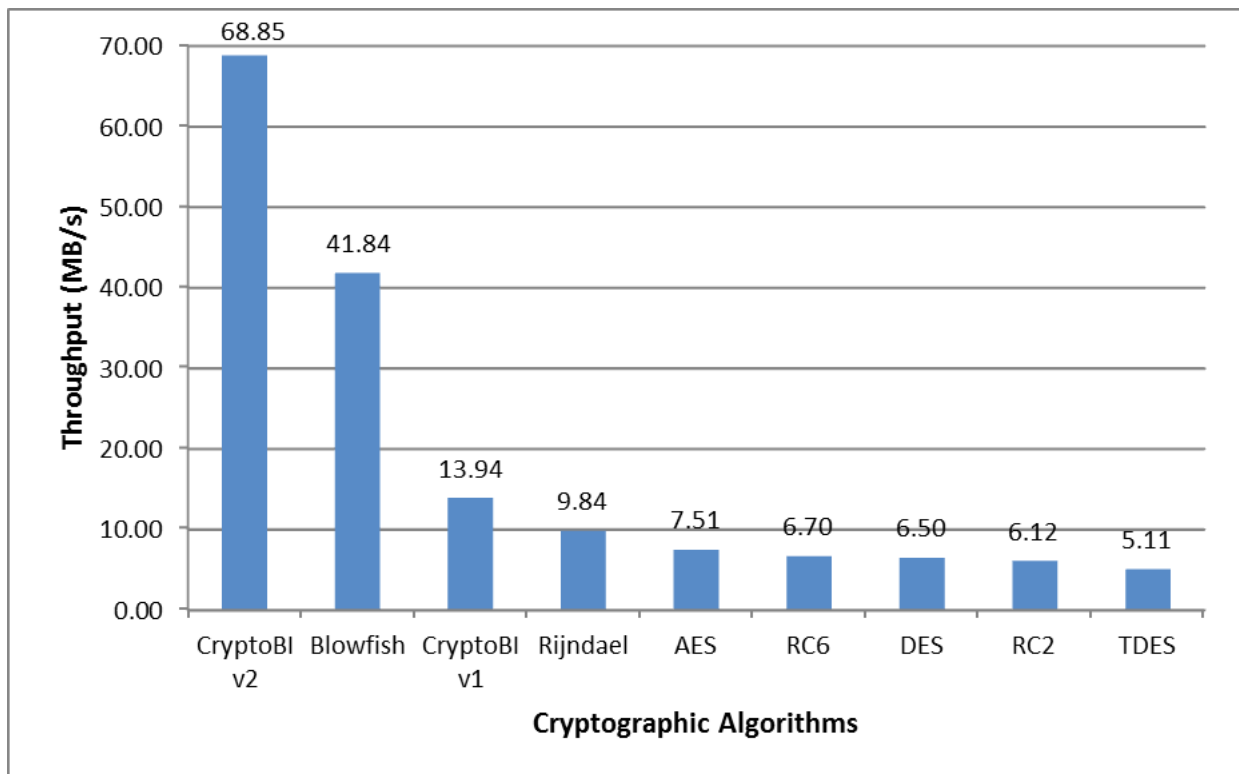


Figure 5-10: Throughput of each encryption algorithm (MB/Sec).

5.2.3.2 Decryption of Different Data Size

Experimental results for this comparison point are shown in Figure 5-11. We can find that CryptoBI version2 is better than other algorithms in terms of throughput and power consumption. The second point that can be noticed here is that CryptoBI version1 requires less time than all algorithms except Blowfish and CryptoBI version2. A third point is that AES has an advantage over other 3DES, DES, RC2 and RC6. The fourth point that can be considered is that Triple DES (3DES) still has low performance of these algorithm. Finally, RC2 still requires more time than RC6.

Table 5-4 : Throughput “Average data rate” comparison at decryption stage.

Decryption algorithm	Throughput or average data rate (KB /s)
CryptoBI v2	78,429.03
Blowfish	65,279.88
CryptoBI v1	14,473.43
Rijndael	8,769.56
AES	7,102.04
RC6	6,187.10
RC2	4,874.94
DES	4,781.85
TDES	4,089.39

