International Journal of Electrical and Computer Engineering (IJECE)
Vol. 12, No. 3, June 2022, pp. 3287~3297
ISSN: 2088-8708, DOI: 10.11591/ijece.v12i3.pp3287-3297

Combined Chebyshev and logistic maps to generate pseudorandom number generator for internet of things

Sameeh Abdulghafour Jassim, Alaa Kadhim Farhan
Department of Computer Sciences, University of Technology, Baghdad, Iraq

Article Info

Article history:

Received Jul 14, 2021
Revised Jan 15, 2022
Accepted Jan 31, 2022

Keywords:

Chaotic maps
Chebyshev maps
Pseudo-random number generator
Stream cipher
symmetric key

ABSTRACT

Sensitive data exchanging among things over the internet must be protected by a powerful cryptographic system. Conventional cryptographic such as advanced encryption standard (AES), and respiratory sinus arrhythmia (RSA) are not effective enough to protect internet of things (IoT) because of certain inveterate IoT properties like limited memory, computation, and bandwidth. Nowadays, chaotic maps with high sensitivity to initial conditions, strong ergodicity, and non-periodicity have been widely used in IoT security applications. So, it is suitable for IoT. Also, in a stream cipher method, the user needs to deliver the keystream to all clients in advance. Consequently, this paper proposed a method to solve the keys distribution problem based on combine both Chebyshev and logistic maps techniques as well as a master key to generate a random key. The suggested method was compared with the other stream cipher algorithms (Chacha20, RC4, Salsa20) by utilizing the same plaintext and master key as input parameters and the results were successful in the statistical national institute of standards and technology (NIST) test. Simultaneously, the suggestion was evaluated through different evaluation methods like statistical NIST test, histogram, Shannon entropy, correlation coefficient analysis, keyspace and key sensitivity, and others. All mentioned tests are passed successfully. Therefore, the suggested approach was proved it is effective in security issues.

Corresponding Author:

Sameeh Abdulghafour Jassim
Department of Computer Sciences, University of Technology
Baghdad, Iraq
Email: cs.19.22@grad.uotechnology.edu.iq

1. INTRODUCTION

Nowadays, information technology is permeated into virtually all areas. Therefore, the amount of sensitive and critical information carried via the internet in different dimensions has increased incredibly. As a result, fraudulent and illegal access attempts to private and confidential information are becoming more attractive. Consequently, in the digital world, data security is still a major issue [1]. Thus, cryptosystems are used to achieve the protection of sensitive data. There are two types of cryptosystems: asymmetric and symmetric [2], [3]. Asymmetric uses two various keys for encryption and decryption operations. While symmetric uses the identical private key in both encryption and decryption operations. Furthermore, symmetric-key cryptography is separated into two kinds: stream cipher and block cipher [4]. In this paper, we will focus on stream cipher, because it is more suitable for devices and applications of internet of things (IoT). In cryptography, generating random numbers is a significant issue. Random numbers are used in a variety of applications to generate unpredictable results, such as games, cryptography algorithms, and so on. Recently, several resource-constrained devices have been widely utilized. A resource-constrained system has insufficient energy sources, processing capacity, and storage space such as wireless sensors, and radio frequency identification (RFID) tag [5]. Because of their features, conventional ciphers are difficult to apply in this resource-constrained field. As a result, lightweight ciphers have gotten a lot of attention. Also, pseudorandom number generator (PRNGs) are a fundamental use of nonlinear chaotic systems in cryptography. A feedback shift register technique, including a carry forward feedback shift register, linear or nonlinear feedback shift register are the most popular ways to create PRNGs [6], [7].

