Public key cryptosystem based on multiple chaotic maps for image encryption

Yousif S. Najaf, Maher K. Mahmood Al-Azawi
Department of Electrical Engineering, Al-Mustansiriyah University, Iraq

Article Info

Article history:

Received Dec 19, 2020
Revised Apr 4, 2021
Accepted Apr 27, 2021

Keywords:

Chebyshev map
Cryptosystem
Image encryption
Multiple chaotic maps
Public key

ABSTRACT

Image is one of the most important forms of information. In this paper, two public key encryption systems are proposed to protect images from various attacks. Both systems depend on generating a chaotic matrix (I) using multiple chaotic maps. The parameters for these maps are taken from the shared secret keys generated from Chebyshev map using public keys for Alice and secret key for Bob or vice versa. The second system has the feature of deceiving the third party for searching for fake keys. Analysis and tests showed that the two proposed systems resist various attacks and have very large key space. The results are compared with other chaos based systems to show the superiority of these two proposed systems.

Corresponding Author:

Yousif S. Najaf
Department of Electrical Engineering
Al-Mustansiriyah University
Baghdad, Iraq
Email: yousifs.jaafar@uokufa.edu.iq

1. INTRODUCTION

In recent years, electronic communication using the internet played important role in various areas of life. Therefore, a security system must be provided to enhance the reliability of electronic communication. The main target from it is to protect information from any possible attacks. One of the important forms of information is the image, which is considered a basic form in many communication applications. Therefore, several image encryption techniques have been designed over years. Some of these techniques include optical encryption [1] block-based encryption [2], [3] decomposition encryption [4], public key and distributive key encryption [5], [6] deoxyribonucleic acid (DNA) and genetic encryption [7], digital image encryption [8], [9] and others.

Recently, many researchers used chaos in encrypting images since chaotic systems have high sensitivity to initial conditions and parameters. Where [10]-[12] proposed new chaotic maps to design image encryption systems with high security and better performance. Wang *et al.* [13] proposed a colour image encryption approach based on chaos. Wang *et al.* [14] used a dynamic random growth technique for an image encryption system. Hua *et al.* [15] proposed image encryption based on a 2D sine logistic modulation map. Wang *et al.* [16] proposed a chaotic image encryption algorithm based on the perceptron model. Wu *et al.* [17] proposed image encryption using the two-dimensional logistic chaotic map. On the other hand, the encryption of information without the need to transfer the key was first introduced by Diffie and Hellman [18]. They proposed what is known as public-key encryption systems, where two keys are used, one of them is public used to encrypt the information and the other is privately used to decrypt the ciphertext. The ability to calculate the private key from the public is almost very difficult if not impossible.

Usually, the public key encryption based on the chaotic map used the Chebyshev map due to the semi-group property of this map [19]. Prasadh *et al*. [20] designed a public-key cryptosystem based on the Chebyshev map to encrypt the image. Kocarev *et al*. [21], [22] were the first who proposed the use of the Chebyshev map in public-key encryption but the use of this map made the algorithms suffer from multi-keys problem (more than one key may decrypt the system) [23], [24]. This problem was solved by [25] by introducing a new parameter to the original algorithm.

The public key cryptosystems based on chaotic maps have many diffrences over the symmetric keys cryptosystems based on chaotic maps [10]-[17], the high security property due to the unnecessity to share the keys periodically (daily, weekly, and monthly) in a very very secure way. The differences are also in the general structure of the algorithms of the system because of the number of keys, in addition to the ease of connecting for anyone by knowing its public key with the asymmetric key system. This paper proposed two new public key cryptosystems for image encryption based on multiple chaotic maps (to make the cryptosystems more complicated aginst the various attacks), where both systems depend on generating a chaotic matrix using multiple chaotic maps. The parameters for these maps are derived from the shared secret keys generated from the Chebyshev map. The paper is arranged as follows: The section 2 introduces some well-known chaotic maps. The section 3 explains how to recruit chaotic maps in the two proposed algorithms and introduces the two proposed algorithms in details. Section 4 gives some statistical analysis for the two proposed algorithms and comparisons the result with some chaotic-based systems and computes key space and sensitivity. Finally, in section 5, a conclusion is given.

2. SOME WELL-KNOWN CHAOTIC MAPS

The proposed cryptosystems employ four chaotic maps in this paper, logistic map [2], tent map [11], quadratic map and Chebyshev map [10].

a. Logistic map

This map is one of the most common chaotic maps. It is defined by the following difference equation, where the chaotic behavior of the map is obtained when $\mu \in [3.57,4]$ and the chaotic sequence $X \in [0,1]$:

$$X_{n+1} = X_n \, \mu(1 - X_n) \tag{1}$$

b. Tent map

This is defined by the following difference equation:

$$X_{n+1} = \mu \, min(X_n, 1 - X_n) \tag{2}$$

c. Quadratic map

This is defined by the following difference equation, where the chaotic behavior of the map is obtained when $\mu \in [1.5,2]$ with chaotic sequence $X \in [-2,2]$.

$$X_{n+1} = \mu - ({X_n}^2) \tag{3}$$

d. Chebyshev map

Chebyshev polynomial map is defined as (4),

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x) \tag{4}$$

where (n) is the order for the polynomial and (x) is the variable, with initials $T_0(x) = 1$ and $T_1(x) = x$, some next orders are: $for\ n = 2,\ T_2(x) = 2x^2 - 1,\ for\ n = 3,\ T_3(x) = 4x^3 - 3x$ and so on. In order to deal with integers, a simple modification of the basic map is made by adding modN to handle a specific field where the definition of the modular map becomes:

$$Y = T_n(x) mod\ N \tag{5}$$

3. THE PROPOSED ALGORITHMS

This section describes how to recruit chaotic maps in the two proposed algorithms by explaining the purpose of using them and the method of generation chaotic matrix (I) then describes the structure of the two proposed algorithms in details.