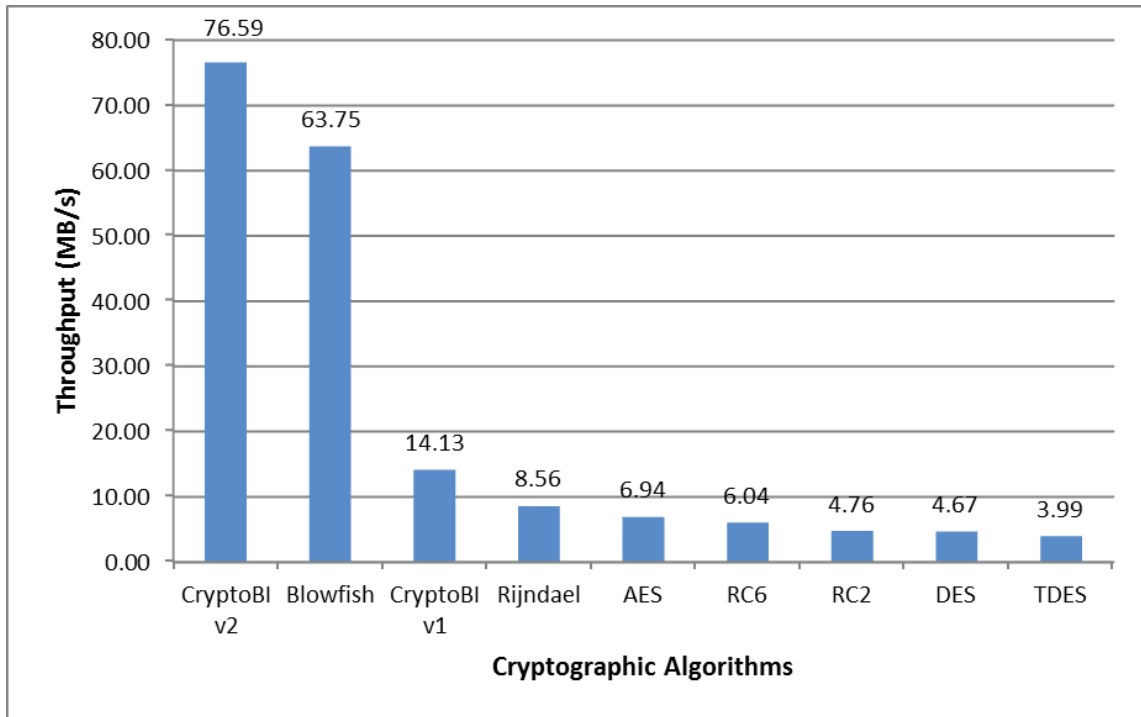


Figure 5-11: Throughput of each decryption algorithm (MB/Sec).

5.2.4 The Effect of Changing Key Size of CryptoBI on Power Consumption

The last performance comparison point is changing different key sizes for CryptoBI version2 and CryptoBI version1 algorithm. In case of CryptoBI version1, we consider the three different key sizes possible i.e., 128-bit, 256-bit and 512-bit keys. The Experimental results are shown in Figure 5-12 and Figure 5-13.

In case of CryptoBI version1, it can be seen that a higher key size does not lead to a clear change in the battery and time consumption. In case of CryptoBI version2, we consider the three different key sizes possible i.e., 128-bit, 256-bit and 512-bit keys. The result is shown in Figure 5-12. It can be seen that a higher key size does not lead to a clear change in the battery and time consumption.

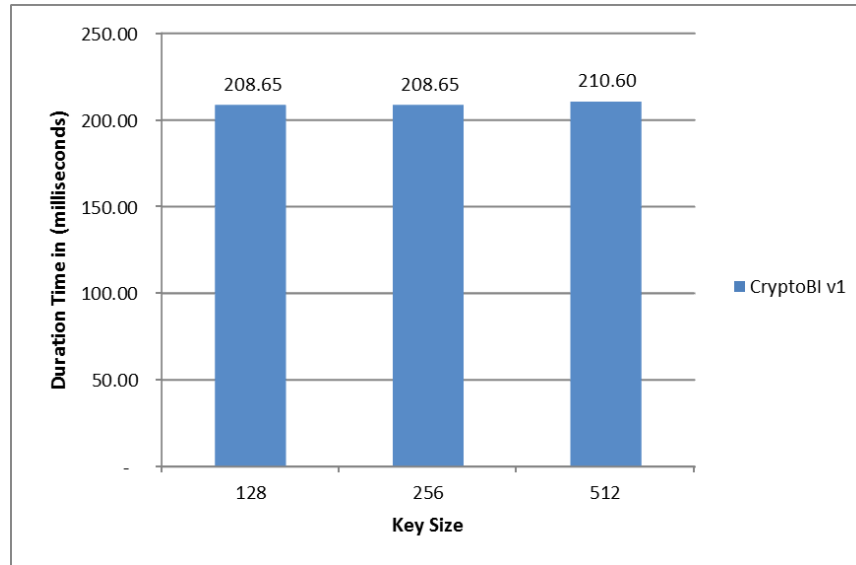


Figure 5-12: Time consumption for different key size for CryptoBI v1.

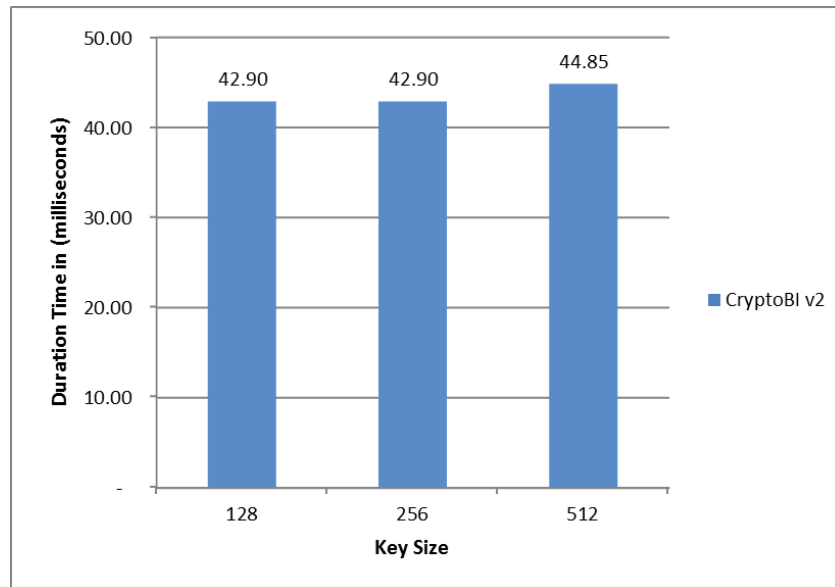


Figure 5-13: Time consumption for different key size for CryptoBI v2.