In this paper, a chaotic system is utilized to produce a new PRNGs combined with the plain stream to generate a stream cipher. This method can be used for effective encryption in applications or areas with limited resources [8]. There are two types of chaotic maps currently available: one-dimensional and multi-dimensional chaotic maps [9]. One variable and several parameters are usually used in one-dimensional chaotic maps, such as Sine, Chebyshev, logistic, and Tent maps. Furthermore, it can create hybrid chaotic maps by combining several chaotic maps [10]. The Chebyshev map is a kind of one-dimensional chaotic map. Since the encryption computation demands a smaller key size. While the semi-group property of the Chebyshev chaotic map provides faster computation speed. Consequently, Chebyshev's chaotic map-based schemes used reduced battery life and small computation capacity. Therefore, it is more suited for IoT devices [11]. The logistic map is a one-dimensional recursive mapping that creates chaos in the system by producing pseudo-random numbers. Many security researchers have attempted to create techniques for generating stream cipher keys with high autocorrelation, frequency, and randomness. The most important and relevant to this paper's topic are: In [12] to strengthen the randomization process, the RC4 method is modified by employing hybrid chaotic maps that combine logistic and tent maps. In [13], a new S-Box based on a 1D logistic map chaotic system was designed. Ding *et al*. [8] used a chaotic method and two nonlinear feedback shift registers (NFSRs) to create a new stream cipher. In this paper, we apply these interesting chaotic maps of Chebyshev and logistic properties to the production of random numbers. The remainder of the paper is arranged as follows: in Section 2, chaotic systems are presented. In Sections 3 and 4 the proposed technique is implemented, and the experimental results are analyzed respectively. Section 5 gives the conclusion.

2. CHAOTIC SYSTEMS

Chaos theory is a branch of mathematics that deals with nonlinear and deterministic systems. It is more sensitive to its setup conditions, such as control parameters and initial values (seeds). As a result, a small change in its control parameters or seed causes a significant change in the chaotic outputs [14]. These features are linked to those of a good cipher in cryptography, like diffusion and confusion. Consequently, this has led to many security researchers using chaotic schemes to improve the security of many cryptographic systems [13]. Since the procedure for generating a single chaotic series is so simple, information security cannot be ensured [15]. Therefore, a one-dimensional chaotic sequence generation approach based on the Chebyshev and logistic maps is suggested. Experimental results show that the outputs of the chaotic sequence produced by the proposed method are pseudorandom.

2.1. Chebyshev chaotic system

The Chebyshev map is a kind of one-dimensional chaotic map that can be defined [16]:

$$xn + 1 = \cos(b \times \arccos xn) \tag{1}$$

where, b∈N is the control parameter; n is a non-negative integer. When b>1, the Chebyshev map displays chaotic behavior, and its outputs are limited to the interval [-1, 1]. Furthermore, when b∈[1, 2], the chaotic sequences' distribution is non-uniform. Figure 1 shows the result of applying the Chebyshev function by using b=37179 and x0=0.9 as the initial value.

2.2. Logistic chaotic system

Logistic mapping is a one-dimensional (1D) chaotic system and it is mostly used by researchers. The logistic map is a fundamental formula that describes both chaos and how it arises from well-organized actions. The (2) shows the structure of a logistic equation [15]:

$$f(x) = \mathrm{r}.x(1 - x) \tag{2}$$

where, (n) is a non-negative integer; (r) is a real number parameter ranging from zero to four; f(x) is a discrete population dynamic ranging from zero to one. Figure 1 shows the result of applying the logistic function by using r=3.6 and x0=0.37179 as the initial value.

