


Cryptanalyzing an image encryption scheme using reverse 2-dimensional chaotic map and dependent diffusion

Mousa Farajallah¹  · Safwan El Assad² · Olivier Deforges³

Received: 13 October 2017 / Revised: 5 April 2018 / Accepted: 16 April 2018 /
Published online: 27 April 2018
© Springer Science+Business Media, LLC, part of Springer Nature 2018

Abstract In the recent literature, many research studies have proven that Known and Chosen plaintext attacks are very efficient tools that are widely used to cryptanalyze partially or completely some chaos-based and non-chaos cryptosystems. In this paper, we addressed some weaknesses in the first Zhang et al., cryptosystem “An image encryption scheme using reverse 2-dimensional chaotic map and dependent diffusion”. First, we analyzed the encryption process of Zhang et al., and we found that the non-linear diffusion process can be removed because its argument is present in the ciphered image. Then, based on this observation we derived a partial cryptanalysis equation that removes the effect of the diffusion function and accordingly permits to recover the permuted version of the ciphered image. As a result of the previous operation, the brute-force attack became more suitable. In addition, we mounted a chosen plaintext attack based on a proposed chosen plain image. Consequently, the encryption key space is reduced or recovered for one round, also, the average values of NPCR and UCAI randomness parameters become small compared to the optimal values, and moreover, they are very low for specific pixel position attacks.

Keywords Cryptanalysis · Chosen plain-text attack · Chaos-based cryptosystem · Diffusion effect · Brute force attack · Key space

✉ Mousa Farajallah
mousa_math@ppu.edu

¹ College of Information Technology and Computer Engineering, Palestine Polytechnic University, Hebron, Palestine

² Ecole Polytech - University of Nantes, Rue Christian Pauc, Cedex 3, 44306 Nantes, France

³ INSA of Rennes, Rennes, France

1 Introduction

In the last years, algorithms that are used to secure information become more important, specially with the rapid growth of multimedia networking and communication [1, 6, 24, 48]. For example, data security is one of the main obstacles against wide adoption of cloud computing [22]. In cryptography, two essential requirements are important in any cryptosystem: a high encryption throughput and a high security level. Normally, it is difficult to design a cryptosystem that can achieve these two requirements together. A trade off between security level and encryption speed depends on the target application [8, 11, 29], for example, some applications require a security level for some days or hours and a slow encryption time, while other applications require a high security level with extra computational time. Shannon in his research [38] defines the importance of confusion and diffusion effects, and stated that, “in a strongly ideal cipher all statistics of the cryptogram are independent of the particular key used”. The confusion property aims to make the statistical relationship between the cipher image and the secret key as complex as possible [33], whereas the diffusion property aims to make the statistical relationship between the plain image and the cipher image as complex as possible [9, 14, 18]. The diffusion process modifies the statistical properties of the plain image by spreading the effect of each bit/byte of the plain image all over the cipher image. As a result, the efficiency of using differential attacks is decreased significantly [5, 33, 45]. Based on Kerckhoffs’ principle [34], the security of the encryption algorithm should only be based on the secret key and all the system parts should be known for the public [15]. A cryptanalyst tried to break the cipher without knowing the secret key, with several levels of difficulties based on the available resources.

Any proposed cryptosystem should pass all statistical tests, and it should be evaluated regarding to the existing and known mathematical attacks.

The cryptanalysis research papers confirm that the security evaluation of cryptosystems by standard statistical tools is not a sufficient proof of their security [2, 7, 10, 16, 21, 32, 35, 36, 40–43, 47]. Also, in recent years many studies demonstrate the weaknesses of some chaos-based and non-chaos cryptosystems against the plain-chosen text attacks and against a combination of differential-chosen text attacks. For example, Y. Zhang et al., [51] broke a chaotic image encryption algorithm based on the Perceptron Model by finding the equivalent secret key, using only one pair of known-plaintext/ciphertext. Moreover, the proposed encryption algorithm “A simple, sensitive and secure image encryption algorithm based on hyper-chaotic system with only one round diffusion” [31] is cryptanalyzed by applying known plaintext and chosen plaintext attacks by Y. Zhang et al., [53]. Li et al., [20] broke a chaotic image encryption algorithm based on the modulo addition and XOR operation by using two known plain-images and the corresponding cipher-images. The attack is based on some properties of solving a composite function involving a carry bit, which is composed of the modulo addition and bitwise OR operations. Zhang and Xiao [49] carried out cryptanalysis of S-box-only chaotic image ciphers by using chosen plaintext attack, and demonstrated that the computational complexity of the attack is only $O(128L)$, where the constant L is the total number of pixels with respect to the image. Again, Y. Zhang et al., broke a novel image cipher based on mixed transformed logistic maps [54], by applying chosen plaintext attack, six odd integer keys and three chaotic keystreams equivalent to the chaotic keys were revealed. Liu and Liu [26] investigated the security of an image scheme based on a permutation layer carried out by the Cat-map and chaos-based substitution layer. Eight images are selected in revealing all keystreams. Where as when seven images are selected the block-Cat-map is broken. By using a combination of chosen-plaintext attack and differential attack, Y. Zhang et al., revealed all the keystream of an image scrambling

based on chaotic sequences and Vigenère cipher In [55]. L. Y. Zhang et al., [50] applied the differential cryptanalysis on a chaos-based image encryption algorithm using an alternate structure. The differential attack can recover an equivalent secret key with only a small number of chosen plain-images. Finally, li et al., [19] pointed out that all permutation-only image ciphers are insecure against known/chosen-plaintext attacks in the sense that only $O(\log_L(MN))$ known/chosen plain-images are sufficient to break the ciphers, where MN is the size of the image and L is the number of different pixel values. Also, it is found that the attack complexity is only $O(n(MN)^2)$, where n is the number of known/chosen plain-images used.

In the proposed work, partial cryptanalysis of the first Zhang et al., cryptosystem is carried out. It is based on a chosen plaintext attack and a mathematical model to remove the diffusion effect of the last round of dependent diffusion, decrease the security level based on differential attacks, and decrease the encryption key space (i.e., sometimes called sub-key, it is generated from the secret key for each new encryption round and the used mathematical function should be difficult to invert. [12, 30, 56]).

It is important to note that Zhang et al., cryptosystem can be secure when the variable n is greater than 2, but in this case the system becomes time consuming and not suitable for real-time applications.

This paper is organized as follows: Section 2 presents the cryptosystem of Zhang et al., [52]. In Section 3, the partial cryptanalysis of the Zhang et al., cryptosystem is described in detail. Section 4 presents our conclusion.

2 The first Zhang et al., cryptosystem

Two cryptosystems were designed based on Fridrich’s architecture in Zhang et al., paper [52]. The first consists of a dependent diffusion layer based on the reverse 2-D cat map. The second presents new mapping from a pseudo-random position to another for the confusion effect. The diffusion layer of both cryptosystems is based on the logistic map. In these versions, Zhang et al., tried to achieve the confusion and the diffusion effects sequentially. Then, the effect of one ciphered pixel is transferred to the next and so on. 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 separate confusion and diffusion processes used in the traditional structures such as Fridrich cryptosystem [46].

Our work is directed to the first Zhang et al., cryptosystem. The mathematical model of the first Zhang et al., cryptosystem, (Enc=the encryption process) is:-

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

The general block diagram of the first Zhang et al., cryptosystem is shown in Fig. 1. It consists of the following steps iterated n times (with $n > 0$):

1. Selection: this step generates a random pair $arr(r_x^j, r_y^j)$ from the whole image. The values of the variable r_x^j and the variable r_y^j are calculated using (2) and (3), where the variable j is a counter ranging from 0 to $n - 1$ encryption rounds.

$$r_x^j = (SQ_1(2000 + 100 + j) \times 10^9) mod 512 \tag{2}$$

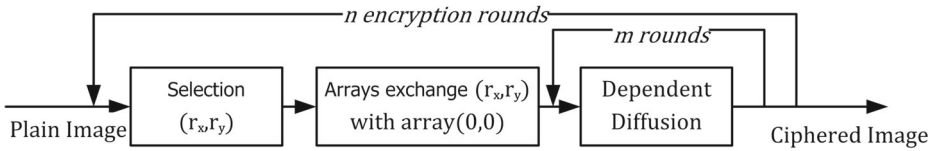


Fig. 1 Zhang et al., image encryption cryptosystem architecture

$$r_y^j = (SQ_2(2000 + 100 + j) \times 10^9) \bmod 512 \tag{3}$$

SQ_1 and SQ_2 series are calculated using (7).