5.2.5 The Effect of Cryptanalysis Attack against the Strength of CryptoBI Algorithms.

The most difficult problem is presented when all that is available is the ciphertext only. In some cases, not even the encryption algorithm is known, but in general we can assume that the opponent does know the algorithm used for encryption. One possible attack under these circumstances is the brute-force approach of trying all possible keys. If the key space is very large, this becomes impractical.

To break the CryptoBI algorithms and to decrypt the ciphertext, we need to know the images database, color image channel, the key value, the Kv value, and the session type which are the

synchronization setting between the two sides of communication. Without knowing those settings one possible attack under these circumstances is the brute-force approach of trying all possible keys. It's impractical to break the algorithm because the algorithm encrypts the data by using key length between 128 bits and 512 bits in case the algorithm is run on the default settings. However, the key length can be more than 512 bits by changing the default setting.

Thus, the attack must rely on knowledge of the CryptoBI synchronization settings especially the images database. This experiment is performed under the following circumstances; having a ciphertext of a specific plaintext and having the images database, but the other synchronization settings are unknown which are color image channel, the k value, the session type, and the Kv value. These settings are necessary to get the cryptography key of ciphertext, so we have to guess values for these unknown synchronization settings then run the key generation algorithm to produce the key which is used to decrypt the ciphertext, then guess operation repeated until getting the plaintext from the ciphertext. The color image channel has three possible values R, G and B. The k value is between 0 and N. The session time type determines the number of images in the database where there are 7 images for daily, 24 images for hourly, or M images for custom session type. Finally, the Kv value is between 0 and 2^{16} and this value is used for CryptoBI version2 only.

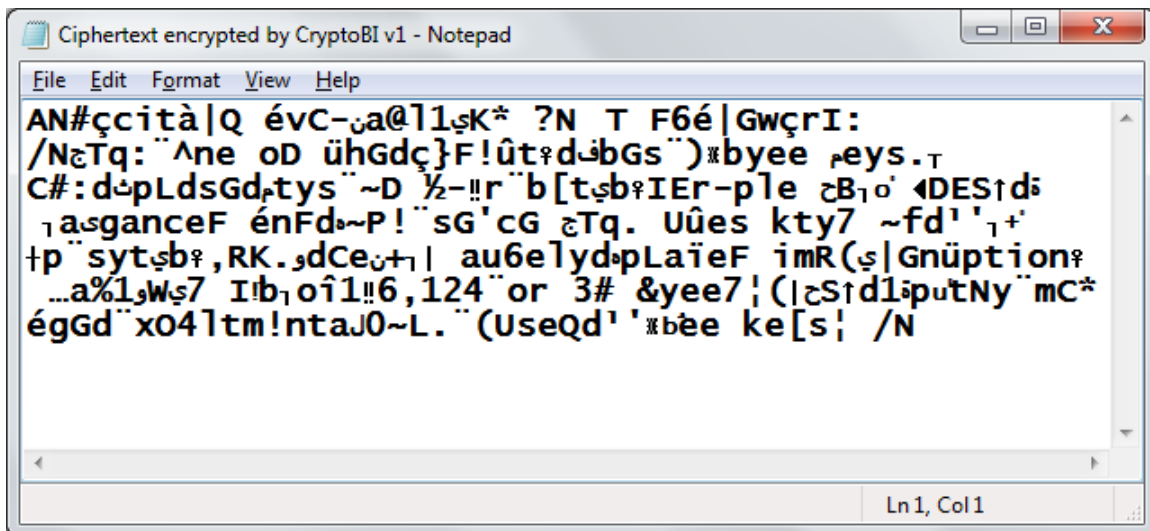


Figure 5-14: CryptoBI v1 ciphertext for cryptoanalysis

After applying this scenario on the CryptoBI version1 and version2 algorithms, the experimental result for CryptoBI version1 algorithm is positive, and we got the key after 0.5 hour of guessing

for the unknown synchronization settings, running the key generation algorithm, and running the decryption algorithm. Some of this experiment results are shown in table 5-5. The table shows the guessed values of the synchronization settings, the generated key using the guessed values, and the result of ciphertext decryption. The Figure 5-14 shows the ciphertext that is used for CryptoBI version1 crypto analysis experiment.

Table 5-5: Some of cryptanalysis result for CryptoBI v1.