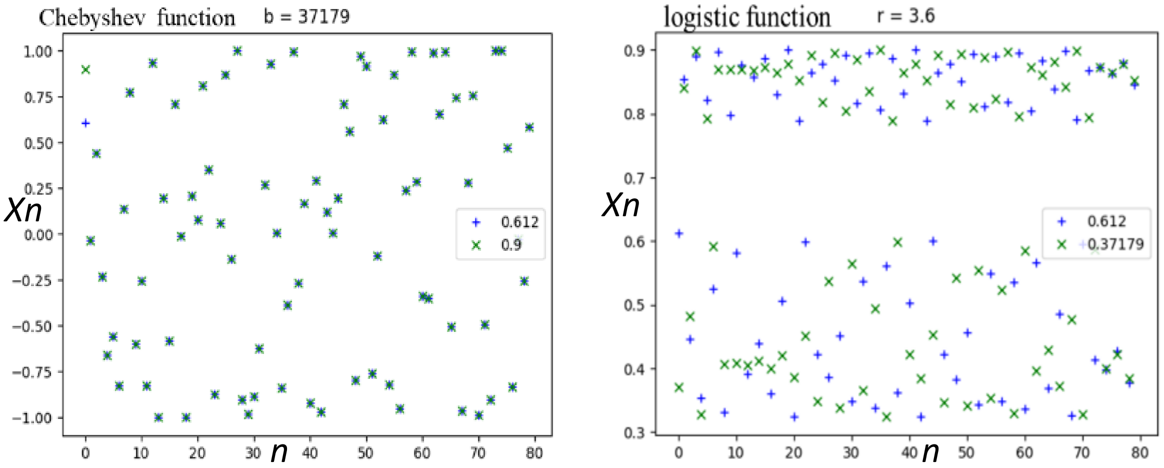


Figure 1. The Chebyshev and Logistic function results

3. THE PROPOSED TECHNIQUE

One of the most important criteria for the security of cryptographic schemes is key generation. The key must have three features to be secret: first, it must be unpredictable and not exposed; second, it must be not duplicated; third, the statistics produced should be statistically robust [17]. Besides, the PRNG gained from both the Chebyshev maps and the logistic maps separately is so simple. Consequently, information security cannot be ensured. Moreover, both the Chebyshev maps and the logistic maps are suitable for IoT environments because of their simplicity in implementation in hardware and software. However, both of them are suffering from trial-and-error attacks and correlation attacks if they are implemented separately. Therefore, a one-dimensional chaotic sequence generation approach based on the Chebyshev and logistic maps is suggested. The suggestion is based on using the same initial value as a master key, and as a seed for both Chebyshev and logistic maps. Accordingly, the proposed method will get the advantages of both Chebyshev and logistic maps and overcomes their weakness. The length of PRNG obtained from the suggested approach will be random and longer so, the trial-and-error strategy would be useless.

3.1. General description of the proposed technique

The proposed approach is illustrated in Figure 2(a) the client has used a secret master key as an input parameter. This master key is used as the initial value (seed) for both Chebyshev and logistic maps. The pros of this method are to minimize the numbers of the input parameters while the secret keys space is remaining long. Consequently, the security fundamentals are not affected. Whereas, Figure 2(b) illustrated the encryption process. The master key (K) and plaintext (P) must be translated to binary numbers. The binary master key is then tested to see if the first bit is one, then the first 32 bits of the Chebyshev key are XORed with the first 32 inverted bits of plaintext to produce the ciphertext (C) (using (4) as an equation of logical operations which was simplified from (3)). Otherwise, the first 32 bits of the logistic key are XORed with the first 32 inverted bits of plaintext to produce the ciphertext. And so on until the plaintext is over.

$$C = \left(\overline{\overline{P} + \overline{K}}\right) + \left(\overline{P + K}\right) + \left(\overline{P + \overline{K}}\ \right) \qquad (3)$$

This function could be simplified by the following laws:

$$C = \left(\overline{\overline{P}}.\overline{\overline{K}}\right) + (\overline{P}.\overline{K}) + (\overline{P}.K), \qquad DeMorgan's$$