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: the cryptosystem goes to the dependent diffusion layer for m rounds ($m = 2$ in the Zhang et al., cryptosystem 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 (4).

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & p_i \\ q_i & p_i q_i + 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \pmod{N} \tag{4}$$

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

$$p_i = (SQ_1(2000 + i) \times 10^9) \bmod 512 \tag{5}$$

$$q_i = (SQ_2(2000 + i) \times 10^9) \bmod 512 \tag{6}$$

The variable i is a counter ranging from 0 to $m - 1$. The two sequences SQ_1 and SQ_2 used in (2, 3, 5), and (6) are calculated using the following equation:

$$f(x_n) = \alpha \times x_{n-1}(1 - x_{n-1}) \tag{7}$$

Where the initial values $x_{-1} = 0.12345678912345$ for SQ_1 , and for SQ_2 is $x_{-1} = 0.67856746347633$. 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:

$$ciph(x, y) = arr(x', y') \oplus f(t) \tag{8}$$

where

$$f(t) = \left[\alpha \left(\frac{t}{1000} \right) \times \left[1 - \frac{t}{1000} \right] \times 1000 \right] \bmod 256 \tag{9}$$

- (c) Update of t : the last step of the dependent diffusion layer is to change the value of the variable t using (10).

$$t = ciph(x, y) \tag{10}$$

The initial value t_0 is defined by the following equation:

$$t_0 = [4 \times key_d \times (1 - key_d) \times 1000] \bmod 256. \tag{11}$$

Where the initial value of the diffusion key variable $key_d = 0.33456434300001$.

3 Research objectives

In the proposed work, the first Zhang et al., cryptosystem is presented and analyzed in order to perform the following research objectives:

1. Deriving a mathematical equation to recover a per-mutated version of the ciphered image.
2. Removing the diffusion effect.
3. Applying the Chosen Plaintext Attack (CPA).
4. Applying brute force attack and CPA together.
5. Decreasing the UACI and NPCR values significantly.
6. One encryption round and two dependent diffusion rounds have been broken completely. Moreover, the partial cryptanalysis equation is used to significantly decrease the robustness of the cryptosystem for two encryption rounds and two dependent diffusion rounds (assumed secure in the Zhang et al., cryptosystem) regarding differential attacks.

4 Partial cryptanalysis of Zhang et al., cryptosystem

In this section, the number of depended diffusion rounds and cryptanalysis levels are presented. The partial cryptanalysis equation and the used scenario are described to decrease the key space. Based on our chosen image the CPA is analyzed. NPCR and UACI values are reproduced using our proposed equation.

4.1 Conditions for the proposed partial cryptanalysis

The proposed partial cryptanalysis is highly dependent on the number of encryption rounds. The conditions and the level of the proposed cryptanalysis are summarized in Table 1.

The Zhang et al., cryptosystem is susceptible to the brute force attack for $(n = 1, m = 1)$, where as for other cases it seems to be secure. The robustness against the brute force attack is described in detail in Section 4.3.1. Regarding differential attacks, the Zhang et al., cryptosystem does not pass this type of attack because after applying our partial cryptanalysis, the UACI and NPCR values become too far from the optimal values. Section 4.3.5 presents a detailed study of this type of attacks. Finally, the used encryption keys of the Zhang et al., cryptosystem can be recovered as described in Section 4.3.4.

Table 1 Partial cryptanalysis level

Encryption rounds		Cryptanalysis level		
n	m	Decreasing the key space	Pass differential attacks	Find the dynamic key
1	1	Yes	No	Yes
1	2	Yes	No	Yes
2	1	Yes	No	Yes
2	2	Yes	No	No
3	1	Yes	No	No
3	2	Yes	No	No

4.2 Partial cryptanalysis method and the scenario principle

In fact, the encryption key space of the diffusion function $f(t)$ (8) can be removed from the calculation of the total encryption key space independently of the used key, because the t values are clearly presented in the ciphered image. This is the core of the partial cryptanalysis method. To prove the correctness of the above assumptions, we derive the following scenario:

$$\begin{aligned}
 ciph_0 &= arr_{k'_0} \oplus f(t_0) \\
 ciph_1 &= arr_{k'_1} \oplus f(t_1) = arr_{k'_1} \oplus f(ciph_0) \\
 ciph_2 &= arr_{k'_2} \oplus f(t_2) = arr_{k'_2} \oplus f(ciph_1) \\
 ciph_3 &= arr_{k'_3} \oplus f(t_3) = arr_{k'_3} \oplus f(ciph_2) \\
 ciph_4 &= arr_{k'_4} \oplus f(t_4) = arr_{k'_4} \oplus f(ciph_3) \\
 ciph_k &= arr_{k'_k} \oplus f(t_k) = arr_{k'_k} \oplus f(ciph_{k-1}),
 \end{aligned}$$

Where $arr_{k'_0}$ is coming from the plain image of the current encryption round, $ciph_0$ is the first ciphered pixel which is the result of (8), (as an example, $arr_{k'_0}$ in the case of $n = 1, m = 1$ it is the original plain image ($arr(x', y')$), while in the case of $n = 1, m = 2$ it is the encrypted image which is the output of (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 et al., cryptosystem as:

$$arr_{k'_k} = ciph_k \oplus f(ciph_{k-1}) \tag{12}$$

where

$$\begin{aligned}
 k &= x \times N + y \\
 k' &= x' \times N + y' \ ((x', y') \text{ is the new position calculated from the old one } (x, y), (4)).
 \end{aligned}$$

Note that $arr_{k'_k}$ 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, (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.

4.3 Partial Cryptanalysis results and benefits

The proposed partial cryptanalysis scenario can be used by the cryptanalyst to perform one of the following attacks:

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

4.3.1 Decreasing the encryption key space of the whole cryptosystem

The brute-force attack is the basic attack that can be used against any cryptosystem. It tries all possible keys until the correct key is found. In the worst case, all possible keys in the key space are tested [37, 39]. From (4), it is clear that the key space of the 2-D cat map for Zhang et al., 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 (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 time complexity spent on performing the brute force attack on Zhang et al., cryptosystem is:

$$KS = (S_1 \times S_2)^{n \times m},$$

where the parameter KS is the total encryption key space, S_1 represents the encryption key space for the standard 2-D cat map, and S_2 represents the encryption key space for the logistic map implemented by a lookup table $S_1 = N^2$, $S_2 = 2^8$, and so

$$KS = (N^2 \times 2^8)^{n \times m}.$$

For one encryption round and one dependent diffusion round ($n = 1, m = 1$).

With $N = 512$

$$KS = (2^{18} \times 2^8)^1.$$

The cryptanalysis time complexity is 2^{26} which is less than 2^{128} .

Equation (12) can be used to find the permuted plain image which means that the encryption key space of the parameter t can be removed from the encryption key space analysis:

$$KS = (2^{18})^1.$$

The cryptanalysis time complexity is 2^{18} which is less than 2^{128} .

For one encryption round, two dependent diffusion rounds ($n = 1, m = 2$), ($N = 512$) and if we assume that the Zhang et al., cryptosystem encrypts a plain image P into a cipher image C_1 in the first dependent diffusion round, and then encrypts the cipher image C_1 into a cipher image C_2 in the second dependent diffusion round. Using (12) we can at least find non-order C_1 pixels easily within 10 ms. To find the original plain image P (at $n = 1, m = 2$), the cryptanalytic system needs 10 ms to obtain the cipher image C_1 from the cipher image C_2 . For each possible value of the encryption keys p_2 and q_2 , the ciphered image C_1 is decrypted using the brute-force attack for all possible values of the encryption keys p_1 and q_1 :

$$KS = (S_1)^2, S_1 = 2^{36}, \text{ The cryptanalysis time complexity is } 2^{36} \text{ which is less than } 2^{128}.$$

For one encryption round, one dependent diffusion round (i.e., $n = 1, m = 1$), and ($N = 256$),

$$KS = (S_1)^1, S_1 = 2^{16},$$

The cryptanalysis time complexity is 2^{16} which is less than 2^{128} .

For one encryption round, two dependent diffusion rounds (i.e., $n = 1, m = 2$), and ($N = 256$)

$$KS = (S_1)^2, S_1 = 2^{32} :$$

The cryptanalysis time complexity is 2^{32} which is less than 2^{128} .

Note that, the possible values of the encryption keys p_1 and q_1 are completely independent for each round. Which opens the possibility of using parallel cryptanalysis.

