Chaos-based crypto and joint crypto-compression systems for images and videos

Mousa Farajallah

l'université nantes angers le mans
COMMUNAUTÉ D'UNIVERSITÉS ET ÉTABLISSEMENTS

UNIVERSITÉ DE NANTES

Thèse de Doctorat

Mousa FARAJALLAH

Mémoire présenté en vue de l'obtention du
grade de Docteur de l'Université de Nantes
sous le label de l'Université de Nantes Angers Le Mans

École doctorale : Sciences et technologies de l'information, et mathématiques

Discipline : Electronique/spécialité: Traitement du signal et des images
Unité de recherche : Institut d'Electronique et de Télécommunications de Rennes UMR CNRS 6164(IETR)

Soutenue le 30 Juin 2015

Chaos-based crypto and joint crypto-compression systems for images and videos

JURY

Présidente :	**M^me^ Danièle FOURNIER-PRUNARET**, Professeur des universités, INSA de Toulouse
Rapporteurs :	**M^me^ Christine GUILLEMOT**, Directrice de Recherche, IRISA, Rennes
	M. Thomas GROSGES, Professeur des universités, Université de Technologie de Troyes
Examinateurs :	**M. Pascal MOLLI**, Professeur des universités, Université de Nantes
	M. Calin VLADEANU, Professeur des universités, University Politehnica of Bucharest
Directeur de thèse :	**M. Safwan EL ASSAD**, Maître de Conférences, HDR, Université de Nantes
Co-encadrant de thèse :	**M. Olivier DEFORGES**, Professeur des universités, INSA de Rennes

ED STIM 503

Contents

* Say, "Indeed, my prayer, my rites of sacrifice, my living and my dying are for Allah , Lord of the worlds". (Surat Al-'An 'am 6.162).

I would like to dedicate this Doctoral dissertation to my sons Mohammed and Omar, my wife, my father , and my mother for their endless love, encouragement and help, to Martyrs of Palestine.

Acknowledgements

Special thanks to my supervisor, Dr. Safwan El Assad, for instructions, guidance, and for his efforts to orient my thinking. You have been a tremendous mentor for me. I would like to thank you for encouraging my research and for allowing me to grow as a research scientist. Your advice on both research has been invaluable, specially in publications.
I would like to express my special appreciation and deepest gratitude to my co-supervisor Prof. Olivier Deforges for his support, providing me with an excellent atmosphere for doing research.
I would like to to thank Dr. Wassim, was always help me and work hard with me specially in the compression part.
I acknowledge and thank my IETR lab for allowing me to conduct my research and providing any assistance requested, specially image team.
I won't forget the French government scholarship department. They gave me the opportunity to pursue my education that improve my academic. REZE municipality where they gave me a good and comfortable accommodation during this three years. Finally, The European Celtic-Plus project 4KREPROSYS - 4K ultraHD TV wireless REmote PROduction SYStems

Personal Publications

Published Journal paper:

1. Farajallah Mousa, El Assad Safwan and Deforges Olivier, " Fast and secure chaos based cryptosystem for images", International Journal of Bifurcation and Chaos (IJBC), Accepted in 15-June-2015, (Q1 journal, H Index: 65).
2. Chetto Maryline, El Assad Safwan, and Farajallah Mousa, "A lightweight chaos-based cryptosystem for dynamic security management in real-time overloaded applications." International Journal of Internet Technology and Secured Transactions, 5, no. 3 (2014): 262-274, (Q4 journal, H Index: 4). .

Published and accepted Conference papers:

1. Farajallah Mousa, Hamidouche Wassim, Deforges Olivier, and El Assad Safwan, "ROI Selective Video Encryption in the HEVC Video Standard", International Conference on Image Processing (ICIP) 2015 to be held from the 27th 30th September 2015, Quebec City, Canada. Accepted in 29-Avril-2015, (H Index: 40).
2. Hamidouche Wassim, Farajallah Mousa, Raulet Mickaël, Deforges Olivier, and El Assad Safwan, "Selective video encryption using chaotic system in the SHVC extension," in 40th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 19th 24th April 2015, Brisbane, Australia, April 2015, Brisbane, Australia, 5 Pages, (H Index: 70).
3. El Assad Safwan, Farajallah Mousa, and Calin Vladeanu, " Chaos-based Block Ciphers: An Overview", IEEE, 10th International Conference on Communications, COMM-2014, Bucharest, Romania, May 2014, pp. 23-26.
4. Farajallah Mousa, El Assad Safwan, and Chetto Maryline, "Dynamic adjustment of the chaos-based security in real-time energy harvesting sensors." In Green Computing and Communications (GreenCom), 2013 IEEE and Internet of Things (iThings/CPSCom), IEEE International Conference on and IEEE Cyber, Physical and Social Computing, pp. 282-289. IEEE, 2013, (H Index: 2).
5. Farajallah, Mousa, Fawaz Zeinab, El Assad Safwan, and Deforges Olivier, "Efficient image encryption and authentication scheme based on chaotic sequences." In SECURWARE 2013, The Seventh International Conference on Emerging Security Information, Systems and Technologies, pp. 150-155. 2013, (H Index: 1).
6. Fawaz Zeinab, El Assad Safwan, Farajallah Mousa, Khalil Ayman, Lozi René, and Deforges Olivier, "Lightweight chaos-based cryptosystem for secure images." In Information Science and Technology (ICIST), 2013 International Conference on, pp. 26-30. IEEE, 2013.
7. Chetto Maryline, Noura Hassan, El Assad Safwan, and Farajallah Mousa, "How to guarantee secured transactions with QoS and real-time constraints." In Internet Technology And Secured Transactions, 2012 International Conference for, pp. 40-44. IEEE, 2012.

Submitted Journal papers:

1. Hamidouche Wassim, Farajallah Mousa, Raulet Mickaël, Deforges Olivier, and El Assad Safwan, "Real Time Selective Video Encryption based on stream cipher in the Scalable HEVC Extension", IEEE Transactions on. Circuits and Systems for Video Technology, Submission date:23-Dec-2014, 13 Pages.

2. Farajallah Mousa, El Assad Safwan and Deforges Olivier, "Partial cryptanalysis of the first Zhang algorithms", International Journal of Bifurcation and Chaos (IJBC), Submission date:27-Mar-2015, 17 Pages.

3. El Assad Safwan and Farajallah Mousa, "A New Efficient Structure of a Cryptosystem based on Two Chaotic Layers: a Binary Diffusion Layer and a Bit-Permutation Layer", IEEE Systems Journal. under submission, 13 Pages.

Chaos-Based Cryptosystems

1

Introduction

The root of Cryptology is coming from the Greeks; it is the science that studies the theoretical and practical techniques for secure communications. It is divided into two related branches: cryptography and cryptanalysis. The cryptographer tries to find some techniques to guarantee the message secrecy and sometimes to ensure the authenticity of that message, while the cryptanalyst tries to cancel the cryptographer's work by breaking the secrecy of the message to retrieve the original one or by forging a message to be accepted as authentic one.

By using cryptography many goals can be achieved, and these goals can be all achieved at the same time in one application, or only some of them can be achieved. The main goals are [85, 4]:

- Confidentiality: ensuring that nobody can understand the received message excepting the authorized persons.
- Authentication: confirming and testing the identity of an entity, by assuring that the communicating entity is the one that it claims to be.
- Integrity: ensures that the received message has not been altered in any way from its original form.
- Non-Repudiation: a mechanism to prove that the sender really sent the message, so the recipient cannot claim that the message was not sent.
- Access Control: the process of preventing unauthorized use of resources, i.e., it controls who can have access to the resources, when he can access, under which restrictions and conditions the access can be granted, and finally, what is the permission level of a given access.
- Signature: a method to link or bind information to an entity or to prove the entity authorship.
- Authorization: the owner gives the authority for someone to do something on behalf of him.

Until the 1970s, cryptography was the exclusive domain of the military and governments. Nowadays, with the development of both computer and Internet technology, and with the globalization of the exchanges (Internet, email, e-business,etc.) of multimedia data such as: images, videos, audios, etc., is being used more and more widely used in many applications, including banks, commerce, education, hospitals, mobile communications, wireless sensor networks (WSNs), etc. These applications often include sensitive data to be protected before transmission, and only authorized users can have access (by using a secret key) to the useful information at any time. The confidentiality of data is mainly assured by encryption, which transforms the data from the original form (plaintext) into unreadable and non understandable form (ciphertext). A universal assumption of cryptography established by A. Kerckhoffs in the nineteenth century is that the secrecy must reside entirely in the secret key [108]. This means that the cryptanalyst has full access to the cryptographic algorithm and its implementation.
The cryptography field also includes, in addition to the encryption/decryption algorithms that can be Symmetric (Secret Key)/Asymmetric (Public-Key), protocols that deal with the application of cryptographic algorithms. The Transport Layer Security (TLS) and its predecessor, Secure Sockets Layer (SSL), which are used by web browsers to connect securely to web servers, are cryptographic protocols.
In order to secure a communication over public networks, cryptography techniques, such as symmetric and asymmetric key encryption can be used. In a few words, in the symmetric key encryption, the same encryption and decryption key is shared by both of the sender and the receiver.
Asymmetric or public key encryption has a pair of public and private keys, i.e., one at the sender and the other at the receiver. Symmetric encryption has two man advantages over asymmetric encryption: it is faster, since it does not consume large time in the encryption and the decryption processes, the key generation is less complex. However, the problem of sharing the secret key between the sender and the receiver is more complicated in symmetric encryption. In fact, public key cryptography is used to solve the problem of sharing the secret key, which is produced by symmetric key encryption. Symmetric algorithms are divided into two types: stream and block ciphers. The former operates on the plaintext as a single bit or byte or N-bit sample at a time. While the latter operates on the blocks of the plaintext, having a fixed size of Nb (NB is the block size in bits) bits.
Over the time, cryptanalysis on Internet and Internet-attached systems is rapidly growing, and attack techniques become more automated, causing significant damage. Moreover, these days, the hackers are able to penetrate systems with less information. The designer of a cryptosystem tries to increase its resistance against cryptanalysis techniques: the penalty is in increasing calculus time and the complexity structure of the designed cryptosystem. Besides, for many applications which demand both real-time performance and high level of security, such as: private multimedia data exchanged by portable devices over the wireless networks and sensitive data exchanged in wireless sensor networks, we need to design cryptosystems with some constraints, such as: low complexity, small memory, and limited energy resources. Designing cryptosystems to secure multimedia data, by taking into account the previous constraints, is a big challenge [70, 151, 106].

1.1 Cryptography based chaos

Today, based on some important properties of chaos, such as ergodicity, quasi-randomness, high sensitivity to the changes of control parameters and initial conditions, etc., chaos

is a hot research field in secured communications. Recently, a variety of chaos-based cryptosystems have been investigated. Most of them are based on the Fridrich structure [44], which is based on the traditional confusion-diffusion architecture proposed by Shannon, [8, 28, 22, 14, 21]. Compared to the conventional cryptographic algorithms (DES, 3DES, AES, etc.), the chaos-based cryptosystems provide several advantages, such as: very high security level, high speed especially in stream ciphers, increased flexibility, increased modularity, low computational overheads and computational power, and easier to be implemented. These features make them more suitable for large scale-data encryption, such as images and videos. Indeed, with a fixed block size, the advanced encryption standard (AES) is not suitable for selective video encryption and stream ciphers, [108, 36, 104, 96, 49]. For example, in selective encryption of the real-time video applications, we need to wait to have 128 bits before the starting of encryption process. This is an inconvenience for the latency of the system. Experimental results show that the chaos-based encryption algorithms can achieve security issues in efficient and adaptive ways, [2, 46, 69, 52, 15, 80, 16, 22, 129, 45, 44, 165, 84, 83]
. Generally, different algorithms encrypt different data volumes and thus get different security and efficiency (computational time). In direct encryption, the multimedia content or compressed content is encrypted directly with a novel or traditional cipher directly. In partial encryption, only some significant parts of the multimedia content are encrypted, while the others parts are lefts unencrypted. In joint compression encryption, the encryption operation is combined with a compression operation, and they are implemented simultaneously. Naturally, direct encryption often encrypts the largest data volumes, and thus, it has the highest security and lowest efficiency. Partial encryption and joint compression encryption, reduce the encrypted data volumes, and thus, get higher efficiency and necessary level of security for a given application [70]. In perceptual encryption, multimedia content is encrypted under the control of the encryption strength that determines the perceptibility of the encrypted multimedia content. A typical case of perceptual encryption is secure multimedia preview, in which the multimedia content is first encrypted with slight encryption strength and decrypted after payment. In scalable encryption, the scalable multimedia content is encrypted layer by layer, in a progressive manner, according to the significance of the layers. It can be used in secure media trans-coding. When the encrypted media content is transmitted from the Internet to bandwidth-limited mobile networks, the significant layers can be cut off directly without decryption [70].

1.2 Thesis contributions

In this study, we designed, implemented, and evaluated the performance in terms of the security level and of the structure complexity (speed of calculus) of some chaos-based cryptosystems for images and videos. All of them have a high security level, while keeping high throughput encryption compared to most chaos-based cryptosystems of the literature. We analyzed the security level of the proposed cryptosystems, relative to the different types of well-known attacks. Normally, with enough effort, any practical cryptosystem can be cryptanalyzed. The question is how much effort it takes to cryptanalyze a system. An easy way to quantify the workload of an attack is to compare it to an exhaustive search. That means the attacker tries all possible values for some target objective, like the key. If an attack requires 2^{120} steps of works, then this corresponds to an exhaustive search for 120-bit value. Any cryptosystem designed nowadays needs at least 2^{128} steps in order to resists this type of trivial attacks. Notice that, for some multimedia

encryption applications, the encryption algorithm may be regarded as secure if the cost for breaking it is no smaller than the one paid for the multimedia content. For example, in broadcasting, the news may be of no value after an hour. Thus, if the attacker cannot break the encryption algorithm over the course of an hour, then the encryption algorithm is regarded as secure in this application [41, 70], which means the cryptanalysis should be on the given time. Otherwise, it will be useless because is not a real-time cryptanalysis.

This thesis is organized in two parts. In the first part of this thesis, including chapter 2, there are introduced the concepts, the state of the art, and the common security evaluation tools and efficiency tools. In chapters 3 and 4, we propose four efficient chaos-based cryptosystems, defined on finite numbers, for real-time applications on images and videos. All of them are blocks ciphers and the first two cryptosystems are based on the substitution-permutation network (SPN). The substitution layer is achieved by a proposed modified Finite Skew Tent Map (FSTM) to overcome following problems: fixed point, restriction of the key space and limitation of mapping between plaintext and ciphertext, and vice versa. The structure of the third cryptosystem is new and efficient. It is based on two chaotic layers: a binary diffusion layer of pixels, followed by a bit-permutation layer. The permutation process is achieved by a proposed formulation of the 2-D cat map that allows an efficient implementation in C code. The fourth cryptosystem, which is 4 times faster than the other cryptosystems, with a very high security level is designed, based on a partial cryptanalysis that we performed against one of the best chaos-based cryptosystems, recently published by Zhang in 2013. In this cryptosystem, the confusion process (based on the modified 2D-cat map) and the diffusion process (based on the modified FSTM as a generator) are performed simultaneously, in a single scan of plain-image pixels. In the second part, (chapter 5), we introduce all necessary concepts and definitions to understand the High Efficiency Video Coding (HEVC) and its scalable version (SHVC), for a cryptographer researcher working on video encryption. Chapter 6 describes in details our contribution on designing two fast and secure selective chaos-based crypto-compression systems to encrypt and secure the HEVC and its scalable version SHVC. In the first crypto-compression system, we propose a new algorithm to define the encryptable bit in the bit stream of the HEVC and the SHVC. The proposed solution encrypts a set of sensitive SHVC parameters with a minimum delay and complexity overheads. The encryption process is performed at the CABAC bin string level and fulfills both constant bit rate and format compliant video encryption requirements. It preserves all SHVC functionalities, including bit stream extraction for mid-network adaptation, error resilience, and real-time SHVC decoder. The second crypto-compression system protects the Region Of Interest (ROI) of the HEVC based on the tile concept. It performs encryption at the bit stream level by encrypting all HEVC syntax elements within the ROI tiles, or a selective encryption of the ROI tiles under constant bit rate and format compliant requirements. To avoid temporal propagation of the encryption outside the ROI boundaries caused by inter prediction, the motion vectors of non ROI regions are restricted inside the non encrypted tiles in the reference frames.

2

chaos-based cryptosystems, related work and measurement tools of performances

In section 2.1 of this chapter, and also in chapter 3, we give a very brief description of the main recent chaos-based cryptosystems of the literature, which have a direct relation to our contributions. Section 2.2 presents some metrics and measurement tools of performances that are used to quantify the resistance of cryptosystem's and of crypto-compression systems against some typical attacks and for evaluating their efficiency.

2.1 Chaos-based cryptosystems, related work

In any public communication network, such as satellite, mobile-phone, and the Internet, it is almost impossible to prevent unauthorized people from eavesdropping. To use the already existed public communication networks and to maintain the secrecy, cryptographic techniques are applied [109]. The security of image and video content has become increasingly important for many applications, including video conferencing, medical imaging, industrial, military imaging systems, private multimedia messages exchanged by portable devices over the wireless networks and sensitive data exchanged in wireless sensor networks, etc.
Chaos theory has been established since 1970s [89], it is a field of mathematics and has many applications in different branches of knowledge, including engineering, physics, economics, philosophy, and biology.
A chaotic system is a deterministic system, but it has a pseudorandom behavior. It is a nonlinear system that has a large sensitivity to the initial conditions and control parameters [59]. Matthews has published a first paper on the derivation of a chaotic encryption algorithm in 1989 [146].
Today, chaos-based encryption algorithms have been widely used in image and video encryption systems. Research has shown that chaotic systems are extremely sensitive to the changes of control parameters and initial conditions. They have a pseudorandom behav-

ior for non authorized parties [8, 57, 28, 22, 14, 150, 21]. Experimental results show that the chaos-based encryption algorithm can achieve security issues in efficient and adaptive ways as compared to the classical encryption algorithms (such as DES and AES) [2, 46, 69, 52, 15, 80, 16, 49]. Thus, chaos has become a hot research topic over the last decades, and many chaos-based encryption algorithms have been recently introduced [22, 129, 45, 44, 165, 84, 83, 36].
Most of described chaos-based encryption-decryption schemes are blocks ciphers and they are based on the substitution-permutation network (SPN). In a chaos-based encryption algorithm, the substitution and the permutation operations are done in accordance with chaotic sequences to achieve the required diffusion and confusion effects.

2.1.1 Confusion and diffusion in chaos

In order to be robust against several types of attacks, any cryptosystem must achieve the confusion and diffusion effects. This has been explained in Shannon's famous paper [123] "In a strongly ideal cipher all statistics of the cryptogram are independent of the particular key used". Confusion property aims to make the statistical relationship between the cipher-image and the secret key as complex and involved as possible, whereas the diffusion property aims to make the statistical relationship between the plain image and the cipher-image as complex as possible. The confusion principle can be described as: the key should not relate to the cipher-text and each bit/byte of the cipher-text should depend on a complex mathematical relation of the key. The diffusion effect principle can be described as: each plain-text byte/bit affects many cipher text bytes/bits. The former can be achieved using chaotic maps for permutation and/or substitution, whereas the latter can be achieved using chaotic maps to transfer the single byte/bit effect to other bytes/bits.

2.1.2 State of the art

The idea of using chaos for image encryption has become a hot research topic during the last two decades, but in fact, the basic and the definition of this idea is very old and it was founded in Shannon's paper [123]. A large number of chaos-based cryptosystems for image encryption were introduced during the last decade. The most cited and important chaos-based stucture was introduced by Fridrich in his research from 1997 [44].

2.1.2.1 Fridrich cryptosystems

In 1997, a chaos-based encryption scheme was introduced by Fridrich [44, 45]. It became the core structure of the most chaos-based cryptosystems and it has been widely referenced since 1997. The general architecture of such cryptosystem is shown in Fig.2.1.

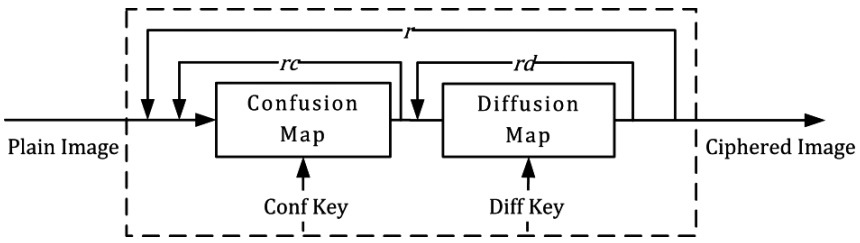


Figure 2.1: Encryption scheme of Fridrich

In Fridrich scheme, the confusion is achieved by permuting all the pixels as a whole, using one of the three types of 2-D chaotic maps, namely, Standard map, Cat map, and generalized Baker map. For example, using the 2-D BAKER chaotic map, the new byte position is given by (2.1) :

$$\begin{aligned} B(x', y') &= (2x, \tfrac{y}{2}) \qquad when\ 0 \leq x < \tfrac{1}{2} \\ B(x', y') &= (2x - 1, \tfrac{y}{2} + \tfrac{1}{2})\ when\ \tfrac{1}{2} \leq x \leq 1 \end{aligned} \tag{2.1}$$

The diffusion process changes sequentially the pixel values, in such a manner that the change to a particular pixel depends on the accumulated effect of all previous pixel values. It is implemented using the following mathematical equations:

$$\begin{aligned} v_k &= v_k + G(v_{k-1}) Mod\ 256 \\ v_{-1} &= initial\ value \end{aligned} \tag{2.2}$$

The function G is an arbitrary function of the gray level. It was chosen as a fixed random permutation, which can be implemented using a simple lookup table.

In [74], Lian et al., analyzed the security level of the Fridrich scheme. They found some weaknesses and proposed some improvements to overcome these security failures. In 2010, the Fridrich encryption algorithm was broken by Solak et.al., [132]. It has been proved that the Fridrich algorithm could be broken using a chosen cipher-text attack. Using this type of attack, some secret permutation of the algorithm has been revealed.

2.1.2.2 A symmetric image encryption scheme based on 3D chaotic cat maps

Guanrong et al, introduced a symmetric image encryption scheme based on 3D chaotic maps. They generalized the 2D into 3D chaotic maps to shuffle the pixel positions and to confuse the relationship between the plain and the cipher images. However, the speed of their algorithm is not high for real time applications. Also, the propagation error is large, since it works in cubes and each cube affects all the others [22].

2.1.2.3 Enhanced 1-D Chaotic Key-Based Algorithm for Image Encryption

In [129], a robust image encryption algorithm called Enhanced 1-D Chaotic Key-Based Algorithm for Image Encryption (ECKBA) was proposed. The algorithm performs r rounds of an SP-network on each pixel. Two PWLCM maps are used, one in the substitution process (addition modulo 256 and bitwise operations) and the other one in the permutation process. The permutation is of degree 8, and its index in the full symmetric group S8 is sorted in lexicographical Cartesian of i order (see appendix of reference [129]). The weakness of this algorithm is the error propagation caused by the used perturbation technique and the low encryption speed.

2.1.2.4 Chaotic block ciphers: from theory to practical algorithms

Masuda et al., [84] considered two classes of chaotic finite-state maps: key-dependent chaotic S-boxes and chaotic mixing transformation. They proposed two chaotic block ciphers, i.e., uniform and Feistel. In fact, they estimated bounds for differential and linear

probability to make their cryptosystems resistant to differential and linear cryptanalysis. They proposed an interesting nonlinear map, namely "Skew tent map" and its two versions as a substitution layer. In chapter 3, we present, analyze, and point out in details some existing drawbacks of their Skew tent map and we propose a modified version that overcomes the previous drawbacks.

2.1.2.5 A fast image encryption and authentication scheme based on chaotic maps

Yang et al. [159] derived a fast image encryption and authentication scheme. A key hash function is introduced to generate a 128 bit hash value from both the plain image and the secret hash keys. The hash value plays the role of a secret key for the encryption and the decryption processes, while the secret hash keys are used to authenticate the decrypted image. Permutation and substitution are performed in a single scan of the plain image pixels. The permutation process is achieved by the modified standard map and the substitution process (based on a logistic map) is done in such a way that the change of a particular pixel depends on the accumulated effect of all previous pixel values.

2.1.2.6 An image encryption scheme based on new spatiotemporal chaos

Song et al. presented a new image encryption scheme based on a new spatiotemporal chaos. From the presented security resuls, it is clear that this algorithm has good security level, but it has also a slow execution time of encryption/decryption. Indeed, they used a sort operation, which is a time consuming operation [133].

2.1.2.7 A new chaos-based fast image encryption algorithm

In their paper, Wang et al [152] introduced the idea of mixing the two layers of permutation and diffusion into a single layer. As a result, one image scanning is required instead of two scanning stages. The main contribution of this paper was to accelerate the encryption algorithm. This idea is translated into dependent permutation-diffusion layer in one of our proposed cryptosystems, described in chapter 4.
The main steps of Wang et.al., cryptosystem [152] are summarized in the following:

1. Division step: The main image is divided into a number of blocks (num), each ons is 64 pixels.
$$num = \frac{L \times P}{64} \tag{2.3}$$
Where L and P are the height and the width of the image, respectively.

2. First generation step: In this step, the used keys $K_0, K_1, ...K_{15}$ are generated, and each one is an 8-bit number.

3. Second generation step: 64 pseudorandom values are generated from the spatiotemporal chaotic map [55] (for more information on how these numbers are generated, see [152]). These pseudorandom values are defined below as $\Phi(i, j)$.

4. Diffusion step: The pixel values inside each block are modified based on the following equation:

$$\begin{aligned} &G_t(i,j) = cycl\{X,Y\} \\ &X = [P_t(i,j) \oplus \Phi(i,j) + C_t(i-1,j)] Mod\, G \\ &Y = LSB_3(C_t(i-1,(j-1) mod\, 8) \oplus \Phi(i,j)) \end{aligned} \tag{2.4}$$

Where:
$G_t(i,j)$ is the number of levels in the gray image (here is 256 levels) and is performed for all pixels in the t^{th} block.
$P_t(i,j)$ and $C_t(i,j)$ are the plain and the ciphered pixels at i and j positions, respectively.
$C_t(i-1,j)$, $C_t(i,j-1)$ and $C_t(i-1,(j-1)\ mod\ 8)$ are the previous ciphered pixels. The values of ciphered pixels at the negative indexes are set into K_{j+8}.
$cycl\{X,Y\}$ is a function to perform a left cyclic shift of X by Y.
$LSB_3(f)$ denotes the three least significant bits of f.

5. Moving block positions: A block at the position t is moved to a new position according to the following equation:

$$t_{new} = \lfloor X(0) \times num \rfloor \tag{2.5}$$

$X(0)$ is a generated value from the used chaotic map. If the t_{new} is a non-visited place, then the block is moved to this place; otherwise, the t_{new} is incremented until the new value point to a non-visited place. As not all pixels inside a given block are permuted, this step is necessary to increase the security of this model to the statistical analysis attacks.

6. Exchanging lattice values: The pixels $C(7,i)$ and $C(7,(i+d) mod\, 8)$ are exchanged, where d is calculated as:

$$d = LSB_3(C(7,0)) \tag{2.6}$$

7. First Block pixel exchange: $C_0(0,s)$ is exchanged with $C_{k_l}(7,7)$ where:
$s = LSB_3(k_l)$
$k_l = \left\lfloor \frac{[K_0 \oplus K_1 \oplus K_2 \oplus ... K_{15}] \times num}{256} \right\rfloor$

Steps, $3 \rightarrow 6$ are performed for all blocks, and repeated R rounds until the required security level is reached.

2.1.2.8 Zhang et al cryptosystem

Finally, two fast and secure cryptosystems were proposed by Zhang et al., [166]. To the best of our knowledge, these cryptosystems renowned very secure against attacks, and faster than the previous chaos-based cryptosystems. In chapter 4, first, we describe, analyze and partially break the first Zhang cryptosystem. Second, we design and realize a very robust and fast chaos-based cryptosystem.

2.2 Common and standard security evaluation tools

In this section, we give the definition of the main known attacks and then, we describe in detail the well-known statistical security measurements and evaluation tools. These measurements and tools are used in the chapters 3, 4, and 6.

2.2.1 Cryptanalysis attacks

Based on Kirchhoff's principle, the security of the encryption algorithm should only be based on the secret key and all other system parts should be known for the public [54]. A cryptanalyst tries to break the cipher without knowing the secret key, and this with several levels of difficulties based on the available resources:

1. Cipher-text only attack: A group of cipher-texts is available for the attacker; he wins the game if he finds the corresponding plain-text. It is logical that this type of complexity is based on the amount of available cipher-texts (i.e., the difficulty for the attacker is increased when the cipher-text is decreased).

2. Known plain-text attack: It is easier for the attacker to use it. To put it simply, the attacker has a group of plain-texts and the corresponding cipher-text; by the way, this type includes the previous one.

3. Chosen plain-text attack (the easiest one for the attacker): The attacker has access to the system without knowing the secret keys. Then, he has the possibility to choose a plain-text message and to encrypt it.

4. Chosen cipher-text attack: The attacker chooses the cipher-text and obtains the corresponding plain-text using the decryption system, without knowing the secret key. In this type of attack, the attacker's objective is to find the secret key.

The cryptanalysis research papers show that the security of ciphers, which is based only on statistical analysis, can be partially or completely broken down [93, 101, 150, 132, 56, 47, 9, 131, 29, 102].

2.2.2 Plain-text sensitivity attack

A cryptosystem should be sensitive to one bit change in the plain-text. This requirement is the most important to resist the known and the chosen plain-text attacks [74, 81]. In a chosen plain-text attack, more than one plain-text (with one-bit changes between them) is selected to analyze the difference between their corresponding cipher-texts. The measurement tool to test the sensitivity of any cryptosystem to these attacks is carried out based on the following procedure:

1. Select P_1 as the first plain image.

2. Change one bit in P_1 and name it as P_2. (i.e., P_1 and P_2 are exactly the same except for one bit; this bit is chosen to be located in the beginning, middle or the end of the first block; the plain-text results are calculated as an average of these three cases).

3. Both images (P_1 and P_2) are encrypted using the same secret key.

4. The previous encryption process produces two cipher images C_1 and C_2.

5. Then, a group of statistical security tests is applied on the ciphered imagesC_1 and C_2.

Most researchers use two security parameters to measure the resistance of any chaos-based cryptosystem for plain-text sensitivity attacks. These parameters are: the Number of Pixel Change Rate (NPCR) and the Unified Average Changing Intensity (UACI); they are given by the following equations, respectively:

$$NPCR = \frac{1}{L \times C \times P} \times \sum_{p=1}^{P} \sum_{i=1}^{L} \sum_{j=1}^{C} D(i,j,p) \times 100\% \tag{2.7}$$

where

$$D(i,j,p) = \begin{cases} 0, & \text{if} \quad C_1(i,j,p) = C_2(i,j,p) \\ 1, & \text{if} \quad C_1(i,j,p) \neq C_2(i,j,p) \end{cases} \tag{2.8}$$

$$UACI = \frac{1}{L \times C \times P \times 255} \times \sum_{p=1}^{P} \sum_{i=1}^{L} \sum_{j=1}^{C} |C_1(i,j,p) - C_2(i,j,p)| \times 100\% \tag{2.9}$$

In the previous equations, i, j and p are the row, column, and plane indexes of the image, respectively. L, C and P are, respectively, the length, width, and plane sizes of the image. Figure.2.2 shows a schematic view of the plain-text sensitivity attack.

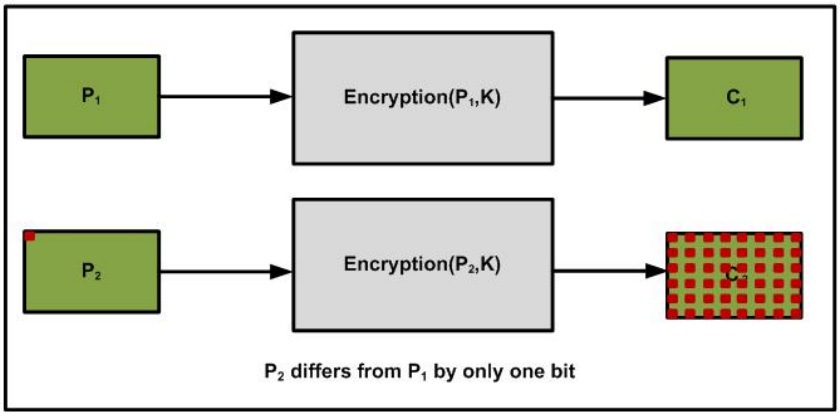


Figure 2.2: Plain-text sensitivity attack

The optimal NPCR value is 99.61%, and the optimal UACI value is 33.46% [79, 157]. The above two metrics (NPCR, UACI) are usually used to measure the resistance of a proposed cryptosystem against the differential attacks introduced by Eli Biham and Adi Shamir [17].
The previous metrics are necessary but not sufficient to ensure that the proposed cryptosystem is resistant against plain-text sensitivity attacks. Then, a new metric measurement, the Hamming Distance (HD) is used to quantify the avalanche effect. The Avalanche effect is achieved for any block cipher, when a small change (for example, flipping a single bit) in either the plain-text or the secret key, causes a drastic change in the cipher-text (e.g., half of the output bits are flipped), [82]. Therefore, this evaluation test is used to measure the resistance of any cryptosystem to the plain-text and the key

sensitivity attacks.
The HD is defined by:

$$HD(C_1, C_2) = \frac{1}{|Ib|} \sum_{K=1}^{|Ib|} (C_1(K) \oplus C_2(K)) \tag{2.10}$$

where $|Ib| = L \times C \times P \times 8$, is the size of the image in bits. The optimum HD value is 50%. A good block cipher should produce an HD close to 50% (probability of bit changes, which means that a one bit difference in plain-image will make every bit of the corresponding cipher-image change with a probability of a half [150]). Therefore, the plain-text sensitivity attack would become a useless attacking method.

2.2.3 Key sensitivity attack

Key sensitivity is extremely crucial for any cryptosystem. A cryptosystem has a high security level relative to the key sensitivity attack if a slight change in the secret key will produce a completely different ciphered image [94]. The scenario of the key sensitivity attack, which is almost identical to the plain-text sensitivity attack, is as follows:

1. Only one plain-text is used for encryption process $P_1 = P_2 = P$.
2. ***Two secret keys*** **(i.e.,** K_1 **and** K_2**)** which different in one bit are used.
3. P is encrypted using K_1 to obtain C_1.
4. P is encrypted using K_2 to obtain C_2.
5. Then, the equations of the $NPCR$, $UACI$ and HD(2.7, 2.8 and 2.10) are used to evaluate the key sensitivity attack.

Figure.2.3 shows a schematic view of the key sensitivity attack.

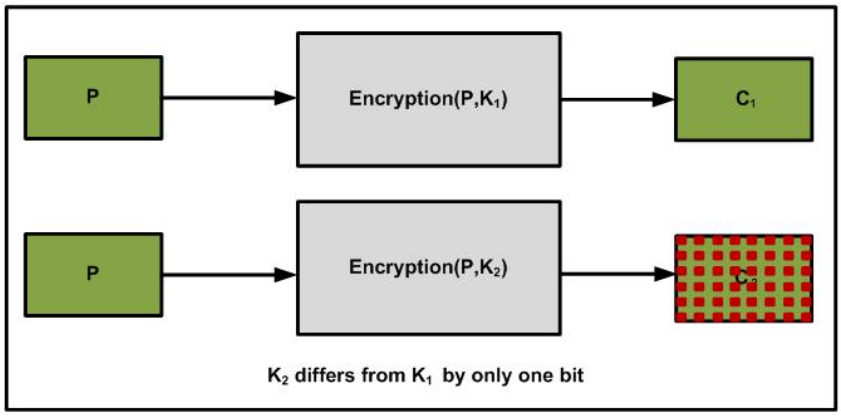


Figure 2.3: Key sensitivity attack

2.2.4 Histogram analysis

One of the most common cryptosystem attacks is the one based on statistical analysis. A cryptosystem is considered to be strong against these attacks, if the encrypted image histogram is uniformly distributed.
The visual test is necessary, but it is not sufficient. To ensure image uniformity, the chi-square test is applied (see equation (2.11)) to statistically confirm the uniformity of the histogram:

$$\chi^2_{exp} = \sum_{i=0}^{Q-1} \frac{(o_i - e_i)^2}{e_i} \tag{2.11}$$

In equation (2.11), Q is the number of levels (here $Q = 256$), o_i is the observed occurrence frequency of each color level (0-255) on the histogram of the ciphered image, and e_i is the expected occurrence frequency of the uniform distribution, given here by $e_i = \frac{L \times C \times P}{Q}$. For a secure cryptosystem, the experimental chi-square value must be less than the theoretical chi-square one, which is 293 in case of $\alpha = 0.05$ (α here is the level of significance) and $Q = 256$ [66].

2.2.5 Correlation analysis

Correlation analysis is also one of the statistical attacks that are used to cryptanalyze the cryptosystem. It should not give any information of the used secret key or any partial information of the original plain image. This means that the encrypted image should be greatly different from its original version. Correlation analysis is one of the regular and standard methods to measure this property. Indeed, it is well-known that adjacent pixels in the plain images are very redundant and correlated. So, in the encrypted images, adjacent pixels should have a redundancy and a correlation as low as possible. To test the security of any new algorithm, regarding to this type of attacks, first N pairs of adjacent pixels in vertical, horizontal, and diagonal directions are selected from the plain image and its ciphered version. Then, the following mathematical equations are used to calculate the correlation coefficient [133]:

$$\rho_{xy} = \frac{cov(x, y)}{\sqrt{D(x)}\sqrt{D(y)}} \tag{2.12}$$

Where

$$cov(x, y) = \frac{1}{N} \sum_{i=1}^{N} ([x_i - E(x)][y_i - E(y)]) \tag{2.13}$$

$$D(x) = \frac{1}{N} \sum_{i=1}^{N} (x_i - E(x))^2 \tag{2.14}$$

$$E(x) = \frac{1}{N} \sum_{i=1}^{N} (x_i) \tag{2.15}$$

In the above equations, x_i and y_i are the values of the two adjacent pixels in the plain image or the corresponding ciphered image.

2.2.6 Information entropy

The image pixel values are ranging from 0 up to 255. In a robust cipher algorithm, the occurrence probability of any pixel should be the same (or almost the same). The random behavior of the encrypted message can be evaluated using the entropy information defined by:

$$H(C) = \sum_{i=0}^{Q-1} Pro(c_i) \times \log_2 \frac{1}{Pro(c_i)} \tag{2.16}$$

Where $H(C)$ is the entropy of the ciphered image C, $Pro(c_i)$ is the occurrence number of each level ($i = 0, 1, 2...255$).
In case of equal probability levels ($Pro(c_i) = 2^{-8}$), the information entropy is maximal, $H(C) = \sum_{i=0}^{256-1} 2^{-8} \times \log_2 256 = 8$, according to (2.16).

2.2.7 Measurement of encryption quality

The difference between the frequency of occurrence for each byte level before and after encryption is called Encryption Quality (EQ), and is defined by [6]:

$$EQ = \frac{\sum_{i=0}^{255} |o_i(P) - o_i(C)|}{256} \tag{2.17}$$

where $o_i(C)$ are the observed occurrence for the byte level i in the ciphered image C, and $o_i(P)$ are the observed occurrences of the same byte level i in the plain image P.
Thus, the larger the value of EQ, the higher the level of security of the cryptosystem. For a need of comparison, it is necessary to estimate the optimal value of EQ. For that, below we propose a derivation from equation 2.17 to find the maximal value of EQ, denoted as EQ_{max}:

$$EQ_{max} = \frac{510 \times L \times C}{256^2} \tag{2.18}$$

where L and C are the line and the column of the gray image/frame. The maximal value of the EQ is derived, based on the following two assumptions:

1. For an encryption algorithm, the bad original image is the one that has a very low entropy, and the worst case is when all pixels have the same value, for example, all pixels are black or white. Therefore, the total number of occurrences of the pixel i_1 in the original image P is $o_{i_1}(P) = L \times C$, where $i_1 \in \{0, 255\}$. Moreover, the total number of occurrences of the pixel i_2 (i_2 is any pixel except i_1) in the plain image P is $o_{i_2}(P) = 0$, where $i_2 \in \{0, 255\}$ and $i_2 \neq i_1$.

2. A very secure algorithm should produce a ciphered image in which all pixels are equally distributed. Therefore, the total number of occurrences of any pixel i in the ciphered image is $o_i(C) = \frac{L \times C}{256}$, where $i \in \{0,\ 255\}$.

Based on these two assumptions and using equation (2.17), we derive EQ_{max} as follows:

$$EQ_{max} = \frac{|\frac{L \times C}{256} - L \times C| + |\frac{L \times C}{256} - 0| \times 255}{256} \tag{2.19}$$

After simplification, we find:

$$EQ_{max} = \frac{510 \times L \times C}{256^2} \tag{2.20}$$

2.2.8 Time performance

To quantify the time performance of any cryptosystem, first, we have to analyze the complexity of the algorithm in terms of the used logical and mathematical operations, and read-write memory operations. Second, the performance is determined by evaluating the running speed that can be measured by: the average encryption/decryption times, the encryption throughput, and the needed number of cycles to encrypt one byte. The encryption throughput (ET) and the number of cycles, which are required to encrypt or decrypt one byte, are defined as:

$$ET = \frac{Image_{Size}(Byte)}{Encryption_{Time}(second)} \tag{2.21}$$

$$\text{Number of cycles per Byte} = \frac{\text{CPU Speed}_{(Hertz)}}{ET_{(Byte)}} \tag{2.22}$$

The last equation permits to compare the running speed of different cryptosystems working on different platforms.

3

First Contribution: Design and Realization of Efficient Chaos-based Cryptosystems

The process of designing and developing a chaos-based cryptosystem, starts by identifying the needed requirements of the application: degree of the security level, time and memory consuming, and if it is designed to support a real time application.
In this chapter three chaos-based cryptosystems to secure transmitted images are designed and realized. The realized cryptosystems are very secure and they are faster than many chaos-based cryptosystems of the literature. All of them are defined on finite numbers and can be used for real time applications.
The first and the second cryptosystems use the same substitution process, based on Finite State Skew Tent Map (FSTM), and the same process of pixel-permutation, based on 2-D cat map. The main difference between these two cryptosystems is that, the first cryptosystem uses a pre-diffusion process to enhance and accelerate the process of diffusion while the second cryptosystem uses a hash function. The hash function generate the secret key of the chaotic generator that provides the dynamic keys for the substitution-permutation layers, and the secret hash key is used to authenticate the decrypted image. The first cryptosystem is faster than the second one, while the later is very immune against chosen plaintext attack. The structure of the third proposed cryptosystem is new and efficient. It is based on two chaotic layers: a binary diffusion layer of pixels, followed by a bit-permutation layer. Moreover, the permutation process is achieved by a proposed formulation of the 2-D cat map that allows an efficient implementation in C code. The three realized cryptosystems use an implemented simple version of a chaotic generator published by El Assad et Noura in a Patent [37]. Below we describe in details the realized three cryptosystems. The first cryptosystem (Crypto-A) is introduced in details in section 3.1, [38]. the implemented chaos-based generator for all cryptosystems is presented in section (3.1.2). The second cryptosystem, (Crypto-B) is presented in section 3.2, [39]. The third cryptosystem (Crypto-C) is described in details in section 3.3, [35]. A comparative study of performance, in terms of security and calculation time, of the three proposed cryptosystems, is given in the section 3.4, before concluding in section 3.5.

3.1 Cryptosystem-A: Chaos-based substitution permutation network

The proposed cryptosystem consists of four components: a substitution layer, a diffusion layer, a permutation layer, and a chaotic generator. The structure of the cryptosystem is shown in Figure.3.1, for the encryption part and in Figure.3.4, for the decryption part.

3.1.1 Encryption structure

During the encryption process, the plain image is divided into N blocks, with $N = imagesize/blocksize$, where the block size value is chosen to be a square value. The image size must be a multiple of the block size, so, in case of that image size is not a multiple of the block size, we pad the last block by $(block\ size - mod\ (image\ size,\ block\ size))$ bytes. The divided blocks are ciphered one by one by applying the substitution layer on the block for rs times to achieve the confusion effect. Next the diffusion layer is used to spread a single byte effect to the other bytes in the same block, this layer is repeated rd times. Then the output of this layer is forwarded to the permutation layer to exchange the byte position inside the same block to add more confusion effect. The permutation layer also is repeated rp times. These three layers are repeated r rounds to get the required security level. For each round r, new necessary dynamic keys are produced by a robust chaotic generator [37]

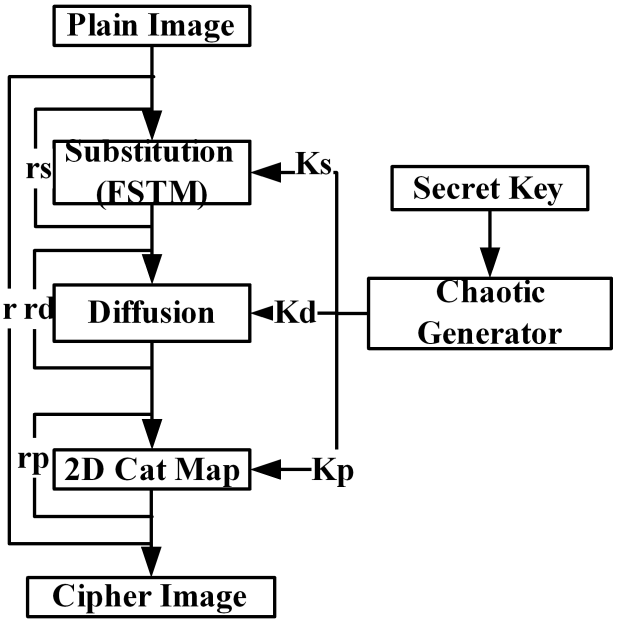


Figure 3.1: Encryption part of Crypto-A

3.1.1.1 Substitution layer

Substitution is the process of replacing a single byte value by another byte value based on a mathematical equation. The substitution layer is based here on the Finite State Skew Tent Map (FSTM). This layer performs a nonlinear transformation in the image data, and so the cryptosystem becomes more resistant against differential cryptanalysis attacks.

This layer works on byte-by-byte transformation and it changes the value of the byte depending on a chaotic parameter which comes from a robust chaotic generator. The mathematical model of the FSTM as one-to-one map for the substitution is introduced by[84, 83]:

$$Y = F_A(X) = \begin{cases} \lceil \frac{Q}{A} \times X \rceil & if\ 1 \leq X \leq A \\ \\ \left\lfloor \frac{Q \times (Q-X)}{Q-A} \right\rfloor + 1 & if\ A < X \leq Q \end{cases} \tag{3.1}$$

with $Q = 256$, and $X, A, Y \in \{1,\ 2,\ ...,\ Q\}$, note that, the value 0 is not included.

The inverse of equation (3.1) was calculated as:

$$X = {F_A}^{-1}(Y) = \begin{cases} X_1 & if\ m(Y) = Y\ and\ \frac{X_1}{A} > \frac{Q-X_2}{Q-A} \\ X_2 & if\ m(Y) = Y\ and\ \frac{X_1}{A} \leq \frac{Q-X_2}{Q-A} \\ X_1 & if\ m(Y) = Y + 1 \end{cases} \tag{3.2}$$

Where

$$X_1 = \left\lfloor \frac{A \times Y}{Q} \right\rfloor \tag{3.3}$$

$$X_2 = \left\lceil (\frac{A}{Q} - 1) \times Y + Q \right\rceil \tag{3.4}$$

$$m(Y) = Y + \left\lfloor \frac{A \times Y}{Q} \right\rfloor - \left\lceil \frac{A \times Y}{Q} \right\rceil + 1 \tag{3.5}$$

Note that the inverse substitution equation is introduced in this section because they are required to describe the FSTM and its modifications.

Shifted Masuda equations

To use the previous equations for image encryption/decryption process where a byte is represented in eight bits, the authors include 0 and exclude Q from the domain. Then, the obtained equations are:

$$Y = F_A(X) = \begin{cases} \lceil \frac{Q}{A} \times (X+1) \rceil - 1 & if\ 0 \leq X < A \\ \\ \left\lfloor \frac{Q \times (Q-X-1)}{Q-A} \right\rfloor & if\ A \leq X < Q \end{cases} \tag{3.6}$$

with $Q = 256$, $X,\ Y \in \{0,\ 1,\ 2,\ ...,\ Q-1\}$, and $A \in \{1,\ 2,\ ...,\ Q\}$.

The inverse of the shifted equation (3.6) is calculated as:

$$X = {F_A}^{-1}(Y) = \begin{cases} X_1 - 1 & if\ \ m(Y+1) = Y + 1\ and\ \frac{X_1}{A} > \frac{Q-X_2}{Q-A} \\ X_2 - 1 & if\ \ m(Y+1) = Y + 1\ and\ \frac{X_1}{A} \leq \frac{Q-X_2}{Q-A} \\ X_1 - 1 & if\ \ m(Y+1) = Y + 2 \end{cases} \tag{3.7}$$

Where

$$X_1 = \left\lfloor \frac{A \times (Y+1)}{Q} \right\rfloor \tag{3.8}$$

$$X_2 = \left\lceil (\frac{A}{Q} - 1) \times (Y+1) + Q \right\rceil \tag{3.9}$$

$$m(Y+1) = Y + \left\lfloor \frac{A \times (Y+1)}{Q} \right\rfloor - \left\lfloor \frac{A \times (Y+1)}{Q} \right\rfloor + 2 \tag{3.10}$$

We found some drawbacks on the previous substitution and inverse substitution equations:

1. It is written in the original paper that $A = Q$ is one of the possible values, but in this case the second part of equation (3.6) has a division by zero.

2. For some output values (Y), it is easy to decrease the possibilities of guessing the input value (X); indeed, let take the following example:

   - To have $Y = 0$, there are only two values of the input X, these values are: $X = 0$ or $X = 255$, because:
     When $X < A$, then the first term of the equation (3.6) $\left\lceil \frac{256}{A} \times X \right\rceil - 1 = 0$, that means, $0 < \frac{256 \times X+1}{A} \leq 1$. The only solution of the previous inequality equation is possible if the term $\frac{X+1}{A}$ is less than or equal to $\frac{1}{256}$. This is only achievable for $X = 0$ and $A = 256$.
     When $A \leq X$, then the second term of the equation (3.6) $\left\lfloor \frac{256 \times (256-X-1)}{256-A} \right\rfloor = 0$, that means, $0 \leq \frac{256 \times (256-X-1)}{256-A} < 1$. The only solution of the previous inequality equation is possible if the term $\frac{256-X-1}{256-A}$ is less than $\frac{1}{256}$. This is only achievable for $X = 255$ and any given A.
   - To have $Y = 1$, for any given A, there are only three values of the input X, these values are: $X = 0$ at $A = 256$, $X = 1$ at $A \in \{128,\ 129,\ 130,\ ...,\ 256\}$ or $X = 254$ at $A \in \{1,\ 2,\ 3,\ ...,\ 127\}$.
     When $X < A$, then the first term of the equation (3.6) $\left\lceil \frac{256}{A} \times X \right\rceil - 1 = 1$, that means, $1 < \frac{256 \times X+1}{A} \leq 2$. The only solution of the previous inequality equation is possible if the term $\frac{X+1}{A}$ is less than or equal to $\frac{2}{256}$ and more than $\frac{1}{256}$. This is only achievable for $X = 0$ at $A \in \{128,\ 129,\ 130,\ ...,\ 255\}$ or for $X = 1$ at $A = 256$.
     When $A \leq X$, then the second term of the equation (3.6) $\left\lfloor \frac{256 \times (256-X-1)}{256-A} \right\rfloor = 1$, that means, $1 \leq \frac{256 \times (256-X-1)}{256-A} < 2$. The only solution of the previous inequality equation is possible if the term $\frac{256-X-1}{256-A}$ is less than $\frac{2}{256}$ and more than or equal to $\frac{1}{256}$. This is only achievable for $X = 254$ at $A \in \{1,\ 2,\ 3,\ ...,\ 127\}$.
   - To have $Y = 255$, there is only one solution which is $X = A - 1$.

3. Some input values (X) don't have the possibility to map them to all output values (Y). For example when $X = 255$ the output can be only $Y = 0$ or $Y = 255$ because:
   The first term of the equation (3.6) at $X = 255$ is only used with $A = 256$ and so the value of the term $\left\lceil \frac{256 \times (255+1)}{256} \right\rceil - 1$ is 255.
   The second term of the equation (3.6) at $X = 255$ is used when $A \leq X$ and so the value of the term $\left\lfloor \frac{256 \times (256-255-1)}{256-A} \right\rfloor$ is 0 for any given A.

To increase the security level of the above equations, the authors impose A and Q to be co-prime, so, the model become:

Third Masuda equations

$$Y = F_A(X) = \begin{cases} \left\lfloor \frac{Q}{A} \times X \right\rfloor + 1 & if \;\; 1 \leq X < A \\ Q & if \;\; X = A \\ \left\lfloor \frac{Q \times (Q-X)}{Q-A} \right\rfloor + 1 & if \;\; A < X \leq Q \end{cases} \tag{3.11}$$

The inverse of equation (3.11) was calculated as:

$$F_A{}^{-1}(Y) = \begin{cases} X_1 & if \;\; X_1 \times (Q-A) > A \times (Q-X_2) \\ X_2 & if \;\; X_1 \times (Q-A) \leq A \times (Q-X_2) \end{cases} \tag{3.12}$$

Where

$$X_1 = \left\lfloor \frac{A \times Y}{Q} \right\rfloor \tag{3.13}$$

$$X_2 = Q - \left\lfloor (1 - \frac{A}{Q}) \times Y \right\rfloor \tag{3.14}$$

with $X,\ Y\ \in \{1,\ 2,\ ...,\ Q\}$, and in this case, the size of the secret key A is restricted to 66 odd values, defined as follows (3.15):

$$K_s = A = \{51, 53, 55, ..., 117, 139, 141, 143, ..., 201\} \tag{3.15}$$

We have the following comments on the third Masuda equations:

First Comment

All used key values outside the predefined interval give less security level and in addition, some of them don't permit the inverse substitution (3.12):

Example 3.1:

When the used A values are outside the interval of (3.15), then the equation (3.11) is two to one map. Indeed, assumes $A = 64$ using (3.11).

$Y_1 = F_{64}(3) = \left\lfloor \frac{256}{64} \times 3 \right\rfloor + 1 = 13$

$Y_2 = F_{64}(247) = \left\lfloor \frac{256 \times (256-247)}{256-64} \right\rfloor + 1 = 13.$

Consequently, equation (3.11) is not invertible.

In fact, there are a large number of pairs (X_1 and X_2) that can be encrypted to the same value (i.e., $Y_1 = Y_2$) if the used A values are outside the interval of (3.15).

Second Comment

For some output values (Y), it is easy to decrease the possibilities of guessing the input value (X); indeed, let take the following example:

- To have $Y = 1$, for any given A, there is only one value $X = 256$, because:
  When $X < A$, the first term of the equation (3.11) $\left\lfloor \frac{Q}{A} \times X \right\rfloor + 1$ will never be one , Indeed, assume that the first term can be 1 then:
  $\left\lfloor \frac{256}{A} \times X \right\rfloor + 1 = 1 \implies 0 \leq \frac{256 \times X}{A} < 1$, based on the possible range of X and A parameters, this inequality does not have a solution.
  When $X > A$, the third term of the equation (3.11) $\left\lfloor \frac{Q \times (Q-X)}{Q-A} \right\rfloor + 1 = 1$, that means, $0 \leq \frac{256 \times (256-X)}{256-A} < 1$. The only solution of the previous inequality equation is possible if the term $\frac{256-X}{256-A}$ is less than $\frac{1}{256}$. This is achievable for $X = 256$ and any given A.

- To have $Y = 2$, the first term of the equation (3.11), can be 2 with probability of $\frac{32}{66}$, since:

  $\left\lfloor \frac{256}{A} \times X \right\rfloor + 1 = 2 \longrightarrow 1 \leq \frac{256}{A} \times X < 2$, this relation has only solution for $A = A_1 \in \{139, 141, 143, ..., 201\}$, at $X = 1$ where the size of A_1 is $|A_1| = 32$.
  The last term of the equation (3.11), can be 2 only at $X = 255$, with probability of $\frac{34}{66}$, since:
  $\left\lfloor \frac{256 \times (256-X)}{256-A} \right\rfloor + 1 = 2 \longrightarrow 1 \leq \frac{256 \times (256-X)}{256-A} < 2$, this relation has only solution for $A = A_2 \in \{51, 53, 55, ..., 117\}$, at $X = 255$ and for A_2 with the size of $|A_2| = 34$.
  This means that, when the output of the STM is $(Y = 2)$, then the probability $X = 1$ is $\frac{32}{66}$, and the probability $X = 255$ is $\frac{34}{66}$. For all other values of $\{X\}$ the probability is zero.

Third Comment
Some input values (X) don't have the possibility to map them to all output values Y. For example when $X = 1$ the output can be only $Y \in \{2,\ 3,\ ...\ 6\}$ because:
$Y = \left\lfloor \frac{256}{A} \times 1 \right\rfloor + 1$, so for all A values:
$1 \leq \left\lfloor \frac{256}{A} \times 1 \right\rfloor \leq 5$, then $2 \leq Y \leq 6$, with
$Pr(Y = 2) = \frac{32}{66}$.
$Pr(Y = 3) = \frac{16}{66}$.
$Pr(Y = 4) = \frac{11}{66}$.
$Pr(Y = 5) = \frac{6}{66}$.
$Pr(Y = 6) = \frac{1}{66}$.
and for all other Y values there probability is zero that means, $Pr(Y = i) = Zero$, with $i \in \{1,\ 7,\ 8,\ ...\ 255\}$.
Furthermore, using the same previous analysis, we can find that for:
$X = 2, Y \in \{3,\ 4, ...,\ 11\}$.
$X = 254, Y \in \{3,\ 4, ...,\ 10\}$.
$X = 255, Y \in \{2,\ 3,\ 4,\ 5\}$.

For overcoming the drawbacks of the first and the second Masuda equations (3.1,3.6, 3.11 and there inverse) and for expanding the size of the key, we propose the following equations for the substitution and the inverse substitution:

Equations of the modified FSTM

Modified equation of the substitution:
The modified equations of the substitution used at the encryption part is:

$$U = F_A(X) = \begin{cases} \left\lceil \frac{Q}{A} \times (X+1) \right\rceil \ Mod\, Q & if \quad 0 \leq X < A \\ \\ \left\lfloor \frac{Q \times (Q-X)}{Q-A} \right\rfloor + 1 \ Mod\, Q & if \quad A \leq X < Q \end{cases} \tag{3.16}$$

where $X, U \in \{0, 1, 2, ...255\}$, $A \in \{1, 2, ...255\}$, and $Q = 256$.
Then,

$$Y = (U + B)\ Mod\, Q \tag{3.17}$$

where B is also a dynamic key as A, supplied by the chaotic generator, and $B, Y \in \{0, 1, 2...255\}$.
So, Y can be written as:

$$Y = F_{A,\,B}(X) = \begin{cases} \left\lceil \frac{Q}{A} \times (X+1) \right\rceil + B \; Mod \, Q & if \quad 0 \leq X < A \\ \left\lfloor \frac{Q \times (Q-X)}{Q-A} \right\rfloor + 1 + B \; Mod \, Q & if \quad A \leq X < Q \end{cases} \tag{3.18}$$

where $X, B \in \{0, 1, 2...255\}$, $Q = 256$ and $A \in \{1, 2, 3...255\}$.