$$C = P.K + \overline{P}.\overline{K} + (\overline{P}.K), \qquad Not\ law$$

$$C = \ P.K + \overline{P}.(\overline{K}.K), \qquad Factoring$$

$$C = P.K + \overline{P}, \qquad OR\ and\ AND\ law$$

$$C = (P + \overline{P}).(K + \overline{P}), \qquad Factoring$$

$$C = \overline{P} + K, \qquad and\ /Commutative\ laws.$$

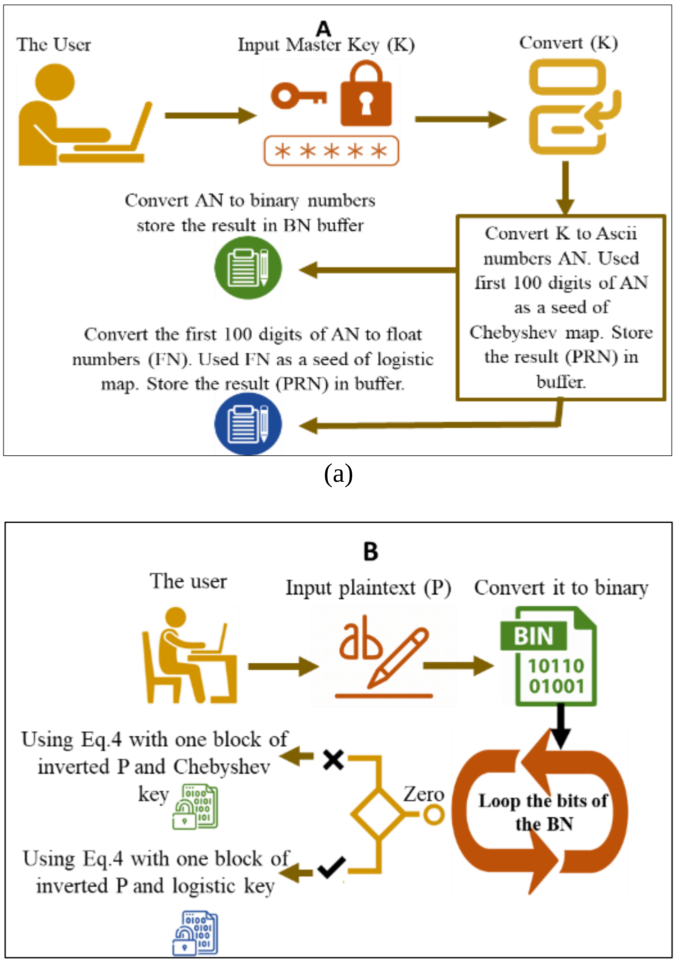
Therefore,

$$C = \bar{P} \oplus K \tag{4}$$

is used to encrypt messages. Whereas,

$$P = \overline{(C \oplus K)} \tag{5}$$

is used to decrypt the messages.


A
The User
Input Master Key (K)
*****
Convert (K)
Convert K to Ascii numbers AN. Used first 100 digits of AN as a seed of Chebyshev map. Store the result (PRN) in buffer.
Convert AN to binary numbers store the result in BN buffer
Convert the first 100 digits of AN to float numbers (FN). Used FN as a seed of logistic map. Store the result (PRN) in buffer.
B
The user
Input plaintext (P)
Convert it to binary
BIN
10110
01001
Loop the bits of the BN
Zero
Using Eq.4 with one block of inverted P and Chebyshev key
Using Eq.4 with one block of inverted P and logistic key

(a)

(b)

Figure 2. The main steps of the proposed technique are (a) initial values generation for Chebyshev and logistic maps and (b) encryption process

3.2. Obtaining the chaotic equation's initial values from the secret key

The master key should be converted into three forms: first, to binary numbers used to swap between Chebyshev and logistic maps; second, to ASCII number used as a seed of Chebyshev map; third, to float number used as a seed of logistic map. As shown in algorithms one and two.

Algorithm 1: seed values generation for Chebyshev map

```
input: text master key;
output: seed value of the decimal number.
begin
convert the text to ASCII numbers;
delete the spaces among the decimal numbers;
end.
```

For example: Password=x1y2z3; In ASCII=120 49 121 50 122 51; So, the master key without spaces will be 120491215012251

Algorithm 2: seed values generation for logistic map

```
1: input: text master key;
2: output: seed value of float number [0,1].
3: begin
4: convert the text to ASCII numbers;
5: delete the spaces among the decimal numbers;
6: convert the result of step 5: to float numbers;
7: end.
```

For example: Password=x1y2z3; In ASCII=120 49 121 50 122 51; So, the master key without spaces and in float form will be 0.120491215012251

Algorithm 3: Chebyshev map process

```
1: Input: x0=the output of the algorithm 1;
2: Output: binary random numbers; //store in buffer
3: Begin
4: control parameter b∈[0,1]
5: For i=0 to 255
6: xi+1=abs (cos(b×arccos xi))
7: Convert the result of step 6: from float to integer number;
8: End For
9: End
```

Algorithm 4: logistic map process

```
1: Input: x0=the output of the algorithm 2;
2: Output: array of binary random numbers; //store in buffer
3: Begin
4: control parameter r∈[0,4];
5: For i=0 to 255
6: Xi+1=r*xi*(1-xi)
7: Convert the result of step 6: from float to integer number;
8: End For
9: End
```

Algorithm 5: Keys mixing and the encryption process

```
1: Input: plaintext (P), master_key (K), the output of the algorithms one and two.
2: Output: ciphertext (C).
3: Begin
4: for i=0 to length(plaintext)
5: if (master_key[i]=='1'):
6: C=~(one block of P)⊕Chebyshev (4);
7: Else:
8: p=~(one block of P)⊕LogisticKey (4);
9: end if;
10: End.
```

4. THE EXPERIMENTAL RESULTS

This section displays the results of an experimental investigation conducted to determine the feasibility of the suggested method. We put the proposed method through different evaluation methods like statistical NIST test, histogram analysis, Shannon entropy analysis, correlation coefficient analysis, keyspace and key sensitivity, and others. All results were carried out on an HP laptop with an Intel(R) Core™ i7-8565U CPU running at 3.79 GHz and 8 GB of RAM, running Windows 10 (64-bit OS) and Python version 3.9.0, Tk version 8.6.9.