3.1. Recruit chaotic maps in the proposed algorithms

Two reasons are behind the use of chaotic maps in the proposed algorithm. The first one is to use the Chebyshev map specifically in order to generate secret shared keys between Alice and Bob, the second one is to use other chaotic maps in order to generate a chaotic matrix by using these secret shared keys.

3.1.1. Secret shared keys generated by using chebyshev map

The Chebyshev map used specifically in order to generate secret shared keys between Alice and Bob, where the generation of the secret shared keys depends on the modified algorithm introduced by [25] in order to solve the multi-keys problem.

3.1.2. Chaotic matrix generated by using chaotic map

This subsection explains the generation of the chaotic matrix from secret shared keys and the mathematical formula used for the chaotic matrix in expressive form.

a. Generation of the chaotic matrix

In most encryption algorithms, image encryption depends on generating a random matrix, as each pixel in this random matrix is used to encrypt the corresponding pixel in the original image. The proposed algorithm is based on generating a chaotic matrix from the chaotic maps. The size of the chaotic matrix is the same as the size of the original image. Let's generate the chaotic matrix (I) by using say logistic chaotic map for example, after choosing (μ, X_n) the first value in chaotic matrix computed from the logistic map is $I(1,1) = X_n\,\mu(1 - X_n)$ the second value in the matrix is $I(1,2) = X_{n+1}\,\mu(1 - X_{n+1})$ and the next value in the matrix is $I(1,3) = X_{n+2}\,\mu(1 - X_{n+2})$ and so on.To make the generation process more complicated, a new parameter (K) is added that represents a multiplicity number to the map before producing the output. For example, if $K = 2$ then the first value in chaotic matrix computed from logistic map is is $I(1,1) = X_{n+2} = X_{n+1}\,\mu(1 - X_{n+1})$ the second value in matrix is $I(1,2) = X_{n+4} = X_{n+3}\,\mu(1 - X_{n+3})$ and the next value in matrix is $I(1,3) = X_{n+6} = X_{n+5}\,\mu(1 - X_{n+5})$ and so on. The same applies if $K = 100$ where the first value in chaotic matrix computed from logistic map is $I(1,1) = X_{n+100} = X_{n+99}\,\mu(1 - X_{n+99})$ the second value in matrix is $I(1,2) = X_{n+200} = X_{n+199}\,\mu(1 - X_{n+199})$ and the next value in matrix is $I(1,3) = X_{n+300} = X_{n+299}\,\mu(1 - X_{n+299})$ and so on .

b. Creating the chaotic matrix in expressive formula

The expressive formula that will be used for the chaotic matrix I creation process will be as (6),

$$I = Re\ [equation\ of\ chaotic\ map]_K \tag{6}$$

where (Re) represents the repetition of the map K times. It should be noted that the best results from these chaotic maps are obtained if the bifurcation diagram of the chaotic map is known. For example, the logistic map has the best result when μ is closer to 4 tent and quadratic maps have the best results when μ is close to 2. Hence $\mu = 3.99$ is chosen for logistic and $\mu = 1.99$ for tent and quadratic maps. The type of chaotic map used does not affect the results and that gives many choices available.

3.1.3. Scaling down and scaling up

In general, the secret shared keys generated from the Chebyshev map are integers. So in order to use some of them in any selected chaotic maps as initial parameters, they must be scaled to the range of maps used. This is called "scaling down". Also, the outputs of the selected chaotic maps must be scaled again for integer values of the chaotic matrix integers. This is called "scaling up". For example, if the logistic map is selected, then it's range is $(low = 0, high = 1)$ so the scaling down for secret shared key Q will be $[\frac{(high-low)(Q)}{(P)} + low]$ and the output will be scaled up as $Int[\frac{(output-low)(P)}{(high-low)} + low]$.

3.2. The general structure of two proposed algorithms

Two algorithms are suggested, the second algorithm differs from the first algorithm in some points only, but in the general structure they are very similar and consist of three basic stages:

a. The first stage is the keys generation stage, where Alice generates her private keys in addition to generating the public keys,
b. The second stage is the generation of shared secret keys and encryption, where Bob generates his secret keys he will use in addition to the public keys for Alice for the purpose of generating shared secret keys, and then uses the shared keys for the purpose of encryption the image.
c. The third stage is the generation of shared secret keys and decryption, where Alice generates the shared secret keys using her private keys and the cipher parameters are sent from Bob after that, Alice decrypts the cipher image**.** Figures 1 and 2 illustrate these stages in both algorithms respectively.

