SECURITY ANALYSIS OF ADVANCED ENCRYPTION STANDARD (AES) FOR IMAGE ENCRYPTION

RIDVAN KÜÇÜKBACAK

JUNE 2022

SECURITY ANALYSIS OF ADVANCED ENCRYPTION STANDARD (AES) FOR IMAGE ENCRYPTION

BY

RIDVAN KÜÇÜKBACAK

DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF ARTS IN MANAGEMENT INFORMATION SYSTEMS

YEDITEPE UNIVERSITY

2022

PLAGIARISM

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Rıdvan KÜÇÜKBACAK

23.05.2022

ABSTRACT

In recent years, there have been major advancements in multimedia technologies and accordingly, file transfer over the internet has become extremely frequent. However, some security issues can occur in the internet, since it is a very insecure channel. To preserve the privacy and security of multimedia data transmitted over the internet, a number of encryption algorithms have been developed. This research presents a methodology for examining practically used image processing encryption algorithms. To measure the quality of encoded images, a number of factors such as correlation coefficient and information entropypixel change rate are employed instead of visual evaluation. In this study, the image encryption efficiency of the Advanced Encryption Standard (AES) is analyzed and the security level is evaluated. For digital pictures, AES security evaluations have been conducted against brute force, statistical, and other attacks. The results of the tests are provided to see if these algorithms are secure for digital photos. Examination of the image encryption algorithms reveal that images can be encrypted by the AES method robustly. Applied tests are independent from encryption algorithm and thus they are applicable to all image encryption algorithms.

Key words: Image Encryption, Security Analysis, AES, Cryptography

ÖZET

Son yıllarda multimedya teknolojilerinde önemli gelişmeler oldu. İnternet üzerinden ses, video ve görüntü gibi multimedya verilerinin iletimi oldukça yaygınlaşmıştır. Bununla birlikte İnternet, çok güvensiz bir kanaldır ve bu, bir dizi güvenlik sorununa neden olmaktadır. İnternet gibi güvenli olmayan bir kanal üzerinden multimedya verilerinin gizliliğini ve güvenliğini sağlamak için bir çok şifreleme şeması kullanılmaktadır. Bu çalışma, literatürde önerilen görüntü şifreleme şemalarını değerlendirmek için bir çerçeve sunmaktadır. Görsel inceleme yerine, şifrelenmiş görüntülerin kalitesini ölçmek için korelasyon katsayısı, bilgi entropisi piksel değişim oranı gibi bir dizi kontrol parametresi kullanılmaktadır. Bu kontrol parametreleri kullanılarak Gelişmiş Şifreleme Standardı (AES) şemasının görüntü şifreleme verimliliğinin analizi yapılmış ve güvenlik seviyesi değerlendirilmiştir. Kaba kuvvet, istatistiksel ve farklı saldırılar için yapılan güvenlik analizleri, AES yönteminin görüntü şifrelemede oldukça güvenilir olduğunu ortaya koymaktadır. Yapılan testler görüntü şifreleme yönteminden bağımsız olup tüm algoritmalar için kullanılabilir.

Anahtar Kelimeler: Görüntü Şifreleme, Güvenlik Analizi, AES, Kriptografi

ACKNOWLEDGEMENTS

First of all, I would like to express my sincere thanks to my advisor, Mahmut Bağcı, who greatly contributed to increasing my knowledge and encouraged me to acquire new perspectives, for giving me the opportunity to work with him on this thesis.

Secondly, I would like to express my deepest gratitude to my managers for being considerate and supportive of me during this process.

Thirdly, I would like to express my deepest gratitude to my dear teachers Aşkın Demirağ and Barış Yalçınkaya, who supported me to complete my master's with thesis.

I would like to thank my dear wife Özge Küçükbacak for her presence, support and patience while writing my thesis.

TABLE OF CONTENTS

abstract I

ÖZET II

ACKNOWLEDGEMENTS III

TABLE OF CONTENTS IV

LIST OF FIGURES VI

LIST OF TABLES VIII

1. INTRODUCTION 1

2. CRYPTOGRAPHY 3

2.1. Classification Of Cryptographic Algorithms 5

2.2. Diffusion And Confusion 11

2.3. Randomness And Unpredictability 13

3. CRYPTANALYSIS 15

3.1. Image Encryption Schemes 18

1.3.1. Spatial Domain Based Encryption Techniques of Image 23

3.3.1.1. Chaos Based Encryption Techniques of Image: 23

3.3.1.2. DNA Based Image Encryption Techniques: 26

3.3.1.3. Cellular Automata Based Image Encryption Techniques 27

3.3.1.4. Meta-heuristics Based Image Encryption Techniques 28

3.3.1.5. Elliptic Curve and Fuzzy Based Image Encryption Techniques: 29

4. SECURITY ANALYSIS OF ENCRYPTED IMAGES 31

4.1. Image Encryption Using Aes 31

4.2. Security Analysis 37

4.2.1. Histogram Analysis 38

4.2.2. Information Entropy 39

4.2.3. Correlation Coefficient 40

4.2.4. Differential Attack 43

4.2.5. Key Space Study 45

4.2.6. Key Sensitivity 46
4.2.7. Encryption Quality 47
4.2.8. Performance Analysis 49
5. CONCLUSION 50
REFERENCES .....
APPENDICES .....

LIST OF FIGURES

Figure 1 Classification of information security techniques 2

Figure 2 The structure of a general cryptosystem. Encryption and Decryption 4

Figure 3 Cipher Block Chainig Mode in block cipher 6

Figure 4 Encryption of n bits with Stream (a) and Block (b) encryption 7

Figure 5 General structure of symmetric encryption 8

Figure 6 General structure of asymmetric encryption 9

Figure 7 Mixing and propagation, respectively, in a general cryptographic structure 11

Figure 8 A two-loop replacement displacement net 13

Figure 9 Classification of cryptanalysis 16

Figure 10 Shows the RGB color space. 19

Figure 11 Classification of image encryption techniques 23

Figure 12 Classification of image encryption techniques 25

Figure 13 DNA-based image encryption diagram 27

Figure 14 Image encryption with Cellular Automata 28

Figure 15 The work of Evolutionary Optimization in the encryption process 29

Figure 16 Image encryption using elliptic curve and chaotic system 30

Figure 17 AES input and output parameters 33

Figure 18 Block diagram of AES algorithm 35

Figure 19 AES algorithm Image input and output 36

Figure 20 Histogram comparison (a) Histogram of plain mandrill image (b) Histogram of encrypted image 38

Figure 21 Correlation graphs (a), (b), (c) relations of adjacent horizontal, vertical and diagonal pixels in mandrill imafe, (d), (e), (f) horizontal, vertical and diagonal neighboring encrypted image pixels, respectively 42

Figure 22 Key sensitivity in decryption (a) encrypted mandrill picture (b) plain mandrill picture decrypted with real key (c) meaningless picture resulting from decryption by 1-bit change in key 47

LIST OF TABLES

Table 1 Comparison of symmetric and asymmetric encryption structures.....10

Table 2 Break times with brute-force attack of different key sizes 17

Table 3. Cycle counts and key sizes for AES 34

Table 4 Entropy values of mandrill image encrypted with AES 40

Table 5 Correlation coefficients of neighboring pixels of images encrypted with AES ..43

Table 6 NPCR and UACI comparisons of chaotic image coding in the study 45

Table 7 The pixel differences of the picture encrypted with the keys that are very slightly different between them 47

Table 8 Encryption quality for image encryption with AES 48

Table 9 Performance comparisons in encryptions with different key sizes of images of different sizes.....49

1. INTRODUCTION

Multimedia encryption technology originally appeared in the 1980s, and by the mid-1990s, it had become a hot topic of research. Raw data encryption, compressed data encryption, and partial encryption are the three stages of development (Li et al. 2004).

Only a few multimedia encoding methods were standardized prior to the 1990s. The majority of multimedia data (images, videos) was transferred or stored in its unprocessed state. Multimedia encryption was primarily dependent on pixel permutation or scrambling, in which the video/image was changed in such a way that the resulting data was incomprehensible(Li et al., 2001,p.321). Gap-filling curves, for example, are used to alter image/video data, complicating the link between adjacent images/video pixels. The Eurocrypt standard is used by European television networks to encrypt signals (Saha & Majumdar, 2021). These methods are used because they have low computational complexity and cost. However, such changes in the relationship between neighboring pixels, causing confining not to work. Therefore, these encryption algorithms are only advantageous for an application that does not require compression.

With the advancement of multimedia technology in the early 1990s, JPEG, MPEG, and other picture and audio/video encoding standards such multimedia data was compressed before being saved or transferred, and was not ideal for raw data encryption.

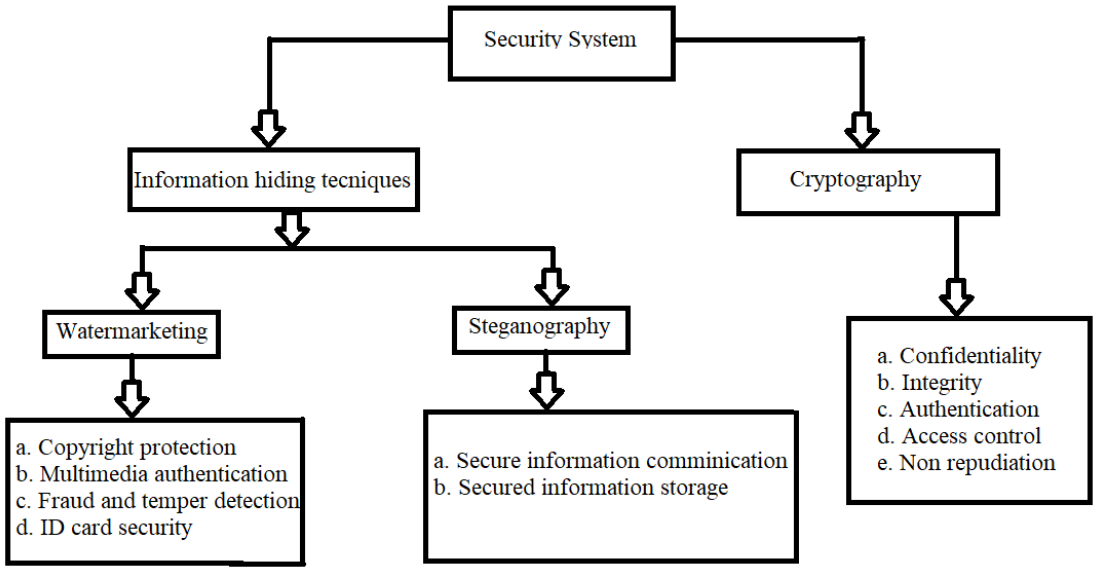
Multimedia applications required greater real-time processing after the advancement of internet technology in the late 1990s. The size of the final encrypted file is reduced and the encryption efficiency is boosted by encrypting only particular parts of the data. Due to the fast proliferation of computer networks, information security needs have become critical to safeguard privacy and confidentiality. Accordingly, encryption and decryption algorithms such as the Advanced Encryption Standard (AES) and Data

Encryption Standard (DES) have been introduced, and they have been widely used to encrypt files containing large amounts of data.

In general, security systems are based on two main components that are data hiding and cryptography. A detailed classification of these components are shown in Figure 1. In this thesis, the robustness of information security systems are tested from the perspective of image encryption techniques, and the reliability of the AES method is demonstrated.

Figure 1

Classification of information security techniques



2. CRYPTOGRAPHY

Cryptography is a field of study that adopts the principle of achieving goals such as data integrity, confidentiality, authentication and non-repudiation (Bruce, 1996; Kocarev, 2001; Stinson, 2002). Cryptanalysis is a field of study that deals with cracking existing cryptosystems. Cryptography and cryptanalysis, which are connected to each other, are sub-branches of the science of cryptology. Cryptology is the whole of skills and applications based on mathematically difficult complications, which approve two or more communicating parties to exchange information security.

Confidentiality of information must be protected because an unauthorized end (client) can access information that should be confidential by connecting to the system communication network. However, confidentiality is not the only restricted definition of secure transmission. Even if the sending and receiving systems authenticate each other, they want to make sure that the transmitted content has not been altered by malicious people. In order to ensure data integrity, when information is changed in an unauthorized way, the communicating parties detect this situation.

Both the sending and receiving parties need to certify the identity of the other party participating in the communication. Verifying the other side means finding out whether they are actually the side they claim to be. With the authentication methods, the communicating parties can mutually determine each other's identities, and it can also be determined whether the message comes from the real source. Non-repudiation is the ability of the receiving end system to confirm to everyone that the sender actually sent the message.

Main purpose of encryption is to obfuscate the message so that only the authorized party can restore it to its original form. An encryption system can be expressed as in (1)

with plaintext (P), ciphertext (C), keys (K_e , K_d), encryption (E), and decryption (D) transformations. It is encrypted using a plain text encryption key.

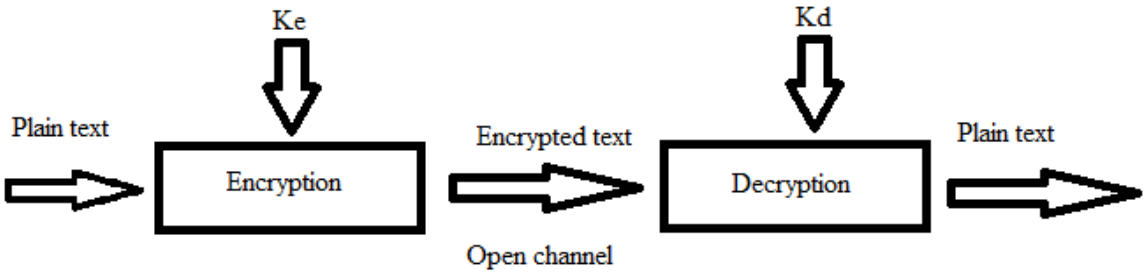
$$C = E_{K_e}(P) \quad (1)$$

Here E represents the encryption function and K_e represents the encryption key. The resulting C ciphertext is converted back to the original P plaintext as in (2) using the decryption function D and the decryption key K_d . Here, a plain text can only be converted to its original text only with the decryption key. A general encryption structure is shown in Figure 2.

$$P = D_{K_d}(C) \quad (2)$$

Figure 2

The structure of a general cryptosystem. Encryption and Decryption



2.1. Classification Of Cryptographic Algorithms

Modern encryption algorithms can be classified differently according to how they do the encryption and whether the keys are private or open (Mao et al. 2004; Li ,2003).