Modified equation of the inverse substitution:

The modified equation of the inverse substitution used at the decryption part is:

$$X = F_A^{-1}(U) = \begin{cases} A & if \quad U = 1 \\ \zeta_1 - 1 & if \quad \theta(U) = U + 1 \\ \zeta_1 - 1 & if \quad \theta(U) = U \; and \; \frac{\zeta_1}{A} > \frac{Q - \zeta_2}{Q - A} \\ \zeta_2 & if \quad \theta(U) = U \; and \; \frac{\zeta_1}{A} \leq \frac{Q - \zeta_2}{Q - A} \end{cases} \tag{3.19}$$

where

$$U = (Y - B) \; Mod \, Q \tag{3.20}$$

Remark: if $U = 0$, then in the following equations we have to replace it by $U = Q$.

$$\zeta_1 = \left\lfloor \frac{A}{Q} \times U \right\rfloor \tag{3.21}$$

$$\zeta_2 = \left\lceil (\frac{A}{Q} - 1) \times U + Q \right\rceil \tag{3.22}$$

$$\zeta_3 = \left\lceil \frac{A}{Q} \times U \right\rceil \tag{3.23}$$

$$\theta = U + \zeta_1 - \zeta_3 + 1 \tag{3.24}$$

Below we give some examples to demonstrate that all previous drawbacks are overcome:

Example 3.2:
In the encryption part, for a given A, let us take the following values for (X) and then calculate the corresponding values (Y).
$A = 64$, $X_1 = 0$, $X_2 = 1$, $X_3 = 2$, $X_4 = 3$, $X_5 = 244$, $X_6 = 245$, $X_7 = 246$ and $X_8 = 247$. Using (3.18), we obtain:
$Y_1 = F_{64,B}(0) = \left\lceil \frac{256}{64} \times 1 \right\rceil + B \; Mod \, 256 = (4 + B) \; Mod \, 256.$
$Y_2 = F_{64,B}(1) = \left\lceil \frac{256}{64} \times 2 \right\rceil + B \; Mod \, 256 = (8 + B) \; Mod \, 256.$
$Y_3 = F_{64,B}(2) = \left\lceil \frac{256}{64} \times 3 \right\rceil + B \; Mod \, 256 = (12 + B) \; Mod \, 256.$
$Y_4 = F_{64,B}(3) = \left\lceil \frac{256}{64} \times 4 \right\rceil + B \; Mod \, 256 = (16 + B) \; Mod \, 256.$
$Y_5 = F_{64,B}(244) = \left\lfloor \frac{256 \times (256-244)}{256-64} \right\rfloor + 1 + B \; Mod \, 256 = (17 + B) \; Mod \, 256.$

$Y_6 = F_{64,B}(245) = \left\lfloor \frac{256\times(256-245)}{256-64} \right\rfloor + 1 + B \; Mod \; 256 = (15 + B) \; Mod \; 256.$
$Y_7 = F_{64,B}(246) = \left\lfloor \frac{256\times(256-246)}{256-64} \right\rfloor + 1 + B \; Mod \; 256 = (14 + B) \; Mod \; 256.$
$Y_8 = F_{64,B}(247) = \left\lfloor \frac{256\times(256-247)}{256-64} \right\rfloor + 1 + B \; Mod \; 256 = (13 + B) \; Mod \; 256.$
with the same secret key A, it is clear that there are no two input values (X_1 and X_2) that give the same output (i.e., $Y_1 = Y_2$).
Now at the decryption side, for a given value $Y = Y_1$ and a given B, such that
$U_1 = (Y_1 - B) \; Mod \; 256 = 4$
then, using equations (3.19-3.24), we obtain:
$\zeta_1 = \left\lfloor \frac{64}{256} \times 4 \right\rfloor = 1$
$\zeta_2 = \left\lceil (\frac{64}{256} - 1) \times 4 + 256 \right\rceil = 253$
$\zeta_3 = \left\lceil \frac{64}{256} \times 4 \right\rceil = 1.$
$\theta = 4 + 1 - 1 + 1 = 5$. According to the equation (3.19), only the second term is valid and so the original value is $\zeta_1 - 1$ which gives $X_1 = 0$.
The same procedure are used for $Y \in \{Y_2, \; Y_3, \; Y_4\}$, such that $U \in \{8, \; 12, \; 16\}$, using the same value of B Let us consider now Y_5, and a given B, such that
$U_5 = (Y_5 - B) \; Mod \; 256 = 17$
Then using (3.19-3.24), we obtain:
$U_5 = (Y_5 - B) \; Mod \; 256 = 17$
$\zeta_1 = \left\lfloor \frac{64}{256} \times 17 \right\rfloor = 4$
$\zeta_2 = \left\lceil (\frac{64}{256} - 1) \times 17 + 256 \right\rceil = 244$
$\zeta_3 = \left\lceil \frac{64}{256} \times 17 \right\rceil = 5.$
$\theta = 17 + 4 - 5 + 1 = 17.$
According to the equation (3.19), the first three terms are false, while the last term is correct, so the original value is ζ_2 which gives $X_5 = 244$.
The same procedure are used for $Y \in \{Y_6, \; Y_7, \; Y_8\}$, such that $U \in \{15, \; 14, \; 13\}$, using the same value of B.

The remaining drawbacks are solved by introducing the parameter B which insures that all input values of (X) can be mapped to any output value (Y). Especially, we solve the problem of the critical points of the input values $X = 0$ and $X = 255$, in Masuda et.al., maps. To illustrate our says we give the following examples:
Example 3.3:
Assume $X = 0$, using equation (3.18):
$Y = \left\lceil \frac{Q}{A} \times (0 + 1) \right\rceil + B \; Mod \; Q$, then, it is clear that all Y values are achievable in this case.
Example 3.4:
Now assume $X = Q - 1$, using equation (3.18):
$Y = \left\lfloor \frac{Q\times(Q-Q+1)}{Q-A} \right\rfloor + 1 + B \; Mod \; Q = \left\lfloor \frac{Q}{Q-A} \right\rfloor + 1 + B \; Mod \; Q$, also, it is clear that all Y values are achievable.
Example 3.5:
Assume $X = A - 1$, using equation (3.18):
$Y = \left\lceil \frac{Q}{A} \times (A - 1 + 1) \right\rceil + B \; Mod \; Q = B$, also, it is clear that all Y values are achievable, since B has all possible values.
Example 3.6:
Assume $X = A$, using equation (3.18):
$Y = \left\lfloor \frac{Q\times(Q-X)}{Q-A} \right\rfloor + 1 + B \; Mod \; Q = B + 1 \; Mod \; Q$, all Y values are achievable.

In order to reduce the execution time of the substitution and the inverse substitution, equations (3.16) and (3.19), each can be implemented as a lookup table; the first one (for the substitution operation) is used to find U from X for a given A, and the second one (for the inverse substitution) is used to retrieve X from U based on a given A.
The lookup function of the equation (3.16) is:

$$U = Lookup[A, X] \tag{3.25}$$

A part of the lookup table of equation (3.16) is presented in Table (3.1).
The inverse lookup function of the equation (3.19) is:

$$X = Lookup^{-1}[A, U] \tag{3.26}$$

Also, a part of the inverse lookup table of equation (3.19) is presented in Table (3.2).

Table 3.1: Lookup table based on the equation (3.16)

A \ X	0	1	2	.	.	.	149	.	.	.	255
1	0	1	255	.	.	.	.	.	.	.	2
2	128	0	1	.	.	.	.	.	.	.	2
3	86	171	0	.	.	.	109	.	.	.	2
.											
.											
.											
.											
255	2	3	4	.	.	.	.	.	.	.	1

Table 3.2: Inverse lookup table based on the equation (3.19)

A \ U	0	1	2	.	.	.	109	.	.	.	255
1	0	1	255	.	.	.	.	.	.	.	2
2	1	2	255	.	.	.	.	.	.	.	3
3	2	3	255	.	.	.	149	.	.	.	4
.											
.											
.											
.											
255	254	255	0	.	.	.	.	.	.	.	253

Example 3.7: Substitution operation:
For $X = 149$, $A = 3$, then by using equation (3.16) or lookup Table (3.1), we find $U = 109$:
Now, with, $B = 19$, then, $Y = 128$ based on equation (3.17) or (3.19).
Inverse Substitution operation:
With $Y = 128$, $A = 3$ and $B = 19$, we find $U = 109$ by equation (3.20), then by using equation (3.19) or the inverse lookup Table (3.2), we find $X = 149$:

Moreover, to increase the security level of our modified substitution/inverse substitution equations we can choose Q value to be prime (i.e., $Q = 257$) and shift all parameters to meet this modification, without restricted the size of the key A. We are working on this enhancement.

Structure of the dynamic key

The structure of the dynamic key during the substitution process and the inverse one is:

$$K_s = \left[K_{s0} \parallel K_{s1} \parallel K_{s2} \parallel \cdots \parallel K_{sr-1}\right] \tag{3.27}$$

Where r is the number of rounds of each block, and inside each round the substitution layer is repeated rs times. The proposed chaotic generator produces a 32 bit samples (this generator is described in section 3.1.2), whereas, the substitution layer key is 8 bit for A and 8 bits for B. Each sample of the implemented chaotic generator is used to provide the required A and B bits for one encryption round.

In terms of time complexity, the both versions (Crypto-A1:implement the substitution layer based on a predefined lookup table, Crypto-A2:implement the substitution layer based on a mathematical calculations without using lookup table) are evaluated, whereas the security evaluation results are identical for the both versions.

3.1.1.2 Pre-diffusion

Diffusion is defined as the process of spreading single bit/byte effect over other bits/byte(s). It is needful to make the relationship between the plain and the corresponding ciphered image as complex as possible [123]. The permutation layer alone is not sufficient for diffusion effect, for that and to transfer the diffusion effect of each byte we introduce a new simple and fast diffusion layer according to equation (3.28):

$$\begin{aligned} &Z_i = Mod\left((Y_i + Z_{i-1}), Q\right) \\ &Z_i = (Y_i + Z_{i-1})\ AND\ (Q-1) \\ &i = 0, 1, 2...M \times M - 1 \end{aligned} \tag{3.28}$$

Where Z_i is the current diffused byte, Z_{i-1} is the previous diffused byte, Y_i is the current substituted byte (the output of equation 3.18), M is the square root of the block size, AND is the logical and operator, Mod is the modular operator, regarding to the first byte (i.e., at $i = 0$) Z_{-1} is equal to initial value (a dynamic key value) supported by the implemented chaotic generator.

In our implementation, to speed up the encryption process, we replace the modulus operation by the AND operation according to the following mathematical rule:

$$\begin{aligned} &Mod(A, B) = A\ AND\ (B-1)\ if\ B = 2^i \\ &i = 1, 2, 3, ... \end{aligned} \tag{3.29}$$

Where i is any integer value. in our implementation $i = 8$ since each image pixel is represented by 8 bits. Before applying the permutation process, each block is converted onto a square matrix of size $M \times M$.

3.1.1.3 Permutation layer

The permutation process changes the pixel position inside the block/image under the test without changing its value. In our proposed cryptosystem, the permutation process is

based on the modified version of the basic 2-D Cat Map, given by equation (3.30), [143, 45]:

$$\begin{bmatrix} i_n \\ j_n \end{bmatrix} = Mod\left(\begin{bmatrix} 1 & u \\ v & 1+u\times v \end{bmatrix} \begin{bmatrix} i \\ j \end{bmatrix} + \begin{bmatrix} r_x + r_y \\ r_y \end{bmatrix}, \begin{bmatrix} M \\ M \end{bmatrix} \right) \tag{3.30}$$

Where $(i,\ j)$ and $(i_n,\ j_n)$ are the original and the permuted byte-positions of the data pixel-matrix of the size $M \times M$; and $u,\ v,\ r_x$, and r_y are the dynamic system parameters in the range of $[0, M-1]$. The last two parameters are added to the basic 2-D cat map model to overcome the fixed point problem and to extend the key space. The structure of the dynamic keys is:

$$K_p = \left[K_{p0} \parallel K_{p1} \parallel K_{p2} \parallel \cdots \parallel K_{p_{r-1}}\right] \tag{3.31}$$

$$K_{p_j} = \left[u_j \parallel v_j \parallel r_{xj} \parallel r_{y_j}\right] \tag{3.32}$$

The required dynamic keys bits for the permutation layer, supplied by the chaotic generator, are giving by the following equation:

$$4 \times q \times r \tag{3.33}$$

Where $q = log(M)$, is the number of required bits for each sub dynamic key.

We implemented the modified 2-D cat map, as done usually in the literature, by swap operation as in algorithm 1, and also by copy operation as in algorithm 2. Indeed, since the 2-D cat map is one-to-one function (bijective), which means every point of the square matrix can be transferred to exactly one unique point, from this fact we can replace the swap operation by the copy operation to reduce the execution time.
The swap code is shown in algorithm.1 for the encryption side:

Algorithm 1 Permutation steps of the 2-D cat map based on swap operation

```
for k = 0: rp − 1 : step = 1 do
  for i = 0: M − 1 : step = 1 do
    for j = 0: M − 1 : step = 1 do
      i_n = (i + u[k] × j + r_i[k] + r_j[k]) Mod M
      j_n = (v[k] × i + (1 + v[k] × u[k]) × j + r_j[k]) Mod M
      Temp = Data_matrix(i, j)
      Data_matrix(i, j) = Data_matrix(i_n, j_n)
      Data_matrix(i_n, j_n) = Temp
    End j
  End i
End k
```

Permutation based on copy operation instead of swap operation for fast implementation
The copy code of the 2-D cat map is shown in algorithm.2 for the encryption side: At the beginning of each code, the initial matrix content are saved in $Temp$ matrix.

3.1.2 Proposed chaotic generator

To generate a chaotic sequences, a process of discretization is used. Therefore, a small periodicity of the trajectory could appear, depending on the initial conditions and the

Algorithm 2 Permutation steps of the 2-D cat map based on copy operation

for $k = 0: rp - 1 : step = 1$ do
  for $i = 0: M - 1 : step = 1$ do
    for $j = 0: M - 1 : step = 1$ do
      $i_n = (i + u[k] \times j + r_i[k] + r_j[k]) Mod\ M$
      $j_n = (v[k] \times i + (1 + v[k] \times u[k]) \times j + r_j[k]) Mod\ M$
      $Data\_matrix(i, j) = Temp(i_n, j_n)$
    End j
  End i
End k

parameters of the generator. This is a dangerous weakness that must be avoided. The proposed generator of a discrete chaotic sequences in this cryptosystem (and in other proposed ones) is a simplified version of the one proposed by El Assad and Noura in a patent [37]. It consists of two chaotic maps (i.e., Skew tent and PWLCM), connected in parallel as shown in Fig-3.2. Each component generator is perturbed using a Linear Feedback Shift Register (LFSR). This ensures a very large periodicity for all generated sequences. The discrete Skew Tent Map and the Discrete Piece-wise Linear Chaotic Map

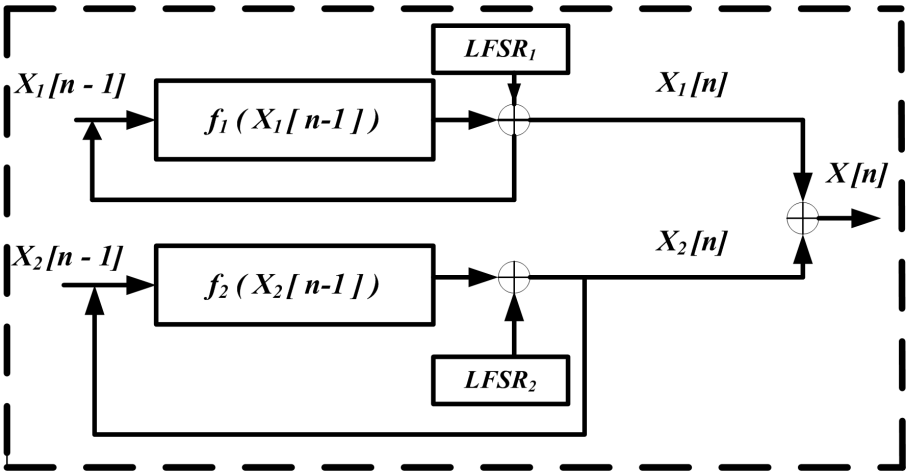


Figure 3.2: Proposed chaotic sequence generator

(PWLCM) are respectively defined as follows [84, 72]:

$$X_1[n] = F[X_1[n-1]] = \begin{cases} \left\lfloor \frac{2^N \times X_1[n-1]}{P} \right\rfloor & \text{if } 0 < X_1[n-1] < P_1 \\ 2^N - 1 & \text{if } X_1[n-1] = P_1 \\ \left\lfloor \frac{2^N \times (2^N - X_1[n-1])}{2^N - P_1} \right\rfloor & \text{if } P_1 < X_1[n-1] < 2^N \end{cases} \tag{3.34}$$

Where P_1 is the control parameter and is ranging from 1 to $2^N - 1$, and the finite precision $N = 32$ bits.

$$X_2[n] = F[X_2[n-1]] = \begin{cases} \left\lfloor \frac{2^N \times X_2[n-1]}{P_2} \right\rfloor & \text{if } 0 < X_2[n-1] < P_2 \\ \left\lfloor \frac{2^N \times (X_2[n-1] - P_2)}{2^{N-1} - P_2} \right\rfloor & \text{if } P_2 < X_2[n-1] < 2^{N-1} \\ \left\lfloor \frac{2^N \times (2^N - X_2[n-1] - P_2)}{2^{N-1} - P_2} \right\rfloor & \text{if } 2^{N-1} \leq X_2[n-1] < 2^N - P_2 \\ \left\lfloor \frac{2^N \times (2^N - X_2[n-1])}{P_2} \right\rfloor & \text{if } 2^N - P_2 \leq X_2[n-1] < 2^N - 1 \\ 2^N - 1 & \text{otherwise} \end{cases} \tag{3.35}$$

Here, the control parameter P_2 of PWLCM is ranging from 1 to $2^{(N-1)} - 1$. The proposed chaotic generator has the following cryptographic properties: random pseudo mapping, delta-like auto-correlation, nearly zero cross correlation, uniform distribution, passing empirical statistic NIST (National Institute of Standards and Technology) tests, and having a large size of the secret key. The size of the secret key is determined by four initial conditions: 2 values for the Skew tent and PWLCM maps of size N and the other for the two LFSRs, and two control parameters, i.e., P_1 (for the Skew tent) and P_2 (for the PWLCM).

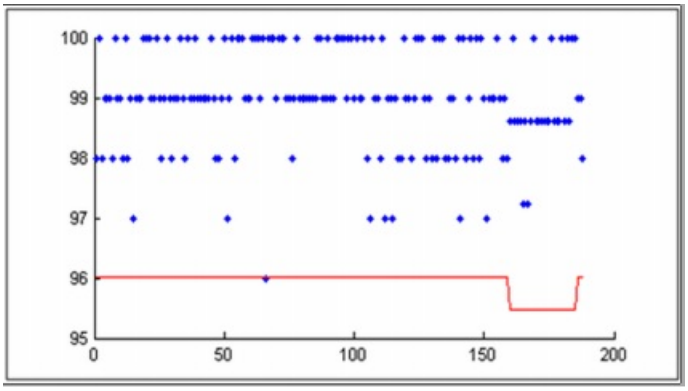


Figure 3.3: Proportion values of NIST test versus the index of the test

$$|K| = 2 \times N + |P_1| + |P_2| + |K_1| + |K_2| = 169\ bits \tag{3.36}$$

with $N = 32$, $|K_1| = 23$, $|K_2| = 19$, $|P_1| = 32$, $|P_2| = 31$. We have performed the NIST test (a battery of 188 tests) on 100 sequences, each containing one million bits. In Fig-3.3, we show the obtained proportion value of sequences passing a test, versus the indice of the test (from 1 to 188), [105]. As we can see, the proposed generator passes all the tests and therefore, it can be used in secure communication systems.

3.1.3 Decryption structure

The decryption process of the proposed cryptosystem is just the inverse operations of the encryption one (see Figure.3.4). However, the permutation layer works in reverse order for each block and the dynamic keys rp, rd, rs and r are used in reverse order. For that, all parameters of the dynamic keys must be generated and stored for all iterations and then used in reverse order to retrieve the original matrix.

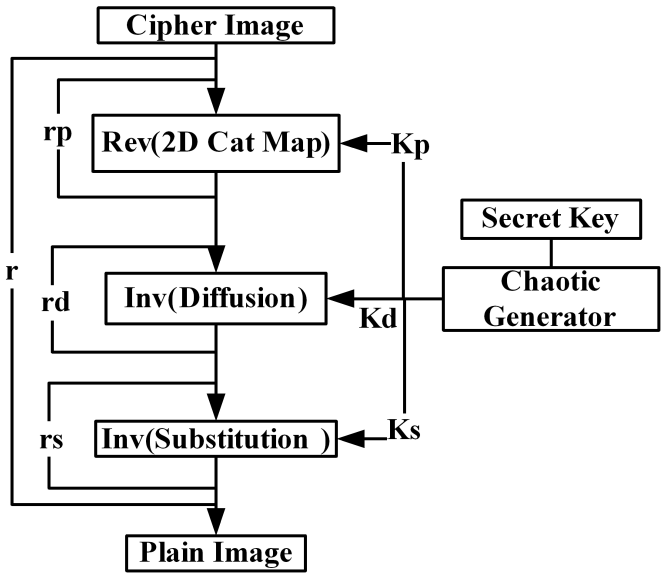


Figure 3.4: Decryption part of the proposed cryptosystem

3.1.3.1 Reverse Permutation Layer

It is clear from the equation (3.30) that the determinant of the Jacobean matrix is 1, and so the 2-D cat map process is bijective. However, because of the modulo operation, the 2-D cat map function is non-invertible one, but it is reversible by applying the permutation loop in reverse order. This means that the beginning of the reverse permutation must be applied on the last byte of the permuted block, so the dynamic keys must be also used in reverse order.
Below, we give the implementations of the reverse permutation of the 2-D cat map by the swap operation in algorithm 3 and by the copy operation in algorithm 4.

Algorithm 3 Reverse permutation steps of the 2-D cat map based on swap operation

for $k = rp - 1 : 0 : step = -1$ do
  for $i = M - 1 : 0 : step = -1$ do
    for $j = M - 1 : 0 : step = -1$ do
      $i_n = (i + u[k] \times j + r_i[k] + r_j[k]) Mod\ M$
      $j_n = (v[k] \times i + (1 + v[k] \times u[k]) \times j + r_j[k]) Mod\ M$
      $Temp = Data\_matrix(i, j)$
      $Data\_matrix(i, j) = Data\_matrix(i_n, j_n)$
      $Data\_matrix(i_n, j_n) = Temp$
    End j
  End i
End k

Before applying the inverse pre-diffusion layer, each square matrix, is converted back to block of size $M \times M$.

Algorithm 4 Reverse permutation steps of the 2-D cat map based on copy operation

for $k = rp - 1 : 0 : step = -1$ do
  for $i = M - 1 : 0 : step = -1$ do
    for $j = M - 1 : 0 : step = -1$ do
      $i_n = (i + u[k] \times j + r_i[k] + r_j[k]) Mod\ M$
      $j_n = (v[k] \times i + (1 + v[k] \times u[k]) \times j + r_j[k]) Mod\ M$
      $Data\_matrix(i_n, j_n) = Temp(i, j)$
    End j
  End i
End k

3.1.3.2 Inverse Pre-diffusion layer

The inverse diffusion layer is achieved as follows:

$$\begin{aligned} &Y_i = Mod\ ((Z_i - Z_{i-1}), Q) \\ &Y_i = (Z_i - Z_{i-1})\ AND\ (Q - 1) \\ &i = 0, 1, 2...M \times M - 1 \end{aligned} \tag{3.37}$$

Equation (3.37) is used to implement the inverse diffusion layer in a fast manner.

3.1.3.3 Inverse Substitution layer

The inverse of the modified FSTM is implemented using equations (3.19-3.24), (3.26) and by a lookup table.

Crypto-A is implemented in two versions: Crypto-A1, where the substitution/inverse substitution layers are achieved by the implemented lookup tables, and Crypto-A2, in which the substitution/inverse substitution layers are achieved by using the mathematical equations.

3.2 Cryptosystem-B: Chaos-based SPN with authentication process

The general structure of the proposed cryptosystem is presented in Figure.3.5 for the encryption and in Figure.3.6 for the decryption. The encryption schema is chaos-based substitution permutation Network, that includes a process of source authentication. The processes of substitution and permutation are similar to the ones used in cryptosystem-A, and they are implemented using the same chaotic maps: the skew tent and the 2-D cat maps. The authentication process is based on a hash function, implemented by SHA-256 [42], whose inputs are the secret hash key and the plain image and its output is the key of the chaotic generator. As we can see from the encryption scheme, a one bit change in the plain image changing the dynamic keys of the processes of substitution and permutation and then the encrypted image become completely different from the previous encrypted one. So, the confusion-diffusion properties of the cryptosystem are reached and the immunity against known-chosen plain-text attack is obtained.
The decryption process (see figure.3.6), is based on inverse permutation layer achieved by the same 2-D cat map, following by the inverse substitution achieved by the inverse

skew tent map. During this decryption process the rounds of each layer are starting in reverse order and also the dynamic keys are using in reverse order. Notice that, from the estimated plain-image and the hash secret key we can determine whether the estimated plain-image (the decrypted image) is the correct one.

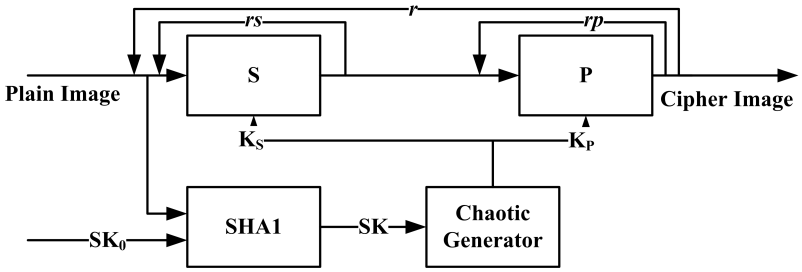


Figure 3.5: Encryption Parts of Crypto-B

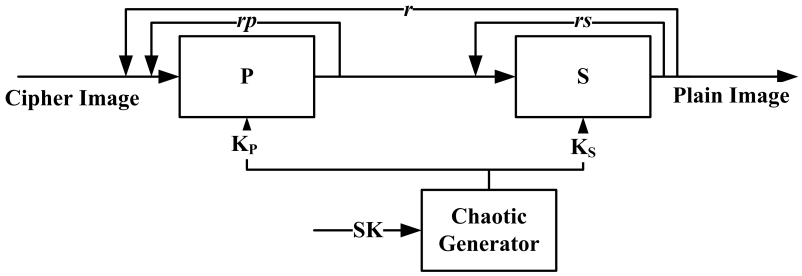


Figure 3.6: Decryption Parts of Crypto-B

The encryption steps are described in details in Figure.3.7. The plain image is divided into NB blocks, each one contains BS bytes. The first step is to generate the dynamical keys for the substitution-permutation layers. To do this, the plain image and the secret hash key are used as input for the hash function, and then the hash function produces the secret key of the chaotic generator, that provides the dynamic keys at each round of the cryptosystem. After that, for each plain block, first we use the CBC mode [58] (bit-wise XOR operation between the current plain block and the previous ciphered one, while in the first block the previous ciphered block is the initial random block IV) and then we apply the substitution layer rs times and the permutation layer for rp times. Finally these processes (CBC, substitution, permutation) are repeated r rounds and for each round a new dynamic keys are generated from the chaotic generator to be used in both layers, and so on.

Figure.3.8, shows the decryption part of the proposed cryptosystem. The decryption process is similar to the encryption one; the differences are in the inverse substitution and the reverse permutation layers, first of all, the dynamic keys are used in reverse order in both layers. Second, the permutation layer is starting the recover process from the last byte of the current block until the first one. Third, all counters will be starting in reverse order. Finally, the CBC mode function is used at the end of decryption of each block. The inverse substitution layer is starting as normal from the first to the last byte. The receiver uses the decrypted image to test the authentication source, and to test the integrity of the message.

The decryption and authentication processes suppose that the secret hash key SK_0, and the secret key SK of the chaotic generator are known (transmitted in secret manner by

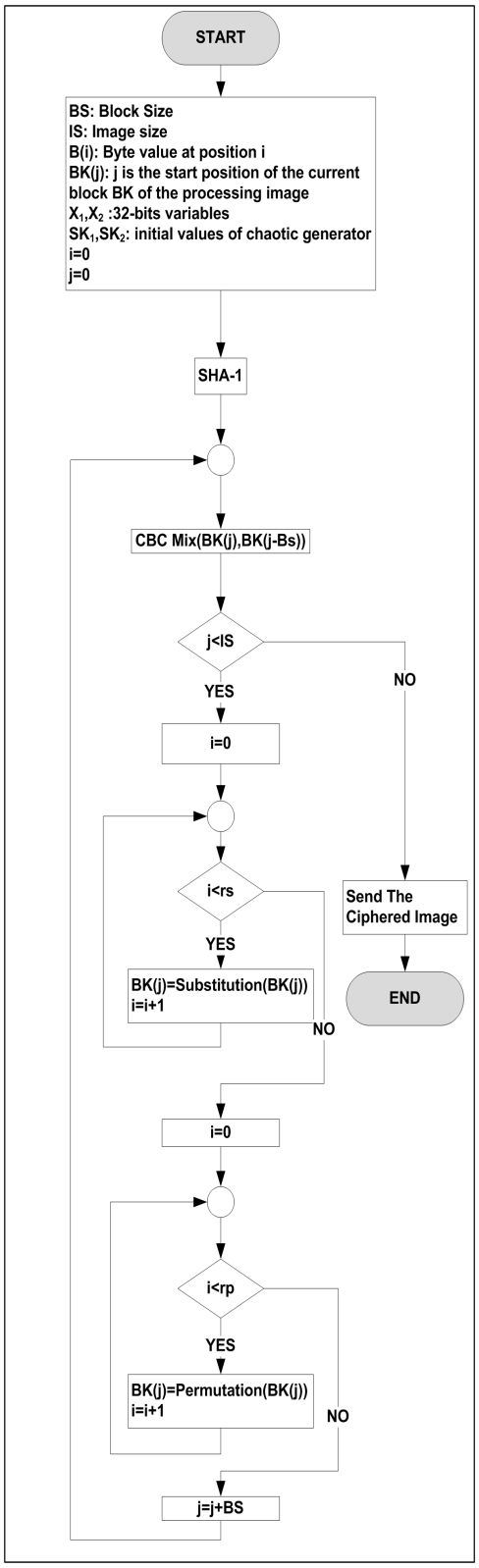


Figure 3.7: Encryption Components of Crypto-B

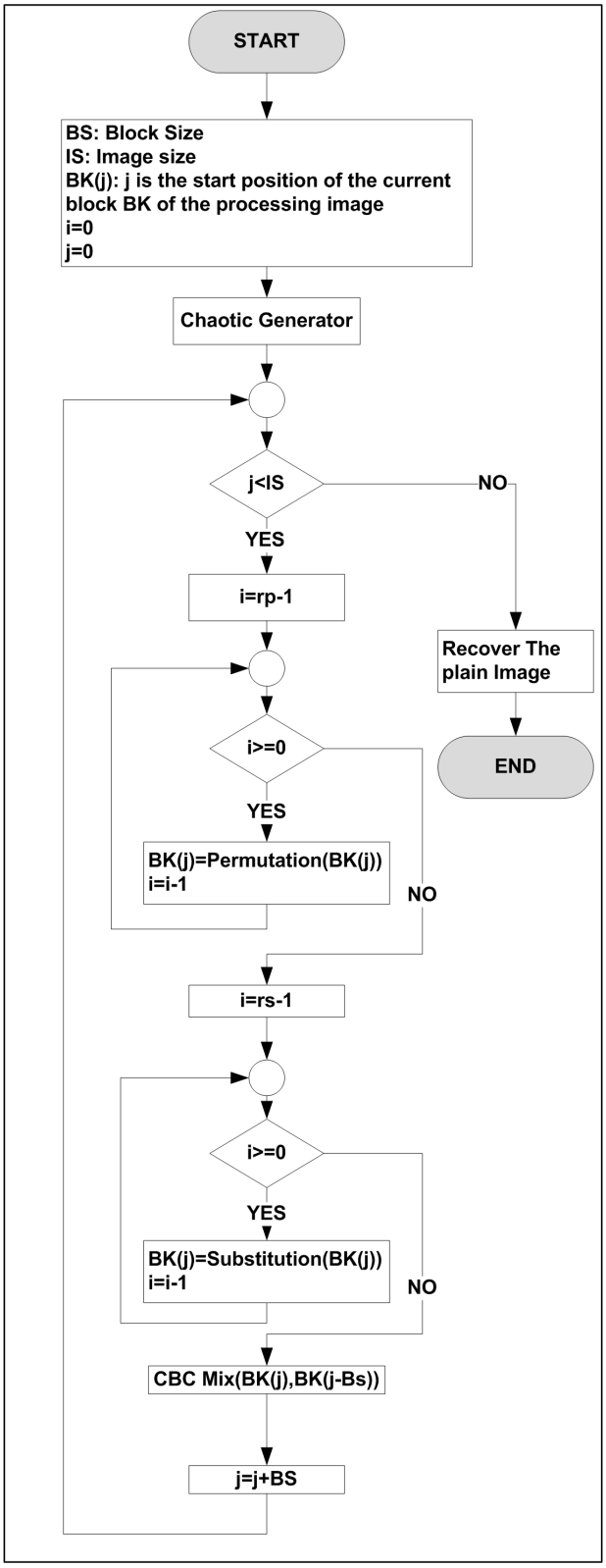


Figure 3.8: Decryption Components of Crypto-B

the sender to the receiver). The secret key of the chaotic generator is only required for decryption process, and the secret hash key is used to authenticate the sender and to test the integrity of the encrypted image according to Figure 3.9. Indeed, at the receiver, the

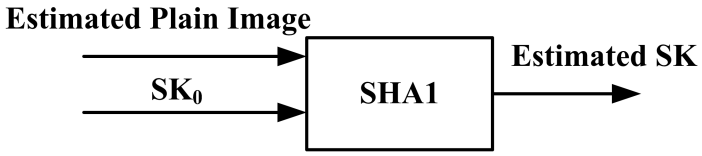


Figure 3.9: Authentication process at the decryption

same hash function is used to calculate the estimated secret key SK_e from the decrypted image and the secret hash key SK_0. If the estimated secret key SK_e is equal to the shared SK, then the integrity of the transmitted encrypted image and the authentication of the sender are valid. The required time to perform the hash operation is a part of encryption time, while, it is not the case in the decryption process. The needed time to perform a hash operation over an Lina image of size $256 \times 256 \times 3$ is $1.27ms$ [30].
Crypto-B is implemented in two versions: Crypto-B1, where the substitution/inverse substitution layers are achieved by an implemented lookup tables, and Crypto-B2 in which the substitution/inverse substitution are achieved by using the mathematical equations.

3.3 Cryptosystem-C: Binary diffusion layer and a bit-permutation layer cryptosystem

In this section, we propose a new efficient chaos-based cryptosystem structure and we analyze its performances. The cryptosystem uses a binary diffusion layer followed by a bit-permutation layer, instead of byte-permutation, to shuffle the positions of the image pixels. Moreover, the permutation layer is achieved by a new proposed formulation of the 2-D cat map that allows an efficient implementation, measured by the time complexity, in terms of arithmetic and logic operations, and also, in terms of clock cycles, of the key-dependent permutation process in comparison with the standard one. Hence, it provides a fast diffusion process to spread the influence of a single bit over the others, but at the price of a more calculation time in comparison with the pixel-permutation. The cryptosystem is implemented in CBC mode, and its speed is faster than many of chaos-based cryptosystems, while having a very high security level. The security analysis and the obtained simulation results show that the proposed cryptosystem is resistant to various types of attacks and it is efficient for hardware and software implementation (using FPGA card or an ASIC).

3.3.1 Description of the encryption process

The encryption part of the proposed cryptosystem (see Fig-3.10) consists of: Firstly, a diffusion layer based on a binary matrix of size 32×32 is considered. This process is repeated r_d times. In the next step, the data is prepared for the permutation layer using the integer to binary number converter function $Int2Bin()$. The permutation layer is based on a new modified 2-D cat map, working at the data bit level, and it gets the dynamic

parameters from the proposed chaotic generator (see section 3.1.2). It is iterated r_p times. The advantage of the bit permutation layer is that it produces a higher security than both separated permutation and substitution layers based on bytes. Indeed, theoretically, when a bit permutation layer is applied on a block, it performs, on one scan, a substitution and a diffusion operations on the bytes. As a byte contains 8 bits, then, 8 bit permutations possibly will affect 8 bytes, and then each byte (8 bit permutations) possibly will transfer the effects of the confusion and of the diffusion to other 8 bytes. As a result, the effects of the confusion and of the diffusion are better than in the case of byte permutation followed by byte substitution, because in this last case one byte affects only one byte. In the proposed cryptosystem, 6 encryption rounds based on byte permutation and on the byte substitution are needed to produce the same level of security as a process of bit-permutation which is executed in only one round. At the end of the first iteration, a binary to integer converter function $Bin2Int()$ is used to prepare the data for the next iteration. The whole process is repeated r times until it reaches the required avalanche effect and therefore the maximum security level.

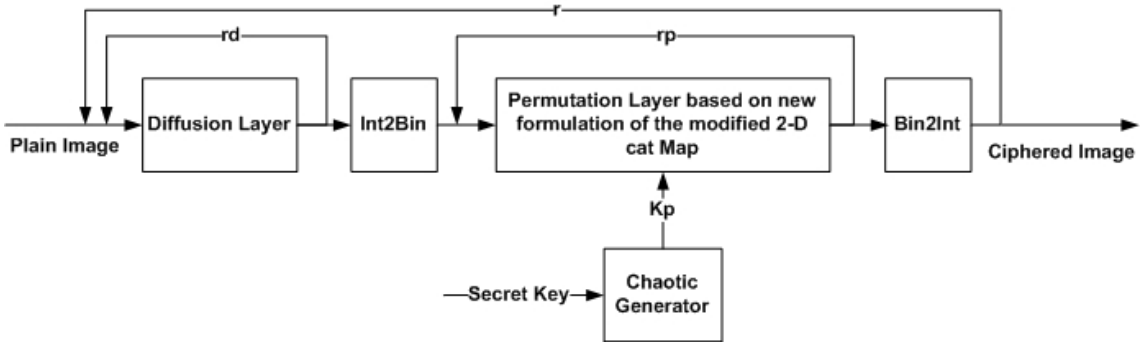


Figure 3.10: Description of the encryption process of Crypto-C

3.3.1.1 Diffusion layer

The diffusion layer works on blocks of 32 bytes each, it achieves a local diffusion, and it is defined by the following equation.

$$\begin{bmatrix} Od_0 \\ Od_1 \\ \vdots \\ Od_{31} \end{bmatrix} = \left[DM\right] \odot \begin{bmatrix} O_0 \\ O_1 \\ \vdots \\ O_{31} \end{bmatrix} \tag{3.38}$$

Where $Od_i, O_i \in [0, 255]$ are the output and the input bytes of the diffusion process, $\left[DM\right]$ is the binary diffusion square matrix of 32×32, which must be invertible but not self-invertible. Indeed, the self-invertible matrix is one of the weak points of any cryptosystem. We implemented the diffusion layer using the binary matrix proposed in [63]. The number of branches obtained from this matrix is 10 and therefore, the diffusion power is important. Actually, the maximum branch of 32×32 binary matrix is regarded to be 12, but those matrices of branch 12 do not meet the following criteria [63]: efficient implementation in 8-bit processor; secure against truncated and impossible differential

attacks; the branch number is equal or bigger than 10. The branch number B_{DM} of the diffusion layer DM is defined by:

$$B_{DM} = \min_{x \neq 0} \left[w_H(x) + w_H(DM(x)) \right] \tag{3.39}$$

where $w_H(x) = card\{i/0 \leq i < n, y_i \neq 0\}$ is the Hamming weight of n dimentional vector x. In order to define the $\odot$ matrix operator in (3.38), we write the first output Od_0 in an equivalent extended form as:
$Od_0 = O_0 \oplus O_2 \oplus O_3 \oplus O_4 \oplus O_5 \oplus O_8 \oplus O_9 \oplus O_{10} \oplus O_{12} \oplus O_{13} \oplus O_{17} \oplus O_{18} \oplus O_{19} \oplus O_{24} \oplus O_{25} \oplus O_{29} \oplus O_{31}$
Where $\oplus$ is the bitwise operator (xor).

3.3.1.2 Permutation Layer

The permutation layer here, is achieved by a new formulation of the modified 2-D cat map. For that, from the mathematical model of the modified cat map, (3.30), first we rewrite the 2-D cat map as (3.40) [143], and then we derive the new formulation, for an adequate implementation in C code.

Permutation layer based on the modified 2-D cat map
The modified 2-D cat map is given by (3.40).

$$\begin{aligned} Mln &= mod(Ml + u \times Mc + LC, MM) \\ Mcn &= mod(v \times Ml + (v \times u + 1) \times Mc + C, MM) \end{aligned} \tag{3.40}$$

Where Ml, Mc, LC, and C are square matrices $(M \times M)$ given by the following forms:

$$Ml = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 2 & 2 & \cdots & 2 \\ \vdots & \vdots & \ddots & \vdots \\ M & M & \cdot & M \end{bmatrix} \tag{3.41}$$

$$Mc = \begin{bmatrix} 1 & 2 & \cdots & M \\ 1 & 2 & \cdots & M \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 2 & \cdot & M \end{bmatrix} \tag{3.42}$$

$$LC = \begin{bmatrix} rl + rc & rl + rc & \cdots & rl + rc \\ rl + rc & rl + rc & \cdots & rl + rc \\ \vdots & \vdots & \ddots & \vdots \\ rl + rc & rl + rc & \cdots & rl + rc \end{bmatrix} \tag{3.43}$$

$$C = \begin{bmatrix} rc & rc & \cdots & rc \\ rc & rc & \cdots & rc \\ \vdots & \vdots & \ddots & \vdots \\ rc & rc & \cdots & rc \end{bmatrix} \tag{3.44}$$

and MM is a square matrix $(M \times M)$, in which all its elements have the same value M. Mln and Mcn are the permuted bit positions of the Ml and Mc matrices.

New formulation of the modified 2-D cat map for an adequate implementation in C code

The mathematical form of (3.40) takes a lot of time in matrix multiplication to find the Mln and Mcn matrices, it needs much more memory for matrix's data. For that, we derive a new formulation to reduce the consumed time and memory. The new formulation is derived from (3.40) and (3.41-3.44) as follows:

$$Mln = mod(\begin{bmatrix} 1 & 1 & \cdots & 1 \\ 2 & 2 & \cdots & 2 \\ \vdots & \vdots & \ddots & \vdots \\ M & M & \cdot & M \end{bmatrix} + \begin{bmatrix} 1 \times u & 2 \times u & \cdots & M \times u \\ 1 \times u & 2 \times u & \cdots & M \times u \\ \vdots & \vdots & \ddots & \vdots \\ 1 \times u & 2 \times u & \cdots & M \times u \end{bmatrix} + LC, MM) \tag{3.45}$$

Let us assume that:

$$\begin{aligned} Mv &= \begin{bmatrix} 1 & 2 & \cdots & M \end{bmatrix} \\ Mv &= \begin{bmatrix} Mv(0) & Mv(1) & \cdots & Mv(M-1) \end{bmatrix} \end{aligned} \tag{3.46}$$

$$\begin{aligned} UMv = u \times Mv &= \begin{bmatrix} u & 2 \times u & \cdots & M \times u \end{bmatrix} \\ UMv &= \begin{bmatrix} UMv(0) & UMv(1) & \cdots & UMv(M-1) \end{bmatrix} \end{aligned} \tag{3.47}$$

Then, (3.45) can be rewritten as:

$$\begin{aligned} Mln = mod(& \\ & \begin{bmatrix} Mv(0) & Mv(0) & \cdots & Mv(0) \\ Mv(1) & Mv(1) & \cdots & Mv(1) \\ \vdots & \vdots & \ddots & \vdots \\ Mv(M-1) & Mv(M-1) & \cdots & Mv(M-1) \end{bmatrix} + \\ & \begin{bmatrix} UMv(0) & UMv(1) & \cdots & UMv(M-1) \\ UMv(0) & UMv(1) & \cdots & UMv(M-1) \\ \vdots & \vdots & \ddots & \vdots \\ UMv(0) & UMv(1) & \cdots & UMv(M-1) \end{bmatrix} + LC, MM) \end{aligned} \tag{3.48}$$

Now, let us suppose that:

$$UMvn = \begin{bmatrix} u + rl + rc & 2u + rl + rc & \cdots & Mu + rl + rc \end{bmatrix} \tag{3.49}$$

Then

$$Mln(i, j) = mod(Mv(i) + UMvn(j), M) \tag{3.50}$$

To save memory and time in the implementation, (3.50) can be equivalently written as:

$$xrow = mod(Mv(i) + UMvn(j), M) \tag{3.51}$$

Mcn is derived in a similar procedure as above:

$$\begin{aligned} Mcn = mod(& \begin{bmatrix} 1 \times v & 1 \times v & \cdots & 1 \times v \\ 2 \times v & 2 \times v & \cdots & 2 \times v \\ \vdots & \vdots & \ddots & \vdots \\ M \times v & M \times v & \cdots & M \times v \end{bmatrix} + \\ & \begin{bmatrix} u \times v + 1 & 2 \times u \times v + 2 & \cdots & M \times u \times v + M \\ u \times v + 1 & 2 \times u \times v + 2 & \cdots & M \times u \times v + M \\ \vdots & \vdots & \ddots & \vdots \\ u \times v + 1 & 2 \times u \times v + 2 & \cdots & M \times u \times v + M \end{bmatrix} + \\ & C, MM) \end{aligned} \tag{3.52}$$

Let us assume that:

$$\begin{array}{l} VMv = v \times Mv = \begin{bmatrix} v & 2 \times v & \cdots & M \times v \end{bmatrix} \\ VMv = \begin{bmatrix} VMv(0) & VMv(1) & \cdots & VMv(M-1) \end{bmatrix} \end{array} \tag{3.53}$$

$$\begin{array}{l} UVMv = (u \times v + 1) \times Mv = \\ \begin{bmatrix} u \times v + 1 & 2 \times u \times v + 2 & \cdots & M \times u \times v + M \end{bmatrix} \\ UVMv = \begin{bmatrix} UVMv(0) & UVMv(1) & \cdots & UVMv(M-1) \end{bmatrix} \end{array} \tag{3.54}$$

Also, (3.52) can be equivalently written as:

$$\begin{array}{l} Mcn = mod(\\ \begin{bmatrix} VMv(0) & VMv(0) & \cdots & VMv(0) \\ VMv(1) & VMv(1) & \cdots & VMv(1) \\ \vdots & \vdots & \ddots & \vdots \\ VMv(M-1) & VMv(M-1) & \cdots & VMv(M-1) \end{bmatrix} + \\ \begin{bmatrix} UVMv(0) & UVMv(1) & \cdots & UVMv(M-1) \\ UVMv(0) & UVMv(1) & \cdots & UVMv(M-1) \\ \vdots & \vdots & \ddots & \vdots \\ UVMv(0) & UVMv(1) & \cdots & UVMv(M-1) \end{bmatrix} + \\ C, MM) \end{array} \tag{3.55}$$

let us suppose that:

$$UVMvn = \begin{bmatrix} vu + 1 + rc & 2uv + 2 + rc \cdots M \times u \times v + M + rc \end{bmatrix} \tag{3.56}$$

Then

$$Mcn(i,j) = mod(VMv(i) + UVMvn(j), M) \tag{3.57}$$

To save memory and time in the implementation, (3.57) can be equivalently written as:

$$ycol = mod(VMv(i) + UVMvn(j), M) \tag{3.58}$$

The pseudo C code implementation of (3.51) and (3.58) is given in algorithm-5.

where $data\_bit\_2(xrow, ycol)$ is the source matrix (the one contains the data bits of the image after diffusion), and $data\_bit\_1(i,j)$ is the destination matrix (the permuted data bits).

Time Complexity Analysis

We propose below a comparative theoretical analysis of the time complexity in terms of arithmetic, logic operations and clock cycles of our new formulation of the modified 2-D cat map and the standard cat.

Assume $X1$ is the addition operation, $X2$ is the multiplication operation and $X3$ is the modulus operation.

Remark:

1. The analysis is carried out for a block of size $M \times M$ and for one encryption round.
2. The assign operation (see line 14 in algorithm 5) is identical in both implementations. It requires M^2 assign operations and M^2 memory locations.

Algorithm 5 Pseudo C code for the implementation of the new 2-D cat map formulation

1: for $i = 0$ to $M - 1$
2: $Mv[i] = i + 1$
3: End i
4: for $k = 0$ to $r - 1$
5: for $i = 0$ to $M - 1$
6: $UMvn[i] = Mv[i] \times u[k] + rl[k] + rc[k]$
7: $VMv[i] = Mv[i] \times v[k]$
8: $UVMvn[i] = u[k] \times VMv[i] + Mv[i] + rc[k]$
9: End i
10: for $i = 0$ to $M - 1$
11: for $j = 0$ to $M - 1$
12: $xrow = mod(Mv[i] + UMvn[j], M)$
13: $ycol = mod(VMv[i] + UVMvn[j], M)$
14: $data\_bit\_1(i, j) = data\_bit\_2(xrow, ycol)$
15: End j
16: End i
17: End k

Time complexity of Eq(3.30)
The time complexity of Eq(3.30) is derived based on the following assumptions:

1. The $v \times u + 1$ value is calculated one time per block and it is saved in a register (Reg_1) to be used.

2. The $rx + ry$ is calculated one time per block and it is saved in a register Reg_2 .

The time complexity of Eq(3.30) is given by:
$TC_{Eq(3.30)} = TC_{i_n} + TC_{j_n}$
where TC_{i_n} and TC_{j_n} are the time complexity of calculating i_n and j_n parameters respectively:
$TC_{i_n} = 2M^2 \times X1 + M^2 \times X2 + M^2 \times X3$
$TC_{j_n} = 2M^2 \times X1 + 2M^2 \times X2 + M^2 \times X3$
$TC_{Eq(3.30)} = 4M^2 \times X1 + 3M^2 \times X2 + 2M^2 \times X3$
In terms of memory, Eq(3.30) needs M^2 locations.

Time complexity of the proposed implementation given by Eq(3.51) and Eq(3.58)
Based on the previous assumptions 1) and 2) and on the following two assumptions in algorithm 5:

1. The array $Mv[i]$ of lines 6, 7, 8 and 12 is replaced by $Reg_1 = i + 1$.

2. The value $rl[k] + rc[k]$ is calculated once for the loop begins at line 5 and saved in a register (Reg_2).

The time complexity of Eq(3.51) and Eq(3.58) is derived as follows:

1. Line 6: M addition operations are needed to calculate $Mv[i]$, M multiplication operations are needed to evaluate $Mv[i] \times u[k]$ and M addition operations are needed to add the value of the Reg_2. The total required operations are:
$TC_6 = 2M \times X1 + M \times X2$.

2. Line 7: Only M multiplication operations are needed to compute $Mv[i] \times v[k]$, because $Mv[i]$ was already calculated in the previous line. The total needed operations are:
   $TC_7 = M \times X2$.

3. Line 8: M multiplication operations are needed to compute $u[k] \times VMv[i]$, and $2M$ addition operations. The total required operations are:
   $TC_8 = 2M \times X1 + M \times X2$.

4. Line 12: M addition operations are needed to calculate $Mv[i]$, M^2 addition operations are needed to calculate $Mv[i] + UMvn[j]$, and M^2 modulus operations are needed. The total required operations are:
   $TC_{12} = M \times X1 + M^2 \times X1 + M^2 \times X3$.

5. Line 13: M^2 addition operations are needed to evaluate $VMv[i] + UVMvn[j]$, and M^2 modulus operations are needed. The total required operations are:
   $TC_{13} = M^2 \times X1 + M^2 \times X3$.

The time complexity $TC_{Proposed}$ is:
$TC_{Proposed} = TC_6 + TC_7 + TC_8 + TC_{12} + TC_{13}$
$TC_{Proposed} = (2M^2 + 5M) \times X1 + 3M \times X2 + 2M^2 \times X3$
Table 3.3, presents the comparative of the time complexity in terms of arithmetic and logic operations. It is clear from table 3.3 that the time complexity of the proposed implementation is less than the standard one given by Eq(3.30). Indeed, for the proposed implementation, the required multiplication operations is M times less than the standard implementation. And, the required addition operations is less than the standard implementation (for $M > 2$).

Table 3.3: Time complexity of arithmetic and logic operations

	$X1$	$X2$	$X3$	TC
Eq(3.30)	$4M^2$	$3M^2$	$2M^2$	$4M^2X1 + 3M^2X2 + 2M^2X3$
Ours	$2M^2 + 5M$	$3M$	$2M^2$	$(2M^2 + 5M)X1 + 3MX2 + 2M^2X3$

Moreover, for a fair comparison, we give in table 3.4, the time complexity in terms of clock cycles (TCC) for the operations: addition, multiplication, modulus, Read (R) and Write (W). For each operation, the TCC is calculated according to the published manual

Table 3.4: Time complexity in clock cycles for all operations

	$X1$	$X2$	$X3$	R	W	$Total$
Eq(3.30)	$2M^2$	$3M^2$	$2M^2$	0	0	$7M^2$
Ours	$M^2 + 2.5M$	$3M$	$2M^2$	$4M$	$3M$	$3M^2 + 12.5M$

of the instruction table in pages 159-160 of [43]. The Microprocessor version (Sandy Bridge Microprocessor code) of [43] is close to the used one in our simulation tests. The addition operation ($X1$) takes 0.5 clock cycle, the multiplication operation ($X2$) takes 1

clock cycle, the modulo operation ($X3$) takes 1 clock cycle, the read operation (R) takes 1 clock cycle, and the write operation (W) takes 1 clock cycle. Then, according to table 3.4, for $M > 3$, the proposed implementation consumes less number of cycles than the standard implementation. As an example, for $M = 16$, the proposed implementation requires 968 cycles while Eq(3.30) requires 1792 cycles. However, compared with the standard implementation, the proposed implementation needs of $M^2 + 3M$ locations of memories instead of M^2 and also, 3 operations of writing (in lines, 6, 7 and 8) and 4 operations of reading (in lines 8, 12 and 13), of the algorithm-5.

3.3.2 Description of the decryption process

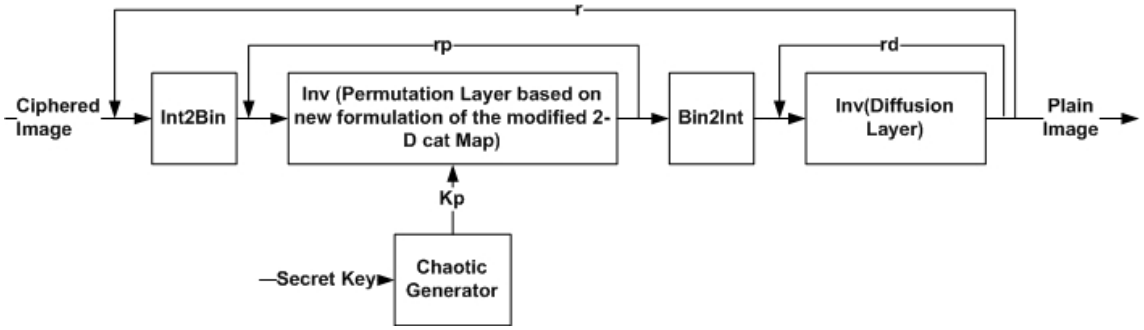


Figure 3.11: Description of the decryption process

The decryption process is almost similar to the encryption one. From Fig-3.11, first of all, the bytes in a ciphered block are converted into a stream of bits using an Int2Bin() function. Then the inverse permutation layer is applied in reverse order from the last pair of bits into the first pair. The Bin2Int() function is used to convert the resultant bits from permutation layer into a stream of bytes. Finally, the inverse diffusion layer, based on the inverse matrix, is used. All previous processes are repeated r times. Furthermore, the decryption process uses the same chaotic generator as in the encryption part.

3.3.2.1 Reverse of the new formulation based on the modified 2-D cat map

The cat map is a non-invertible function because of the modulus operation, but it is a reversible one. For that, all the parameters of the dynamic keys $Kp = u, v, rl, rc$ must be generated and stored for all iterations and then used in reverse order to retrieve the original matrix. The reverse permutation based on the derived formulas, is implemented using the pseudo C code as in algorithm-6. As we can see, the first two parts of the code are similar to those given by the permutation layer of algorithm-5, while the last part (nested loops) is achieved in reverse order.

Algorithm 6 Algorithm in pseudo C code of the reverse operation of the new permutation formula

for $i = 0$ to $M - 1$
  $Mv[i] = i + 1$
End i
for $k = r - 1$ to 0
for $i = 0$ to $M - 1$
  $UMvn[i] = Mv[i] \times u[k] + rl[k] + rc[k]$
  $VMv[i] = Mv[i] \times v[k]$
  $UVMvn[i] = u[k] \times VMv[i] + Mv[i] + rc[k]$
End i
for $i = M - 1$ to 0
  for $j = M - 1$ to 0
    $xrow = mod(Mv[i] + UMvn[j], M)$
    $ycol = mod(VMv[i] + UVMvn[j], M)$
    $data\_bit\_2(xrow, ycol) = data\_bit\_1(i, j)$
  End j
End i
End k

3.3.2.2 Inverse Diffusion Layer

The inverse diffusion layer is achieved by the inverse binary matrix as:

$$\begin{bmatrix} O_0 \\ O_1 \\ \vdots \\ O_{31} \end{bmatrix} = \left[DM\right]^{-1} \odot \begin{bmatrix} Od_0 \\ Od_1 \\ \vdots \\ Od_{31} \end{bmatrix} \tag{3.59}$$

Where $Od_i, O_i \in [0, 255]$ and $[DM]^{-1}$ is the inverse of the binary diffusion square matrix of size 32×32 [68]. We develop the calculation of the first byte O_0 from (3.59) as:
$O_0 = Od_0 \oplus Od_8 \oplus Od_9 \oplus Od_{10} \oplus Od_{16} \oplus Od_{17} \oplus Od_{19} \oplus Od_{20} \oplus Od_{22} \oplus Od_{23} \oplus Od_{24} \oplus Od_{25} \oplus Od_{27} \oplus Od_{29} \oplus Od_{31}$

3.4 Time performance and security analysis

In this section, we evaluate the performance of time and the level of security against known cryptographic attacks for the realized cryptosystems and we compare the results with some known cryptosystems in the literature. To this end, analysis methods and tools that was described in chapter 2, will be used here.

3.4.1 Performance of the speed of calculations

The performance of the speed of calculations of our proposed cryptosystems (Crypto-A1, Crypto-A2, Crypto-B1, Crypto-B2 and Crypto-C) is evaluated, on different images of different sizes, in terms of mean encryption/decryption times, encryption throughput and needed number of cycles per byte. All results are carried out by using a C compiler, on a

PC with 3.1 GHz processor Intel Core i3-2100 CPU, 4GB RAM, and Windows 7, 32-Bit Operation System.

Table 3.5: Average encryption/decryption Time of the proposed cryptosystems and some known cryptosystems (in milli-seconds)

Algorithm Name	Encryption	Decryption
Crypto-A1	6.1	6.2
Crypto-A2	9.9	32.4
Crypto-B1	7.2	5.8
Crypto-B2	10.5	31.6
Crypto-C	8.38	8.48
Zhang et al [166]	7.5	8.25
Wang et al [152]	7.79	8.39
Akhshani et al [8]	14.4	–
Wong et al [155]	15.59	16.77
Yang et al [159]	23.45	25.65
Lian et al [73]	87.25	89.5
Pareek et al [94]	160	–
Socek et al [128]	294	294.3

Table 3.5 presents the obtained average encryption/decryption times of the proposed cryptosystems (Crypto-A, Crypto-B, and Crypto-C) on Lena color image of size ($256 \times 256 \times 3$). We also give, in the same table, the average encryption/decryption times of some known chaos-based algorithms. The average time is calculated as follows: we execute the algorithm 100 times and for each time we compute the encryption/decryption times for the whole image, and then, we evaluate the average time. Table 3.6 presents the obtained results of some of the above cryptosystems in terms of encryption throughput and needed number of cycles per byte.

From these tables, it is clear that, the efficiency of proposed cryptosystems are better than some known cryptosystems, that are reputed very efficient.

3.4.2 Plain-text sensitivity attack

To achieve the plaintext sensitivity attack, we apply all steps of the procedure described in section (2.2.2) of chapter 2. Table.3.7, presents the obtained results of the average values of the NPCR, UACI and HD parameters, for 1000 different secret keys, and different images of various sizes, and this for various chaos-based cryptosystems including ours. As we can see, from this table, the obtained values of the NPCR, UACI and HD parameters by the proposed cryptosystems are very close to the optimal values, which are $NPCR_{optimal} = 99.61\%$, $UACI_{optimal} = 33.46\%$ and $HD_{optimal} = 50\%$. .

Table 3.6: Encryption throughput and number of cycles per byte of the proposed cryptosystems and some known cryptosystems

	Type	ET (MBps)	Cycles per Byte
Crypto-A1	Chaos	30.737	96.2
Crypto-A2	Chaos	18.93	156.2
Crypto-B1	Chaos	26.04	113.52
Crypto-B2	Chaos	17.85	165.62
Crypto-C	Chaos	22.37	132.2
Zhang et al [166]	Chaos	25	122
Socek et al [129]	Chaos	0.63	1977
Lian et al [73]	Chaos	2.09	772
Yang et al [159]	Chaos	7.31	287
Wong et al [155]	Chaos	7.19	417
Pareek et al [94]	NOT	0.39	2445

3.4.3 Key sensitivity attack

To evaluate the key sensitivity attack, we apply the described procedure in section (2.2.3) of chapter 2. Three parameters : $NPCR$, $UACI$ and the Hamming distance HD (2.7-2.10) are used for this evaluation. Table 3.8, presents the results of the key sensitivity attack, for some known cryptosystems including ours.
From Table 3.8, it is clear that all proposed cryptosystems have a high security level for one bit change on the secret key. This result is predicable because of the properties of chaotic signals.