mage	Image Channel	K	Generated Key	Result (Success/Fail)
1.JPG	R	1	jjfjZfZ-UUšf ^a Vf	Fail
1.JPG	R	2	jjfjZfZ-UUšf ^a Vf	Fail
1.JPG	R	3	jjfjZfZ-UUšf ^a Vf	Fail
1.JPG	R	4	jjfjZfZ-UUšf ^a Vf	Fail
1.JPG	R	5	jjfjZfZ-UUšf ^a Vf	Fail
1.JPG	R	6	jjfjZfZ-UUšf ^a Vfi ^a	Fail
1.JPG	R	7	jjfjZfZ-UUšf ^a Vfi ^a V ^a •	Fail
1.JPG	R	8	jjfjZfZ-UUšf ^a Vfi ^a V ^a • TM -e TM	Fail
1.JPG	R	9	jjfjZfZ-UUšf ^a Vfi ^a V ^a • TM -e TM -jV	Fail
1.JPG	R	10	jjfjZfZ-UUšf ^a Vfi ^a V ^a • TM -e TM -jVŸUY	Fail
1.JPG	R	11	jjfjZfZ-UUšf ^a Vfi ^a V ^a • TM -e TM -jVŸUYf•e	Fail
1.JPG	R	12	jjfjZfZ-UUšf ^a Vfi ^a V ^a • TM -e TM -jVŸUYf•ešVY	Fail
1.JPG	R	13	jjfjZfZ-UUšf ^a Vfi ^a V ^a • TM -e TM -jVŸUYf•ešVY •f	Fail
1.JPG	R	14	jjfjZfZ-UUšf ^a Vfi ^a V ^a • TM -e TM -jVŸUYf•ešVY •f •Ÿ	Fail
1.JPG	R	15	jjfjZfZ-UUšf ^a Vfi ^a V ^a • TM -e TM -jVŸUYf•ešVY •f •Ÿ ^a	Fail
1.JPG	R	16	jjfjZfZ-UUšf ^a Vfi ^a V ^a • TM -e TM -jVŸUYf•ešVY •f •Ÿ ^a	Fail
3.JPG	R	4	ŸZ© ^{TMTM} •YššifVVZšU	Fail
3.JPG	G	4	UU• iZf©fj VšŸš	Fail
3.JPG	B	4	šjjfjZfZ-UUšf ^a Vf	Fail
3.JPG	R	8	ŸZ© ^{TMTM} •YššifVVZšUišf-©fYY	Fail
5.JPG	R	8	jZeV i©Zf ^{TMa} •• TM Veff iš©e	Fail
5.JPG	R	16	eV Z-šiVijšeVšZVZŸš ^a feŸViVŸZ TM ©VjYZZejjUŸUššieVj	Fail
12.JPG	R	1	VieVf• TM Ze Ve©fVU	Fail
12.JPG	R	2	VieVf• TM Ze Ve©fVU	Fail
12.JPG	R	6	VieVf• TM Ze Ve©fVUVe	Fail
12.JPG	R	7	VieVf• TM Ze Ve©fVUVeŸ ^a	Fail
12.JPG	R	8	VieVf• TM Ze Ve©fVUVeŸ ^a TM ^{YU}	Success

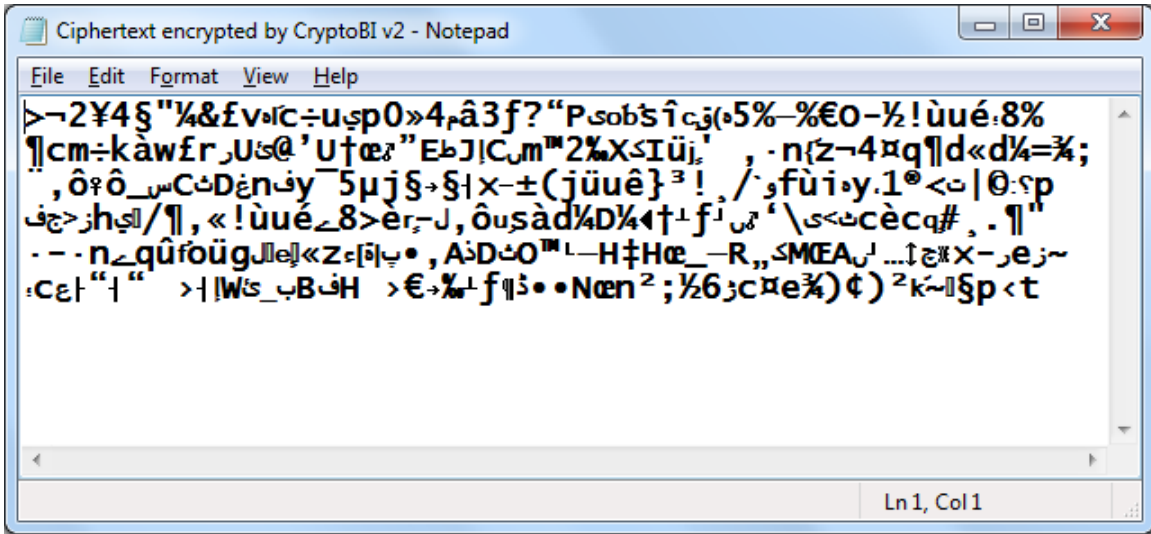


Figure 5-15: CryptoBI v2 ciphertext for cryptoanalysis

Similarly, the experimental result for CryptoBI version2 algorithm is negative because we spent 12 hours of guessing for the unknown synchronization settings without finding the right key to get the right plaintext. Some of this experiment results are shown in table 5-6. Figure 5-15 shows the ciphertext that is used for CryptoBI version2 crypto analysis experiment.

Table 5-6: Some of cryptanalysis result for CryptoBI v2.