Block ciphers and stream ciphers are two types of encryption algorithms that are categorised based on their structure. In block ciphers, a fixed-size plaintext block is converted into a ciphertext block of the same size. Block sizes differ in algorithms. Large-block cipher algorithms have greater security, but since they have a more complex cipher, their performance is inversely proportional to the block size. Modern block cipher algorithms should have the following features (Robertson, 1980).

i. Variable key size
ii. Mixing processes providing non-linearity
iii. Variable plaintext and ciphertext blocks
iv. Variable cycle numbers

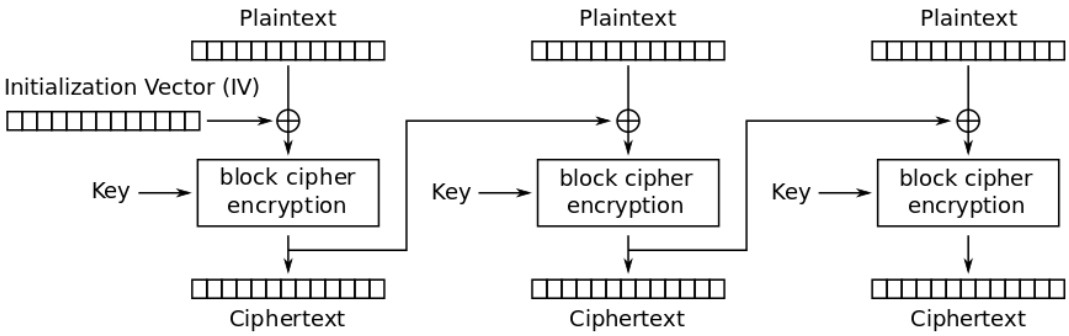
In block ciphers, blocks are encrypted separately or in conjunction with each other, depending on the format of the algorithm. In block cipher algorithms, the length of the key must be long enough against the most basic attack brute force attacks. For example, the DES algorithm has a 56-bit encryption key, while the AES algorithm covers this weakness of DES and has 128, 192, 256-bit key options.

There are several different encryption modes in block cipher. One of them is Electronic Code Book (ECB) mode in which blocks with the same values are converted to identical ciphertext blocks. This indicates that it is not secure to encrypt plaintexts containing similar blocks in this mode. To overcome this low security, different block cipher modes such as Cipher Block Chaining (CBC), Cipher FeedBack (CFB) and Output FeedBack (OFB) are used. CBC mode, each plaintext block is XORed with the previous ciphertext block. This eliminates the lack of security, as the same plaintext blocks in

electronic codebook mode produce the same ciphertext blocks. The same effect can be achieved with the password feedback mode. Electronic codebook mode works better than other block cipher modes, since the computational complexity is lower than other modes and it has a structure suitable for parallelization. All modes have advantages and disadvantages according to the intended use of encryption. The working principle of the Cipher Block Chaining mode is shown in.

Figure 3

Cipher Block Chainig Mode in block cipher



Note. (Seo et al, 2014).

In stream encryption, bits or smaller units of plaintext are encrypted instead of large blocks per unit time. Therefore, they are faster than block ciphers. Generally, in stream cryptography, the key is generated as a bit string by cryptographic pseudo-random number generators (CPRNG) and then logically combined with plain text. This operation is usually performed with operators such as XOR, XNOR, add, mod (Mao et al.,2004 ;Mao & Wu , 2006). There is no standardized stream encryption algorithm available today.

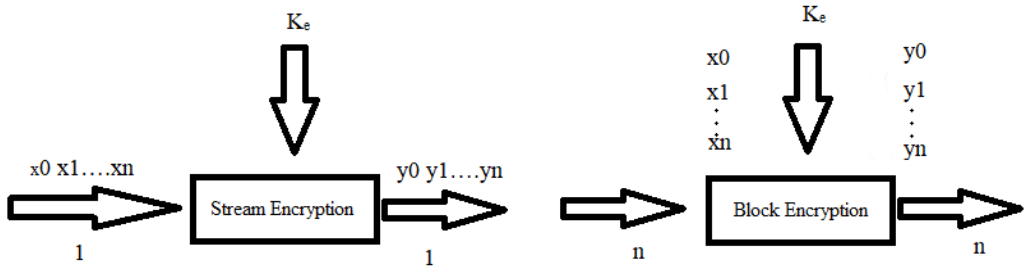
Stream ciphers are a type of symmetric encryption algorithm that is extremely important The Vernam (One-Time-Pad) cipher, which XOR encrypts plain text along a random key, is the fundamental design inspiration (Li et al., 2001,p.321). The key in the

Vernam encryption is a real random sequence shared by both the sender and recipient, and it can only be used once. This is a major issue when it comes to key creation and distribution. Figure 4 shows the structrural difference between block and stream ciphers. Stream encryption algorithms have the fallowing features (Li et al., 2001,p.321).

i. They do not provide very high security
ii. Security depends on crytographic pseudo-random number generators
iii. CPRNG should genarate unpredictable sequences
iv. They are very fast

Figure 4

Encryption of n bits with Stream (a) and Block (b) encryption

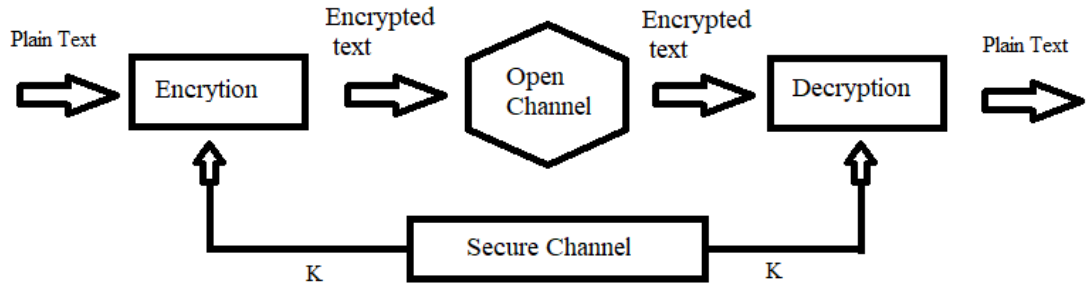


Encryption and decryption are performed using two keys, K_e and K_d. The algorithm is known as secret key encryption or symmetric encryption when their keys are identical. The key must be supplied to the recipient over a secure channel over the communication channel in symmetric encryption. The system is known as public-key or asymmetric encryption when it uses distinct keys for encryption and decryption. In asymmetric encryption, everyone knows the encryption key K_e, but the decryption key K_d is kept secret.

In Figure 5, the sender encrypts the plain text and sends it to the receiver using the encryption key. The receiver uses the same key to decode the plain text.

Figure 5

General structure of symmetric encryption



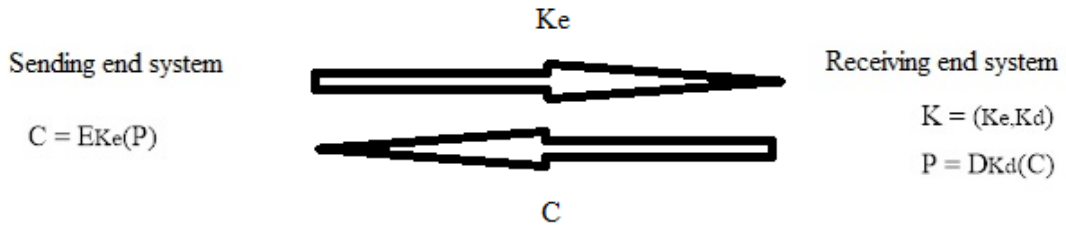
Symmetric encryption deals with authentication, not just encryption. One of the methods used in authentication is Message Authentication Codes (MACs). Message verification codes are used for similar purposes to digital signatures. However, for authentication with MACs, single-key encryption is used instead of double-key encryption, i.e. only the agreed-upon public key.

In other words, only the desired user can perform the authentication in this way. It is not feasible to validate the signature with the sender's public key, as it is with a digital signature. The main problem in private-key cryptography is that the sender and receiver allow on a public key, preventing it from falling into the hands of a third party. For this, he needs a method that will allow the two parties to communicate without fear of being listened to. Spam should block both user-accepted keys used in private key encryption. Generating, transmitting and storing keys is known as key management.

In asymmetric encryption, different keys are utilized for encryption and decoding. Traditional cryptography's key distribution problem is solved by public-key cryptography (Kusters &Tuengertsbalg, 2009,p.298). In contrast to symmetric encryption techniques that employ a single key, public-key cryptography envisions the asymmetric usage of two independent keys. The public key is one of these keys, while the private key is the other. The public key is shared by one party with other parties with whom it will communicate in an encrypted manner, i.e. it is not private and accessible to all. The owner's private key is kept private so that only he or she has access to it. There is a functional relationship between these keys. Figure 6 depicts asymmetric encryption.

Figure 6

General structure of asymmetric encryption



The fundamental advantage of asymetric cryptography is that it provides more security because the private key is never transferred. It has, in particular, provided a solution to the problem of secret key security. In structures with a private key, on the other hand, the secret key must be communicated manually or over communication channels since the key used in encryption and decryption is the same. This elevates the possibility of the private key being accessed by unauthorized individuals.

Another significant benefit of public-key algorithms is the ability to create indisputable digital signatures. Confidential information must be shared over an open network or through a third party when employing secret-key cryptography for

authentication. In this instance, one of the parties may gather evidence that the key is being misused by the others. This is not the case in public-key arrangements, where everyone is responsible for their own digital signature. Non-repudiation is the name for this characteristic.

The encryption speed is one of the drawbacks of employing public-key architectures. The majority of private-key structures are faster than the majority of public-key structures. Using the two structures together is the safest and fastest option. Public-key algorithms should not be used to replace private-key structures; rather, they should be used to enhance their security. Public-key cryptography, for example, is used to send private keys via public networks. A comparison of symmetric and asymmetric encryption techniques is shown in Table 1.

Table 1

Comparison of symmetric and asymmetric encryption structures

Symmetric Encryption	**Asymmetric Encryption**
The encryption and decryption keys are the same.	The encryption and decryption keys are not the same.
Sharing the key is required between sender and receiver.	The end parties must have one of two complementary keys.
The algorithm and some plain text knowledge shouldn't be enough to determine the secret key.	The algorithm and encryption key information should be sufficient to determine the decryption key.
The private key should not be stored securely.	The decryption key must be stored securely.

2.2. Diffusion And Confusion

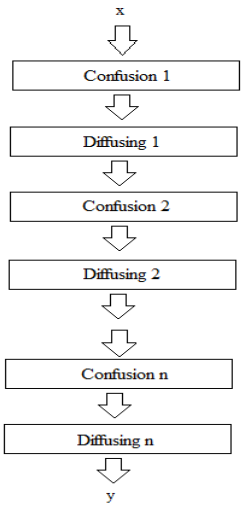
Cloude Shannon emphasized that strong encryption algorithms should be based on confusion and diffusion techniques (Shannon, 1949). Confusion occurs when the distinction between key and plain text is blurred. In general, substitution techniques are used to perform mixing. The purpose of hashing is to remove the statistical correlation between plaintext blocks and ciphertext blocks.

Diffusion effect is when a single symbol of plaintext acts on many symbols in ciphertext. The purpose here is to hide the statistical properties of the plain text. Although the cryptanalyst can analyze similar plaintext and ciphertext pairs, he cannot guess about the plaintext corresponding to the ciphertext y^1.

Today, all block cipher systems apply operations in data, respectively, as in Figure 7. Systems that perform only mixing and propagation are not safe. By performing both operations, very secure encryption systems can be established.

Figure 7

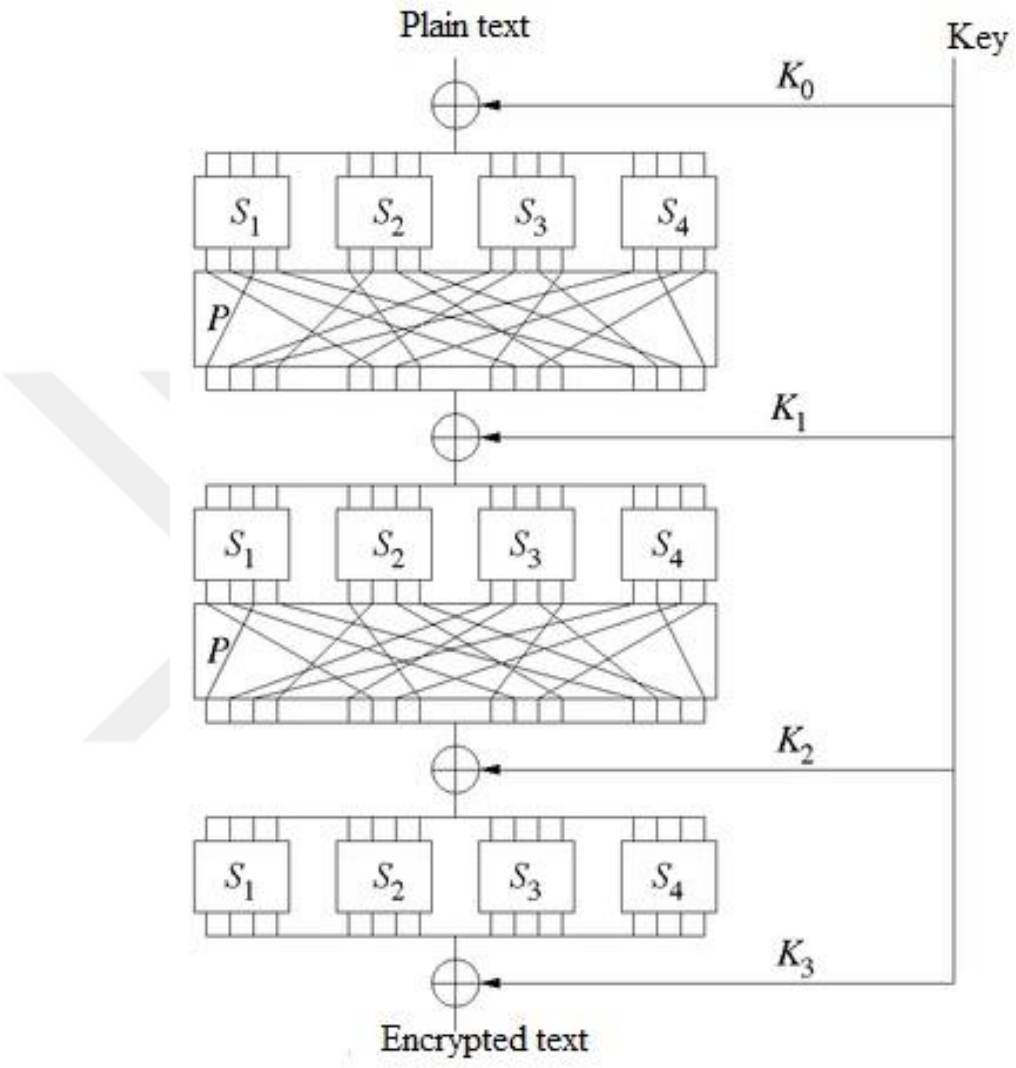
Mixing and propagation, respectively, in a general cryptographic structure



A Substitution - Permutation Network is a simple approach to achieve both mixing and diffusion. Substitution networks are inherent in some encryption algorithms, such as AES. Substitution-Boxes and Permutation-Boxes are applied to these inputs in more than one loop, with the key and a plain text block as input. In this way, mixing and spreading stages are provided (Belazi et al., 2016). A two-loop substitution displacement network is shown in Figure 8.