4.3.2 Chosen plaintext attack (CPA) on the first Zhang et al., cryptosystem

The CPA is defined as a cryptanalysis model when the adversary has the capability of choosing some plaintexts and encrypting them. The adversary studies the corresponding ciphertexts to obtain some information on the used keys or even to reduce the security level of the cryptosystem [3, 13, 17, 23]. From this well-known definition, we can choose a plaintext and encrypt it without knowing the encryption keys, and our objective is to calculate the encryption keys. The following scenario describes the proposed partial cryptanalysis of

the first Zhang et al., cryptosystem. The chosen plain image is used to find the encryption keys (q_1, p_1) of the first dependent diffusion round. Using these keys we can decipher any ciphered image with the same encryption keys (this scenario is only applicable in the case where $n = 1, m = 1$). We choose a specific plain image P of size $512 \times 512 \times 1$ byte. This plain image is chosen to help us in the process of finding the encryption keys (q_1, p_1) . Indeed, we try to fill each position with a predefined value to decrease the range of possible values of the encryption keys q_1 and p_1 .

The possible pixel value ranges from 0 to 255 and, the Zhang et al., image size has 512 rows and 512 columns. We fill every two rows with the same pixel value except pixels 0, 1 and 2 (the justification of this exception is described later). Knowing the plain pixel value helps to determine the row which is related to the value of (p_1) . To determine the value of $(q_1, \text{ which refers to the column number})$ the second and the fourth rows are filled using the sequential series 1, 2, 3, 4, ... 255, 1, 2, 3, 4, ... 255, 1, 2. Now, the exceptions are:

- A pixel value of 0 has been placed in the top left of the plain image, which is not a useful pixel because of the second step of the Zhang et al., (Array Exchanges).
- A pixel value of 1 has been placed in three rows (it can be any pixel of value 2, 3, ... 255 the idea here is to have three rows instead of two filled with the same value. We can replace the fifth row by 2's instead of 1's with no major changes, but we cannot add 0 more than one time in the whole image).

For example, assuming that the decrypted value is 5, then by looking at the following given matrix P , we can be sure that only rows (1, 3, 9 or 10) contain this value. This will be helpful in the process of finding values of the encryption keys q_1 and p_1 as illustrated below. The chosen plain image P is encrypted, and the ciphered image (C_1) is obtained. A group of ciphered pixels is used to analyze the encryption process and to try to recover the encryption keys. For this we introduce the steps:

$$P = \begin{pmatrix} & 0 & 1 & 2 & \dots & 254 & 255 & \dots & 509 & 510 & 511 \\ 0 & 0 & 1 & 1 & \dots & 1 & 1 & \dots & 1 & 1 & 1 \\ 1 & 1 & 2 & 3 & \dots & 255 & 1 & \dots & 255 & 1 & 2 \\ 2 & 2 & 2 & 2 & \dots & 2 & 2 & \dots & 2 & 2 & 2 \\ 3 & 3 & 4 & 5 & \dots & 2 & 3 & \dots & 2 & 3 & 4 \\ 4 & 1 & 1 & 1 & \dots & 1 & 1 & \dots & 1 & 1 & 1 \\ 5 & 3 & 3 & 3 & \dots & 3 & 3 & \dots & 3 & 3 & 3 \\ 6 & 3 & 3 & 3 & \dots & 3 & 3 & \dots & 3 & 3 & 3 \\ 7 & 4 & 4 & 4 & \dots & 4 & 4 & \dots & 4 & 4 & 4 \\ 8 & 4 & 4 & 4 & \dots & 4 & 4 & \dots & 4 & 4 & 4 \\ 9 & 5 & 5 & 5 & \dots & 5 & 5 & \dots & 5 & 5 & 5 \\ 10 & 5 & 5 & 5 & \dots & 5 & 5 & \dots & 5 & 5 & 5 \\ 11 & 6 & 6 & 6 & \dots & 6 & 6 & \dots & 6 & 6 & 6 \\ 12 & 6 & 6 & 6 & \dots & 6 & 6 & \dots & 6 & 6 & 6 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 507 & 254 & 254 & 254 & \dots & 254 & 254 & \dots & 254 & 254 & 254 \\ 508 & 254 & 254 & 254 & \dots & 254 & 254 & \dots & 254 & 254 & 254 \\ 509 & 255 & 255 & 255 & \dots & 255 & 255 & \dots & 255 & 255 & 255 \\ 510 & 255 & 255 & 255 & \dots & 255 & 255 & \dots & 255 & 255 & 255 \\ 511 & 1 & 1 & 1 & \dots & 1 & 1 & \dots & 1 & 1 & 1 \end{pmatrix}$$

First encryption key calculations (q_1) To find the value of the encryption key q_1 , as described before, the second and the fourth rows are only helpful to decrease the possible values of the encryption key q_1 . Two steps are carried out:

First Step As mentioned before, q_1 refers to the column position. Using the ciphered pixel at position ($x = 1$, and $y = 0$), the plain pixel position (x' , y') is calculated using (4):

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

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

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 (12): $arr(1, q_1) = ciph_k \oplus f(ciph_{k-1})$, where $ciph_k = ciph(1, 0)$ and $ciph_{k-1}$ is calculated using (13).

$$(x_{previous}, y_{previous}) = \begin{cases} (x - 1, N - 1), & k \text{ is a multiple of } N \\ (x, y - 1), & \text{otherwise} \end{cases} \tag{13}$$

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

The ciphered pixels ($ciph(1, 0)$, $ciph(0, 511)$) are known. The function $f(t)$ is also known. 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.

Equation (14) is used to decrease the key space of the encryption key q_1 .

$$q_1 \in \begin{cases} \text{Skip this value,} & arr(1, q_1) = 0 \\ \{0, 255, 510\}, & arr(1, q_1) = 1 \\ \{1, 256, 511\}, & arr(1, q_1) = 2 \\ \{arr(1, q_1) - 1, arr(1, q_1) + 254\}, & \text{otherwise} \end{cases} \tag{14}$$

If $arr(1, q_1) = 0$, it is skipped since it can come from any position of the chosen plain image P (Fig. 1 justifies this, since the first pixel is swapped with a random pixel from the whole image). If $arr(1, q_1) = 1$, the value can be located in one of three positions on row number 1 (second row): [(1, 0), (1, 255) or (1, 510)]. As a result, the possible values of q_1 are 0, 255 or 510. If that $arr(1, q_1) = 2$, the value can be located in one of three positions on row number 1: [(1, 1), (1, 256) or (1, 511)]. Finally, if $arr(1, q_1) > 2$, the 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 possible range values of the encryption key q_1 , another ciphered pixel at position ($x = 3$, and $y = 0$) is considered. The pixel is located in the fourth row. The position (x' , y') of the corresponding plain pixel is calculated using (4):

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

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

Using (12) and (13)

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

The second decrypted pixel $arr(3, 3q_1)$ is located in row number 3, and the column position $3q_1$, with $0 \leq 3q_1 \leq 511$. Equation (15) is used to decrease the key space of the encryption key q_1 (simplification of (15) is given in Appendix A).

$$3q_1 \in \begin{cases} \text{Skip this value,} & arr(3, 3q_1) = 0 \\ \{253, 508\}, & arr(3, 3q_1) = 1 \\ \{254, 509\}, & arr(3, 3q_1) = 2 \\ \{0, 255, 510\}, & arr(3, 3q_1) = 3 \\ \{1, 256, 511\}, & arr(3, 3q_1) = 4 \\ \{a, b\}, & \text{otherwise} \end{cases} \tag{15}$$

where

$$a = \begin{cases} \frac{arr(3, 3q_1)-3}{3}, & (arr(3, 3q_1) - 3) \text{ MOD } 3 = 0 \\ \frac{arr(3, 3q_1)+509}{3}, & (arr(3, 3q_1) - 3 + 512) \text{ MOD } 3 = 0 \\ \frac{arr(3, 3q_1)+1021}{3}, & \text{otherwise} \end{cases} \tag{16}$$

$$b = \begin{cases} \frac{arr(3, 3q_1)+252}{3}, & (arr(3, 3q_1) + 252) \text{ MOD } 3 = 0 \\ \frac{arr(3, 3q_1)+764}{3}, & (arr(3, 3q_1) + 252 + 512) \text{ MOD } 3 = 0 \\ \frac{arr(3, 3q_1)+1276}{3}, & \text{otherwise} \end{cases} \tag{17}$$

From the previous four ciphered pixels ($ciph(1, 0)$, $ciph(0, 511)$, $ciph(3, 0)$ and $ciph(2, 511)$), the exact value of the encryption key q_1 is calculated by finding the overlap values between the **first** and the **second steps**. More details of this analysis are given in the Appendix A.