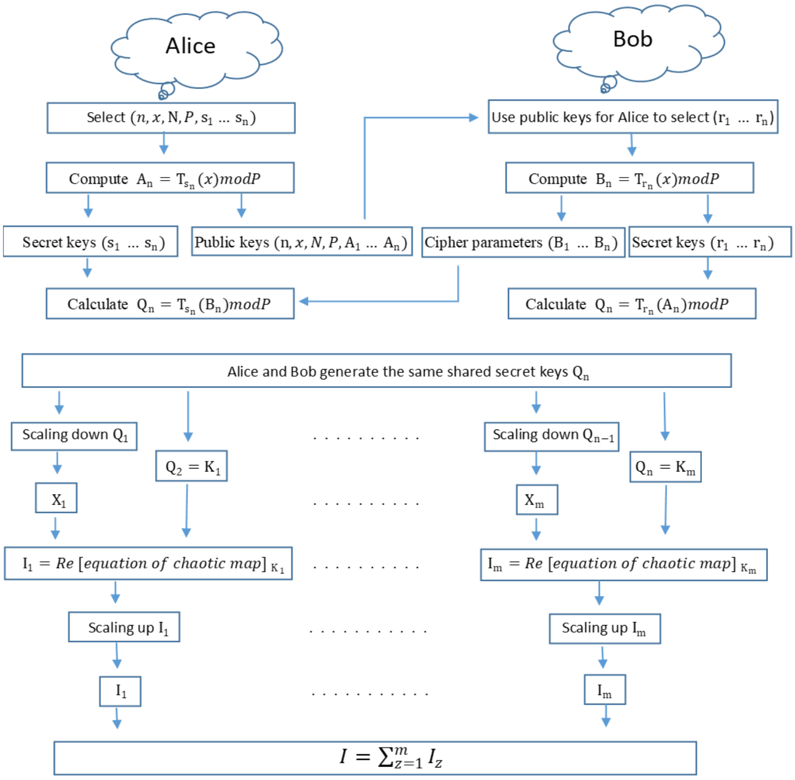

Alice
Bob
Select $(n, x, N, P, s_1 \ldots s_n)$
Use public keys for Alice to select $(r_1 \ldots r_n)$
Compute $A_n = T_{s_n}(x) modP$
Compute $B_n = T_{r_n}(x) modP$
Secret keys $(s_1 \ldots s_n)$
Public keys $(n, x, N, P, A_1 \ldots A_n)$
Cipher parameters $(B_1 \ldots B_n)$
Secret keys $(r_1 \ldots r_n)$
Calculate $Q_n = T_{s_n}(B_n) modP$
Calculate $Q_n = T_{r_n}(A_n) modP$
Alice and Bob generate the same shared secret keys Q_n
Scaling down Q_1
$Q_2 = K_1$
Scaling down Q_{n-1}
$Q_n = K_m$
X_1
X_m
$I_1 = Re\ [equation\ of\ chaotic\ map]_{K_1}$
$I_m = Re\ [equation\ of\ chaotic\ map]_{K_m}$
Scaling up I_1
Scaling up I_m
I_1
I_m
$I = \sum_{z=1}^{m} I_z$

Figure 1. Generation of chaotic matrix (I) in first algorithm when Bob sends a message to Alice

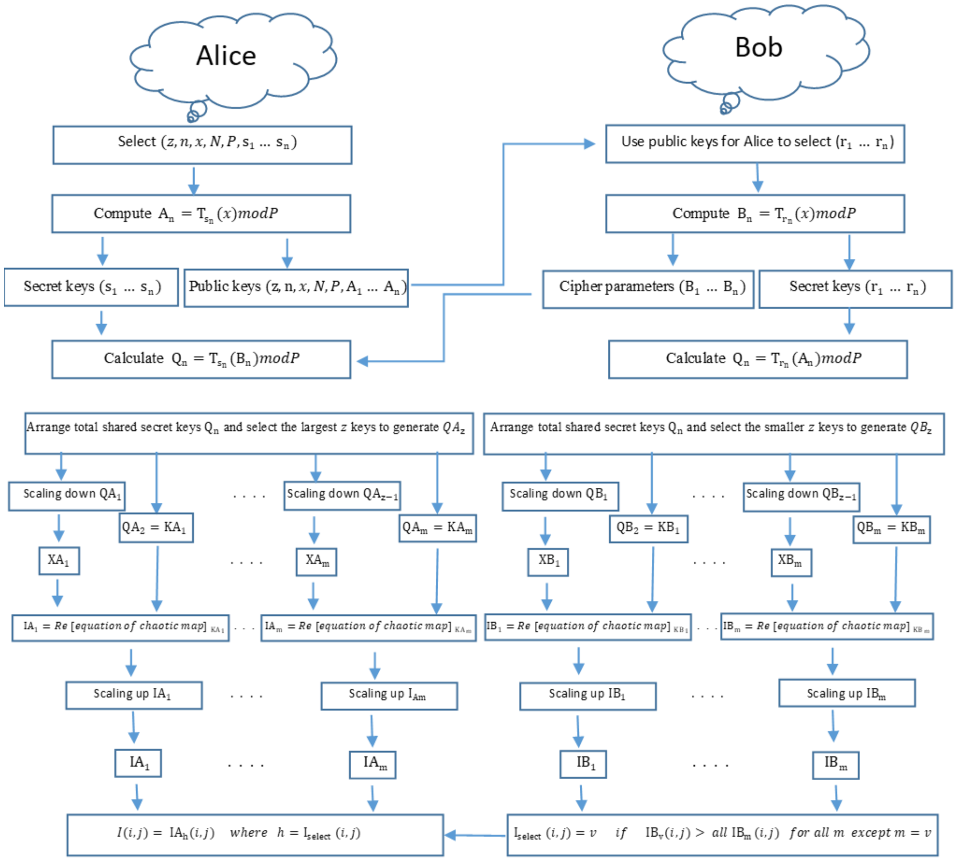

Alice
Bob
Select $(z, n, x, N, P, s_1 \ldots s_n)$
Use public keys for Alice to select $(r_1 \ldots r_n)$
Compute $A_n = T_{s_n}(x) modP$
Compute $B_n = T_{r_n}(x) modP$
Secret keys $(s_1 \ldots s_n)$
Public keys $(z, n, x, N, P, A_1 \ldots A_n)$
Cipher parameters $(B_1 \ldots B_n)$
Secret keys $(r_1 \ldots r_n)$
Calculate $Q_n = T_{s_n}(B_n) modP$
Calculate $Q_n = T_{r_n}(A_n) modP$
Arrange total shared secret keys Q_n and select the largest z keys to generate QA_z
Arrange total shared secret keys Q_n and select the smaller z keys to generate QB_z
Scaling down QA_1
$QA_2 = KA_1$
Scaling down QA_{z-1}
$QA_m = KA_m$
Scaling down QB_1
$QB_2 = KB_1$
Scaling down QB_{z-1}
$QB_m = KB_m$
XA_1
XA_m
XB_1
XB_m
$IA_1 = Re\ [equation\ of\ chaotic\ map]_{KA_1}$
$IA_m = Re\ [equation\ of\ chaotic\ map]_{KA_m}$
$IB_1 = Re\ [equation\ of\ chaotic\ map]_{KB_1}$
$IB_m = Re\ [equation\ of\ chaotic\ map]_{KB_m}$
Scaling up IA_1
Scaling up I_{Am}
Scaling up IB_1
Scaling up IB_m
IA_1
IA_m
IB_1
IB_m
$I(i,j) = IA_h(i,j)\quad where\quad h = I_{select}(i,j)$
$I_{select}(i,j) = v\quad if\quad IB_v(i,j) > all\ IB_m(i,j)\quad for\ all\ m\ except\ m = v$