Figure 8

A two-loop replacement displacement net



Note. (Belazi et al., 2016).

2.3. Randomness And Unpredictability

Randomness is a series of randomly generated numbers that do not have any functional relationship between them. Unpredictability, on the other hand, is the inability to predict the values before and after any n-position value in any number sequence in cryptography. There are differences between randomness and unpredictability, which are

highly correlated. A completely random process is also unpredictable. However, the reverse is not true. Unpredictable processes may not be truly random.

True random number generators are systems whose outputs cannot be reproduced in practice. For example, when we toss a coin 100 times by accepting one side as 0 and the other side as 1 and recording the result as 100 bits, it is almost impossible to obtain the same bit sequence again in the same way. Generating the same sequence is possible with a very low probability of 1/2^100. Because the generated array elements are functionally independent of each other.

Pseudo-random number generators are not truly random. Because they are completely deterministic, they can be recalculated. The most important and general feature of the outputs of these generators is their statistical approximation to the real random number sequence. Outside of cryptography, there are many areas where pseudo-random numbers are used. They are utilized in simulation, software testing and many more areas that need random data entry. One of the reasons for the rand() function in American National Standards Institute (ANSI) for the C programming language declarations is the widespread use of pseudo-random numbers.

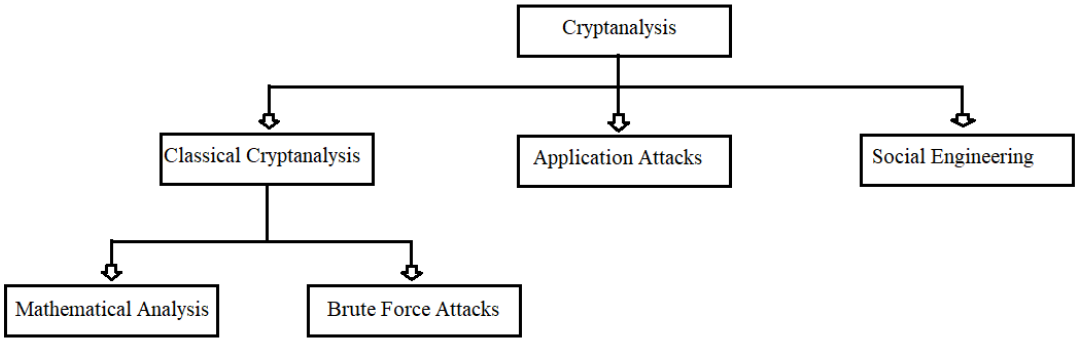
Cryptographically secure pseudo-random number generators are a subset of pseudo-random number generators. Generators of this particular type produce unpredictable outputs. This means that given the n symbol of the escape sequence, the next or previous symbol sequences of this substring cannot be calculated. Cryptographic secure random number generators are used in applications such as key generation and digital signature in cryptography.

3. CRYPTANALYSIS

Cryptanalysis is the science of cracking cryptographic systems. Cryptanalysis is of great importance for modern cryptosystems. In a situation where encryption methods are not analyzed, we cannot know whether algorithms are secure or not. Security analysis is the only way to be sure that cryptographic systems are secure. The encryption approach is designed to keep the plaintext hidden from an attacker, whereas cryptanalysis is the science of recovering the plaintext without the key. Cryptology is a broad term that encompasses both cryptography and cryptanalysis. During the cryptanalysis process, the cryptanalyst is assumed to understand the subtleties of cryptosystems and to have thorough understanding of an encryption method (Zeng et al., 2006; Zhou et al.,2008). An attack is defined as the discovery of a flaw in the ciphertext, encryption mechanism, or key management scheme. There are many methods for cracking encryption systems. The general classification of these methods is shown in Figure 9.

Figure 9

Classification of cryptanalysis



The goal of traditional cryptanalysis is to extract the plaintext from the ciphertext or the key from the ciphertext. It is tried to obtain information by evaluating the data with mathematical analysis and examining the local structure of the encryption method.

Without knowledge of plaintext and keys, the cryptanalyst has some knowledge about ciphertext. In this case, the predicted plain texts are tried to be obtained. To obtain the decryption key or plaintext, an attacker attempts to crack the algorithm or simply see the ciphertext. If an attacker has access to the plaintext or key associated with a cryptosystem, it becomes insecure. The attack's main goal is to get access to the plaintext and/or secret key. The known plaintext attack is based on a known number of plaintext ciphertext pairs. In this case, the appropriate plaintext is encrypted with different keys and the ciphertext is tried to be obtained and the key can be guessed.

The chosen plaintext attack aims to obtain the unknown key. The cryptanalyst tries to obtain the key by encrypting the plaintexts with any key and comparing the obtained ciphertexts with the plaintexts. Differential cryptanalysis, which may be used against block ciphers and hash functions, is the most well-known example of this attack (Heys , 2001).

Brute Force attacks attempt to crack the password by using all available keys in the key space. Today, with the existence of very fast computers, systems with larger key sizes are being developed to resist brute force attacks. This method is used by cryptanalysts to test all possible keys in the finite key space one by one to determine if the plaintext that goes with them makes sense. In order to succeed, 50% of all viable keys are attempted on average, yet a brute force attack takes a lot of time and effort. Due to the tremendous complexity, a brute force approach may not be practicable. Table 2 displays the amount of time required for various key sizes (El-Fishawy & Zaid ,2007).

Table 2

Break times with brute-force attack of different key sizes

Key Size (bits)	**Number of Alternative Keys**	**Time required at 10^6 Decryption/µs**
32	2^{32}	2.15 milliseconds
56	2^{56}	10 hours
128	2^{128}	5.4 x 10^{18} years
256	2^{256}	5.9 x 10^{30} years

In application attacks, the secret key is tried to be obtained by using the hardware where the algorithm is runs. For example, the private key can be obtained by measuring the current drawn by the central processing unit while operating on the secret key.

In Social Engineering, the secret key can be obtained by people through blackmail and deception.

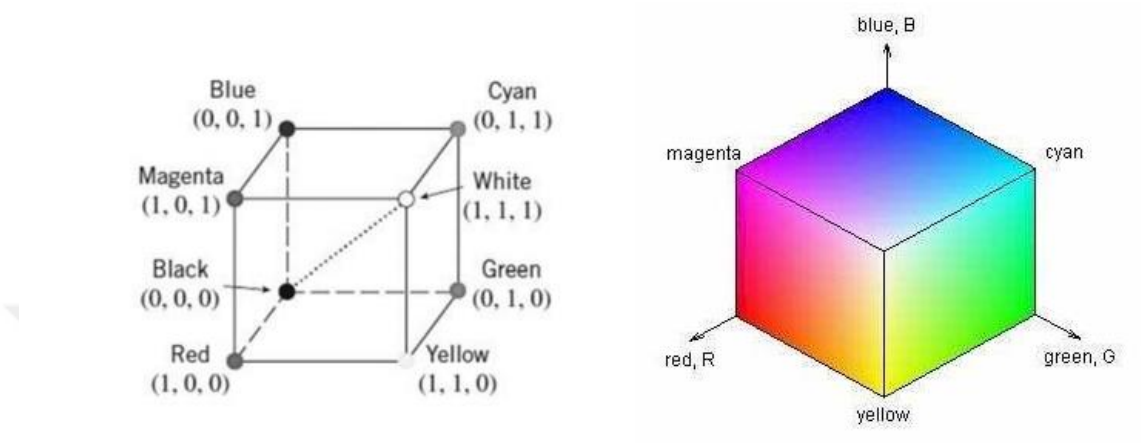
3.1. Image Encryption Schemes

Image Processing is a approach to establish and digitize the image and perform some operations, to obtain a specific image or to extract some useful information from it. It is like a signal processor, where the input is a photograph made up of an image or attributes associated with that image. These days, studies on image processing are increasing rapidly. It also frames the central examination area within the building disciplines. Many disciplines related to these areas have also been a part of the studies.

A picture is just a two-dimensional object. That is characterized by the scientific capacity f (x, y), where x and y are two plane dimensions, horizontal and vertical directions.. The f(x,y) estimation at any given time gives the pixel value by a picture running around 0 and 255. The dimensions of the photograph are really the measure of this two-dimensional set. These images can be monochrome (two-tone), grayscale, or shaded, depending on the allowable strength levels of each pixel. A monochrome image can have only one pixel value (0 or 1), while a grayscale image can have 8-bit pixels and a shaded image can have 24-bit planes (8-bit each for the Red, Green, and Blue channels).

Figure 10

Shows the RGB color space.



The word pixel is derived from the combination of the first syllables of the English word Picture Element. The smallest controllable points on the screen, which are the basis of all digital images. A pixel consists of a mixture of red, green and blue colors. The more numerous and small pixels the image consists of, the better quality and clearer it will be.

The measure of the distance between the pixels that make up an image is called resolution. The maximum number of pixels that may be shown on a screen at any given moment. The greater of the image quality, the higher of the resolution (i.e., the number of pixels in the image). Images that are physically the similar size may not be of the same resolution. Some of the commonly used image formats are:

Joint Photographic Group (JPG) is a widely used image format on the Internet, developed for professional photographers, in which image processing programs can compress high-megabyte files and save them on disk. It is used in photographic imaging because it contains true color values. Joint Photographic Experts Group compression method is

among the lossy formats because it is a format that finds and discards the details that are not necessary for the image to be perceived and compresses the file. This balance needs to be well maintained, as there is a connection between the lost details and the compression ratio. More compression means more data loss, less compression means larger file size.

Bitmap is an image file format developed by Windows and Microsoft by modifying the Picture Exchange format, which keeps the properties of images without any compression. Because it does not compress, the file size is very large compared to other image types. The BMP format can contain pixel depths ranging from 1-24 bits.

Graphics Interchange Format (GIF) is developed by CompuServe. Along with JPEG, it is the most widely used digital image storage formats in the computer world. There is no true color support. It provides 8-bit color support up to 256 colors. This image format is generally used for storing graphics because it contains less color. It is used to create pixel-based animated images on web pages.

Portable Network Graphics (PNG) is an image format that performs lossless compression, can be edited in image processing software programs and is used for displaying images on the web. It uses different filtering algorithms for compression. It is upgraded as a non-patentable version. PNG pictures can be 24-bit and have a transparent backdrop with no jagged edges. With the developing and widespread hardware technology, the GIF format began to fall short.

Tag Image File Format (TIFF) is a file format that may be customized and adapted. Using tags in the file header, it may contain numerous pictures and data contained in a single file.Tags can specify the underlying geometry of the image, such as dimensions, or which compression preference is used. Multi-page documents can be saved as a single TIFF file rather than as distinct files because the TIFF format allows multiple pages. For example, a

TIFF file can contain compressed images in both JPEG format (lossy) and PackBits format (lossless). TIFF files can be used as good image archives thanks to their lossless image storage features. Because unlike standard JPEG files, a losslessly TIFF file can be edited and resaved without loss of image quality.

Computer networks carry not simply text, but also a variety of other sorts of online data. In recent years, multimedia security has evolved as a topic of research for the protection and growth of digital media against various attack scenarios. Previously, it allowed for flawless copying of digital media material as well as near-seamless data alteration. New sorts of security assaults have emerged that have never been seen before in the industry. The transition from analog to digital multimedia has had an impact on artists, publishers, copyright holders, and consumers (Qiao, 2007, p.247 ; Stallings, 1999).

Image data, unlike text information, has distinguishing characteristics such as large size and pixel correlation that make it harder to use typical encryption methods and slow down encryption. Real-time processing, image format integrity and data reduction for transmission are common properties in image applications. Real-time image processing systems have significant challenges in achieving these goals while maintaining high security and quality. There are key differences between image and text data encryption.

Once the ciphertext is generated, it should be completely losslessly decrypted into the main plaintext. All the same, the encrypted image might be lossy decrypted to the main plain image.

i. Text data consists of word strings. Block or stream ciphers can be used to encrypt them directly. Digital pictures, on the other hand, are frequently represented as two-dimensional (2D) arrays. Before applying various traditional encryption

approaches to safeguard stored 2D data arrays using text processing algorithms, they must first be transformed to 1D arrays.

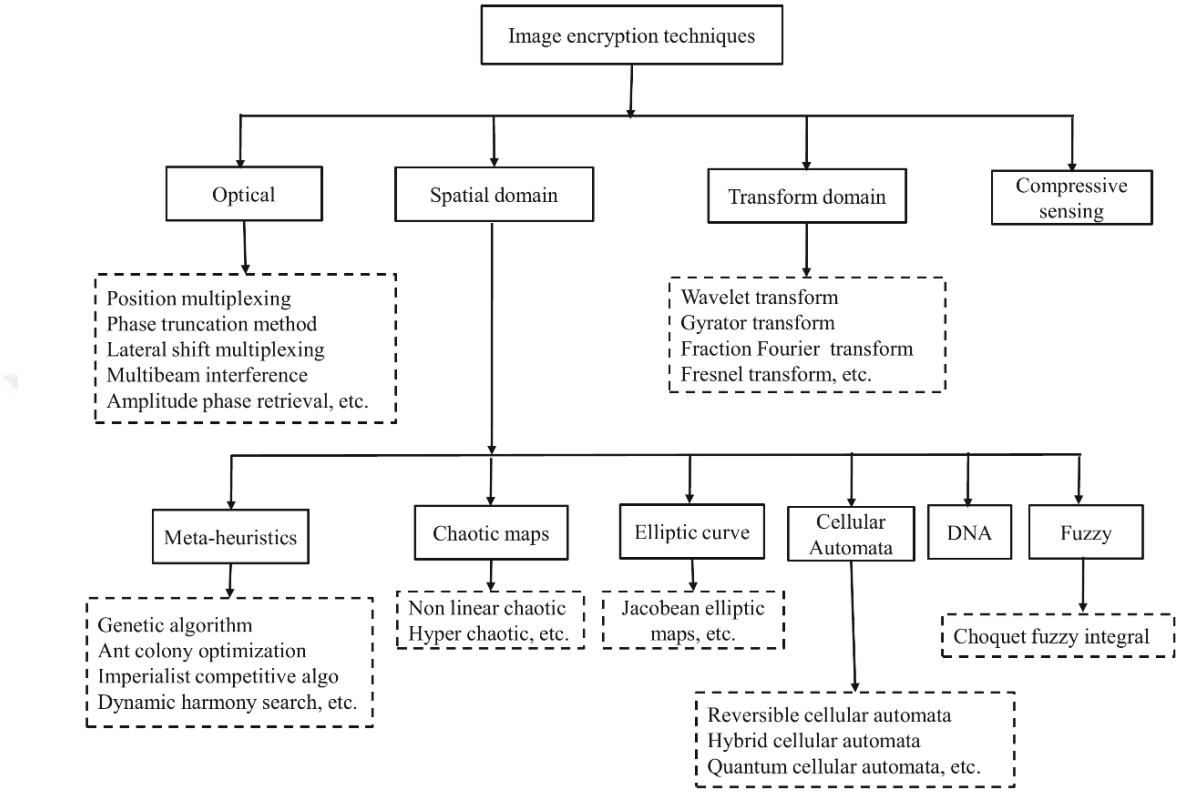
ii. Because an image's footprint is so huge, it's sometimes wasteful to encrypt or decode it directly. To save both storage space and transmission time, one of the best approaches is to encrypt / decrypt the information after compressing the image.