3.4.4 Correlation analysis

To evaluate the security of the proposed cryptosystems regarding to the correlation analysis, we randomly selected $N = 10000$ pairs of adjacent pixels in vertical, horizontal, and diagonal directions from the plain image and its ciphered one. Then the correlation coefficient is calculated according to the equations (2.12-2.15), of section (2.2.5) of chapter 2. [1]. Figure.3.12 shows the correlation coefficients of the adjacent pixels in horizontal, ver-

[1]
H= Horizontal direction, V= Vertical direction, and D= Diagonal direction
1 = Barbara $512 \times 512 \times 1$
2 = Lena $512 \times 512 \times 1$
3 = Lena $128 \times 128 \times 1$
4 = Barbara $256 \times 256 \times 1$
5 = Lena $256 \times 256 \times 1$
6 = Boat $256 \times 256 \times 3$
7 = Boat $256 \times 256 \times 1$

Table 3.7: NPCR and UACI for the plaintext sensitivity test

Algorithm Name	**Image Name**	**Image Size**	**UACI**	**NPCR**	**HD**
Crypto-C	Lena.bmp	$512 \times 512 \times 3$	33.462	99.609	0.499912
ECKBA [129]	Lena.bmp	$512 \times 512 \times 3$	33.36	99.612	
Crypto-C	Lena.bmp	$512 \times 512 \times 1$	33.463	99.607	
Ahmed et al [2]	Lena.bmp	$512 \times 512 \times 1$	33.4	99.6	
Yang et al [159]	Lena.bmp	$512 \times 512 \times 1$	33.479	99.618	
Wong et al [155]	Lena.bmp	$512 \times 512 \times 1$	33.427	99.609	
Lian et al [73]	Lena.bmp	$512 \times 512 \times 1$	33.419	99.587	
Crypto-C	Barbara.bmp	$512 \times 512 \times 1$	33.463	99.607	
Chen et al [22]	Barbara.bmp	$512 \times 512 \times 1$	25.200	-	
Crypto-C	Barbara.bmp	$256 \times 256 \times 1$	33.452	99.597	
Behnia et al [14]	Barbara.bmp	$256 \times 256 \times 1$	33.25	0.41962	
Crypto-C	Boat.bmp	$256 \times 256 \times 1$	33.448	99.596	
Crypto-A	Boat.bmp	$256 \times 256 \times 1$	33.418	99.631	0.500009
Crypto-B	Boat.bmp	$256 \times 256 \times 1$	33.420	99.598	0.499798
Song et al [133]	Boat.bmp	$256 \times 256 \times 1$	33.453	99.625	
Akhshani et al [8]	Boat.bmp	$256 \times 256 \times 1$	33.200	-	0.499900
Crypto-C	Lena.bmp	$256 \times 256 \times 1$	33.44	99.58	

Table 3.8: NPCR and UACI for the key sensitivity test

Algorithm Name	Image Name	Image Size	UACI	NPCR	HD
Crypto-C	Barbara.bmp	$512 \times 512 \times 1$	33.465	99.609	
Chen et al [22]	Barbara.bmp	$512 \times 512 \times 1$	-	99.610	
Crypto-C	Lena.bmp	$128 \times 128 \times 1$	33.465	99.611	
Zhao et al [167]	Lena.bmp	$128 \times 128 \times 1$	-	99.568	
Crypto-C	Airplane.bmp	$512 \times 512 \times 1$	33.465	99.610	
Ahmed et al [2]	Airplane.bmp	$512 \times 512 \times 1$	33.410	99.598	
Crypto-C	Lena.bmp	$256 \times 256 \times 1$	33.463	99.609	
Song et al [133]	Lena.bmp	$256 \times 256 \times 1$	-	99.610	
Crypto-C	Lena.bmp	$512 \times 512 \times 1$	33.462	99.609	0.499941
Yang et al [159]	Lena.bmp	$512 \times 512 \times 1$	-	99.62	
Crypto-A	Camerman.bmp	$128 \times 128 \times 3$	33.463	99.609	0.500030
Crypto-B	Boat.bmp	$256 \times 256 \times 3$	33.466	99.615	0.500020

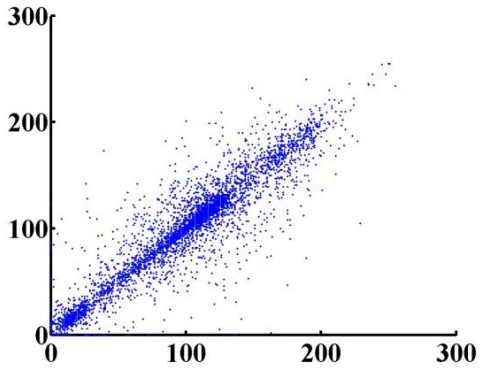
tical and diagonal directions for both Cameraman plain image and the corresponding ciphered image of size $256 \times 256 \times 3$ for the first cryptosystem Crypto-A (similar results are observed with Crypto-B and Crypto-C). In Figure.3.13 we show the resulst of the same test for both Boat plain image and the corresponding ciphered image of size $256 \times 256 \times 3$ for the second cryptosystem Crypto-B (similar results are observed also with Crypto-A and Crypto-C).

Moreover, for the proposed cryptosystems Crypto-A, Crypto-B, Crypto-C and others known cryptosystems, Table.3.9 presents the correlation values in the three directions for different images of size $256 \times 256 \times 3$. For all cryptosystems, the obtained results indicate that the correlation coefficient, in all directions, of plaintext images is close to one, and the correlation coefficient of the ciphered images is close to zero. Consequently, there is no detectable correlation between plain images and their ciphered images.

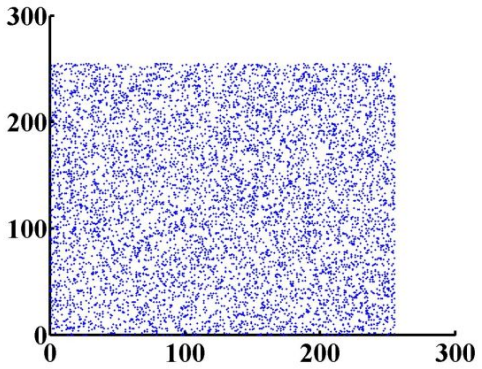
8 = Lena $512 \times 512 \times 3$
9 = Airplane $512 \times 512 \times 3$
10 = Cameraman $256 \times 256 \times 3$

Table 3.9: Correlation coefficient values of two adjacent pixels in the plain and the cipher images

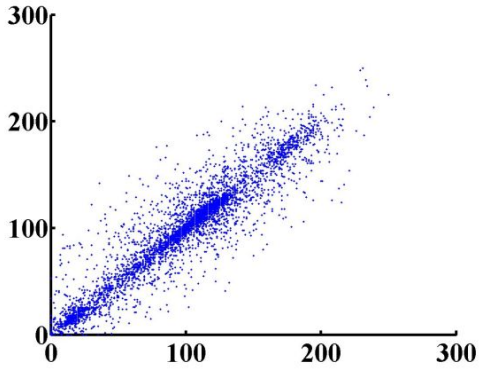
Algorithm Name	**Image**	Plain image correlation results			Ciphered image correlation results		
		H	**V**	**D**	**H**	**V**	**D**
Crypto-C	1	0.89538	0.95887	0.88304	0.00155	0.00163	0.00148
Chen et al [22]	1	0.91765	0.95415	0.90205	0.01183	0.00016	0.01480
Crypto-C	2	0.98071	0.98214	0.96547	0.00131	0.00121	0.00114
Pareek et al [94]	2	0.78200	0.59220	0.64240	0.00310	-0.00160	0.00670
Crypto-C	3	0.91019	0.96104	0.87166	-0.00622	0.00611	-0.00626
Zhao et al [167]	3	0.94800	0.88510	0.85460	0.02480	-0.00940	-0.01830
Crypto-C	4	0.92287	0.95024	0.91523	0.00317	-0.00326	-0.00309
Behnia et al [14]	4	0.95740	0.93990	0.91830	0.00380	0.00230	0.00040
Crypto-C	5	0.97165	0.98730	0.95440	0.00312	-0.00317	-0.00310
Wu et al [158]	5	-	-	-	0.00533	-0.00276	0.00166
Song et al [133]	5	0.96592	0.94658	0.92305	0.00550	0.00411	0.00021
Crypto-B	6	0.93622	0.94419	0.89243	0.00868	0.00864	0.00837
Crypto-C	7	0.94417	0.95263	0.90701	0.00320	-0.00309	-0.00306
Akhshani et al [8]	7	0.95160	0.94470	0.90590	0.00650	0.00550	0.00820
Crypto-C	8	0.99233	0.99649	0.98712	-0.00158	0.00159	-0.00147
Yang et al [159]	8	0.98022	0.98663	0.96468	-0.00209	-0.01618	0.01780
Wong et al [155]	8	0.97510	0.98892	0.96704	0.00681	0.00782	0.00323
Crypto-C	9	0.96606	0.96384	0.93674	0.00157	-0.00151	-0.00158
Ahmed et al [2]	9	0.96775	0.95753	0.93002	0.00247	0.00182	0.00038
Crypto-A	10	0.89849	0.92518	0.85137	0.01052	0.01065	0.01084



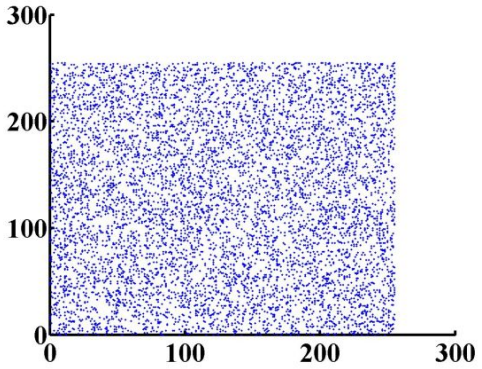
(a) Horizontal correlation of the plain image



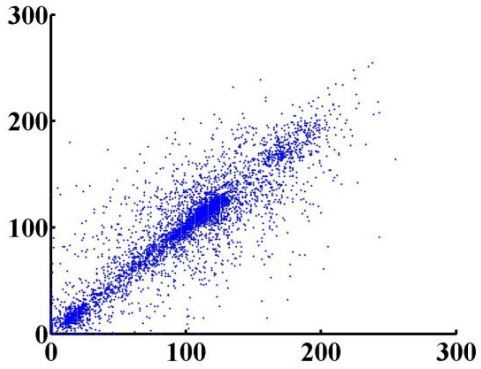
(b) Horizontal correlation of the ciphered image



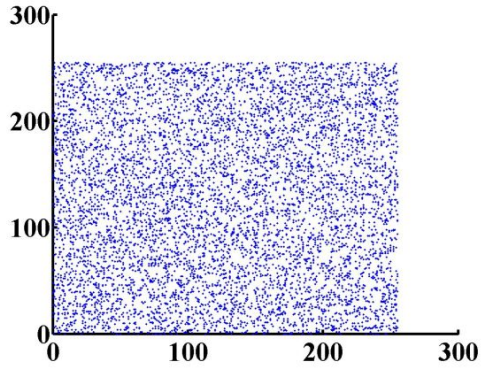
(c) Vertical correlation of the plain image



(d) Vertical correlation of the ciphered image

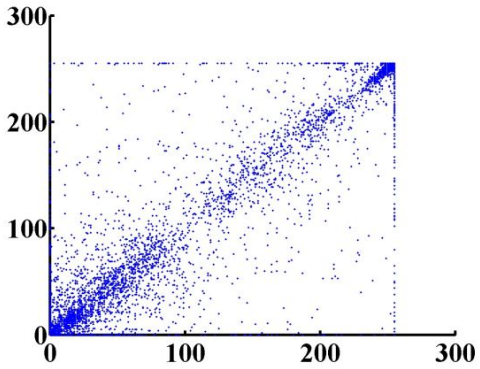


(e) Diagonal correlation of the plain image

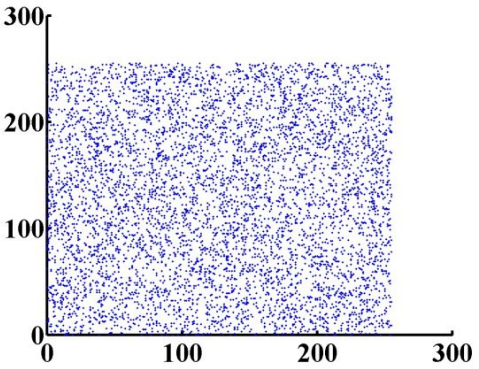


(f) Diagonal correlation of the ciphered image

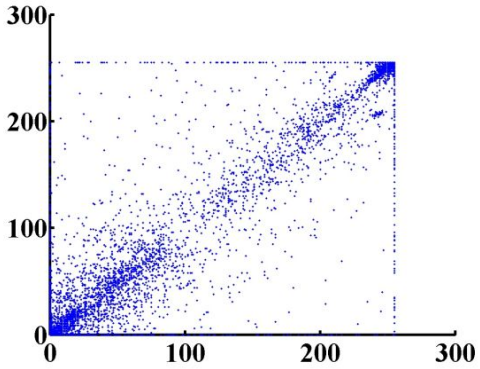
Figure 3.12: Correlation analysis of Boat and its ciphered image in three directions:Crypto-A



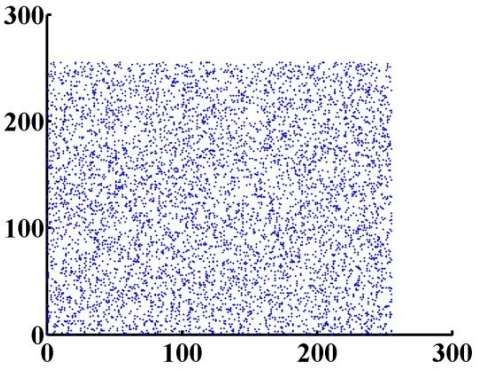
(a) Horizontal correlation of the plain image



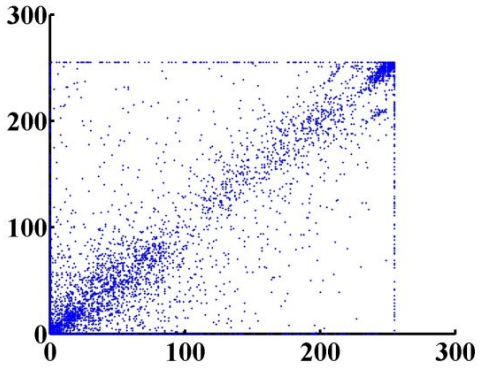
(b) Horizontal correlation of the ciphered image



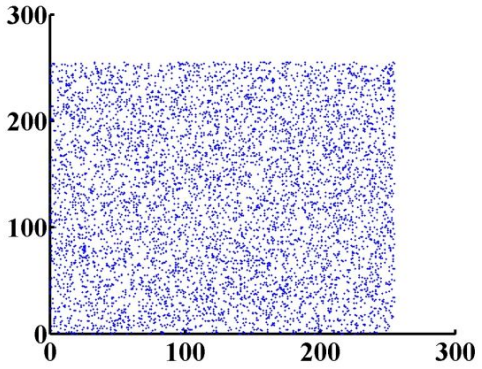
(c) Vertical correlation of the plain image



(d) Vertical correlation of the ciphered image



(e) Diagonal correlation of the plain image



(f) Diagonal correlation of the ciphered image

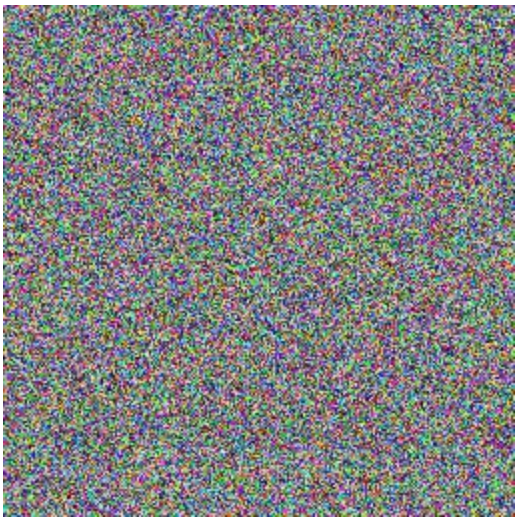
Figure 3.13: Correlation analysis of Cameraman and its ciphered image in three directions:Crypto-B

3.4.5 Histogram analysis

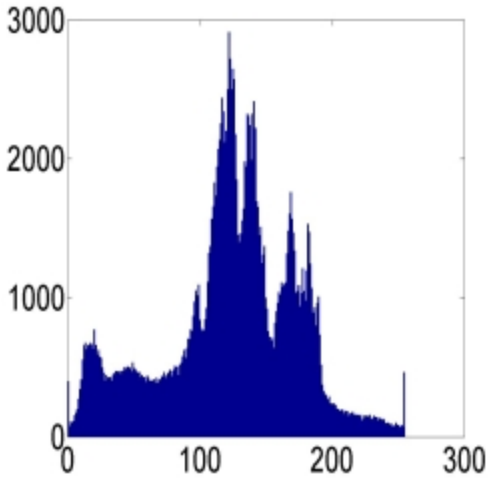
The histograms of the plain images and their ciphered ones are presented in Figures (3.14) for Crypto-A (similar results are obtained with Crypto-B), and in Figures (3.15) for Crypto-B and Crypto-C (similar results are obtained with Crypto-A). It is clear that the pixel values show a pattern in part c), which presents the histogram of the plain image of each figure, while in part d) the distribution of the pixel values of the ciphered image are almost uniform and significantly different from the histogram of the plain image.

Remark: In figure (c) of Figures (3.15), for more clarity, we limit the maximum value of the y-axis to 500. Indeed, the histogram values for all colors (except the black and white colors) are less than 500. We found that the white color is repeated more than 7500 times (16%) and the black color is repeated more than 3300 times (7%). The previous visual test of the histogram is necessary but not sufficient for confirming the uniformity of the ciphered histogram. For that, the chi-square test is applied (see (2.11)) for statistically confirming the uniformity of the histogram: For the proposed cryptosystems, we present in Table 3.10 the obtained results of the Chi square test of histograms for five ciphered images of three different natures (i.e., Boat, Airplane, Cameraman, Peppers and Parrots). All experimental values of the chi-square are less than the theoretical value, which is 293, and so, the uniformity of the histograms are verified.

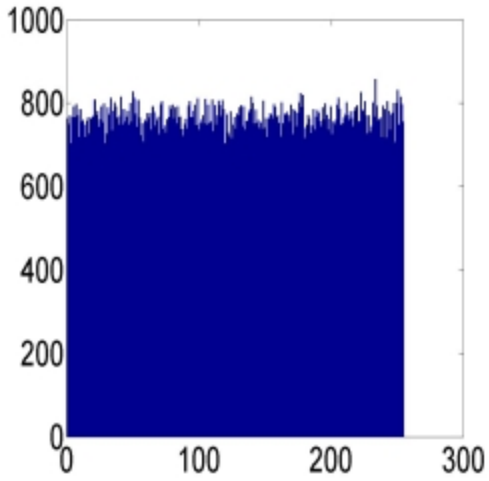
(a) Plain Boat image



(b) Ciphered Boat image



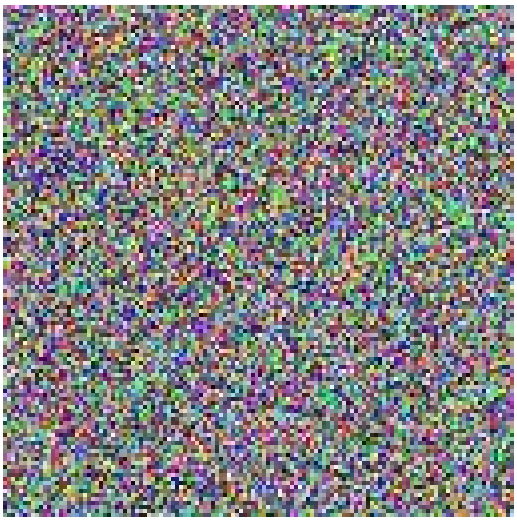
(c) Histogram of the plain Boat image



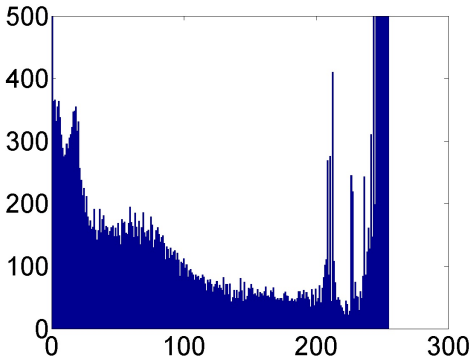
(d) Histogram of the ciphered Boat image

Figure 3.14: Plain and ciphered Boat images and their histograms:Crypto-A

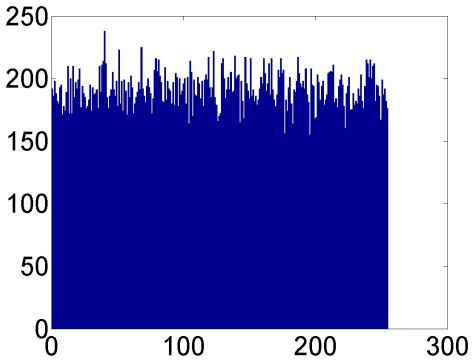
(a) Plain Cameraman image



(b) Ciphered Cameraman image



(c) Histogram of the plain Cameraman image



(d) Histogram of the ciphered Cameraman image

Figure 3.15: Plain and ciphered Cameraman images and their histograms:Crypto-B

Table 3.10: Chi-Square Results

Image name	Crypto-B Chi-square	Crypto-A Chi-square	Crypto-C Chi-square
Cameraman	255.74	255.12	/
Boat	255.43	255.12	/
Jet	253.787	253.78	/
Peppers	261.17	260.36	253.5
Parrots	260.07	259.44	/
Lena	/	/	252.1
Baboon	/	/	256.1

3.5 Conclusion

In this chapter, we designed, realized and tested three efficient chaos-based cryptosystems, defined on finite numbers, for securing images, in real-time applications. All of them are blocks ciphers and the two first cryptosystems crypto-A and Crypto-B are based on the substitution-permutation network (SPN). The substitution layer is achieved by a proposed modified Finite Skew Tent Map (FSTM) to overcome the problems of : fixed point, restriction of the key space and limitation of mapping between plaintext and ciphertext and vice versa. Crypto-B is rather used for some applications that need two security requirements: confidentiality and authentication.
The structure of the third cryptosystem is new and efficient. It is based on two chaotic layers: a binary diffusion layer of pixels, followed by a bit-permutation layer. The permutation process is achieved by a proposed formulation of the 2-D cat map that allows an efficient implementation in C code. Experimental results show that the proposed chaos-based cryptosystems are faster than many chaos based cryptosystems of the literature, while keeping a very high security level. Indeed, the proposed encryption/decryption schemes are robust against all known attacks in the literature.

4

Second Contribution: Partial Cryptanalysis of Zhang cryptosystem and design of a very fast and secure cryptosystem

In this chapter, a new category of cryptosystems, which are faster than the previous cryptosystems with a very high security level, are introduced. In such cryptosystems, the confusion and diffusion layers are combined, and then, they are performed simultaneously in a single scan of plain-image pixels. The chapter is splits into two related parts. In the first part (section (4.1), we describe and partially break one of best chaos-based cryptosystem recently published by Zhang in 2013, namely, the first cryptosystem of Zhang et al., model [166]. To that end, a mathematical model is introduced to remove the dynamic key space of the diffusion effect. In the second part (section (4.2), based on the previous analysis a very fast and very secure cryptosystem is designed, realized with two versions of the diffusion layer, and tested. The diffusion layer of the first version is achieved by a discrete Logistic map, while the diffusion layer of the second version is achieved by a finite skew tent map. In section (4.3) an example of real-time application is described. The conclusion of this chapter is presented in section (4.4).

4.1 Partial cryptanalysis of the first Zhang cryptosystem

In this section, we address some weaknesses in the first Zhang et al., algorithm "An image encryption scheme using reverse 2-dimensional chaotic map and dependent diffusion", and we propose methods for breaking it partially. Moreover, the differential attacks analysis, based on the proposed partial cryptanalysis equation, are reevaluated. As a result, the obtained average values of NPCR and UCAI parameters become small, as compared to the optimal values, and in addition, they are very low for specific pixel position attacks.

4.1.1 The first Zhang cryptosystem

In Zhang's paper [166], two cryptosystems were designed based on Fridrich's architecture. The first one consists of a dependent diffusion layer based on the reverse 2-D cat map. The second algorithm presents new mapping from a pseudo-random position to another pseudo-random one for the confusion effect. The diffusion layer in the cryptosystems is based on the logistic map. In these versions, Zhang tried to achieve the confusion and the diffusion effects sequentially. Then, the effect of one ciphered pixel is transferred to the next one and so on. From this idea, only two rounds (in the first version) and one round (in the second version) of the diffusion-confusion process are/is needed instead of many rounds of separated confusion and diffusion processes used in the traditional structures such as Fridrich cryptosystem and other cryptosystems. In the following, our work is directed to the first Zhang cryptosystem. The mathematical model of the first Zhang algorithm is:(Enc=the encryption process).

$$Enc = \begin{cases} \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & p_i \\ qi & p_i q_i + 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} (ModN) \\ ciph(x,\ y) = arr(x',\ y') \oplus f(t) \\ t = ciph(x,\ y) \end{cases} \tag{4.1}$$

The general block diagram of the first Zhang cryptosystem is shown in Figure. 4.1. It consists of the following steps iterated n times (with $n > 0$):

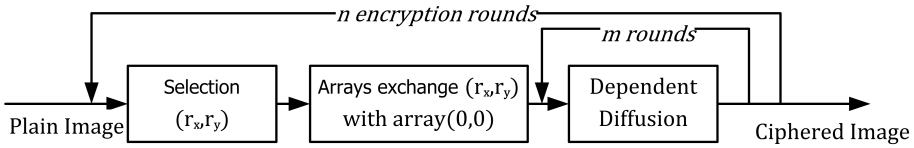


Figure 4.1: Zhang image encryption cryptosystem architecture

1. Selection: this step generates a random pair $arr({r_x}^j,\ {r_y}^j)$ from the whole image. The values of ${r_x}^j$ and ${r_y}^j$ are calculated using equation (4.2) and equation (4.3), where j is a counter ranging from 0 to $n-1$ encryption rounds.

$${r_x}^j = (SQ_1(2000 + 100 + j) \times 10^9) mod\ 512 \tag{4.2}$$

$${r_y}^j = (SQ_2(2000 + 100 + j) \times 10^9) mod\ 512 \tag{4.3}$$

2. Array exchanges: the second step is to exchange the first byte $arr(0,\ 0)$ with the random byte from the previous step $arr({r_x}^j,\ {r_y}^j)$.

3. Dependent diffusion: then the cryptosystem goes to the dependent diffusion layer for m rounds ($m = 2$ in the Zhang algorithm case), which also includes three stages.

   (a) New position estimation: in the dependent diffusion layer the first step is to calculate the new byte position $(x',\ y')$ from the old byte position $(x,\ y)$ using equation (4.4).

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & p_i \\ qi & p_i q_i + 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} (ModN) \tag{4.4}$$

where
N is the size of the square test image.
p_i and q_i are calculated using the following equations:

$$p_i = (SQ_1(2000 + i) \times 10^9) mod\ 512 \tag{4.5}$$

$$q_i = (SQ_2(2000 + i) \times 10^9) mod\ 512 \tag{4.6}$$

The variable i in the last equations is a counter ranging from 0 to $m - 1$. The two sequences SQ_1 and SQ_2 as can be seen in equations (4.2, 4.3, 4.5, and 4.6) are calculated using the following equation:

$$f(x_n) = \alpha \times x_{n-1}(1 - x_{n-1}) \tag{4.7}$$

With the initial values x_{-1}=0.12345678912345 for SQ_1, and x_{-1}=0.67856746347633 for SQ_2. The value of α is set to 3.99999.

(b) Calculation of the local ciphered pixel: the next step of the dependent diffusion layer is to calculate the $ciph(x,\ y)$ value using the following equation:

$$ciph(x,\ y) = arr(x',\ y') \oplus f(t) \tag{4.8}$$

where

$$f(t) = [\alpha(\frac{t}{1000}) \times [1 - \frac{t}{1000}] \times 1000] mod\ 256 \tag{4.9}$$

(c) Update of t: the last step of the dependent diffusion layer is to change the value of the t variable using equation (4.10).

$$t = ciph(x,\ y) \tag{4.10}$$

The initial value t_0 is defined by the following equation:

$$t_0 = [4 \times key_d \times (1 - key_d) \times 1000] mod\ 256. \tag{4.11}$$

Where
The initial value of the key_d=0.33456434300001.

4.1.2 Partial cryptanalysis of the Zhang cryptosystem

Our contribution in this work focuses on partial cryptanalysis of the first Zhang cryptosystem. Also, in the case of one encryption round and two dependent diffusion rounds, we have broken it completely. Moreover, the partial cryptanalysis equation will be used to significantly decrease the robustness of the cryptosystem for two encryption rounds and two dependent diffusion rounds. In fact, the dynamic key space of the diffusion function $f(t)$ (see equation 4.9)can be removed from the calculation of the total dynamic key space independently of the used key, because the t values are clearly presented in the ciphered image. To prove the correctness of the above assumptions, we derive the following scenario: The encrypted data of the Zhang cryptosystem are obtained by equation (4.1), then we can write the following sequence:
$ciph_0 = arr_{k'_0} \oplus f(t_0)$
$ciph_1 = arr_{k'_1} \oplus f(t_1) = arr_{k'_1} \oplus f(ciph_0)$
$ciph_2 = arr_{k'_2} \oplus f(t_2) = arr_{k'_2} \oplus f(ciph_1)$

$ciph_3 = arr_{k'_3} \oplus f(t_3) = arr_{k'_3} \oplus f(ciph_2)$
$ciph_4 = arr_{k'_4} \oplus f(t_4) = arr_{k'_4} \oplus f(ciph_3)$
$ciph_k = arr_{k'_k} \oplus f(t_k) = arr_{k'_k} \oplus f(ciph_{k-1})$
Where $arr_{k'_0}$ is the plain image of the current encryption round, as an example for $n = 1,\ m = 1$ it is the original plain image ($arr(x',\ y')$), for $n = 1,\ m = 2$ it is the encrypted image which is the output of equation (4.1) (i.e $ciph(x,\ y)$) and so on.
From the last equation in the previous sequences, we can write the main partial cryptanalysis equation of the Zhang cryptosystem as:

$$arr_{k'_k} = ciph_k \oplus f(ciph_{k-1}) \tag{4.12}$$

where $k = x \times N + y$ and $k' = x' \times N + y'$ (see equation (4.4)), $arr_{k'_k}$ is the input pixel of the last dependent diffusion round (m) in the last encryption round (n) and $ciph_k$ is the ciphered pixel. As the function $f(t)$ is known, then equation (4.12) can be used to remove the diffusion effect of the last (m and n) rounds from the ciphered pixels. This allows recovery of a permuted version of the previous ciphered image. This removal gives the attacker the possibility to:

1. Decrease the dynamic key space of the whole cryptosystem.
2. Perform partial cryptanalysis of the Zhang cryptosystem for ($n = 1$, $m = 1$) and ($n = 1$, $m = 2$).
3. Decrease the UACI and NPCR values significantly.

4.1.2.1 Decreasing the dynamic key space of the whole cryptosystem

The brute-force attack is the basic attack that can be used against any cryptosystem. It trys all possible keys until the correct one is founded. In the worst case, all possible keys in the key space are tested[125],[108]. From equation (4.4), it is clear that the key space of the 2-D cat map for the Zhang cryptosystem is N^2 for one encryption round and one dependent diffusion round (n=1, m=1) where N is the square root of the image size. In equation (4.9), the key space of the function $f(t)$ is independent of the image size, and it is 2^8 for (n=1, m=1). The brute force attack time on the Zhang cryptosystem is: $KS = (S_1 \times S_2)^{n \times m}$ where KS is the total dynamic key space, S_1 represents the dynamic key space for the standard 2-D cat map, and S_2 represents the dynamic key space for the logistic map implemented by a lookup table, then $S_1 = N^2$, $S_2 = 2^8$, and so, $KS = (N^2 \times 2^8)^{n \times m}$ For one encryption round, one dependent diffusion round (n=1, m=1), and ($N = 512$) $KS = (2^{18} \times 2^8)^1$, then, the cryptanalysis time is: $C_{Time} = 10 \times 2^{26}\ ms \approx 186.4135$ hours $= 7.77$ days.
Notice that equation (4.12) can be used to find the permuted plain image which means that the dynamic key space of the t parameter can be removed from the dynamic key space analysis as: $KS = (2^{18})^1$, then, the cryptanalysis time: $C_{Time} = 10 \times 2^{18}\ ms \approx 186.4135$ hours $= 43.6907$ minutes.
For one encryption round, two dependent diffusion rounds (n=1, m=2), and ($N = 512$) if we assume that the Zhang cryptosystem encrypts P to C_1 in the first dependent diffusion round, and then encrypts C_1 to C_2 in the second dependent diffusion round, then using equation (4.12) we can at least find non-order C_1 pixels easily with 10 ms. To find the original plain image (at $n = 1,\ m = 2$), the cryptanalytic system needs $10\ ms$ to obtain C_1 from C_2. Then, for each possible value of p_2 and q_2 the C_1 is decrypted using the brute

force attack for all possible values of p_1 and q_1: $KS = (S_1)^2$, $S_1 = 2^{36}$, $C_{Time} = 10\times = 2^{36}\ ms$, $C_{Time} \approx 21.79$ years.

For one encryption round, one dependent diffusion round (n=1, m=1), and ($N = 256$) $KS = (S_1)^1$, $S_1 = 2^{16}$, then: $C_{Time} = 2.5 \times 2^{16}\ ms \approx 39.32$ seconds

For one encryption round, two dependent diffusion rounds (n=1, m=2), and ($N = 256$) $KS = (S_1)^2$ $S_1 = 2^{32}$ $C_{Time} = 2.5 \times 2^{32}\ ms \approx 124.27$ days

4.1.2.2 Chosen plaintext attack on the first Zhang cryptosystem

The chosen plaintext attack is defined as a cryptanalysis model when the adversary has the capability of choosing some plaintexts and encrypting them. Then, the adversary studies the corresponding ciphertexts to obtain some information of the used keys or even to reduce the security level of the cryptosystem[11],[58]. From this well-known definition, we can choose a plaintext and encrypt it without knowing the dynamic keys, and our objective is to calculate the dynamic keys. The following scenario describes the proposed partial crytpanalysis of the first Zhang cryptosystem. The chosen plain image is used to find the dynamic keys (q_1, p_1) of the first dependent diffusion round and then using these keys we can decipher any ciphered image with the same keys (this scenario is only applicable in the case where $n = 1, m = 1$). We choose a specific plain image P of $512 \times 512 \times 1$ size. This plain image is chosen to help us in the process of finding the dynamic keys (q_1, p_1). Indeed, we try to fill each position with a predefined value to decrease the range values of the dynamic keys q_1 and p_1. For example, assuming that the decrypted value is 5, we can be sure that only rows ($1,\ 3,\ 9$ or 10) contain this value. This will be helpful in the process of finding q_1 and p_1 values as illustrated below. The image P is encrypted, and the ciphered image (C_1) is obtained. Then, a group of ciphered pixels is used to analyze the encryption process and to try to recover the dynamic keys. For this we introduce the following steps:

$$
\mathrm{P} = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \\ \vdots \\ 507 \\ 508 \\ 509 \\ 510 \\ 511 \end{array}
\begin{array}{c}
\begin{array}{cccccccccc} 0 & 1 & 2 & \cdots & 254 & 255 & \cdots & 509 & 510 & 511 \end{array} \\
\left(\begin{array}{cccccccccc}
0 & 1 & 1 & \cdots & 1 & 1 & \cdots & 1 & 1 & 1 \\
1 & 2 & 3 & \cdots & 255 & 1 & \cdots & 255 & 1 & 2 \\
2 & 2 & 2 & \cdots & 2 & 2 & \cdots & 2 & 2 & 2 \\
3 & 4 & 5 & \cdots & 2 & 3 & \cdots & 2 & 3 & 4 \\
1 & 1 & 1 & \cdots & 1 & 1 & \cdots & 1 & 1 & 1 \\
3 & 3 & 3 & \cdots & 3 & 3 & \cdots & 3 & 3 & 3 \\
3 & 3 & 3 & \cdots & 3 & 3 & \cdots & 3 & 3 & 3 \\
4 & 4 & 4 & \cdots & 4 & 4 & \cdots & 4 & 4 & 4 \\
4 & 4 & 4 & \cdots & 4 & 4 & \cdots & 4 & 4 & 4 \\
5 & 5 & 5 & \cdots & 5 & 5 & \cdots & 5 & 5 & 5 \\
5 & 5 & 5 & \cdots & 5 & 5 & \cdots & 5 & 5 & 5 \\
6 & 6 & 6 & \cdots & 6 & 6 & \cdots & 6 & 6 & 6 \\
6 & 6 & 6 & \cdots & 6 & 6 & \cdots & 6 & 6 & 6 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
254 & 254 & 254 & \cdots & 254 & 254 & \cdots & 254 & 254 & 254 \\
254 & 254 & 254 & \cdots & 254 & 254 & \cdots & 254 & 254 & 254 \\
255 & 255 & 255 & \cdots & 255 & 255 & \cdots & 255 & 255 & 255 \\
255 & 255 & 255 & \cdots & 255 & 255 & \cdots & 255 & 255 & 255 \\
1 & 1 & 1 & \cdots & 1 & 1 & \cdots & 1 & 1 & 1
\end{array}\right)
\end{array}
$$

First dynamic key calculations ***(q_1)***

To find the value of q_1 (which refers to the column position) two steps are carried out:

First Step

Using the ciphered pixel at position ($x = 1$, and $y = 0$), the plain pixel position (x', y') is calculated using equation (4.4):

$x' = 1 \times x + p_1 \times y = 1 + 0 = 1 \implies \mathbf{x' = 1}$

$y' = q_1 \times x + (q_1 \times p_1 + 1) \times y y' = q_1 \times 1 + (q_1 \times p_1 + 1) \times 0 \implies \mathbf{y' = q_1}$

This means that from the value of the ciphered pixel $ciph(1,\ 0)$, the value of the plain pixel $arr(1,\ q_1)$ can be easily calculated using equation (4.12) as:

$arr(1,\ q_1) = ciph_k \oplus f(ciph_{k-1})$

where $ciph_k = ciph(1,\ 0)$ and $ciph_{k-1}$ is calculated using equation (4.13).

$$
\begin{aligned}
&if\ k \text{ is a multiple of N then} \\
&x_{previous} = x - 1 \\
&y_{previous} = N - 1 \\
&else \\
&x_{previous} = x \\
&y_{previous} = y - 1
\end{aligned}
\tag{4.13}
$$

So, $arr(1,\ q_1) = ciph(1,\ 0) \oplus f(ciph(0,\ 511))$

Now, the ciphered pixels ($ciph(1,\ 0)$ and $ciph(0,\ 511)$) are known, also the function $f(t)$ is known (see equation 4.9), and the plain pixel value $arr(1,\ q_1)$ is located in row number 1 of the plain image P. The only unknown parameter is the column position which is q_1, ranging from 0 to 511. The following equation (4.14) is used to decrease the key space of the dynamic key q_1.

$$
\begin{aligned}
&if\ arr(1,\ q_1) == 0 \text{ skip this value} \\
&else\ if\ arr(1,\ q_1) == 1\ then \\
&q_1 \in \{0,\ 255,\ 510\} \\
&else\ if\ arr(1,\ q_1) == 2\ then \\
&q_1 \in \{1,\ 256,\ 511\} \\
&else \\
&q_1 \in \{arr(1,\ q_1) - 1,\ arr(1,\ q_1) + 254\}
\end{aligned}
\tag{4.14}
$$

Firstly, assume that $arr(1,\ q_1) = 0$, then it is skipped since it can be in any position of the P image (Figure.4.1 justifies this, since the first pixel is swapped with a random pixel from the whole image). Secondly, assume that $arr(1,\ q_1) = 1$, this value can be located in one of three positions on row number 1: [$(1,\ 0)$, $(1,\ 255)$ or $(1,\ 510)$]. Thirdly, assume that $arr(1,\ q_1) = 2$, this value can be located in one of three positions on row number 1: [$(1,\ 1)$, $(1,\ 256)$ or $(1,\ 511)$]. Finally, assume that $arr(1,\ q_1) > 2$, this value can be located in one of two positions on row number 1: [$(1,\ arr(1,\ q_1) - 1)$ or $(1,\ arr(1,\ q_1) + 254)$] which are the positions containing the $arr(1,\ q_1)$ value.

Second Step

To further decrease the range values of q_1, another ciphered pixel at ($x = 3$, and $y = 0$) position is considered. The position (x', y') of the corresponding plain pixel is calculated using equation (4.4):

$x' = 1 \times x + p_1 \times y x' = 3 + 0 = 3 \implies \mathbf{x' = 3}$

$y' = q_1 \times x + (q_1 \times p_1 + 1) \times y y' = q_1 \times 3 + (q_1 \times p_1 + 1) \times 0 \implies \mathbf{y' = 3q_1}$

Using equations (4.12) and (4.13)

$arr(3,\ 3q_1) = ciph(3,\ 0) \oplus f(ciph(2,\ 511))$

The second decrypted pixel $arr(3,\ 3q_1)$ is located at row number 3, and the column position $3q_1$, with $0 \leq 3q_1 \leq 511$. Then the following equation (4.15) is used to decrease the key space of the dynamic key q_1.

$$
\begin{aligned}
&if\ arr(3,\ 3q_1) == 0 \text{ skip this value}\\
&else\ if\ arr(3,\ 3q_1) == 1\ then\\
&3q_1 \in \{253,\ 508\} \implies q_1 \in \{255,\ 340\}\\
&else\ if\ arr(3,\ 3q_1) == 2\ then\\
&3q_1 \in \{254,\ 509\} \implies q_1 \in \{426,\ 511\}\\
&else\ if\ arr(3,\ 3q_1) == 3\ then\\
&3q_1 \in \{0,\ 255,\ 510\} \implies q_1 \in \{0,\ 85,\ 170\}\\
&else\ if\ arr(3,\ 3q_1) == 4\ then\\
&3q_1 \in \{1,\ 256,\ 511\} \implies q_1 \in \{171,\ 256,\ 341\}\\
&else\\
&q_1 \in \{a,\ b\}
\end{aligned}
\tag{4.15}
$$

$if((arr(3,\ 3q_1) - 3)\ MOD\ 3 == 0)Then$
$a = \frac{arr(3,\ 3q_1)-3}{3}$
$else\ if((arr(3,\ 3q_1) - 3 + 512)\ MOD\ 3 == 0)Then$
$a = \frac{arr(3,\ 3q_1)+509}{3}$
$else$
$a = \frac{arr(3,\ 3q_1)-3+512+512}{3} = \frac{arr(3,\ 3q_1)+1021}{3}$
$if((arr(3,\ 3q_1) + 252)\ MOD\ 3 == 0)Then$
$b = \frac{arr(3,\ 3q_1)+252}{3}$
$else\ if((arr(3,\ 3q_1) + 252 + 512)\ MOD\ 3 == 0)Then$
$b = \frac{arr(3,\ 3q_1)+764}{3}$
$else$
$b = \frac{arr(3,\ 3q_1)+252+512+512}{3} = \frac{arr(3,\ 3q_1)+1276}{3}$

Equation (4.15) is used to decrease the number of possible column positions (i.e., the range of the q_1). Using the P matrix, the row of the $arr(3,\ 3q_1)$ is 3. Firstly, assume that $arr(3,\ 3q_1) = 0$, then, this value is skipped. Secondly, assume that $arr(3,\ 3q_1) = 1$, then, this value can be located in one of two column positions in row number 3 [$(3,\ 253)$ or $(3,\ 508)$] (see P matrix for more details). Thirdly, assume that $arr(3,\ 3q_1) = 2$, then, this value can be located in one of two column positions in row number 3 [$(3,\ 254)$ or $(3,\ 509)$]. Fourthly, assume that $arr(3,\ 3q_1) = 3$, then, this value can be located in one of three column positions in row number 3 [$(3,\ 0)$, $(3,\ 255)$ or $(3,\ 510)$]. Fifthly, assume that $arr(3,\ 3q_1) = 4$, this value can be located in one of three column positions in row number 3 [$(3,\ 1)$, $(3,\ 256)$ or $(3,\ 511)$]. Finally, for all $arr(1,\ q_1) > 4$, these values can be located in one of two column positions in row number 3 [$(3,\ arr(3,\ 3q_1) - 3)$ or $(1,\ arr(1,\ q_1)+252)$]. Now, let us take the case when the $arr(3,\ 3q_1) = 4$ as an example, without loss of generality, i.e., $3q_1 \in \{1,\ 256,\ 511\}$
Firstly, $3q_1 = 1$; to solve this equation, which contains the module operation, $q1$ is calculated as:
$q1$ =the integer value of $(\frac{1}{3},\ \frac{1+512}{3},\ \frac{1+512+512}{3}) \implies q1 = 171$.
Secondly, $3q_1 = 256$, then $q1$ =the integer value of $(\frac{256}{3},\ \frac{256+512}{3},\ \frac{256+512+512}{3}) \implies q1 = 256$.

Finally, $3q_1 = 511$, then $q1$ =the integer value of $(\frac{511}{3}, \frac{511+512}{3}, \frac{511+512+512}{3}) \implies q1 = 341$

For all other values ($arr(3,\ 3q_1) > 4$), there is a general scheme to calculate the range of q_1 values (i.e., a and b parameters in equation (4.15)). Now, let us take $arr(3,\ 3q_1) = 5$ as another example. i.e., $3q_1 \in \{2,\ 257\}$
Firstly, $3q_1 = 2$, then: $q1$ =the integer value of $(\frac{2}{3}, \frac{2+512}{3}, \frac{2+512+512}{3}) \implies q1 = 342$.
Secondly, $3q_1 = 257$, then $q1$ =the integer value of $(\frac{257}{3}, \frac{257+512}{3}, \frac{257+512+512}{3}) \implies q1 = 427$. From previous developments, we can deduce the following rules for equation (4.15):

1. In the case where $arr(3,\ 3q_1) \in \{1,\ 2\}$, then we have two positions given by: $arr(3,\ 3q_1) + 252$ or $arr(3,\ 3q_1) + 507$.

2. In the case where $arr(3,\ 3q_1) \in \{3,\ 4\}$, then we have three positions given by: $arr(3,\ 3q_1) - 3$, $arr(3,\ 3q_1) + 252$ or $arr(3,\ 3q_1) + 507$.

3. In the case where $arr(3,\ 3q_1) > 4$, then we have two positions given by: $arr(3,\ 3q_1) - 3$ or $arr(3,\ 3q_1) + 252$.

Now, from the four previous ciphered pixels ($ciph(1,\ 0)$, $ciph(0,\ 511)$, $ciph(3,\ 0)$ and $ciph(2,\ 511)$), the exact value of the dynamic key q_1 is calculated by finding the overlap values between the ***first*** and the ***second step***.
Second dynamic key calculations (p_1)
To find the value of p_1 (which refers to the row position), at least two steps are needed:
First Step
From the ciphered pixel at ($x = 0$, and $y = 1$) position, the plain pixel position ($x',\ y'$) is calculated using equation (4.4):
$x' = 0 \times x + p_1 \times 1 x' = 0 + p_1 = p_1 \implies \mathbf{x' = p_1}$
$y' = q_1 \times x + (q_1 \times p_1 + 1) \times y y' = q_1 \times 0 + (q_1 \times p_1 + 1) \times 1 \implies \mathbf{y' = q_1 \times p_1 + 1}$
Using equation (4.12), the value of $arr(p_1,\ p_1 q_1 + 1)$ is calculated as:
$arr(p_1,\ p_1 q_1 + 1) = ciph(0,\ 1) \oplus f(ciph(0,\ 0))$
The decrypted pixel $arr(p_1,\ p_1 q_1 + 1)$ is located at row number p_1 of the plain image P. Here we focus on finding the p_1 value (p_1 ranges from 0 to 511). Equation (4.16) is used to decrease the dynamic key space p_1 as:

$$
\begin{aligned}
&if\ a == 0\ skip\ this\ value \\
&else\ if\ a == 1\ then \\
&p_1 \in \{0,\ 1,\ 3,\ 4,\ 511\} \\
&else\ if\ a == 2\ then \\
&p_1 \in \{1,\ 2,\ 3\} \\
&else \\
&p_1 \in \{1,\ 3,\ a \times 2 - 1,\ a \times 2\} \\
&where \\
&a = arr(p_1,\ p_1 q_1 + 1)
\end{aligned}
\tag{4.16}
$$

To justify the equation (4.16), firstly, assume $arr(p_1,\ p_1 q_1 + 1) = 1$, from the plain image P, the plain pixel value 1 is located in five rows [0, 1, 3, 4, 511], and so, the possible values of the dynamic key p_1 is: [0, 1, 3, 4, 511]. Secondly, assume the $arr(p_1,\ p_1 q_1 + 1) = 2$, then this value is located in three rows: [1, 2, 3], and so, the possible values of the dynamic key p_1 is: [1, 2, 3]. Thirdly, assume the $arr(p_1,\ p_1 q_1 + 1) = 3$, then this

value is located in four rows [1, 3, 5, 6], and so, the possible values of the dynamic key p_1 is: [1, 3, 5, 6]. Finally, for any value such that $arr(p_1,\ p_1q_1+1) > 2$, this value can be located in one of four rows [1, 3, $arr(p_1,\ p_1q_1+1) \times 2 - 1$, $arr(p_1,\ p_1q_1+1) \times 2$], and so, the possible values of the dynamic key p_1 is [1, 3, $arr(p_1,\ p_1q_1+1) \times 2 - 1$, $arr(p_1,\ p_1q_1+1) \times 2$].

Second Step

To further decrease the range values of p_1, another ciphered pixel at ($x = 0$, and $y = 2$) position is considered. The position (x', y') of the corresponding plain pixel is calculated using equation (4.4): $x' = 1 \times x + p_1 \times y \implies x' = 0 + 2p_1 = 2p_1 \implies \mathbf{x' = 2p_1}$
$y' = q_1 \times x + (q_1 \times p_1 + 1) \times y \implies y' = q_1 \times 0 + (q_1 \times p_1 + 1) \times 2 \implies \mathbf{y' = 2(q_1 \times p_1 + 1)}$
Again, using equation (4.12) the value of $arr(2p_1,\ 2p_1q_1+2)$ is calculated as:
$arr(2p_1,\ 2p_1q_1+2) = ciph(0,\ 2) \oplus f(ciph(0,\ 1))$
This pixel $arr(2p_1,\ 2p_1q_1+2)$ is located at row $2p_1$ of the plain image P. The following equation (4.17) is used to decrease the key space of the dynamic key p_1 as:

$$
\begin{aligned}
&if\ a == 0\ skip\ this\ value \\
&else\ if\ a == 1\ then \\
&p_1 \in \{0,\ 2,\ 256,\ 258\} \\
&else\ if\ a == 2\ then \\
&p_1 \in \{1,\ 257\} \\
&else \\
&p_1 \in \{a,\ a + 256\} \\
&where \\
&a = arr(2p_1,\ 2(p_1q_1+1))
\end{aligned}
\tag{4.17}
$$

To justify equation (4.17), we consider the following cases: Firstly, assume that the decrypted pixel is $arr(2p_1,\ 2p_1q_1+2) = 1$, so from the plain image P, the plain pixel value 1 is located in five rows [0, 1, 3, 4, 511], and the possible values of the dynamic key p_1 are:
$2p_1 = 0 \implies p_1 \in \{0,\ 256\}$
$2p_1 = 1$, $2p_1 = 3$ and $2p_1 = 511$ cannot have an integer solution, since the right-hand side is odd and the left-hand side is even and the modulus value is even. $2p_1 = 4 \implies p_1 \in \{2,\ 258\}$. Secondly, assume the decrypted pixel is $arr(2p_1,\ 2p_1q_1+2) = 2$, then the plain pixel value 2 is located in three rows [1, 2, 3], and the possible values of p_1 are:
$2p_1 = 1$ and $2p_1 = 3$ cannot have an integer solution.
$2p_1 = 2 \implies p_1 \in \{1,\ 257\}$.
Thirdly, assume the decrypted pixel is $arr(2p_1,\ 2p_1q_1+2) = 3$, then the plain pixel value 3 is located in four rows [1, 3, 5, 6], and the possible values of p_1 are: $2p_1 = 1$, $2p_1 = 3$ and $2p_1 = 5$ cannot have integer solution. $2p_1 = 6 \implies p_1 \in \{3,\ 259\}$. Finally, for any $arr(2p_1,\ 2p_1q_1+2) > 2$ the possible values of p_1 are $arr(2p_1,\ 2p_1q_1+2)$ or $arr(2p_1,\ 2p_1q_1+2) + 256$.
Note that in most cases the first and the second steps are sufficient to find the dynamic key value of p_1. However, if the overlap between the obtained range values of p_1 in the first step and the obtained range values of p_1 in the second step is not a single value, then we use the following calculation.

Extra Step

To find the exact value of p_1, the plain pixel position (x', y') is calculated from the ciphered position pixel at ($x = 0$, $y = 3$), using equation (4.4):
$x' = 1 \times x + p_1 \times y = 0 + 3p_1 = 3p_1 \implies \mathbf{x' = 3p_1}$

$y' = q_1 \times x + (q_1 \times p_1 + 1) \times y = q_1 \times 0 + (q_1 \times p_1 + 1) \times 3 \implies \mathbf{y' = 3(q_1 \times p_1 + 1)}$
Again, here the plain pixel $arr(3p_1,\ 3(q_1 \times p_1 + 1))$ is calculated from the ciphered pixels $ciph(0,\ 3)$ and $ciph(0,\ 2)$ using equation (4.12). Now equation (4.18) is used to decrease the dynamic key space of p_1. To simplify the notation of the calculations, assume:
$T = arr(3p_1,\ 3(p_1 \times q_1 + 1))$

$$
\begin{aligned}
&if\ T == 0 \text{ skip this value}\\
&else\ if\ T == 1\ then\\
&3p_1 \in \{0,\ 1,\ 3,\ 4,\ 511\} \implies p_1 \in \{0,\ 1,\ 171,\ 172,\ 341\}\\
&else\ if\ T == 2\ then\\
&3p_1 \in \{1,\ 2,\ 3\} \implies p_1 \in \{1,\ 171,\ 342\}\\
&else\\
&3p_1 \in \{1,\ 3,\ T-1,\ T\} \implies p_1 \in \{1,\ 171,\ a,\ b\}
\end{aligned}
\tag{4.18}
$$

where
$if((T \times 2 - 1)\ MOD\ 3 == 0)Then$
$a = \frac{T \times 2 - 1}{3}$
$else\ if((T \times 2 + 511)\ MOD\ 3 == 0)Then$
$a = \frac{T \times 2 + 511}{3}$
$else$
$a = \frac{T \times 2 + 1023}{3}$
$if((T \times 2)\ MOD\ 3 == 0)Then$
$b = \frac{T \times 2}{3}$
$else\ if((T \times 2 + 512)\ MOD\ 3 == 0)Then$
$b = \frac{T \times 2 + 512}{3}$
$else$
$b = \frac{T \times 2 + 1024}{3}$
To justify equation (4.18), assume that the ciphered pixel value is one (i.e., $T = 1$ in equation (4.18)). From the plain image P, the value 1 is located in five rows, actually in positions [0, 1, 3, 4, 511], and so: $3p_1 \in \{0,\ 1,\ 3,\ 4\}$.
Firstly, $3p_1 = 0$; to solve this equation, which contains the module operation, p_1 is calculated as: $p1$ =the integer value of $(\frac{0}{3}, \frac{0+512}{3}, \frac{0+512+512}{3}) \implies p1 = 0$.
Secondly, $3p_1 = 1$, then $p1$ =the integer value of $(\frac{1}{3}, \frac{1+512}{3}, \frac{1+512+512}{3}) \implies p1 = 171$.
Thirdly, $3p_1 = 3$, then $p1$ =the integer value of $(\frac{3}{3}, \frac{3+512}{3}, \frac{3+512+512}{3}) \implies p1 = 1$.
Fourthly, $3p_1 = 4$, then $p1$ = the integer value of $(\frac{4}{3}, \frac{4+512}{3}, \frac{4+512+512}{3}) \implies p1 = 172$.
Finally, $3p_1 = 511$, then $p1$ = the integer value of $(\frac{511}{3}, \frac{511+512}{3}, \frac{511+512+512}{3}) \implies p1 = 341$.
$p_1 \in \{0,\ 1,\ 171,\ 172,\ 341\}$ The same analysis is used for the other values of T, where for $T = 2$, there are only three possible rows, and for $T > 2$, there are four possible rows.
Example 4.1:calculation of p_1 ***and*** q_1
An original P image is encrypted using 3 random values of the secret key of the Logistic map, namely: $[x_{-1},\ SQ_1\ ]$, $[x_{-1},\ SQ_2]$, $[key_d,\ t_{-1}]$. The following values are obtained from the Lena encrypted image for: $[n = 1,\ m = 1]$
$\mathbf{ciph(1,\ 0) = 190}$
$\mathbf{ciph(0,\ 511) = 27}$
$\mathbf{ciph(3,\ 0) = 149}$

$\mathbf{ciph(2,\ 511) = 159}$
$\mathbf{ciph(0,\ 1) = 132}$
$\mathbf{ciph(0,\ 0) = 165}$
$\mathbf{ciph(0,\ 2) = 140}$
$\mathbf{ciph(0,\ 1) = 132}$
For the first decrypted pixel, using equation (4.12) we obtain:
$arr(1,\ q_1) = ciph(1,\ 0) \oplus f(ciph(0,\ 511))$
$arr(1,\ q_1) = 190 \oplus f(27)$
$arr(1,\ q_1) = 190 \oplus 105$
$arr(1,\ q_1) = 215$
Using equation (4.14), q_1 values are restricted to:
$q_1 \in \{215 - 1,\ 215 + 254\} \implies q_1 \in \{214,\ 469\}$
For the second decrypted pixel, using equation (4.12) we obtain:
$arr(3,\ 3q_1) = ciph(3,\ 0) \oplus f(ciph(2,\ 511))$
$arr(3,\ 3q_1) = 149 \oplus f(159)$
$arr(3,\ 3q_1) = 149 \oplus 22$
$arr(3,\ 3q_1) = 131$
Using equation (4.15), $q_1 \in \{a,\ b\}$.
To find the value of the parameter a of equation (4.15), the following conditions are performed:
$((131 - 3)\ MOD\ 3) = 2$
$((131 + 509)\ MOD\ 3) = 1$
$((131 + 1021)\ MOD\ 3) = 0$
The last condition allows us to calculate the value of parameter a:
$a = \frac{arr(3,\ 3q_1)+1021}{3} \implies a = 384$
To find the value of the parameter b in equation (4.15), the following conditions are performed:
$((131 + 252)\ MOD\ 3) = 2$
$((131 + 764)\ MOD\ 3) = 1$
$((131 + 1276)\ MOD\ 3) = 0$
The last condition allows us to calculate the value of parameter b:
$b = 469$, so, $q_1 \in \{384,\ 469\}$
The overlap between the first range $\{214,\ 469\}$ and the second $\{384,\ 469\}$range gives the exact value of q_1: $\mathbf{q_1 = 469}$

To calculate the exact value of the dynamic key p_1, we consider the third decrypted pixel and equation (4.12), then:
$arr(p_1,\ p_1q_1 + 1) = ciph(0,\ 1) \oplus f(ciph(0,\ 0))$
$arr(p_1,\ p_1q_1 + 1) = 132 \oplus f(165)$
$arr(p_1,\ p_1q_1 + 1) = 132 \oplus 39$
$arr(p_1,\ p_1q_1 + 1) = 163$
Now by using equation (4.16), p_1 values are restricted to:
$p_1 \in \{1,\ 3,\ 163 \times 2 - 1,\ 163 \times 2\} \implies p_1 \in \{1,\ 3,\ 325,\ 326\}$
Finally, the exact value of p_1 is obtained by using the fourth decrypted pixel and equation (4.12), as follows:
$arr(2p_1,\ 2p_1q_1 + 2) = ciph(0,\ 2) \oplus f(ciph(0,\ 1))$
$arr(2p_1,\ 2p_1q_1 + 2) = 140 \oplus f(132)$
$arr(2p_1,\ 2p_1q_1 + 2) = 140 \oplus 202$

$arr(2p_1,\ 2p_1q_1 + 2) = 70$
Using equation (4.17):
$p_1 \in \{70,\ 326\}$
The overlap between the first range ($\{1,\ 3,\ 325,\ 326\}$) and the second range ($\{70,\ 326\}$) gives the exact value of p_1: $\mathbf{p_1 = 326}$
Remark: the above procedure can be used to find any used p_1 and q_1 parameters.

4.1.2.3 Combination of brute force and chosen plaintext attacks

In the case where $n = 1,\ m = 2$ which is assumed be a secured case as it is written in[166] at page 2074 "In Algorithm 1, the round numbers m and n as shown in Figure.7 are selected as 2 and 1, respectively", we will show that the algorithm can be partially cryptanalyzed. A combination of the aforementioned attacks are used to find the dynamic key pairs (p_1, q_1, p_2, q_2) and then to find any plain image, ciphered by these pairs of keys. First, the plain image P, is encrypted in the first dependent diffusion round ($n = 1,\ m = 1$) with the dynamic key parameters (p_1, q_1) to obtain the middle cipher image (C_1). Then, in the second dependent diffusion round ($n = 1,\ m = 2$), the image (C_1) with parameters (p_2, q_2) is ciphered to obtain (C_2). The cryptanalysis scenario is carried out as follows:

1. Using equation 4.12, and (C_2), we can find the permuted version of (C_1) in 10 ms.

2. We guess the pairs (p_2, q_2), and for each guess, the image pixels C_1 are reordered. Then, the proposed chosen plaintext attack of section (4.1.2.2) is used to find the pair (p_1, q_1) at ($n = 1,\ m = 1$). In the worst case, this process requires less than one hour to find (p_1, q_1) and (p_2, q_2) (for the image of size 512×512) since:
   The number of all possible dynamic keys (p_2, q_2) is 2^{18} keys. The total time to find (p_1, q_1) and (p_2, q_2) is referred to $T_{n=1,\ m=2}$: $T_{n=1,\ m=2} = T_1+T_2+T_3$
   T_1 is the required time to calculate (C_1) from (C_2) which is 10 ms.
   T_2 is the required time to guess the second dynamic key (p_2, q_2) and to order C_1 pixels. It is equal to 5 ms for each try. In the worst case it requires $2^{18} \times 5$ ms.
   T_3 is the required time to achieve the proposed attack of section 4.1.2.2, which is less than 1 ms for each guess. $T_{n=1,\ m=2} = 10+2^{18} \times 5+2^{18} \times 1 = 30.583$ minutes.