Image	Channel	K value	Kv value	Key	Result
1.JPG	R	1	10	W2ìbJ²íí,g	Fail
1.JPG	R	2	10	W2ìbJ²íí,g	Fail
1.JPG	R	3	10	W2ìbJ²íí,g	Fail
1.JPG	R	4	10	P ⁻ e™¼?—e•fbÿZÔ	Fail
1.JPG	R	5	10	í^É3}∅/í,î¿A,-	Fail
1.JPG	R	6	10	í^É3}∅/í,î¿A,-L.	Fail
1.JPG	R	7	10	B½—gù'œ[,x†r^šb•é	Fail
1.JPG	R	8	10	W2ìbJ²íí,g'G°? ì¿	Fail
1.JPG	R	9	10	W2ìbJ²íí,g'G°? ì¿5 â	Fail
1.JPG	R	10	10	W2ìbJ²íí,g'G°? ì¿5 â—[Ö	Fail
1.JPG	R	11	10	W2ìbJ²íí,g'G°? ì¿5 â—[Ö ' <	Fail
1.JPG	R	12	10	W2ìbJ²íí,g'G°? ì¿5 â—[Ö ' <.Áy	Fail
1.JPG	R	13	10	P ⁻ e™¼?—e•fbÿZÔ\$•bv∅Û†n+Ë7' =r~g^ÿñ'4	Fail
1.JPG	R	14	10	„{.ÐöÀ<¶Eðê½Æ1Ô"úú" <hð ∅ Ë%xw/¿ Ú@-ç	Fail
1.JPG	R	15	10	„{.ÐöÀ<¶Eðê½Æ1Ô"úú" <hð ∅ Ë%xw/¿ Ú@-ç Y'	Fail
1.JPG	R	16	10	„{.ÐöÀ<¶Eðê½Æ1Ô"úú" <hð ∅ Ë%xw/¿ Ú@-ç Y' Ò	Fail
1.JPG	R	17	10	„{.ÐöÀ<¶Eðê½Æ1Ô"úú" <hð ∅ Ë%xw/¿ Ú@-ç Y' Ò ~i	Fail
1.JPG	G	1	10	òq• ú~MP«x,Æ3	Fail
1.JPG	G	2	10	ù8Æ• W¥\$T»?b—	Fail
1.JPG	G	3	10	ÿ 'lòùH³¹AÓ&Í	Fail
1.JPG	G	4	10	ù>c¿>*ÒR©Û-Ï•	Fail
1.JPG	G	5	10	òq• ú~MP«x,Æ3	Fail

1.JPG	G	6	10	ù8Æ• W¥\$T»?b—ör	Fail
1.JPG	G	7	10	ÿ 'lòùH³!AÓ&í+3Äx~	Fail
1.JPG	G	8	10	ÿ 'lòùH³!AÓ&í+3Äx~Sç{	Fail
1.JPG	G	9	10	ÿ 'lòùH³!AÓ&í+3Äx~Sç{y°C	Fail
1.JPG	G	10	10	ù8Æ• W¥\$T»?b—ör...È. y è[~®Dw	Fail
1.JPG	G	11	10	òq• ú~MP«x,Æ3 iè "bùù Ý%7b•òý_'	Fail
1.JPG	G	12	10	òq• ú~MP«x,Æ3 iè "bùù Ý%7b•òý_ 'KxW	Fail
1.JPG	G	13	10	òq• ú~MP«x,Æ3 iè "bùù Ý%7b•òý_ 'KxW—W—	Fail
1.JPG	G	14	10	ù8Æ• W¥\$T»?b—ör...È. y è[~®Dwz—D"(Æ"E#BÉ	Fail
1.JPG	G	15	10	þM±ß-è(ÓíVÈ>òÙ î [špöÒ!&ì~Y%ËÄ-Ã+cŠ#Éè '6	Fail
1.JPG	G	16	10	ÿ &Ø'í ó"hu...@P•[j•ô#Kùèe•cH@âÚÛ,Áào Ôâp	Fail
1.JPG	G	17	10	þM±ß-è(ÓíVÈ>òÙ î [špöÒ!&ì~Y%ËÄ-Ã+cŠ#Éè '6Ö iù7°	Fail
1.JPG	B	1	10	8Çâ•mâk•H²á	Fail
1.JPG	B	2	10	á-%ouCqí"³H(Ò•j%o	Fail
1.JPG	B	3	10	Ã<ê%otÖ&j'T "x	Fail
1.JPG	B	4	10	p•Ä:!Ü4ÈÛ""gÆ3C	Fail
1.JPG	B	5	10	Ã<ê%otÖ&j'T "x	Fail
1.JPG	B	6	10	á-%ouCqí"³H(Ò•j%oo	Fail
1.JPG	B	7	10	p•Ä:!Ü4ÈÛ""gÆ3CμÖ"/Ç	Fail
1.JPG	B	8	10	Ã<ê%otÖ&j'T "xâ_~É-j;G	Fail
1.JPG	B	9	10	8Çâ•mâk•H²áÙh•al-ðu~	Fail
1.JPG	B	10	10	p•Ä:!Ü4ÈÛ""gÆ3CμÖ"/Ç—^æñ>W#	Fail
1.JPG	B	11	10	8Çâ•mâk•H²áÙh•al-ðu~J"Q À0	Fail
1.JPG	B	12	10	ãq•HμCεp5Æ#xοóŠé²EÈ.çS•u7%!ÑxMÛîq	Fail
1.JPG	B	13	10	p•Ä:!Ü4ÈÛ""gÆ3CμÖ"/Ç—^æñ>W#G^h ìà¥H	Fail
4.JPG	R	1	23	ëQóÜ e— ág'S	Fail
4.JPG	R	2	23	uŠÖ(y,,m•±Ji2Ç"	Fail
4.JPG	R	3	23	uŠÖ(y,,m•±Ji2Ç"	Fail
4.JPG	R	4	23	uŠÖ(y,,m•±Ji2Ç"	Fail
4.JPG	R	5	23]ç5É] šb+DùŠo'	Fail
4.JPG	R	6	23]ç5É] šb+DùŠo'Ñ®	Fail
4.JPG	R	7	23	²Ej"»B5ÇWxóâR!a-â	Fail
4.JPG	R	8	23	ëQóÜ e— ág'S¥•h"cm"0	Fail
4.JPG	R	9	23	ëQóÜ e— ág'S¥•h"cm"0Ä Ú	Fail
4.JPG	R	10	23	ëQóÜ e— ág'S¥•h"cm"0Ä ÚÛ ñ	Fail
4.JPG	R	11	23	ëQóÜ e— ág'S¥•h"cm"0Ä ÚÛ ñ •q	Fail
4.JPG	R	12	23	ëQóÜ e— ág'S¥•h"cm"0Ä ÚÛ ñ •qLf	Fail