In this section, the design of cryptographic algorithms are presented. It is offered an overview of the many issues that emerge while selecting, creating, and assessing a cryptographic method. Diffusion-based encryption methods and their operating modes are addressed together.

Optical, spatial, transform domain, and compression sensing are the four kinds of image encryption technologies. The categorization of photo encryption algorithms is shown in Figure 11. Comparison between these techniques of image encryption are performed using certain performance and security analysis metrics. In this study, we will examine Spatial Based Image Encryption Techniques.

Figure 11

Classification of image encryption techniques



Note. (Khan &Shah, 2014, p.20)

1.3.1. Spatial Domain Based Encryption Techniques of Image

The picture plane, which is the direct manipulation of pixels, is referred to as the Spatial Domain. Approaches that are applied directly to these pixels are known as spatial domain based techniques. The following sections summarize various Spatial Domain Based image encryption algorithms.

3.3.1.1. Chaos Based Encryption Techniques of Image: The chaos approach is a field of study that describes the behavior of dynamical systems that are highly sensitive to initial conditions. Time-dependent periodic changes are not expected to occur in chaotic

systems. The original initial conditions make chaotic systems inherently unpredictable. (Gu & Ling, 2014). A discrete chaotic system is defined by an iterative feedback function f (chaotic map) and state space I as in (3).

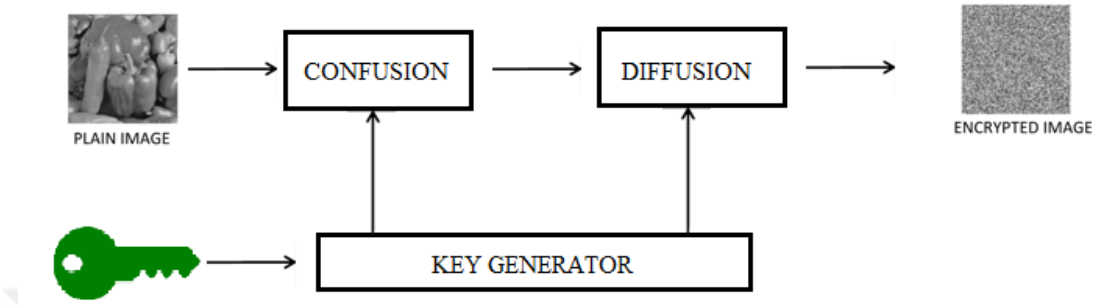
$$x_{n+1} = f(x_n) \tag{3}$$

Here $x_n \in I$ represents the system state in discrete time. Chaotic systems are controlled by control parameters. In the real world, these parameters may not remain constant throughout the trajectory. The output of the chaotic generator reveals a chaotic sequence, showing the time-varying the chaotic system's current condition. Chaotic encryption schemes' characteristics depend on beginning circumstances, determinism, and ergodicity.

A basic chaos-based picture encryption approach is shown in Figure 12. Chaotic systems process an image in two aspects to apply diffusion and permutation (or confusion). Cryptosystem engineers utilize permutation and diffusion in the same real time system to achieve considerable computational security.

Figure 12

Classification of image encryption techniques



Permutation or Confusion randomly shuffles the pixels of the flat image into random sequences produced by chaotic systems. At diffusion stage, scrambled image is decomposed into several blocks. Chaotic system uses its preconditions from the previous block to update its state. The key sequence is based on the given image(Huyang & Ye, 2014).

Diffusion must be evaluated to determine the unpredictability of the encryption method. The connection between the encrypted image and the original image is quite complex and unpredictable if the algorithm used is robustly designed. To measure the diffusion of any algorithm, one bit is changed in the plain image, and the difference between the encrypted image created from the original plain image and the encrypted image acquired from the modified one is computed (Umamageswari & Suresh , 2013 p.1118).

Chaotic systems can be employed as safe cryptographically random number generators because of their sensitivity to beginning circumstances and topological transitivity. Because of these properties, they have been widely used in generating random numbers in

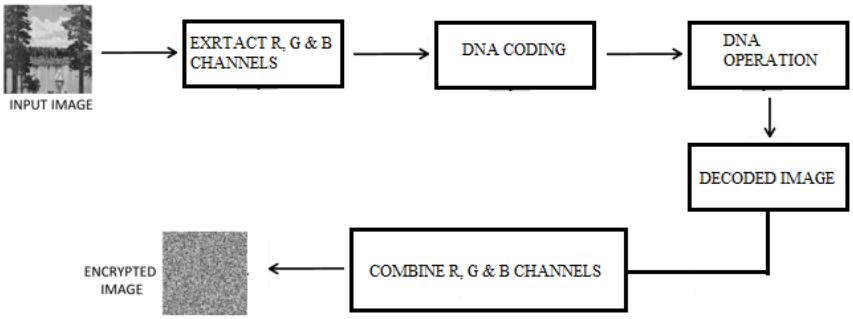
cryptography. Chaotic systems are generally continuous systems and are identified in the set of real numbers or complex numbers.

3.3.1.2. DNA Based Image Encryption Techniques: In recent years, DNA based technologies find application in medical systems, information technology and all other industries. DNA molecules contain genetic code that stores data and may switch between genetic codes. Researchers have developed a simulation environment for studying biological DNA technology, dubbed pseudo-DNA technology. This concept led to the use of DNA technology to encrypt data (Zhang et al. 2014, p.188).

The DNA-based picture coding process is depicted in Figure 13 as a block diagram. The picture is first split into three color channels: red (R), green (G), and blue (B). These channels are transmitted into binary matrices and these matrices are subsequently encoded using DNA coding principles To obscure similarities between pixel values, DNA techniques are applied to coded matrices. After that, decoding methods are used to transform them back to binary matrices. Finally, the three color channels are merged to create a color image that has been encoded.

Figure 13

DNA-based image encryption diagram

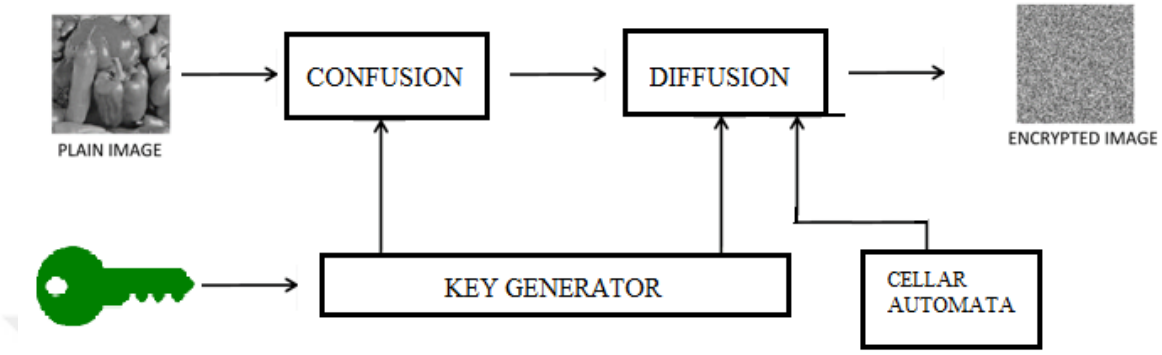


Parallel processing, relatively low power consumption, and vast storage are the key reasons for employing DNA in picture encryption (Wang et al., 2015, p. 129).

3.3.1.3. Cellular Automata Based Image Encryption Techniques: Cellular Automata (CA) is made up of cells that are arranged in a grid and have various structural types. These structures emerge over a finite number of time steps, following rules given by the states of nearby cells. Cells evolve complicated behavior by basic logic computations with faux randomness. Developers frequently employ reversible CA to implement block cipher techniques (Wang & Luan, 2013). The huge quantity of rule space and parallelism are the key advantages of CA in encryption (Mohamed , 2014 p. 88). It also offers great security and quick operation (Enayatifar et al., 2015 p. 38). Figure 1.14 shows the encryption of image diagram using cellular automata.

Figure 14

Image encryption with Cellular Automata

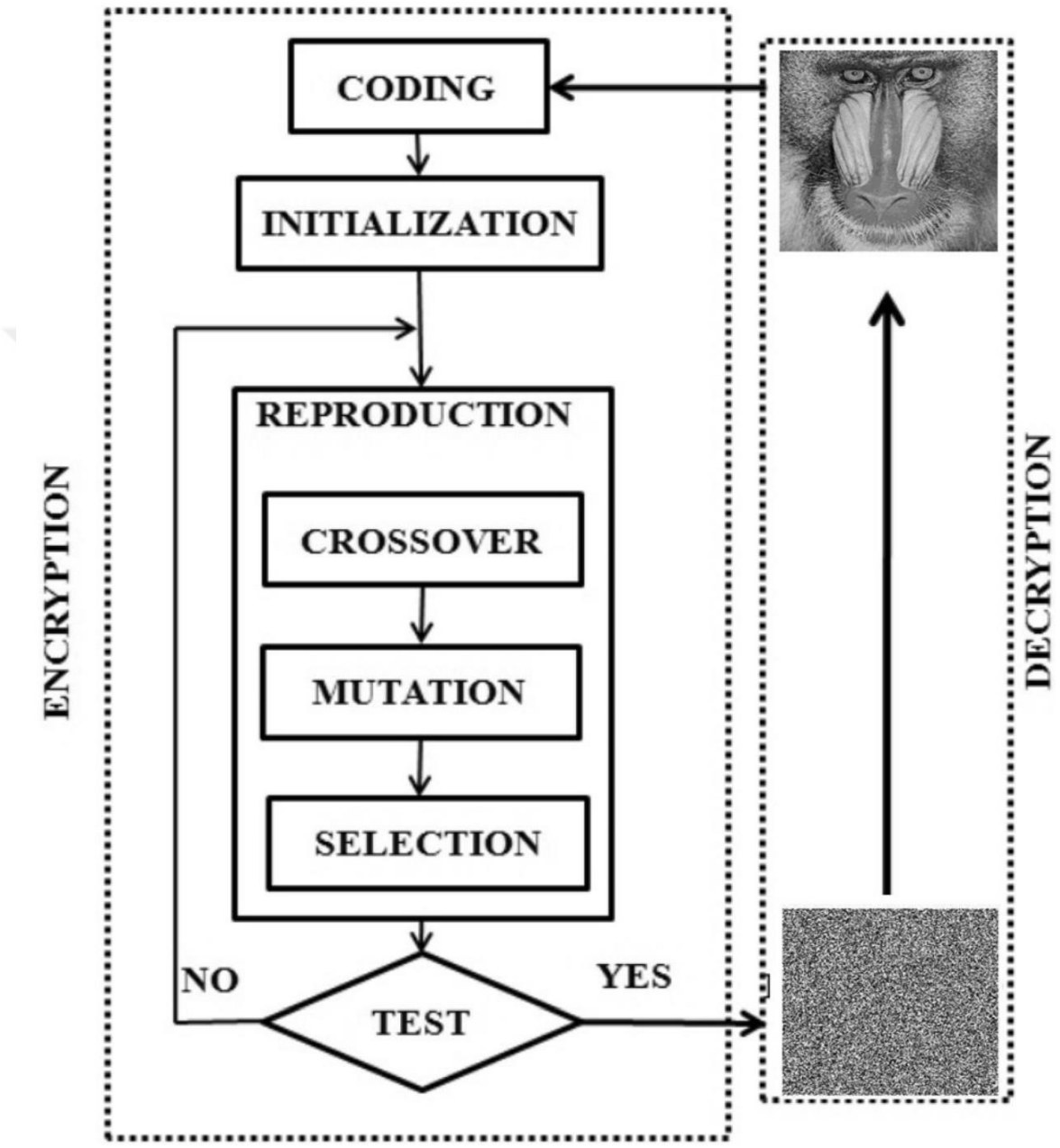


3.3.1.4. Meta-heuristics Based Image Encryption Techniques: In non-deterministic polynomial-time (NP) hard problem optimization, meta-heuristic techniques play an important role. These approaches are utilized via optimizing the encryption process's fixed parameters. Because evolutionary algorithms (EAs) are based on population behavior, they can produce a large number of feasible solutions in a single evolutionary cycle (Enayatifar et al., 2014 p. 87).

The EA's role in the encryption process is depicted in Figure 15. EA is derived from the generation of random samples from a certain population, i.e. the issue domain. The EA identifies the best examples among the provided examples based on the relevant goal function. Mutation, crossover, and recombination are the three primary operators in EA. These operators are used to determine the optimal solution from the available options. The EA provides the optimal option if the halting requirements are satisfied. Otherwise, the entire procedure is iterated to identify appropriate answers (Akdag et al.,2015 p.54).