Figure 2. Generation of chaotic matrix (I) in second algorithm when Bob sends a message to Alice

3.2.1. Description of the first proposed algorithm

a. Alice performs the following steps for stage 1:
   1) Choose a large integer number N and an appropriate integer number $x < N$.
   2) Choose a large prime number $P >> N$.
   3) Choose random integer numbers $(s_1, s_2, s_3, \dots, s_n)$, less than N, where n represents the number of secret keys to be chosen as an even number.
   4) Compute $A_n = T_{s_n}(x)\text{mod}P$.
   5) The public keys for Alice are $(x, N, P, A_1, A_2, A_3, \dots, A_n)$, and the private keys are $(s_1, s_2, s_3, \dots, s_n)$.

b. Bob performs the following steps for stage 2:
   1) Use the public key for Alice $(x, N, P, A_1, A_2, A_3, \dots, A_n)$.
   2) Choose a random integers $(r_1, r_2, r_3, \dots, r_n)$, less than N.
   3) Calculate $B_n = T_{r_n}(x)modP$ and $Q_n = T_{r_n}(A_n)modP$.
   4) Calculate $K_m = Q_n$ for even value of n, where $m = n/2$.
   5) Calculate X_m from scaling down Q_n for the odd value of n, where this scaling depends on the chaotic maps used in the next step.
   6) Generate chaotic matrix elements $I_m = scaling\ up\ (Re\ [equation\ of\ chaotic\ map]_{Km})$.
   7) Calculate $I = \sum_{z=1}^{m} I_z$.
   8) Calculate $C = (M + 1)mod256$ where M is the original image and C is the cipher image.
   9) Send the cipher parameters to Alice $(B_1, B_2, B_3, \dots, B_n)$, where these parameters are sent only once and not related to the plain image.
   10) Send the ciphered image to Alice (C).

c. Alice performs the following steps for stage 3:
   1) Use cipher parameters sent from Bob $(B_1, B_2, B_3, \dots, B_n)$ and her private keys $(s_1, s_2, s_3, \dots, s_n)$ to calculate $Q_n = T_{s_n}(B_n)modP$.
   2) Repeat steps 4-7 as in the second stage until all elements of (I) are calculated.
   3) Finally compute the original image $\text{M} = (\text{C-I})\text{mod}256$.

3.2.2. Description of the second proposed algorithm

a. Alice performs the following steps for stage (1):
   1) Choose a large integer number N and an appropriate integer number $x < N$.
   2) Choose a large prime number $P >> N$.
   3) Choose random integer number $(s_1, s_2, s_3, \dots, s_n)$, less than N, where n represents the number of total shared secret keys.
   4) Choose an even number z as, $4 \leq z \leq n$, where the number of actual shared secret keys will be $2z$.
   5) Compute $A_n = T_{s_n}(x)modP$.
   6) The public keys for Alice are $(z, x, N, P, A_1, A_2, A_3, \dots, A_n)$and the private keys are $(s_1, s_2, s_3, \dots, s_n)$.

b. Bob performs the following steps for stage (2):
   1) Use the public key for Alice $(z, x, N, P, A_1, A_2, A_3, \dots, A_n)$.
   2) Choose a random integers $(r_1, r_2, r_3, \dots, r_n)$ less than N.
   3) Calculate $B_n = T_{r_n}(x)modP$ and $Q_n = T_{r_n}(A_n)modP$.
   4) Arrange total shared secret keys and select the largest z keys to generate QA_z either the smaller z keys to generate QB_z.
   5) Calculate $KA_m = QA_z$and $KB_m = QB_z$ for even value of z.
   6) Calculate XA_m from scaling down QA_z and XB_m from scaling down QB_z for the odd value of z, where the scaling depends on the chaotic maps that are used in the next step.
   7) Generate matrix elements $IA_m = scaling\ up\ (Re\ [equation\ of\ chaotic\ map]_{KA_m})$ and $IB_m = scaling\ up\ (Re\ [equation\ of\ chaotic\ map]_{KB_m})$
   8) Calculate the selector matrix I_{select} from matrix IB_m, where $I_{select}(i,j) = v\ \ if\ IB_v(i,j) > all\ IB_m(i,j)\ for\ all\ m = v$.
   9) Calculate chaotic matrix I from matrixes I_{select} and IA_m, where $I(i,j) = IA_h(i,j), where\ h = I_{select}(i,j)$.
   10) Calculate $C = (M + I)mod256$, where M is the original image and C is the cipher image.
   11) Send the cipher parameters $(B_1, B_2, B_3, \dots, B_n)$ to Alice, where these parameters are sent only once and not related to the plain image.
   12) Send the cipher image to Alice (C).

c. Alice performs the following steps for stage (3):
   1) Use cipher parameters sent from Bob $(B_1, B_2, B_3, \dots, B_n)$ and her private keys $(s_1, s_2, s_3, \dots, s_n)$ to calculate $Q_n = T_{s_n}(B_n)modP$.

2) Repeat steps 4-9 as in the second stage until all elements of (I) are calculated.
3) Finally compute the original image $M = (C - I) mod 256$.

4. SECURITY ANALYSIS

In this section, the two algorithms were applied to a number of standard images to obtain the statistical analysis of the two algorithms by calculating (histogram, entropy, correlation of two adjacent pixels, MSE and PSNR) then compared these results with another chaos-based systems, in addition to compute the key space and sensitivity for both algorithms.