Second encryption key calculations (p_1) To find the value of the encryption key p_1 (which refers to the row position), at least two steps are needed:

First Step From the ciphered pixel at the ($x = 0$, and $y = 1$) position, the plain pixel position (x', y') is calculated using (4):

$$x' = 0 \times x + p_1 \times 1 \implies x' = p_1$$

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

Using (12), the value of $arr(p_1, p_1q_1 + 1)$ is calculated: $arr(p_1, p_1q_1 + 1) = ciph(0, 1) \oplus f(ciph(0, 0))$.

The decrypted pixel $arr(p_1, p_1q_1 + 1)$ is located at row number p_1 of the chosen plain image P . Here we focus on finding the p_1 value (p_1 ranges from 0 to 511). Equation (18) is used to decrease the encryption key space p_1 as:

$$p_1 \in \begin{cases} \text{Skip this value,} & arr(p_1, p_1q_1+1)=0 \\ \{0, 1, 3, 4, 511\}, & arr(p_1, p_1q_1+1)=1 \\ \{1, 2, 3\}, & arr(p_1, p_1q_1+1)=2 \\ \{1, 3, arr(p_1, p_1q_1+1) \times 2 - 1, arr(p_1, p_1q_1+1) \times 2\}, & \text{otherwise} \end{cases} \tag{18}$$

To justify (18), firstly, assume that $arr(p_1, p_1q_1 + 1) = 1$, from the chosen 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 encryption key p_1 are: [0, 1, 3, 4, 511]. Secondly, assume that $arr(p_1, p_1q_1 + 1) = 2$, then this value is located in three rows: [1, 2, 3], and so, the possible values of the encryption key p_1 are: [1, 2, 3]. Thirdly, assume that $arr(p_1, p_1q_1 + 1) = 3$, then this value is located in four rows [1, 3, 5, 6], and so, the possible values of the encryption key p_1 are: [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 encryption key p_1 are $[1, 3, arr(p_1, p_1q_1 + 1) \times 2 - 1, arr(p_1, p_1q_1 + 1) \times 2]$.

Second Step To further decrease the range values of the encryption key p_1 , another ciphered pixel at position $(x = 0, \text{ and } y = 2)$ is considered. The position (x', y') of the corresponding plain pixel is calculated using (4):

$$x' = 1 \times x + p_1 \times y \implies x' = 0 + 2p_1 = 2p_1 \implies \mathbf{x'} = 2\mathbf{p_1},$$

$$y' = q_1 \times x + (q_1 \times p_1 + 1) \times y \implies y' = q_1 \times 0 + (q_1 \times p_1 + 1) \times 2 \implies \mathbf{y'} = 2(q_1 \times p_1 + 1),$$

Again, using (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 chosen plain image P . The following (19) is used to decrease the key space of the encryption key p_1 as:

$$p_1 \in \begin{cases} \text{Skip this value,} & arr(2p_1, 2(p_1q_1 + 1)) = 0 \\ \{0, 2, 256, 258\}, & arr(2p_1, 2(p_1q_1 + 1)) = 1 \\ \{1, 257\}, & arr(2p_1, 2(p_1q_1 + 1)) = 2 \\ \{arr(2p_1, 2(p_1q_1 + 1)), arr(2p_1, 2(p_1q_1 + 1)) + 256\}, & \text{otherwise} \end{cases} \tag{19}$$

To justify (19), we consider the following cases:

Firstly, assume that the decrypted pixel value is $arr(2p_1, 2p_1q_1 + 2) = 1$, so from the chosen plain image P , the plain pixel value 1 is located in five rows $[0, 1, 3, 4, 511]$, and the possible values of the encryption key p_1 are as follows:

Using the equality equation $2p_1 = 0$, it gives that $p_1 \in \{0, 256\}$. While the equality equations $2p_1 = 1, 2p_1 = 3$ and $2p_1 = 511$ have no integer solution, since the right-hand side is odd and the left-hand side is even and the modulus value is even. The equality equation $2p_1 = 4$, it gives that $p_1 \in \{2, 258\}$.

Secondly, assume that the decrypted pixel value 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 the encryption key p_1 are as follows:

The equality equations $2p_1 = 1$ and $2p_1 = 3$ have no integer solution. While the equality equation $2p_1 = 2$, it gives that $p_1 \in \{1, 257\}$.

Thirdly, assume that the decrypted pixel value 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 the encryption key p_1 are as follows:

The equality equations $2p_1 = 1, 2p_1 = 3$ and $2p_1 = 5$ have no integer solution. While the equality equation $2p_1 = 6$, it gives that $p_1 \in \{3, 259\}$.

Finally, for any decrypted pixel value such that $arr(2p_1, 2p_1q_1 + 2) > 2$, the possible values of p_1 are as follows:

$$p_1 = arr(2p_1, 2p_1q_1 + 2) \text{ or } p_1 = arr(2p_1, 2p_1q_1 + 2) + 256.$$

Note that in the most cases the first and the second steps are sufficient to find the encryption key value of p_1 . However, if the overlap between the obtained range values of the encryption key p_1 in the first step and the obtained range values of p_1 in the second step is not a single value, then the following calculation is required.

Extra Step To find the exact value of the encryption key p_1 , the plain pixel position (x', y') is calculated from the ciphered position pixel at $(x = 0, y = 3)$, using (4):

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

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

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 (12). Now (20) is used to decrease the encryption key space of the encryption key p_1 . To simplify the notation of the calculations, assume:

$$3p_1 \in \begin{cases} \text{Skip this value,} & arr(3p_1, 3(p_1 \times q_1 + 1)) = 0 \\ \{0, 1, 3, 4, 511\}, & arr(3p_1, 3(p_1 \times q_1 + 1)) = 1 \\ \{1, 2, 3\}, & arr(3p_1, 3(p_1 \times q_1 + 1)) = 2 \\ \{1, 3, arr(3p_1, 3(p_1 \times q_1 + 1)) - 1, arr(3p_1, 3(p_1 \times q_1 + 1))\}, & \text{otherwise} \end{cases} \tag{20}$$

Regarding p_1 ,

$$p_1 \in \begin{cases} \text{Skip this value,} & arr(3p_1, 3(p_1 \times q_1 + 1)) = 0 \\ \{0, 1, 171, 172, 341\}, & arr(3p_1, 3(p_1 \times q_1 + 1)) = 1 \\ \{1, 171, 342\}, & arr(3p_1, 3(p_1 \times q_1 + 1)) = 2 \\ \{1, 171, a, b\}, & \text{otherwise} \end{cases} \tag{21}$$

Where

$$a = \begin{cases} \frac{(arr(3p_1, 3(p_1 \times q_1 + 1))) \times 2 - 1}{3}, & ((arr(3p_1, 3(p_1 \times q_1 + 1))) \times 2 - 1) \text{ MOD } 3 = 0 \\ \frac{(arr(3p_1, 3(p_1 \times q_1 + 1))) \times 2 + 511}{3}, & (((arr(3p_1, 3(p_1 \times q_1 + 1))) \times 2 + 511) \text{ MOD } 3 = 0) \\ \frac{(arr(3p_1, 3(p_1 \times q_1 + 1))) \times 2 + 1023}{3}, & \text{otherwise} \end{cases} \tag{22}$$

$$b = \begin{cases} \frac{(arr(3p_1, 3(p_1 \times q_1 + 1))) \times 2}{3}, & ((arr(3p_1, 3(p_1 \times q_1 + 1))) \times 2) \text{ MOD } 3 = 0 \\ \frac{(arr(3p_1, 3(p_1 \times q_1 + 1))) \times 2 + 512}{3}, & (((arr(3p_1, 3(p_1 \times q_1 + 1))) \times 2 + 512) \text{ MOD } 3 = 0) \\ \frac{(arr(3p_1, 3(p_1 \times q_1 + 1))) \times 2 + 1024}{3}, & \text{otherwise} \end{cases} \tag{23}$$

Simplification of (20) is given in Appendix A.