Figure 15

The work of Evolutionary Optimization in the encryption process



Note. (Enayatifar et al., 2014)

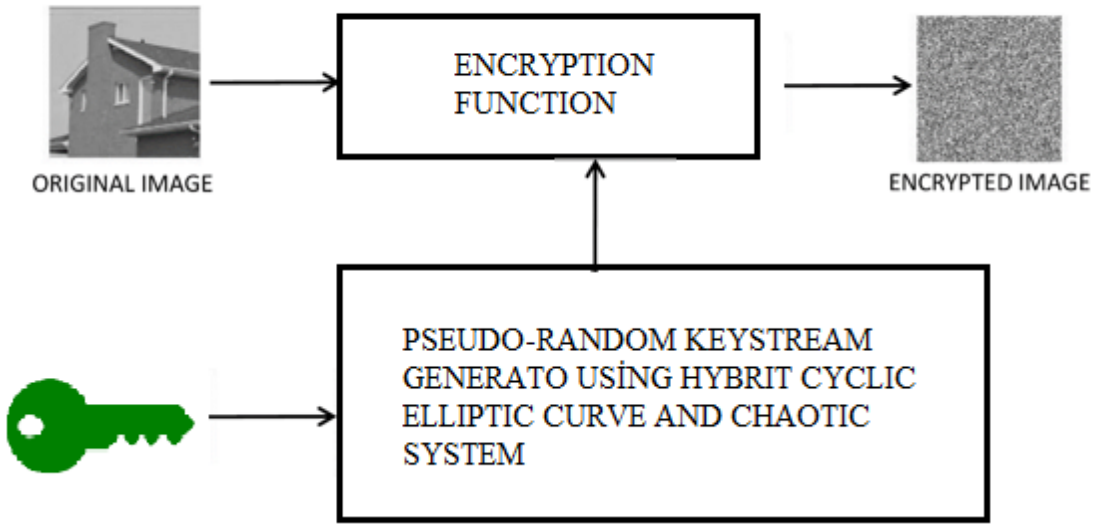
3.3.1.5. Elliptic Curve and Fuzzy Based Image Encryption Techniques: Elliptic curves are mostly based on algebraic curve features. The elliptic curve is used by Koblitz and Miller to build a public key encryption system. The short key size and reduced

computation time arethe advantages of elliptic curve encryption(Foruzan & Mukhopadhyay , 2011).

The application of the elliptic curve on an image is shown in Figure 16. Using the chaotic system and the cyclic elliptic curve, a pseudo-random key sequence is generated. It consists of two steps: first, a chaotic system is used to generate the first key, which is then combined with a pseudo-random bit sequence. An picture is deconstructed into a binary data string at first. This information is concealed using a random key. The key is generated through the hybridization of the elliptic curve and the chaotic system.

Figure 16

Image encryption using elliptic curve and chaotic system



4. SECURITY ANALYSIS OF ENCRYPTED IMAGES

The pixel values of the original image are changed when the encryption technique is applied to it. Using a strong encryption system that makes these modifications occasionally, the difference in pixel values between the original and encrypted photographs can be maximized. In order to get a decent encrypted image, it is also necessary to get one that is fully made up of random patterns that don't reveal any of the attributes of the original image. The encrypted image must be distinguishable from the original. The encrypted image should bear only a passing similarity to the original (Kuo, 1993, p. 347).

Visual examination is one of the most significant requirements for evaluating an encrypted picture. The more hidden image properties results in better encryption algorithm. Unfortunately, visual inspection of the content of the image alone is not enough to claim that it is fully obscured. Therefore, additional criteria are required to quantitatively evaluate the degree of encryption (Yeung et al., 2010, p. 104).

A two-dimensional vector array can be used to represent a picture. Two-dimensional data may be considered as a one-dimensional textual bitstream for picture encryption, and any standard encryption algorithm can be utilized. In this study, image encryption is performed with AES, which is a traditional encryption method, and security analyzes are performed on the encrypted image.

4.1. Image Encryption Using Aes

The Advanced Encryption Standard (AES) has become the most widely used symmetric encryption standard. Despite the term "Standard" in its name pertains largely to US government processes, the AES block cipher is required in numerous industry

standards and is used in many commercial systems (Fips, 2001). Commercial standards that utilise AES include the Internet security standard IPsec, TLS, the Wi-Fi encryption standard IEEE 802.11i, the secure shell network protocol SSH (Secure Shell), Internet telephony Skype, and a variety of security solutions throughout the world. The most well-known attack against AES is brute force.

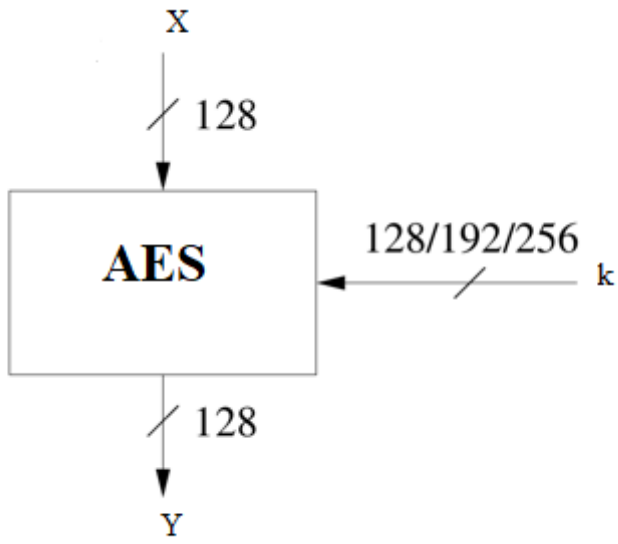
The National Institute of Standards and Technology (NIST) requested proposals for a new Advanced Encryption Standard in 1997 (AES). Unlike DES, the AES algorithm is chosen in an open procedure controlled by NIST. In the three rounds of AES evaluation that followed, NIST and the international scientific community argued the pros and cons of the proposed ciphers, which are pared down by the number of feasible alternatives. In 2001, NIST declared Rijndael to be the replacement for AES, and it is published as the final standard (FIPS PUB 197). Rijndael is established by two adolescent Belgian cryptographers.

It is expected that the AES algorithm will also play an active role in strategic applications. Because it can be implemented in hardware enables the encryption algorithm to take place in strategic communication equipment as well. It is possible to develop applications that require high speed on hardware by using the AES algorithm. Encryption and decryption are secure with AES encryption used in many highly sensitive applications such as image encryption, ATM networks, Confidential Collaboration Documents, Personal Storage Devices, Smart Cards, Government Documents, Personal Information Protection and other applications. In the AES selection process, NIST's performance criteria are determined as high speed and low RAM requirement. Designed with these criteria in mind, AES works with high performance on many different hardware.

The Rijndael block cipher is roughly comparable to the AES encryption technique. A Rijndael block and key can have a size of 128, 192, or 256 bits. Only a 128-bit block size is required under the AES standard. As a consequence, with a block length of only 128 bits, the AES algorithm is known as Rijndael. Figure 17 depicts the input and output parameters. In this study, we'll look at a version of the traditional Rijndael with a block length of only 128 bits.

Figure 17

AES input and output parameters



As previously stated, Rijndael is required by NIST to support all three key lengths. The cipher's internal cycles are a function of the key length as specified in Table 3.

Table 3

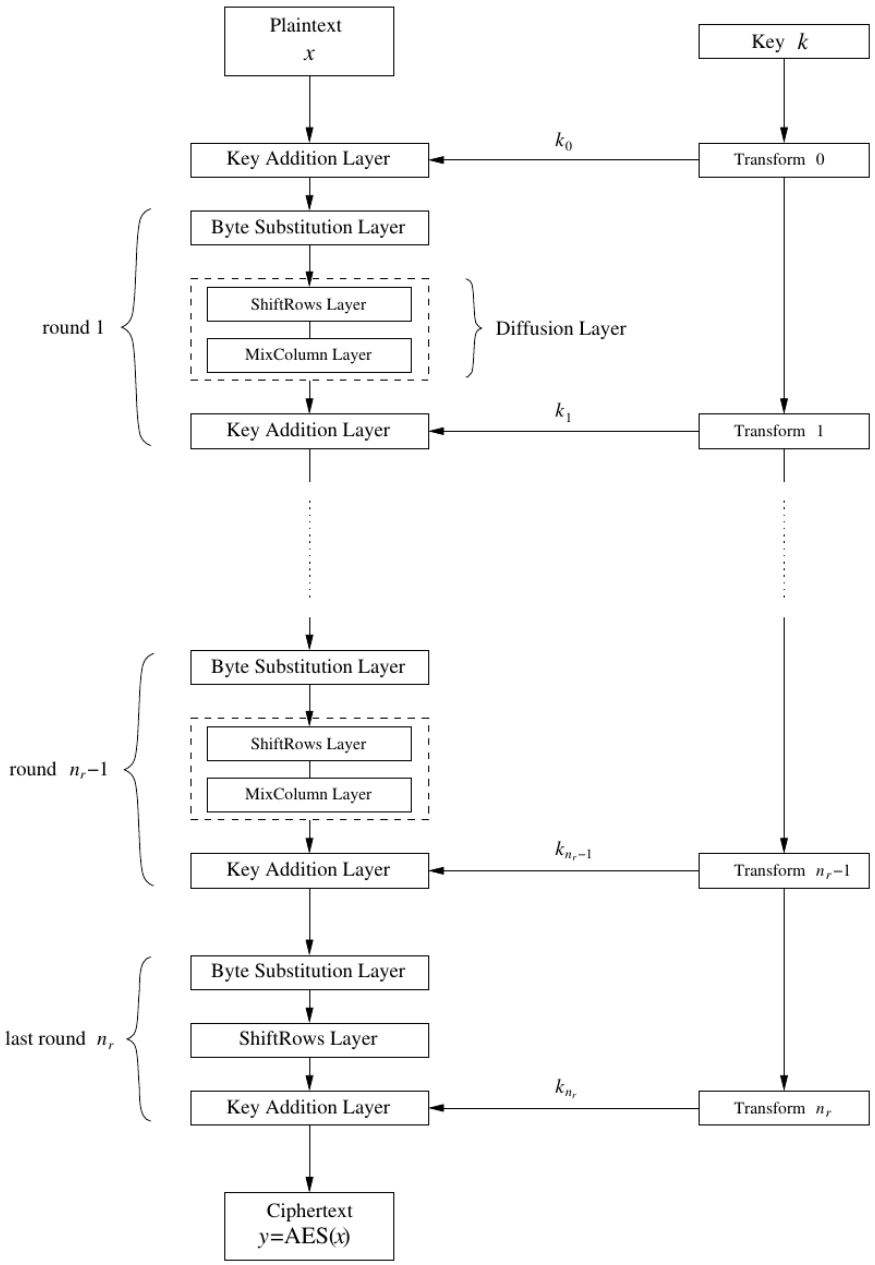
Cycle counts and key sizes for AES

Key Lengths	**# Rounds = n_r**
128 bit	10
192 bit	12
256 bit	14

Layers make up AES encryption. The bus's 128 bits are processed by each tier. The algorithm's state is also known as the bus. There are three different types of layers. Every round, except the first, has three layers: plaintext x, ciphertext y, and round number n_r, as illustrated in Figure 18. Furthermore, the last round number does not employ the MixColumn transform, resulting in a symmetrical encryption and decryption technique.

Figure 18

Block diagram of AES algorithm



Note. (Bhat et al., 2015)

Brief description for layers:

In key timing, a 128-bit loop key or subkey obtained from the master key is XORed on a case-by-case basis in the Key Addition Layer.

Each element of the state in the loop is converted non-linearly using lookup tables with particular mathematical features in the Byte Substitution layer (S-Box). This causes data to get jumbled, resulting in changes in each state bit propagating fast throughout the bus. Nonlinearity is provided by this layer.

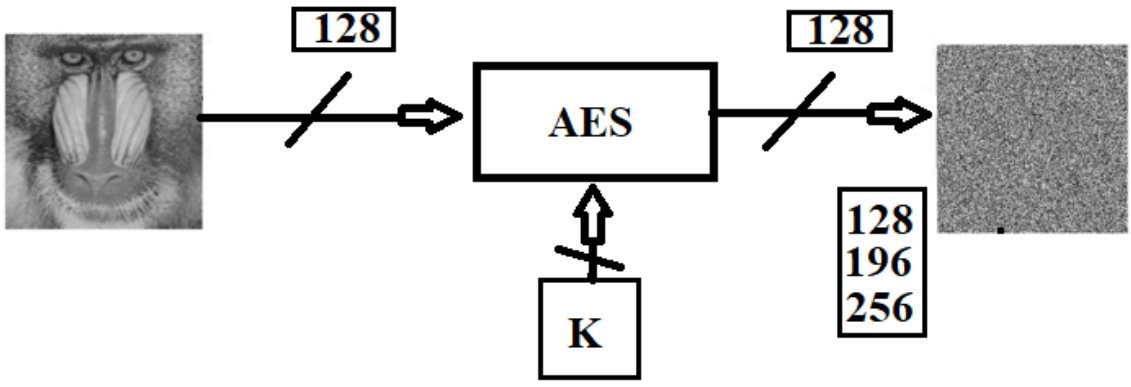
Diffusion layer allows diffusion to occur across all state bits. It is made up of two sublayers, each of which performs linear operations:

- ShiftRows Layer: Data is mixed at the byte level.
- MixColumn Layer: It's a matrix operation that combines (mixes) 4-byte chunks. Key setup calculates subkeys from the original AES key.

In this study, gray level image pixels are extracted from the image file and directly encrypted with the AES algorithm using a 256-bit key as in Figure 19. The image is first within pixel values and then given to the AES algorithm as input. The output is divided into 256-bit pixels and then represented as pixel values of a bitmap image. Each block is encrypted using CBC mode. Security analyzes are made on the encrypted image obtained as a result of encryption.

Figure 19

AES algorithm Image input and output



4.2. Security Analysis

The capacity to execute encryption and decryption with suitable performance, as well as the unauthorized end system's inability to access information about plain text, are directly proportional to the quality of an encryption system. An excellent encryption system must be safe against all known attacks.

Confidentiality (the message cannot be read by unauthorized people), integrity (the message cannot be changed or corrupted by unauthorized persons, no change can be made on the information), and usability (the message must be fully accessible by the person to whom it is intended) used in the field of information security to accept that the images are securely encrypted. Apart from the conditions, some basic conditions given below must be met.

The encryption system must be computationally very secure. Cracking the password should require huge computation time.

1. Encryption and decryption algorithms should be rapid enough to not affect system performance and simple enough to be implemented by people using home computers.
2. The security method should be adaptable and have a wide range of applications.
3. Encrypted image data should not increase in size too much.

Some security analyses of picture encryption using the 256-bit AES encryption standard are performed in this section. All tests and analyses are carried out on a PC with an Intel Core i7 3.00 GHz CPU, 16 GB of RAM, a 500 GB hard drive, and the 64-bit Windows 10 Professional operating system. The program is created with the Visual Studio

2019 IDE and the C# language on the.NET 4.8 platform, and the visuals are created on this platform.

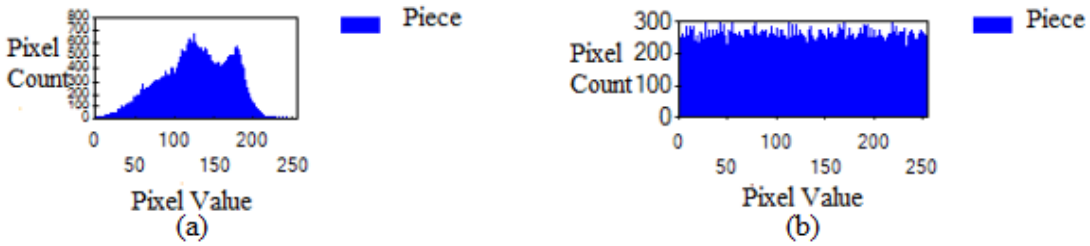
4.2.1. Histogram Analysis

A histogram is a graphical depiction of the number of different color values in a numerical image. It is possible to obtain brightness or tone information about pictures by the histograms. In Figure 20, the histogram graphics of the images obtained by encoding 256x256 flat Mandrill (a) and AES (b) are compared, while encrypted image histogram have the following properties:

1. The histogram of the original image is completely different.
2. Uniform distribution, i.e., any gray level pixel value has the same probability of occurring.