4.1. Statistical analysis

Some standard pictures (Lena, boat, Barbara and Man) are used together with the following tests (histogram analysis, entropy analysis, correlation of two adjacent pixels, mean square error (MSE) and peak signal to noise ratio analysis (PSNR). In order to carry out these tests and obtain the results, we need to know the private keys for both Bob and Alice and the public keys for Alice, in addition to the types of chaotic maps used. To make it clear, let us take the following numerical example for the purpose of systems analysis:

a. For the first algorithm, the public keys for Alice are $(x, N, P, A_1, \dots, A_4)$ the private keys are $(s_1, \dots, s_4)$, and the private keys for Bob are $(r_1, \dots, r_4)$. The cipher parameters $(B_1, \dots, B_4)$ are sent only once. Logistic map is used with $((X_1, K_1)$ and tent map is used with (X_2, K_2) as shown in Table 1.
b. For the second algorithm, the public keys for Alice are $(z, x, N, P, A_1, \dots, A_{20})$, the private keys are $(s_1, \dots, s_{20})$ and the private keys for Bob are $(r_1, \dots, r_{20})$. The cipher parameters are $(B_1, \dots, B_{20})$ set only once. The total shared secret keys are $(Q_1, \dots, Q_{20})$ and the actual shared secret keys $(QB_1, \dots, QB_4, QA_1, \dots, QA_4)$. The logistic map is used with (XA_1, KA_1, XA_2, KA_2) and the quadratic map is used with (XB_1, KB_1, XB_2, KB_2) as shown in Table 2.

Table 1. The assumed private keys, public keys and types of chaotic maps used in first algorithm

	$x = 120$ $N = 149$	$P = 29803$	
	$A_1 = 2439$	$r_1 = 25$	$B_1 = 4930$
$s_2 = 47$	$A_2 = 5334$	$r_2 = 68$	$B_2 = 10723$
$s_3 = 55$	$A_3 = 1382$	$r_3 = 37$	$B_3 = 4764$
$s_4 = 75$	$A_4 = 22135$	$r_4 = 67$	$B_4 = 27390$
	$X_1 = 506.157098278697e - 003$	$K_1 = 17904$	
	$X_2 = 467.134181122706e - 003$	$K_2 = 2502$	

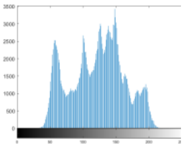
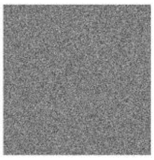
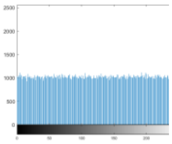
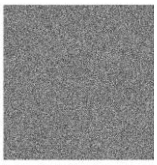
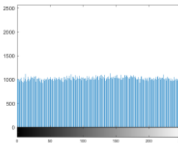
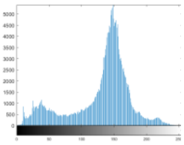
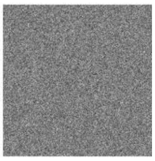
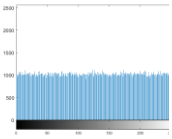
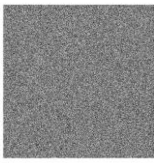
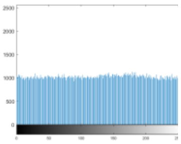
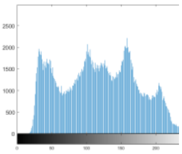
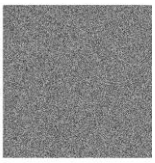
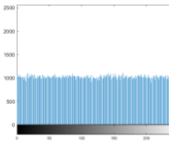
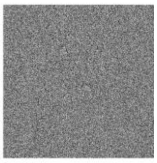
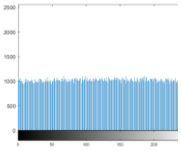
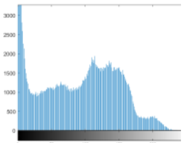
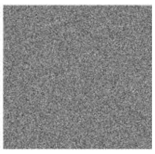
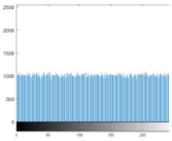
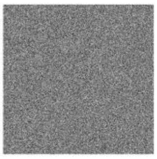
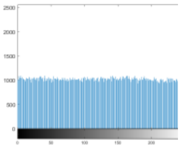
Table 2. The assumed the private keys, public keys and types of chaotic maps used in second algorithm

	$z = 4$	$x = 120$	$N = 149$	$P = 29803$
Random integers	Public keys	Random integers	Cipher parameters	Shared secret keys
$s_1 = 54$	$A_1 = 21202$	$r_1 = 48$	$B_1 = 11954$	$QB_4 = Q_1 = 5430$
$s_2 = 110$	$A_2 = 5063$	$r_2 = 80$	$B_2 = 18975$	$QB_1 = Q_2 = 202$
$s_3 = 59$	$A_3 = 18296$	$r_3 = 14$	$B_3 = 4110$	$QB_2 = Q_3 = 3674$
$s_4 = 102$	$A_4 = 25943$	$r_4 = 17$	$B_4 = 5295$	$QA_3 = Q_4 = 29187$
$s_5 = 105$	$A_5 = 18687$	$r_5 = 21$	$B_5 = 18230$	$Q_5 = 19810$
$s_6 = 66$	$A_6 = 6217$	$r_6 = 100$	$B_6 = 2625$	$Q_6 = 11772$
$s_7 = 3$	$A_7 = 27147$	$r_7 = 74$	$B_7 = 1422$	$Q_7 = 5963$
$s_8 = 50$	$A_8 = 1106$	$r_8 = 22$	$B_8 = 21080$	$Q_8 = 14811$
$s_9 = 64$	$A_9 = 22801$	$r_9 = 9$	$B_9 = 18571$	$Q_9 = 24621$
$s_{10} = 41$	$A_{10} = 22025$	$r_{10} = 127$	$B_{10} = 950$	$Q_{10} = 8338$
$s_{11} = 30$	$A_{11} = 17377$	$r_{11} = 44$	$B_{11} = 7339$	$Q_{11} = 10419$
$s_{12} = 123$	$A_{12} = 816$	$r_{12} = 139$	$B_{12} = 24282$	$Q_{12} = 7534$
$s_{13} = 65$	$A_{13} = 4343$	$r_{13} = 87$	$B_{13} = 4135$	$Q_{13} = 22047$
$s_{14} = 133$	$A_{14} = 8259$	$r_{14} = 25$	$B_{14} = 4930$	$QA_2 = Q_{14} = 28862$
$s_{15} = 59$	$A_{15} = 18296$	$r_{15} = 120$	$B_{15} = 1404$	$Q_{15} = 11855$
$s_{16} = 115$	$A_{16} = 20623$	$r_{16} = 96$	$B_{16} = 15264$	$QB_3 = Q_{16} = 4088$
$s_{17} = 60$	$A_{17} = 22068$	$r_{17} = 136$	$B_{17} = 5509$	$QA_4 = Q_{17} = 29556$
$s_{18} = 121$	$A_{18} = 9276$	$r_{18} = 32$	$B_{18} = 18700$	$QA_1 = Q_{18} = 24640$
$s_{19} = 113$	$A_{19} = 23300$	$r_{19} = 77$	$B_{19} = 27684$	$Q_{19} = 17220$
$s_{20} = 57$	$A_{20} = 5838$	$r_{20} = 18$	$B_{20} = 3449$	$Q_{20} = 12259$
	$XA_1 = 826.762406469147e - 003$		$KA_1 = 28862$	
	$XA_2 = 979.330939838271e - 003$		$KA_2 = 29556$	
	$XB_1 = -1.97288863537228$		$KB_1 = 3674$	
	$XB_2 = -1.45133040297957$		$KB_2 = 5430$	