As a result, the zhang cryptosystem is secure against the previous proposed attacks, if the number of encryption rounds is more than or equal to ($n = 2,\ m = 2$, i.e., the dependent diffusion round is at least equal to 4). However, in this case, the Zhang algorithm is time consuming, because each dependent diffusion round needs 10 ms. The total time required is 40 ms, which means that the Running Speed (RS) of 512×512 gray image is:$RS = \frac{512 \times 512}{0.040} = 6.25$ Mega bytes per second which is not suitable for real-time application.

4.1.3 Decreasing the UACI and NPCR values significantly

A cryptosystem should be sensitive to one-bit changes in the plaintext. This requirement is most important to resist the known plaintext and the chosen plaintext attacks[74][81]. In a chosen plaintext attack, more than one plaintext (with one-bit changes between them) are selected to analyze the difference between their corresponding ciphertexts. The measurement tool to test the sensitivity of any cryptosystem of these attacks is presented and described in section (2.2.2) of the chapter 2. We try to reproduce the NPCR and UACI results of the Zhang cryptosystem using exactly the same tested image (i.e., Barbara $512 \times 512 \times 1$ gray-scale image), the same secret keys (i.e., x_{-1}=0.12345678912345 for SQ_1,

Table 4.1: Sample of NPCR and UACI results under the same parameters and conditions in the original research paper for all pixel positions)

Row Position	Column Position	UACI	NPCR	UACI_p	NPCR_p
0	0	33.529	99.624	31.747	93.872
14	103	33.299	98.883	11.950	35.471
26	476	33.212	98.569	07.313	21.654
39	98	32.987	97.865	05.073	14.990
97	475	32.699	96.828	03.425	10.135
125	87	33.473	99.611	33.171	98.668
296	337	32.797	97.139	04.482	13.224
471	375	33.151	98.256	07.441	22.040
511	511	33.478	99.611	33.034	98.057

x_{-1} =0.67856746347633 for SQ_2. α is set to 3.99999, and key_d=0.33456434300001), and with ($n = 2,\ m = 2$). We performed the following steps:

1. Encrypting the Barbara image using the first Zhang cryptosystem and using the same secret keys mentioned above to obtain C_1. Then, encrypting the Barbara image with a one-bit change to obtain C_2. Then UACI and NPCR are calculated between C_1 and C_2.

2. The ciphered images C_1 and C_2 are used as input for a process based on the equation (4.12) to remove the diffusion effect in less than 1 ms, then the UACI_p and NPCR_p are calculated.

The results obtained in Table 4.1.1, measured by the UACI_p and the NPCR_p, show that for some pixel positions [$(511,\ 511)$, $(0,\ 0)$ and $(125,\ 87)$], with a one-bit change value (the LSB bit), the Zhang cryptosystem is not so secure. We have done the same experiment already explained above, for two pixel positions, but in relation to the nature of the image and also in relation to the secret key of the Logistic map. The results given in Table 4.2, of the parameters UACI_p, NPCR_p, show the sensitivity of the Zhang cryptosystem regarding the image under test and the used secret key of the Logistic map.

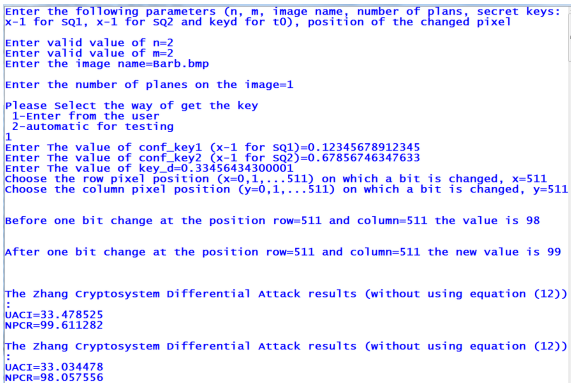
In Table. 4.2, the following keys are used
Key 1:x_{-1}=0.4251533555101169 for SQ_1, and x_{-1}=0.80288094729453419 for SQ_2, the initial value of the key_d=0.2251045258949553.
Key 2:x_{-1}=0.96368297372356337 for SQ_1, and x_{-1}=0.80925931577501753 for SQ_2, the initial value of the key_d=0.50190740684224977.
We show the Graphical User Interface (GUI) of our demonstration to test the weaknesses of the first Zhang cryptosystem, using our cryptanalysis based on the equation (4.12). In these Figures (4.2, 4.3 and 4.4), the results of parameters NPCR and UACI before and after differential attack, are given in relation to the nature of the image and also in relation to the secret key of the logistic map. Figure. 4.5, 4.6 and 4.7 concern the results of pixel positions $(511,\ 511)$, $(0,\ 0)$ and $(125,\ 87)$ respectively, In that figures:

Table 4.2: Sample of NPCR and UACI results

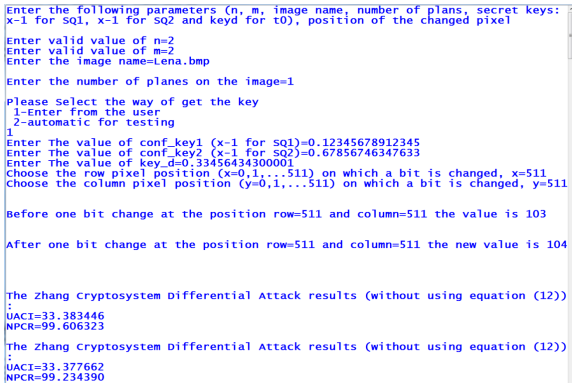
Row,Column Positions	UACI	NPCR	$UACI_p$	$NPCR_p$	Key	used image
511, 511	33.000	97.859	15.223	45.221	1	Barb
	27.184	80.579	13.366	39.665		Lena
	33.286	99.026	23.934	71.059		Boat
	27.486	81.781	14.083	41.806		Baboon
0, 0	30.730	91.057	14.557	43.183	1	Barb
	33.258	98.777	20.295	60.161		Lena
	28.804	85.552	15.665	46.507		Boat
	27.244	81.135	11.233	33.308		Baboon
125, 87	32.670	97.047	18.440	54.797	1	Barb
	33.575	99.580	30.954	91.762		Lena
	30.940	92.003	14.789	43.671		Boat
	32.443	96.348	18.876	56.158		Baboon
511, 511	5.979	17.734	3.453	10.269	2	Barb
	6.838	20.343	2.993	8.883		Lena
	6.217	18.449	3.820	11.314		Boat
	7.504	22.371	4.578	13.579		Baboon
0, 0	7.691	22.753	5.039	14.939	2	Barb
	7.065	20.919	4.524	13.484		Lena
	8.048	23.861	3.575	10.585		Boat
	6.434	19.092	3.985	11.753		Baboon
125, 87	05.737	17.081	02.746	08.123	2	Barb
	08.885	26.581	03.950	11.695		Lena
	06.510	19.318	04.538	13.546		Boat
	07.970	23.809	04.533	13.369		Baboon

- Number of encryption rounds n, and number of dependent diffusion rounds m.
- Image name.
- Number of planes in the image, where 1 refers to the gray-scale and 3 to color image.
- Way of obtaining the secret key, 1, means that the user wants to use the same secret keys of the Zhang paper, and 2, means that the user wants to insert keys manually.
- Row position, column position of the pixel that will be changed by one bit (LSB bit).

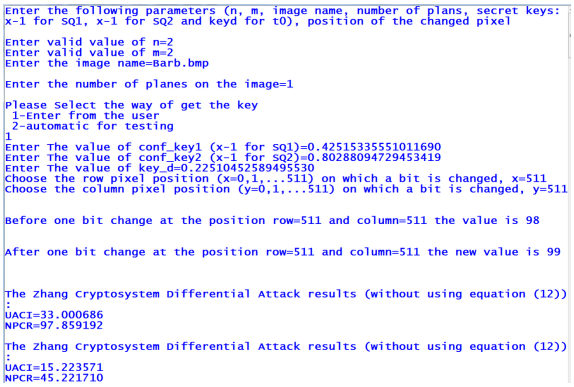
Then, the Graphical User Interface (GUI) of the demo shows the calculated results of parameters UACI and NPCR before and after using our cryptanalysis equation (Equation (4.12)).



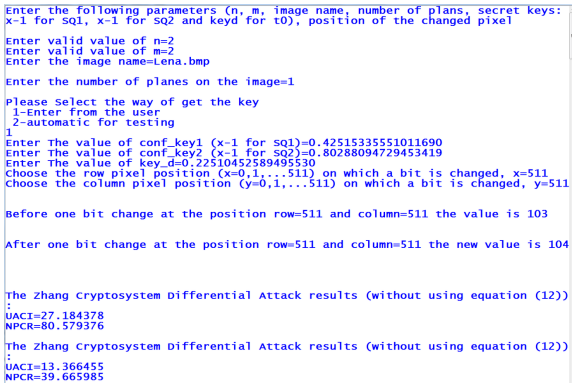
(a) Barb using the same Zhang key



(b) Lena using the same Zhang key

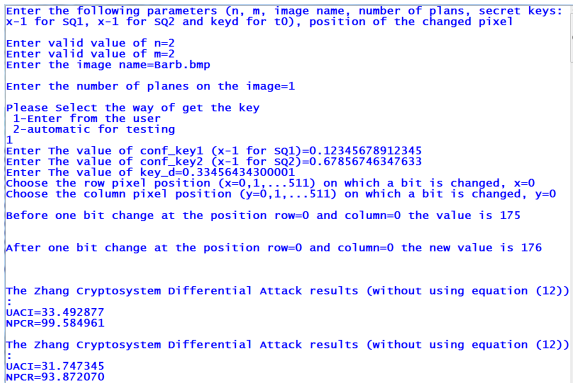


(c) Barb using key1

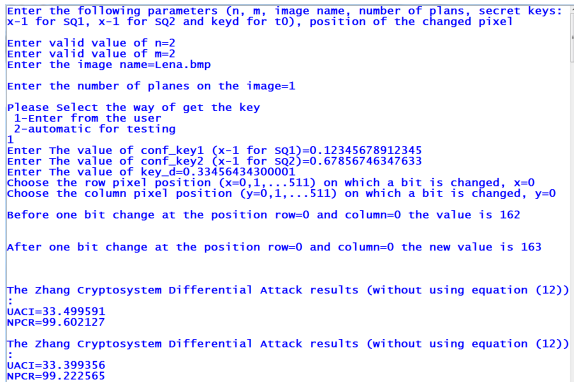


(d) Lena using key1

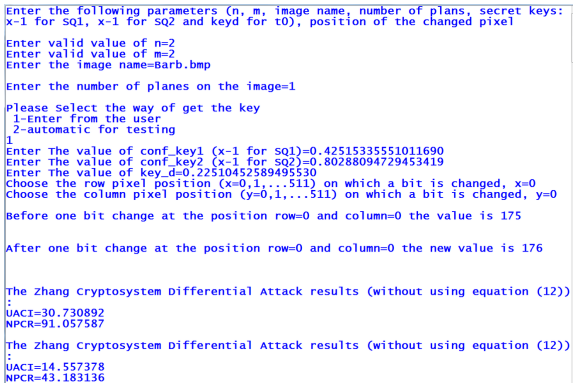
Figure 4.2: Results concerning pixel position $(511,\ 511)$



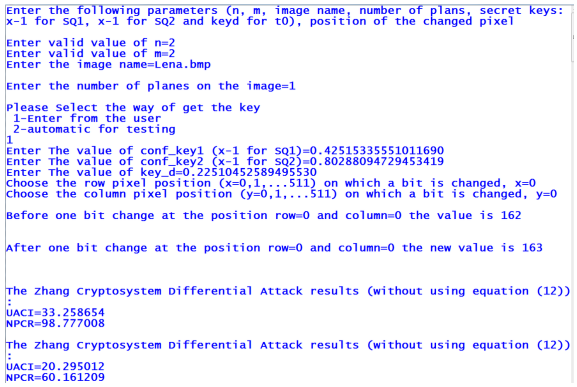
(a) Barb using the same Zhang key



(b) Lena using the same Zhang key

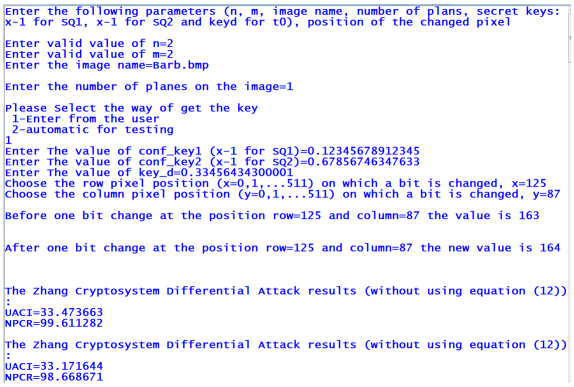


(c) Barb using key1

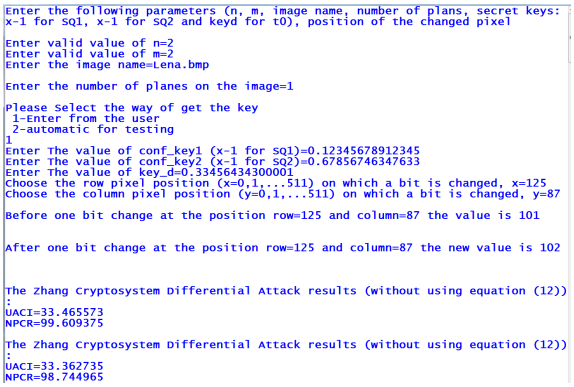


(d) Lena using key1

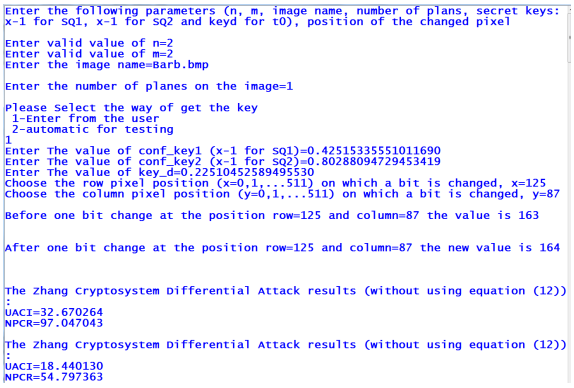
Figure 4.3: Results concerning pixel position $(0,\ 0)$



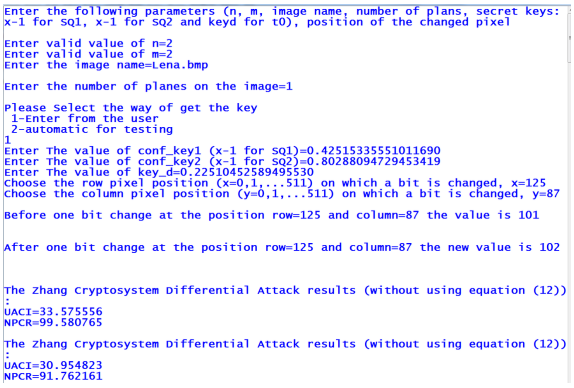
(a) Barb using the same Zhang key



(b) Lena using the same Zhang key



(c) Barb using key1



(d) Lena using key1

Figure 4.4: Results concerning pixel position $(125,\ 87)$

4.2 Designe and realization of very fast and secure cryptosystems

4.2.1 General concepts

The first step of designing a chaos-based encryption algorithm is to define the chaotic maps which will be used in such a structure. These maps are used to achieve the confusion and the diffusion effects, which are the most important properties of any cryptosystem. The general block diagram of all proposed cryptosystems is shown in Figure 4.5. The main objective of this structure is to achieve the dependent confusion-diffusion effects byte by byte. All proposed cryptosystems work in the CBC mode and use the El Assad

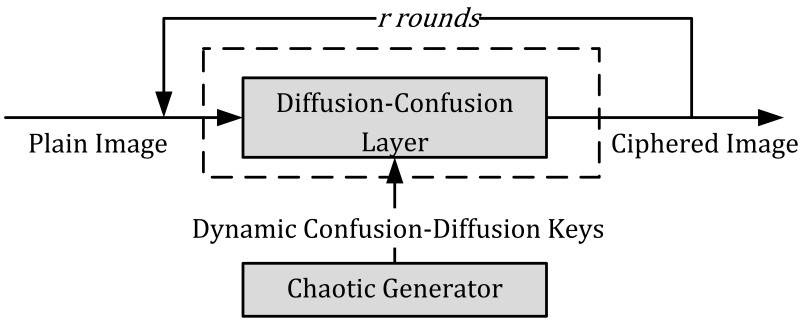


Figure 4.5: General block diagram of the proposed cryptosystems

and Noura chaotic generator that produces 32-bit samples [37]. Figure 4.6 shows the general diagram of the encryption part of the CBC mode. In Figure 4.6, P_0 is the first

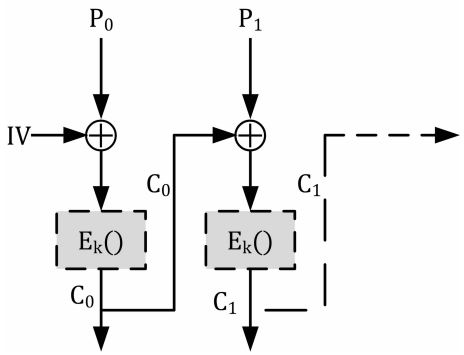


Figure 4.6: Encryption structure of the CBC mode

block from the plain image, IV is the initial vector generated by the chaotic generator, C_0 is the first encrypted block which will be transferred to the receiver side. The dash boxes represent the encryption/decryption parts of the proposed cryptosystem (see Figures 4.5, 4.7, and 4.9).

4.2.2 General differences of the proposed cryptosystem with the Zhang one

In this section, we introduce some general differences between our proposed cryptosystem, in all three versions and the Zhang's one:

- In the Zhang cryptosystem, a simple implementation of the logistic map was used to generate and manage the dynamic keys. In all versions of our proposed cryptosystem, a strong and robustness tested chaotic generator [37] is used to manage and generate the dynamic keys. These dynamic keys are used for the dependent confusion-diffusion layer.

- To overcome the fixed-point problem of the standard 2-D cat map, the Zhang cryptosystem selects a random pixel $arr({r_x}^j,\ {r_y}^j)$ and swaps it with the first pixel in the plain image $arr(0,\ 0)$. In our proposed cryptosystem this problem is solved by using the modified 2-D cat map. This modification of the standard 2-D cat map also enhances the security of the proposed cryptosystem. Indeed the number of dynamic keys in the 2-D cat map increases from two dynamic keys to four. The first pixel $(0,\ 0)$ can be mapped to any new position $(i_n,\ j_n)$.

- The Zhang cryptosystem works on the whole image. It is well known that the image encryption algorithms that work on the whole image give bad results for error propagation. The influence of one error bit of the ciphered data (due to the channel) on the decryption algorithm depends on the cryptographic modes. All versions of our proposed cryptosystem perform the encryption/decryption operations based on the Cipher-Block Chaining (CBC) mode [34].

- One of the most important differences between our proposed cryptosystems and the Zhang cryptosystem is in the structure of the dynamic keys. In the Zhang cryptosystem, the dynamic keys consist of two keys (q_i, p_i) for the reverse 2-D cat map. These key values are changed just twice for the whole encryption process. For the logistic map, t is the initial value. The value of t is changed in each byte. When comparing with our proposed cryptosystem, in the 2-D cat map, four keys are used $(v,\ u,\ ri$, and $rj)$. These keys' values are changed for every new encryption round and also for every new block.

- All chaos-based cryptosystems that use more than one encryption round (i.e., $r > 1$) to reach the required security level, must save their dynamic keys for all rounds in order to use them later in the decryption process. From a security point of view, it is normal that the security level of any cryptosystem is strongly related to the environment where these dynamic keys have been temporarily saved. To the best of our knowledge, the only possible method to manage the dynamic keys in the decryption process in case of more than one decryption round is to save them temporarily, since the decryption process is achieved by starting from the last decryption round and finishing with the first decryption round. It is important to note that the chaotic generator has to be a non-invertible generator. To obtain the current key, the chaotic generator should generate all previous keys and use them in the reverse order.

All previously mentioned points are taken into consideration in the design process of a fast and secure cryptosystem, and it should use only one encryption round($r = 1$), to obtain the required security level.

4.2.3 First proposed cryptosystem

The first proposed cryptosystem has some similarity to the Zhang cryptosystem [166]. In this cryptosystem, the two dependent confusion-diffusion layers are the modified 2-D

cat map (used to achieve the confusion effect) followed by the discrete logistic map (to achieve the diffusion effect). Using this structure, the required confusion-diffusion effects are obtained by one encryption round, and hence the execution time is decreased.

4.2.3.1 Encryption scheme of the first proposed cryptosystem

In the encryption scheme (see Figure 4.7), for the first block, each pixel from the plain block ($p_0(k)$) is XOR-ed with the initial byte ($iv(k)$) from the initial vector (IV), then the output is XOR-ed with the discrete logistic map output to carry out the diffusion process. Then, the 8 least significant bits resulting from the diffusion process ($LSB_8(y_0(k))$ are relocated using the modified 2-D cat map to obtain the ciphered pixel at the new position ($c_0(k_n)$)[39, 38]. It is important to note that the input of the discrete logistic map is based on the previous ciphered pixel (since $c_0(k_n) = LSB_8(y_0(k))$ and the input of the discrete logistic map is 32 bits and the ciphered pixel is 8 bits. That is why the cryptosystem takes ($y_0(k-1)$) before the LSB_8 function and not after. For the first encrypted byte, the input of the discrete logistic map is Kd_m, and this value is re-initialized every new encryption round. Because the $c_0(k_n)$ is only a part of the logistic map input, it is impossible to recover $y_0(k-1)$ from $c_0(k_n)$ only. The encryption of the next blocks is almost the same. Each pixel from the plain block ($p_l(k)$) is XOR-ed with ciphered byte from the previous block at the same position(i.e., $c_{l-1}(k)$ to achieve the CBC mode). Then the rest of the operations are the same as in the first encryption block. The 2-D cat map was tested and

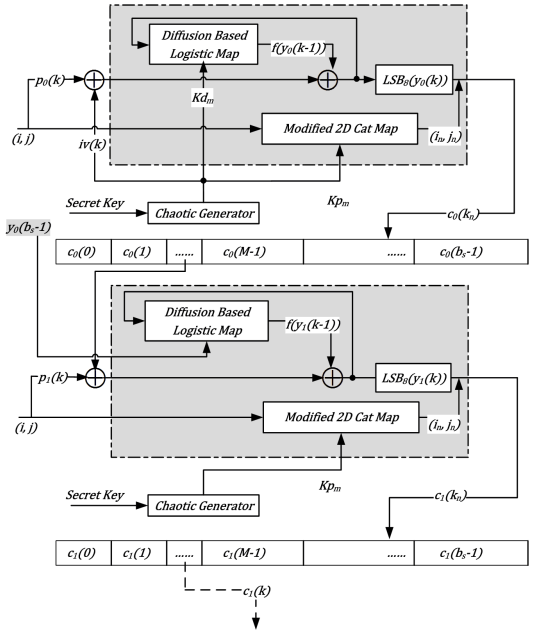


Figure 4.7: Encryption structure of the first proposed cryptosystem

analyzed by [45] and [155]. To overcome the fixed-point problem of the Arnold cat map model, the parameters ri and rj are added to the standard model. Also, in our scheme the elements of the square matrix and the parameters ri and rj of equation (4.19) become dynamic, they form the dynamic keys of the permutation process.

$$\begin{bmatrix} i_n \\ j_n \end{bmatrix} = Mod\left(\begin{bmatrix} 1 & u \\ v & 1+uv \end{bmatrix}\begin{bmatrix} i \\ j \end{bmatrix} + \begin{bmatrix} ri+rj \\ rj \end{bmatrix}, \begin{bmatrix} M \\ M \end{bmatrix}\right) \tag{4.19}$$

Equation (4.19) is a one-to-one function, which means that each point of the square matrix can be transferred to exactly one unique point. So, instead of exchanging the values at the new position $(i_n,\ j_n)$ with the old one $(i,\ j)$, we use a transfer operation because of its speed compared to the swap operation that is usually used. The block size b_s is M^2 (M is the square root of the block size in our proposed cryptosystem). The system parameters u, v, ri and rj are in the range of $[0,\ M-1]$. The structure of the dynamic keys which are produced by the chaotic generator during the permutation process is:

$$\begin{aligned} Kp &= \left[Kp_0 \parallel Kp_1 \parallel Kp_2 \parallel \cdots \parallel Kp_{r-1}\right] \\ Kp_m &= \left[u_m \parallel v_m \parallel ri_m \parallel rj_m\right] \end{aligned} \tag{4.20}$$

The modulo operation of equation (4.19) makes it a non-invertible equation. But it is still a reversible one. Thus, in the decryption part of the proposed cryptosystem, the reverse layer is also achieved by equation (4.19).
The implementation of this modified 2-D cat map is carried out by an optimized process, indeed:
- the value of $Z_1 = ri + rj$ is calculated once per round.
- the value of $Z_2 = u \times v + 1$ is calculated once per round.
- the value of $Z_3 = v \times i + rj$ is calculated M times per round.

Then the modified 2-D cat map is implemented as:

$$\begin{bmatrix} i_n \\ j_n \end{bmatrix} = Mod\left(\begin{bmatrix} i + u \times j + Z_1 \\ Z_3 + Z_2 \times j \end{bmatrix}, \begin{bmatrix} M \\ M \end{bmatrix}\right) \tag{4.21}$$

The Logistic map is a non-linear chaotic discrete function that produces random sequences. In the proposed cryptosystem, the logistic map is used as a diffusion function to achieve the diffusion effect, by transferring the effect from one byte in the block to other bytes in the same block. This structure makes the proposed cryptosystem highly sensitive to the plaintext. The mathematical model of the discrete logistic map is:

$$X_{k+1} = \begin{cases} \left\lfloor \frac{X_k \times (2^N - X_k)}{2^{N-2}} \right\rfloor & \text{if} X_k \neq \left[3 \times 2^{N-2},\ 2^N\right] \\ 2^N - 1 & \text{if} X_k = \left[3 \times 2^{N-2},\ 2^N\right] \end{cases} \tag{4.22}$$

where X_{k+1} is the new value calculated from the previous one X_k. N is the number of bits representing the integer output of the discrete logistic map, which is equal to 32 bits in all versions of our proposed cryptosystem. Figure 4.7 shows the block diagram of the encryption part of the first version of the proposed cryptosystem. From the figure, we write the encryption mathematical model as:

$$c_l(k_n) = LSB_8[y_l(k)] \tag{4.23}$$

$$y_l(k) = p_l(k) \oplus s_{l-1}(k) \oplus f(y_l(k-1)) \tag{4.24}$$

where $y_l(k)$ is a 32-bit variable, $p_l(k),\ s_{l-1}(k)$ are 8-bit variables and f is the logistic map. The following remarks should be considered:

1. During the encryption, equation (4.24) should be evaluated before equation (4.23), for each byte of a block and for all blocks.

2. The input of the logistic map for $k = 0$ is kd_m when $l = 0$ and it is $y_{l-1}(b_s - 1)$ for $l > 0$.

Algorithm 7 Encryption steps

Generate the IV values using the chaotic generator to encrypt the first block of the image.
for $m = 0$: $r - 1$: $step = 1$ do
  Generate the values of Kd_m and Kp_m using the chaotic generator to encrypt the first block.
  $k = 0$
  for $i = 0$: $M - 1$: $step = 1$ do
    for $j = 0$: $M - 1$: $step = 1$ do
      Initialize the value of $s_{-1}(k) = iv(k)$
      Calculate $(i_n,\ j_n)$ using equation (4.19)
      Calculate $k_n = i_n \times M + j_n$
      Calculate $y_0(k)$ value using equation (4.22) and equation (4.24)
      Calculate $c_0(k_n)$ value using equation (4.23)
      $k = k + 1$
    End j
  End i
End m
for $l = 1$: $b_n - 1$: $step = 1$ do
  for $m = 0$: $r - 1$: $step = 1$ do
  Generate the values of Kp_m using the chaotic generator to encrypt the current block of the plain
  image.
  $k = 0$
    for $i = 0$: $M - 1$: $step = 1$ do
      for $j = 0$: $M - 1$: $step = 1$ do
        Initialize the value of $s_{l-1}(k) = c_{l-1}(k)$
        Calculate $(i_n,\ j_n)$ using equation (4.19)
        Calculate $k_n = i_n \times M + j_n$
        Calculate $y_l(k)$ value using equation (4.22) and equation (4.24)
        Calculate $c_l(k_n)$ value using equation (4.23)
        $k = k + 1$
      End j
    End i
  End m
End l

3. For $k > 0$ and for all l, the input of the logistic map is the result of equation (4.24) and not the previous output (see equation 4.22).

Note that:
$k = i \times M + j$
$k_n = i_n \times M + j_n$
i_n and j_n are calculated using equation (4.19). The sequence $s_{l-1}(k)$ is given by the following equation:

$$s_{l-1}(k) = \begin{cases} iv(k) & \text{if } l = 0 \\ c_{l-1}(k) & \text{if } l > 0 \end{cases} \tag{4.25}$$

where
$l = 0,\ 1,\ 2,\ ...b_n - 1$
$k = 0,\ 1,\ 2,\ ...b_s - 1$
$IV = \{iv(0),\ iv(1),\ iv(2),\ ...,\ iv(b_s - 1)\}$
b_s is block size in bytes
$b_n = \frac{\text{image size}}{\text{block size}} = \frac{L \times C \times P}{b_s}$, is the number of blocks.
with, L, C, and P are the number of lines, the number of columns, and the number of planes of the image respectively.
In algorithm-7, we describe in detail the exact steps to achieve the ciphering process for all blocks.

4.2.3.2 Decryption scheme of the first proposed cryptosystem

The decryption scheme of the first proposed cryptosystem is almost identical to the encryption one. Figure 4.8 shows the decryption structure of the CBC mode in all proposed cryptosystems, while Figure 4.9 shows the decryption structure of the first proposed cryptosystem.

The decryption equation resulting from the encryption equations is: (4.23) and (4.24):

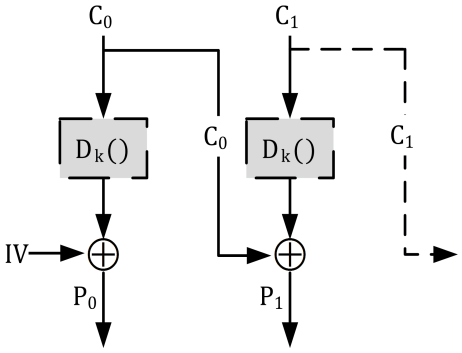


Figure 4.8: Decryption structure of the CBC mode

$$p_l(k) = c_l(k_n) \oplus s_{l-1}(k) \oplus LSB_8[f(y_l(k-1))] \tag{4.26}$$

where

$$y_l(k) = p_l(k) \oplus s_{l-1}(k) \oplus f(y_l(k-1)) \tag{4.27}$$

As we can see, equation (4.26) is firstly evaluated followed by equation (4.27) for each byte of a block and for all blocks.

Algorithm 8 Decryption steps

Generate $Kd_m,\ Kp_m$ for all m values (i.e., $m = 0,\ 1,\ ...r-1$) and IV using the chaotic generator to decrypt the first ciphered block.
for $m = r-1$: 0: $step = -1$ do
  $k = 0$
  for $i = 0$: $M-1$: $step = 1$ do
    for $j = 0$: $M-1$: $step = 1$ do
      Initialize the value of $s_{-1}(k) = iv(k)$
      Calculate $(i_n,\ j_n)$ using equation (4.19)
      Calculate $k_n = i_n \times M + j_n$
      Calculate $p_0(k)$ value using equation (4.26)
      Calculate $y_0(k)$ value using equation (4.22) and equation (4.24)
      $k = k + 1$
    End j
  End i
End m
for $l = 1$: $b_n - 1$: $step = 1$ do
  Generate the Kp_m for all m values (i.e., $m = 0,\ 1...r-1$)
  using the chaotic generator to decrypt the current block from the ciphered image.
  for $m = r-1$: 0: $step = -1$ do
    $k = 0$
    for $i = 0$: $M-1$: $step = 1$ do
      for $j = 0$: $M-1$: $step = 1$ do
        Initialize the value of $s_{l-1} = c_{l-1}k$
        Calculate $(i_n,\ j_n)$ using equation (4.19)
        Calculate $k_n = i_n \times M + j_n$
        Calculate $p_l(k)$ value using equation (4.26)
        Calculate $y_l(k)$ value using equation (4.22) and equation (4.24)
        $k = k + 1$
      End j
    End i
  End m
End l

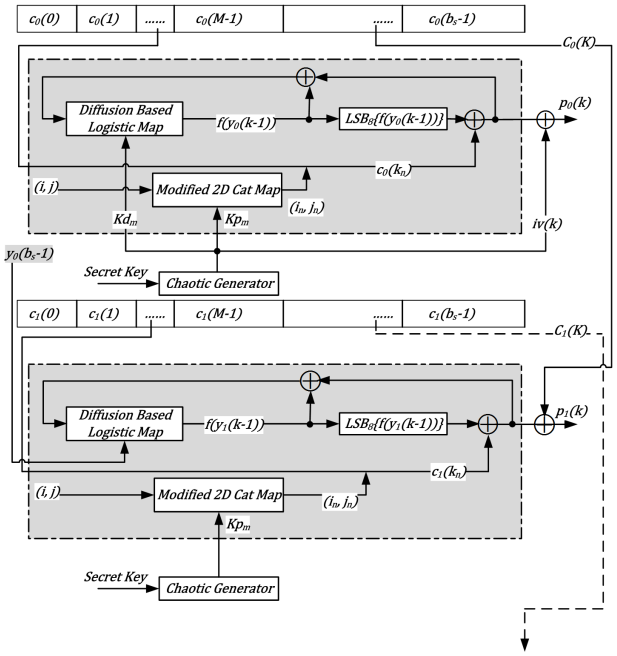


Figure 4.9: Decryption structure of the first proposed cryptosystem

In reality, as we can see in Figure 4.9, equation (4.27) is implemented as follows:
$y_l(k) = c_l(k_n) \oplus s_{l-1}(k) \oplus LSB_8[f(y_l(k-1))] \oplus s_{l-1}(k) \oplus f(y_l(k-1))$

$$y_l(k) = c_l(k_n) \oplus LSB_8[f(y_l(k-1))] \oplus f(y_l(k-1) \tag{4.28}$$

4.2.3.3 Analysis of the first proposed cryptosystem

Lian et al., in their paper [74], analyzed the Fridrich model, and they pointed out some possible attacks on that model. We apply and analyze these attacks on our proposed cryptosystem to ensure its robustness against them.

4.2.3.4 Dynamic key space analysis of Fridrich, Zhang and our cryptosystems

Lain in his paper assumes that the dynamic key space of the whole Fridrich cryptosystem is $S^r = (S_1 \times S_2)^r$, where S_1 is the dynamic key space of the confusion layer, S_2 is the dynamic key space of the diffusion layer, and r is the number of iterations.
The total dynamic key space of the Frdirich model can be calculated as:
$S_1 = (N^2)^r$
$S_2 = L^r$
$KS_{Fridrich} = (N^2)^r \times L^r = (N^2 \times L)^r$
where M is the square root of the tested image size and $L = 256$, is the number of gray levels.
In the Zhang cryptosystem, the first algorithm has q_1, q_2, p_1 and p_2 in the range [0, 511], and also t_0 which is 8 bits, so:
$KS_{Zhang1} = (N^4 \times 2^8)^r$.
The second algorithm has q_1, q_2, p_1 and p_2 in the range [0, 511], and also $temp1$ and $temp2$ of 8 bits each, so:

$KS_{Zhang2} = (N^4 \times 2^{16})^r$.
In our proposed cryptosystem, the total dynamic key space is defined as:
$S^r = (S_1 \times S_2)^r$ where S_1 is the dynamic key space of the confusion layer (the modified 2-D cat map), S_2 is the dynamic key space of the diffusion layer (the logistic map).
The total dynamic key space of our proposed cryptosystem can be calculated as:

1. The whole image is divided into a number of blocks and the dynamic keys are changed for every new encryption round and every new block.

2. Dynamic keys for the confusion layer are (u, v, ri and rj: in the range of $[0,\ M-1]$). Then, the confusion key space is M^4.

3. The diffusion key space (S_1) is 32 bit instead of 8 bit as in the Fridrich or the Zhang cryptosystems.

4. The total number of blocks is b_n where $b_n = \frac{L \times C \times P}{b_s} = \frac{L \times C \times P}{M^2}$.

$KS = (S_1 \times S_2)^r \times b_n$
$S_1 = M^4$
$S_2 = 2^{32}$
For $r = 1$:
$KS_{Proposed} = (M^4 \times 2^{32}) \times \frac{L \times C \times P}{M^2}$
As an example and to make the comparison between cryptosystems, of Zhang, Fridrich and ours, the gray-scale Lena image of 512×512 is taken, and then for one encryption round (remark, for our proposed cryptosystem $M = 32$):
$KS_{Fridrich} = 512^2 \times 256 = 2^{26}$
$KS_{Zhang1} = 512^4 \times 2^8 = 2^{44}$
$KS_{Zhang2} = 512^4 \times 2^{16} = 2^{52}$
$KS_{Proposed} = (32^4 \times 2^{32}) \times (\frac{512 \times 512}{32^2}) = 2^{60}$
It is clear from the previous calculations that the dynamic key space of our proposed cryptosystem is $\mathbf{2^{16}}$ times more than the first Zhang cryptosystem, and $\mathbf{2^8}$ times more than the second Zhang cryptosystem.

4.2.3.5 Chosen-plaintext attack

Fridrich proved that his proposed model is secure against a chosen-plaintext attack based on the fact that the difference between the ciphertexts encrypted by the same key for two plaintexts differs on one bit is large enough to keep a high security level against the chosen-plaintext attack. However, Lian in [74] pointed out another kind of attack that can be used to cryptanalyze the Fridrich model. Since the fixed-point problem was not solved in the 2-D cat map, Baker or standard maps were used in the Fridrich model, and so the cipher of the first plain pixel of any image will remain in the first position (that means c_0 is the encryption of the p_0, and so, no permutation is done on the first pixel). Then it is easy to find the initial value of the diffusion key (Q_{-1}) in the Fridrich model (more details of this attack are in [74]).

In Zhang cryptosystems, a simple solution to overcome this problem is introduced, the solution is carried out by swapping the first pixel with a random pixel from the image before starting the encryption process.

In our proposed cryptosystem, the first kind of chosen-plaintext attack is solved, and it satisfies a high security level. This is well stated by our proposed cryptosystem in section (4.2.6).
The fixed-point problem does not exist in our proposed cryptosystem, since it is solved by adding the ri and the rj dynamic keys to the original 2-D cat map, and the chaotic generator insures that ri and rj will never be zero at the same time.
Our solution has two advantages, first, it increases the dynamic key space of the 2-D cat map, second, any pixel can be mapped to any position with the same probability.

4.2.3.6 Some specific differences in the diffusion process

Zhang cryptosystems implement the logistic map based on 8 bits. This means, that the input and the output of the logistic map for all operations are in 8 bits, whereas in our proposed cryptosystem, the logistic map is implemented to receive 32 bits as input and to produce 32 bits as output. As a result, the dynamic key of the logistic map is increased from 8 bits to 32 bits, and the security level of the overall cryptosystem is increased. Finally, it is well known that the implementation of the logistic map based on 8 bits has some security weaknesses and failures.
Finally, as the proposed cryptosystem uses new dynamic keys for each new block, then, the cryptosystem is secure against the chosen-plaintext attacks, according to the fact that a chosen-plaintext attack will be useless if different keys are used to encrypt different plaintexts in the same message.

4.2.4 Second proposed cryptosystem

The construction and design process of the following cryptosystems begins by keeping in mind the structure of the first proposed cryptosystem. As we know, there is always a trade off between the security level and the encryption time. Increasing the security level in general leads to making the system more complex, and then adding some additional delays on the encryption operations. For this reason, two sub-versions of the second version are described in this section. These sub-versions have the same structure as the first proposed cryptosystem, but using a different diffusion layer, namely the Finite Skew Tent Map (FSTM) as a generator instead of the logistic map.

4.2.4.1 Finite Skew Tent Map as diffusion layer

The diffusion layer is implemented using a modified version as a generator of the original FSTM in [84], [83], based on a lookup table of 8 bits for the first subversion and on a mathematical calculation of 32 bits for the second subversion (see equation (4.29)). The FSTM has a better non-linear transformation than the logistic map and so its diffusion is stronger than that of the logistic map. This implies that the cryptosystem is more resistant against differential cryptanalysis attacks. So, using the FSTM as a dependent diffusion layer increases cryptosystem sensitivity to plain sensitivity attacks. The mathematical model of the modified FSTM generator is [39]:

$$F(X) = \begin{cases} \left\lfloor \frac{Q}{A_m} \times X \right\rfloor + A_{0m} \; Mod \, Q & 0 \leq X \leq A_m \\ \left\lfloor \frac{Q \times (Q-X)}{Q-A_m} \right\rfloor + 1 + A_{0m} \; Mod \, Q & A_m < X < Q \end{cases} \tag{4.29}$$

Where
$A_{0m},\ X,\ F(X) \in \{0,\ 1,\ 2,\ ...,\ Q-1\}$, and $A_m \in \{1,\ 2,\ ...Q-1\}$.
In equation (4.29), Q is equal to (2^8) for the first subversion (lookup table), and equal to (2^{32}) for the second subversion (the mathematical implementation of the FSTM). Relative to the first proposed cryptosystem, the input value X of equation (4.29) is equation (4.24), while $F(X)$ in equation (4.29) is $f(y_l(k-1)$ and the initial value X_0 is Kd_m (see Figures 4.7 and 4.9).
The structure of the dynamic keys during the diffusion process is:

$$\begin{aligned} Ks &= \lfloor Ks_0 \parallel Ks_1 \parallel Ks_2 \parallel \cdots \parallel Ks_{r-1} \rfloor \\ Ks_m &= A_m || A_{0m} \end{aligned} \tag{4.30}$$

where r is the number of rounds for each block. In the standard FSTM, the fixed-point problem is not solved (i.e., when the input of the FSTM is ZERO the output is ZERO). To overcome this problem we introduce A_{0m} in the FSTM equation. As a result, any input value is mapped to any output value with the same probability without any restrictions. Moreover, introducing the dynamic key A_{0m} increases the dynamic key space.
The first sub-version uses 8 bits to implement the FSTM generator as a lookup table, and so it is faster than the first cryptosystem, while still having a high security level. The lookup table is created since the input and the output of the FSTM are limited to 8 bits. Figure 4.7 can be used to describe the encryption process of this sub-version. The first step is to generate the dynamic keys (Kp_m, X_0, A_m and A_{0m}). The permutation process is applied on the plain pixels by taking each byte, and calculating their new position according to the equation (4.19). Then, equation (4.24) is applied to obtain the y_k value which defines the ciphered pixel as shown in equation (4.23). Note that the value of y_k is 8 bit and so there is no need for the LSB function of equation (4.23). It is important to note that equation 4.29 is implemented in lookup table of 64 KB size without the A_{0m}, the returned value from the lookup table is added to the A_{0m}.

The second sub-version uses 32 bits to implement the FSTM as a generator. It is slower than the first cryptosystem, but it has a dynamic key space greater than the first cryptosystem and thus it is very robust against cryptanalysis. The encryption process in this version is exactly identical to the first one, except in this version equation (4.29) is used instead of equation (4.22).

Using the lookup table based on 8 bits for the first sub-version, the first 8 bits from each sample of the chaotic generator are taken to be used as the dynamic key (A_m or A_{0m}), and the remaining 24 bits are skipped. If the first 8 bits are zeros, then the next 8 bits are taken and so on. The dynamic keys of the first sub-version need two samples for each round of each block. It is important to note that the used chaotic generator never produces a sample of 32 bits where all of the bits are zeros. The dynamic keys A_m and A_{0m} in the second sub-version are 32 bits each, so, two samples are also needed for each round of each block.

$$Ks_{samples} = 2 \times r \tag{4.31}$$

So the total number of required samples in this version is:

$$Total_{samples} = 2 \times r + \frac{b_s}{4} + \left\lceil \frac{b_n \times r \times 4 \times \log_2 \sqrt{b_s}}{32} \right\rceil \tag{4.32}$$

4.2.4.2 Analysis of the second proposed cryptosystem

In this section, we analyze the dynamic key space and the chosen-plaintext attacks.

4.2.4.3 Dynamic key space analysis

In our proposed cryptosystem, the total dynamic key space is:
$KS = (S_1 \times S_2)^r \times b_n$
For one encryption round ($r = 1$):
First subversion key space:
$S_1 = M^4$
$S_2 = 2^{24}$, because X_0, A_m and $A_0 m$ are 8 bits each.
$KS_{Subversion1} = M^4 \times 2^{24} \times \frac{L \times C \times P}{M^2}$
Second subversion key space:
$S_1 = M^4$
$S_2 = 2^{96}$, because X_0, A_m and $A_0 m$ are 32 bits each.
$KS_{Subversion2} = M^4 \times 2^{96} \times \frac{L \times C \times P}{M^2}$

Again, to make the same comparison between our proposed cryptosystem, the Zhang and the Fridrich cryptosystems, the Lena image of 512×512 bytes is taken, then for one encryption round (remark, for our proposed cryptosystem $M = 32$):
$KS_{Fridrich} = 2^{26}$
$KS_{Zhang1} = 2^{44}$
$KS_{Zhang2} = 2^{52}$
$KS_{Subversion1} = (32^4 \times 2^{24}) \times \frac{512 \times 512}{32^2} = 2^{52}$
$KS_{Subversion2} = (32^4 \times 2^{96}) \times \frac{512 \times 512}{32^2} = 2^{124}$
It is clear from the previous calculations that the dynamic key space of the first subversion is $\mathbf{2^8}$ times more than the first Zhang cryptosystem, and the same as the second Zhang cryptosystem, whereas the dynamic key space of the second subversion is $\mathbf{2^{80}}$ times more than the first Zhang cryptosystem, and $\mathbf{2^{72}}$ times more than the second Zhang cryptosystem.

4.2.4.4 Chosen-plaintext attack

In this cryptosystem version, the modified 2-D cat map is the same as before, and so the analysis of the chosen-plaintext attack is the same. The FSTM is enhanced by adding the parameter A_0. This means that the fixed-point problem is solved also for the diffusion layer, and so, this type of attack will be useless.

4.2.5 Time and complexity analysis

The speed evaluation of our proposed cryptosystem in all versions is carried out using the same environment and parameters which are used in section (3.4.1). It encrypts different images of different sizes. ($256 \times 256 \times 3$, $512 \times 512 \times 3$ and $1024 \times 1024 \times 3$). We compare the speed of our proposed cryptosystems with the fastest chaos-based cryptosystems. In particular, the security and the performance analysis of the proposed cryptosystems are compared with [166] cryptosystem, since, to the best of our knowledge, Zhang cryptosystem is the fastest chaos-based cryptosystem.
Table 4.3 presents the encryption and the decryption times for our proposed algorithms,

based on two different block sizes ($b_s = 256$ and $b_s = 1024$ bytes) and the image under test was Lena. The time calculation process of encryption and decryption is evaluated as follows: the test image (Lena with block size 1024) is encrypted for 1000 different secret keys, then the average of these executions is calculated. From Table 4.4, it is clear

Table 4.3: Average encryption/decryption time of the proposed algorithm(in milliseconds)

Our Cryptosystem Version	bs	$256 \times 256 \times 3$	$512 \times 512 \times 3$	$1024 \times 1024 \times 3$
Proposed V1	256	2.21/2.82	8.75/11.16	34.87/44.34
Proposed V1	1024	2.04/2.68	8.08/10.57	31.85/41.83
Proposed V2-8bit	256	1.73/1.78	6.82/7.10	27.01/28.04
Proposed V2-8bit	1024	1.38/1.45	5.42/5.74	21.17/22.39
Proposed V2-32bit	256	4.57/5.24	18.29/20.89	73.05/83.43
Proposed V2-32bit	1024	4.15/4.79	16.56/19.08	66.12/76.17

Table 4.4: Encryption/decryption time of different algorithms(in milliseconds)

Proposed Cryptosystem	$256 \times 256 \times 3$	$512 \times 512 \times 3$	$1024 \times 1024 \times 3$
Proposed V1	2.04/2.68	8.08/10.57	31.85/41.83
Proposed V2-8bit	1.38/1.45	5.42/5.74	21.17/22.39
Proposed V2-32bit	4.15/4.79	16.56/19.08	66.12/76.17
Zhang 1[166]	7.5/7.5	30/30	120/120
Zhang 2[166]	7.5/8.25	30/33	120/132
Wang [152]	7.79/8.39	31.16/33.54	124.64/134.16
Akhshani [8]	14.4	57.6	230.4
Wong [155]	15.59/16.77	62.37/67.11	249.48/268.44
Pareek [94]	160	920	5650

that our proposed cryptosystems are faster than both of the Zhang algorithms and other known cryptosystems. Table 4.5 presents a comparison of performance of our proposed cryptosystem with some known recent cryptosystems in the literature. The performance is evaluated in terms of encryption throughput (running speed) in Mega Byte Per Second (MBps) and number of needed cycles to encrypt/decrypt one byte. As it can be seen in Tables (4.3-4.5), our proposed cryptosystem in all proposed versions is faster than the other chaos-based cryptosystems.

Table 4.5: Encryption throughput and Number of cycles for one encrypted byte

Proposed Cryptosystem	ET in MBps	Number of cycles per byte
Proposed V1	93.817/71.486	31.51/41.35
Proposed V2- 8 bits	140.776/133.114	21/22.21
Proposed V2-32 bits	45.347/39.359	65.19/75.11
Zhang 1 [166]	25/25	122.07/122.07
Zhang 2[166]	25/22.72	122.07/134.27
Wang [152]	24.06/22.35	122.85/132.24
Akhshani [8]	13.02	194.83
Wong [155]	12.03/11.18	245.7/264.38
Pareek [94]	0.39	2445

4.2.6 Plain-text sensitivity attack

The mathematical equations of chapter 2 (see section 2.2.2) are used to produce the presented results of Table 4.6. All versions of our proposed cryptosystem achieve the avalanche effect from the first round ($r = 1$) and then, they overcome the plaintext sensitivity attacks. The plain images under test were Lena, Barb, and Boat images of the same size 512×512 gray scale images (the selected images are chosen like those in the literature). Moreover, Lena and Peppers color images of the same size were used on the same test. Table 4.6 presents all of these results, and it is clear that the HD value is very close to the optimal value of 50% for the three proposed cryptosystems. Also the $UACI$ and $NPCR$ values are close to optimal. These values indicate that the proposed cryptosystems are very sensitive to one bit change in the plaintext. Hence, a high security level is reached.

Table 4.6: HD, UACI and NPCR plaintext sensitivity tests

Proposed Cryptosystem	Image Name	Image Size	HD	UACI	NPCR
Proposed V1	Barb	$512 \times 512 \times 1$	0.499630	33.438	99.532
Proposed V2-8 bit	Barb	$512 \times 512 \times 1$	0.499975	33.462	99.611
Proposed V2-32 bit	Barb	$512 \times 512 \times 1$	0.499978	33.460	99.609
Zhang 1 [166]	Barb	$512 \times 512 \times 1$	/	33.475	99.663
Zhang 2 [166]	Barb	$512 \times 512 \times 1$	/	33.420	99.582
Proposed V1	Lena	$512 \times 512 \times 1$	0.499587	33.437	99.521
Proposed V2-8 bit	Lena	$512 \times 512 \times 1$	0.499986	33.463	99.611
Proposed V2-32 bit	Lena	$512 \times 512 \times 1$	0.499975	33.459	99.609
Pareek [94]	Lena	$512 \times 512 \times 1$	/	31.79	99.6
Wong [155]	Lena	$512 \times 512 \times 1$	/	32.82	99.44
Wang [152]	Lena	$512 \times 512 \times 1$	/	33.435	99.607
Proposed V1	Boat	$256 \times 256 \times 1$	0.499576	33.434	99.524
Proposed V2-8 bit	Boat	$256 \times 256 \times 1$	0.499993	33.461	99.611
Proposed V2-32 bit	Boat	$256 \times 256 \times 1$	0.499955	33.466	99.609
Song [133]	Boat	$256 \times 256 \times 1$	33.453	99.625 /	
Akhshani [8]	Boat	$256 \times 256 \times 1$	0.499900	33.200	/
Proposed V1	Lena	$512 \times 512 \times 3$	0.499823	33.454	99.579
Proposed V2-8 bit	Lena	$512 \times 512 \times 3$	0.500001	33.466	99.611
Proposed V2-32 bit	Lena	$512 \times 512 \times 3$	0.499981	33.466	99.610
Proposed V1	Peppers	$512 \times 512 \times 3$	0.499853	33.451	99.576
Proposed V2-8 bit	Peppers	$512 \times 512 \times 3$	0.500035	33.466	99.610
Proposed V2-32 bit	Peppers	$512 \times 512 \times 3$	0.500001	33.463	99.609

4.2.7 Key sensitivity attack

To evaluate the proposed cryptosystems in terms of key sensitivity attacks, the previous described equations of the $NPCR$, $UACI$ and HD(see section 2.2.3) are used.

Table 4.7presents the results obtained from the key sensitivity attack test for the three versions of the proposed cryptosystem using the same parameters as were used in Table 4.6. From Table 4.7, it is clear that the proposed cryptosystem has a high security

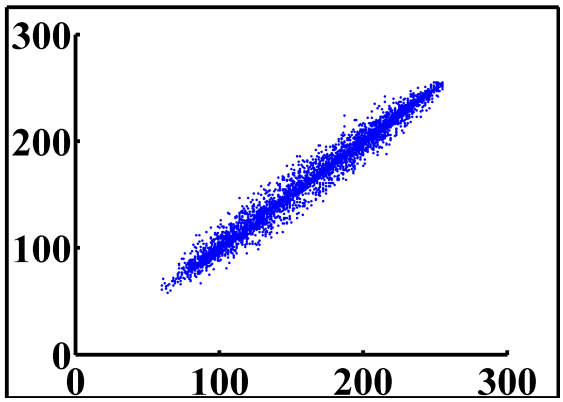
Table 4.7: HD, UACI and NPCR key sensitivity tests

Proposed Cryptosystem	Image Name	Image Size	HD	UACI	NPCR
Proposed V1	Barb	$512 \times 512 \times 1$	0.500029	33.464	99.610
Proposed V2-8 bit	Barb	$512 \times 512 \times 1$	0.499980	33.463	99.609
Proposed V2-32 bit	Barb	$512 \times 512 \times 1$	0.499989	33.461	99.609
Proposed V1	Lena	$512 \times 512 \times 1$	0.500014	33.463	99.610
Proposed V2-8 bit	Lena	$512 \times 512 \times 1$	0.499995	33.464	99.608
Proposed V2-32 bit	Lena	$512 \times 512 \times 1$	0.499952	33.465	99.608
Proposed V1	Boat	$256 \times 256 \times 1$	0.500015	33.462	99.609
Proposed V2-8 bit	Boat	$256 \times 256 \times 1$	0.499988	33.462	99.608
Proposed V2-32 bit	Boat	$256 \times 256 \times 1$	0.499995	33.460	99.609
Proposed V1	Lena	$512 \times 512 \times 3$	0.500007	33.464	99.610
Proposed V2-8 bit	Lena	$512 \times 512 \times 3$	0.499992	33.463	99.609
Proposed V2-32 bit	Lena	$512 \times 512 \times 3$	0.499994	33.465	99.609
Proposed V1	Peppers	$512 \times 512 \times 3$	0.500001	33.465	99.609
Proposed V2-8 bit	Peppers	$512 \times 512 \times 3$	0.499987	33.461	99.610
Proposed V2-32 bit	Peppers	$512 \times 512 \times 3$	0.499998	33.465	99.609

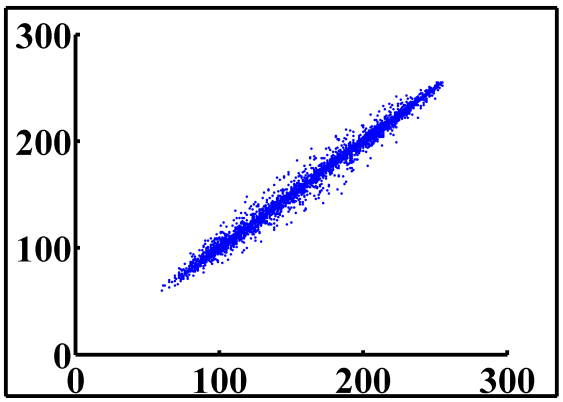
level relative to the key sensitivity attacks.

4.2.8 Correlation analysis

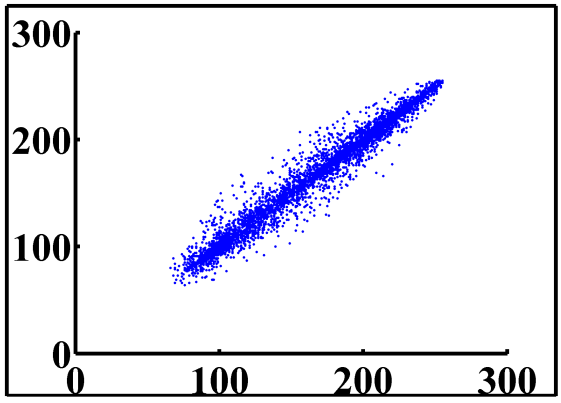
To test the correlation between adjacent pixels, the following procedure was carried out. Firstly, 8000 pairs of two adjacent pixels are selected randomly in vertical, horizontal, and diagonal directions from the original and the encrypted images. Then, the correlation coefficient is computed according to the equations (2.12-2.15), of section (2.2.5) of chapter 2.



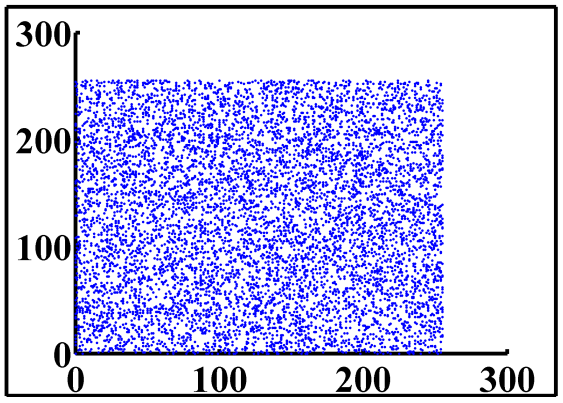
(a) Horizontal correlation of the plain image



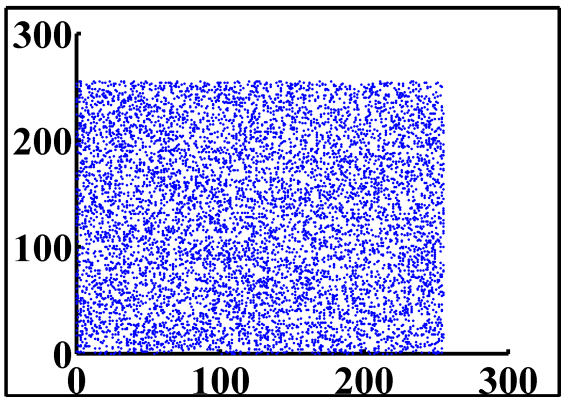
(b) Vertical correlation of the plain image



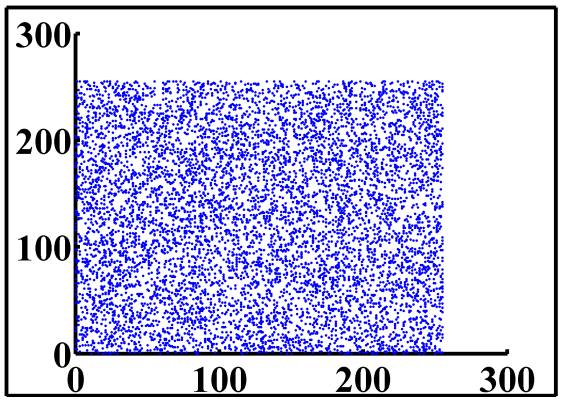
(c) Diagonal correlation of the plain image



(d) Horizontal correlation of the ciphered image



(e) Vertical correlation of the ciphered image



(f) Diagonal correlation of the ciphered image

Figure 4.10: Correlation analysis of Lena and its ciphered image in three directions

One example of this extensive study is the Barb gray scale image of $512 \times 512 \times 1$. The obtained results are shown in Table 4.8 and Figure 4.10. These results demonstrate that the correlation coefficient, in all directions, of the plain images is close to one (see Figure 4.10), and the correlation coefficient of the encrypted images is close to zero. This means that there is no detectable correlation between the original and its corresponding ciphered image, and also, there is no relation between pixels of the ciphered image.