Figure 20

Histogram comparison (a) Histogram plot of plain mandrill image (b) Histogram of AES encoded image



As can be seen from the Figure 20 (b), the histogram graph of the encrypted picture created as a consequence of encryption using AES is close to linear character and regularity. This can be explained by the change of pixel values.

4.2.2. Information Entropy

Entropy is a concept that is defined by Shannon in 1948 and comes from information theory (Shannon, 1949, p. 687). When analyzing an encryption system, the idea of entropy is extremely significant. The main feature of uncertainty is information entropy. Entropy measures the degree of uncertainty that exists in the system. The more predictable a sequence is, the more information it contains. So the way to hide information is to remove predictability. Information theory is now used extensively in cryptography, network security, communication systems, data compression, error correlation, and other fields (Enayatifar , 2011, p. 221).

H(m) is the entropy of a message, and it may be computed as follows:

$$H(m) = \sum_{i=0}^{L-1} p(m_i) \log \frac{1}{p(m_i)} \quad (4)$$

In image coding, L stands for the total number of symbols and the total amount of pixels. The logarithm is taken in terms of base 2. The probability that a pixel is m_i is denoted by $p(m_i)$. If the entropy of the encoded picture comes close enough to the logarithm L, the histogram of the picture can be said to be smooth enough. For example, assuming that a message contains the symbol 2^8 ($s = \{ s_1, s_2, \dots, s_{2^8} \}$) with the same probability, equation (4) generates a true random value such that H(s) = 8. This value is the ideal value. But in reality, the content of an encrypted message seldom contains genuine random symbols. As a result, the encrypted message's entropy is lower than its optimal value.

Table 4 shows the H(m) entropy values of the AES encrypted picture after calculating the likelihood of discovering all gray level values in the encrypted image. These values are extremely close to the optimal gray image value of 8. Therefore, it can be said that the encryption system is safe according to entropy attacks.

Table 4

Entropy values of mandrill image encrypted with AES

	Entropy	
	Plain Image	**Encrypted Image**
Entropy Value	7,1869	7,9993

4.2.3. Correlation Coefficient

Correlation is a term used in probability theory and statistics to describe the direction and strength of a linear relationship between two random variables. In a statistical context, correlation reflects how far off two variables are from being independent. The direction and amount of the link between the independent variables are indicated by the correlation coefficient. This coefficient can have a value of (-1) or (+1). Positive numbers represent a direct linear relationship, whereas negative values represent an inverse linear relationship. There is no linear relationship between the variables if the correlation coefficient is 0. As a result of the various conditions, many correlation coefficients have been produced. The Pearson product-moment correlation coefficient is one of the most well-known of these (Newton, 1997).

It's calculated by multiplying the product of two variables standard deviations by their covariance. Pearson correlation coefficients are utilized to determine the degree of reliance between the two pictures in this investigation.

In a meaningful picture, the correlation between two neighboring horizontal, vertical, and diagonal pixels is generally strong. Because the pixel values of a meaningful picture are quite near to each other. The correlation between two nearby pixels is tested using correlation analysis. First, from the encrypted image, p pairs of neighboring horizontal, vertical, and diagonal pixels are randomly picked. Each pair's correlation coefficients are determined as in r_{uv} (8).

$$E(u) = \frac{1}{p}\sum_{i=1}^{p} u_i \tag{5}$$

$$D(u) = \frac{1}{p}\sum_{i=1}^{p} (u_i - E(u))^2 \tag{6}$$

$$cov(u, v) = E\{(u - E(u))(v - E(v))\} \tag{7}$$

$$r_{uv} = \frac{cov(u, v)}{\sqrt{D(u)D(v)}} \tag{8}$$

Here, $\sqrt{D(u)}$ is standard deviation and E(u) is mean. cov(u, v) is covariance of pixels, u and v are the values of two adjacent pixels in the image and p is the number of the selected pixel pair. The same calculation is calculated in the vertical and diagonal directions. Correlation coefficients between neighboring pixels of the flat image are usually large and approach 1. In encrypted images, it is usually small and close to 0. A

good cryptosystem should remove the associations of adjacent pixels in the plain picture as much as possible in the encrypted picture. The graphical representation of the correlation coefficient analysis with 1000 pairs of neighboring pixels is shown in Figure 21. By looking at Figure 22, we can easily understand that the linear pixel relationship in the plain picture is not existed in the encrypted picture. It is calculated according to the horizontal, vertical and diagonal neighborhoods of the image encrypted with the AES encryption standard. The correlation coefficients calculated in the picture below are compared and shown in Table 5.

Figure 21

Correlation graphs (a), (b), (c) relations of adjacent horizontal, vertical and diagonal pixels in flat mandrill picture, (d), (e), (f) horizontal, vertical and diagonal neighboring pixels, respectively, in AES relations in the mandrill image encrypted

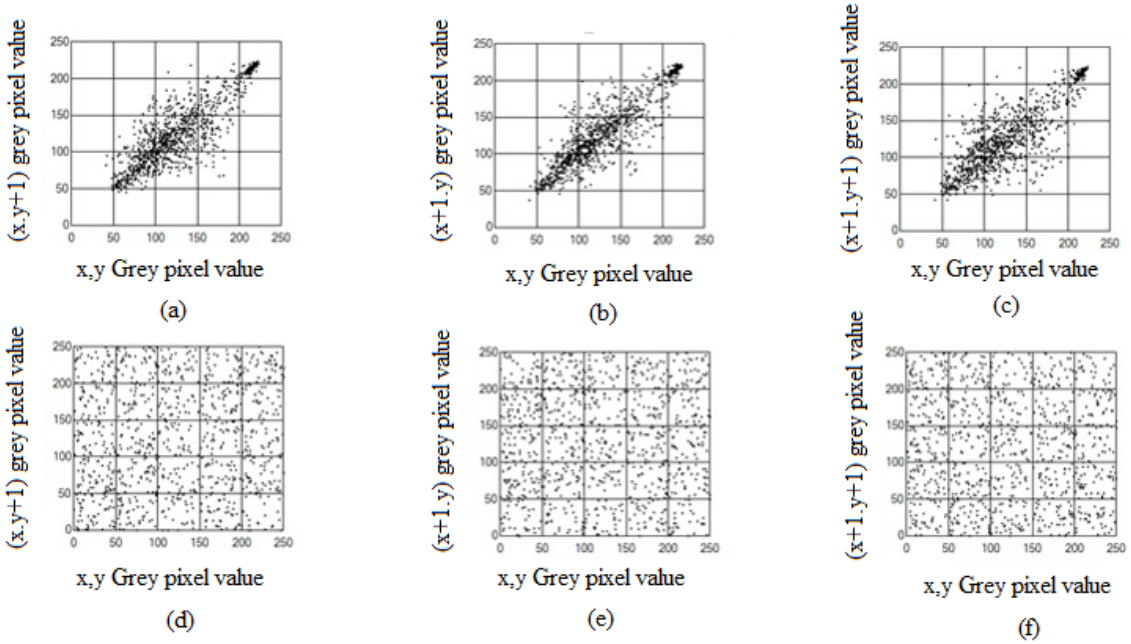


Table 5

Correlation coefficients of neighboring pixels of images encrypted with AES

	Plain Mandril Image	Image encrypted with AES 256-bit key
Horizontal	0.96290	0.01048
Vertical	0.93512	-0.02881
Diyagonal	0.94884	-0.01724

Correlation coefficients according to horizontal, vertical and diagonal pixels in flat picture are 0.96290, 0.93512, 0.94884, respectively. This means the pixels in the flat picture have a strong relationship with their neighbors. The horizontal, vertical, and diagonal correlation coefficients for the picture encrypted with AES using a 256-bit key are 0.01048, -0.02881, and -0.01724, respectively. This means that the encrypted image's pixels have a poor correlation with their neighbors. The correlation coefficients are really close to the ideal value of 0.

4.2.4. Differential Attack

Biham and Shamir are the first to introduce differential cryptanalysis (Alvarez et al., 2000, p. 194). The purpose of the differential attack is to discover the secret key. Differential attack examines how small changes in the plain text make changes to the encrypted image. These changes can be used to identify possible keys. Diffusion should be a strong feature of a solid encryption scheme. When a single bit of plaintext is changed,

the ciphertext must likewise change unexpectedly. The propagation nature of an image encoding technique can be defined by whether the output pixels of the ciphertext image and the input pixels of the plain image are extremely closely related. The more complex these features of the encryption method are, the more secure it is against differential attacks.

The general strategy for a differential attack is to compare the randomly selected original image with the encrypted images obtained by encrypting a randomly modified pixel with the same key. A small change in the flat image creates a large and unexpected change in the encoded image, neutralizing different attacks. The number of pixel change rate (NPCR - Number of Pixel Change Rate) and the combined average changing intensity (UACI - Unified Average Changing Intensity) techniques are employed in this study to examine the reliability of the picture encryption method using AES against differential assaults.

NPCR is the ratio of pixel differences in encrypted images obtained by encrypting two plain images with the equal key, and it is calculated as in (10). The average density difference between two encrypted pictures is obtained using the formula in UACI (11). The encrypted pictures c_1 and c_2 are the plain image and its one-pixel changed variant, respectively, encrypted with the same key.

$$D(i,j) = \begin{cases} 1, & c_{1(i,j)} \neq c_{2(i,j)} \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

$$NPCR = \frac{\sum_{i,j} D(i,j)}{M \times N} \times 100\% \quad (10)$$

$$\mathrm{UACI} = \frac{1}{\mathrm{MxN}}\left[\sum_{\mathrm{i,j}} \frac{|\mathrm{c}_{1(\mathrm{i,j})} - \mathrm{c}_{2(\mathrm{i,j})}|}{255}\right] \mathrm{x}\ 100\% \quad (11)$$

Table 6 shows the NPCR and UACI values obtained in encryption with various keys. Because the NPCR and UACI values in this study are close to ideal levels, they can withstand differential type assaults. The ideal value of NPCR and UACI are 99.61 % and 33.46% , respectively(Bensikaddour et al., 2020).

Table 6

NPCR and UACI comparisons of chaotic image coding in the study

Key	**NPCR**	**UACI**
	With bit-based scrambling	**With bit-based scrambling**
1	99,62697	33,48819
2	99,64311	33,56764
3	99,60474	33,60573
4	99,60046	33,49480
5	99, 64811	33,58362

4.2.5. Key Space Study

An effective encryption system must have sufficient key length to with stand assaults using brute force. Theoretically, there is no encryption system that cannot be cracked with brute force attacks. With the advancement of technology, the success times of brute force attacks can be extended with sufficient key sizes. Therefore, encryption

systems such as AES are standardized with multiple different key sizes. The AES encryption standard can be used with keys of 128, 192 and 256 bits. To make encryption systems resistant to brute force assaults, Alvarez and Li proposed that key sizes be at least 2100 characters long (Alvarez & Li, 2006, p. 2135). In this study, the AES standard using a 256-bit key is used. This key size is considered secure.

4.2.6. Key Sensitivity

Minor changes in encryption and decryption keys must be detected by a powerful cryptosystem. During the encryption phase, the encrypted picture produced by a little change in the encryption key should be considerably different from the real encrypted image. Similarly, there should be no statistical similarities between the new plain picture acquired and the real flat picture when decryption is conducted with the new key obtained with a little modification in the key used in the decryption step.

The system is sensitive to the smallest change in secret keys, according to encryption and decryption experiments with keys that are very close to each other.The pixel differences between the encrypted picture and the actual encrypted picture produced when we encrypt the same plain picture with a small change in the actual secret key are shown in Table 7. The flat pictures obtained when we decrypt the picture encrypted with the real key and the slightly modified key are shown in Figure 22. As can be seen from Table 7 and Figure 22, the encryption system in this study is sensitive to very small changes in the keys during the encryption and decryption stages.

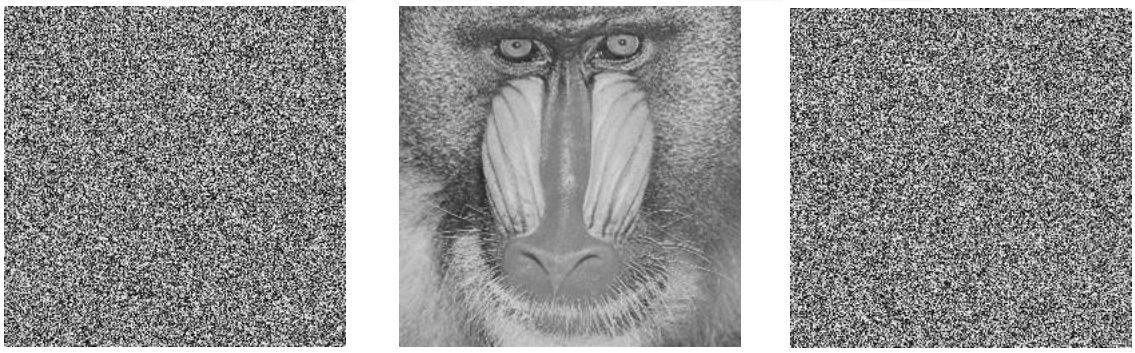
Table 7

The pixel differences of the picture encrypted with the keys that are very slightly different between them

	NPCR	
Image	**Image size 256x256**	**Image size 512x512**
Mandrill	99.60352	99.60645

Figure 22

Key sensitivity in decryption (a) encrypted mandrill picture (b) plain mandrill picture decrypted with real key (c) meaningless picture resulting from decryption by 1-bit change in key



(a) (b) (c)

4.2.7. Encryption Quality

Image encryption quality measurement is a criterion used in the evaluation of image encryption systems. The quality of the encryption is measured by comparing the pixel values of an image that are changed by the encryption process with their original values

before encryption. The pixel values of an image change drastically with the encoding process. This change is irregular. The larger the variation in pixel values, the worse the encryption method's quality. The encryption quality of an image algorithm is defined as the difference between the plain image and the encrypted picture (Kalash et al.,2006, p. 279).