4.1.1. Histogram analysis

Table 3 shows the histogram for a number of standard images (Lena, boat, Barbara and man) after and before encryption. The results of the test show that the histograms for cipher images are uniform and flat.

Table 3. Histograms for number of standard image (Lena, Boat, Barbara and Man)

Plain image	Histogram for plain image	Cipher image for first algorithm	Histogram for cipher image for first algorithm	Cipher image for second algorithm	Histogram for cipher image for second algorithm
					
					
					
					

4.1.2. Entropy analysis

The (7) and (8) are used to find the entropy as shown in [11]:

$$H(Y) = \sum_{j=1}^{n} \mathrm{pr}(yj)\log_2 \mathrm{pr(yj)} \tag{7}$$

$$\mathrm{pr}(Y = yj) = 1/\mathrm{L} \tag{8}$$

yj is the jth possible value in Y, and $\mathrm{pr}(yj)$ is the probability of $(Y = yj)$, i.e. the probability of pulling a random pixel in Y its value is yj. This $H(Y)$is maximum when Y is uniformly distributed. Also, L is the number of intensity levels.

4.1.3. Correlation of two adjacent pixels

The correlation of the encrypted image is calculated to find out the strength of encryption, and this is known as correlation analysis of two adjacent pixels. It is calculated as shown in [13]:

$$\mathrm{E(x)} = \frac{1}{\mathrm{N}}\sum_{\mathrm{i=1}}^{\mathrm{N}} \mathrm{x_i} \tag{9}$$

$$\mathrm{D(x)} = \frac{1}{\mathrm{N}}\sum_{\mathrm{i=1}}^{\mathrm{N}} (\mathrm{x_i} - E(\mathrm{x_i}))^2 \tag{10}$$

$$\mathrm{cov(x,y)} = \frac{1}{\mathrm{N}}\sum_{\mathrm{i=1}}^{\mathrm{N}} (\mathrm{x_i} - E(\mathrm{x_i}))(\mathrm{y_i} - E(\mathrm{y_i})) \tag{11}$$

$$\mathrm{r_{xy}} = \frac{\mathrm{cov(x,y)}}{\sqrt{\mathrm{D(x)}}\sqrt{\mathrm{D(y)}}} \tag{12}$$

where x, y are two adjacent pixels value in image test and $\mathrm{r_{xy}}$ is the correlation coefficient of two adjacent pixels.

4.1.4. Mean square error (MSE)

The MSE is calculated as shown in (13) [11]:

$$\mathrm{MSE} = \frac{1}{\mathrm{NM}}\sum_{i=1}^{N}\sum_{j=1}^{M}(\mathrm{m(i,j)} - c(\mathrm{i,j}))^2 \tag{13}$$

where $\mathrm{m(i,j)}$ is the value of pixel in plain text image and $c(\mathrm{i,j})$ is the value of pixel in encrypted image both at the same location $(\mathrm{i,j})$, NM is the size of (plain text or encrypted) image. The large MSE value indicates the quality of the encrypting.

4.1.5. Peak signal to noise ratio PSNR analysis

The $PSNR$ is calculated as follows [11]:

$$\mathrm{PSNR} = 10 \times \log_{10}\left[\frac{\mathrm{R}^2}{\mathrm{MSE}}\right] \tag{14}$$

where R is the maximum pixel value in an image. The less $PSNR$ is, the highest encryption strength is obtained.

4.2. Comparison of the two proposed systems with other chaos-based systems

In this subsection, the statistical results (entropies, correlation of two adjacent pixels, MSE and PSNR of encrypted images) for both proposed algorithms were compared with other chaos-based systems in Tables 4-7 respectively, where FPS referred to the first proposed algorithm and SPS referred to the second proposed algorithm. One can see that the results of both algorithms in the four tables are very close to the best value if it is not the best, i.e., they have very good immunity to various attacks.

Table 4. Comparison of Entropies of encrypted images

Image	FPS	SPS	Ref 11	Ref 12	Ref 13	Ref 14	Ref 15	Ref 16
Lena	7.9993	7.9991	7.9991	7.9963	7.9991	7.9951	7.9965	7.9964
Boat	7.9990	7.9991	7.9993	7.9980	7.9979	7.9960	7.9959	7.9985
Barbara	7.9992	7.9992	7.9993	7.9978	7.9964	7.9937	7.9957	7.9964
Man	7.9993	7.9992	7.9998	7.9975	7.9974	7.9990	7.9965	7.9949