4.3.3 Time complexity analysis of the CPA

To evaluate the Time Complexity (TC) of the proposed CPA in the previous section, we need the following operations (later it is referred as TC_{CPA}):

1. One execution of (12) to find the $arr(1, q_1)$.
2. TC to perform (14).
3. One execution of (12) to further decrease the range values of the encryption key q_1 during the first step.
4. TC to perform (15).
5. XOR and shifting operations to find the overlap between the first and the second steps.
6. One execution of (12) to find the $arr(p_1, p_1 \times q_1 + 1)$.
7. TC to perform (18).
8. One execution of (12) to further decrease the range values of the encryption key p_1 during the second step.
9. TC to perform (19).
10. One execution of (12) to perform the **Extra Step** to further decrease the range values of the encryption key p_1 during the second step.
11. TC to perform (20).
12. XOR and shifting operations to find the overlap between the first and the second steps.

The time complexity of the proposed CPA can be determined using the basic operations required to achieve one execution from (12) to (20). 2 logical Shift operations, 5 logical IF operations, 7 XOR operations, 15 addition and subtraction operations, 15 multiplication and division operations, and 98 lock up and memory access operations are needed. In terms of instruction complexity, the required TC is 7 XOR operations, 5 logical If statements and 2 shift operations. In terms of execution time, all these operations take less than 5 ms in our simulation environment. (see the given examples in Appendix A.)

4.3.4 Combination of brute-force and chosen plaintext attacks

In the case where $n = 1, m = 2$, which is assumed as a secured case as it is mentioned in [52] at page 2074 “In Algorithm 1, the round numbers m and n as shown in Fig. 7 are selected as 2 and 1, respectively”, the algorithm can be partially cryptanalyzed. A combination of the aforementioned attacks are used to find the encryption 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 Zhang et al., cryptosystem is used to encrypt our chosen plain image: the chosen plain image P , is encrypted in the first dependent diffusion round ($n = 1, m = 1$) with the encryption key parameters (p_1, q_1) to obtain the middle cipher image (C_1) . In the second dependent diffusion round ($n = 1, m = 2$), the middle ciphered image (C_1) with encryption key parameters (p_2, q_2) is ciphered to obtain the cipher image (C_2) . The cryptanalysis scenario is carried out as:

1. Using (12) and the cipher image (C_2) , the permuted version of the middle ciphered image (C_1) can be recovered in 10 ms.
2. To guess the encryption key pair (p_2, q_2) , in the worst case, we need to try 2^9 possible encryption keys for the encryption key p_2 as well as for the encryption key q_2 . For each guess (p_2, q_2) , the required time to reorder the middle ciphered image pixels C_1 is 5 ms (achieved by inverse permutation). The proposed CPA in Section 4.3.2 is used to find the encryption key pair (p_1, q_1) of the first dependent diffusion round at ($n = 1, m = 1$) within 5 ms.

Time analysis In the worst case, the process requires less than two hours to find the values of the encryption keys (p_1, q_1) and (p_2, q_2) (for the image of size 512×512) since, the number of all possible encryption 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 as $T_{n=1,m=2}$:

$$T_{n=1,m=2} = T_1 + T_2 \times T_3$$

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 encryption key (p_2, q_2) and to order C_1 pixels. It is TC_{CPA} for each try. In the worst case it requires $2^{18} \times TC_{CPA}$.

T_3 is the required time to achieve the proposed attack of Section 4.3.2, which is less than the TC_{CPA} for each guess.

$$T_{n=1, m=2} = 10 + 2^{18} \times TC_{CPA} \times 5 \approx 110 \text{ minutes (i.e., } O(2^{22})), \text{ when } TC_{CPA} = 5.$$

To generalize the time complexity, assume the used image size is N . For each dependent diffusion round, number of bits for each key is $Log_2(N)$. The time complexity of the proposed attack depends on the image size and number of dependent diffusion round. Formally,

$$TC = 2^{m \times 2 \times Log_2(N)} \tag{24}$$

where, m is number of dependent diffusion round, and N is image size. In order to have secure version of the proposed Zhang et al., cryptosystem, TC should be greater than or equal to 2^{128} . Thus, $m \times 2 \times Log_2(N) \geq 128$. Assume $N = 512$, m should be greater

than 7. However, in this case, the Zhang et al., cryptosystem is time consuming, because each dependent diffusion round needs 10 ms. The total time required is 70 ms, which means that the Running Speed (RS) of the 512×512 gray image is $RS = \frac{512 \times 512}{0.070} = 3.58$ Mega bytes per second. This not suitable for real-time application or even high speed encryption applications.

4.3.5 Decreasing the UACI and NPCR values significantly

In this section, a sample experiment is carried out to show the effect of using (12) on the security level of the proposed Zhang et al., cryptosystem exactly in terms of differential attacks.

A cryptosystem should be sensitive to one-bit changes in the plaintext. This requirement is important to resist the known plaintext and the chosen plaintext attacks [25, 28]. In a chosen plaintext attack, more than one plaintext (with one-bit changes between them) is selected to analyze the difference between the corresponding ciphertexts. The measurement tool to test the sensitivity of any cryptosystem regarding these attacks is carried out as: Select P_1 as the first plain image, change one bit in P_1 and name it P_2 . (i.e., P_1 and P_2 are exactly the same except for one bit, to be more accurate, this bit should be located at the beginning, middle or the end of the tested image/block). Then both images (P_1 and P_2) are encrypted using the same secret key. This encryption produces two cipher images Cd_1 and Cd_2 . Most researchers use two security parameters to measure the resistance of any chaos-based cryptosystem against a plaintext sensitivity attack (differential attacks introduced by Eli Biham and Adi Shamir [4]). These parameters are the Number of Pixel Change Rate (NPCR) and the Unified Average Changing Intensity (UACI), given by the equations:

$$NPCR = \frac{1}{L \times C \times P} \times \sum_{p=1}^P \sum_{i=1}^L \sum_{j=1}^C D(i, j, p) \times 100\% \tag{25}$$

where

$$D(i, j, p) = \begin{cases} 0, & \text{if } Cd_1(i, j, p) = Cd_2(i, j, p) \\ 1, & \text{if } Cd_1(i, j, p) \neq Cd_2(i, j, p) \end{cases} \tag{26}$$

$$UACI = \frac{1}{L \times C \times P \times 255} \times \sum_{p=1}^P \sum_{i=1}^L \sum_{j=1}^C |Cd_1(i, j, p) - Cd_2(i, j, p)| \times 100\% \tag{27}$$

The optimal NPCR value is 99.61%, and the optimal UACI value is 33.46% [27, 44].

We try to reproduce the NPCR and UACI results of the Zhang et al., cryptosystem using exactly the same parameters as were used in Zhang et al., cryptosystem analysis. (i.e., Barbara $512 \times 512 \times 1$ gray-scale image, $x_{-1} = 0.12345678912345$ for SQ_1 , $x_{-1} = 0.67856746347633$ for SQ_2 . The value of the constant α is set to 3.99999, and the diffusion key $key_d = 0.33456434300001$, with $n = 2, m = 2$). steps performed are:

1. Encrypting the Barbara image (i.e., P_1) using the first Zhang et al., cryptosystem and using the same secret keys mentioned above to obtain the ciphered image Cd_1 . Then, changing one bit in the Barbara image (i.e., P_2) and encrypting it to obtain another ciphered image Cd_2 . The UACI and NPCR are calculated between the two ciphered images Cd_1 and Cd_2 .
2. The ciphered images Cd_1 and Cd_2 are used as input for the partial cryptanalysis process based on (12) to remove the diffusion effect in less than 1 ms and the UACI and NPCR

Table 2 Sample of NPCR and UACI results under the same parameters and conditions in the original research paper for all pixel positions

RP ^a	CP ^b	UACI	Result	NPCR	Result	UACI _p	Result	NPCR _p	Result
0	0	33.529	Pass	99.624	Pass	31.747	Fail	93.872	Fail
14	103	33.299	Fail	98.883	Fail	11.950	Fail	35.471	Fail
26	476	33.212	Fail	98.569	Fail	07.313	Fail	21.654	Fail
39	98	32.987	Fail	97.865	Fail	05.073	Fail	14.990	Fail
97	475	32.699	Fail	96.828	Fail	03.425	Fail	10.135	Fail
125	87	33.473	Pass	99.611	Pass	33.171	Fail	98.668	Fail
296	337	32.797	Fail	97.139	Fail	04.482	Fail	13.224	Fail
471	375	33.151	Fail	98.256	Fail	07.441	Fail	22.040	Fail
511	511	33.478	Pass	99.611	Pass	33.034	Fail	98.057	Fail

^aRow position

^bColumn position

Table 3 Sample of NPCR and UACI results

RP ^a , CP ^b	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

^aRow position

^bColumn position

are recalculated again (in Table (2), the UACI for this case is $UACI_p$ while the NPCR in this case is the $NPCR_p$).