P and C represent the flat picture and the encrypted picture of this flat picture with dimensions M x N and L gray level, respectively. P(x, y), C(x, y) ∈ { 0,1,2…L-1 } will be 0<x<M-1 and 0<y<N-1 will be P plain image and C encrypted image, are the gray level values at (x,y) position in the figure, respectively. If HL(P) is defined as the amount of occurrence of each L gray level pixel value in the encrypted picture and HL(C) as the amount of occurrence of each L amount of gray pixel value in the plain picture, the level of encryption is the average change of each L gray level that is calculated as

$$\text{Encryption Quality} = \frac{\sum_{L=0}^{255} |H_L(C) - H_L(P)|}{256} \quad (12)$$

Table 8 shows the estimated encryption quality when the picture is encrypted by the AES method with varied key and image sizes.

Table 8

Encryption quality for image encryption with AES

	Image size 256x256	**Image size 512x512**
AES 128 bit	193.27	793,76
AES 256 bit	193.45	794,37

4.2.8. Performance Analysis

The encryption algorithm's execution speed is crucial in real-time apps on the internet Table 9 shows the performance of the AES encryption technique for various key sizes.

Table 9

Performance comparisons in encryptions with different key sizes of images of different sizes

	Encryption time (ms)	
Algorithm	**Image size 256x256**	**Image size 512x512**
AES 128 bit	0.132	0.467
AES 192 bit	0.136	0.472
AES 256 bit	0.146	0.481

As shown in Table 9, the encryption speed decreases as the key size increases. The most appropriate key size should be chosen according to the application needs and the performance-security trade-off. Large key sizes should not be suitable for real-time applications

5. CONCLUSION

Multimedia files contain larger size data than plain text, so studies have been done with more specific encryption algorithms for their encryption. At the same time, there are specific security analysis methods in addition to the security analysis of plain text encryption methods. Multimedia applications must pass all known security analyzes, thus it can be understood whether it is safe.

As a result of the examinations, it has been observed that in today's world where confidentiality is extremely important and information theft is at its peak, data encryption will further develop and the number of cryptology techniques will become stronger due to the point that technology has reached.

In this study, an evaluation framework has been presented to test the reliability of image encryption methods used in information systems.

The entropy value of the encrypted image is close to the ideal value. Histogram graphs show that the pixel values in the encrypted image are randomly and close to each other. As a result of differential attack analysis, it is observed that NPCR and UACI values are very high for AES. Key sizes of AES are long enough to resist brute force attacks. In the key susceptibility test, NPCR and UACI values are quite high and were close to ideal values.

As a result of a series of known security analyzes for image encryption methods, it has been observed that the AES encryption method is a secure method in image encryption.

This security framework has been implemented as follows:

i. Visual check is the first stage of the test and, if the plain image's characteristics are not entirely masked, the considered cryptosystem is not secure.
ii. Histogram uniformity is essential to evaluate encryption quality.

iii. The correlation coefficients test examines similarity of pixels with their neighbors to determine whether information is destroyed or not.

iv. The key sensitivity test is a useful tool for assessing the encryption algorithm's diffusion qualities.

v. The entropy analysis tests whether the information has been removed on the encrypted image.

vi. The encryption method's key length should be long enough to withstand brute force assaults.

vii. The most suitable key length should be selected considering the performance-security for the requirements of the application.

An image encryption method has been cosidered as a secure algorithm if it passes all the security mesasurements given above. As a special case, security analyzes were performed on the images encrypted with the AES symmetric encryption method in CBC mode using a 256-bit key, and it has been demonstrated that the AES method is a secure method in image encryption.It is important to note that this security framework can be utilized to examine the security of all image encryption methods, since it just needs the data of files stored in standard image formats such as PNG, JPEG or TIFF.

REFERENCES

Ahmed, H. H., Kalash, H. M.& Farag Allah, O. S. (2006). Encryption quality analysis of rc5 block cipher algorithm for digital images, *Journal of Optical Engineering, 45, 10 (2006)* 277-284.

Alvarez, G. & Li, S.J. (2006). Some basic cryptographic requirements for chaos-based cryptosystem, Int. J. Bifurcat. Chaos,16, 8 (2006) 2129–2151.

Alvarez, G., Montoya, F., Romera, M.& Pastor, G. (2000). Cryptanalysis of a chaotic secure communication system, *Physics Letters A, 276 (2000)* 191-196.

Belazi, A., Abd El-Latif, A. A., & Belghith, S. (2016). A novel image encryption scheme based on substitution-permutation network and chaos. *Signal Processing*, *128*, 155-170.

Bhat, B., Ali, A. W., & Gupta, A. (2015, May). DES and AES performance evaluation. In *International Conference on Computing, Communication & Automation* (pp. 887-890). IEEE.

Bensikaddour, E. H., Bentoutou, Y., & Taleb, N. (2020). Embedded implementation of multispectral satellite image encryption using a chaos-based block cipher. *Journal of King Saud University-Computer and Information Sciences*, *32*(1), 50-56.

Bruce, S. (1996). Applied cryptography: protocols, algorithms, and source code in C. El Fishawy, N. F., & Zaid, O. M. A. (2007). Quality of encryption measurement of bitmap images with RC6, MRC6, and Rijndael block cipher algorithms. *Int. J. Netw. Secur.*, *5*(3), 241-251.

Enayatifar, R. (2011) Image encryption via logistic map function and heap tree, *Int. J. Phys. Sci, vol. 6, no. 2, p. 221*

Enayatifar, R.& Abdullah, A.H. Isnin, I.F. (2014). Chaos-based image encryption using a hybrid genetic algorithm and a DNA sequence. *Opt Lasers Eng 56:83–93*

Enayatifar, R., Sadaei, H.,Abdullah, A.H., Lee, M. & Isnin, I.F. (2015). A novel chaotic based image encryption using a hybrid model of deoxyribonucleic acid and cellular automata. *Opt Lasers Eng 71:33–41*

FIPS 197, (2001). *Advanced Encryption Standard,* US Department of Commerce, Washington D. C,

Forouzan, B.A. & Mukhopadhyay, D. (2011). *Cryptography and network security (Sie).* McGraw-Hill Education, New York

Gu, G., & Ling, J. (2014). A fast image encryption method by using chaotic 3D cat maps. *Optik*, *125*(17), 4700-4705.

Hasselblatt, B. & Katok, A. (2003). *First Course in Dynamics: With a panorama of recent developments*. Cambridge University Press, Cambridge

Heys, H. M. (2002). A tutorial on linear and differential cryptanalysis. *Cryptologia*, *26*(3), 189-221.

Huang, X., & Ye, G. (2014). An efficient self-adaptive model for chaotic image encryption algorithm. *Communications in Nonlinear Science and Numerical Simulation*, *19*(12), 4094-4104.

Jin, H., Liao, Z., Zou, D., & Li, C. (2008, February). An Asymmetrical Encryption Based automated trust negotiation model. In *2008 2nd IEEE International Conference on Digital Ecosystems and Technologies* (pp. 363-368). IEEE.

Khan, M., & Shah, T. (2014). A literature review on image encryption techniques. *3D Research, 5(4), 1-25.*

Kocarev, L. (2001). Chaos-based cryptography: a brief overview. *IEEE Circuits and Systems Magazine, 1*(3), 6-21.

Kuo, C. J. (1993). Novel image encryption technique and its application in progressive transmission. *Journal of Electronic Imaging, 2*(4), 345-351.

Küsters, R., & Tuengerthal, M.(2009, July). Universally composable symmetric encryption. In *2009 22nd IEEE Computer Security Foundations Symposium* (pp. 293-307). IEEE.

Li, S. J. (2003). *Analyses and new designs of digital chaotic ciphers* (Doctoral dissertation, Xi'an Jiaotong University).

Li, S., Chen, G. ve Zheng, X., (2004*). Multimedia Security Handbook*, CRC Press LLC, Boca Raton, 2004.

Li, S., Mou, X. & Cai, Y. (2001) Pseudo-Random Bit Generator Based on Couple Chaotic Systems and Its Applications in Stream Cipher Cryptography, *Indocrypt, Berlin, Springer-Verlag LNCS, 316-329 Shujun Li Xuanqin Mou*

Mao, Y. (2003). Research on chaos-based image encryption and watermarking technology. *Ph. D. thesis*.

Mao, Y., Chen, G., & Lian, S. (2004). A novel fast image encryption scheme based on 3D chaotic baker maps. *International Journal of Bifurcation and chaos*, *14*(10), 3613-3624.

Mao, Y. & Wu, M. (2006). A Joint Signal Processing and Cryptographic Approach to Multimedia Encryption , *IEEE Transactions on Image Processing, 15, 7 (2006)* 2061-2075.

Mohamed, F.K. (2014). A parallel block-based encryption schema for digital images using reversible cellular automata. *Eng Sci Technol Int J 17*(2):85–94

Newton, D. E. (1997) *Encyclopedia of Cryptology*, ABC-CLIO, Santa Barbara, CA

Qiao, L. (1998). *Multimedia security and copyright protection*. University of Illinois at Urbana-Champaign.

Standard, D. E. (1999). Data encryption standard. *Federal Information Processing Standards Publication, 112*.

Saha, A. & Majumdar, R. (2021). Overview of Multimedia Security. *Retrieved from http://www.academia.edu/8199308/Overview_of_Multimedia_Security*.

Seo, H., Choi, J., Kim, H., Park, T., & Kim, H. (2014, March). Pseudo random number generator and hash function for embedded microprocessors. In *2014 IEEE World Forum on Internet of Things (WF-IoT)* (pp. 37-40). IEEE.

Shannon, C. E. (1949). Communication theory of secrecy systems. *The Bell system technical journal, 28*(4), 656-715.

Stallings, W. (1999). Secure hash algorithm. *Cryptography and Network Security: Principles and Practice*, 193-197.

Stinson, D. (2002) *Cryptography: Theory and Practice* , Second Edition. Boca Raton: Chapman & Hall / CRC;

Souici, I., Seridi, H., & Akdag, H. (2011). Images encryption by the use of evolutionary algorithms. *Analog Integrated Circuits and Signal Processing*, *69*(1), 49-58.

Umamageswari, A. & Suresh, G. (2013). Security in medical image communication with Arnold's cat map method and reversible watermarking. *International conference on circuits, power and computing technologies (ICCPCT). IEEE, pp* 1116–1121.

Wang, X.& Luan, D. (2013) A novel image encryption algorithm using chaos and reversible cellular automata. *Commun Non-linear Sci Numer Simul 18*(11):3075–3085

Wang, X. Y., Gu, S. X., & Zhang, Y. Q. (2015). Novel image encryption algorithm based on cycle shift and chaotic system. *Optics and Lasers in Engineering*, *68*, 126-134.

Yeung, S.A., Zhu, S. & Zeng, B.(2010). Quality Assessment for a Perceptual Video Encryption System, *Proceedings of the IEEE Wireless Communications, Networking and Information Security (WCNIS) Conference, pp.* 102–106

Zhang, Q., Liu, L. & Wei, X. (2014). Improved algorithm for image encryption based on DNA encoding and multi-chaotic maps. *AEU Int J Electron Commun 68*(3):186–192

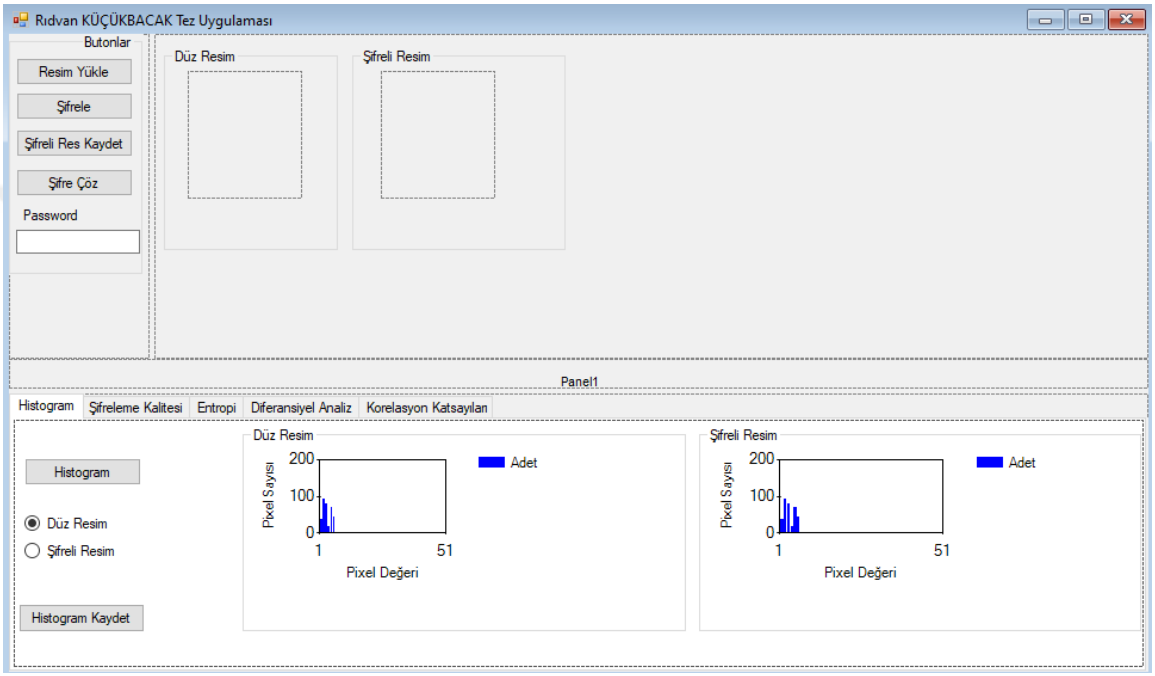
Zeng, W., Yu, H., & Lin, C. Y. (Eds.). (2011). *Multimedia security technologies for digital rights management*. Elsevier.

Zhou, J., Au, O., Fan, X. & Wong, P. (2008). Joint security and performance enhancement for secure arithmetic coding, in image processing, *ICIP 2008. 15th IEEE International Conference on. IEEE, 2008, pp.* 3120–3123.

APPENDICES

In this section, there are code blocks of the algorithms analyzed in the thesis. You can make the results executable with visual studio 2019 .net framework 4.8 by copying the code blocks.