Table 5. Comparison in terms of correlation of two adjacent pixels

Image	Position	FPS	SPS	Ref 11	Ref 12	Ref 13	Ref 14	Ref 15	Ref 16
Lena	Horizontal	0.0006	0.0007	−0.0047	−0.0047	−0.0086	−0.0066	0.0011	−0.0063
	Vertical	-0.0007	0.0005	0.0015	0.0015	−0.0102	−0.0089	0.0098	−0.0109
	Diagonal	-0.0016	-0.00001	0.0030	0.0030	−0.0125	0.0424	−0.0227	−0.0154
Boat	Horizontal	0.0030	0.0007	0.0025	−0.0100	−0.0054	−0.0189	−0.0295	−0.0138
	Vertical	-0.0022	0.00001	0.0018	−0.0124	−0.0009	0.0003	−0.0150	−0.0199
	Diagonal	0.00007	0.0020	0.0005	−0.0185	0.0026	−0.0204	−0.0224	−0.0057
Barbara	Horizontal	0.0004	0.0045	0.0033	−0.0033	−0.0052	−0.0212	−0.0187	0.0037
	Vertical	0.0019	0.0010	0.0032	-0.0269	−0.0067	−0.0161	−0.0016	−0.0202
	Diagonal	-0.0003	-0.0011	0.0025	-0.0121	0.0068	−0.0110	0.0001	0.0046
Man	Horizontal	0.0038	0.0007	−0.0010	−0.0155	−0.0190	−0.0083	0.0022	−0.0100
	Vertical	-0.0005	-0.0004	0.0029	−0.0276	−0.0095	−0.0180	−0.0226	−0.0027
	Diagonal	0.0008	-0.0009	−0.0005	−0.0157	−0.0141	0.0250	0.0060	−0.0195

Table 6. Comparison of MSE of encrypted images

Image	FPS	SPS	Ref 11	Ref 17
Lena	7151.51	7038.77	7694.30	7779.56
Boat	7658.36	7528.22	7532.69	7530.09
Barbara	8562.43	8434.24	9275.24	8285.23
Man	10,276.69	10,125.38	10,288.25	10,137.09

Table 7. Show the results of the comparison of PSNR of encrypted images

Image	FPS	SPS	Ref 11	Ref 17
Lena	9.58	9.65	9.30	9.26
Boat	9.28	9.36	9.40	9.40
Barbara	8.80	8.87	8.49	8.98
Man	8.01	8.07	8.04	8.11

4.3. Key space and sensitivity analysis

The purpose of implementing an image encryption system is to protect images from a third party. Hence, the third party should not be able to easily know the plain image from the encrypted image. This protection is related to the keys. The perfect encryption system is unbreakable when the third party searches for a correct key within a very large key space, close to infinite, which means that the probability of breaking the system is so small or close to zero. Hence, the strength of the system is related to the number of trials needed to find the correct key.

The very important issue in both proposed algorithms is that the key space for both algorithms depends on the parameter n (number of keys) that makes the two algorithms have very large key space. This makes the attack on the two systems is very complicated because the third party must not find only one correct key to break the system as in the single map cryptosystem [25] but he must find (n) correct keys. The key space of the two proposed systems was described according to the method of attack by the third party, either attacks the system by finding the secret keys or attacks the system by finding the shared secret keys directly.

In the first method of attack, the key space is related with the parameter (N) that limits the range of the selected private keys $(s_1 \ldots \mathrm{s_n})$, hence the third party must try to find the correct value of all these keys by searching within a set of N values. That makes the correct key has a probability equals to $\frac{1}{n \times N}$ to decrypt the system and the key space equals to $(n \times N)$ so the value of (N) must be chosen practically large enough to ensure the security of the system. In the second method of attack, the key space is related to the parameter (P) which has the same effect as (N). That makes the correct key has a probability equals to $\frac{1}{n\times P}$ to decrypt the system and the key space equals to $(n \times P)$ so the value of (P) must be chosen large enough to ensure the security of the system.

The key space for the second algorithm is larger than that for the first algorithm because the number of chaotic maps used in the first algorithm equals to $(n/2)$ while in the second algorithm it is equal to (z), where for each chaotic map need two shared secret keys. Therefore, in the second algorithm the value of (n) does not have a limit value, unlike in the first algorithm increasing (n) leads to the computational complexity for both the sender and the recipient.

Key sensitivity is very important in a cryptographic system i.e. a tiny change in the key produces a large difference. The chaotic map is very sensitive to small changes in both the initial conditions and the parameters. Figure 3 shows the high key sensitivity of proposed algorithms. where one of the secret keys s_1 is slightly changed from 38 to 39.

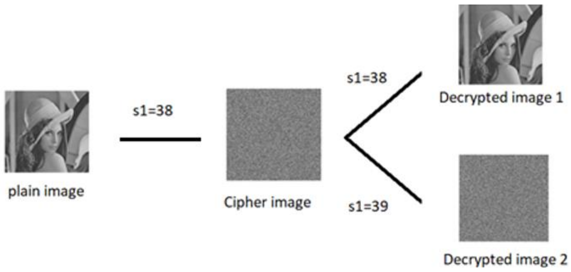


Figure 3. Simple sensitivity test

5. CONCLUSION

Multiple chaotic maps are used in two proposed public key cryptosystems for image encryption, the main advantage of using multiple chaotic maps in a cryptosystem instead of a single chaotic map is to make the attack on the systems very difficult. The second cryptosystem forces the third party to search for a large number of keys used for deception only which makes the cryptanalysis very complicated. Both systems construction was based on the Chebyshev map to generate the shared secret keys. These keys are then used to generate the chaotic matrix (I) by using any other chaotic maps (logistic, tent or Quadratic). Simulation tests and different statistical analysis showed that the two proposed systems are characterized by a high security (very large key space) to resist various attacks in addition to the high sensitivity of the keys, the results are compared with many chaos-based systems to show the superiority of these two proposed systems.

REFERENCE

[1] W. Chen, X. Chen, and C. J. R. Sheppard, "Optical image encryption based on diffractive imaging," *Optics Letters*, vol. 35, no. 22, pp. 3817-3819, Nov. 2010, doi: 10.1364/OL.35.003817.

[2] D. F. Chalob, A. A. Maryoosh, Z. M. Essa, and E. N. Abbud, “A new block cipher for image encryption based on multi chaotic systems,” *TELKOMNIKA Telecommunication Computing Electronics and Control*, vol. 18, no. 6, pp. 2983-2991, Dec. 2020, doi: 10.12928/telkomnika.v18i6.13746.