Table 4.8: Correlation Analysis Results

Cryptosystem Name	Horizontal	Vertical	Diagonal
Proposed Algorithm V1	0.0085	0.0097	0.0092
Proposed Algorithm V2-8 bit	0.0087	0.0098	0.0096
Proposed Algorithm V2-32 bit	0.0087	0.0098	0.0096

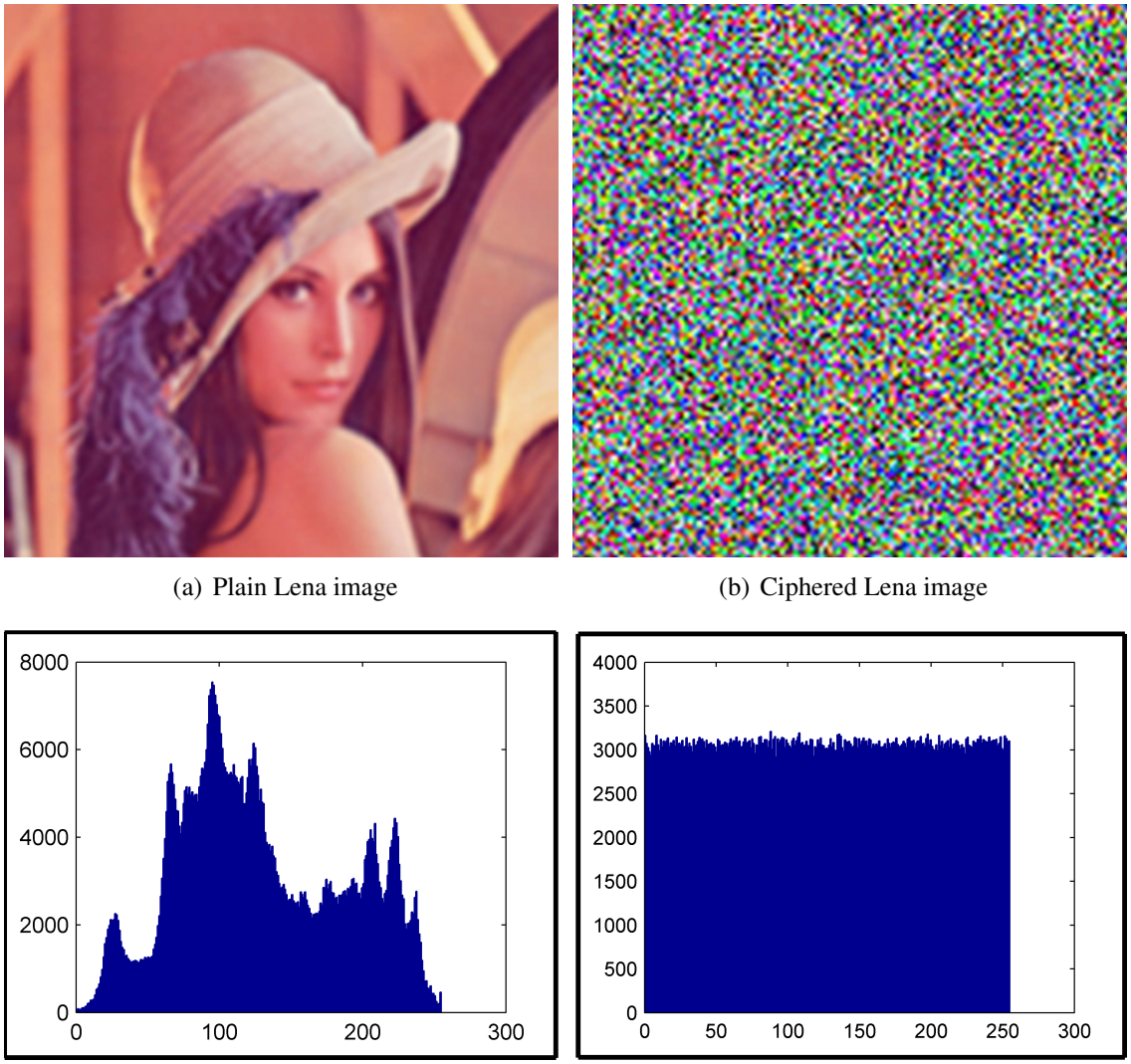
4.2.9 Histogram analysis

In Figure 4.11, we show some visual results obtained with the third version of the proposed cryptosystem (similar results are obtained for the other versions): a) the plain Lena image of size $512 \times 512 \times 3$, b) the corresponding cipher image, c) the histogram of the plain image, and d) its corresponding cipher image. The histogram of the encrypted image is very close to the uniform distribution and completely different from the plain image histogram. This means that there is no visual information than can be observed from the ciphered image of the proposed cryptosystem.

Table 4.9: Chi-Square Results

Crypto version	Ciphered image	Chi-square
Proposed V1	Lena	256.27
	Boat	262.46
	Baboon	259.91
Proposed V2-8bit	Lena	259.63
	Boat	254.69
	Baboon	258.78
Proposed V2-32bit	Lena	253.12
	Boat	252.44
	Baboon	253.14

The visual test is necessary but is not sufficient. To ensure uniformity, the chi-square test is applied to statistically confirm the uniformity of the histogram.



(a) Plain Lena image

(b) Ciphered Lena image

(c) Histogram of the plain Lena image

(d) Histogram of the ciphered Lena image

Figure 4.11: Lena image and its ciphered version and their corresponding histograms

We present in Table 4.9 the results obtained from the chi-square test of histograms for three ciphered images of different nature (i.e., Lena, Boat, and Baboon). All of them have the same size of $128 \times 128 \times 3$, with a significant level of 0.05. From the obtained values, we can observe that $\chi^2_{exp} < \chi^2_{th}(255,\ 0.05) = 293$, and then the tested histograms are uniform and do not reveal any information for statistical analysis.

4.3 Example of a real-time application

An example of a real-time application of all realized chaos-based cryptosystems could be in real-time energy harvesting sensor.
A wireless sensor network consists of sensor nodes deployed over a geographical area for monitoring physical phenomena like temperature, humidity, vibrations, seismic events, and sensitive information such as enemy movement on the battlefield or the location of persons in a building. Typically, a sensor node is a tiny device that includes four basic components: a sensing subsystem for data acquisition from the physical surrounding environment, a processing subsystem for local data processing and storage, a wireless communication subsystem for data transmission, and a power source to supply the energy needed by all the subsystems to perform the programmed tasks. The power source often consists of a battery with limited life time. In many applications, it is impossible or costly to recharge the battery because nodes may be deployed in hostile or unpractical environments. The sensor network should have a lifetime long enough to fulfill the application requirements. Therefore, the question is how to prolong the network lifetime to such a long time [10].
Sensors communicate broadcasted messages to each other which makes them vulnerable to different types of attack, such as eavesdropping, malicious modification on insertion of unauthorized data in packages [151]. To prevent attacks, we should implement security services in WSNs. Among them, the most important are confidential, authentication of nodes as well as data integrity. Most of traditional security mechanisms are not sufficiently efficient. All existing algorithms including cryptographic algorithms adapted to WSNs aim to reduce the energy consumed and thus to maximize the lifetime of the networks. Nevertheless, in the paradigm of energy harvesting sensor networks, the computing/communication system needs to be energy-neutral i.e. to consume as much energy as harvested. In this context, new algorithms have to be developed for power management, routing and other networking issues.
In this section, we address the question of security in new embedded systems which are supplied by ambient energy. A variety of techniques are available for energy harvesting, including solar and wind powers, piezoelectricity, thermoelectricity, and physical motions. Energy harvesting is perfectly convenient for wireless electronic devices that otherwise rely on battery power. In the energy harvesting paradigm, the lifetime of a network can be considered as infinite. A central question that remains open is about the possibility of fully secure communications with energy harvested sensors.
We propose an on-line strategy based on software redundancy which, at any time permits to select the adequate encryption algorithm so as to optimize the resulting Quality of Service measured in terms of security and rapidity. Further, we propose a security-aware scheduling strategy which Quality of Service measured in terms of security and rapidity. Further, we propose a security-aware scheduling strategy which incorporates the Earliest Deadline First (EDF) scheduling algorithm.

4.3.1 Real-time computing

4.3.1.1 Issues in conventional real-time computing systems

Data sensing and retrieval in wireless sensor systems have a widespread application. The software on a node generally comprises a set of periodic tasks that consist of streams of jobs. Task periods are usually set by the application requirements. Every job is char-

acterized by a release time, an execution requirement and a deadline. The goodness of such real-time system depends on whether all the jobs of all the tasks can be guaranteed to complete their executions before deadlines. If they can, then we say the task set is feasible.
It has been established for about forty years that the EDF scheduling policy which assigns priorities according to urgency is optimal [75]. In case of timing constraints cannot be met, one way is to trade computation quality for timeliness. This is often achieved with software redundancy in order to provide a graceful degraded mode. Indeed, a real-time system should continue to operate even in the presence of time starvation or energy starvation with a lower but acceptable Quality of Service.

4.3.1.2 The deadline mechanism

The so-called Deadline Mechanism (DM) provides software redundancy in hard real-time periodic task systems [24]. Each task has two versions: primary and alternate (also called back-up). The primary version contains more functions and produces good quality results, but requires high computation time (and high energy requirement). The alternate version contains only the minimum required functions and produces less precise results with minimum execution requirements. However, if the primary of a task fails (due to time or energy starvation), the execution of the associated alternate should be guaranteed before deadline. The challenge in the implementation of the deadline mechanism is consequently twofold:

- How to guarantee that either the primary or the alternate version of each task be completed in time and.
- How to complete as many primaries as possible to optimize the quality of service.

4.3.1.3 Scheduling framework

In [25] the so-called last chance strategy is proposed to maximize the number of successful primaries. An offline scheduler reserves time intervals for the alternates. Each interval is chosen so that any alternate starts execution at the latest possible time. At run-time, the primaries are processed during the remaining intervals before their respective alternate. Alternates can preempt any primary when a time interval reserved for the alternates overlaps the execution interval of primaries. Whenever a primary completes successfully, the execution of its corresponding alternate is no longer needed. Hence, the online scheduling algorithm dynamically rearranges the alternate schedule so as to increase processor time available for the execution of primaries (see Figure.4.13).

4.3.2 Security in energy harvesting systems

4.3.2.1 System model

Our system consists of four components: the energy source, the energy harvester, the energy storage and the sensor node as it is seen in Figure.4.12. The ambient energy source is characterized by an instantaneous charging rate $P_r(t)$ that incorporates all losses. We assume that energy production times can overlap with the consumption times. The energy produced by the source is not considered as controllable and not necessarily predictable. Our system uses an ideal energy storage unit that has a nominal capacity, namely C. Let

define $E(t)$ as the residual capacity of the storage unit at time t. We consider a uni-

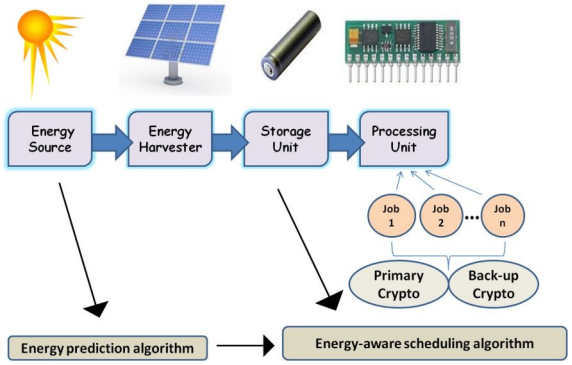


Figure 4.12: Framework of the energy harvesting system

processor sensor node that consumes negligible energy in idle state. Transactions require to be performed periodically. We assume that two encryption algorithms are available. The first one is characterized by a very low execution time and energy requirement, called by the alternate in energy starvation situations. The second one has a greater execution time and greater energy requirement but provides the highest quality of security. Consequently, every encryption task τ_i is modeled by a five-tuple $(Cp_i, Ca_i, Ep_i, Ea_i, T_i)$ where it is denoted:

- Cp_i respectively Ca_i: the Worst Case Execution Time (WCET) of the primary respectively the alternate of τ_i with $Cp_i > Ca_i$.
- Ep_i respectively Ea_i the Worst Case Energy Consumption (WCEC) of the primary respectively the alternate of τ_i with $Ep_i > Ea_i$.
- T_i the period of τ_i.

We assume that E_i is not necessarily proportional to C_i. Moreover the average incoming power is higher than or equal to the average power consumed by the alternates. And the processor utilization rate due to the alternates is less than one.

4.3.2.2 The scheduling issue

The introduction of energy harvesting capabilities in sensor networks has introduced additional design questions. How to intelligently use the ambient incoming energy to optimize the QoS of the system measured in terms of security? Furthermore, how to adapt the processing activity so as to subsist perennially on a given energy source?
We need a scheduling algorithm which:

1. Guarantees either the primary or alternate version of each encryption task to be completed in a time.
2. Attempts to complete as many primaries as possible.

Our basic strategy uses Earliest Deadline as late as possible to pre-allocate time intervals to the alternates. Even if EDF is not the optimal scheduler in energy harvesting systems, we proved recently that EDF is the optimal non idling scheduler [26]. In contrast to an

optimal scheduler, EDF is no clairvoyant and consequently easy to implement. Here, we want to delay execution of alternates as much as possible to save both time and energy. At run-time, it attempts to execute primaries first. An alternate will be executed only (1) if its primary fails due to lack of time, lack of energy or manifestation of bugs, or (2) when the latest time to start execution of the alternate without missing the corresponding task deadline is reached. This algorithm has been shown to be effective and easy to implement.

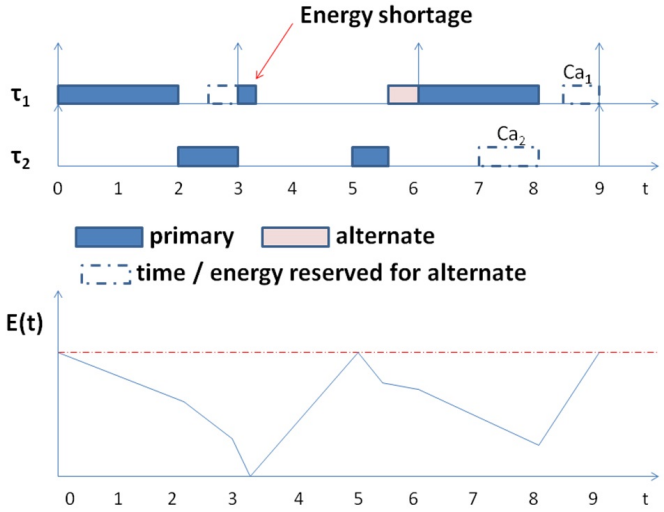


Figure 4.13: The Deadline Mechanism

Figure.4.13, depicts a very simple example where two periodic tasks of respective periods 3 and 9 have to execute before deadline. The Last chance strategy attempts to execute the primaries first. From time 0 up to time 3, primaries execute timely because of sufficient energy available in the storage unit. When primary τ_1 succeeds at 2, the interval reserved for the alternate is removed and additional time could be used for primaries. Here, we do not illustrate the process for reserving energy that should guarantee feasibility of all alternates. Second primary τ_1 fails before completion due to depletion of the storage unit. The primary is aborted and the storage starts recharging since we let the processor inactive until the storage fully replenishes. That permits primary τ_2 and alternate τ_1 to complete successively as third primary τ_1.

4.4 Conclusion

In the first part of this chapter, we studied and analyzed one of the best chaos-based cryptosystems of the literature, namely Zhang cryptosystems. We have partially cryptanalyzed the first Zhang cryptosystem by using a combination of a chosen plaintext attack, and a brute force attack, for ($n = 1,\ m = 2$). Also, after removing the diffusion effect using equation (4.12), we succeeded in decreasing the security level, measured by means of parameters NPCR and UACI, by applying the differential attacks for ($n = 2,\ m = 2$). Then, in the second part of the chapter, based on a similar structure of the Zhang cryptosystem, we designed two versions, with two different implementation of the second version, of a chaos-based cryptosystem. We have shown that all versions of our proposed cryptosystem are faster and more secure than Zhang and many chaos-based cryptosystems. The

time performance is carried out using average encryption/decryption times, encryption throughput, and the number of cycles needed to encrypt or decrypt one byte. The last measure is necessary to compare different cryptosystems working on different platforms. All versions of our proposed cryptosystem are implemented using the CBC mode and a robust chaotic generator to produce the dynamic keys for each new encryption round and new block. The high security level of all versions of the proposed cryptosystem is verified by testing them for different kinds of known attacks, and by using the well-known statistical analysis. Finally, all results prove the superiority of the proposed cryptosystems for use in secure and real-time applications.

Joint Crypto-Compression

5

Video coding and crypto-compression stat of the art

A general overview of the video compression is introduced in the first section, with some details on the last two video coding standards in sections 5.2 and 5.3. Related works on video encryption algorithms are presented in Section 5.4. Finally, section 5.5 concludes this chapter.

5.1 Video compression steps

The video consists of successive pictures captured at a specific frequency called frame rate. For instance, a color video in High Definition (HD) resolution captured at 24 pictures per second (1920x1080p24) requires a bit-rate of around $\approx$ 600 Mbps in case of 4:2:0 color format. Therefore, it is difficult to deliver video in real-time over Internet and wireless networks without compression.
Video compression consists in removing spatial and temporal redundancies in the video and thus, it considerably decreases the required data to represent the video. The general block diagram of video compression processes is shown in Figure 5.1. The video compression processes can be classified into lossy and loss-less compression. The consequence of the lossy compression is that it is not possible to recover, at the decoder side, the original video. During the compression process, the loss of information is mainly due to the quantization step. It consists in reducing the dynamics of the encoded coefficients, and then the reconstructed video at the decoder is not identical to the original one.

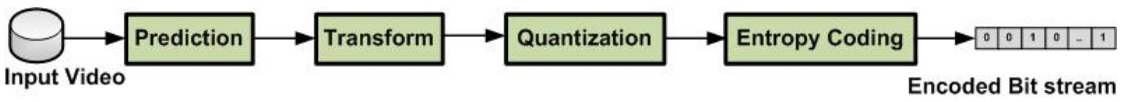


Figure 5.1: General block diagram of the video compression at the encoder

In loss-less compression, the non-invertible steps are not involved (i.e., all steps during the compression process are reversible ones). Therefore, a loss-less compression process

enables to recover at the decoder side the original video, without any loss of information. In terms of compression ratio (which is defined as uncompressed over compressed video size), the lossy compression enables higher compression ratio than the loss-less one. For instance, typical values for compression ratio in the HEVC ranges from 300 to 1000 while in loss-less compression scheme the compression ratio at max can be 4 [139].
In the following, we give a brief description of the video compression steps.

5.1.1 Prediction

The video consists of a set of pictures, with high spatial and temporal redundancies. Predictions in video are used to remove spatial (called **Intra prediction**) and temporal (called **Inter prediction**) dependencies present in the video. Intra prediction is carried out between blocks of pixels within the same picture, while Inter prediction is carried out between the current block and the best match of the predicted or reconstructed block that can be in the previous or the future decoded frames. The difference between the original and the predicted blocks (i.e., prediction error) is forwarded to the transform process.

5.1.2 Transform

The transform allows to de-correlate the pixels of an image and to compact the energy in a restricted number of coefficients. The choice of the transform method is an essential step during the design process of a new compression encoder. The properties of the chosen transform characterizes the performances and the characteristic of the encoder. Several reversible transform methods are used in the field of image and video compression, The widespread ones are: Karhunen Lovev (KL), Discrete Cosine Transform (DCT), Integer Wavelet Transform (IWT), Hadamard Transform (HT), also known as the Walsh–Hadamard transform and Discrete Wavelet Transform (DWT), etc [60, 77, 99].

5.1.3 Quantization

The quantization step assigns values, taken from a finite countable set, for all of the transform coefficients. Unlike the previous steps, the quantization step is irreversible and introduces distortion, so this step is only applied to a lossy coding. There are several types of quantification, such as uniform and non-uniform, Scalar Quantization (SQ) and Vector Quantization (VQ).

5.1.4 Entropy coding

The entropy encoding operates on the statistics of the quantized coefficients to remove the statistical redundancy between them. Moreover, an entropy coder uses the Variable Length Code (VLC) for code-words, such that the most occurrences coefficients are represented by the short binary codes, and the lowest occurrences coefficients are represented by the long binary codes. Consider a discrete random (unpredictable) variable X with realization x_i where $i = 1,\ 2,\ \ldots\ n$ which correspond to the coefficients of the transformed image, and $P(x_i)$ is the probability of appearance of the coefficient x_i. The entropy of the transformed image is defined by the expected value of the quantity of information

associated with each coefficient, known by the theoretical limit of the Shannon entropy:

$$H(X) = -\sum_{i=1}^{n} P(x_i) log_2 \left(P(x_i)\right) \tag{5.1}$$

There are several entropy coder who allows to approach at best the theoretical limit of the Shannon entropy such as: Huffman Coding (HC), Arithmetic Coding (AC), Lempel-Ziv coding.

5.2 H.264/Advance Video Coding (AVC) and scalable extension

The AVC has been introduced in 2003 as a joint project of the International Standards Organization (ISO) and the International Telecommunications Union (ITU). The AVC is also known as H.264 and MPEG-4 Part 10 [154]. In this thesis, for simplicity, MPEG-4/H.264 is referred as AVC. This standard is developed to achieve low bit rate visual

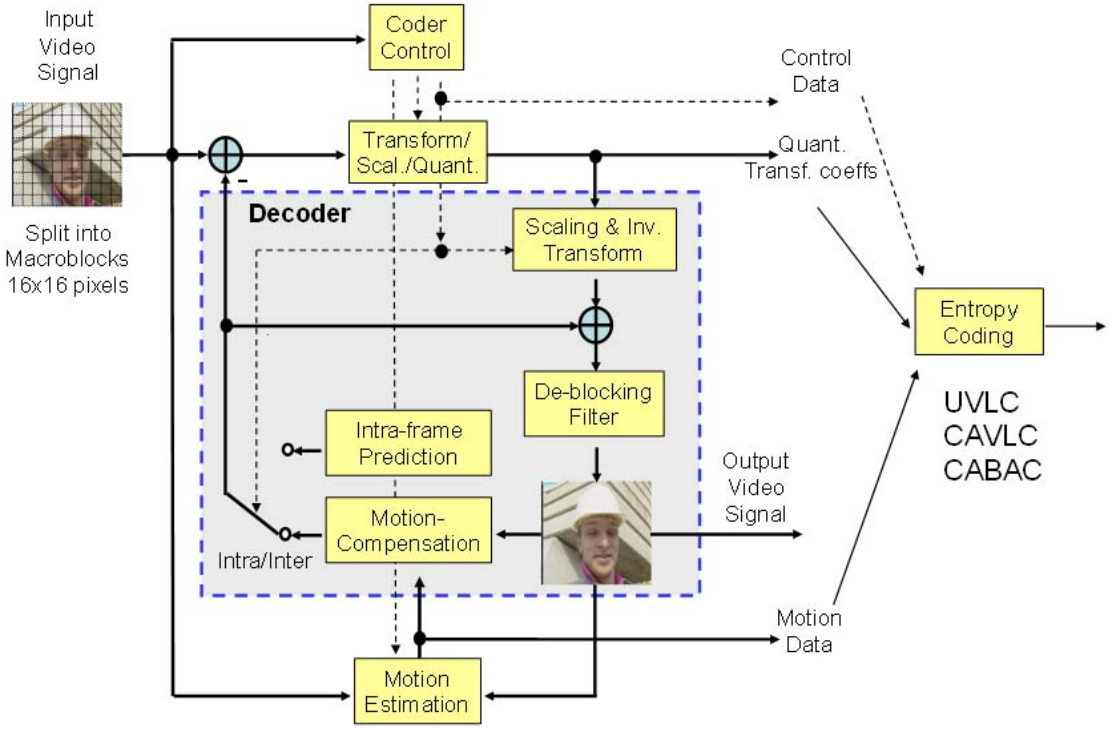


Figure 5.2: Typical AVC video encoder (from [127])

communication, with better video compression efficiency than previous ITU-T and ISO standards. The basic encoder structure of this standard is described in Figure 5.2. For further details on the AVC standard, the reader can refer to [153].
On the one hand, the AVC encoder compresses a video to decrease the required capacity for transmission or even for storage, while the decoder decompresses the AVC bit-stream to reconstruct an estimation of the original video. The AVC encoder and decoder structure is similar to the previous video coding structures (i.e., H.263, H.262...). At the encoder side the compression steps are the following: [20, 103]:

1. **Prediction:** each frame is partitioned into a number of processing units named Macro Blocks (MBs) of size 16×16 pixels. The prediction in the AVC for each MB is carried out through:

   - Intra prediction: the current MB can be encoded using coded MB within the current frame (i.e., the both MBs are in the same frame). Intra prediction uses (4×4 or 16×16) block sizes to perform the prediction.
   - Inter prediction: the current MB can be encoded using similar previous coded MBs outside the current frame. Note that previous coded MBs can be before or even after the current frame in display order. Inter prediction uses (4×4, 4×8, 8×4, 8×8, 8×16, 16×8 or 16×16) block size to perform this prediction. More details of Intra and Inter predictions are described in section 5.3.

2. **Transform**: the error prediction block (i.e., residual block) is transformed using integer DCT transform to produce the transform coefficients. The residual block size can be (4×4, 8×8 or 16×16) values. One of the major enhancement of the AVC is that the primary block size can be 4×4 values. Finally, the result of the transform step is forwarded to the next step to rescale them down.

3. **Quantization:** this step rescale (reduce) the value of coefficients based on a Quantization Parameter (QP). All coefficients of the MB (16 coefficients in case of 4×4, 64 in the case of 8×8 and 256 in case of 16×16 MB size) are mapped to the corresponding level based on the QP value and then, they are rounded to the nearest integer. The QP can take 52 possible values, raning from $\{0, 1, 2, ...51\}$.

4. **Encoding :** In the AVC two entropy coding methods are used: Context-Adaptive Variable-Length Coding (CAVLC) and Context-Adaptive Binary Arithmetic Coding (CABAC). In this step, syntax elements (i.e., all values that should be encoded including: coefficients after quantization, and all necessary information for the decoding process...) are converted into a binary representation form during the encoding process.

The Scalable Video Coding (SVC) is the scalable extension of the AVC standard. SVC enables temporal, spatial and quality scalability [110]. Moreover, this extension introduces significant improvements regarding to the coding efficiency with respect to the AVC.

5.3 High Efficiency Video Coding (HEVC) standard

The first version of the HEVC standard has been finalized in January 2013 by the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG) under a partnership known as a Joint Collaborative Team on Video Coding (JCT-VC).
The HEVC has the basic structure (see Figure 5.3) of the previous video coding standards with new coding tools that enable up to 65% bit-rate reduction compared to the AVC standard: [100, 48, 62, 92, 138]. These new tools are listed bellow:

- More flexible and efficient block partitioning structure.

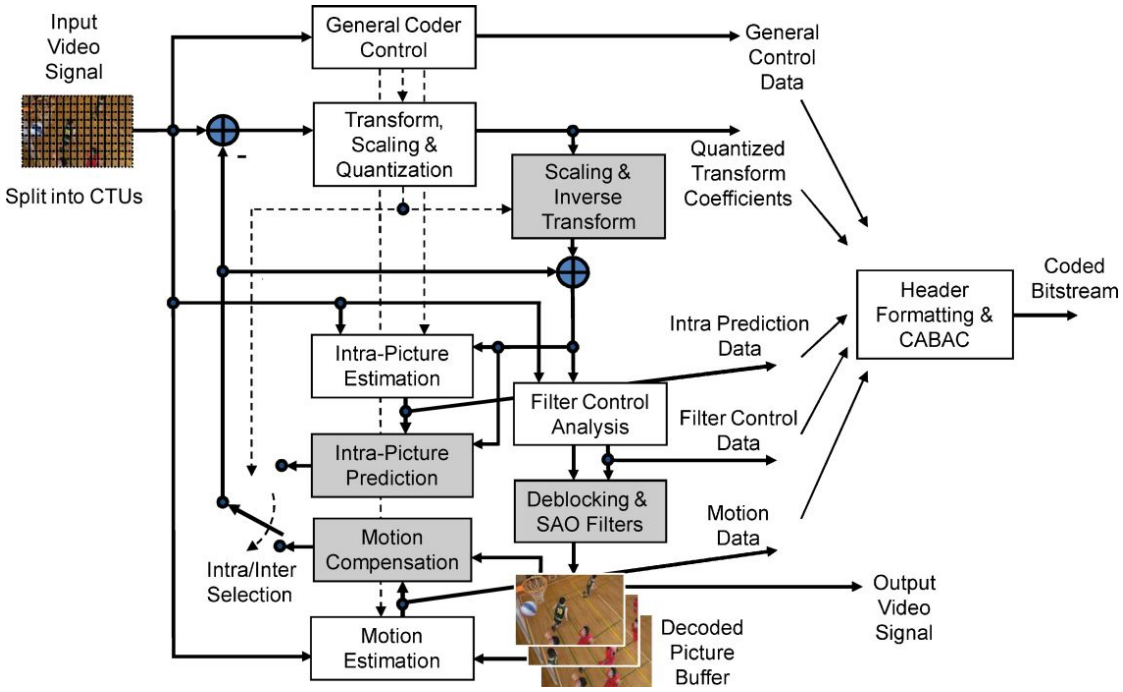


Figure 5.3: Typical HEVC video encoder (from [138])

- More angular Intra prediction modes.
- More flexibility in the transform block sizes.
- More acuurate interpolation filters.
- Better prediction and signaling of modes and Motion Vectors (MVs).
- New Sample Adaptive Offset (SAO) filter.

As a result of these new technologies of the HEVC, it becomes more suitable for parallel processing and for different kinds of applications.

5.3.1 HEVC partitioning

Figure (5.4) illustrates picture partitioning process in the HEVC standard. The partitioning steps are detailed in the following:

- Each video picture is partitioned into N_CTU Coding Tree Units (CTU), and the CTU size can be $M \times M$ where $M = 16$, 32 or 64.

$$N\_CTU = \frac{Frame\ Size}{CTU\ Size} \tag{5.2}$$

- Each CTU has luma block, called Coding Tree Block (CTB) and also two CTB chroma blocks. The size of luma CTB is identical to the CTU size, whereas each chroma CTB size depends on the input video representation, and in case of 4:2:0 color format, it is equal to $M/2 \times M/2$.

- Each CTB is further divided into a number of Coding Blocks (CBs), where the size of each CB can be 8×8, 16×16, 32×32 or 64×64. Then a Coding Unit (CU) is formed using three CBs (luma CB, and two chroma CB) and their associated syntax elements. The CUs are reading in Z-scan order as illustrated in Figure 5.5.

- Each Inter prediction CB can be one Prediction Block (PB) of $M \times M$ size, or 2 PBs of sizes $M/2 \times M$ or $M \times M/2$, or four PBs with size $M/2 \times M/2$ and other PB sizes as given in Figure 5.4.
  The Intra predictions CB is formed by one PB of size $M \times M$ or four PBs of size $M/2 \times M/2$.

- Finally, the CB has another partitioning structure relative to the transformation process, each CB can be one TB of size $M \times M$, or four TBs of size $M/2 \times M/2$ each, or sixteen TB of size $M/4 \times M/4$ each.

5.3.2 HEVC Intra prediction

The intra prediction modes in the HEVC are increased from eight in the AVC up to 35 modes. These modes are classified into: 33 Intra_Angular prediction modes (from $2 \rightarrow 34$), and the Intra_Planer mode (mode 0) and the Intra_DC prediction mode (mode 1) [64]. Note that with the same PB only one intra prediction mode can be used.

The Most Probable Mode (MPM) concept, in all video coding standards is usually used to decrease the required bits during the encoding process and then, to improve the coding efficiency. It is derived from the Intra-prediction modes of the neighboring blocks. During the intra prediction in the AVC, only one MPM is defined, while in the HEVC, three MPM are defined and so, the remaining Intra modes in the HEVC are 32, which means 5 bits are required to binarize the selected mode. This binarization is carried out using Fixed Length Coding (FLC).
Three syntax elements ($prev\_intra\_luma\_pred\_flag$, mpm_idx, $rem\_intra\_luma\_pred\_mode$) are used to specify the Intra prediction mode for PB luma. The $prev\_intra\_luma\_pred\_flag$ is a syntax element used to specify if the luma intra mode is one of the MPM or not. In the case that $prev\_intra\_luma\_pred\_flag$ is one, then the mpm_idx parameter is parsed to indicate which mode from the MPM list is used for the current luma intra prediction mode. In case $prev\_intra\_luma\_pred\_flag$ is zero, then the $rem\_intra\_luma\_pred\_mode$ is encoded using the FLC.
Figure 5.6 shows the derivation process of the MPM. Unlike the AVC, in the HEVC, the chroma intra prediction mode is derived from the luma one. The derivation process of the chroma intra prediction mode is described in Table 5.1, it works based on if the derived chroma intra mode is identical to the initial chroma intra mode, then the Intra angular mode (i.e., mode-34) is used for the chroma, otherwise, the derived one is used.

An adaptive scanning method is applied in the HEVC, and it is used with the block sizes of 4×4 and 8×8 to benefit from the statistical distribution of the active coefficients in 2-D transform blocks. The scan is selected according to Table 5.2.

5.3.3 HEVC Inter prediction

The inter prediction in the HEVC is carried out by:

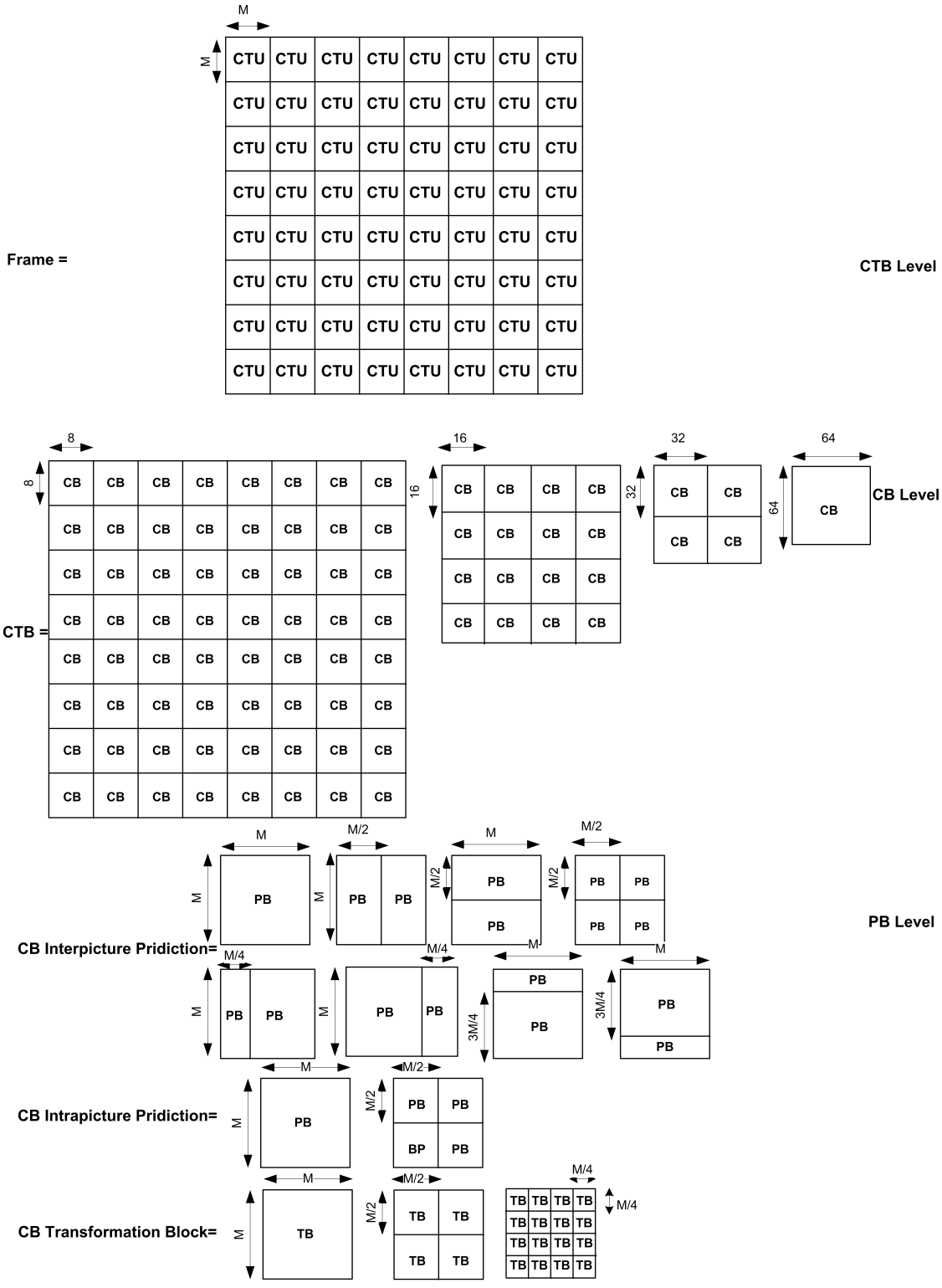


Figure 5.4: Typical HEVC partitioning process

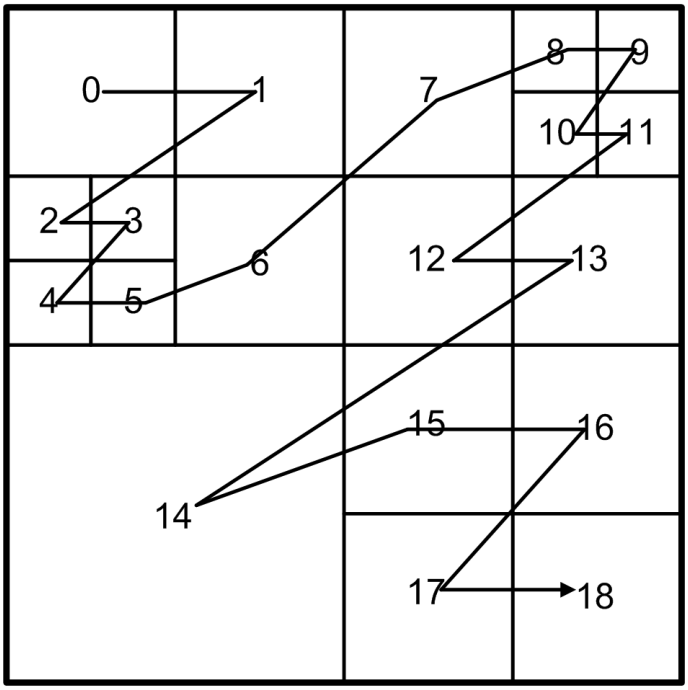


Figure 5.5: Z-scan order of CUs inside the CTU (from [124])

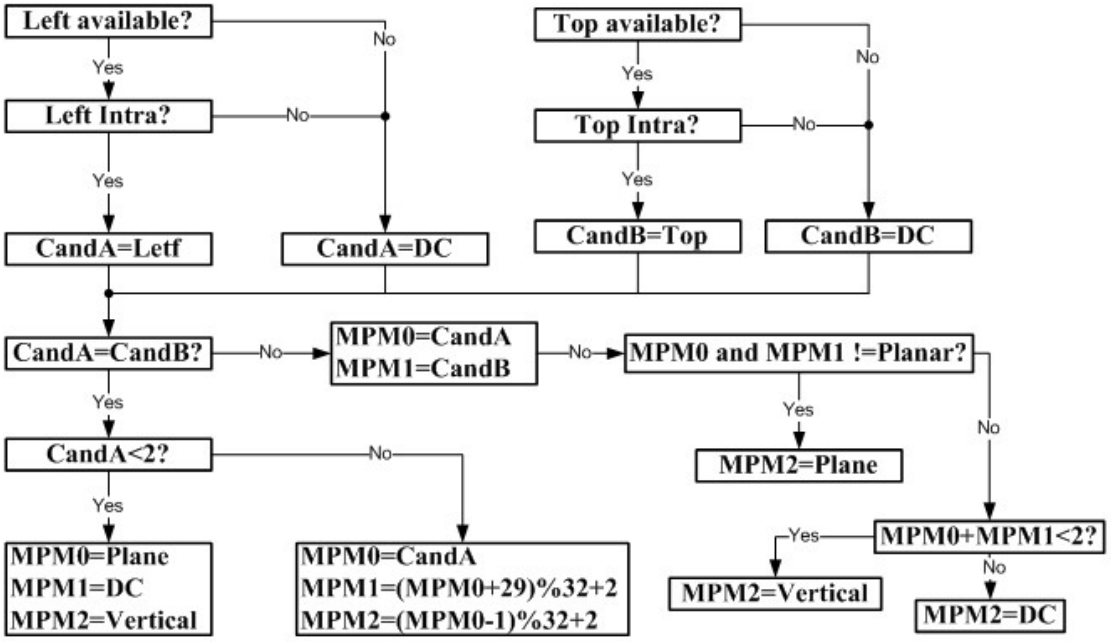


Figure 5.6: Derivation of the MPM Modes(from [124])

Table 5.1: Derivation process of the chroma intra prediction mode (from [124])

Intra_chroma_pred_mode	Luma Intra Pred Mode			
	0	26	10	1
0 (i.e., mode-0)	34	0	0	0
1 (i.e., mode-26)	26	34	26	26
2 (i.e., mode-10)	10	10	34	10
3 (i.e., mode-1)	1	1	1	34
4 (i.e., use the luma mode)	0	26	10	1

Table 5.2: Mapping between Intra prediction mode and coefficient scanning order

Intra prediction mode	Coefficient scanning for 4×4 and 8×8	Coefficient scanning for 16×16 and 32×32
Angular (6-14)	Vertical	Diagonal
Angular (22-30)	Horizontal	Diagonal
All other modes	Diagonal	Diagonal

- Motion Estimation (ME): is an inter prediction process to find the best match between the current processing unit(such as PB_1) and a region in a previous or a future frames (such as PB_2). The output of this process are: MV information, skip mode information or merge mode information.
- Motion Compensation (MC): is the process of generating the identical interpicture prediction signals using the output of the previous step (i.e., MV and the mode information) calculating the difference between PB_1 and PB_2.

The MV is a vector information indicating the moving direction of the PB by calculating the offset between the current PB and the reference PB. The Inter-prediction of each PB is predicted using a bi-prediction or a uni-prediction. In uni-prediction the smallest PB luma size is 4×8 or 8×4 whereas for bi-directional the smallest PB luma size is 8×8.
As this part of the HEVC compression process is used in our next contribution, the used parameters in inter prediction step are described in details. In Inter-prediction, the main modes are Merge and Skip, for which the MV and reference index information are derived from spatially or temporally neighboring blocks. In merge mode, the current block and the neighbor block(s) have a shared region that contains the required motion information, and so, the coding efficiency is improved since no need to transmit the motion information for each block. The equivalent structure of this mode in the AVC is the skip and direct modes. Anyway, there are two main differences between the HEVC and the AVC for this mode: the index information is transmitted in order to identify one of available independent candidates. Moreover, it identifies the references of the available picture in the list picture index, while in the AVC direct mode, part of predefined values already exist.

In Figure 5.7, the current PB is D, and so, the same MV information of B,C or D_{t-1} prediction blocks are available. Assumes that the merged block with D is X where $X \in \{B, C, D_{t-1}\}$, in this case only one MV information is required for the both D and X blocks, and there is a flag used to determine that the D prediction block is merged with block X [138].
The HEVC is divided into frames and slices, the frame can be I, P or B frame. The prediction in I frames is intra prediction, while in P frames the intra and inter predictions are used. The inter prediction in P frames uses only one MV per PB, in B frames inter and intra predictions are used and the inter prediction uses one or two MVs for each PB. The Motion Vector Differences (MVD) is a difference between the Motion Vector Predictor (MVP) and the actual motion vector used to predict a current block [164].

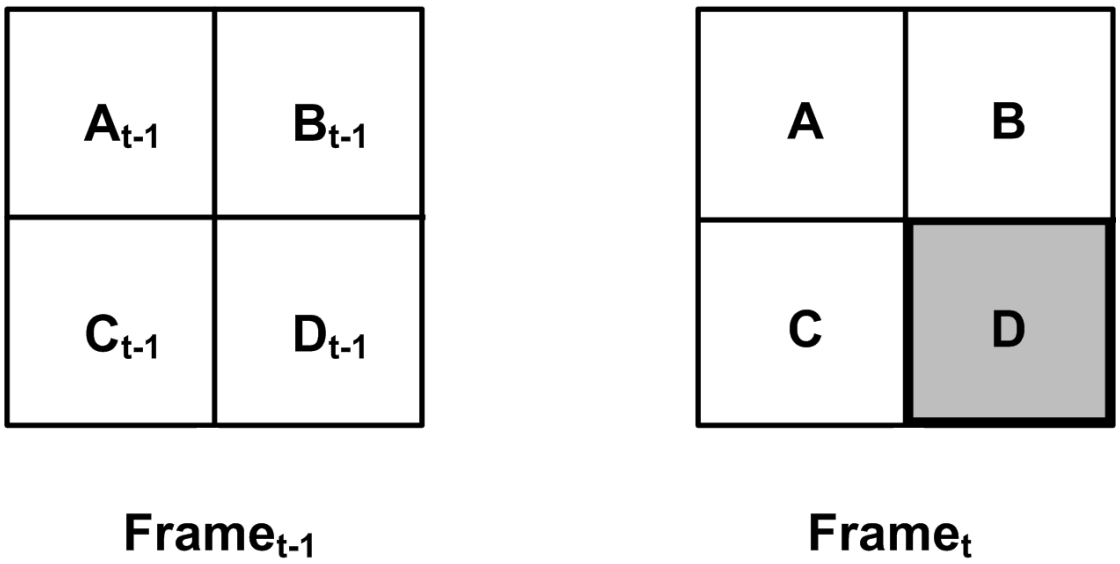


Figure 5.7: Merge mode example

The sign of the MVD is indicated by mvd_sign_flag, when this flag is one the MVD has a negative value, whereas the MVD has a positive value when this flag is zero. This flag sets to zero in case of it is not signaled in the bits-stream. In P frame, there is only one MVD in the both horizontal and vertical directions. The parameter abs_mvd_minus2 specifies the absolute value of an MVD minus 2. This means, if the absolute value of the MVD is zero or one, the previous parameter is not signaled. Hence, two parameters are used, $abs\_mvd\_greater0\_flag$ to specify if the absolute value of an MVD is greater than 0 or not. $abs\_mvd\_greater1\_flag$ to specify if the absolute value of an MVD is greater than 1 or not. In case the $abs\_mvd\_greater1\_flag$ is not present, it is set to 0. In case of B frame, we have two MVD and for each one, we have the following flags: mvd_sign_flag, $abs\_mvd\_greater0\_flag$ and $abs\_mvd\_greater1\_flag$. Finally, in Inter prediction there are $List0$ and $List1$. The former uses the parameter called ref_idx_l0 to indicate the reference picture index of the current prediction unit. The later uses ref_idx_l1 for the same function. In case ref_idx_l0 or ref_idx_l1 are not signaled, they are considered equal to 0.

5.3.4 HEVC transformation and quantization

According to Figure 5.3, the predicted residual values are grouped in TB to perform the transformation process using integer transform blocks of four sizes (4×4, 8×8, 16×16 and 32×32) the last three sizes are derived from a DCT, where the derivation process of the 4×4 luma intra-prediction is based on the Discrete Sine Transform (DST). The used integer transforms in the HEVC have a better approximations than the used ones in the AVC.
The quantization process in the HEVC uses the same scheme as the one used in the AVC: Uniform Reconstruction Quantization (URQ) scheme. This scheme is controlled by the Quantization Parameter (QP), the QP range is defined as $QP \in \{0 \rightarrow 51\}$.
The output of the quantization process is forward to be encoded using Entropy Coding (EC) step.

5.3.5 CABAC entropy coding

In the HEVC, comparing to the AVC, only CABAC entropy coding is used and the main contribution of our proposed work of the crypto-compression systems are based on the CABAC steps, that we describe them in details. In general, CABAC process consists of three steps as illustrated in Figure 5.8:

- Binarization is used to convert the Quantized Transform Coefficients (QTCs), into bin strings using one of five basic binarization methods: Unary code, Truncated Unary (TU) code, Fixed Length Code (FLC), Kth order Exp-Golomb (EGK) code and Truncated Rice code with context p (TRp).
- Context modeling is used to perform two functions: context model selection and context model access. It uses the statistics of the coding symbols to update their probabilities.
- Binary Arithmetic Coding (BAC) is applied on each bin according to the selected probability model[138], and it consists of:
    1. Context coding: it provides an estimation of the conditional probabilities of the coding symbols.
    2. Bypass coding: this mode permits equal probability (0.5 for each bin).

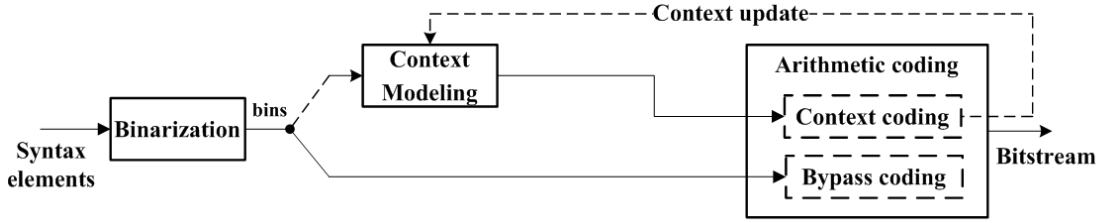


Figure 5.8: Three main functions in the CABAC

The binarization process
The binarization process in the CABAC consists of five basic methods, that we describe bellow:

1. Unary Code (U): The non-binary value Y is coded into binary form by formulating it with Y ones and ending by zero. For example, the binary code of 4 is 11110, while for 7 is 11111110.

2. Truncated Unary Code (TU): It is almost identical to the previous one, it is defined for any Y value such that $0 \leq Y \leq S$ (where $S = cMax$ in our work). For all $Y < S$ the TU code is exactly same as the Unary code, while for $Y = cMax$, the ending bit (i.e., the 0) is removed.

3. Fixed Length Code (FLC): the inputs of this process are a syntax element and the cMax parameter. The output is the FL binarization code based on arithmetic binary representation(see equation 5.4). The length of the code-word, called $fixedLength$, is derived from the $cMax$ by using equation (5.3).

$$fixedLength = \lceil \log_2(cMax + 1) \rceil \tag{5.3}$$

and

$$FLC(Y) = arithmetic\, binary(Y) \tag{5.4}$$

Example 5.1: Assume the syntax element is $Y = 3$, and $cMax = 4$, then:
$fixedLength = \lceil \log_2(4 + 1) \rceil = 3$, which means three bits is required to represent the FLC binary value and so
FLC(3)=011.

4. Truncated Rice Code with context p (TRp): this method of binarization is introduced for the first time in the HEVC. It consists of prefix and suffix terms. The code of the prefix term is given using equation (5.5) if the condition of the equation (5.6) is valid, otherwise, the prefix bin strings are $\lfloor \frac{cMax}{p} \rfloor$ ones. Where $cMax$ is an input, p is given by equation (5.8) and the $cRiceParam$ parameter is determined according to the algorithm 10 and its value is ranged $\{0,\ 1,\ 2,\ 3,\ 4\}$.

$$TRp(Y)\_Prefix = Unary\ code(\lfloor \frac{Y}{p} \rfloor). \tag{5.5}$$

$$\lfloor \frac{Y}{p} \rfloor < \lfloor \frac{cMax}{p} \rfloor. \tag{5.6}$$

$$p = 2^{cRiceParam} \tag{5.7}$$

The suffix code (if presents) is given using the equation (5.8). The length of this code is calculated by the equation (5.3), in which $cMax$ is replaced by $cMax_S$ defined by the equation (5.9).

$$TRp(Y)\_Suffix = FLC(Y\ Mod\ p). \tag{5.8}$$

$$cMax_S\ = p - 1 \tag{5.9}$$

Finally, the code-word of the TRp is a concatenation of the prefix and the suffix codes.
Example 5.2: Assume $Y = 7$, $cMax = 8$ and $cRiceParam = 1$, then:
$p = 2$, $\lfloor \frac{7}{2} \rfloor < \lfloor \frac{8}{2} \rfloor$:
$TRp(7)\_Prefix = Unary\ binary(\lfloor \frac{7}{2} \rfloor) = 1110$.

For the suffix, $cMax_S = 2^1 - 1 = 1$, $fixedLength = \lceil \log_2(1+1) \rceil = 1$. which means one bit is required to represent the arithmetic binary for the suffix value and so,
$TRp(7)\_Suffix = FLC(7 \; Mod \; 2) = 1$.
The word-code is 11101.
Example 5.3: Assume $Y = 7$, $cMax = 7$ and $cRiceParam = 1$, then:
$p = 2$, $\lfloor \frac{7}{2} \rfloor \; is \; not \; less \; than \; \lfloor \frac{7}{2} \rfloor$:
Consequently, the TRp prefix is $\lfloor \frac{cMax}{p} \rfloor$ ones=111.
For the suffix, $cMax_S = 2^1 - 1 = 1$, $fixedLength = \lceil \log_2(1+1) \rceil = 1$. which means one bit is required to represent the arithmetic binary for the suffix value and so,
$TRp(7)\_Suffix = FLC(7 \; Mod \; 2) = 1$.
The word-code is 1111.
Example 5.4: Assume $Y = 8$, $cMax = 10$ and $cRiceParam = 1$, then:
$p = 2$, $\lfloor \frac{8}{2} \rfloor < \lfloor \frac{10}{2} \rfloor$:
$TRp(8)\_Prefix = Unary \; binary(\lfloor \frac{8}{2} \rfloor) = 11110$.
For the suffix, $cMax_S = 2^1 - 1 = 1$, $fixedLength = \lceil \log_2(1+1) \rceil = 1$. which means one bit is required to represent the arithmetic binary for the suffix value and so,
$TRp(8) \; Suffix = FLC(8 \; Mod \; 2) = 0$.
The word-code is 111100.

5. Kth Order Exp-Golomb Code (EGK): The EGk code is also a concatenation of prefix and suffix terms. The prefix term of the EGk code is the U representation of $l(Y)$ calculated based on equation (5.10). The suffix term is the arithmetic binary representation of the $(Y + 2^k(1 - 2^{l(Y)}))$ using $k + l(Y)$ significant bits.

$$\begin{aligned} &EGK(Y)\_Prefix = Unary \; code(l(Y)). \\ &l(Y) = \lfloor \log_2(\tfrac{Y}{2^k} + 1) \rfloor. \end{aligned} \tag{5.10}$$

Example 5.5: Assume $Y = 7$, and we want to binarize it using EG0 and EG1, then:
For $k = 0$
$l(7) = \lfloor \log_2(\frac{7}{2^0} + 1) \rfloor = 3$.
Prefix of $EGK(7)\_Prefix = 1110$.
The length of the suffix code is $0 + 3 = 3$, then
$EG0(7)\_Suffix = arithmetic \; binary(7 + 2^0 - 2^{0+3}) = 000$.
The code-word is 1110000 For $k = 1$.
$l(7) = \lfloor \log_2(\frac{7}{2^1} + 1) \rfloor = 2$.
Prefix of $EGK(7)\_Prefix = 110$.
The length of the suffix code is $1 + 2 = 3$, then
$EG1(7)\_Suffix = arithmetic \; binary(7 + 2^1 - 2^{1+2}) = 001$.
The code-word is 110001.

The following programming code is published in the last drift of the JCT standard (High Efficiency Video Coding (HEVC) text specification draft 10 [61]), and is used to simulate the previous mathematical equations.

Algorithm 9 [K-th order Exp-Golomb binarization process (from [61]])

```
Y=abs(Syntax Element)
stopLoop=0
do
  {
    if (Y >= (1 << K))
      {
        put(1)
        Y = Y − (1 << K)
        K++
      }
    else
      {
        put(0)
        while(K − −)
          put((Y >> k) & 1)
        stopLoop=1
      }
  }while(!stopLoop)
```

5.4 Video encryption algorithms - related works

Video encryption is a hot research topic in the last decades [5]. The security and confidentiality of video coentent are very important in commercial applications. In this section, related works of video encryption techniques are presented.

A traditional algorithm, such as AES and DES, were originally used to encrypt texts. Their usage for video encryption in most of the time is done in a naive way. Thus, this class of encryption methods is called Naive Encryption Algorithm (NEA) [12, 144]. In the NEA, the video bit-stream is encrypted based on a straight-forward method, encrypting each bit. In fact, the NEA manages the video bit-stream as text without using the special structure of the compressed video [70, 156]. However, NEA is not suitable for video encryption (a justification is given later).
To encrypt the video contents, there are three main possible approaches, see Figure 5.9:

- Encrypt the video at position 1, and then compress it.
- Compress the video and then encrypt it at position 9.
- Joint encryption-compression video between position 7 and 9 included (to have invertable encryption).

Confidentiality of multimedia contents should be considered with a particular attention from both point of views: compression and encryption requirements. Several encryption solutions are used to encrypt the video, either before or after the compression process. In general, separating encryption and compression processes is not a good solution for video. In deed, the encryption at position 1 (before compression) has the following obstacles:

1. First of all, the compression process is usually a lossy one based on a non-reversible operations; which led to incorrect decryption.

2. Secondly, encrypt the whole bit of the video is a time consuming process, whereas, the decoder should be fast to be used for real-time applications.

3. Thirdly, when the encryption process is achieved using a robust encryption algorithm, the ciphered data becomes a random or a pseudorandom data. That means the redundancy between data bit is removed and therefore the compression scheme becomes not efficient (i.e., the compression rate is significantly decreased).

The encryption at position 9 (after the compression process) consists of encrypting all coded bit stream. This scenario, that usually used has the following drawbacks:

1. Firstly, the coded bit-stream are random and then the encryption scheme becomes less efficient

2. Secondly, this solution is a time consuming since all bits must be encrypted and then the bit-rate is increased, especially for high quality and resolution video.

3. Thirdly, the encrypted video bit-stream desynchronizes the decoder since the encrypted coded bit-stream is not conforming with the video bit-stream syntax standards(a non format compliant encryption scheme).

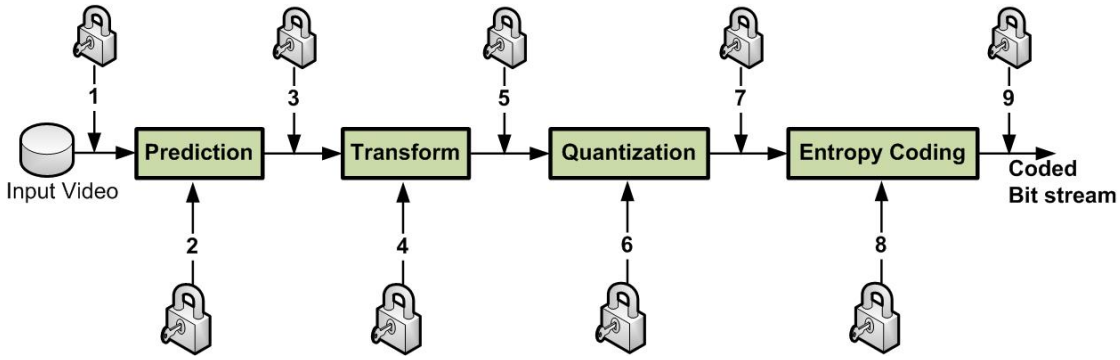


Figure 5.9: Different possible encryption positions in the video compression process

Joint Encryption compression

The joint encryption compression schemes are mainly classified into:

Selective Encryption (SE): encrypts some parts of the video content (sensitive content) to achieve the required security level with a minimum extra computation cost during the video compression process.

Perceptual Encryption (PE): encrypts some parts of the video content (sensitive content) to degrade the quality of the video according to the quality requirements, with a minimum extra computation cost. The other video content parts are left as plain video for non-authorized users. In fact, the difference between SE and PE is depends on the target and the application of each one, while the encryption method is almost the same, since the both are used to encrypt some parts of the video content.

In general, encryption algorithms for video can be used for several purposes based on the target applications.

5.4.1 MPEG Video encryption algorithms

In this section, we briefly present some encryption algorithms of the video standards before the AVC standard.

5.4.1.1 I-frames encryption

The security of MPEG-I video stream was proposed by [134], and the basic idea of this kind of algorithms is to encrypt the most important frames in the video which are I-frames, because P and B frames are useless without knowing the corresponding I-frames. However, Agi and Gong [5], have demonstrated that some of important parts of the encrypted video can be recovered independently of I-frames, as the encryption scheme does not consider the I-blocks in the P and B frames, and other important factors. Moreover, the compression ratio decreases significantly when the I-frame values are encrypted (i.e., the I-frames coefficients are encrypted before the entropy coding stage). Finally, encryption of I-frame values increases the bit rate significantly, while one of the most important requirements of the video compression is to decrease the bit rate. To preserve the same bit rate, the encrypted value and the original one should have the same length, and also the encrypted value should not affect the length for the other values. This requirement (constant bit rate) is more complex to achieve if the encryption process is performed outside or independent of the entropy coding unit.

5.4.1.2 A non-compatible four level of security

Based on the previous work of Agi, Meyer and Gadegast [86] have implemented four levels of video encryption algorithm. The first level is to apply the encryption process on all frame headers. The second level includes the first level task and it encrypts the DC and AC values of the I-blocks. The third level is to encrypt I-frames and I-blocks in P and B frames. The fourth level is to encrypt all data. The implemented video encryption algorithm uses DES standards to encrypt MPEG-1. One of the drawbacks of this algorithm is the compatibility problem with the standard encoder/decoder (i.e., it is not a format compliant algorithm). It needs a specific decoder to deal with the encrypted bit stream, since the encrypted bit stream crashes the decoder. Moreover, the compression ratio decreases significantly and the bit rate increases in this algorithm.