The results obtained in Table 2, measured by the $UACI_p$ and the $NPCR_p$ (index p refers to the calculated NPCR and UACI values after applying our cryptanalysis (12), show that for some pixel positions [(511, 511), (0, 0) and (125, 87)], with a one-bit change value (the LSB bit), Zhang et al., cryptosystem is not secure.

We have done the same experiment explained above, for two pixel positions, but in relation to the nature of the image and to the secret key of the Logistic map. The results given in Table 3, of the parameters $UACI_p$, $NPCR_p$, show the sensitivity of Zhang et al., cryptosystem regarding the image under test and the used secret key of the Logistic map. The used values of Key 1 and Key 2, are:

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 $key_d = 0.50190740684224977$.

5 Conclusion

In this paper, one of the fastest chaos-based cryptosystems, namely the Zhang et al., cryptosystem is studied and analyzed. We partially cryptanalyzed it using a combination of a precise plaintext, as chosen plaintext attack, and a brute-force attack, for parameters ($n = 1$, $m = 2$). Next, after removing the diffusion effect using (12), we succeeded in significantly decreasing the security level measured by the well-known parameters NPCR and UACI, by applying the differential attacks for ($n = 2$, $m = 2$).

Acknowledgements This work is supported by the European Celtic-Plus project 4KREPROSYS - 4K ultraHD TV wireless REmote PROduction SYStems.

Appendix A: Numerical example on chosen plaintext attack of the first Zhang et al., cryptosystem

The Justification of (15) is given as: (15) is used to decrease the number of possible column positions (i.e., the range of the encryption key q_1). Using the chosen plain image 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 the chosen plain 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, the equality equation $3q_1 = 1$; to solve this equation, which contains the module operation, the encryption key value q_1 is calculated as:

$$q_1 = \text{the integer value of } \left(\frac{1}{3}, \frac{1+512}{3}, \frac{1+512+512}{3} \right) \implies q_1 = 171.$$

Secondly, the equality equation $3q_1 = 256$, then the encryption key $q_1 =$ the integer value of $\left(\frac{256}{3}, \frac{256+512}{3}, \frac{256+512+512}{3} \right) \implies q_1 = 256$.

Finally, the equality equation $3q_1 = 511$, then the encryption key $q_1 =$ the integer value of $\left(\frac{511}{3}, \frac{511+512}{3}, \frac{511+512+512}{3} \right) \implies q_1 = 341$.

For all other values ($arr(3, 3q_1) > 4$), there is a general scheme to calculate the range of the encryption key q_1 values (i.e., a and b parameters in (15)). Now, let us take $arr(3, 3q_1) = 5$ as another example. i.e., $3q_1 \in \{2, 257\}$.

Firstly, the equality equation $3q_1 = 2$, then: the encryption key $q_1 =$ the integer value of $\left(\frac{2}{3}, \frac{2+512}{3}, \frac{2+512+512}{3} \right) \implies q_1 = 342$.

Secondly, the equality equation $3q_1 = 257$.

The encryption key $q_1 =$ the integer value of $\left(\frac{257}{3}, \frac{257+512}{3}, \frac{257+512+512}{3} \right) \implies q_1 = 427$.

From previous developments, we can deduce the following rules for (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 $arr(3, 3q_1) + 252$.

To justify (20), assume that the ciphered pixel value is one (i.e., $T = 1$ in (20)). From the chosen 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, the equality equation $3p_1 = 0$; to solve this equation, which contains the module operation, the encryption key p_1 is calculated as: the encryption key $p_1 =$ the integer value of $\left(\frac{0}{3}, \frac{0+512}{3}, \frac{0+512+512}{3} \right) \implies p_1 = 0$.

Secondly, the equality equation $3p_1 = 1$.

The encryption key $p_1 =$ the integer value of $\left(\frac{1}{3}, \frac{1+512}{3}, \frac{1+512+512}{3} \right) \implies p_1 = 171$.

Thirdly, the equality equation $3p_1 = 3$.

The encryption key $p_1 =$ the integer value of $\left(\frac{3}{3}, \frac{3+512}{3}, \frac{3+512+512}{3} \right) \implies p_1 = 1$.

Fourthly, the equality equation $3p_1 = 4$, then the encryption key $p_1 =$ the integer value of $\left(\frac{4}{3}, \frac{4+512}{3}, \frac{4+512+512}{3} \right) \implies p_1 = 172$.

Finally, the equality equation $3p_1 = 511$.

The encryption key $p_1 =$ the integer value of $\left(\frac{511}{3}, \frac{511+512}{3}, \frac{511+512+512}{3} \right) \implies p_1 = 341$, $p_1 \in \{0, 1, 171, 172, 341\}$. The same analysis is used for the other values of the variable T , where for $T = 2$, there are only three possible rows, and for $T > 2$, there are four possible rows.

Example 1 Calculation of the encryption keys p_1 and q_1 .

A chosen plain image P 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: $[n = 1, m = 1]$,

$$\begin{aligned} \mathbf{ciph(1, 0)} &= \mathbf{190}, \\ \mathbf{ciph(0, 511)} &= \mathbf{27}, \\ \mathbf{ciph(3, 0)} &= \mathbf{149}, \\ \mathbf{ciph(2, 511)} &= \mathbf{159}, \\ \mathbf{ciph(0, 1)} &= \mathbf{132}, \\ \mathbf{ciph(0, 0)} &= \mathbf{165}, \\ \mathbf{ciph(0, 2)} &= \mathbf{140}, \\ \mathbf{ciph(0, 1)} &= \mathbf{132}. \end{aligned}$$

For the first decrypted pixel, using (12) we obtain:

$$\begin{aligned} arr(1, q_1) &= ciph(1, 0) \oplus f(ciph(0, 511)), \\ arr(1, q_1) &= 190 \oplus f(27), \\ arr(1, q_1) &= 190 \oplus 105, \\ arr(1, q_1) &= 215. \end{aligned}$$

Using (14), the encryption key 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 (12) we obtain:

$$\begin{aligned} arr(3, 3q_1) &= ciph(3, 0) \oplus f(ciph(2, 511)), \\ arr(3, 3q_1) &= 149 \oplus f(159), \\ arr(3, 3q_1) &= 149 \oplus 22, \\ arr(3, 3q_1) &= 131. \end{aligned}$$

Using (15), the encryption key value $q_1 \in \{a, b\}$.

To find the value of the parameter a of (15), the following conditions are met:

$$\begin{aligned} ((131 - 3) \text{ MOD } 3) &= 2, \\ ((131 + 509) \text{ MOD } 3) &= 1, \\ ((131 + 1021) \text{ MOD } 3) &= 0. \end{aligned}$$

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 (15), the following conditions are met:

$$\begin{aligned} ((131 + 252) \text{ MOD } 3) &= 2, \\ ((131 + 764) \text{ MOD } 3) &= 1, \\ ((131 + 1276) \text{ MOD } 3) &= 0. \end{aligned}$$

The last condition allows us to calculate the value of parameter b :

$$b = 469, \text{ so, } q_1 \in \{384, 469\}.$$

The overlap between the first range {214, 469} and the second {384, 469} range gives the exact value of the encryption key q_1 : $q_1 = 469$.

To calculate the exact value of the encryption key p_1 , we consider the third decrypted pixel and (12), then:

$$\begin{aligned} arr(p_1, p_1q_1 + 1) &= ciph(0, 1) \oplus f(ciph(0, 0)), \\ arr(p_1, p_1q_1 + 1) &= 132 \oplus f(165), \\ arr(p_1, p_1q_1 + 1) &= 132 \oplus 39, \\ arr(p_1, p_1q_1 + 1) &= 163. \end{aligned}$$

Now by using (18), the encryption key 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 the encryption key p_1 is obtained by using the fourth decrypted pixel and (12), as follows:

$$\begin{aligned} arr(2p_1, 2p_1q_1 + 2) &= ciph(0, 2) \oplus f(ciph(0, 1)), \\ arr(2p_1, 2p_1q_1 + 2) &= 140 \oplus f(132), \\ arr(2p_1, 2p_1q_1 + 2) &= 140 \oplus 202, \\ arr(2p_1, 2p_1q_1 + 2) &= 70. \end{aligned}$$

Using (19):

$$p_1 \in \{70, 326\}.$$

The overlap between the first range ({1, 3, 325, 326}) and the second range ({70, 326}) gives the exact value of the encryption key p_1 : $p_1 = 326$.