[3] J. A. E. Fouda, J. Y. Effa, S. L. Sabat, and M. Ali, “A Fast Chaotic Block Cipher for Imageencryption,” *Communications in Nonlinear Science and Numerical Simulation*, vol. 19, pp. 578-588, Mar. 2014, doi: 10.1016/j.cnsns.2013.07.016.

[4] H. Deng, Q. Zhu, X. Song, and J. Tao, “Chaos-based image encryption algorithm using decomposition,” *TELKOMNIKA Telecommunication Computing Electronics and Control*, vol. 12, no. 1, pp. 575-583, Jul. 2014, doi: 10.11591/telkomnika.v12i1.3018.

[5] J. I. Ahmad, R. Din, and M. Ahmad, “Analysis review on public key cryptography algorithms,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 12, no. 2, pp. 447-454, Nov. 2018, doi: 10.11591/ijeecs.v12.i2.pp447-454.

[6] S. R. M. Zeebaree, “DES encryption and decryption algorithm implementation based on FPGA,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 18, no. 2, pp. 774-781, May 2020, doi: 10.11591/ijeecs.v18.i2.pp774-781.

[7] Z. N. Al-kateeb and M. Jader, “Encryption and hiding text using DNA coding and hyperchaotic system,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 19, no. 2, pp. 766-774, Aug. 2020, doi: 10.11591/ijeecs.v19.i2.pp766-774.

[8] X. Wang and C. Tu, “A chaos-based medical image encryption method,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 19, no. 3, pp. 1316-1324, Sept. 2020, doi: 10.11591/ijeecs.v19.i3.pp1316-1324.

[9] S. M. T., E. Nurpeti, and D. Widya, “Performance of chaos-based encryption algorithm for digital image,” *TELKOMNIKA Telecommunication Computing Electronics and Control*, vol. 12, no. 3, pp. 675-682, Jul. 2014, doi: 10.12928/telkomnika.v12i3.106.

[10] B. Chaboki and A. Shakiba, “An image encryption algorithm with a novel chaotic coupled mapped lattice and chaotic image scrambling technique,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 21, no. 2, pp. 1103-1112, Feb. 2021, doi: 10.11591/ijeecs.v21.i2.pp1103-1112.

[11] R. Zahmoul, R. Ejbali, and M. Zaied, “Image encryption based on new beta chaotic maps,” *Optics and Lasers in Engineering*, vol. 96, pp. 39-49, Sept. 2017, doi: 10.1016/j.optlaseng.2017.04.009.

[12] A. Belazi, A. A. A. El-Latif, and S. Belghith, “A novel image encryption scheme based on substitution-permutation network and chaos,” *Signal Process*, vol. 128, pp. 155-170, Nov. 2016, doi: 10.1016/j.optlaseng.2017.04.009.

[13] X. Wang, L. Teng, and X. Qin, “A novel colour image encryption algorithm based on chaos,” *Signal Process,* vol. 92, no. 4, pp. 1101-1108, Apr. 2012, doi: 10.1016/j.sigpro.2011.10.023.

[14] X. Wang, L. Liu, and Y. Zhang, “A novel chaotic block image encryption algorithm based on dynamic random growth technique,” *Optics and Lasers in Engineering*, vol. 66, pp. 10-18, Mar. 2015, doi: 10.1016/j.optlaseng.2014.08.005.

[15] Z. Hua, Y. Zhou, C. M. Pun, and C. L. P. Chen, “2D sine logistic modulation map for image encryption,” *Information Sciences*, vol. 297, pp. 80-94, Mar. 2015, doi: 10.1016/j.ins.2014.11.018.

[16] X. Y. Wang, L. Yang, R. Liu, and A. Kadir, “A chaotic image encryption algorithm based on perceptron model,” *Nonlinear Dynamics*, vol. 62, no. 3, pp. 615-621, Nov. 2010, doi: 10.1007/s11071-010-9749-8.

[17] Y. Wu, G. Yang, H. Jin, and J. P. Noonan, “Image encryption using the two-dimensional logistic chaotic map,” *Journal of Electronic Imaging*, vol. 21, no. 1, pp. 1-15, Mar. 2012, doi: 10.1117/1.JEI.21.1.013014.

[18] W. Diffie and M. E. Hellman, “New directions in cryptography,” *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644-654, Nov. 1976, doi: 10.1109/TIT.1976.1055638.

[19] M. Lawnik and A. Kapczynski, “The application of modified chebyshev polynomials in asymmetric cryptography,” *Computer Science*, vol. 20, no. 3, Jan. 2019, doi: 10.7494/csci.2019.20.3.3307.

[20] K. Prasadh, K. Ramar, and R. Gnanajeyaraman, “Public key cryptosystems based on chaotic chebyshev polynomials,” *Journal of Engineering and Technology Research*, vol. 1, no. 7, pp. 122-128, 2009.

[21] L. Kocarev and Z. Tasev, “Public-key encryption based on chebyshev maps,” *IEEE International symposium on circuits systems*, 2003, pp. 28-31, doi: 10.1109/ISCAS.2003.1204947.

[22] L. Kocarev, J. Makraduli, and P. Amato, “Public-key encryption based on chebyshev polynomials,” *Circuits Systems and Signal Processing*, vol. 24, no. 5, pp. 497-517, 2005.

[23] D. Yoshioka, “Properties of chebyshev polynomials modulo pk,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 65, no. 3, pp. 386-390, 2018.

[24] X. Liao, F. Chen, and K.-W. Wong, “On the security of public-key algorithms based on chebyshev polynomials over the finite field Z_N,” *IEEE Transactions on Computers*, vol. 59, no. 10, pp. 1392-1401, 2010, doi: 10.1109/TC.2010.148.

[25] Y. S. Najaf, and M. K. Mahmood, “An improved public key cryptosystem based on chebyshev chaotic map over finite field,” *Scientific Conference for Graduate Engineering Students,* 2020, pp.314-324.