5.4.1.3 Zig-Zag permutation algorithms

More encryption algorithms in MPEG-1 are developed based on a permutation scanning method instead of the zig-zag one. The main steps of this algorithm are [142]:

- Formulating a one denominational vector of 64 value , where the first value is the DC and the remaining 63 values are the ACs and the last AC value is zero.
- The first value of the previous vector is the four Least Significant Bit of the DC, $LSB_4(DC)$, and the last value is the Most Significant Bit of the DC, $MSB_4(DC)$.
- A random permutation process is applied on the 64 values of the previous vector.

It is proved in [98] that this algorithm is susceptible to the known plaintext attack and the ciphertext only attack. Furthermore, it is not a format compliant, decreases the compression ratio and increases the bit rate.

5.4.1.4 Change the sign bits or the values of DCT coefficients

Tosun and Feng in [145], have designed three levels of security for the MPEG-2 video stream. The designed algorithm is based on the 64 coefficients of the DCT. A joint en-

cryption and compression algorithm is proposed. It divides the 64 values into three intervals: the first range is $[0, ..., x_1]$, the second range is $(x_1, ..., x_2]$, and the last range is $(x_2, ..., 63]$. Based on the required security level, we encrypt the first range or the first two ranges or all ranges. Since the proposed scheme encrypts coefficients before the entropy coding, it decreases the compression ratio, increases the execution time and also the the bit rate.

Further work on the frequency domain is carried out by Zeng and Lei [163]. They have proposed a selective joint encryption and compression scheme (SEA) for H.263. The proposed scheme flipping the sign of the DCT coefficients. The encryption of the sign for any value does not affect the compression ratio and the bit rate. Moreover, it is a format compliant encryption. However, it is not sufficient to achieve a good security level. The proposed SEA shuffles some blocks of the DCT coefficients and/or Motion Vectors (MV). This operation decreases the compression ratio and then increases the bit rate. Finally, in case of performing blocking shuffling between non-zero and zero DCT coefficients, the output bit-stream will not be format compliant.

In [15], Bharat et. al, have proposed a SEA for MPEG-1 bit-stream. The proposed SEA uses the secret key bits to change the sign bits of the DCT coefficients and the sign bits of the MV. This encryption method should be fast, but the authors stated, that the coding complexity is increased by about 2.55%. Moreover, the encryption process of the sign bit of the DCT and the MV does not enable a high security level.

Lian et. al., [72], have constructed a chaotic stream cipher to protect video content. This work is directed to the MPEG-2 video codec. The scheme is similar to the previous ones, encrypts the DCT coefficients and the MV sings of the MPEG-2 videos. In fact, this kind of encryption decreases the compression ratio and corrupts the standard decoder.

5.4.2 AVC and SVC encryption algorithms

Most of the proposed schemes are developed to provide a high security level for the video as a target. In perceptual/transparent encryption, the coded part without encryption can be decoded for any user to get a low quality preview of the video content. While the coded and encrypted part can not be decrypted for none authorized user.

5.4.2.1 Transparent encryption techniqes for AVC and SVC

Magli et al.,[78] have proposed a perceptual (transparent) encryption techniques for the AVC and the SVC video. The proposed encryption algorithms were classified into three joint encryption and compression categories:

- First category: Encoding with drift control to decrease the video quality. This can be achieved by encrypting the DCT coefficients of I Macro Blocks (MB) in I and P frames. The Least Significant Bit (LSB) of the chosen DCT coefficients is removed by right-shift, then all removed bits are collected and compressed using the arithmetic coding revisited [87]. At the end, the compressed file is encrypted using stream cipher in feedback mode before sending to the decoder. One of the major drawback of this scheme is the removing LSB bit, since this affects the compression ratio, and the side information (the collected, encrypted and compressed LSB

bits) must be sent to the decoder, which is not preferable in video compression and transmission.

- Second category: is used to degrade the video quality, and two quality level of I frames are created, one is the correct encoded and not encrypted I frames, while the other is the incorrect (encrypted and encoded) I frames. The clear drawback of this version is the redundancy, and also it needs a specified decoder since it corrupts the standard decoder.

- Third category: Encryption of the enhancement layers (SVC) using AES algorithm. The base layer is not encrypted, thus it can be used as a preview. To decode the enhancement layers the user should have the secret key. Since the AES is used to encrypt the enhancement layer at position 9 of the figure 5.9, as the author wrote, the encrypted enhancement layer is not format compliance.

5.4.2.2 Digital video scrambling method using Intra prediction mode

In [7], Ahn et. al., have proposed a simple and fast scrambling method for Intra prediction mode of the AVC video. The basic idea of this scheme is to change the 3-bits of the prediction mode by x-oring them with a 3 bits of a random sequence, in case of the MB size is 4×4.
Indeed, the intra prediction modes are nine modes and so four bits are required to encode the intra prediction modes. However, only eight modes are encoded using the $prev\_intra4x4\_pred\_mode$ parameter, which means one mode of the nine modes is excluded and it is signaled by MPM parameter.
In case of Intra 16×16 MB size, only four modes are available. The Intra prediction mode is encoded using VLC, and is jointly coded with luma and chroma coded block pattern values. The encryption process should be performed with ensuring that the Coded Block Patterns (CBP) does not change. One bit from the random sequence is used to encrypt the prediction mode based on: if this bit is 1 the mode is changed to the value which preserves the CBP, otherwise the mode is not changed (for more details on the CBP refer to [7]).
The process of changing only the I-frames as discussed and proved does not guarantee a high security level. Moreover, the process of changing the prediction mode decreases the quality of the prediction process, but it does not cause a completely incorrect prediction. Finally, this method can be a format compliant encryption scheme for the AVC, but not for the HEVC decoder, because of the scanning direction.

5.4.2.3 Entropy coding encryption in AVC

AVC uses CAVLC and CABAC to accomplish the entropy coding step. Lie et. al.,[65], have introduced a selective encryption scheme to encrypt the CABAC bit stream using a chaotic stream cipher based on the discrete Piece-Wise Linear Chaotic Map (PWLCM). In this proposed selective encryption scheme, each binarization process has a specific encryption and decryption operation. It is a format compliant, but it affects the compression ratio and the bit rate since it encrypts the Unary Code (UC), Truncated Unary code (TU), and Fixed Length Code (FLC). Finally, not all parameters during the binarization process can be encrypted while preserving the format compliance property.

5.4.2.4 Selective video encryption based on AVC

Lian et. al., [71], have proposed an encryption scheme for the AVC. This scheme has four parts:

1. Partial Encryption of Intra-prediction Mode: it encrypts the suffix term of the EGK code of the intra-prediction mode. The suffix term is encrypted using a stream cipher algorithm [85].

2. Partial Encryption of Coefficients: the DCs are encoded with VLC and then encrypted using the same stream cipher algorithm proposed in [85] based on ciphertext feedback mode. While the sign of ACs is firstly encrypted and then encoded by VLC.

3. Partial Encryption of Motion Vectors: encryption the sign of the motion vector differences is fast and format compliant without effecting the compression ratio.

4. Key Generator:all previous encryption methods in this scheme are controlled by different sub-keys. The needed sub-keys are generated using key generator.

The proposed scheme has a good performance in terms of the encryption overhead which is less than 1.1%. Moreover, it introduces a new idea to encrypt the suffix of some values to preserve the format compliant and constant compression ratio.

5.4.2.5 Fast protection of the AVC by selective encryption

To preserve the format-compliant and for a minimal impact on the compression performance, only the encoded suffix bits in bypass mode during the CABAC entropy encoding process can be encrypted [135].
Shahid et. al., [118], proposed a fast protection for the AVC using selective encryption of CABAC bin strings for I and P frames. The encryption process is performed as follows:

$$y = (x + \gamma)\ mod\ log_2(x + 1) \tag{5.11}$$

where γ is given by:

$$\gamma = rand()\ mod\ log_2(x + 1) \tag{5.12}$$

The x parameter is the suffix of the exponential-Golomb code, and y is the encrypted value of x.
Later, Shahid et. al., [115] enhanced this encryption scheme to encrypt CAVLC bit stream and CABAC bin strings of I and P frames using Cipher feedback (CFB) mode of the AES algorithm.
Asghar et. al., [13], present some encryptable parameters in the SVC that guarantee a constant bit rate and format compliant video encryption, such as: UEG3 suffix (where UEG3 is a concatenation of the Unary code and the EGK code when $K = 3$), UEG0 suffix and sign of all none zero QTCs.

5.4.2.6 Fast protection of AVC by reduced selective encryption of CAVLC

Fast protection of the AVC video content, based on a selective encryption for a subset of code-words/bin-strings of the CAVLC entropy coding is introduced by Dubois et. al.,[31]. This encryption scheme, like others uses the AES based on the CFB mode to encrypt a

subset of equal length code-words/bin-strings in the CAVLC entropy stage. Five syntax elements are used in the CAVLC entropy stage to code levels and runs: coeff token, signs of trailing ones, remaining non-zero levels, total number of zeros and runs of zeros. The encryptable parameters of this scheme are the sign of the trailing ones and the remaining non-zero levels to preserve the format compliance.
Nothing new in this scheme except that it analyses the level of the prediction error to decide which MB should be encrypted, but the selected encryptable parameters are not sufficient to provide a high security level and it affects the compression ratio.

5.4.2.7 Design of new unitary transforms for perceptual video encryption

A New and different approach of applying selective encryption for the AVC content has been proposed by Yeung et. al.,[160, 161]. The proposed scheme selects one out of multiple transform matrices based on the secret key bits. The proposed perceptual video encryption scheme introduced a new transformation matrices to be used instead of the DCT and DST matrices, during the transformation stage. Moreover, the sign of DCs is included in this encryption process. It is clear that this process requires to include these alternating transformation matrices in the standard encoder and decoder.

5.4.3 HEVC encryption algorithms

A perceptual encryption scheme has been introduced in [148] for the HEVC to degrade the video quality, whereas the full quality version is only permitted for authorized customers. The proposed scheme uses a similar encryption techniques that have been proposed in the previous video standard such as: encrypting the sign of the residual information, the MVD, the MV prediction index and the MV reference index.

Wallendael et. al., extended their previous research [148] to identify the HEVC syntax elements which can be changed/encrypted and preserved the format-compliant for the HEVC decoder as[147]:

- Short-term reference picture set (RPS).
- QP information (initial QP, Chroma delta QP, Slice delta QP, CU delta QP).
- Inter information (reference picture indices, MV prediction indices, motion merge indices, MVDs).
- Residual information.
- Deblocking filter parameters.
- Sample adaptive offset parameters.

Later, Shahid et. al., [121] proposed a selective encryption scheme to protect the HEVC. Most of Shahid works [120, 119, 117, 3, 116, 122, 114, 113, 112] has the same idea, which introduces a joint encryption and compression system based on CABAC bin-strings, see Figure 5.10.
In Shahid et.al., research, the sign bit of the QTCs, the suffix of the TRp, EG0, EG1, and the sign bit of the MVDs are the selective encryption scheme inputs. The encryption process of the sign bit is straightforward, while the encryption process of TRp, EG0 and

EG1 suffix codes needs a specific attention to preserve the format-compliance and to keep the bit ratio almost the same as compression without encryption.

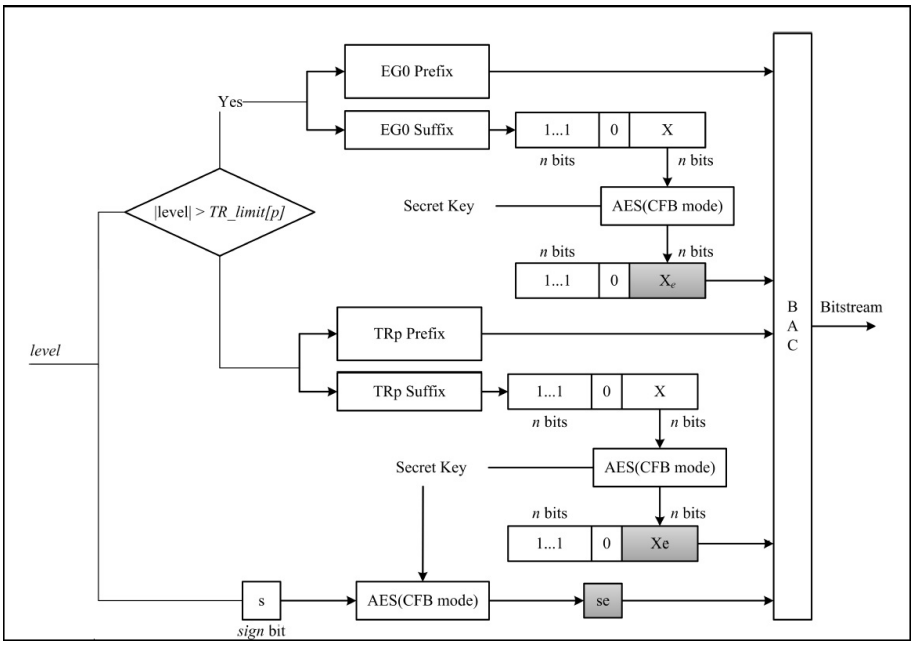


Figure 5.10: Basic structure of selective encryption scheme for the HEVC

Shahid et. al., define three basic requirements for SEA [121].

1. Preserve the same bit rate: the encrypted bin strings must have the same length as the unencrypted bin strings.

2. Preserve the format-compliant: the encrypted bin strings must be valid and decodable by any standard decoder.

3. Encrypt the dyadic encryption space: the author defines the dyadic encryption space as the one that its range is multiple of 2 (i.e., 7 is not a dyadic space where 8 is a dyadic one).

As described before, the binarization step is carried out based on five methods. Shahid et. al., work assumes that the first two methods (unary code and truncated unary code) do not fulfill the first requirement.
Then, they assume that the FLC does not fulfill the second requirement, since different bits in the FLC indicate different information regarding the header. However, the FLC binarization is not restricted to the header only, since the FLC is used in the suffix of TRp and EGk codes.
Finally, they assume that the suffix of the fourth (TRp) and fifth (EGk) codes can be encrypted without violation of any requirements.
The proposed encryption algorithm in Shahid et. al., [121] encrypts the suffix of the TRp and EGk codes except two cases:

1. In case of $p = 0$ in the TRp code, since the binary representation is identical to the truncated unary code, and so TR_0 does not fulfill the first requirement, as a result TR_0 is not encryptable.

2. In case of the last equal-length group of the TRp code, in this case, two binary codes are identical whether EG0 code is appended or not, and so, the encryption of this bin strings violates the format compliance requirement.

The encryption and decryption schemes use the AES algorithm in CFB mode [33].The process of encryption is done by grouping the EGk and TRp suffixes bits to prepare the 128 encryptable bits as a plaintext for the AES CFB algorithm.

Bellow, we address some comments on the Shahid et. al., work:

1. Binarization process of the residuals: Shahid et. al., in their work assume that the binarization of residuals is carried out using TRp and EG0. However, in the HEVC standard, as it is clear in algorithm 10, the EG0 will never be used. Consequently, this assumption is not valid.

2. In their encryption mechanism, it is obligatory to wait until the plaintext is completely filled or the slice boundary is reached (as they wrote "we fill with X_i encryptable bits until either the vector is completely filled or the slice boundary is reached."). This mechanism includes additional delay in the latency of the system and uses more memory to fill the plaintext vector (this is clear in the time complexity analysis of Shahid et. al., paper). Notice that, the decoder is working based on a prediction of unit by unit, and in most cases, one prediction unit is not sufficient to obtain a plaintext vector of 128 encryptable bits. Therefore, more memory and more delay are needed.

3. The number of encryptable bits in Shahid et. al., method is smaller than the number of encryptable bits that are obtianed in our proposed work (see algorithm 12 in chapter 6). In addition, their method does not compatible with the last HEVC standard.

5.5 Conclusion and discussion

In this chapter, a state of the art of video encryption and compression schemes is briefly introduced, to help researcher in cryptography to understand the critical and important properties of the video encryption standard.

Encryption and compression should be designed jointly to make sure that the encryption solution is compatible with any standard video decoder. In the last video standard (the HEVC) not all parameters are encryptable (in terms of format compliant). For instance, in the HEVC, three MPMs are defined instead of one MPM in the AVC. This difference has an important for instance for encryption systems previously defined for the AVC, that will be no longer compliant with the HEVC. Notice that, the encryption of some HEVC parameters is format compliant but they affect the compression ratio and the bit-rate, so the choosing of the HEVC parameters for encryption process depends on the target application.

Finally, to the best of our knowledge, there is no published work to identify the encryptable bin-string of the HEVC and of its scalable extension SHVC. Moreover, there is no published work in selective encryption algorithm for the HEVC or the SHVC in real-time applications.

6

Third Contribution: Selective encryption algorithms for Video

In this chapter, two fast and secure selective chaos-based crypto-compression systems are realized to encrypt the HEVC and its scalable version (SHVC). In the first section 6.1, we propose a new algorithm to identify the encryptable bits in the bin-string of the HEVC and the SHVC. Most of the encryption process is performed at the CABAC bin string level and it preserves all SHVC functionalities.
In the second section 6.2, the Region Of Interest (ROI) of the HEVC is encrypted based on the tile concept. The encryption is achieved at the bin-string level by encrypting all HEVC syntax elements within the ROI tiles, or by using a selective encryption of the ROI tiles under constant bit rate and format compliant requirements.

In fact, the design process of the video encryption algorithms should be compatible with any standard decoder and it should be fast to meet the real time application at the decoder side. Therefore, the most important requirements that should be taken into consideration during the designing process of any selective video encryption algorithms are:

1. Format Compliance: the proposed selective encryption algorithm should not violate the decoder, i.e., the encrypted video bit-stream must remain compliant under the standards of the video decoder.

2. Secure and fast encryption (low delay and low complexity): the proposed selective encryption should be secure and not increase the complexity or introduce additional delay to the original execution time of the real time encoding and decoding processes without encryption.

3. Constant bit-rate or at least no significant increasing of the compression ratio: in optimal video encryption, the encryption algorithm should not affect the original bit-rate. Otherwise, the increasing of the compression ratio should be small as possible.

4. Maintain the scalability features of the video.

6.1 Selective video encryption based on chaos system for SHVC

SHVC is the scalable extension of the HEVC standard. SHVC extension enables spatial, temporal, quality (SNR); bitdepth and color ghamut scalability. All defined technologies and techniques in the HEVC standard are also used in the SHVC: quadtree-based block partitioning, large transforms and prediction blocks, accurate intra/inter predictions, in-loop sample adaptive offset filter, highly adaptive entropy coding, CABAC binarization [137] and many more. Moreover, the HEVC standard uses the concept of dQP to adapt the QP value at the coding unit level for visual quality optimization and rate control. The SHVC extension encodes the original video in several layers corresponding to different spatial and quality representations of the video. The SHVC extension is defined to provide spatial, fidelity, bit depth and color scalability with a simple and efficient coding architecture [136, 23]. SHVC extension adopts an inter-layer prediction to take

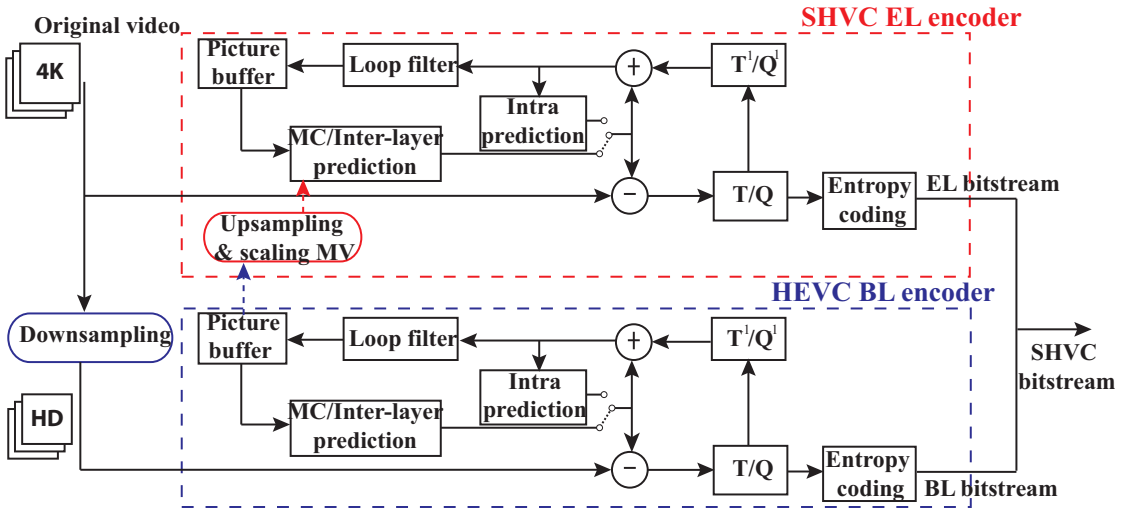


Figure 6.1: Block diagram of the SHVC encoder encoding two spatial scalability layers

advantage of spatial correlation and improve the rate-distortion performance compared to independently encoding of the layers. The SHVC encoder consists of L layers HEVC encoders, (one encoder by layer): one Base Layer (BL) and $L-1$ Enhancement Layers (EL). In the case of spatial scalability, the BL HEVC encoder encodes a down-sampled version of the original video and feeds the first EL encoder with the decoded picture and its MVs. The enhancement layer encoder encodes a higher resolution video with using the decoded picture from lower layer as an additional reference picture. The inter-layer reference picture is up-sampled and its MVs up-scaled to match with the resolution of the EL being decoded. Figure 6.1 shows an example of the SHVC encoder encoding two layers in spatial scalability configuration. In the case of SNR scalability, the encoding process remains unchanged except that the picture used for inter-layer prediction is used without being up-sampled and its MVs up-scaled. The binary arithmetic coder can be performed either by an estimated probability of a syntax element (context coded) or by considering equal probability of 0.5 (bypass coded). The three main functions of the CABAC at each SHVC layer are illustrated in Figure 5.8. The CABAC engine at each SHVC layer is initialized at the start of each frame and then the frame of each layer is encapsulated in

independent slice. The CABAC arithmetic coder in SHVC is similar to the one in the HEVC which is described in the previous chapter (see section 5.3.5). The proposed selective encryption algorithm is implemented using a chaos-based stream cipher (see Figure 6.2). A group of sensitive SHVC parameters is selected to be used as input for this selective encryption solution, the selected parameters meet the defined requirements in the beginning of this chapter. As it has been mentioned before, the encryption solution preserves all SHVC functionalities such as bit-stream extraction for mid-network adaptation and error resilience (i.e., enable the compressed bit-stream to resist channel errors so that the impact on the reconstructed frame is minimal).

A selective encryption scheme of SHVC is presented in this work: and it is applied on

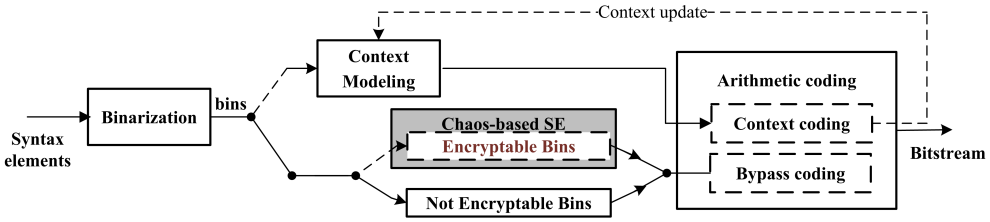


Figure 6.2: Encryption part of Crypto-A

three different stages: In the first stage, we encrypt only the lowest SHVC layer which is called SE-SHVC-BL, in the second one we encrypt all layers (SE-SHVC-ALL) and in the last stage we encrypt all layers except the base one (SE-SHVC-EL). Notice that, The encryption of the base layer (i.e., SE-SHVC-BL) will also affect the quality of the ELs since the decoded BL picture and its MVs are used as reference for inter-layer prediction at the EL encoders. Therefore, SE-SHVC-BL stage can achieve a high security level of encryption. The SE-SHVC-All stage achieve a very high security level of encryption as all layers are encrypted in this stage. Only end-users holding the secret key have the ability of the higher quality of the video. A comprehensive study is performed to assess and evaluate the performance of the proposed scheme on the three stages: different video encryption criteria, different scalability configurations and various High Definition (HD) video sequences.

6.1.1 Encryptable bit

The proposed encryption solution is SHVC format compliance and does not affect the compression ratio of the SHVC encoder. Therefore, only syntax elements binarized in fixed code are encrypted and then bypassed. The selective encryption is performed at position 1 in the CABAC process as illustrated in Figure 5.8.
The CABAC uses EG1 code for the binarization of MV differences which are then bypassed. Thus, the suffix part of the MV difference is safely encrypted since:

- The EG1 suffix has a fixed number of bits and encrypt them to the same number of bits, so this operation does not impact or violate any requirements defined requirements in the beginning of this chapter.

- The encrypted bins MV difference change 1 to 0 or vise verse and the result is bypassed, and so the compression ratio does not affected.

The sign of the MV difference is also encrypted since it is binarized in FL code with $cMax = 1$ and bypassed. The absolute value of the dQP is context coded so its encryption will affect its probability and so the compression ratio. We propose to encrypt only the dQP sign which is bypassed in the SHVC. Concerning the TCs, they are binarized with a combination of TRp with $cRiceParam \in \{0, 1, 2, 3, 4\}$ and EGk codes ($k = cRiceParam + 1$), encrypted and then bypassed.
The binarization process for non-zero QTC which is called $coeff\_abs\_level\_remaining$ in the JCT standard report [61], is carried out using the Truncated Rice Code with context P and (if presents) the Kth Order Exp-Golomb Code. The binarization process for the non-zero syntax element ($coeff\_abs\_level\_remaining$) is described in algorithm 10. The binarization bin strings are formulated by a concatenation of prefix bin strings and suffix (if it is existed) bin strings.

Algorithm 10 [Binarization process for non-zero coefficient, [61]]

if this process is invoked for the first time for the current sub-block scan index i **then**
$cLastAbsLevel = 0.$
$cLastRiceParam = 0$
else
$cLastAbsLevel = cAbsLevel.$
$cLastRiceParam = cRiceParam$
end if

$cAbsLevel = baseLevel + coeff\_abs\_level\_remaining$
if $cLastAbsLevel > (3 \times (1 << cLastRiceParam))$ **then**
$cLastRiceParam++$
end if
$cRiceParam = Min(cLastRiceParam, 4)$
$cMax = 4 << cRiceParam$
$prefixVal = Min(cMax, coeff\_abs\_level\_remaining)$
Called Trp function for $prefixVal$ using $cMax$ and $cRiceParam$
if $Binarization(prefixVal) = 1111$ **then**
$suffixVal = coeff\_abs\_level\_remaining - cMax$
Called EGK function for $suffixVal$ using $K = cRiceParam + 1$
end if

The suffix of the EGk code can be safely encrypted, while encrypt the TRp suffix is not a format compliance since, it affects the $cRiceParam$ parameter value and consequently the compression ratio.
A proposed algorithm is introduced to accurately determine the bins of the TRp suffix that can be encrypted without changing/updating the value of the $cRiceParam$. The proposed algorithm is tested and evaluated to be sure that the defined bins are safe to encrypt while preserving the defined requirements. The $cRiceParam$ parameter value is updated after the binarization of each TC depending on its absolute value $cAbsLevel$, based on the following algorithm.

where $cRiceParam$ is initialized to 0 at the start of each transform sub-block [1].
The absolute value of the QTC (which is here $cAbsLevel$) is composed of the base level named $baseLevel$ plus the remaining part of the QTC named $Coef$ (i.e., $Coef = cAbsLevel - baseLevel$). The value of the $baseLevel$ is computed depending on the value

Algorithm 11 [Update the $cRiceParam$ parameter in TRp code]

if ($cAbsLevel > 3 \times 2^{c}RiceParam$) **then**
$cRiceParam = min(cRiceParam + 1, 4)$
end if

$cAbsLevel$ and the position of the QTC in the transform sub-block where $baseLevel \in \{1, 2, 3\}$ before applying the previous equation. The base level value is first signaled in the bit-stream with a specific syntax elements and then only $Coef$ different from 0 is binarized in TRp and EGk codes.
We propose Algorithm 12 to identify the bit positions (from the least significant bin) of the encryptable bins in the TRp suffix. This algorithm is compatible with all encoder versions of the HEVC standard. In general, the original and the encrypted QTC, both should be less than the threshold (i.e., $3 \times 2^{c}RiceParam$) or both should be more than the threshold.
In the case of base level equals to 1, the whole suffix of the binarized QTC can be encrypted since the $Coef$ value plus 1 (the $baseLevel$ here is 1) never exceeds the threshold to update the $cRiceParam$ for all possible suffix values. Indeed
$3 \times 2^{cRiceParam} - baselevel$ gives the maximum possible suffix of the $cAbsLevel$ value (see tables 1-5 of appendix Appendix 2). The $cRiceParam$ parameter (see algorithm 11) is never updated in case of the $baseLevel$ is 1 since the encrypted $Coef$ plus 1 is $\leq 3 \times 2^{cRiceParam}$. The following examples are given to illustrate that when the baseLevel is 1 then the whole suffix part of the QTC is encrypted safely, the binarization of the QTC is based on the algorithm 10 using TRp and EGK binarization methods.
Example6.1: Assume that $cAbsLevel = 6$, $cRiceParam = 1$ and $baselevel = 1$ (note that the threshold in this case is 6), then the binary code-word of 6 is 1101 (see line 6 of Table 1 in the appendix 2) with one bit as suffix, so the possible encryption of the suffix bit is 1 or 0, and consequently binary code-word are:$\{1101$ or $1100\}$. As the both encrypted values less than the threshold there is no update on the $cRiceParam$. The same producer can be applied for $cAbsLevel = \{1,\ 2,\ 3,\ 4,\ 5\}$.
Now under the same previous parameter but with $cAbsLevel = 7$, then the binary code-word of 7 is 11100 (see line 7 of Table 1 in the appendix 2) with one bit as suffix, so the possible encryption of the suffix bit is 1 or 0, and consequently binary code-word are:$\{11100$ or $11101\}$. As the both encrypted values more than the threshold there is no update on the $cRiceParam$. The same producer can be applied for $cAbsLevel > 7$.
The same derivation of Example6.1 can be used when the $cRiceParam$ is 2, 3 or 4.

In the following we discuss the encryption configurations provided in Table 6.1 for base level different from 1. Table 6.1 provides the encryptable bins in the suffix of the TC binarized in TRp code with $cRiceParam = 3$. The threshold computed by Algorithm 11 to update the parameter $cRiceParam = 3$ is equal to 24 (i.e., 3×2^{cRiceParam}). When the $Coef$ value is less than 16 or greater than 23 the three bins of the suffix can be safely encrypted since, in the first case ($Coef$ less than 16) then the encryption is less less than 16 which is less than the threshold, while in the second case ($Coef$ more than 23), then the encryption is more than 23 which is more than the threshold.
When the $Coef$ value is equal or bigger than 16 and less than 20, only the first two bins of the suffix can be encrypted for the $baseLevel \in \{2,\ 3\}$. Because encrypting the third bin (the not bold one) can increase the $cABsLevel$ to be greater than the threshold which means the $cRiceParam$ parameter is updated in this case, while the non encrypted value

Algorithm 12 [Encryptable bins in the TRp suffix of TCs]

if ($baseLevel == 1$) **then**
The whole suffix is encryptable.
else if ($cRiceParam == 1$) **then**
if ($baseLevel == 2$ AND ($Coef == 4$ OR $Coef == 5$)) **then**
No encryption.
else
The whole suffix is encryptable.
end if
else if ($cRiceParam == 2$) **then**
if ($Coef \leq 7$ OR $Coef \geq 12$) **then**
The whole suffix is encryptable.
else if ($baseLevel == 2$ AND ($Coef == 10$ OR $Coef == 11$)) **then**
No encryption.
else
The first bin of the suffix is encryptable.
end if
else if ($cRiceParam == 3$) **then**
if ($Coef \leq 15$ OR $Coef \geq 24$) **then**
The whole suffix is encryptable.
else if ($Coef \leq 19$) **then**
The first two bins of the suffix are encryptable.
else if ($baseLevel == 2$ AND ($Coef == 22$ OR $Coef == 23$)) **then**
No encryption.
else
The first bin of the suffix is encryptable.
end if
else if ($cRiceParam == 4$) **then**
if ($Coef \leq 31$ OR $Coef \geq 48$) **then**
The whole suffix is encryptable.
else if ($Coef \leq 39$) **then**
The first three bins of the suffix are encryptable.
else if ($Coef \leq 43$) **then**
The first two bins of the suffix are encryptable.
else if ($baseLevel == 2$ AND ($Coef == 46$
OR $Coef == 47$)) **then**
No encryption.
else
The first bin of the suffix is encryptable.
end if
end if

Table 6.1: Encryptible bins in bold font of the TC suffix binarized in TRp code with $cRiceParam = 3$ and *cABsLevel=baseLevel+Coef*

Coef	*baseLevel*		Preffix	Suffix
	2	3		
	$cAbsLevel$			
14	16	17	10	**110**
15	17	18	10	**111**
16	18	19	110	0**00**
17	19	20	110	0**01**
18	20	21	110	0**10**
19	21	22	110	0**11**
20	22	23	110	10**0**
21	23	24	110	10**1**
22	24	-	110	110
22	-	25	110	11**0**
23	25	-	110	111
23	-	26	110	11**1**
24	26	27	1110	**000**

$cABsLevel$ is less than the threshold (see Table 3 in appendix 2). As an example, assumes the binary code-word of the original $cABsLevel$ is 110001 which is 19 or 20 based on the $baseLevel = 2$ or $= 3$. The suffix consists of the last three bits, now assumes that the encryption of the last three bits is 111, then the encrypted $cABsLevel$ is 25 or 26 (the both are bigger than threshold which is 24), which means, the both encrypted $cABsLevel$ update the $cRiceParam$ while the original $cABsLevel$ (less than the threshold) does not update the $cRiceParam$.
In case of the $Coef$ is 20 or 21, only the first bit (LSB) is safely encryptable. The possible encryption of $Coef = 20$ is 20 or 21 and the possible encryption of the $Coef = 21$ is also 20 or 21. To justify the restriction (only one encryptable bit) on the encryption of $Coef = 20$ or $Coef = 21$, let us consider the suffix part [100,101] of each one respectively. We have three possible encryption scenario:

- Encrypt one bit (the LSB one), this is a safe encryption since, the both $cABsLevel$ and the encrypted $cABsLevel$ are less than the threshold.
- Encrypt 2 bits, the possible encryption of the suffix term is [100,101,110,111], then as the last term (111) gives an encrypted $cABsLevel = 25$ which is bigger than 24 consequently, this scenario of encryption is rejected.
- Encrypt 3 bits, this scenario is rejected, because scenario two is is included in it.

In the case of $Coef$ value is equal to 22 or 23 with base level equals to 2, the two suffixes can not be encrypted since, changing one bin in the suffix *produces*a $cABsLevel$ on the other side of the threshold.
Algorithm 12 enables to encrypt in a format compliance all possible bins of the suffix of the TCs binarized in TRp code, which maximize the number of the encryptable bits.

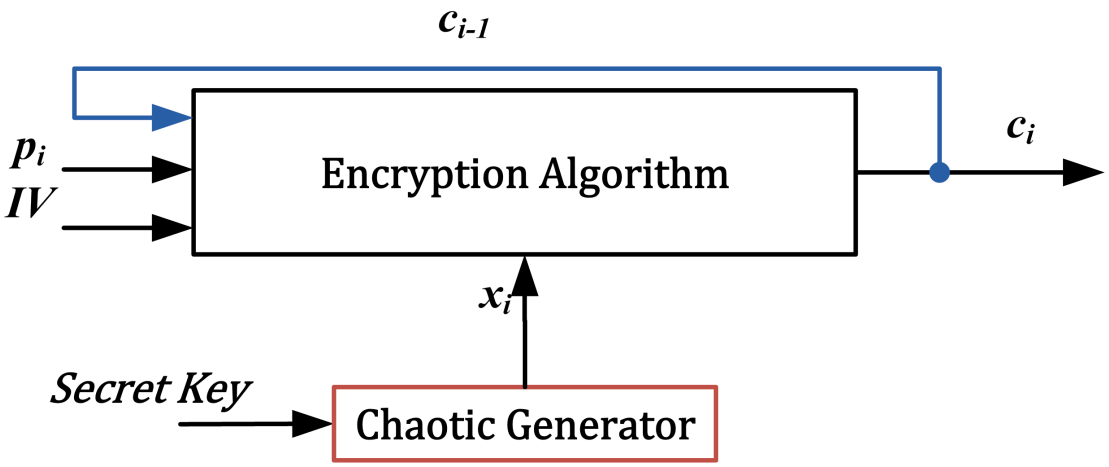
Table 6.2: Encrypted syntax elements in the proposed SHVC selective encryption solution, all theses syntax elements are bypass coded

Syntax elements	Binarization	Encrypted part
MV dif.	EG1	Suffix
MV dif. sign	FL	1 bin
TCs	TRp and EGk	EGk suffix and TRp suffix as in Alg. 12
TC sign	FL	1 bin
dQP sign	FL	1 bin

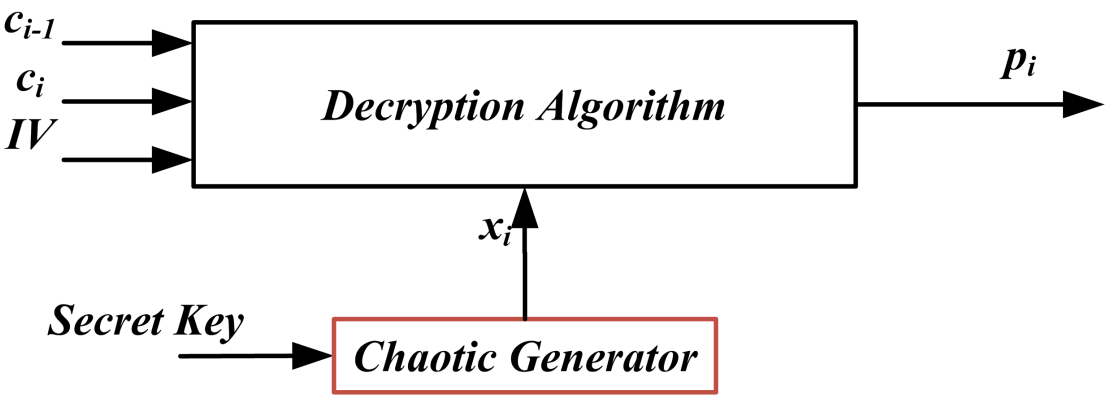
Finally, the sign of the TC is encryptable. Table 6.2 summarizes the encryptable parameters in the proposed selective encryption solution of the SHVC.

6.1.2 Chaotic encryption system

The principle of the chaotic encryption/decryption stream cipher system is illustrated in Figure 6.3. The encryption process is carried-out, syntax element by syntax element using a simple xor and addition operations.



(a) Encryption process



(b) Decryption process

Figure 6.3: Chaos-based video encryption/decryption stream cipher

$$c_i = p_i \oplus (x_i + c_{i-1}) \tag{6.1}$$

Where p_i is a vector of the encryptable bins which is a part of the syntax element, c_{i-1} is a vector of the encrypted bins of the previous syntax element, and x_i is the generated bits (keystream) from the chaotic generator described in chapter 3, section 3.1.2. The first encrypted value (c_0) is computed as follows:

$$c_0 = p_0 \oplus (x_0 + IV) \tag{6.2}$$

with IV is an a Initial Vector of 32 bit size (the IV size is justified bellow).
The confusion effect is obtained by mixing the plain vector p_i with the key stream, while the diffusion effect is obtained by using the previous ciphered value c_{i-1}.
At the decoder side, the decryption is performed as follows:

$$p_i = c_i \oplus (x_i + c_{i-1}) \tag{6.3}$$

with $p_0 = c_0 \oplus (x_0 + IV)$.

6.1.3 Synchronization problem

Synchronization between the encoder and the decoder is a challenging issue when the selective encryption algorithm is used with a stream cipher system. Moreover, the encryption must be robust to packet losses (i.e., enable re-synchronization of the decryption even after packet loss occurs in the network). Therefore, the joint compression/encryption system should be carefully designed to enable a secure encryption while preserving all SHVC video features including sub-stream extraction and error resilience; and also with introducing a minimum bitrate overhead.
To preserve all SHVC functionalities, the dependency of the chaotic generator will follow the SHVC coding dependency, including temporal dependency (inter prediction) between frames of the same layer and dependency between SHVC layers (inter-layer prediction). We consider one independent version of the chaotic generator for each SHVC layer, where each version will generate the keystream (x_i of the equation 6.1) used to encrypt the syntax elements of the corresponding SHVC layer. This enables an independent encryption and decryption of the L SHVC layers. L different secret keys must be shared between the encoder and the decoder to initialize the L versions chaotic generator. The end-to-end transmission of the secret key is out of the scope of this work and can be carried-out with any usual method (a symmetric algorithms and quantum cryptography) . Each chaotic generator is initialized with the same secret key and a new IV at each new Clean Random Access (CRA) frame [126]. This enables a safe sub-stream extraction with a correct decryption of the BL and the corresponding ELs, even when previous frames are not extracted and decoded. The IV is generated by a pseudo random generator of 32 bits, the maximum size of one syntax element binstring in the HEVC standard. The pseudo random IV is encrypted with AES encryption algorithm in block cipher mode and then is inserted in the SHVC bit-stream as Supplemental Enhancement Information (SEI) at the start of each CRA frame of the corresponding layer. Figure 6.4 illustrates the structure of the encrypted SHVC bit-stream with two layers using SE-SHVC-All encryption scheme. The Video Parameter Set (VPS) header contains information related to the whole video then, the Sequence Parameter Set (SPS) and Picture Parameter Set (PPS) headers containing information of the BL are signaled. The SEI message containing the IV of the

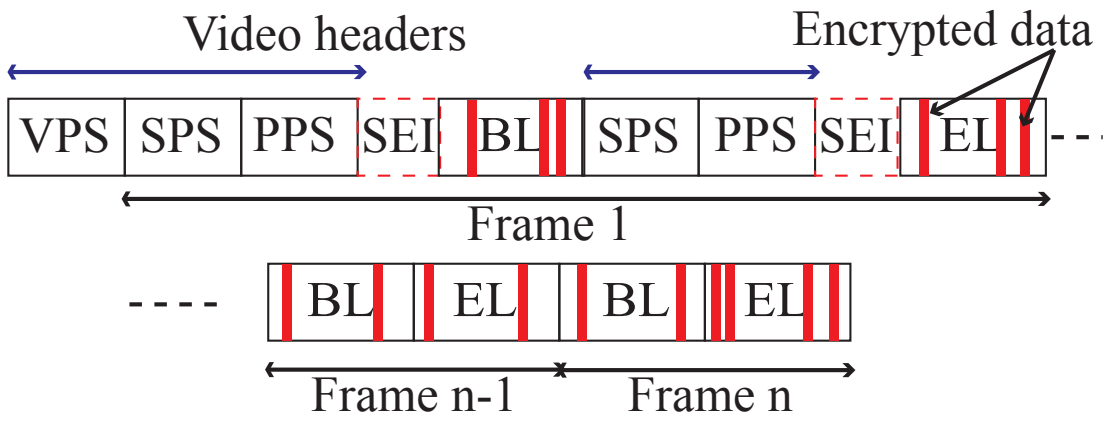


Figure 6.4: Structure of the encrypted SHVC bit-stream with two layers using SE-SHVC-All encryption scheme

first BL frame is inserted before the BL slice. The SPS and PPS headers of the EL are signaled before the SEI message containing the IV of the second chaotic generator then followed by the first EL slice. The encrypted data at both the BL and EL slices in the SE-SHVC-All stage is highlighted by red segments, referring to the selective encryption. Therefore, the proposed selective encryption scheme does not encrypt the video headers including slice headers, information usually used for mid-network adaptation. The SEI messages containing the IV of 32 bits introduces rate distortion overhead. However, this overhead remains very low since the SEI are inserted only before CRA frames and the size of this SEI message is much lower than the size of the BL plus EL slices.

6.1.4 Experimental configuration

The proposed encryption scheme is implemented in the Scalable Reference software Model (SHM) encoder version 4.1 [51]. The decryption algorithm is implemented under the optimized [1] SHVC decoder *OpenHEVC* [50]. This allows to assess the complexity overhead of the decryption process in the context of the real time SHVC decoder. We consider the common SHVC test conditions [111]. The configuration of the test video sequences is provided in Table 6.3. These video sequences are encoded in low delay P configuration (I frame followed by P frames), with two layers ($L = 2$) and three scalability configurations: two spatial configurations with ratios 2x, 1.5x and one fidelity (SNR) configuration. We consider three QP configurations: EL QP is $QP_{EL} \in \{22, 26, 34\}$ and the corresponding BL QP which is equal to the QP_{EL} in spatial scalability configurations and $QP_{BL} \in \{26, 30, 38\}$ in SNR scalability. We use both Peak Signal to Noise Ration (PSNR) and the Structural SIMilarity (SSIM) criteria to assess the quality of the decoded of the correct and incorrect decryption of the videos.

6.1.5 Objective quality and Encryptable Bit (EB)

Tables (6.4-6.6) present the average performance in terms of PSNR of the Y component, SSIM and EB respectively, using the proposed encryption scheme of the three stages, for

[1] Optimized software refers in this work to a code-source written in Single Instruction Multiple Data (SIMD) operations.

Table 6.3: Video sequences considered in the experiments

Class	Sequences	Resolution	Frame rate (Hz)	duration (second)
B	*Kimono*	1920x1080	24	10
	ParkScene		24	10
	Cactus		50	10
	BasketBallDrive		50	10
	BQTerrace		60	10
A	*Traffic*	2560x1600	30	5
	PeopleOnStreet		30	5

video classes A and B at one particular EL QP configuration (QP_{EL}=22). The encryption of the BL of both stages SE-SHVC-BL and SE-SHVC-All, encrypting only the BL and both layers respectively, drastically decrease the objective quality of the video sequences. Indeed, the obtained average PSNR Y values are below 10 dB and their average SSIM values are below 0.2 in all scalability configurations. The encryption scheme SE-SHVC-BL considerably decreases the objective quality of both layers since the inter-layer prediction used in the SHVC extension propagates errors that resulted from the encrypted information of the base layer.

Table 6.4: Video quality PSNR Y of the proposed SHVC encryption scheme of the three stages

Class	Scalability	QP_{BL}-QP_{EL}	No Enc	BL PSNR	ALL PSNR	EL PSNR
A	SNR	26-22	41.12	8.28	8.25	23.69
	2X	22-22	41.3	9.04	8.96	17.72
	HEVC	.-22	41.25	-	-	8.55
B	SNR	26-22	39.54	9.18	9.13	25.77
	2X	22-22	39.6	9.82	9.66	18.65
	1.5X		39.57	9.11	9.03	21.97
	HEVC	.-22	39.6	-	-	9.29

In fact, the EL decoder uses reconstructed samples of the BL picture as reference for the inter-layer prediction and also uses the encrypted BL MVs in the inter layer merge

Table 6.5: Video quality (SSIM) of the proposed SHVC encryption scheme of the three stages

Class	Scalability	QP_{BL}-QP_{EL}	No Enc	BL SSIM	ALL SSIM	EL SSIM
A	SNR	26-22	0.95	0.16	0.13	0.69
	2X	22-22	0.95	0.14	0.1	0.48
	HEVC	.-22	0.95	-	-	0.1
B	SNR	26-22	0.91	0.18	0.15	0.69
	2X	22-22	0.91	0.19	0.14	0.42
	1.5X		0.91	0.18	0.15	0.6
	HEVC	.-22	0.91	-	-	0.14

Table 6.6: Video quality (EB) of the proposed SHVC encryption scheme of the three stages

Class	Scalability	QP_{BL}-QP_{EL}	No Enc	BL PSNR	ALL PSNR	EL ES
A	SNR	26-22	0	7.27	15.06	7.79
	2X	22-22	0	5.62	16.61	10.98
	HEVC	.-22	0	-	-	17.83
B	SNR	26-22	0	6.23	15.72	9.49
	2X	22-22	0	4.11	17	12.89
	1.5X		0	6.31	16.41	10.1
	HEVC	.-22	0	-	-	18.27

mode[2]. We can also notice that the SE-SHVC-All encryption scheme enables to further decrease the objective quality of the SHVC video by 0.1 dB to 0.2 dB with respect to the SE-HEVC-BL encryption scheme.
The SE-SHVC-EL encryption scheme (encrypting only the EL) slightly decreases the objective quality of the EL video. This is because the most part of the information is predicted from the clear BL while only details (difference between BL and EL) of the video are encoded and encrypted at the EL.
The EB of the SE-SHVC-BL remains in average less than 8% of the whole SHVC video bit-stream including BL and EL. This low ES is obtained thanks to the selective encryption and the proposed algorithm 12, where only the most sensitive syntax elements are encrypted and also because of the size of the BL bit-stream which is lower than the size of the non-encrypted EL, particularly in spatial scalability configurations. The EB in the SE-HEVC-EL stage is around 11% of the whole SHVC video bit-stream and encryption of both layers with SE-SHVC-All stage increases the EB to 16% in average. The increasing in the EB (from 8% to 11% to 16%) introduces an extra complexity regarding the brute force attack.
The PSNR Y and the SSIM performance of the three considered scheme in all scalability configurations is provided in Table 6.7 for the 1920×1080p50 *Cactus* video sequence at different QP configurations. We can notice, that the proposed encryption scheme of the three stages decreases the objective quality of the video to the same low quality level, whatever the QP values and the corresponding initial quality of the video.

Table 6.8, presents the EB of the same parameter that used in the previous Table(6.7). We observe that, the EB is slightly decreased with high QP values in SE-SHVC-All encryption scheme since less syntax elements are present at low bit-rate configuration. Moreover, in the SE-SHVC-BL and SE-SHVC-EL encryption stages, the EB depends, not only on the QP but also on the scalability configuration which changes the resolution of the BL and the correlation degree between the two layers (related to the video sequence, the scalability configuration and the QP used at each layer). Tables (6.9-6.10) show the BL and the EL PSNR of the three color components (Y, U and V) of the 2560×1600p30 *Traffic* video sequence, encrypted with the SE-SHVC-BL encryption stage at different scalability and QP configurations. Table6.10 shows that the SE-SHVC-BL encryption stage decreases the PSNR of both layers to the same PSNR value in different QP and scalability configurations. The PSNR of the BL and EL is decreased to around 8.5 dB for luminance Y component and 13 dB and 15 dB for U and V color components, respectively.

Table 6.11 gives the contributed of the EB of each SHVC syntax elements for the three stages of the *Traffic* video sequence in two QP configurations. for high and low bit rate configurations, the TC sign represents the most encrypted syntax element with more than 85% and 73% in the two cases of the bit rate configurations for the SE-SHVC-All encryption scheme.

[2]Merge mode in the SHVC inter-layer prediction uses the MVs from the BL for motion compensation.

Table 6.7: Video quality of the three proposed SHVC encryption schemes for the 1080p50 $Cactus$ video sequence

QP_{BL}-QP_{EL}	Sca.	No encryption		SE-SHVC-BL		SE-SHVC-All		SE-SHVC-EL	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
26-22	SNR	38.65	0.9	8.69	0.19	8.65	0.16	24.68	0.7
22-22	2x	38.68	0.9	8.72	0.2	8.63	0.14	17.5	0.47
	1.5x	38.66	0.9	9.27	0.21	9.17	0.16	20.25	0.6
.-22	HEVC	38.68	0.9	-	-	-	-	9.04	0.14
30-26	SNR	37.09	0.87	8.78	0.25	8.73	0.21	23.25	0.66
26-26	2x	37.16	0.88	8.99	0.27	8.83	0.18	17.79	0.49
	1.5x	37.10	0.88	8.68	0.2	8.67	0.17	21.76	0.64
.-26	HEVC	37.15	0.88	-	-	-	-	8.81	0.2
38-34	SNR	33.66	0.82	7.75	0.29	7.75	0.27	24.51	0.69
34-34	2x	33.78	0.82	8.28	0.23	8.28	0.2	19.73	0.55
	1.5x	33.66	0.82	9.61	0.25	9.59	0.24	22.97	0.66
.-34	HEVC	33.92	0.83	-	-	-	-	8.31	0.26

Table 6.8: Encryptable bit of the proposed SHVC encryption scheme for the 1080p50 $Cactus$ video sequence

QP_{BL}-QP_{EL}	Scalability	SE-SHVC-BL	SE-SHVC-All	SE-SHVC-EL
26-22	SNR	4.97	14.99	10
22-22	2x	3.38	16.39	13.01
	1.5x	5.23	15.6	10.37
.-22	HEVC	-	-	16.91
30-26	SNR	6.86	14.16	7.3
26-26	2x	4.92	15.61	10.69
	1.5x	7.28	14.64	7.36
.-26	HEVC	-	-	16.66
38-34	SNR	6.66	12.41	5.74
34-34	2x	4.69	13.63	8.93
	1.5x	6.69	12.58	5.61
.-34	HEVC	-	-	14.69

Table 6.9: BL PSNR of the $Traffic$ video sequence in different scalability and QP configurations: SE-SHVC-BL encryption scheme

QP_{BL}-QP_{EL}	Scalability	BL PSNR (dB)					
		No encryption			SE-SHVC-BL		
		Y	U	V	Y	U	V
26-22	SNR	39.46	39.98	42.45	8.29	12.78	14.12
22-22	2x	40.77	42.42	44.25	8.78	13.96	14.42
.-22	HEVC	-	-	-	-	-	-
30-26	SNR	37.37	38.87	41.3	8.96	12.36	13.7
26-26	2x	38.04	40.55	42.51	8.46	15.68	17.15
.-26	HEVC	-	-	-	-	-	-
38-34	SNR	33.06	36.92	39.28	8.36	14.52	20.29
34-34	2x	32.98	37.46	39.76	9.11	16.36	18.11
.-34	HEVC	-	-	-	-	-	-

Table 6.10: EL PSNR of the $Traffic$ video sequence in different scalability and QP configurations:SE-SHVC-BL encryption scheme

QP_{BL}-QP_{EL}	Scalability	EL PSNR (dB)					
		No encryption			SE-SHVC-BL		
		Y	U	V	Y	U	V
26-22	SNR	41.60	41.46	43.86	8.08	13.53	14.62
22-22	2x	41.67	41.50	43.96	9.44	12.49	14.61
.-22	HEVC	41.69	41.52	44.05	8.39	13.19	14.66
30-26	SNR	39.31	39.85	42.24	8.40	14.01	19.92
26-26	2x	39.41	39.91	42.32	9.21	16.97	17.54
.-26	HEVC	39.46	39.98	42.45	8.29	12.78	14.12
38-34	SNR	34.96	37.47	39.83	8.40	14.01	19.92
34-34	2x	35.06	37.50	39.84	8.84	14.03	14.83
.-34	HEVC	35.23	37.66	40.00	8.08	16.41	19.68

Table 6.11: Repartition of the encrypted SHVC syntax elements in the proposed scheme for $Traffic$ video sequence

Syntax element	QP_{EL}	SE-SHVC-BL		SE-SHVC-All		SE-SHVC-EL		
		SNR	2x	SNR	2x	SNR	2x	HEVC
MV diff.	22	2	0.68	3.15	2.88	1.15	2.2	3.22
MV diff. sign		3.45	1.75	6.61	6.91	3.15	5.1	6.36
TCs		3.66	3.01	3.67	3.4	0.01	0.38	7.39
TC sign		34.53	24.17	85.74	86.28	51.2	62.06	82.52
dQP sign		0.33	0.11	0.81	0.55	0.48	0.44	0.49
Sum		43.98	29.73	100	100	56.01	70.26	100
Bitrate (Mbit/s)		12.6	9.4	28.66	31.63	16.05	22.22	29.03
MV diff.	34	5.67	2.73	9.59	9.27	3.91	6.53	7.62
MV diff. sign		5.94	4.33	12.06	13.48	6.11	9.14	9.37
TCs		2.94	1.76	2.94	1.79	0	0.02	6.39
TC sign		44.89	32.75	73.56	74.05	28.66	41.3	75.53
dQP sign		0.86	0.31	1.83	1.39	0.96	1.07	1.07
Sum		60.32	41.90	100	100	39.67	58.09	100
Bitrate (Mbit/s)		1.8	1.25	2.99	3	1.18	1.74	3.28

The proportion of the TC sign is higher than the TC, since the most part of the TC values x are different from zero and lower than the base level ($baseLevel$) which are not binarized (i.e., remaining part $Coef$ is equal to 0) while their sign is signaled. We can also notice that the proportion of the MVs difference sign and the MVs difference are slightly increased at low bit rate configuration. The EB of the encryption scheme for SE-SHVC-BL and SE-SHVC-EL shows that the scalability configuration impacts the ES re-partition, mostly caused by the resolution of the BL and the correlation between these two layers. Finally, the dQP sign represents less than 1% and 2% of the encrypted syntax elements in SE-SHVC-All scheme at high and low bit rate configurations, respectively.

6.1.6 Visual quality

Figure 6.5 shows the visual quality of the frame #9 of *BasketballDrive* video sequence. It is encrypted by the proposed scheme for the BL and the EL. Figure 6.5(a) shows the effect

(a) BL PSNR Y = 8.97 dB

(b) EL PSNR Y = 8.98 dB

(c) BL PSNR Y = 8.97 dB

(d) EL PSNR Y = 8.88 dB

(e) BL PSNR Y = 39.35 dB

(f) EL PSNR Y= 17.59 dB

Figure 6.5: Visual quality of frame #9 of the $BasketballDrive$ video sequence in SNR scalability configuration (a) (b) SE-SHVC-BL, (c) (d) SE-SHVC-ALL and (e) (f) SE-SHVC-EL)

of applying the encryption scheme (SE-SHVC-BL) on the base layer, while Figure 6.5(b) shows the effect of applying the same encryption scheme (SE-SHVC-BL) on the enhancement layer. Figure 6.5(c) shows the effect of applying the encryption scheme (SE-SHVC-ALL) on the base layer, while Figure 6.5(d) shows the effect of applying the same encryption scheme (SE-SHVC-ALL) on the enhancement layer. Figure 6.5(e) shows the effect of applying the encryption scheme (SE-SHVC-EL) on the base layer, while Figure 6.5(f) shows the effect of applying the same encryption scheme (SE-SHVC-EL) on the enhancement layer. It is clear that, applying the encryption scheme (SE-SHVC-EL) does not affect the base layer (see Figure 6.5(f)) and it remains without encryption.
We can notice that SE-SHVC-BL scheme encrypts only the BL which affects the visual quality of the EL. The inter-layer prediction using the decoded BL picture and its MVs propagates the errors to the EL. However, SE-SHVC-EL scheme encrypts only the EL, while the BL remains clear and the quality of the EL is slightly decreased compared to the BL. This is because the most part of information is predicted from the base layer and only details (encrypted data) are encoded at the level of the EL. We can notice that the proposed SE-SHVC-EL encryption scheme leads to a perceptual (transparent) encryption solution by decreasing the visual quality of EL below the quality of the BL while the EL video is still recognized. However, the SE-SHVC-BL encrypting scheme enables a more secure encryption solution by drastically decreasing the visual quality of all layers. Additional analysis on security parameters of these proposed selective encryption scheme of the three stages are investigated in the next paragraph.

6.1.7 Security analysis

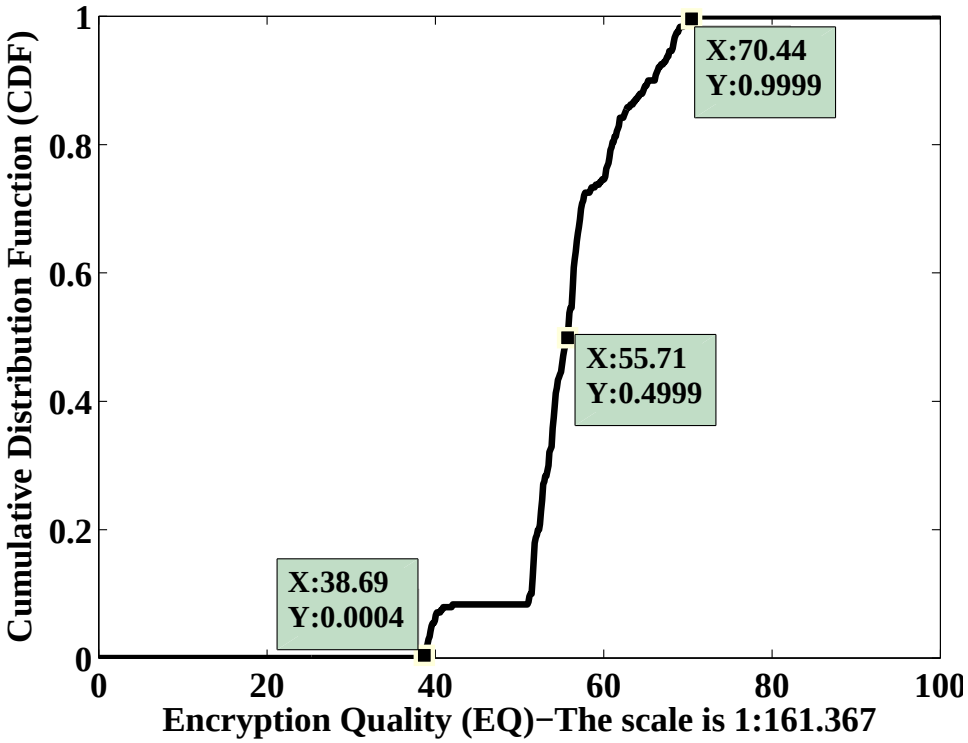
The evaluation process of the selective encryption is a little bit different from the evaluation process of the standard encryption algorithms. Normally, the expected security level of the stream selective encryption algorithm is less than the expected security level of the full stream encryption algorithms (and also, it is less than block encryption algorithm). In this section different types of attacks and security parameters are used to evaluate the robustness of the proposed stream selective encryption scheme.