By doing some calculations, we can find the amount of time that is needed to get plaintext from the ciphertext. This is in case the attacker has the images database of CryptoBI algorithm. According to that, the attack for CryptoBI algorithms involves trying every possible value of synchronization setting until an intelligible translation of the ciphertext into plaintext is obtained. On average, half of all possible values must be tried to achieve success. The equation (5.2) shows number of attempts involved for various synchronization setting values,

$$\text{number of attempts} = N_{\text{Img}} \times \text{ImgC} \times K_{\text{max}} \times K_{v\text{max}} \times N_{\text{Seg}} \text{ ----- (5.2)}$$

Where,

N_{Img} is the number of images in the images database.

ImgC is the number of color image channels which are three.

K_{max} is the maximum value of K and its value ranges between 0 and N, where $N = \sqrt{\text{size of data}}$

K_{vmax} is the maximum value of K_v variable and its use in CryptoBI version2 only.

N_{Seg} is the number of image segments, according to the algorithm it's four segments, its use in CryptoBI version2 only.

The equation (5.3) shows the time involved for all the Cryptanalysis tries.

$$\text{time required} = \text{number of attempts} \times \text{time to perform a single try} \text{ ----- (5.3)}$$

Where, the time to perform a single try is the time to generate the key and decrypt the ciphertext.

Table 5-7 shows the time involved in various synchronization setting spaces. Results are shown for 1 Kilobytes of ciphertext, and I assumed the time to perform a single try as follows; 1 second, 1 millisecond and 1 microsecond, which are a reasonable order of magnitude for today's machines. With the use of massively parallel organizations of microprocessors, it may be possible to achieve processing rates many orders of magnitude greater.

Table 5-7: Time Required for Exhaustive Key Search.

data size (byte)	Number of Images	Image Channels	Max of K	max of Kv	number of segments	time of single try	Time required for CryptoBI v1		Time required for CryptoBI v2	
1024	24	3	32	65536	4	1 s	38.40	Minutes	19.42	Years
1024	200	3	32	65536	4	1 s	5.33	Hours	161.82	Years
1024	1000	3	32	65536	4	1 s	26.67	Hours	809.09	Years
1024	24	3	32	65536	4	1 ms	2.30	Seconds	6.99	Days
1024	200	3	32	65536	4	1 ms	19.20	Seconds	1.94	Months
1024	1000	3	32	65536	4	1 ms	1.60	Minutes	9.71	Months
1024	24	3	32	65536	4	1 μ s	2.30	millisec	10.07	Minutes
1024	200	3	32	65536	4	1 μ s	19.20	millisec	1.40	Hours
1024	1000	3	32	65536	4	1 μ s	0.10	Seconds	6.99	Hours

5.3 Conclusion

This chapter described the techniques and simulation choices made to evaluate the performance and strength of the CryptoBI algorithms (version1 and version2). In addition to that, this chapter discussed the methodology related parameters such as: system parameters, experiment factors, and experiment initial settings.

In the performance evaluation results of enhanced algorithm CryptoBI version2 and selected symmetric encryption algorithms which are CryptoBI version1, AES, Rijndael, DES, 3DES, RC2, RC6 and Blowfish, we concluded that the CryptoBI version2 has a better performance than the other symmetric encryption algorithms, followed by Blowfish.

Also in the strength evaluation results of CryptoBI version2 and version1 algorithms against cryptanalysis attack, it was concluded that the CryptoBI version2 is stronger than the CryptoBI version1 against the attack where CryptoBI version1 can be broken in a short time in minutes. But the CryptoBI version2 can be broken in long time in months as average that depends on the speed of the attacker machine and algorithm synchronization setting. In this experiment, we assume that the attacker knows the algorithm and images database. Finally, in the experiment of changing key size for CryptoBI version1 and version2, it can be seen that higher key size does not leads to a change in the battery and time consumption.

Chapter 6. Conclusions and Future Directions

In this chapter, we will discuss the conclusions for this work and the future directions

6.1.1 The conclusion

The Cryptography provides security to the data that is transfer. The security of cryptographic system relies on the fact that the cryptographic keys are secret and known only to a legitimate user. Hence, we enhanced CryptoBI version1 algorithm based on the key that is generated directly from an image which is selected from a database of images according to session time, so the process of key generation is based on sessions. This creates more complexity to crack or guess the keys by using cryptanalysis techniques; this approach is called the key-updating method which is a new approach to increase the difficulty to discover the key. To break this algorithm, we need to know the images database, color image channel, the k value, Kv value and the session type.