Remark The above procedure can be used to find any encryption keys p_1 and q_1 parameters.

Example 2 The original image P is encrypted using new random values of key_d and x_{-1} , where x_{-1} for SQ_1 is differed from the value for SQ_2 . Also, Lena image of the same size is encrypted using the same dynamic keys, The following values are obtained from the encrypted image of P :

$$\begin{aligned} ciph(1, 0) &= 226 \\ ciph(0, 511) &= 166 \\ ciph(3, 0) &= 142 \\ ciph(2, 511) &= 64 \\ ciph(0, 1) &= 87 \\ ciph(0, 0) &= 45 \\ ciph(0, 2) &= 202 \\ ciph(0, 1) &= 87 \\ arr(1, q_1) &= ciph(1, 0) \oplus f(ciph(0, 511)) \\ arr(1, q_1) &= 226 \oplus f(166) \\ arr(1, q_1) &= 190 \oplus 41 \\ arr(1, q_1) &= 203 \\ q_1 &\in \{203 - 1, 203 + 254\} \\ q_1 &\in \{202, 457\} \end{aligned}$$

$$\begin{aligned}
 arr(3, 3q_1) &= ciph(3, 0) \oplus f(ciph(2, 511)) \\
 arr(3, 3q_1) &= 142 \oplus f(64) \\
 arr(3, 3q_1) &= 142 \oplus 239 \\
 arr(3, 3q_1) &= 97 \\
 ((97 - 3) \text{ MOD } 3) &= 1 \\
 ((97 + 509) \text{ MOD } 3) &= 0 \\
 ((97 + 1021) \text{ MOD } 3) &= 2
 \end{aligned}$$

Then

$$\begin{aligned}
 a &= 202 \\
 ((97 + 252) \text{ MOD } 3) &= 1 \\
 ((97 + 764) \text{ MOD } 3) &= 0 \\
 ((97 + 1276) \text{ MOD } 3) &= 2 \\
 b &= 287 \\
 q_1 &\in \{202, 287\}
 \end{aligned}$$

The overlap of the first and the second solutions is at

$$\mathbf{q_1 = 202}$$

To calculate the exact value of the dynamic key p_1 , for the third decrypted pixel, using (12):

$$\begin{aligned}
 arr(p_1, p_1q_1 + 1) &= ciph(0, 1) \oplus f(ciph(0, 0)) \\
 arr(p_1, p_1q_1 + 1) &= 87 \oplus f(45) \\
 arr(p_1, p_1q_1 + 1) &= 87 \oplus 171 \\
 arr(p_1, p_1q_1 + 1) &= 252 \\
 p_1 &\in \{1, 3, 252 \times 2 - 1, 252 \times 2\} \\
 p_1 &\in \{1, 3, 503, 504\}
 \end{aligned}$$

For $p_1 = 1$

$$arr(1, 203) = 204$$

So this value is rejected For $p_1 = 3$

$$arr(3, 202 \times 3 + 1) = arr(3, 95) = \text{Mod}(\text{Mod}(95, 255) + 3, 255) = 98$$

also, this value is rejected

$$\begin{aligned}
 arr(2p_1, 2p_1q_1 + 2) &= ciph(0, 2) \oplus f(ciph(0, 1)) \\
 arr(2p_1, 2p_1q_1 + 2) &= 202 \oplus f(87) \\
 arr(2p_1, 2p_1q_1 + 2) &= 202 \oplus 61 \\
 arr(2p_1, 2p_1q_1 + 2) &= 247 \\
 p_1 &\in \{247, 503\}
 \end{aligned}$$

The overlap of the first and the second solutions is at

$$\mathbf{p_1 = 503}$$

Now, the encrypted Lena image is decrypted using these dynamic values ($q_1 = 202, p_1 = 503$).

References

1. Alsmirat MA, Al-Alem F, Al-Ayyoub M, Jararweh Y, Gupta B (2018) Impact of digital fingerprint image quality on the fingerprint recognition accuracy. *Multimedia Tools Appl* 1–40. <https://doi.org/10.1007/s11042-017-5537-5>
2. Alvarez G, Li S (2006) Some basic cryptographic requirements for chaos-based cryptosystems. *Int J Bifurcation Chaos* 16(08):2129–2151
3. Anderson R (2001) *Security engineering: a guide to building dependable distributed systems* 2001
4. Biham E, Shamir A (1991) Differential cryptanalysis of DES-like cryptosystems. *J Cryptol* 4(1): 3–72
5. Chen G, Mao Y, Chui CK (2004) A symmetric image encryption scheme based on 3d chaotic cat maps. *Chaos, Solitons & Fractals* 21(3):749–761
6. Chen X, Huang X, Li J, Ma J, Lou W, Wong DS (2015) New algorithms for secure outsourcing of large-scale systems of linear equations. *IEEE Trans Inf Forensics Secur* 10(1):69–78
7. Cokal C, Solak E (2009) Cryptanalysis of a chaos-based image encryption algorithm. *Phys Lett A* 373(15):1357–1360
8. Eisenbarth T, Kumar S, Paar C, Poschmann A, Uhsadel L (2007) A survey of lightweight-cryptography implementations. *IEEE Des Test Comput* 24(6):522–533
9. El-Wahed MA, Mesbah S, Shoukry A (2008) Efficiency and security of some image encryption algorithms. In: *Proceedings of the world congress on engineering*, vol 1, London, pp 2–4
10. Ge X, Liu F, Lu B, Wang W (2011) Cryptanalysis of a spatiotemporal chaotic image/video cryptosystem and its improved version. *Phys Lett A* 375(5):908–913
11. Haleem M, Mathur C, Chandramouli R, Subbalakshmi K (2007) Opportunistic encryption: a trade-off between security and throughput in wireless networks. *IEEE Trans Dependable Secure Comput* 4(4):313–324
12. Healy SJ, Quiles S, Von Arx JA (2007) Cryptographic authentication for telemetry with an implantable medical device. US Patent 7,228,182
13. Huang Z, Liu S, Mao X, Chen K, Li J (2017) Insight of the protection for data security under selective opening attacks. *Inf Sci* 412:223–241
14. Kadir R, Shahril R, Maarof MA (2010) A modified image encryption scheme based on 2d chaotic map. In: *2010 international conference on computer and communication engineering (ICCCCE)*. IEEE, pp 1–5
15. Kahn D (1996) *The codebreakers: the comprehensive history of secret communication from ancient times to the internet*. Simon and Schuster, New York
16. Karim L, Anpalagan A, Nasser N, Almhana J, Woungang I (2014) Fault tolerant, energy efficient and secure clustering scheme for mobile machine-to-machine communications. *Transactions on Emerging Telecommunications Technologies* 25(10):1028–1044
17. Katz J, Lindell Y (2008) *Introduction to modern cryptography*. CRC Press, Boca Raton
18. Kavitha T, Sridharan D (2010) Security vulnerabilities in wireless sensor networks: a survey. *Journal of Information Assurance and Security* 5(1):31–44
19. Li S, Li C, Chen G, Zhang D, Bourbakis NG (2004) A general cryptanalysis of permutation-only multimedia encryption algorithms. *IACR's Cryptology ePrint Archive: Report 374:2004*
20. Li C, Liu Y, Zhang LY, Chen MZ (2013) Breaking a chaotic image encryption algorithm based on modulo addition and xor operation. *Int J Bifurcation Chaos* 23(04):1–12
21. Li C, Liu Y, Zhang LY, Wong KW (2014) Cryptanalyzing a class of image encryption schemes based on chinese remainder theorem. *Signal Process Image Commun* 29(8):914–920
22. Li J, Huang X, Li J, Chen X, Xiang Y (2014) Securely outsourcing attribute-based encryption with checkability. *IEEE Trans Parallel Distrib Syst* 25(8):2201–2210
23. Li J, Li J, Chen X, Jia C, Lou W (2015) Identity-based encryption with outsourced revocation in cloud computing. *IEEE Trans Comput* 64(2):425–437
24. Li J, Yu C, Gupta B, Ren X (2018) Color image watermarking scheme based on quaternion hadamard transform and schur decomposition. *Multimedia Tools and Applications* 77(4):4545–4561
25. Lian S, Sun J, Wang Z (2005) Security analysis of a chaos-based image encryption algorithm. *Physica A: Statistical Mechanics and its Applications* 351(2):645–661
26. Liu H, Liu Y (2014) Security assessment on block-cat-map based permutation applied to image encryption scheme. *Opt Laser Technol* 56:313–316
27. Maleki F, Mohades A, Hashemi SM, Shiri ME (2008) An image encryption system by cellular automata with memory. In: *Third international conference on availability, reliability and security*, 2008. ARES 08. IEEE, pp 1266–1271