6.1.7.1 Encryption Quality

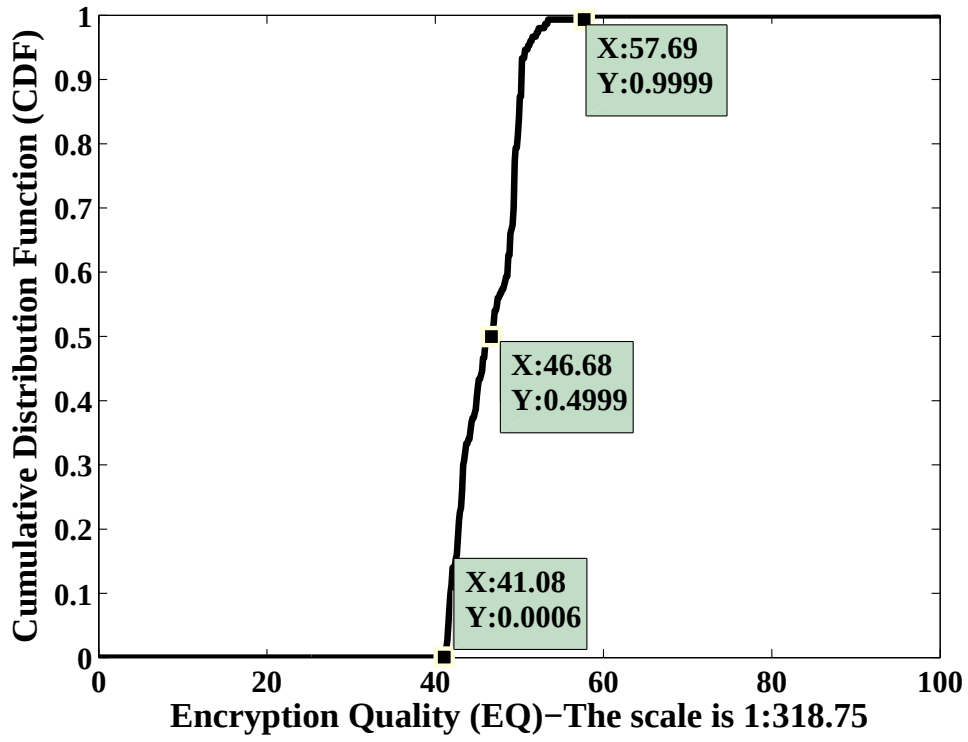
The difference between the frequency of occurrence for each byte with and without encryption is called Encryption Quality (EQ), which is defined in section 2.2.7 of chapter 2. Based on the equation (2.20), we calculate the maximum EQ value for a frame of *Kimono* and *PeopleOnStreet* video sequences which are equal to 16136.7 and 31875, respectively (all frames for each video sequence have the same maximum EQ) . We can notice from Table 6.12 that SE-SHVC-BL and SE-SHVC-All reach in average EQ value around the half of the maximum EQ value in all scalability configurations for the both video sequences. However, the SE-SHVC-EL gives low EQ value, that corresponds to the perceptual video encryption target. In Figure 6.6, we give the curves of the Cumulative Distribution Function (CDF) of the EQ for *Kimono* (a) and *PeopleOnStreet* (b) video sequences. We can notice that all frames of *Kimono* video sequence have an EQ higher than 38.69% of its maximum EQ. While all frames of the *PeopleOnStreet* video sequence have an EQ value higher than 41.08% of its maximum EQ.

Table 6.12: Encryption Quality for$Kimono$ and$PeopleOnStreet$ video sequences at QP$_{EL}$=22

Video	Sca.	SE-SHVC-BL	SE-SHVC-All	SE-SHVC-EL
Kimono	SNR	9025	8996	684
	2x	7651	7675	1101
	1.5x	9895	9900	571
	HEVC	-	-	9355
People-OnStreet	SNR	14833	14884	3355
	2x	14528	14739	5161
	HEVC	-	-	14129



(a) Kimono



(b) PeopleOnStreet

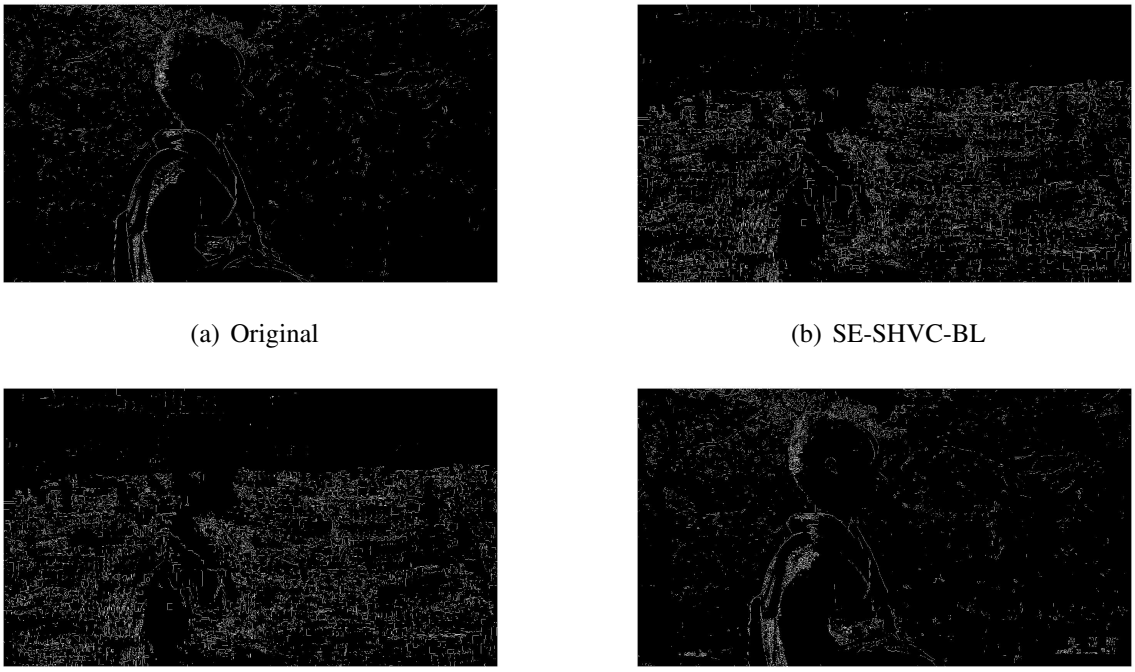
Figure 6.6: CDF of the EQ for $Kimono$ and $PeopleOnStreet$ video sequences in SNR scalability, QP_{EL}=22 and SE-SHVC-BL encryption scheme

6.1.7.2 Edge differential ratio

Edge Differential Ratio (EDR) evaluates the edges differences between the original and the encrypted frame [141, 140]. The encryption solution is more efficient when the the edges of the encrypted frames are not detectable. The EDR is calculated as:

$$EDR = \frac{\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}|P_E(i,j) - C_E(i,j)|}{\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}|P_E(i,j) + C_E(i,j)|} \tag{6.4}$$

Where P_E and C_E are the edge detected binary matrix for the plain-frame and the ciphered-frame, respectively, produced by using the Laplacian of Gaussian method [53].
Table 6.13 presents the average evaluation results of the EDR for the proposed scheme of all stages.
When the average EDR values are close to 1, means that the encryption solution of SE-SHVC-BL and SE-SHVC-All have a high ability to hide the edges of the encrypted frames and so a high security level is reached.
In fact, the structural information (edges...) of the encrypted frame by the SE-SHVC-BL and SE-SHVC-All are completely hidden and becoming useless by the attacker.
Figure 6.7 shows the edges of frame #8 of *Kimono* video sequences. It confirms the obtained results of Table 6.13, calculated based on the equation (6.4).



(a) Original

(b) SE-SHVC-BL

(c) SE-SHVC-All

(d) SE-SHVC-EL

Figure 6.7: Edges illustration of frame #8 $Kimono$ video sequence of the proposed encryption scheme with SNR scalability and QP_{EL}=22

6.1.7.3 Key sensitivity test

The testing scenario of the key sensitivity is described in section 2.2.3 based on equations that given in section 2.2.2. As it is mentioned before, the optimal values of the

Table 6.13: Edge differential ratio for Kimono and $PeopleOnStreet$ video sequences at QP_{EL}=22

Video	Scalability	SE-SHVC-BL	SE-SHVC-ALL	SE-SHVC-EL
Kimono	SNR	0.95	0.95	0.78
	2x	0.96	0.95	0.88
	1.5x	0.96	0.95	0.78
	HEVC	-	-	0.95
People-OnStreet	SNR	0.93	0.93	0.54
	2x	0.93	0.92	0.81
	HEVC	-	-	0.92

block encryption algorithms of the NPCR and the UACI are 99.61% and 33.46%, respectively [162]. While theoretically, in the selective encryption algorithms achieving these results is impossible.

Table 6.14: Average performance of Key sensitivity attack over all frames of $Kimono$ and $PeopleOnStreet$ video sequences: SE-SHVC-BL, QP_{EL}=22

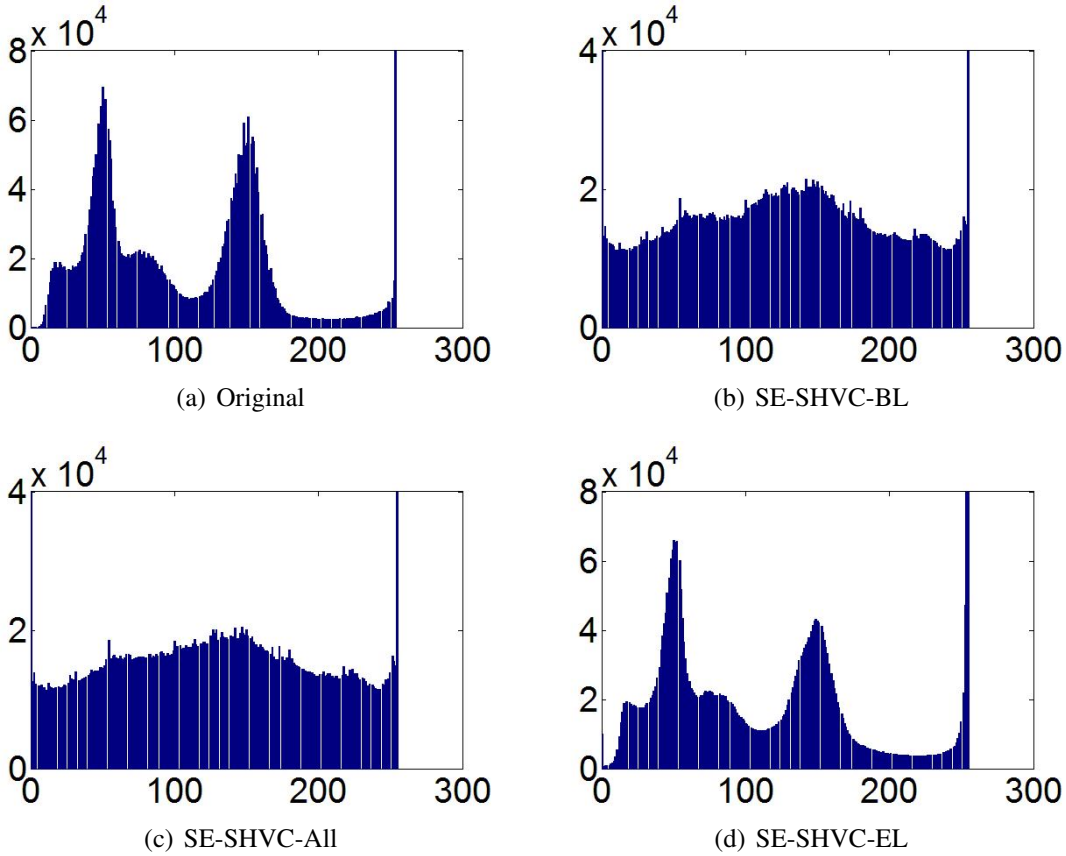
Sca.	$Kimono$		$PeopleOnStreet$	
	UACI	NPCR	UACI	NPCR
SNR	31.74	97.02	35.36	98.29
2x	36.75	98.98	39.34	98.37
HEVC	34.87	98.44	36.06	97.76

The obtained values of the UACI and the NPCR given in Table 6.14 by SE-SHVC-BL are not so far from the optimal. Moreover, the average Hamming Distance (HD) between the two encrypted frames are 49.69 and 50.87 for $Kimono$ and $PeopleOnStreet$, respectively. These HD values are too close to the optimal value of 50%. All these results indicate that the proposed chaos-based stream selective encryption is sensitive to the one bit change in the secret key.

6.1.7.4 Histogram analysis

Figure 6.8 shows the histograms of the frame #8 of the *PeopleOnStreet* video sequence at QP_{EL}=22 in SNR scalability for the proposed selective encryption scheme of all stages: In Figure 6.8(a), we present the histogram of the original frame; in Figures 6.8(b) and 6.8(c) we present the histograms of the frame encrypted by SH-SHVC-BL and SE-SHVC-All, respectively. As we can see, the previous histograms are more uniform and different from the original one. However, the histogram given in Figure 6.8(d) representing the encrypted frame by SH-SHVC-EL is not uniform since the total number of the encrypted

bits is smaller than the total number of the encrypted bits of the other stages. This result is compatible to the transparent encryption.



(a) Original (b) SE-SHVC-BL

(c) SE-SHVC-All (d) SE-SHVC-EL

Figure 6.8: Histograms of frame #8 $Kimono$ video sequence in the three encryption schemes with SNR scalability and QP_{EL}=22

6.1.7.5 Known plain-text attack

This kind of attack is described in details in chapter 2 for full encryption algorithm. When it is used to cryptanalysis the selective encryption algorithms for video, the Known plain-text attack is evaluated using PSNR and SSIM values over the whole frames of the tested video. For that, we calculate the average of the PSNR and SSIM for three cases:

1. PSNR1 and SSIM1 between the original video (the not compressed video) and the decoded video (without encryption).

2. PSNR2 and SSIM2 between the original video and the decoded hacking bit-stream, where the hacking bit-stream consists of replacing each encrypted bit by zero.

3. PSNR3 and SSIM3 between the original video and the decoded of the encrypted bit-stream (Tables (6.4-6.5) present the average over the both classes of the PSNR3 and SSIM3 respectively).

In Table 6.15, we present the average of PSNR1, SSIM1, PSNR2 and SSIM2 values (using the VQMT tools from [88]) over the whole frames of *Kimono* and *PeopleOnStreet* video sequences after replacing all encrypted bits by zeros for the BL.

Table 6.15: Average PSNR and SSIM for replacing encrypted bits by zero: QP_{EL}=22, SE-SHVC-BL

	Sca.	*Kimono*		*PeopleOnStreet*	
PSNR (dB)		PSNR1	PSNR2	PSNR1	PSNR2
	SNR	41.55	7.82	40.64	9.2
	2x	41.63	7.87	40.92	9.05
	1.5x	41.51	8.47	-	-
	HEVC	41.66	8.75	40.8	9.01
SSIM		SSIM1	SSIM2	SSIM1	SSIM2
	SNR	0.93	0.17	0.95	0.17
	2x	0.93	0.24	0.95	0.16
	1.5x	0.93	0.2	-	-
	HEVC	0.92	0.2	0.95	0.09

It is clear from Table 6.15, that the average of PSNR2 and SSIM2 values remain low compared to PSNR1 and SSIM1. This results confirm the robustness of the proposed schemes against this scenario of the known plain-text attacks.

6.1.7.6 Brute force attack

It breaks the cryptosystem by trying a large number of possible keys until the correct one is found. In the worst case, all possible keys in the practical key space are tested [125], [108]. Encryption of MVD and residual signs was classified by [107] to be secure selective encryption algorithms. In our proposed cryptosystem extra parameters as the sign of dQP together with suffixes of the MVD and residuals are increasing the complexity of the brute force attack.

6.1.7.7 Complexity analysis

In this section, we assess the computational complexity of the encryption scheme in the context of ***a real time SHVC decoder***. We use a computer fitted with an Intel Core i5 processor running at 2.5 GHz.
Table 6.16 presents the Decoding Time (DT) and the Complexity Overhead (CO) of the proposed encryption scheme, and Table 6.17 presents the results obtained by the AES algorithm in Counter (CTR) mode. The CO is defined by the difference between DT with encryption and DT without encryption over the DT without encryption.
The both implementations are carried out using 1080p50 *Cactus* video sequence at different QP configurations.

We can notice the high CO at high bitrate configuration (QP_{EL}=22) with respect to low bitrate configuration (QP_{EL}=34). This is mainly caused by the encryption of more syntax elements, including the complexity introduced by Algorithm 12 to safely encrypt the

Table 6.16: Decoding Time (DT) in second and Complexity Overhead (CO) in $\%$ of the proposed chaos-based SE solution for the $Cactus$ video sequence in different scalability and QP configurations

QP_{BL}-QP_{EL}	Scalability	DT	SE-SHVC-BL Chaotic		SE-SHVC-All Chaotic		SE-SHVC-EL Chaotic	
			DT	CO$\%$	DT	CO$\%$	DT	CO$\%$
26-22	SNR	19.25	19.88	3.27	20.01	3.94	19.88	3.27
22-22	2x	17.30	17.58	1.16	17.91	3.52	17.92	3.58
	1.5x	18.74	19.19	2.4	19.37	3.36	19.19	2.4
.-22	HEVC	13.53	-	-	-	-	14.24	5.2
30-26	SNR	12.19	12.57	3.11	12.32	1.06	12.35	1.31
26-26	2x	10.02	10.10	0.79	10.32	2.99	10.23	2.09
	1.5x	11.35	11.73	3.34	11.71	3.17	11.59	2.11
.-26	HEVC	7.14	-	-	-	-	7.24	1.4
38-34	SNR	7.63	7.73	1.31	7.99	4.71	7.63	0
34-34	2x	6.09	6.09	0	6.28	3.11	6.13	0.65
	1.5x	7.1	7.22	1.69	7.18	1.12	7.27	2.39
.-34	HEVC	4.07	-	-	-	-	4.13	1.47

Table 6.17: Decoding Time (DT) in second and Complexity Overhead (CO) in % of the proposed SE solution for the $Cactus$ video sequence in different scalability and QP configurations based on AES in CRT mode

QP_{BL}-QP_{EL}	Scalability	DT	SE-SHVC-BL AES		SE-SHVC-All AES		SE-SHVC-EL AES	
			DT	CO%	DT	CO%	DT	CO%
26-22	SNR	19.25	19.71	2.38	20.02	4	19.97	3.74
22-22	2x	17.30	17.53	1.32	17.95	3.75	17.84	3.12
	1.5x	18.74	19.20	2.45	19.48	3.94	19.33	3.14
.-22	HEVC	13.53	-	-	-	-	14.06	3.9
30-26	SNR	12.19	12.28	0.73	12.38	1.55	12.67	3.93
26-26	2x	10.02	10.57	5.48	10.33	3.09	10.31	2.89
	1.5x	11.35	11.79	3.87	12	5.72	11.63	2.46
.-26	HEVC	7.14	-	-	-	-	7.27	1.82
38-34	SNR	7.63	8	4.84	7.69	0.78	7.76	1.7
34-34	2x	6.09	6.2	1.8	6.18	1.47	6.34	4.1
	1.5x	7.1	7.25	2.11	7.32	3.09	7.48	5.35
.-34	HEVC	4.07	-	-	-	-	4.18	2.7

TCs, which are increasing at high bitrate. The encryption scheme SE-SHVC-BL enables the lowest CO since only syntax elements of the BL are encrypted; and more especially in spatial scalability configurations where the resolution of the BL is lower than the EL resolution. On the other hand, the CO using either chaotic or AES encryption systems remains low in the context of real time SHVC decoder and varies between 0% to 5% according to: the bitrate, the scalability configuration and the used encryption scheme. This CO performance is obtained thanks to the low EB of the proposed selective encryption solution where less than 17% of the whole video size is encrypted. The maximum bitrate in Table 6.11 of the encryptable bits for the high resolution 2Kp30 *Traffic* video sequence are 31.63 Mbit/s and 28.66 Mbit/s in 2x and SNR scalability configurations, respectively, with the encryption scheme SE-SHVC-All at high bitrate (QP_{EL}=22). These bitrates remain very low with respect to the Chaotic and AES generators bitrates which are equal to 823.9 Mbit/s and 888 Mbit/s [3], respectively. The low EB not only decreases the complexity of the encryption/decryption (which is convenient for battery-operated devices) but also decreases the end-to-end delay for live and real time video streaming applications.
Table 6.18 compares the SE-SHVC-BL and SE-SHVC-EL encryption schemes with the state of the art on different video encryption criteria. The proposed schemes fulfill format compliance and constant bitrate video encryption requirements. Moreover, the proposed encryption scheme is compatible with any stream cipher encryption algorithm. The SE-SHVC-BL and SE-SHVC-EL enable secured and perceptual video encryption, respectively; and can be applied on different SHVC scalability configurations: temporal, spatial, fidelity, bit depth and color gamut. In respect to transcoding capability, the SE-HEVC-BL scheme is robust to transcoding on all ELs while the SE-HEVC-EL encryption scheme is robust for BL transcoding. Finally, the SE-HEVC-EL encryption scheme can also be applied to the single layer HEVC standard corresponding to simulcast HEVC configuration.

[3]This performance is obtained on a Core-i5-4300M CPU @ 2.6 GHz, using the Cryptopp [91] implementation of the AES encryption algorithm

Table 6.18: Comparison of the proposed encryption scheme for (SE-SHVC-BL and SE-SHVC-EL) with the state of the art

SE schemes	Format compliance	Constant bitrate	Encryption algorithm	Encryption domain	Robust to transcoding	Video standard	Support scalability
Li et al. [67]	No	Yes	Leak Extraction (Stream cipher)	NAL	No	H.264/SVC	Yes
Carrillo et al. [19]	Yes	No	Pseudo random pixel permutation	Pixel	Yes	Independent	ROI
Van et al. [147]	Yes	No	AES as stream cipher	Transform	No	HEVC	No
Shahed et al. [116]	Yes	Yes	AES (block cipher)	Binstrings	No	HEVC	No
SE-SHVC-BL	Yes	Yes	Chaotic (stream cipher)	Binstrings	Yes for ELs	SHVC	Yes
SE-SHVC-EL	Yes	Yes	Chaotic (stream cipher)	Binstrings	Yes for BL	HEVC & SHVC	Yes

6.2 Encryption of ROI in HEVC

In this section, we investigate the protection of Region Of Interest (ROI), (privacy) for the HEVC standard based on the tile concept. Tiles in the HEVC enable the video to be split into independent rectangular regions. Two solutions are proposed to encrypt the tiles containing the Region Of Interest (ROI). The first solution encrypts the whole bit-stream of the ROI. The second solution enables a selective encryption for the encryptable bits (verifying constant bitrate and format compliant requirements) of the ROI tiles. The HEVC standard was designed with a particular attention to complexity, where several steps can be easily performed in parallel. Three high level parallel processing approaches, including independent slice, tile, and wavefront, can be used in the HEVC to simultaneously process multiple regions of a single picture [27]. The tile concept splits the picture into rectangular groups of CTBs, called tiles. The ROI is included in a set of Tiles (called ROI tiles) and the background is within the rest of non ROI tiles. Tiles break the CABAC and the intra prediction dependencies that each tile can be independently processed. Figure 6.9 illustrates a HEVC picture composed of 15 tiles (red rectangles), where each tile consists of 9 CTBs. These tiles represent independent and contiguous regions of the HEVC frame. Figure 6.10 illustrates the same idea composed of 4 tiles.

Figure 6.9: Tile concept in the HEVC standard composed of 15 tiles (red rectangles)

6.2.1 A brief state of the art

Usually the most important part of the video (ROI). In some special application, fast encryption algorithms are required to preserve the privacy of the moving objects in the video with a lower overhead charge. The ROI encryption can be treated in the same manner as the video encryption in the previous chapter (i.e., is it a NEA or Selective encryption, a format compliance encryption, a constant bit-rate encryption, a fast and a secure encryption, etc). Authors in [95] proposed a selective encryption solution of ROI based on Flexible Macroblock Ordering (FMO) concept in the H.264/AVC and chaos encryption system. The macroblocks of each frame are mapped into two slices, one regroups macroblocks within the ROI and the other slice regroups the macroblocks outside the ROI.

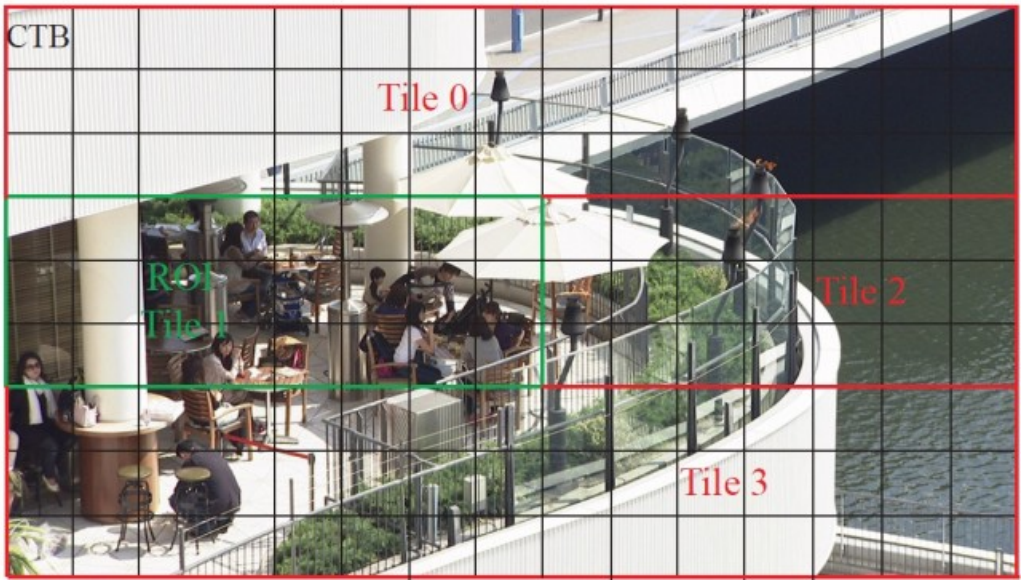


Figure 6.10: Tile concept in the HEVC standard composed of 4 tiles with ROI

Therefore, only the ROI slice is encrypted with a selective encryption solution based on chaos system. The ROI represents human faces which are detected in the video based on the skin color model [76].
Dufaux et al. [32] proposed a solution to hide ROI in MPEG-4 video content for privacy protection in video surveillance. This encryption solution is carried out at the transform domain by pseudo-randomly flipping the selected Transform Coefficient (TC) signs. To avoid the propagation of the encryption outside of the ROI, the macroblocks using the encrypted ROI as reference for inter prediction are rather Intra coded.
Work in [90] enables rectangular region privacy by de-identifying faces. This solution guarantees that, face recognition software cannot reliably recognize the de-identified faces even though if part of the facial details are preserved.
Authors in [130] investigated privacy protection in the H.264/SVC (Scalable Video Coding). This solution first detects face regions (ROI) and then encrypts these ROI in the transform domain by scrambling the sign of the non-zero TCs at all SVC layers.

6.2.2 AES in cipher feedback mode

The AES encryption system [40] is used in cipher feedback (CFB) mode to encrypt the HEVC syntax elements. As shown in Figure 6.11, the AES in CFB mode introduces internal diffusion inside the cipher block encryption and external diffusion by using the previous ciphered block as input for the encryption of the current plain block. The AES in CFB mode enables to produce the 128 bits (X_i) which can be than used to perform the selective encryption of the HEVC syntax element by syntax element on the fly and then send to the Bypass coding step. However, the size of the ciphertext C_i that is required for the encryption of the next block is 128 bits,

6.2.3 ROI encryption solutions in the HEVC

As mentioned before, we propose two solutions based on the tile concept to protect privacy in the HEVC standard. The first common step of the both solutions consists of the

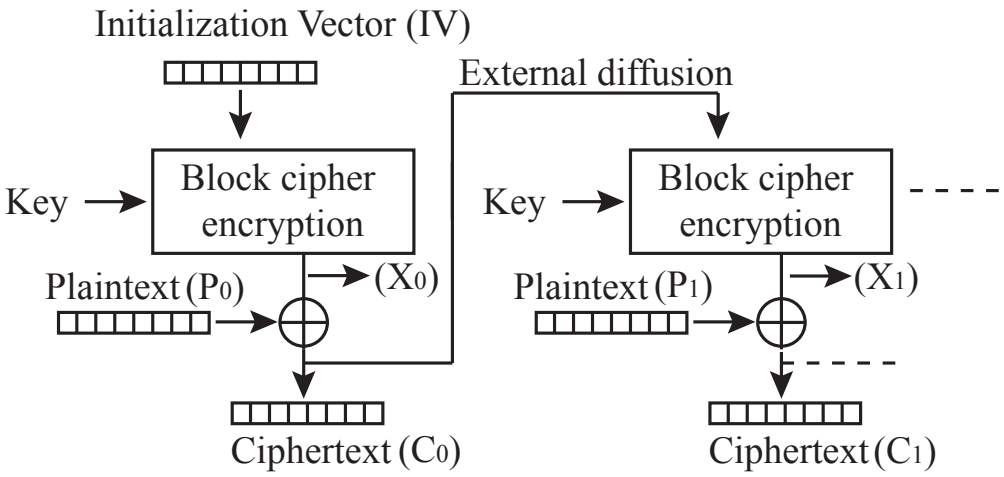


Figure 6.11: AES encryption system in CFB mode

identification and tracking of the ROI in the video. This can be done in real time by any existing algorithm such as face identification and tracking for video surveillance applications [76, 97]. The second common step uses information of ROI localization in the frame provided by the first step to split the HEVC frame into tiles where all ROI are included in ROI tiles and the background in separated non ROI tiles. In Figure 6.9, tiles 1, 2, 6 and 7 including human face represent the ROI tiles and the other tiles represent the background tiles. Note that the tile repartition provided in Figure 6.9 with red edges is not optimal in terms of coding rate-distortion performance. In fact, more efficient tile repartition can be defined to minimize the number of tiles in the frame where all ROI and background regions are included in a minimum number of tiles. For this example, the human face can be represented in one ROI tile which regroups only parts of tiles 1, 2, 6, 7 and the rest of tiles will represent the background. This repartition will provide more efficient coding performance since it reduces the number of tiles from 15 to only 6 tiles in the new repartition illustrated in Figure 6.9: ROI tile is highlighted in green dashed rectangle and non ROI tiles in blue dashed rectangles.

The first proposed encryption solution, called Tile Naive Encryption - HEVC (TNE-HEVC), encrypts all syntax elements coded in CABAC within the ROI tiles. The encryption is performed at the bit-stream level while the video headers, including VPS, SPS, PPS, and slice headers are not encrypted since they are not entropy coded with the CABAC.

The second proposed encryption solution, called Tile Selective Encryption - HEVC (TSE-HEVC), performs a selective encryption of ROI tiles in format compliant and at constant bitrate. The encryption is performed at the CABAC binstring level (see Figure 5.8) by encrypting only the most sensitive HEVC syntax elements to decrease the visual quality of the ROI.

6.2.3.1 TSE-HEVC EB

The TSE-HEVC solution encrypts only syntax element binarized in FL code and then bypassed. This restriction enables the TSE-HEVC solution to perform at the CABAC binstring level constant bitrate and format compliant encryption. The proposed algorithm (algorithm 12) is used to select the encryptable bits under the format compliance and the constant bit-rate. The selected syntax parameter for the encryption process inside the ROI

are:

- MVD suffix.
- MVD sign.
- TC sign.
- TC based on algorithm 12.

6.2.3.2 MV restriction in the background tile

In the two ROI encryption solutions, the decoding of the background tiles can use some information (inter layer prediction) of the encrypted tiles. Moreover, the encrypted MVs can also be used by the background tiles for inter layer prediction in the HEVC merge mode. Merge mode in the HEVC derives the MVs information from a list of spatial neighbor and temporal candidates [137]. Therefore, when of the ROI is not correctly decrypted, the two decoding operations can propagate the effect of the encrypted tiles to the background tiles. In the case of merge mode, we restrict the temporal candidates of the background tiles to be inside the background zone in the reference frame. In the case of regular inter prediction, the MVs differences are signaled. Therefore, we restrict the research window of the temporal candidates to be in the background tiles of the reference frames. Note that the boundary of the research window is equal to non ROI boundaries narrowed by 3 pixels and 1 pixel in luma and chroma components, respectively. This enables to perform a safe interpolation process at the tiles boundaries. Finally, the in-loop filters (deblocking and SAO) are disabled at the tiles boundaries.

6.2.3.3 Encryption process based on AES-CFB mode

In the TNE-HEVC solution, the bit-stream within the ROI tiles is encrypted block by block (P_i of 128 bits) as follows:

$$C_i = E_k(C_{i-1}) \oplus P_i \tag{6.5}$$

where $C_{-1} = IV$, C_i is the encrypted bits in the current ciphered block, C_{i-1} is the previous ciphered-block and E_k is the encryption function. The decryption at the decoder side is achieved with XOR operation in reverse using the same encryption algorithm:

$$P_i = E_k(C_{i-1}) \oplus C_i \tag{6.6}$$

$$c_j = x_j \oplus p_j \tag{6.7}$$

6.2.4 Results and analysis

6.2.4.1 Experimental configuration

The HEVC reference software model version 15 (HM 15.0) is used to encode and decode the video sequences. Encryption and decryption algorithms are integrated into the codec. We consider three HD video sequences (Kimono, ParkScene and BQTerrace) from the common HEVC text conditions [18]. These video sequences are coded in the low delay P-configuration at different quantization parameters $QP \in \{22, 27, 32, 37\}$ with enabling the tile tools.

6.2.4.2 Results

Figure 6.12 shows the average Rate Distortion (RD) performance of the three video sequences in three coding configurations: 1, 15 and 6 tiles with MVs limitations and disabling the in-loop filters across the tile edges. The RD loss of the two tiles configurations is provided in Table 6.19 in terms of Bjontegaard's difference (Matlab function). In the 6 tiles repartition, the RD loss, caused by these limitations, remains low and does not exceed 2.5% in the three video sequences. Figures 6.13(a) and 6.13(b) illustrate the visual quality

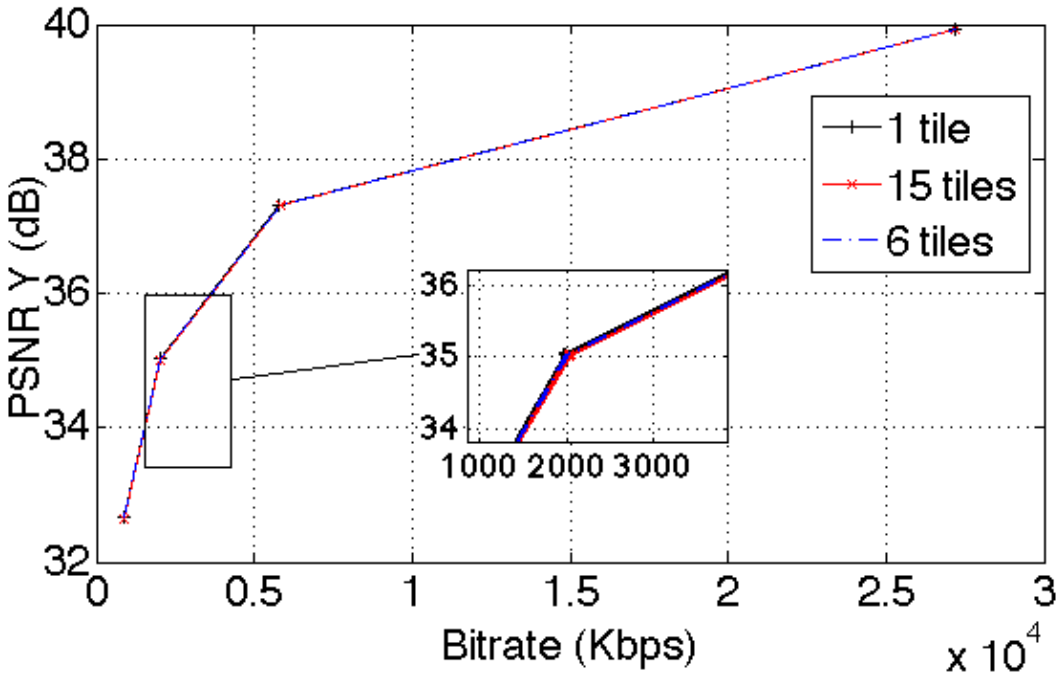


Figure 6.12: Rate distortion performance

of frame #1 of *Kimono* video sequence where the ROI is encrypted with the TNVE-HEVC and TSE-HEVC solutions, respectively. In both solutions, the visual quality is drastically decreased with a PSNR value of the ROI less than 11 dB, while the background region remains clear. However, the TNE-HEVC solution is not HEVC format compliant, thus

Table 6.19: Bjontegaard's difference of HEVC tile repartitions

Schemes	15 tiles per frame			Optimal (6 tiles)		
	Y	U	V	Y	U	V
Kimono	-3.27%	-3.26%	-3.03%	-1.49%	-1.63%	-1.56%
ParkScene	-1.41%	-2.34%	-2.07%	-0.51%	-1.21%	-1.09%
BQTerrace	-1.37%	-3.89%	-5.09%	- 0.5%	-1.64%	-2.54%

the encryption desynchronizes the decoder inside the ROI (the desynchronization of the CABAC is presented in the green part of the ROI in Figure 6.13(a)). The decoder is resynchronized at the next not encrypted tile thanks to the access points at the non encrypted slice header signaling tiles position in the bit-stream.

(a) TNE-HEVC PSNR1=10.95, PSNR2=42.07, PSNR3= 21.15(dB)

(b) TSE-HEVC PSNR1=7.78, PSNR2=42.07, PSNR3=18.01 (dB)

Figure 6.13: Visual quality of frame #1 in the $Kimono$ video sequence $QP = 27$ (PSNR1: PSNR Y ROI, PSNR2: PSNR Y background, PSNR3: PSNR Y frame).

Table 6.20 gives the average PSNR Y and the EB of the three video sequences. The average PSNR inside the ROI remains low for all sequences and does not exceed 11.5 dB. The EB of the TNVE-HEVC solution on average is equal to 14.28% since all syntax elements in the ROI are encrypted while it represents on average only 2.5% in the TSE-HEVC solution.

Finally, face recognition in the *Kimono* video sequence is assessed with using Principal Components Analysis (PCA) algorithm [149] based on Mahalanobis Cosine (MAHCOS) distance using the first rank. This rank corresponds to the best match of the test image compared to the training one. The recognition rate decreases from 68% in the original video to 0.5% in the video encrypted with TSE-HEVC solution.

Table 6.20: EB and PSNR of the ROI encryption solutions

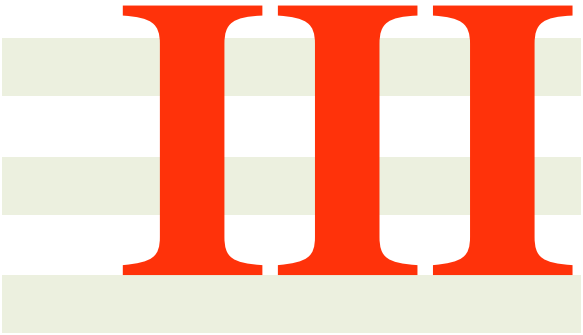
Schemes	TNVE-HEVC		TSE-HEVC		PSNR2
QP=22	PSNR 1 (dB)	ES (%)	PSNR1 (dB)	ES(%)	(dB)
Kimono	9.04	9.82	10.18	2.05	41.67
ParkScene	10.91	12.7	11.27	2.02	39.81
BQTerrace	10.79	20.32	10.89	3.42	38.72

6.3 Conclusion and discussion

In the first section, we have investigated a selective video encryption based on chaotic system in the scalable HEVC extension. A selective encryption solution is proposed to encrypt the most sensitive syntax elements in the SHVC bit-stream at the CABAC bin string level. This encryption solution is SHVC format compliant and does not impact the SHVC compression ratio. The encryption is applied on the SHVC bit-stream in three different configurations: encryption of only the BL, encryption of all layers and encryption of only the highest EL resulting in three encryption stages SE-SHVC-BL, SE-SHVCAll

and SE-SHVC-EL, respectively. The first two stages enable a high security level with drastically decreasing the visual quality of the video, while the third stage performs perceptual video encryption. The three stages preserves all SHVC functionalities including bit-stream extraction and error resilience. This enables the untrusted milde-box to perform network adaptation on the bit-stream and further decrease the end-to-end delay. The EB of SE-SHVC-BL stage remains low and does not exceed 8% of the whole SHVC bit-stream and this stage also passes all security encryption tests. In terms of computational complexity, the SE-SHVC-BL encryption stage introduces a low complexity overhead, which remains lower than 3% of the whole SHVC decoding time of HD video sequences at high bitrate.

In the second section, two encryption solutions are proposed based on the HEVC tile concept and the AES algorithm in CFB mode to protect privacy in the HEVC video content. The first solution performs encryption at the bit-stream level while the second solution carries out format compliant encryption at the CABAC binstring level. Restrictions are introduced in the HEVC coding process to prevent the propagation of the encryption outside the ROI region but at the expense of rate-distortion loss. Experimental results showed that both solutions perform a secure protection of privacy in the HEVC video content; and the TSE-HEVC solution has a low EB and prevents unexpected behavior of the decoder while TNE-HEVC is not format compliant solution.



Conclusion and Future Works

7

Conclusion and Future Work

In this thesis, we mainly study the problem of achieving the confidentiality of transmitted images and videos over public channels, by using chaos-based cryptosystems and joint crypto-compression systems. These systems are designed and implemented for real-time applications with a very high security level.
In chapter one, we introduced the context of the study, the main goals of the cryptography, the concept of the chaos-based cryptography, the motivation of this study, and the thesis contributions.
In chapter 2, first, we presented in a brief manner the main recent chaos-based cryptosystems of the literature that have a direct relation to our contributions. Then, we introduced some metrics and measurement tools, which permit to quantify the performance of the proposed crypto and joint crypto-compression systems in terms of robustness against known and statistical attacks, and in terms of average encryption/decryption times, encryption throughput, and the needed number of cycles to encrypt/decrypt one byte.
The main contributions of the thesis are described in two parts.
In the first part (chapter 3 and chapter 4), we designed, implemented, and tested four efficient chaos-based cryptosystems, defined on finite numbers, for real-time applications of image and video transmissions. All of them are blocks ciphers and the two first cryptosystems (crypto A and crypto B) are based on the substitution-permutation network (SPN). We proposed a modified Finite Skew Tent Map (FSTM) that permitted to overcome the problems of the classic FSTM, namely: fixed point, restriction of the key space and limitation of mapping between plaintext and ciphertext and vice versa. The structure of the proposed third cryptosystem (crypto C) is new and efficient. It is based on two chaotic layers: a binary diffusion layer of pixels, followed by a bit-permutation layer. We proposed a new formulation of the 2-D cat map that allows an efficient implementation in C code of the permutation process. The fourth cryptosystem (chapter 4), which is faster than the other cryptosystems with a very high security level, is implemented based on a partial cryptanalysis that we performed for one of best chaos-based cryptosystems, recently published by Zhang in 2013. In this cryptosystem, the confusion process (based on the modified 2D-cat map) and the diffusion process (based on the modified FSTM as a generator) are performed simultaneously in a single scan of plain-image pixels.
In chapter 5 of the second part, we have introduced the main notions of the selective en-

cryption of the High Efficiency Video Coding (HEVC) and its scalable version (SHVC). Then, in chapter 6, we realized two fast and secure selective chaos-based crypto compression systems to encrypt and secure the HEVC and its scalable version SHVC. In the first crypto-compression, we proposed a new algorithm to define the encrypt-able bit in the bit stream of the HEVC and the SHVC. The proposed solution encrypts a set of sensitive SHVC parameters with a minimum delay and complexity overheads. The encryption process is performed at the CABAC bin string level and fulfills both constant bit rate and format compliant video encryption requirements. It preserves all SHVC functionalities including bit stream extraction for mid-network adaptation, error resilience, and real-time SHVC decoder. The second proposed crypto-compression system protects the Region Of Interest (ROI) of the HEVC based on the tile concept. It performs encryption at the bit stream level by encrypting all HEVC syntax elements within the ROI tiles, or a selective encryption of the ROI tiles under constant bit rate and format compliant requirements. To avoid temporal propagation of the encryption outside the ROI boundaries caused by inter-prediction, the motion vectors of non ROI regions are restricted inside the non encrypted tiles in the reference frames.

Our future works will focus on:

- A prime reversible chaotic map (as FSTM) to be used as substitution and irreversible, i.e., to be used as generator.
- Designing a separated and selective joint encryption-compression system.
- Designing chaos-based stream ciphers.
- A joint compression and watermarking solution for the new generation of video coding systems (HEVC, SHVC, MVHEVC and 3D-HEVC).

Bibliography

[1] High Efficiency Video Coding. In *Rec. ITU-T H.265 and ISO/IEC 23008-2*. Sapporo, JP, January 2013. 138

[2] Ahmed A Abd El-Latif, Xiamu Niu, and Mohamed Amin. A new image cipher in time and frequency domains. *Optics Communications*, 2012. 17, 20, 60, 61, 62

[3] Ala Abu-Zahra, Zafar Shahid, Amjad Rattrout, and William Puech. Independent protection of different layers in spatially scalable video coding. *procedia computer science*, 10:240–246, 2012. 132

[4] Mohammed AbuTaha, Mousa Farajallah, Radwan Tahboub, and Mohammad Odeh. Survey paper: cryptography is the science of information security. *International Journal of Computer Science and Security (IJCSS)*, 5(3):298, 2011. 15

[5] Iskender Agi and Li Gong. An empirical study of secure mpeg video transmissions. In *Network and Distributed System Security, 1996., Proceedings of the Symposium on*, pages 137–144. IEEE, 1996. 126, 128

[6] H. E H Ahmed, H. M. Kalash, and O.S.F. Allah. Encryption Efficiency Analysis and Security Evaluation of RC6 Block Cipher for Digital Images. In *International Conference on Electrical Engineering (ICEE)*, pages 1–7, April 2007. 28

[7] Jinhaeng Ahn, Hiuk Jae Shim, Byeungwoo Jeon, and Inchoon Choi. Digital video scrambling method using intra prediction mode. In *Advances in Multimedia Information Processing-PCM 2004*, pages 386–393. Springer, 2005. 130

[8] A Akhshani, A Akhavan, S-C Lim, and Z Hassan. An image encryption scheme based on quantum logistic map. *Communications in Nonlinear Science and Numerical Simulation*, 17(12):4653–4661, 2012. 17, 20, 58, 60, 62, 98, 99, 100, 190

[9] Gonzalo Alvarez and Shujun Li. Some basic cryptographic requirements for chaos-based cryptosystems. *International Journal of Bifurcation and Chaos*, 16(08):2129–2151, 2006. 24

[10] Giuseppe Anastasi, Marco Conti, Mario Di Francesco, and Andrea Passarella. Energy conservation in wireless sensor networks: A survey. *Ad Hoc Networks*, 7(3):537–568, 2009. 106

[11] Ross Anderson. Security engineering: A guide to building dependable distributed systems. 2001, 2001. 73

[12] Mamoona N Asghar, Mohammed Ghanbari, Martin Fleury, and Martin J Reed. Confidentiality of a selectively encrypted h. 264 coded video bit-stream. *Journal of Visual Communication and Image Representation*, 25(2):487–498, 2014. 126

[13] M.N. Asghar, M. Ghanbari, and M.J. Reed. Sufficient encryption with codewords and bin-strings of h.264/svc. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, pages 443–450, June 2012. 131

[14] S Behnia, A Akhshani, H Mahmodi, and A Akhavan. A novel algorithm for image encryption based on mixture of chaotic maps. *Chaos, Solitons & Fractals*, 35(2):408–419, 2008. 17, 20, 60, 62, 190

[15] Bharat Bhargava, Changgui Shi, and Sheng-Yih Wang. MPEG video encryption algorithms. *Multimedia Tools and Applications*, 24(1):57–79, 2004. 17, 20, 129

[16] Gaurav Bhatnagar and QM Jonathan Wu. Selective image encryption based on pixels of interest and singular value decomposition. *Digital Signal Processing*, 22(4):648–663, 2012. 17, 20

[17] Eli Biham and Adi Shamir. Differential cryptanalysis of DES-like cryptosystems. *Journal of CRYPTOLOGY*, 4(1):3–72, 1991. 25

[18] Frank Bossen. Common Conditions and Software Reference Configurations. Document JCTVC-H1100, JCT-VC of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, San Jose, CA, February 2012. 167

[19] Paula Carrillo, Hari Kalva, and Spyros Magliveras. Compression Independent Reversible Encryption for Privacy in Video Surveillance. *EURASIP Journal on Information Security*, 2009(5):1–13, 2009. 163

[20] SGXV CCITT and H Recommendation. 261-video codec for audiovisual services at px64 kbit/s. *The International Telegraph and Telephone Consultative Committee*, 1990. 115

[21] Wei-Der Chang. Digital secure communication via chaotic systems. *Digital Signal Processing*, 19(4):693–699, 2009. 17, 20, 190

[22] Guanrong Chen, Yaobin Mao, and Charles K Chui. A symmetric image encryption scheme based on 3d chaotic cat maps. *Chaos, Solitons & Fractals*, 21(3):749–761, 2004. 17, 20, 21, 60, 61, 62

[23] J. Chen, J. Boyce, M. Hannuksela Y. Ye, G. J. Sullivan, and Y. K. Wang. HEVC Scalable Extensions (SHVC) Draft Text 7. In *Document JCTVCR100*. Sapporo, JP, July 2014. 136

[24] Houssine Chetto and Maryline Chetto. Some results of the earliest deadline scheduling algorithm. *IEEE Transactions on Software Engineering*, 15(10):1261–1269, 1989. 107

[25] Houssine Chetto and Maryline Chetto. An adaptive scheduling algorithm for fault-tolerant real-time systems. *Software Engineering Journal*, 6(3):93–100, 1991. 107

[26] Maryline Chetto and Audrey Queudet. A note on edf scheduling for real-time energy harvesting systems. *IEEE TRANSACTIONS ON COMPUTERS*, 63(4):1037–1040, 2013. 108

[27] C. C. Chi, M. Alvarez-Mesa, B. Juurlink, G. Clare, F. Henry, S. Pateux, and T. Schier. Parallel Scalability and Efficiency of HEVC Parallelization Approaches. *IEEE TCSVT*, 22:1827–1838, December 2012. 164

[28] Franco Chiaraluce, Lorenzo Ciccarelli, Ennio Gambi, Paola Pierleoni, and Maurizio Reginelli. A new chaotic algorithm for video encryption. *Consumer Electronics, IEEE Transactions on*, 48(4):838–844, 2002. 17, 20

[29] Cahit Cokal and Ercan Solak. Cryptanalysis of a chaos-based image encryption algorithm. *Physics Letters A*, 373(15):1357–1360, 2009. 24

[30] Wei Dai. Crypto++ 5.6.0 benchmarks. `http://www.cryptopp.com/benchmarks.html`. Accessed: 2015-04-08. 49

[31] Loïc Dubois, William Puech, Jacques Blanc-Talon, et al. Fast protection of h. 264/avc by reduced selective encryption of cavlc. In *EUSIPCO'11: 19th European Signal Processing Conference*, 2012. 131

[32] F. Dufaux and T. Ebrahimi. Scrambling for Privacy Protection in Video Surveillance Systems. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(8):1168–1174, 2008. 165

[33] Morris J Dworkin. Sp 800-38c. recommendation for block cipher modes of operation: the ccm mode for authentication and confidentiality. *NIST Special Publication 800-38C*, 2004. 134

[34] William F Ehrsam, Carl HW Meyer, John L Smith, and Walter L Tuchman. Message verification and transmission error detection by block chaining, February 14 1978. US Patent 4,074,066. 87

[35] Safwan El Assad and Mousa Farajallah. A simple introduction to the klt (karhunen—loève transform). *A New Efficient Structure of a Cryptosystem based on Two Chaotic Layers: a Binary Diffusion Layer and a Bit-Permutation Layer*, page 13 Pages, under submission. 31

[36] Safwan El Assad, Mousa Farajallah, and Calin Vladeanu. Chaos-based block ciphers: An overview. In *Communications (COMM), 2014 10th International Conference on*, pages 1–4. IEEE, 2014. 17, 20

[37] Safwan El Assad and Hassan Noura. Generator of chaotic sequences and corresponding generating system, March 28 2011. US Patent App. 13/638,126. 31, 32, 42, 86, 87

[38] Mousa Farajallah, Safwan El Assad, and Maryline Chetto. Dynamic adjustment of the chaos-based security in real-time energy harvesting sensors. In *Green Computing and Communications (GreenCom), 2013 IEEE and Internet of Things (iThings/CPSCom), IEEE International Conference on and IEEE Cyber, Physical and Social Computing*, pages 282–289. IEEE, 2013. 31, 88

[39] Mousa Farajallah, Zeinab Fawaz, Safwan El Assad, and Olivier Deforges. Efficient image encryption and authentication scheme based on chaotic sequences. In *SECURWARE 2013, The Seventh International Conference on Emerging Security Information, Systems and Technologies*, pages 150–155, 2013. 31, 88, 95

[40] N. Ferguson, B. Schneier, and T. Kohno. Cryptography Engineering: Design Principles and Practical Applications. In *Wiley Publishing Inc.,*. ISBN 978-0-470-47424-2, 2010. 165

[41] Niels Ferguson, Bruce Schneier, and Tadayoshi Kohno. *Cryptography Engineering: Design Principles and Practical Applications: Design Principles and Practical Applications*. John Wiley & Sons, 2011. 18

[42] PUB FIPS. 180-1. secure hash standard. *National Institute of Standards and Technology*, 17, 1995. 45

[43] A Fog. Lists of instruction latencies, throughputs and micro-operation breakdowns for intel, amd and via cpus. *Copenhagen University College of Engineering*, 2012. 55

[44] Jiri Fridrich. Image encryption based on chaotic maps. In *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*, volume 2, pages 1105–1110. IEEE, 1997. 17, 20, 190

[45] Jiri Fridrich. Symmetric ciphers based on two-dimensional chaotic maps. *International Journal of Bifurcation and Chaos*, 8(06):1259–1284, 1998. 17, 20, 41, 88

[46] B Furht and D Socek. Multimedia security: Encryption techniques. *IEC Comprehensive Report on Information Security*, pages 335–349, 2003. 17, 20

[47] Xin Ge, Fenlin Liu, Bin Lu, and Wei Wang. Cryptanalysis of a spatiotemporal chaotic image/video cryptosystem and its improved version. *Physics Letters A*, 375(5):908–913, 2011. 24

[48] Dan Grois, Detlev Marpe, Amit Mulayoff, Benaya Itzhaky, and Ofer Hadar. Performance comparison of h. 265/mpeg-hevc, vp9, and h. 264/mpeg-avc encoders. In *PCS 2013 30th Picture Coding Symposium, San Jose, California*, Dec 8-11, 2013. 116

[49] W. Hamidouche, M. Farajallah, M. Raulet, O. Déforges, and S. El Assad. Selective Video Encryption using Chaotic System in the SHVC extension. In *40^{th} IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brisbane Australia, April 2015. 17, 20

[50] Wassim Hamidouche, Mickael Raulet, and Olivier Deforges. Real time SHVC Decoder: Implementation and Complexity Analysis. In *IEEE International Conference on Image Processing (ICIP)*, October 2014. 145

[51] Fraunhofer Heinrich Hertz Institute. Shvc reference software model (shm). `https://hevc.hhi.fraunhofer.de/svn/svn_SHVCSoftware/`. 145

[52] Guo J.-I and Yen J.-C. A new chaotic key-based design for image encryption and decryption. In *Circuits and Systems, 2000. Proceedings. ISCAS 2000 Geneva. The 2000 IEEE International Symposium on*, volume 4, pages 49–52. IEEE, 2000. 17, 20

[53] Rajan L Joshi and Thomas R Fischer. Comparison of generalized gaussian and laplacian modeling in dct image coding. *Signal Processing Letters, IEEE*, 2(5):81–82, 1995. 156

[54] David Kahn. *The Codebreakers: The comprehensive history of secret communication from ancient times to the internet*. Simon and Schuster, 1996. 24

[55] Kunihiko Kaneko. Pattern dynamics in spatiotemporal chaos: Pattern selection, diffusion of defect and pattern competition intermettency. *Physica D: Nonlinear Phenomena*, 34(1):1–41, 1989. 22

[56] Lutful Karim, Alagan Anpalagan, Nidal Nasser, Jalal Almhana, and Isaac Woungang. Fault tolerant, energy efficient and secure clustering scheme for mobile machine-to-machine communications. *Transactions on Emerging Telecommunications Technologies*, 25(10):1028–1044, 2014. 24

[57] Ali Kassem, Hussein Al Haj Hassan, Youssef Harkouss, and Rima Assaf. Efficient neural chaotic generator for image encryption. *Digital Signal Processing*, 25:266–274, 2014. 20

[58] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography*. CRC Press, 2008. 46, 73

[59] Stephen H Kellert. *In the wake of chaos: Unpredictable order in dynamical systems*. University of Chicago press, 1994. 19

[60] Syed Ali Khayam. The discrete cosine transform (dct): theory and application. *Michigan State University*, 2003. 114

[61] IK Kim, K McCann, K Sugimoto, B Bross, WJ Han, and GJ Sullivan. High efficiency video coding (hevc) test model 10 (hm10) encoder description. In *Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, 12th Meeting: Geneva*, 2013. 125, 126, 138

[62] Il-Koo Kim, Junghye Min, Tammy Lee, Woo-Jin Han, and JeongHoon Park. Block partitioning structure in the hevc standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 22(12):1697–1706, 2012. 116

[63] Bon Wook Koo, Hwan Seok Jang, and Jung Hwan Song. On constructing of a 32× 32 binary matrix as a diffusion layer for a 256-bit block cipher. In *Information Security and Cryptology–ICISC 2006*, pages 51–64. Springer, 2006. 50

[64] J. Lainema, F. Bossen, Woo-Jin Han, Junghye Min, and K. Ugur. Intra coding of the hevc standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 22(12):1792–1801, Dec 2012. 118

[65] Bai-Ying Lei, Kwok-Tung Lo, and Jian Feng. Encryption techniques for h. 264 video. *The Handbook of MPEG Applications: Standards in Practice*, page 151, 2010. 130

[66] Michael S. Lewis-Beck. *Data analysis: An introduction*. Number 103. Sage, 1995. 27

[67] Chunhua Li, Xinxin Zhou, and Yuzhuo Zhong. NAL Level Encryption for Scalable Video Coding. *Advances in Multimedia Information Processing (PCM)*, 5353:496–505, 2008. 163

[68] Ruilin Li, Bing Sun, and Chao Li. Impossible differential cryptanalysis of spn ciphers. *Information Security, IET*, 5(2):111–120, 2011. 57

[69] Shujun Li, Guanrong Chen, and Xuan Zheng. Chaos-Based Encryption for Digital Image and Video. *Multimedia Encryption and Authentication Techniques and Applications*, page 129, 2006. 17, 20

[70] Shiguo Lian. *Multimedia content encryption: techniques and applications*. CRC Press, 2008. 16, 17, 18, 126, 189, 190

[71] Shiguo Lian, Zhongxuan Liu, Zhen Ren, and Zhiquan Wang. Selective video encryption based on advanced video coding. In *Advances in Multimedia Information Processing-PCM 2005*, pages 281–290. Springer, 2005. 131

[72] Shiguo Lian, Jinsheng Sun, Jinwei Wang, and Zhiquan Wang. A chaotic stream cipher and the usage in video protection. *Chaos, Solitons & Fractals*, 34(3):851–859, 2007. 42, 129

[73] Shiguo Lian, Jinsheng Sun, and Zhiquan Wang. A block cipher based on a suitable use of the chaotic standard map. *Chaos, Solitons & Fractals*, 26(1):117–129, 2005. 58, 59, 60

[74] Shiguo Lian, Jinsheng Sun, and Zhiquan Wang. Security analysis of a chaos-based image encryption algorithm. *Physica A: Statistical Mechanics and its Applications*, 351(2):645–661, 2005. 21, 24, 80, 93, 94

[75] Chung Laung Liu and James W Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the ACM (JACM)*, 20(1):46–61, 1973. 107

[76] E. N. Lorenz. Deterministic Nonperiodic Flow. *Journal of Atmospheric Sciences*, 20:130 – 141, 1963. 165, 166

[77] Claudio Maccone. A simple introduction to the klt (karhunen—loève transform). *Deep Space Flight and Communications: Exploiting the Sun as a Gravitational Lens*, pages 151–179, 2009. 114

[78] Enrico Magli, Marco Grangetto, and Gabriella Olmo. Transparent encryption techniques for h. 264/avc and h. 264/svc compressed video. *Signal Processing*, 91(5):1103–1114, 2011. 129

[79] Farhad Maleki, Ali Mohades, S Mehdi Hashemi, and Mohammad E Shiri. An image encryption system by cellular automata with memory. In *Availability, Reliability and Security, 2008. ARES 08. Third International Conference on*, pages 1266–1271. IEEE, 2008. 25

[80] Ismail Mansour, Gerard Chalhoub, and Bassem Bakhache. Evaluation of a fast symmetric cryptographic algorithm based on the chaos theory for wireless sensor networks. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, pages 913–919. IEEE, 2012. 17, 20