This thesis presents a performance evaluation of enhanced algorithm CryptoBI version2 and selected symmetric encryption algorithms which are CryptoBI version1, AES, Rijndael, DES, 3DES, RC2, RC6 and Blowfish, as this thesis presents evaluation the strength of CryptoBI version2 and version1 algorithms against cryptanalysis attack to decrypt the cipher text. Several points can be concluded from the experimental results.

1. The CryptoBI Algorithms version1 and version2 support the key-updating approach.
2. In case of changing packet size, it was concluded that CryptoBI version2 has a better performance (higher data rate) than the other symmetric encryption algorithms, followed by Blowfish. As the throughput value is increased, the power consumption is decreased.
3. CryptoBI version1 requires less time than all algorithms except Blowfish and CryptoBI version2
4. In case of attacking CryptoBI algorithms using the cryptanalysis attack, where we assume that the attacker knows the algorithm and images database. It was concluded that the CryptoBI version2 is stronger than the CryptoBI version1 against the attack where CryptoBI version1 can be broken in a short time in minutes. But the CryptoBI version2 can be broken in long time in months as average that depends on the speed of the attacker machine and algorithm synchronization setting.

5. Brute-force attack against the CryptoBI algorithms by trying all possible keys becomes impractical, because the CryptoBI generate large keys.
6. In case of changing key size for CryptoBI version1 and version2, it can be seen that higher key size does not leads to a change in the battery and time consumption.
7. The 3DES has a disadvantage over all other algorithms in terms of time consumption.

The enhanced CryptoBI version2 algorithm process has an advantage, that the key generation is based on session, so in every session, we have a different key, as the key length varies according to data size. The key length can have no boundary and vary according to data size. The key length can be more than 512 bits or bounded by the algorithm setting to determine the minimum and maximum key length. This process is more flexible that any RGB image can be used for key generation as the key generation is directly based on the image content.

From all the previous concluded results in this thesis, we summarize the most important differences between the CryptoBI version1 and version2 in Table 6-1:

Table 6-1: Most important differences between the CryptoBI version1 and version2.

	CryptoBI version1	CryptoBI version2
Generated key	<ul style="list-style-type: none"> • Vary every session • Vary according to data size • can be more than 512 bits 	<ul style="list-style-type: none"> • Vary every session • Vary according to data size • can be more than 512 bits
Performance	Low (226 ms as average)	High (44 ms as average)
Brute-force attack	Impractical	Impractical
Cryptanalysis attack	success in short time (minutes as average)	Need long time (months as average)

Finally, through our work in this thesis, it can be concluded that we can encrypt and decrypt the user and / or application data very easily and simplicity is the spirit of this algorithm. Although it creates complications for the eavesdropper to decrypt the cipher text, but for the end-users this algorithm does not pose any complications to perform functional activities.

6.1.2 The Future Directions

Future scope for CryptoBI version2 algorithm can be one of the following directions:

1. Extending the region of CryptoBI algorithms to work in the wide area network (WAN). Using the algorithm in WAN become serve a large number of users and need a technique or methods to control the generated keys between the users. We propose for this purpose the key distribution technique that is concern key management between the users groups and it make sure that each group has different key.
2. The key generation algorithm could be develop to improve the strengths of the key generation algorithm to generate random and unpredictable keys. Finding a method to generate a key from image that is complicating the cryptanalysis work to break the algorithm by finding the cryptography key and make it impractical, because the strength of the algorithm strongly cryptography key.

Bibliography

- [1] Marwaha, P., "Visual cryptographic steganography in images," Karur, India, 2010.
- [2] W. Stallings, "Cryptography and Network Security - Principles and Practices", Pearson Education Third Edition ed., 2002.
- [3] Seshadri, R. and T.RaghuTrivedi, "Efficient Cryptographic Key Generation using Biometrics," *Int. J. Comp. Tech. Appl.*, vol. 2 (1), pp. 183-187, 2011.
- [4] "Bloombase Knowledgebase," 2014. [Online]. Available: <http://kb.bloombase.com/kb/?View=entry&EntryID=33>.
- [5] "Wikipedia (Key exchange)," Dec. 1, 2013. [Online]. Available: (http://en.wikipedia.org/wiki/Key_exchange).
- [6] Asha A., Liyamol A. and Nisha V K, "'RC5 Encryption Using Key Derived From Fingerprint Image'," in *Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference*, Dec. 2011.
- [7] Wong Siaw Lang, Nur Azman Abu, Shahrin Sahib, "Cryptographic Key From Webcam Image," in *International Journal of Cryptology Research 1 (1): 115-127 (2009)*.
- [8] R. Palraj and P. Murali , "True Random number generator method based on image for key exchange algorithm," in *2009 International Symposium on Computing, Communication, and Control (ISCCC 2009)*, IACSIT Press, Singapore.
- [9] Barhoom, Tawfiq S. and Abusilmiyeh, Zakaria M., "'A Novel Cryptography Method Based on Image for Key Generation'," in *Palestinian International Conference on Information and Communication Technology*, Palestine, Gaza, 2013.
- [10] A. J.MENEZES, P. C. V. OORSCHOT, AND S. A. VANSTONE, *Handbook of applied cryptography*, 1996.