After the necessary designs in the picture below are made, the code blocks can be copied.




```
public static class AesEncryption
  {
     public static byte[] Encrypt(byte[] clearData, byte[] Key, byte[] IV)
     {
        MemoryStream ms = new MemoryStream();
        Rijndael alg = Rijndael.Create();
        alg.Key = Key;
        alg.IV = IV;
```

```
    alg.Mode = CipherMode.CFB;
    alg.Padding = PaddingMode.Zeros;
    CryptoStream cs = new CryptoStream(ms,
       alg.CreateEncryptor(), CryptoStreamMode.Write);
    cs.Write(clearData, 0, clearData.Length);
    cs.Close();
    byte[] encryptedData = ms.ToArray();
    return encryptedData;
}
public static string Encrypt(string clearText, string Password)
{
    byte[] clearBytes =
      System.Text.Encoding.Unicode.GetBytes(clearText);
    PasswordDeriveBytes pdb = new PasswordDeriveBytes(Password,
        new byte[] {0x49, 0x76, 0x61, 0x6e, 0x20, 0x4d,
    0x65, 0x64, 0x76, 0x65, 0x64, 0x65, 0x76});
    byte[] encryptedData = Encrypt(clearBytes,
            pdb.GetBytes(32), pdb.GetBytes(16));
    return Convert.ToBase64String(encryptedData);
```

```
}
public static byte[] Encrypt(byte[] clearData, string Password)
{
   PasswordDeriveBytes pdb = new PasswordDeriveBytes(Password,
      new byte[] {0x49, 0x76, 0x61, 0x6e, 0x20, 0x4d,
   0x65, 0x64, 0x76, 0x65, 0x64, 0x65, 0x76});
   return Encrypt(clearData, pdb.GetBytes(32), pdb.GetBytes(16));
}
public static void Encrypt(string fileIn, string fileOut, string Password)
{
   FileStream fsIn = new FileStream(fileIn,
      FileMode.Open, FileAccess.Read);
   FileStream fsOut = new FileStream(fileOut,
      FileMode.OpenOrCreate, FileAccess.Write);
   PasswordDeriveBytes pdb = new PasswordDeriveBytes(Password,
      new byte[] {0x49, 0x76, 0x61, 0x6e, 0x20, 0x4d,
   0x65, 0x64, 0x76, 0x65, 0x64, 0x65, 0x76});
   Rijndael alg = Rijndael.Create();
   alg.Key = pdb.GetBytes(32);
```

```
            alg.IV = pdb.GetBytes(16);
            CryptoStream cs = new CryptoStream(fsOut, alg.CreateEncryptor(),
CryptoStreamMode.Write);
            int bufferLen = 4096;
            byte[] buffer = new byte[bufferLen];
            int bytesRead;
            do
            {
                bytesRead = fsIn.Read(buffer, 0, bufferLen);
                cs.Write(buffer, 0, bytesRead);
            } while (bytesRead != 0);
            cs.Close();
            fsIn.Close();
        }
        public static byte[] Decrypt(byte[] cipherData,
                                     byte[] Key, byte[] IV)
        {
            MemoryStream ms = new MemoryStream();
            Rijndael alg = Rijndael.Create();
```

```
    alg.Key = Key;
    alg.IV = IV;
    alg.Mode = CipherMode.CFB;
    alg.Padding = PaddingMode.Zeros;
    CryptoStream cs = new CryptoStream(ms,
        alg.CreateDecryptor(), CryptoStreamMode.Write);
    cs.Write(cipherData, 0, cipherData.Length);
    cs.Close();
    byte[] decryptedData = ms.ToArray();
    return decryptedData;
}
public static string Decrypt(string cipherText, string Password)
{
    byte[] cipherBytes = Convert.FromBase64String(cipherText);
    PasswordDeriveBytes pdb = new PasswordDeriveBytes(Password,
        new byte[] {0x49, 0x76, 0x61, 0x6e, 0x20, 0x4d, 0x65,
    0x64, 0x76, 0x65, 0x64, 0x65, 0x76});
    byte[] decryptedData = Decrypt(cipherBytes,
        pdb.GetBytes(32), pdb.GetBytes(16));
```

```
    return System.Text.Encoding.Unicode.GetString(decryptedData);
}
public static byte[] Decrypt(byte[] cipherData, string Password)
{
    PasswordDeriveBytes pdb = new PasswordDeriveBytes(Password,
        new byte[] {0x49, 0x76, 0x61, 0x6e, 0x20, 0x4d,
    0x65, 0x64, 0x76, 0x65, 0x64, 0x65, 0x76});
    return Decrypt(cipherData, pdb.GetBytes(32), pdb.GetBytes(16));
}
public static void Decrypt(string fileIn,
            string fileOut, string Password)
{
    FileStream fsIn = new FileStream(fileIn,
                FileMode.Open, FileAccess.Read);
    FileStream fsOut = new FileStream(fileOut,
                FileMode.OpenOrCreate, FileAccess.Write);
    PasswordDeriveBytes pdb = new PasswordDeriveBytes(Password,
        new byte[] {0x49, 0x76, 0x61, 0x6e, 0x20, 0x4d,
    0x65, 0x64, 0x76, 0x65, 0x64, 0x65, 0x76});
```

```
            Rijndael alg = Rijndael.Create();
            alg.Key = pdb.GetBytes(32);
            alg.IV = pdb.GetBytes(16);
            CryptoStream cs = new CryptoStream(fsOut,
               alg.CreateDecryptor(), CryptoStreamMode.Write);
            int bufferLen = 4096;
            byte[] buffer = new byte[bufferLen];
            int bytesRead;
            do
            {
               bytesRead = fsIn.Read(buffer, 0, bufferLen);
               cs.Write(buffer, 0, bytesRead);
            } while (bytesRead != 0);
            cs.Close();
            fsIn.Close();
         }
      }
   }
public static class AesEncryptionDiff
```

```
{
    public static byte[] Encrypt(byte[] clearData, byte[] Key, byte[] IV)
    {
        MemoryStream ms = new MemoryStream();
        Rijndael alg = Rijndael.Create();
        alg.Key = Key;
        alg.IV = IV;
        alg.Mode = CipherMode.CBC;
        alg.Padding = PaddingMode.Zeros;
        CryptoStream cs = new CryptoStream(ms,
           alg.CreateEncryptor(), CryptoStreamMode.Write);
        cs.Write(clearData, 0, clearData.Length);
        cs.Close();
        byte[] encryptedData = ms.ToArray();
        return encryptedData;
    }
    public static string Encrypt(string clearText, string Password)
    {
        byte[] clearBytes =
```

```
        System.Text.Encoding.Unicode.GetBytes(clearText);
    PasswordDeriveBytes pdb = new PasswordDeriveBytes(Password,
        new byte[] {0x49, 0x76, 0x61, 0x6e, 0x20, 0x4d,
    0x65, 0x64, 0x76, 0x65, 0x64, 0x65, 0x76});
    byte[] encryptedData = Encrypt(clearBytes,
            pdb.GetBytes(32), pdb.GetBytes(16));
    return Convert.ToBase64String(encryptedData);
}
public static byte[] Encrypt(byte[] clearData, string Password)
{
    PasswordDeriveBytes pdb = new PasswordDeriveBytes(Password,
        new byte[] {0x49, 0x76, 0x61, 0x6e, 0x20, 0x4d,
    0x65, 0x64, 0x76, 0x65, 0x64, 0x65, 0x76});
    return Encrypt(clearData, pdb.GetBytes(32), pdb.GetBytes(16));
}
public static void Encrypt(string fileIn,
            string fileOut, string Password)
{
    FileStream fsIn = new FileStream(fileIn,
```

```
    FileMode.Open, FileAccess.Read);
FileStream fsOut = new FileStream(fileOut,
    FileMode.OpenOrCreate, FileAccess.Write);
PasswordDeriveBytes pdb = new PasswordDeriveBytes(Password,
    new byte[] {0x49, 0x76, 0x61, 0x6e, 0x20, 0x4d,
0x65, 0x64, 0x76, 0x65, 0x64, 0x65, 0x76});
Rijndael alg = Rijndael.Create();
alg.Key = pdb.GetBytes(32);
alg.IV = pdb.GetBytes(16);
CryptoStream cs = new CryptoStream(fsOut,
    alg.CreateEncryptor(), CryptoStreamMode.Write);
int bufferLen = 4096;
byte[] buffer = new byte[bufferLen];
int bytesRead;
do
{
    bytesRead = fsIn.Read(buffer, 0, bufferLen);
    cs.Write(buffer, 0, bytesRead);
} while (bytesRead != 0);
```

```
            cs.Close();
            fsIn.Close();
        }
    }
}
    internal class differential
    {
        public static Bitmap onePixelExchange(Bitmap bmp, int a, int b, int c)
        {
            int[] red = new int[256];
            int[] green = new int[256];
            int[] mavi = new int[256];
            Colour r = bmp.GetPixel(a, b);
            Colour newr = Colour.FromArgb(c, c, c);
            if (r = = newr)
                System.Windows.Forms.MessageBox.Show("Pixel values are the same. Please choose another value.");
            else
                bmp.SetPixel(a, b, newr);
```

```
    return bmp;
}
internal static object npcr(Bitmap a, Bitmap b)
{
    //pictureboxEncrypted, Picturebox2
    double _npcr;
    double sigma = 0;
    Colour coloura;
    Colour colourb;
    for (int i = 0; i < a.Size.Height; i++)
    {
        for (int j = 0; j < a.Size.Width; j++)
        {
            coloura = a.GetPixel(i, j);
            colourb = b.GetPixel(i, j);
            int valuea = coloura.R;
            int valueb = colourb.R;
            sigma += D(valuea, valueb);
            //int value = (int)(0.114 * colour.A + 0.587 * colour.B + 0.299 * colour.G);
```

```
            //colour.A = value; colour.B = value; colour.G = value;
        }
    }
    _npcr = (sigma / (a.Size.Height * a.Size.Width)) * 100;
    return _npcr;
}

private static double D(int a, int b)
{
    int d;
    if (a == b)
        d = 0;
    else
        d = 1;
    return d;
}
internal static object uaci(Bitmap a, Bitmap b)
{
    double _uaci;
```

```
double sigma = 0;

Colour coloura;

Colour colourb;

for (int i = 0; i < a.Size.Height; i++)

{

   for (int j = 0; j < a.Size.Width; j++)

   {

      coloura = a.GetPixel(i, j);

      colourb = b.GetPixel(i, j);

      int valuea = coloura.R;

      int valueb = colourb.R;



      sigma += (double)Math.Abs((valuea - valueb)) / 255;

      //int value = (int)(0.114 * colour.A + 0.587 * colour.B + 0.299 * colour.G);

      //colour.A = value; colour.B = value; colour.G = value;

   }

}

_uaci = (sigma / (a.Size.Height * a.Size.Width)) * 100;

return _uaci;
```

```
        }
    }
}
  internal class Entropy
    {
        public static double calculateEntropy (Bitmap bmp)
        {
            int[] hist = Histogram.histogram(bmp);
            double entropy = 0;
            for (int i = 0; i < 256; i++)
            {
                if (hist[i] == 0)
                    continue;
                else
                    entropy += possibility(hist[i], hist) * Math.Log((double)(1 / possibility
(hist[i], hist)), 2);
            }
            return entropy;
        }
```

```
    private static double Possibility (int j, int[] k)
    {
        double possibility;

        possibility = (double)j / (double) TotalPixelValue(k);
        return possibility;
    }
    private static double TotalPixelValue (int[] i)
    {
        double total= 0;
        for (int k = 0; k < 256; k++)
        {
            total += (double)i[k];
        }
        return total;
    }
}
```

internal class Functions

```
{
    public static Image GreyImage(Image img, int size)
    {
        Bitmap bmp = new Bitmap(img, size, size);
        Colour colour;
        for (int i = 0; i < boyut; i++)
        {
            for (int j = 0; j < boyut; j++)
            {
                colour = bmp.GetPixel(i, j);
                int value = (colour.R + colour.B + colour.G) / 3;
                // int value = (int)(0.114 * colour.R + 0.587 * colour.B + 0.299 * colour.G);
                // colour.A = value; colour.B = value; colour.G = value;
                Color newcolour;
                newcolour = Color.FromArgb(value, value, value);
                bmp.SetPixel(i, j, newcolour);
            }
        }
        return bmp;
```

```
        }
        public static int Mod(long n, long m)
        {
            int a = Convert.ToInt32(((n % m) + m) % m);
            return a;
        }
        public static double Mod2(double n, int m)
        {
            return ((n % m) + m) % m;
        }
    }
}
    internal class Histogram
    {
        public static int[] histogram(Bitmap bmp)
        {
            int[] kirmizi = new int[256];
            int[] yesil = new int[256];
            int[] mavi = new int[256];
```

```
        for (int i = 0; i < bmp.Size.Height; i++)
        {
            for (int j = 0; j < bmp.Size.Width; j++)
            {
                Colour colour = bmp.GetPixel(i, j);
                red[colour.R]++;
                green[colour.G]++;
                blue[colour.B]++;
            }
        }
        return red;
    }
}
internal class Correlation
{
    public static double calculateCorrelation(double[] dizia, double[] dizib)
    {
        double correlation = 0;
        correlation = CorrelationCoefficient (arraya, arrayb);
```

```
    return Math.Round(correlation, 5);
}
public static double CorrelationCoefficient (double[] x, double[] y)
{
    double result;
    double xMean = 0;
    double yMean = 0;
    double xDenom = 0;
    double yDenom = 0;
    double denominator;
    double numerator = 0;
    double n;
    // Make sure arrays are same size and greater than 1
    if ((x.Count() == y.Count()) && (x.Count() >= 1))
    {
        n = x.Count();
    }
    else
    {
```

```
    result = 0;
    return result;
}
// Find Means
for (int i = 0; i <= n - 1; i++)
{
    xMean += x[i];
    yMean += y[i];
}
xMean /= n;
yMean /= n;
// Caluculate numerator and denominator
for (int i = 0; i <= n - 1; i++)
{
    //Caluculate numerator
    double numX = x[i] - xMean;
    double numY = y[i] - yMean;
    numerator += numX * numY;
    // Caluculate denominator parts
```

```
            xDenom += Math.Pow(numX, 2);
            yDenom += Math.Pow(numY, 2);
        }
        // Caluculate denominator
        denominator = Math.Sqrt(xDenom * yDenom);
        // Check for division by zero
        if (denominator == 0)
        {
            result = 0;
        }
        else
        {
            result = numerator / denominator;
        }
        return result;
    }
}
internal static class Program
{
```

```
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        private static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
internal class Encryptionquality
    {
        public static double EncryptionQuality(Bitmap a, Bitmap b)
        {
            double kalite;
            int[] _a = Histogram.histogram(a);
            int[] _b = Histogram.histogram(b);
            double sigma = 0;
```

```
            for (int j = 0; j < 256; j++)
            {
                sigma += (double)Math.Abs((_a[j] - _b[j]));
                //int value = (int)(0.114 * colour.A + 0.587 * colour.B + 0.299 * colour.G);
                // colour.A = value; colour.B = value; colour.G = value;
            }
            quality = (sigma / 256);
            return quality;
        }
    }
}
```