[81] Yaobin Mao, Guanrong Chen, and Shiguo Lian. A novel fast image encryption scheme based on 3D chaotic Baker maps. *International Journal of Bifurcation and Chaos*, 14(10):3613–3624, 2004. 24, 80

[82] Phyu Phyu Mar and Khin Maung Latt. New analysis methods on strict avalanche criterion of S-boxes. *World Academy of Science, Engineering and Technology*, 48:150–154, 2008. 25

[83] Naoki Masuda and Kazuyuki Aihara. Cryptosystems with discretized chaotic maps. *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on*, 49(1):28–40, 2002. 17, 20, 33, 95

[84] Naoki Masuda, Goce Jakimoski, Kazuyuki Aihara, and Ljupco Kocarev. Chaotic block ciphers: from theory to practical algorithms. *Circuits and Systems I: Regular Papers, IEEE Transactions on*, 53(6):1341–1352, 2006. 17, 20, 21, 33, 42, 95

[85] Alfred J Menezes, Paul C Van Oorschot, and Scott A Vanstone. *Handbook of applied cryptography*. CRC press, 1996. 15, 131

[86] Jürgen Meyer and Frank Gadegast. Security mechanisms for multimedia data with the example mpeg-1 video. *Project Description of SECMPEG, Technical University of Berlin, Germany*, 1995. 128

[87] Alistair Moffat, Radford M Neal, and Ian H Witten. Arithmetic coding revisited. *ACM Transactions on Information Systems (TOIS)*, 16(3):256–294, 1998. 129

[88] Lausanne Switzerland Multimedia Signal Processing Group, Swiss Federal Institute of Technology (EPFL). Vqmt: Video quality measurement tool. `http://mmspg.epfl.ch/vqmt`. 158

[89] Dhinaharan Nagamalai, Eric Renault, and Murugan Dhanuskodi. *Trends in Computer Science, Engineering and Information Technology: First International Conference, CCSEIT 2011, Tirunelveli, Tamil Nadu, India, September 23-25, 2011, Proceedings*, volume 204. Springer Science & Business Media, 2011. 19

[90] E. M. Newton, L. Sweeney, and B. Malin. Preserving Privacy by de-identifying Face Images. *IEEE Transactions on Knowledge and Data Engineering*, 17:232 – 243, February 2005. 165

[91] NIST and CSE-FIPS 140-2 level 1 conformance. Cryptopp: C++ software encryption library. `http://www.cryptopp.com`. Accessed: 2014-06-30. 162

[92] J Ohm, Gary J Sullivan, Heiko Schwarz, Thiow Keng Tan, and Thomas Wiegand. Comparison of the coding efficiency of video coding standards-including high efficiency video coding (hevc). *Circuits and Systems for Video Technology, IEEE Transactions on*, 22(12):1669–1684, 2012. 116

[93] Fatih Ozkaynak, Ahmet Bedri Ozer, and Sirma Yavuz. Cryptanalysis of a novel image encryption scheme based on improved hyperchaotic sequences. *Optics Communications*, 285(24):4946–4948, 2012. 24

[94] Narendra K Pareek, Vinod Patidar, and Krishan K Sud. Diffusion–substitution based gray image encryption scheme. *Digital Signal Processing*, 23(3):894–901, 2013. 26, 58, 59, 62, 98, 99, 100

[95] Fei Peng, Xiao wen Zhu, , and Min Long. An ROI Privacy Protection Scheme for H.264 Video Based on FMO and Chaos. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(10):1688–1699, October 2013. 164

[96] Fei Peng, Xiao-wen Zhu, and Min Long. A roi privacy protection scheme for h. 264 video based on fmo and chaos. *Information Forensics and Security, IEEE Transactions on*, 8(10):1688–1699, 2013. 17

[97] M. Qi, X. Chen, J. Jiang, and S. Zhan. Face Protection of H.264 Video Based on Detecting and Tracking. In *International Conference on Electronic Measurement and Instruments (ICEMI)*, pages 172 – 177, July 2007. 166

[98] Lintian Qiao and Klara Nahrstedt. Comparison of mpeg encryption algorithms. *Computers & Graphics*, 22(4):437–448, 1998. 128

[99] Julien Reichel, Gloria Menegaz, Marcus J Nadenau, and Murat Kunt. Integer wavelet transform for embedded lossy to lossless image compression. *Image Processing, IEEE Transactions on*, 10(3):383–392, 2001. 114

[100] Martin Řeřábek and Touradj Ebrahimi. Comparison of compression efficiency between hevc/h. 265 and vp9 based on subjective assessments. In *SPIE Optical Engineering+ Applications*, pages 92170U–92170U. International Society for Optics and Photonics, 2014. 116

[101] Rhouma Rhouma and Safya Belghith. Cryptanalysis of a new image encryption algorithm based on hyper-chaos. *Physics Letters A*, 372(38):5973–5978, 2008. 24

[102] Rhouma Rhouma, Ercan Solak, and Safya Belghith. Cryptanalysis of a new substitution–diffusion based image cipher. *Communications in Nonlinear Science and Numerical Simulation*, 15(7):1887–1892, 2010. 24

[103] Iain E Richardson. *The H. 264 advanced video compression standard*. John Wiley & Sons, 2011. 115

[104] Ronald L Rivest, Matt JB Robshaw, Ray Sidney, and Yiqun L Yin. The rc6 block cipher. In *in First Advanced Encryption Standard (AES) Conference*. Citeseer, 1998. 17

[105] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, and Elaine Barker. A statistical test suite for random and pseudorandom number generators for cryptographic applications. Technical report, DTIC Document, 2001. 43

[106] El Assad Safwan. *Master 2*. Nantes University, 2015. 16, 189

[107] A. Said. Measuring the Strength of Partial Encryption Schemes. In *IEEE International Conference on Image Processing (ICIP)*, volume 2, pages II–1126–9, September 2005. 159

[108] Bruce Schneier et al. Applied cryptography: protocols, algorithms, and source code in C, 1996. 16, 17, 72, 159

[109] Daniel Schonberg, Stark C Draper, Chuohao Yeo, and Kannan Ramchandran. Toward compression of encrypted images and video sequences. *Information Forensics and Security, IEEE Transactions on*, 3(4):749–762, 2008. 19

[110] Heiko Schwarz, Detlev Marpe, and Thomas Wiegand. Overview of the scalable video coding extension of the h. 264/avc standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 17(9):1103–1120, 2007. 116

[111] V. Seregin and Y. He. Common SHM test conditions and software reference configurations. In *document JCTVC-O1009*. Geneva, Switzerland, November 2013. 145

[112] Zafar Shahid, Marc Chaumont, and William Puech. Fast protection of h. 264/avc by selective encryption of cabac. In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pages 1038–1041. IEEE, 2009. 132

[113] Zafar Shahid, Marc Chaumont, and William Puech. Selective and scalable encryption of enhancement layers for dyadic scalable h. 264/avc by scrambling of scan patterns. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 1273–1276. IEEE, 2009. 132

[114] Zafar Shahid, Marc Chaumont, and William Puech. Selective encryption of c2dvlc of avs video coding standard for i & p frames. In *Multimedia and Expo (ICME), 2010 IEEE International Conference on*, pages 1655–1660. IEEE, 2010. 132

[115] Zafar Shahid, Marc Chaumont, and William Puech. Fast protection of h. 264/avc by selective encryption of cavlc and cabac for i and p frames. *Circuits and Systems for Video Technology, IEEE Transactions on*, 21(5):565–576, 2011. 131

[116] Zafar Shahid, Marc Chaumont, and William Puech. Fast protection of h. 264/avc by selective encryption of cavlc and cabac for i and p frames. *Circuits and Systems for Video Technology, IEEE Transactions on*, 21(5):565–576, 2011. 132, 163

[117] Zafar Shahid, Marc Chaumont, and William Puech. Considering the reconstruction loop for data hiding of intra-and inter-frames of h. 264/avc. *Signal, Image and Video Processing*, 7(1):75–93, 2013. 132

[118] Zafar Shahid, Marc Chaumont, William Puech, et al. Fast protection of h. 264/avc by selective encryption of cabac for i & p frames. In *EUSIPCO'09: European Signal Processing Conference*, 2009. 131

[119] Zafar Shahid, Marc Chaumont, William Puech, et al. Joint entropy coding and encryption in avs video codec. *Recent Trends in Image and Video Processin*, 2013. 132

[120] Zafar Shahid and William Puech. Investigating the structure preserving encryption of high efficiency video coding (hevc). In *IS&T/SPIE Electronic Imaging*, pages 86560N–86560N. International Society for Optics and Photonics, 2013. 132

[121] Zafar Shahid and William Puech. Visual protection of hevc video by selective encryption of cabac binstrings. *IEEE Transactions on Multimedia*, 16(1):24–36, 2014. 132, 133

[122] Zafar SHAHID, William Puech, and Marc Chaumont. Real-time selective encryption of avs for i & p frames. In *European Signal Processing Conference*. Aalborg University, 2010. 132

[123] Claude E Shannon. Communication theory of secrecy systems. *Bell system technical journal*, 28(4):656–715, 1949. 20, 40

[124] Riabtsev Shevach. Detailed overview of hevc/h.265. 2013. 120, 121, 205, 208

[125] Simon Singh. *The code book: the science of secrecy from ancient Egypt to quantum cryptography*. Random House Digital, Inc., 2011. 72, 159

[126] R. Sjoberg, Y. Chen, A. Fujibayashi, M. M. Hannuksela, J. Samuelsson, T. K. Tan, Y.-K. Wang, and S. Wenger. Overview of HEVC High-Level Syntax and Reference Picture Management. *IEEE Transactions on Circuits and Systems for Video Technology*, 22:1969–1684, December 2012. 144

[127] snipview. H.264/avc. `http://www.snipview.com/q/H.264/AVC`. Accessed: 2015-03-25. 115, 207

[128] Daniel Socek, Shujun Li, Spyros S Magliveras, and Borko Furht. Short paper: Enhanced 1-d chaotic key-based algorithm for image encryption. In *Security and Privacy for Emerging Areas in Communications Networks, 2005. SecureComm 2005. First International Conference on*, pages 406–407. IEEE, 2005. 58

[129] Daniel Socek, Shujun Li, Spyros S.Maglivera, and Borko Furht. Short Paper: Enhanced 1-D Chaotic Key Based Algorithm for Image Encryption, Sep. 2005. 17, 20, 21, 59, 60

[130] H. Sohn, E.T. AnzaKu, W. De Neve, and Y. M. Ro. Privacy Protection in Video Surveillance Systems Using Scalable Video Coding. In *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 424 – 429, September 2009. 165

[131] Ercan Solak and Cahit Çokal. Algebraic break of image ciphers based on discretized chaotic map lattices. *Information Sciences*, 181(1):227–233, 2011. 24

[132] Ercan Solak, Cahit Çokal, Olcay Taner Yildiz, and Türker Biyikoğlu. Cryptanalysis of Fridrich's chaotic image encryption. *International Journal of Bifurcation and Chaos*, 20(05):1405–1413, 2010. 21, 24

[133] Chun-Yan Song, Yu-Long Qiao, and Xing-Zhou Zhang. An image encryption scheme based on new spatiotemporal chaos. *Optik-International Journal for Light and Electron Optics*, 2012. 22, 27, 60, 61, 62, 100

[134] George A Spanos and Tracy Bradley Maples. Performance study of a selective encryption scheme for the security of networked, real-time video. In *icccn*, page 0002. Published by the IEEE Computer Society, 1995. 128

[135] Thomas Stutz and Andreas Uhl. A survey of h. 264 avc/svc encryption. *Circuits and Systems for Video Technology, IEEE Transactions on*, 22(3):325–339, 2012. 131

[136] G. J. Sullivan, J. M. Boyce, Y. Chen, J. R. Ohm, and A. Vetro. Standardized Extensions of High Efficiency Video Coding (HEVC). *IEEE Journal of Selected Topics in Signal Processing*, 7(6):1001–1016, 2003. 136

[137] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand. Overview of the high efficiency video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22:1648–1667, December 2012. 136, 167

[138] Gary J Sullivan, Jens Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (hevc) standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 22(12):1649–1668, 2012. 116, 117, 122, 123, 207

[139] GJ Sullivan and JR Ohm. Meeting report of the 13th meeting of the joint collaborative team on video coding (jct-vc), incheon, kr. doc. jctvc-m1000. In *13th meeting: Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG*, volume 16, 2013. 114

[140] Nidhi Taneja, Balasubramanian Raman, and Indra Gupta. Chaos Based Partial Encryption of SPIHT Compressed Images. *International Journal of Wavelets, Multiresolution and Information Processing*, 9(02):317–331, 2011. 156

[141] Nidhi Taneja, Balasubramanian Raman, and Indra Gupta. Selective Image Encryption in Fractional Wavelet Domain. *AEU-International Journal of Electronics and Communications*, 65(4):338–344, 2011. 156

[142] Lei Tang. Methods for encrypting and decrypting mpeg video data efficiently. In *Proceedings of the fourth ACM international conference on Multimedia*, pages 219–229. ACM, 1997. 128

[143] Relu Laurentiu Tataru, Dalia Battikh, S El Assad, Hassan Noura, and Olivier Déforges. Enhanced adaptive data hiding in spatial lsb domain by using chaotic sequences. In *Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), 2012 Eighth International Conference on*, pages 85–88. IEEE, 2012. 41, 51

[144] Lo'ai Tawalbeh, Moad Mowafi, and Walid Aljoby. Use of elliptic curve cryptography for multimedia encryption. *IET Information Security*, 7(2):67–74, 2013. 126

[145] Ali Saman Tosun and W-C Feng. Efficient multi-layer coding and encryption of mpeg video streams. In *Multimedia and Expo, 2000. ICME 2000. 2000 IEEE International Conference on*, volume 1, pages 119–122. IEEE, 2000. 128

[146] J Vahidi, M Gorji, and Iran Mazandaran. The confusion-diffusion image encryption algorithm with dynamical compound chaos. *Journal of Mathematics and Computer Science (JMCS)*, 9(4):451–457, 2014. 19

[147] Glenn Van Wallendael, Andras Boho, Jan De Cock, Adrian Munteanu, and Rik Van de Walle. Encryption for high efficiency video coding with video adaptation capabilities. *IEEE Transactions on Consumer Electronics*, 59(3):634–642, 2013. 132, 163

[148] Glenn Van Wallendael, Jan De Cock, Sebastiaan Van Leuven, Andras Boho, Peter Lambert, Bart Preneel, and Rik Van de Walle. Format-compliant encryption techniques for high efficiency video coding. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 4583–4587. IEEE, 15-18 Sept, 2013. 132

[149] V. Štruc and N. Pavešić. The complete gabor-fisher classifier for robust face recognition. *EURASIP Advances in Signal Processing*, 2010:26, 2010. 169

[150] Xingyuan Wang, Dapeng Luan, and Xuemei Bao. Cryptanalysis of an image encryption algorithm using chebyshev generator. *Digital Signal Processing*, 25:244–247, 2014. 20, 24, 26

[151] Yong Wang, G. Attebury, and B. Ramamurthy. A survey of security issues in wireless sensor networks. *Communications Surveys Tutorials, IEEE*, 8(2):2–23, Second 2006. 16, 106, 189

[152] Yong Wang, Kwok-Wo Wong, Xiaofeng Liao, and Guanrong Chen. A new chaos-based fast image encryption algorithm. *Applied soft computing*, 11(1):514–522, 2011. 22, 58, 98, 99, 100

[153] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra. Overview of the h. 264/avc video coding standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 13(7):560–576, 2003. 115

[154] Thomas Wiegand, Gary J. Sullivan, Gisle Bjontegaard, and Ajay Luthra. Overview of the H.264/AVC Video Coding Standard. *IEEE Transactions on Circuit and Systems for Video Technology*, 13(7):560–576, July 2003. 115

[155] Kwok-Wo Wong, Bernie Sin-Hung Kwok, and Wing-Shing Law. A fast image encryption scheme based on chaotic standard map. *Physics Letters A*, 372(15):2645–2652, 2008. 58, 59, 60, 62, 88, 98, 99, 100

[156] Chung-Ping Wu and C-CJ Kuo. Design of integrated multimedia compression and encryption systems. *Multimedia, IEEE Transactions on*, 7(5):828–839, 2005. 126

[157] Yue Wu, Joseph P Noonan, and Sos Agaian. NPCR and UACI randomness tests for image encryption. *Cyber Journals: Multidisciplinary Journals in Science and Technology, Journal of Selected Areas in Telecommunications (JSAT)*, pages 31–38, 2011. 25

[158] Yue Wu, Yicong Zhou, Joseph P Noonan, and Sos Agaian. Design of image cipher using latin squares. *Information Sciences*, 264:317–339, 2014. 62

[159] Huaqian Yang, Kwok-Wo Wong, Xiaofeng Liao, Wei Zhang, and Pengcheng Wei. A fast image encryption and authentication scheme based on chaotic maps. *Communications in Nonlinear Science and Numerical Simulation*, 15(11):3507–3517, 2010. 22, 58, 59, 60, 61, 62

[160] Siu-Kei Au Yeung, Shuyuan Zhu, and Bing Zeng. Partial video encryption based on alternating transforms. *Signal Processing Letters, IEEE*, 16(10):893–896, Oct 2009. 132

[161] Siu-Kei Au Yeung, Shuyuan Zhu, and Bing Zeng. Design of new unitary transforms for perceptual video encryption. *Circuits and Systems for Video Technology, IEEE Transactions on*, 21(9):1341–1345, 2011. 132

[162] Wu Yue, Noonan Joseph P, and Agaian Sos. NPCR and UACI randomness tests for image encryption. *Cyber Journals: Multidisciplinary Journals in Science and Technology, Journal of Selected Areas in Telecommunications (JSAT)*, pages 31–38, 2011. 157

[163] Wenjun Zeng and Shawmin Lei. Efficient frequency domain selective scrambling of digital video. *Multimedia, IEEE Transactions on*, 5(1):118–129, 2003. 129

[164] Li Zhang, Ying Chen, and Marta Karczewicz. Disparity vector construction method for 3d-hevc, March 13 2013. US Patent App. 13/802,344. 122

[165] Linhua Zhang, Xiaofeng Liao, and Xuebing Wang. An image encryption approach based on chaotic maps. *Chaos, Solitons & Fractals*, 24(3):759–765, 2005. 17, 20

[166] Wei Zhang, Kwok-wo Wong, Hai Yu, and Zhi-liang Zhu. An image encryption scheme using reverse 2-dimensional chaotic map and dependent diffusion. *Communications in Nonlinear Science and Numerical Simulation*, 2013. 23, 58, 59, 69, 70, 80, 87, 97, 98, 99, 100

[167] Liang Zhao, Avishek Adhikari, Di Xiao, and Kouichi Sakurai. On the security analysis of an image scrambling encryption of pixel bit and its improved scheme based on self-correlation encryption. *Communications in Nonlinear Science and Numerical Simulation*, 17(8):3303–3327, 2012. 61, 62

Appendix A: Synthèse des travaux réalisés

A.1 Crypto-systèmes basés chaos et systèmes de crypto-compression conjoints pour les images et les vidéos

La cryptologie est divisée en deux branches liées: la cryptographie et la cryptanalyse. Le cryptographe tente de trouver des techniques pour garantir le secret de messages transmis et parfois assurer aussi leur authenticité. A l'inverse, le cryptanalyste tente de casser le cryptosystème pour récupérer le message original, ou bien élabore un faux message et cherche à le faire passer comme authentique par le destinataire. En utilisant la cryptographie, plusieurs services (confidentialité, authentification de source, intégrité des données, non-répudiation, contrôle d'accès, signature numérique, etc.,) peuvent être, tous ou un certain nombre, réalisés en même temps dans une application donnée. Jusqu'aux années 1970, la cryptographie était le domaine exclusif des militaires et des gouvernements. De nos jours, avec le développement de la technologie tant informatique qu'Internet, nous assistons à l'explosion et à la mondialisation des échanges (Internet, courrier électronique, commerce électronique, ...). Des données multimédia: fichiers audio, images, et vidéos, sont fréquemment utilisées dans des domaines d'applications diverses: banques, commerce, éducation, santé, communications mobiles, réseaux de capteur sans fil, etc. De nos jours, un nombre croissant d'applications requièrent de la protection en temps réel sous contraintes (faible complexité, petite mémoire, ressources énergétiques limitées), et avec un niveau de sécurité élevé comme par exemple des données multimédia privées échangées par des dispositifs portables sur les réseaux mobiles et des données sensibles échangées entre réseaux de capteur sans fil. Concevoir et réaliser de cryptosystèmes capables de protéger (par chiffrement) des données multimédia (images et vidéos), tout en tenant compte des contraintes précédentes, est un grand défi et constitue l'objectif principal de ce travail [70, 151, 106].
Récemment, pour la protection des données, de nombreux travaux de recherche ont montré l'apport et l'intérêt d'utiliser des signaux chaotiques. En effet, des caractéristiques très intéressantes de ces séquences telles que bonnes propriétés cryptographiques, reproductibilité à l'identique, large bande, et surtout très grande sensibilité à la clé secrète, in-

citent à leur utilisation en cryptographie. Un nombre important de cryptosystèmes basés chaos ont ainsi été publiés récemment. La plupart d'entre eux sont basés sur la structure de Fridrich [44], qui intègre les propriétés de diffusion et de confusion définies par Shannon, [8, 14, 21]. Comparée à la cryptographie classique, la cryptographie basée chaos peut apporter quelques avantages en termes de robustesse et de rapidité, surtout en chiffrement symétrique par flux. De plus, la cryptographie basée chaos est plus flexibles, plus modulaire, et facile à mettre en œuvre, ce qui la rend appropriée pour le chiffrement des images et des vidéos.
Généralement, les algorithmes de chiffrement traitent différents volumes de données, et obtiennent ainsi des efficacités modulables en termes de degré de sécurité et de temps de calculs. Contrairement au chiffrement dit direct où tout le contenu multimédia compressé ou pas est chiffré, le chiffrement partiel opère seulement sur quelques données significatives du contenu multimédia compressé, les autres données étant laissées en clair. Dans un schéma de crypto-compression conjoint, l'opération de chiffrement est combinée avec l'opération de compression, avec une mise en œuvre simultanée. Le chiffrement partiel et le crypto-compression conjoint réduisent le volume de données à chiffrer, et permettent ainsi d'obtenir pour un niveau de sécurité exigé et pour une application donnée, une efficacité plus importante en terme calculatoire [70]. Le chiffrement vidéo scalable a pour objectif de fournir un contenu multimédia adaptable ou échelonnable (scalable).

A.2 Résumé des travaux

En résumé, dans cette thèse nous nous sommes intéressés à la problématique de la sécurité des images et des vidéos pour les applications qui nécessitent du temps réel et un haut niveau de sécurité. Dans ce contexte, dans la première partie de ce travail, quatre cryptosystèmes basés chaos flexibles, efficaces et très robustes contre la cryptanalyse, sont conçus et réalisés.
Les deux premiers s'appuient sur le réseau de substitution-permutation (SPN). La substitution est réalisée par une carte Skew tent (FSTM) modifiée pour surmonter différents problèmes : point fixe, restriction de la taille de la clé et limitation de la cartographie entre les textes d'origines et chiffrés. Le troisième cryptosystème est de structure nouvelle et également efficace. Il est basé sur une couche de diffusion binaire de pixels, suivi par une couche de permutation des bits. La permutation est réalisée par une nouvelle formulation efficace de la carte 2-D Cat.
Le quatrième cryptosystème est plus rapide que les autres avec un niveau de sécurité très élevé. Sa conception s'appuie sur une cryptanalyse partielle que nous avons réalisée, de l'algorithme de Zhang. Dans la deuxième partie de ce travail, deux crypto-compression basés chaos sélectifs et rapides sont réalisés pour sécuriser le flux HEVC et SHVC. Dans le premier crypto-compression, un nouvel algorithme pour définir les bits chiffrables dans le flux binaire du "High Efficiency Video Coding" (HEVC) et sa version "Scalable" (SHVC) est proposé. La solution proposée chiffre un ensemble de paramètres SHVC sensibles au niveau du codeur entropique (CABAC), tout en préservant l'ensemble des fonctionnalités SHVC.
Basé sur le concept de tuile, le deuxième crypto-compression proposé permet une protection de la vidéo au niveau d'une Région d'Intérêt (ROI) définie dans le standard HEVC.

A.3 Contributions

Dans la suite, nous résumons l'essentiel de notre contribution.
Première contribution : Equations améliorées de la substitution par la carte Skew tent, utilisées dans les crypto A et crypto B
La première contribution concerne la modification des équations de la substitution directe et de la substitution inverse, réalisées par la carte Skew tent. Par rapport aux équations de base, les améliorations apportées par les équations modifiées sont l'ajout d'un paramètre supplémentaire dans la clé secrète, permettant ainsi de résoudre le problème de la division par zéro, d'augmenter la taille de la clé secrète, et de réduire significativement la limitation de la cartographie entre les textes d'origines et chiffrés. Nous donnons et commentons ci dessous les équations de base et celles modifiées utilisées dans les crypto A et Crypto B.
Equations de bases : carte Skew tent
Les équations de base de la carte Skew tent sont rappelées ci dessous, réf 1. L'équation de substitution directe est donnée par:

$$Y = F_A(X) = \begin{cases} \left\lceil \frac{Q}{A} \times (X+1) \right\rceil - 1 & if\ 0 \le X < A \\ \\ \left\lfloor \frac{Q \times (Q-X-1)}{Q-A} \right\rfloor & if\ A \le X < Q \end{cases} \tag{1}$$

Avec $Q = 256$, $X,\ Y\ \in \{0,\ 1,\ 2,\ ...,\ Q-1\}$, and $A\ \in \{1,\ 2,\ ...,\ Q\}$.
A est le paramètre de la substitution, la "clé dynamique".
L'équation de substitution inverse est donnée par:

$$X = {F_A}^{-1}(Y) = \begin{cases} X_1 - 1 & if\ \ m(Y+1) = Y+1\ and\ \frac{X_1}{A} > \frac{Q-X_2}{Q-A} \\ X_2 - 1 & if\ \ m(Y+1) = Y+1\ and\ \frac{X_1}{A} \le \frac{Q-X_2}{Q-A} \\ X_1 - 1 & if\ \ m(Y+1) = Y+2 \end{cases} \tag{2}$$

Où:

$$X_1 = \left\lfloor \frac{A \times (Y+1)}{Q} \right\rfloor \tag{3}$$

$$X_2 = \left\lceil (\frac{A}{Q} - 1) \times (Y+1) + Q \right\rceil \tag{4}$$

$$m(Y+1) = Y + \left\lfloor \frac{A \times (Y+1)}{Q} \right\rfloor - \left\lfloor \frac{A \times (Y+1)}{Q} \right\rfloor + 2 \tag{5}$$

Commentaires des équations précédentes:

1. Dans le cas où $A = Q$, qui est une des valeurs possibles, il Y a une division par zéro dans le deuxième terme de l'équation (1).

2. Pour quelques valeurs de la sortie Y, il est facile de monter, que seules quelques valeurs de l'entrée X sont possibles. Exemples:

   - Pour avoir $Y = 0$, il existe seulement deux valeurs possibles pour l'entrée X, et ces valeurs sont:
     X = 0 ou X = 255. En effet : Quand $X < A$, alors le premier terme de

l'équation (1), $\left\lceil \frac{256}{A} \times X \right\rceil - 1 = 0$, cela signifie que $0 < \frac{256 \times X+1}{A} \leq 1$, et la seule solution possible est que le terme $\frac{X+1}{A}$ soit inférieur ou égale à $\frac{1}{256}$. Ceci peut se réaliser seulement pour $X = 0$ et $A = 256$.
Quand $A \leq X$, alors le second terme de l'équation (1), $\left\lfloor \frac{256 \times (256-X-1)}{256-A} \right\rfloor = 0$, cela signifie que $0 \leq \frac{256 \times (256-X-1)}{256-A} < 1$, et la seule solution possible est que le terme $\frac{256-X-1}{256-A}$ soit inférieur ou égale à $\frac{1}{256}$. Ceci peut se réaliser seulement pour $X = 255$, quelque soit A.

- Pour avoir $Y = 1$, quelque soit la valeur de A, nous pouvons montrer (en utilisant le même raisonnement que précédemment) que seules 3 valeurs différentes pour l'entrée X sont possibles, à savoir: $X = 0$ et $A = 256$, $X = 1$ et $A \in \{128,\ 129,\ \ldots,\ 256\}$ ou $X = 254$ et $A \in \{1,\ 2,\ \ldots,\ 127\}$. Et ainsi de suite, pour d'autres valeurs de Y.

3. Certaines valeurs de X ne produisent que quelques valeurs possibles de Y. Par exemple pour $X = 255$, les valeurs possibles de Y sont seulement, $Y = 0$ ou $Y = 255$. En effet:
Pour $X = 255$:

   - le premier terme de l'équation (1) est seulement valide pour $A = 256$, et donc la valeur du terme en question est $\left\lceil \frac{256 \times (255+1)}{256} \right\rceil - 1 = 255$.
   - le second terme de l'équation (1) est valide pour $A \leq X < Q$ et donc, la valeur du terme $\left\lfloor \frac{256 \times (256-255-1)}{256-A} \right\rfloor$ est 0, quelque soit la valeur de A. Et ainsi de suite, pour d'autres valeurs de X.

Equations modifiées proposées de la carte Skew tent Afin de résoudre les problèmes précédents causés par les équations de base, nous avons proposé, les équations modifiées de la carte Skew tent.

$$U = F_A(X) = \begin{cases} \left\lceil \frac{Q}{A} \times (X+1) \right\rceil \ Mod\, Q & if \quad 0 \leq X < A \\ \left\lfloor \frac{Q \times (Q-X)}{Q-A} \right\rfloor + 1 \ Mod\, Q & if \quad A \leq X < Q \end{cases} \tag{6}$$

Avec $X, U \in \{0, 1, 2, ...255\}$, $A \in \{1, 2, ...255\}$, et $Q = 256$.
Puis,

$$Y = (U + B)\ Mod\, Q \tag{7}$$

où B est comme A, un paramètre ajouté, faisant partie de la "clé dynamique", et qui est aussi fourni par le générateur chaotique. $B, Y \in \{0, 1, 2...255\}$.
L'équation de substitution proposée précédente, peut s'écrire:

$$Y = F_{A,\,B}(X) = \begin{cases} \left\lceil \frac{Q}{A} \times (X+1) \right\rceil + B \ Mod\, Q & if \quad 0 \leq X < A \\ \left\lfloor \frac{Q \times (Q-X)}{Q-A} \right\rfloor + 1 + B \ Mod\, Q & if \quad A \leq X < Q \end{cases} \tag{8}$$

L'équation de substitution inverse proposée s'écrit:

$$X = F_A^{-1}(U) = \begin{cases} A & if \quad U = 1 \\ \zeta_1 - 1 & if \quad \theta(U) = U + 1 \\ \zeta_1 - 1 & if \quad \theta(U) = U \; and \; \frac{\zeta_1}{A} > \frac{Q-\zeta_2}{Q-A} \\ \zeta_2 & if \quad \theta(U) = U \; and \; \frac{\zeta_1}{A} \leq \frac{Q-\zeta_2}{Q-A} \end{cases} \tag{9}$$

où:

$$U = (Y - B) \; Mod \, Q \tag{10}$$

Remarque: si $U = 0$, nous remplaçons dans les équations suivantes $U = Q$

$$\zeta_1 = \left\lfloor \frac{A}{Q} \times U \right\rfloor \tag{11}$$

$$\zeta_2 = \left\lceil (\frac{A}{Q} - 1) \times U + Q \right\rceil \tag{12}$$

$$\zeta_3 = \left\lceil \frac{A}{Q} \times U \right\rceil \tag{13}$$

$$\theta = U + \zeta_1 - \zeta_3 + 1 \tag{14}$$

Deuxième contribution: Crypto C, basé une matrice de diffusion binaire et une couche de permutation, réalisé par une nouvelle formulation de la carte 2D Cat pour une implantation optimale en code C

Le troisième cryptosystème conçu et réalisé, est basé sur une couche de diffusion binaire suivie par une couche de permutation des bits, au lieu d'une permutation des pixels (octets). La structure est nouvelle et efficace. En effet, la couche de permutation est réalisée par une nouvelle formulation proposée de la carte 2D cat, qui permet une mise en œuvre efficace, mesurée en termes d'opérations arithmétiques et de logique, mais aussi en termes de cycles d'horloge, en comparaison avec l'implantation de la carte 2D standard. Avec cette structure, le processus de diffusion, qui étale l'influence d'un bit sur les autres bits, devient plus rapide et plus efficace, mais au prix de plus de temps de calcul en comparaison avec la permutation des pixels. C_i dessous nous donnons l'équation de la carte 2D cat standard, incluant les paramètres r_x, r_y permettant de contourner le problème de la permutation du point $(0, 0)$, mais aussi d'augmenter l'espace de la clé dynamique formée par les paramètres dynamiques de la carte: $\{u, \; v, \; r_x \; et \; r_y\}$, et dont l'intervalle de variation de chacun d'eux est $[0, \; M - 1]$:

$$\begin{bmatrix} i_n \\ j_n \end{bmatrix} = Mod \left(\begin{bmatrix} 1 & u \\ v & 1 + u \times v \end{bmatrix} \begin{bmatrix} i \\ j \end{bmatrix} + \begin{bmatrix} r_x + r_y \\ r_y \end{bmatrix}, \begin{bmatrix} M \\ M \end{bmatrix} \right) \tag{15}$$

où (i, j) et (i_n, j_n) sont les positions originales et permutées des bits des données mises sous forme matricielle de taille $M \times M$.

La nouvelle formulation proposée de l'équation précédente, pour une implantation optimale en code C, est donné par:

$$xrow = mod(Mv(i) + UMvn(j), M) \tag{16}$$

$$ycol = mod(VMv(i) + UVMvn(j), M) \tag{17}$$

avec

$$Mv = [1,\ 2,\ 3,\ ...\ M]$$

$$UMvn = [u + r_x + r_y,\ 2u + r_x + r_y,\ 3u + r_x + r_y,\ ...,\ Mu + r_x + r_y]$$

$$VMv = v \times Mv$$

$$UVMvn = [uv + 1 + r_y,\ 2uv + 2 + r_y,\ 3uv + 3 + r_y,\ ...,\ Muv + M + r_y]$$

Troisième contribution : Cryptanalyse partielle du premier algorithme de Zhang et réalisation d'un cryptosystème encore plus efficace que les autres

Dans cette contribution, nous avons d'abord partiellement cryptanalysé un des meilleurs algorithmes de la littérature récente, à savoir le premier algorithme de Zhang. En se basant sur cette cryptanalyse, nous avons ensuite réalisé un cryptosystème (avec plusieurs variantes) qui est plus robuste contre les attaques cryptographiques que le premier algorithme de Zhang, et aussi plus rapide.

Partie 1 : Cryptanalyse partielle du premier algorithme de Zhang

L'algorithme de Zhang utilise une structure basée sur une permutation 2D inverse des pixels, suivie par une diffusion non linéaire, réalisée par une carte Logistique dont les échantillons sont quantifiés sur seulement 8 bits. Nous avons proposé une équation de cryptanalyse partielle permettant d'enlever l'effet de la diffusion de l'algorithme de Zhang. Suite à cette attaque, l'image obtenue n'est autre qu'une version permutée de l'image d'origine. Aussi, basée sur cette équation de cryptanalyse partielle, nous avons réalisé une attaque différentielle, qui a permis de réduire les valeurs moyennes des paramètres NPCR et UCAI.

Partie 2 : Réalisation d'un cryptosystème très efficace avec 3 variantes

La structure du cryptosystème proposé en mode CBC (voir figure ci-dessous) est composée d'une couche de permutation réalisée par la nouvelle formulation de la carte 2D cat, suivie par une couche de diffusion non linéaire réalisée en tant que générateur, selon la variante : soit par une carte Logistique discrétisée sur 32 bits, soit par une carte Skew tent discrétisée sur 8 bits (pour une implantation par une table de correspondance), soit par une carte Skew tent discrétisée sur 32 bits. Même si la structure semble similaire à celle proposée par Zhang, l'équation de chiffrement donnée ci dessous est néanmoins nettement plus robuste.

$$c_l(k_n) = LSB_8[y_l(k)] \tag{18}$$

$$y_l(k) = p_l(k) \oplus s_{l-1}(k) \oplus f(y_l(k-1)) \tag{19}$$

Avec:

$y_l(k)$ est une variable à 32 bits, $p_l(k)$ et $s_{l-1}(k)$ sont des variables à 8 bits, f est la fonction non linéaire qui peut être la carte Logistique ou la carte Skew tent.

La séquence $s_{l-1}(k)$ est donnée par:

$$s_{l-1}(k) = \begin{cases} iv(k) & \text{if } l = 0 \\ c_{l-1}(k) & \text{if } l > 0 \end{cases} \tag{20}$$

où

$IV = \{iv(0),\ iv(1),\ iv(2),\ ...,\ iv(b_{s-1})\}$ est le vecteur initial, fournit par le générateur de séquences chaotique. **Quatrième contribution: Chiffrement des flux vidéos HEVC et SHVC**

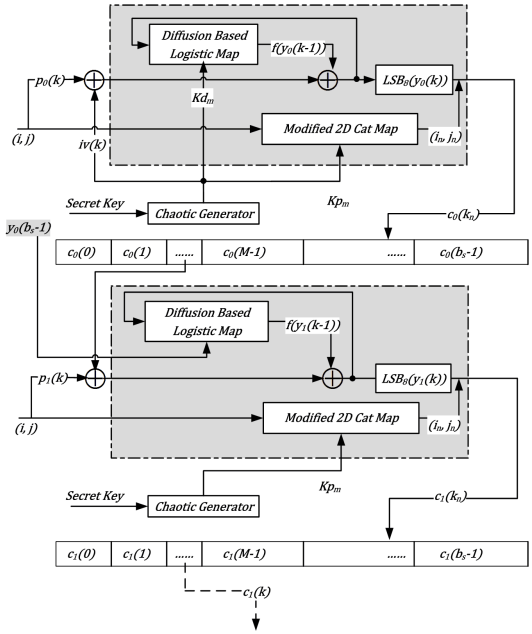


Figure 1: Encryption structure of the first proposed cryptosystem

Dans cette contribution, nous avons réalisé deux crypto-compression basés chaos, sélectifs, rapides, et robustes pour chiffrer les flux HEVC et sa version scalable SHVC. Les deux crypto-compression développés, satisfont les exigences suivantes:

- Conformité de Format: le chiffrement ne doit pas violer le décodeur, le flux chiffré des bits de la vidéo doit rester conforme aux normes du décodeur vidéo.
- Rapidité (latence faible) et sécurité: le chiffrement/déchiffrement ne dois pas augmenter la complexité du système avant chiffrement et assuré la protection des flux binaires.
- Peu d' impact de l'altération de taux de compression.
- Taux de compression constant: le chiffrement ne doit pas affecter le taux de compression vidéo original.
- Scalabilité maintenue du flux binaire de la vidéo d'origine.

Dans le premier crypto-compression, un nouvel algorithme pour définir les bits chiffrables dans le flux binaire de HEVC et du SHVC est proposé. La solution de base proposée de chiffre un ensemble de paramètres HVC sensibles au niveau du codeur entropique (CABAC), tout en préservant l'ensemble des fonctionnalités citées ci-dessus. Pour l'extension scalable SHVC , qui est considère généralement une couche de base (SE-SHVC-BL) et une couche de rehaussement (SE-SHVC-EL), trois scénarii ont été proposés pour répondre à différents objectifs applicatifs : le cryptage uniquement de la couche de base, le cryptage uniquement de la couche de rehaussement, le cryptage des deux couches. Basé sur le concept de tuile, le deuxième crypto-compression proposé permet aussi une protection de la vidéo au niveau d'une Région d'Intérêt (ROI) définie dans le standard HEVC. Il permet d'éviter la propagation du chiffrement à l'extérieur des frontières ROI.

Appendix B

Tables (1-5) present the the decimal value of residuals and their binarization format. The first column (named as 3) contains the $cAbsLevel$ value, where $cAbsLevel = baseLevel + Coef$, and the $baseLevel = 3$. The second column (named as 2) contains the $cAbsLevel$ value, where $cAbsLevel = baseLevel + Coef$, and the $baseLevel = 2$. The third column (named as 1) contains the $cAbsLevel$ value, where $cAbsLevel = baseLevel + Coef$, and the $baseLevel = 1$. Coef column contains the remaining value of the residuals. R contains the $cRiceParam$ value, P contains $2^{cRiceParam}$, Pre column represents the decimal form of the prefix part, BPre represents the binary form of the prefix part, Suf represents the decimal form of the suffix part, EGK represents the EGK value (if it is present), BSuf represents the binary form of the Trp suffix part (if it is present). Finally, Binary column represents the binary representation of the $cAbsLevel$ value (i.e., the residual). To illustrate the presented tables, some examples are given.

Example1: Assume the residual is: $cAbsLevel = 4$, $cRiceParam = 1$ and the $baseLevel = 2$, According to the TRp , EGk and algorithm-10:

$Coef = 4 - 2 = 2.$

$Pre = \lfloor \frac{2}{2} \rfloor = 1$

$BPre =$Unary binary of$(1) = 10$

$Suf = (2\ Mod\ 2) = 0$

The rice parameter is 1, and so the length (number of bits) of the suffix is 1 bit then:

$BSuf = FLC(0) = 0$

Since the BPre is not 1111, the EGK code is not exist.

The Binary representation of (4) is:100, to verify see line 3 of Table 1.

Example2: Assume the residual is: $cAbsLevel = 7$, $cRiceParam = 1$ and the $baseLevel = 1$, According to the TRp , EGk and algorithm 10:

$Coef = 7 - 1 = 6.$

$Pre = \lfloor \frac{6}{2} \rfloor = 3$

$BPre =$Unary binary of$(3) = 1110$

$Suf = (6\ Mod\ 2) = 0$

The rice parameter is 1, and so the length (number of bits) of the suffix is 1 bit then:

$BSuf = FLC(0) = 0$

Since the BPre is not 1111, the EGK code is not exist.
The Binary representation of(7) is:11100, to verify see line 7 of Table 1.

Example3: Assume the residual is: $cAbsLevel = 11$, $cRiceParam = 1$ and the $baseLevel = 3$, According to the TRp , EGk and algorithm 10:
$Coef = 11 - 3 = 8$.
$Pre = \lfloor \frac{8}{2} \rfloor = 4$
$BPre =$Unary binary of$(4) = 1111$
Since the BPre is 1111, as a result the EGK code is invoked with the remainder (in this case the remainder is 0).
EGK(0)=000.
The Binary representation of(11) is:1111000, to verify see line 9 of Table 1. Note in this case there is no TRp suffix.

Table 1: The binary of residuals for Rice=1

3	2	1	Coef	R	P	Pre	BPre	Suf	EGK	BSuf	Binary
3	2	1	0	1	2	0	0	0	NO	0	00
4	3	2	1	1	2	0	0	1	NO	1	01
5	4	3	2	1	2	1	10	0	NO	0	100
6	5	4	3	1	2	1	10	1	NO	1	101
7	6	5	4	1	2	2	110	0	NO	0	1100
8	7	6	5	1	2	2	110	1	NO	1	1101
9	8	7	6	1	2	3	1110	0	NO	0	11100
10	9	8	7	1	2	3	1110	1	NO	1	11101
11	10	9	8	1	2	4	1111	0	000	No	1111000
12	11	10	9	1	2	4	1111	1	001	No	1111001
13	12	11	10	1	2	5	1111	2	010	No	1111010
14	13	12	11	1	2	5	1111	3	011	No	1111011
15	14	13	12	1	2	6	1111	4	10000	No	111110000
16	15	14	13	1	2	6	1111	5	10001	No	111110001
17	16	15	14	1	2	7	1111	6	10010	No	111110010
18	17	16	15	1	2	7	1111	7	10011	No	111110011
19	18	17	16	1	2	8	1111	8	10100	No	111110100
20	19	18	17	1	2	8	1111	9	10101	No	111110101
21	20	19	18	1	2	9	1111	10	10110	No	111110110
22	21	20	19	1	2	9	1111	11	10111	No	111110111
23	22	21	20	1	2	10	1111	12	1100000	No	11111100000
24	23	22	21	1	2	10	1111	13	1100001	No	11111100001
25	24	23	22	1	2	11	1111	14	1100010	No	11111100010
26	25	24	23	1	2	11	1111	15	1100011	No	11111100011
27	26	25	24	1	2	12	1111	16	1100100	No	11111100100
28	27	26	25	1	2	12	1111	17	1100101	No	11111100101
29	28	27	26	1	2	13	1111	18	1100110	No	11111100110
30	29	28	27	1	2	13	1111	19	1100111	No	11111100111
31	30	29	28	1	2	14	1111	20	1101000	No	11111101000
32	31	30	29	1	2	14	1111	21	1101001	No	11111101001
33	32	31	30	1	2	15	1111	22	1101010	No	11111101010
34	33	32	31	1	2	15	1111	23	1101011	No	11111101011
35	34	33	32	1	2	16	1111	24	1101100	No	11111101100
36	35	34	33	1	2	16	1111	25	1101101	No	11111101101
37	36	35	34	1	2	17	1111	26	1101110	No	11111101110
38	37	36	35	1	2	17	1111	27	1101111	No	11111101111
39	38	37	36	1	2	18	1111	28	111000000	No	1111111000000
40	39	38	37	1	2	18	1111	29	111000001	No	1111111000001
41	40	39	38	1	2	19	1111	30	111000010	No	1111111000010
42	41	40	39	1	2	19	1111	31	111000011	No	1111111000011
43	42	41	40	1	2	20	1111	32	111000100	No	1111111000100

Table 2: The binary of residuals for Rice=2

3	2	1	V	R	P	Pre	BPre	Suf	EGK	BSuf	Binary
3	2	1	0	2	4	0	0	0	NO	00	000
4	3	2	1	2	4	0	0	1	NO	01	001
5	4	3	2	2	4	0	0	2	NO	10	010
6	5	4	3	2	4	0	0	3	NO	11	011
7	6	5	4	2	4	1	10	0	NO	00	1000
8	7	6	5	2	4	1	10	1	NO	01	1001
9	8	7	6	2	4	1	10	2	NO	10	1010
10	9	8	7	2	4	1	10	3	NO	11	1011
11	10	9	8	2	4	2	110	0	NO	00	11000
12	11	10	9	2	4	2	110	1	NO	01	11001
13	12	11	10	2	4	2	110	2	NO	10	11010
14	13	12	11	2	4	2	110	3	NO	11	11011
15	14	13	12	2	4	3	1110	0	NO	00	111000
16	15	14	13	2	4	3	1110	1	NO	01	111001
17	16	15	14	2	4	3	1110	2	NO	10	111010
18	17	16	15	2	4	3	1110	3	NO	11	111011
19	18	17	16	2	4	4	1111	0	0000	No	11110000
20	19	18	17	2	4	4	1111	1	0001	No	11110001
21	20	19	18	2	4	4	1111	2	0010	No	11110010
22	21	20	19	2	4	4	1111	3	0011	No	11110011
23	22	21	20	2	4	5	1111	4	0100	No	11110100
24	23	22	21	2	4	5	1111	5	0101	No	11110101
25	24	23	22	2	4	5	1111	6	0110	No	11110110
26	25	24	23	2	4	5	1111	7	0111	No	11110111
27	26	25	24	2	4	6	1111	8	100000	No	1111100000
28	27	26	25	2	4	6	1111	9	100001	No	1111100001
29	28	27	26	2	4	6	1111	10	100010	No	1111100010
30	29	28	27	2	4	6	1111	11	100011	No	1111100011
31	30	29	28	2	4	7	1111	12	100100	No	1111100100
32	31	30	29	2	4	7	1111	13	100101	No	1111100101
33	32	31	30	2	4	7	1111	14	100110	No	1111100110
34	33	32	31	2	4	7	1111	15	100111	No	1111100111
35	34	33	32	2	4	8	1111	16	101000	No	1111101000
36	35	34	33	2	4	8	1111	17	101001	No	1111101001
37	36	35	34	2	4	8	1111	18	101010	No	1111101010
38	37	36	35	2	4	8	1111	19	101011	No	1111101011
39	38	37	36	2	4	9	1111	20	101100	No	1111101100
40	39	38	37	2	4	9	1111	21	101101	No	1111101101
41	40	39	38	2	4	9	1111	22	101110	No	1111101110
42	41	40	39	2	4	9	1111	23	101111	No	1111101111
43	42	41	40	2	4	10	1111	24	11000000	No	111111000000

Table 3: The binary of residuals for Rice=3

3	2	1	V	R	P	Pre	BPre	Suf	EGK	BSuf	Binary
3	2	1	0	3	8	0	0	0	NO	000	0000
4	3	2	1	3	8	0	0	1	NO	001	0001
5	4	3	2	3	8	0	0	2	NO	010	0010
6	5	4	3	3	8	0	0	3	NO	011	0011
7	6	5	4	3	8	0	0	4	NO	100	0100
8	7	6	5	3	8	0	0	5	NO	101	0101
9	8	7	6	3	8	0	0	6	NO	110	0110
10	9	8	7	3	8	0	0	7	NO	111	0111
11	10	9	8	3	8	1	10	0	NO	000	10000
12	11	10	9	3	8	1	10	1	NO	001	10001
13	12	11	10	3	8	1	10	2	NO	010	10010
14	13	12	11	3	8	1	10	3	NO	011	10011
15	14	13	12	3	8	1	10	4	NO	100	10100
16	15	14	13	3	8	1	10	5	NO	101	10101
17	16	15	14	3	8	1	10	6	NO	110	10110
18	17	16	15	3	8	1	10	7	NO	111	10111
19	18	17	16	3	8	2	110	0	NO	000	110000
20	19	18	17	3	8	2	110	1	NO	001	110001
21	20	19	18	3	8	2	110	2	NO	010	110010
22	21	20	19	3	8	2	110	3	NO	011	110011
23	22	21	20	3	8	2	110	4	NO	100	110100
24	23	22	21	3	8	2	110	5	NO	101	110101
25	24	23	22	3	8	2	110	6	NO	110	110110
26	25	24	23	3	8	2	110	7	NO	111	110111
27	26	25	24	3	8	3	1110	0	NO	000	1110000
28	27	26	25	3	8	3	1110	1	NO	001	1110001
29	28	27	26	3	8	3	1110	2	NO	010	1110010
30	29	28	27	3	8	3	1110	3	NO	011	1110011
31	30	29	28	3	8	3	1110	4	NO	100	1110100
32	31	30	29	3	8	3	1110	5	NO	101	1110101
33	32	31	30	3	8	3	1110	6	NO	110	1110110
34	33	32	31	3	8	3	1110	7	NO	111	1110111
35	34	33	32	3	8	4	1111	0	00000	No	111100000
36	35	34	33	3	8	4	1111	1	00001	No	111100001
37	36	35	34	3	8	4	1111	2	00010	No	111100010
38	37	36	35	3	8	4	1111	3	00011	No	111100011
39	38	37	36	3	8	4	1111	4	00100	No	111100100
40	39	38	37	3	8	4	1111	5	00101	No	111100101
41	40	39	38	3	8	4	1111	6	00110	No	111100110
42	41	40	39	3	8	4	1111	7	00111	No	111100111
43	42	41	40	3	8	5	1111	8	01000	No	111101000

Table 4: The binary of residuals for Rice=4

3	2	1	V	R	P	Pre	BPre	Suf	EGK	BSuf	Binary
3	2	1	0	4	16	0	0	0	NO	0000	00000
4	3	2	1	4	16	0	0	1	NO	0001	00001
5	4	3	2	4	16	0	0	2	NO	0010	00010
6	5	4	3	4	16	0	0	3	NO	0011	00011
7	6	5	4	4	16	0	0	4	NO	0100	00100
8	7	6	5	4	16	0	0	5	NO	0101	00101
9	8	7	6	4	16	0	0	6	NO	0110	00110
10	9	8	7	4	16	0	0	7	NO	0111	00111
11	10	9	8	4	16	0	0	8	NO	1000	01000
12	11	10	9	4	16	0	0	9	NO	1001	01001
13	12	11	10	4	16	0	0	10	NO	1010	01010
14	13	12	11	4	16	0	0	11	NO	1011	01011
15	14	13	12	4	16	0	0	12	NO	1100	01100
16	15	14	13	4	16	0	0	13	NO	1101	01101
17	16	15	14	4	16	0	0	14	NO	1110	01110
18	17	16	15	4	16	0	0	15	NO	1111	01111
19	18	17	16	4	16	1	10	0	NO	0000	100000
20	19	18	17	4	16	1	10	1	NO	0001	100001
21	20	19	18	4	16	1	10	2	NO	0010	100010
22	21	20	19	4	16	1	10	3	NO	0011	100011
23	22	21	20	4	16	1	10	4	NO	0100	100100
24	23	22	21	4	16	1	10	5	NO	0101	100101
25	24	23	22	4	16	1	10	6	NO	0110	100110
26	25	24	23	4	16	1	10	7	NO	0111	100111
27	26	25	24	4	16	1	10	8	NO	1000	101000
28	27	26	25	4	16	1	10	9	NO	1001	101001
29	28	27	26	4	16	1	10	10	NO	1010	101010
30	29	28	27	4	16	1	10	11	NO	1011	101011
31	30	29	28	4	16	1	10	12	NO	1100	101100
32	31	30	29	4	16	1	10	13	NO	1101	101101
33	32	31	30	4	16	1	10	14	NO	1110	101110
34	33	32	31	4	16	1	10	15	NO	1111	101111
35	34	33	32	4	16	2	110	0	NO	0000	1100000
36	35	34	33	4	16	2	110	1	NO	0001	1100001
37	36	35	34	4	16	2	110	2	NO	0010	1100010
38	37	36	35	4	16	2	110	3	NO	0011	1100011
39	38	37	36	4	16	2	110	4	NO	0100	1100100
40	39	38	37	4	16	2	110	5	NO	0101	1100101
41	40	39	38	4	16	2	110	6	NO	0110	1100110
42	41	40	39	4	16	2	110	7	NO	0111	1100111
43	42	41	40	4	16	2	110	8	NO	1000	1101000
44	43	42	41	4	16	2	110	9	NO	1001	1101001
45	44	43	42	4	16	2	110	10	NO	1010	1101010
46	45	44	43	4	16	2	110	11	NO	1011	1101011
47	46	45	44	4	16	2	110	12	NO	1100	1101100
48	47	46	45	4	16	2	110	13	NO	1101	1101101
49	48	47	46	4	16	2	110	14	NO	1110	1101110
50	49	48	47	4	16	2	110	15	NO	1111	1101111

Table 5: The binary of residuals for Rice=4

3	2	1	V	R	P	Pre	BPre	Suf	EGK	BSuf	Binary
51	50	49	48	4	16	3	1110	0	NO	0000	11100000
52	51	50	49	4	16	3	1110	1	NO	0001	11100001
53	52	51	50	4	16	3	1110	2	NO	0010	11100010
54	53	52	51	4	16	3	1110	3	NO	0011	11100011
55	54	53	52	4	16	3	1110	4	NO	0100	11100100
56	55	54	53	4	16	3	1110	5	NO	0101	11100101
57	56	55	54	4	16	3	1110	6	NO	0110	11100110
58	57	56	55	4	16	3	1110	7	NO	0111	11100111
59	58	57	56	4	16	3	1110	8	NO	1000	11101000
60	59	58	57	4	16	3	1110	9	NO	1001	11101001
61	60	59	58	4	16	3	1110	10	NO	1010	11101010
62	61	60	59	4	16	3	1110	11	NO	1011	11101011
63	62	61	60	4	16	3	1110	12	NO	1100	11101100
64	63	62	61	4	16	3	1110	13	NO	1101	11101101
65	64	63	62	4	16	3	1110	14	NO	1110	11101110
66	65	64	63	4	16	3	1110	15	NO	1111	11101111
67	66	65	64	4	16	4	1111	0	000000	No	1111000000
68	67	66	65	4	16	4	1111	1	000001	No	1111000001
69	68	67	66	4	16	4	1111	2	000010	No	1111000010
70	69	68	67	4	16	4	1111	3	000011	No	1111000011
71	70	69	68	4	16	4	1111	4	000100	No	1111000100
72	71	70	69	4	16	4	1111	5	000101	No	1111000101
73	72	71	70	4	16	4	1111	6	000110	No	1111000110
74	73	72	71	4	16	4	1111	7	000111	No	1111000111
75	74	73	72	4	16	4	1111	8	001000	No	1111001000
76	75	74	73	4	16	4	1111	9	001001	No	1111001001
77	76	75	74	4	16	4	1111	10	001010	No	1111001010
78	77	76	75	4	16	4	1111	11	001011	No	1111001011
79	78	77	76	4	16	4	1111	12	001100	No	1111001100
80	79	78	77	4	16	4	1111	13	001101	No	1111001101
81	80	79	78	4	16	4	1111	14	001110	No	1111001110
82	81	80	79	4	16	4	1111	15	001111	No	1111001111
83	82	81	80	4	16	5	1111	16	010000	No	1111010000
84	83	82	81	4	16	5	1111	17	010001	No	1111010001
85	84	83	82	4	16	5	1111	18	010010	No	1111010010
86	85	84	83	4	16	5	1111	19	010011	No	1111010011
87	86	85	84	4	16	5	1111	20	010100	No	1111010100
88	87	86	85	4	16	5	1111	21	010101	No	1111010101
89	88	87	86	4	16	5	1111	22	010110	No	1111010110
90	89	88	87	4	16	5	1111	23	010111	No	1111010111
91	90	89	88	4	16	5	1111	24	011000	No	1111011000
92	91	90	89	4	16	5	1111	25	011001	No	1111011001
93	92	91	90	4	16	5	1111	26	011010	No	1111011010
94	93	92	91	4	16	5	1111	27	011011	No	1111011011
95	94	93	92	4	16	5	1111	28	011100	No	1111011100
96	95	94	93	4	16	5	1111	29	011101	No	1111011101

List of Tables

List of Figures

l'université nantes angers le mans
COMMUNAUTÉ D'UNIVERSITÉS ET ÉTABLISSEMENTS

UNIVERSITÉ DE NANTES

Thèse de Doctorat

Mousa FARAJALLAH

Crypto-systèmes basés chaos et systèmes de crypto-compression

Chaos-based crypto and joint crypto-compression systems for images and videos

Résumé
La sécurité des données images et vidéos est importante pour beaucoup d'applications qui exigent du temps réel et un haut niveau de sécurité. Dans la première partie de ce travail, quatre cryptosystèmes basés chaos flexibles, efficaces et très robustes contre la cryptanalyse sont conçus et réalisés. Les deux premiers s'appuient sur le réseau SPN. La substitution est réalisée par une carte Skew tent (FSTM) modifiée pour surmonter différents problèmes : point fixe, restriction de la taille de la clé et limitation de la cartographie entre les textes d'origines et chiffrés. Le troisième cryptosystème est de structure nouvelle et également efficace. Il est basé sur une couche de diffusion binaire de pixels, suivi par une couche de permutation des bits. La permutation est réalisée par une nouvelle formulation efficace de la carte 2-D Cat. Le quatrième cryptosystème, est plus rapide que les autres avec un niveau de sécurité très élevé. Sa conception s'appuie sur une cryptanalyse partielle, que nous avons réalisée, de l'algorithme de Zhang. Dans la deuxième partie, deux crypto-compression basés chaos sélectifs et rapides sont utilisés pour sécuriser le flux HEVC et SHVC. Dans le premier crypto-compression, un nouvel algorithme pour définir les bits chiffrables dans le flux binaire du HEVC et du SHVC est proposé. La solution proposée chiffre un ensemble de paramètres SHVC sensibles au niveau du codeur entropique (CABAC), tout en préservant l'ensemble des fonctionnalités SHVC. Basé sur le concept de tuile, le deuxième crypto-compression proposé permet une protection de la vidéo au niveau d'une Région d'Intérêt (ROI) définie dans le standard HEVC.

Mots clés
Cryptosystèmes basés chaos, Systèmes de crypto-compression conjoints, HEVC, SHVC, Chiffrement sélectif basé chaos, Performances

Abstract
The security of image and video data is important for many applications which require in real-time a high security level. In the first part of this work, four chaos-based cryptosystems, flexible, efficient, and more robust against cryptanalysis, are designed and realized. The first two cryptosystems are based on the substitution-permutation network. The substitution is achieved by a proposed modified Finite Skew Tent Map (FSTM) to overcome various problems: fixed point, key space restriction, and limitation of mapping between plaintext and ciphertext. The third cryptosystem is a new and efficient structure. It is based on a binary diffusion layer of pixels, followed by a bit-permutation layer. The permutation is achieved by an efficient proposed formulation of the 2-D cat map. The fourth cryptosystem is faster than the others, having a very high security level. The confusion and the diffusion are performed in a single scan. Its design is based on a partial cryptanalysis that we performed on the Zhang algorithm. In the second part, two fast and secure selective chaos-based crypto-compressions are designed and realized to secure the High Efficiency Video Coding (HEVC) and its scalable version. In the first crypto-compression, a new algorithm is proposed to define the encryptable bits in the bit stream of the HEVC and the SHVC systems. The proposed solution encrypts a set of sensitive SHVC parameters at the entropy encoder (CABAC), while preserving all SHVC functionalities. Based on the tile concept, the second proposed crypto-compression provides protection of the ROI defined in the standard HEVC.

Key Words
Chaos-based cryptosystems, Chaos-based joint crypto-compression systems, HEVC, SHVC, Chaos-based selective encryption, Performances