- [11] P. Kandept, "AN OVERVIEW OF PUBLIC KEY INFRASTRUCTURE," Fall 2013.
- [12] National Research Council, *Computers at Risk: Safe Computing in the Information Age*, Washington, DC: The National Academies Press, 1991.
- [13] *Communications Privacy: Federal Policy and Actions, General Accounting Office Report GAO/OSI-94-2*, November 1993.
- [14] D. Zhu, *Profiling Symmetric Encryption Algorithms for Implantable Medical Devices*, 2008.
- [15] "Wikipedia," 2014. [Online]. Available: [Http://en.wikipedia.org/wiki/Symmetric-key_algorithm](http://en.wikipedia.org/wiki/Symmetric-key_algorithm).
- [16] "Wikipedia," 2014. [Online]. Available: [Http://en.wikipedia.org/wiki/Public-key_cryptography](http://en.wikipedia.org/wiki/Public-key_cryptography).
- [17] Anoop MS, "*Public Key Cryptography: Applications Algorithms and Mathematical Explanations*", Tata Elxsi Ltd, India, anoopms@tataelxsi.co.in .
- [18] J. Thakur and N. Kumar, "DES, AES and Blowfish: Symmetric Key Cryptography Algorithms Simulation Based Performance Analysis," *International Journal of Emerging Technology and Advanced Engineering*, vol. 1, no. 2, December 2011.
- [19] B. Schneier, "Applied Cryptography - Second Edition", John Wiley & Sons, 1996, pp. 210-211.
- [20] Arjen K. Lenstra and Eric R. Verheul, "Selecting Cryptographic Key Sizes", *J.Cryptology* 14(4), pp. 255-293, 2001.
- [21] (STG), NIST Computer Security Division's (CSD) Security Technology Group, *Block cipher modes*, Cryptographic Toolkit. NIST. Retrieved April 12, 2013.
- [22] *Cryptography Engineering: Design Principles and Practical Applications*. Ferguson, N., Schneier, B. and Kohno, T. Indianapolis: Wiley Publishing, Inc. 2010. pp. 63, 64. ISBN 978-

0-470-47424-2..

- [23] NIST Computer Security Division's (CSD) Security Technology Group (STG) (2013). "Proposed modes". *Cryptographic Toolkit*. NIST. Retrieved April 14, 2013..
- [24] Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone (1996). *Handbook of Applied Cryptography*. CRC Press. pp. 228–233. ISBN 0-8493-8523-7..
- [25] ISO JTC 1/SC 27 (2006). "ISO/IEC 10116:2006 - Information technology -- Security techniques -- Modes of operation for an n-bit block cipher". *ISO Standards catalogue*..
- [26] Kuo-Tsang Huang, Jung-Hui Chiu, and Sung-Shiou Shen (January 2013). "A Novel Structure with Dynamic Operation Mode for Symmetric-Key Block Ciphers". *International Journal of Network Security & Its Applications (IJNSA)* 5 (1): 19..
- [27] "Block cipher mode of operation," Dec 2014. [Online]. Available: http://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#Common_modes.
- [28] "Search Security," Jun, 2014. [Online]. Available: <http://searchsecurity.techtarget.com/definition/cryptanalysis>.
- [29] B. Santhi, K.S. Ravichandran, A.P. Arun and L. Chakkarapani, "A Novel Cryptographic Key Generation Method Using Image Features," *Research Journal of Information Technology* 4(2): 88-92, 2012.
- [30] Lifang Wu ; Xingsheng Liu ; Songlong Yuan ; Peng Xiao, "A Novel Key Generation Cryptosystem Based on Face Features", ICSP2010 Proceedings , IEEE, 2010.
- [31] Priyanka.M, Lalitha Kumari.R, Lizyflorance.C and John Singh. K, "A New Randomized Cryptographic Key Generation Using Image," in *International Journal of Engineering Science and Innovative Technology (IJESIT)*, Volume 2, Issue 6, November 2013.
- [32] Ramesh, G. and Umarani, R., ""UMARAM: A NOVEL FAST ENCRYPTION ALGORITHM FOR DATA"," in *ICCCCT'10, IEEE, 2010*.

- [33] Abd Elminaam, Diaa Salama; Abdual Kader, Hatem Mohamed; Hadhoud, Mohiy Mohamed, "Evaluating The Performance of Symmetric Encryption Algorithms," in *International Journal of Network Security*, Vol.10, No.3, PP.213–219, May 2010.
- [34] OP Verma, Ritu Agarwal, Dhiraj Dafouti and Shobha Tyagi, "Performance Analysis Of Data Encryption Algorithms," IEEE, 2011.
- [35] Mandal, Bijoy Kumar ; Bhattacharyya, Debnath; Bandyopadhyay and Samir Kumar, "Designing and Performance Analysis of a Proposed Symmetric Cryptography Algorithm," in *2013 International Conference on Communication Systems and Network Technologies*, IEEE, 2013.
- [36] 2014. [Online]. Available: <http://msdn.microsoft.com/en-us/library/ms978415.aspx>.
- [37] 2013. [Online]. Available: ([http://en.wikipedia.org/wiki/Channel_\(digital_image\)](http://en.wikipedia.org/wiki/Channel_(digital_image))).
- [38] "ChilKat Software," 2014. [Online]. Available: <http://www.chilkatsoft.com/encryption-features.asp>.
- [39] A. A. Tamimi, "Performance Analysis of Data Encryption Algorithms", Retrieved Oct. 1, 2008.