28. Mao Y, Chen G, Lian S (2004) A novel fast image encryption scheme based on 3D chaotic Baker maps. *Int J Bifurcation Chaos* 14(10):3613–3624
29. Nadeem A, Javed MY (2005) A performance comparison of data encryption algorithms. In: 2005 international conference on information and communication technologies. IEEE, pp 84–89
30. Ngo HH, Wu XP, Le PD, Wilson C (2008) A method for authentication services in wireless networks. In: AMCIS 2008 Proceedings, p 177
31. Norouzi B, Mirzakhaki S, Seyedzadeh SM, Mosavi MR (2014) A simple, sensitive and secure image encryption algorithm based on hyper-chaotic system with only one round diffusion process. *Multimedia Tools and Applications* 71(3):1469–1497
32. Ozkaynak F, Ozer AB, Yavuz S (2012) Cryptanalysis of a novel image encryption scheme based on improved hyperchaotic sequences. *Opt Commun* 285(24):4946–4948
33. Patidar V, Pareek N, Sud K (2009) A new substitution–diffusion based image cipher using chaotic standard and logistic maps. *Commun Nonlinear Sci Numer Simul* 14(7):3056–3075
34. Petitcolas FA (1883) *La cryptographie militaire*
35. Rhouma R, Belghith S (2008) Cryptanalysis of a new image encryption algorithm based on hyper-chaos. *Phys Lett A* 372(38):5973–5978
36. Rhouma R, Solak E, Belghith S (2010) Cryptanalysis of a new substitution–diffusion based image cipher. *Commun Nonlinear Sci Numer Simul* 15(7):1887–1892
37. Schneier B et al (1996) *Applied cryptography: protocols, algorithms and source code in C*
38. Shannon CE (1949) *Communication theory of secrecy systems*. *Bell Syst Tech J* 28(4):656–715
39. Singh S (2000) *The code book: the science of secrecy from ancient Egypt to quantum cryptography*. Fourth Estate, London, p 402
40. Solak E, Çokal C (2011) Algebraic break of image ciphers based on discretized chaotic map lattices. *Inf Sci* 181(1):227–233
41. Solak E, Çokal C, Yıldız OT, Biyikoğlu T (2010) Cryptanalysis of Fridrich’s chaotic image encryption. *Int J Bifurcation Chaos* 20(05):1405–1413
42. Wang X, He G (2011) Cryptanalysis on a novel image encryption method based on total shuffling scheme. *Opt Commun* 284(24):5804–5807
43. Wang X, Luan D, Bao X (2014) Cryptanalysis of an image encryption algorithm using chebyshev generator. *Digital Signal Process* 25:244–247
44. Wu Y, Noonan JP, Aгаian S (2011) NPCR and UACI randomness tests for image encryption. *Cyber Journals: Multidisciplinary Journals in Science and Technology. Journal of Selected Areas in Telecommunications (JSAT)* 1(2):31–38
45. Xiang T, Liao X, Tang G, Chen Y, Wong K-w (2006) A novel block cryptosystem based on iterating a chaotic map. *Phys Lett A* 349(1–4):109–115
46. Xie EY, Li C, Yu S, Lü J (2017) On the cryptanalysis of fridrich’s chaotic image encryption scheme. *Signal Process* 132:150–154
47. Yaseen Q, Aldwairi M, Jararweh Y, Al-Ayyoub M, Gupta B (2017) Collusion attacks mitigation in internet of things: a fog based model. *Multimedia Tools Appl* 1–20. <https://doi.org/10.1007/s11042-017-5288-3>
48. Yu C, Li J, Li X, Ren X, Gupta B (2018) Four-image encryption scheme based on quaternion fresnel transform, chaos and computer generated hologram. *Multimedia Tools and Applications* 77(4):4585–4608
49. Zhang Y, Xiao D (2013) Cryptanalysis of s-box-only chaotic image ciphers against chosen plaintext attack. *Nonlinear Dyn* 72(4):751–756
50. Zhang LY, Li C, Wong KW, Shu S, Chen G (2012) Cryptanalyzing a chaos-based image encryption algorithm using alternate structure. *J Syst Softw* 85(9):2077–2085
51. Zhang Y, Li C, Li Q, Zhang D, Shu S (2012) Breaking a chaotic image encryption algorithm based on perceptron model. *Nonlinear Dyn* 69(3):1091–1096
52. Zhang W, Wong K-w, Yu H, Zhu Z-l (2013) An image encryption scheme using reverse 2-dimensional chaotic map and dependent diffusion. *Commun Nonlinear Sci Numer Simul* 18(8):2066–2080
53. Zhang Y, Xiao D, Wen W, Li M (2014) Breaking an image encryption algorithm based on hyper-chaotic system with only one round diffusion process. *Nonlinear Dyn* 76(3):1645–1650
54. Zhang Y, Xiao D, Wen W, Li M (2014) Cryptanalyzing a novel image cipher based on mixed transformed logistic maps. *Multimedia Tools and Applications* 73(3):1885–1896
55. Zhang Y, Xiao D, Wen W, Nan H (2014) Cryptanalysis of image scrambling based on chaotic sequences and vigenère cipher. *Nonlinear Dyn* 78(1):235–240
56. Zhao Q, Wang Y, Wang A (2009) Eavesdropping in chaotic optical communication using the feedback length of an external-cavity laser as a key. *Appl Opt* 48(18):3515–3520



Dr. Mousa Farajallah received his Ph.D. degree in Computer Engineering from NANTES University and INSA of Rennes University, France in 2015 with exceptional (highest rating in France system). Farajallah Ph.D. thesis in crypto-compression solutions of High-Efficiency Video Coding (HEVC), and crypto solutions for real-time applications. Previously, in 2012 he studied cryptography course from Saarland University in Germany as Pre-PhD course for cryptography filed. From 2007 to 2010 master of Electronics and Computer Engineering from AL-Quds university in Jerusalem with Excellent rating (highest rating in AL-Quds University system), and his B.Eng. in Computer Systems Engineering from Palestine Polytechnic University (PPU) between 2001 and 2006 with high honor and distinguish (highest rating in PPU system). Currently, Farajallah is the head of the Computer Engineering and Security department at PPU and Information Security Consultant. Farajallah research interests includes: Cryptography, Cryptanalysis, and Crypto-Compression solutions. Farajallah has served 8 master students, 30 graduation projects, and since 2015, he published 3 ISI papers, 3 Scopus papers, and 3 high rank conferences, also he is an active reviewer for 5 ISI journals. Farajallah is an active member of many international conferences and workshops.



Dr. Safwan El Assad (Associate Professor) received his PhD degree in electrical engineering from the University of Lille 1, France in 1987. His doctoral thesis was on electromagnetic compatibility. He joins the University of Nantes, France in September 1987, where he is now an Associate Professor. From 1988 to 1996, his main interest was in radar imaging, remote sensing, signal and images processing. From 1996 until 2002, he was interested on digital communications, adaptive equalization for digital channels by neural network, and e-learning. Now his current research is in chaos-based exchanged and stored information security, including: Chaos-based crypto and crypto-compression systems for secure transmitted and stocked data; chaos-based watermarking and steganography; Chaos-based keyed hash functions. He has graduated 13 PhDs and 21 Master students. He published (as an author, co-author) 3 patents, 33 international journals, 6 invited papers 4 book-chapters and 125 articles in international conferences. He coordinated and participated on several French national, regional projects and European projects. He co-organized and co-chaired 8 workshops and special sessions on chaos-based data hiding and security and he chaired 27 sessions on information security. Applications: Transactions Security: IoT security, IP over satellite DVB, UMTS, wireless network security, video security, copyright, data integrity and authentication, etc.



Dr. Olivier Deforges is a professor at National Institute of Applied Sciences of Rennes (INSA). He received a Ph.D. degree in image processing in 1995. In 1996, he joined the Department of Electronic Engineering at the INSA of Rennes, Scientific and Technical University. He is a member of the Institute of Electronics and Telecommunications of Rennes (IETR), UMR CNRS 6164 and leads the IMAGE team of the IETR laboratory (40 peoples). O. Deforges authored more than 130 technical papers. His principal research interests are image and video lossy and lossless compression, image understanding, fast prototyping, and parallel architectures. O. Deforges has also been involved the ISO MPEG standardization group since 2007.