4.1. Randomness test for NIST statistical

As illustrated in Table 1 the sixteen samples of the statistical NIST test are applied to the output results of the proposed approach (more than one million bits). These measurements look for different forms of non-randomness, entropy, frequency, and runs test. That may be found in a sequence. All results are passed successfully the NIST test.

Whereas Table 2 shows the NIST test among the ciphertext output results of the proposed method and the ciphertext output results of the three other algorithms. In Table 2 the input length of the plaintext for the encryption process was 64 Byte. Despite the failure of some test results for some comparative algorithms, all allowed NIST test results of the proposed approach are passed successfully. In addition, the proposed procedure divides the input plaintext into blocks by converting it to binary numbers. Furthermore, the lengths of these blocks should be balanced. Since increasing the block length exacerbates the synchronization problem. Whereas, when the block length is short, the issue of swapping between keys is exacerbated. Accordingly, the proposed block length was 32 bits long.

Table 1. Results randomness of the proposed methods by encrypted 1,000,000 bits utilizing NIST metrics

Test Name		The results values	Status
Frequency test		0.813365	Succeed
Approximate entropy test		0.367138	Succeed
Test for the longest run of ones in a block		0.267548	Succeed
Frequency test within a block		0.651560	Succeed
Runs test		0.729884	Succeed
Discrete Fourier Transform (Spectral) Test		0.660905	Succeed
Linear-complexity		0.788286	Succeed
Random-excursions variant		0.979346	Succeed
Random-excursions		0.770630	Succeed
Binary matrix rank		0.717104	Succeed
Overlapping templates		0.599096	Succeed
Non-periodic templates		0.995789	Succeed
Maurer's 'universal statistical'		0.549671	Succeed
Lempel–Ziv compression		0.061457	Succeed
Serial test	P-v1	1.000000	Succeed
	P-v2	1.000000	Succeed
Cumulative sums test	(Forward)	0.874531	Succeed
	(Reverse)	0.866330	Succeed

Table 2. Comparison of the proposed algorithm by NIST tests

Test Name		Proposed algorithm	Chacha20	RC4	Salsa20
Frequency test		0.110368	0.002082 FAILURE	0.008651 FAILURE	0.365276
Approximate entropy test		1.000000	1.000000	1.000000	1.000000
Test for the longest run of ones in a block		1.000000	1.000000	1.000000	1.000000
Frequency test within a block		0.670519	0.087230	0.063454	0.363610
Runs test		0.708619	0.661803	0.751606	0.000495 FAILURE
Discrete Fourier transform (spectral) test		0.954263	0.347239	0.518136	0.126654
Nonperiodic templates test		0.999252	0.999406	0.999406	0.999406
Overlapping template of all one's test		1.000000	1.000000	1.000000	1.000000
Serial test	P-v1	1.000000	1.000000	1.000000	1.000000
	P-v2	1.000000	1.000000	1.000000	1.000000
Cumulative sums test	(Forward)	0.210789	0.003062 FAILURE	0.013212	0.409666
	(Reverse)	0.196352	0.003574 FAILURE	0.017302	0.552345

4.2. Statistical cryptanalysis

4.2.1. Histogram analysis

The histogram depicts the pixel distribution in an image. A histogram analysis may be used to attack the cipher image, which may be useful to the eavesdropper [18]. As a result, the cipher image's histogram should be as uniform as practicable to avoid statistical histogram attacks. According to the histogram shown in Figure 3, the cipher histogram is more uniform than the plain histogram. As a result, the scheme is stable and resistant to histogram attacks.

4.2.2. Shannon entropy analysis

In computer science, information entropy is a foundational principle. It's essential in security and information coding for data compression. In addition, the Shannon entropy is a valuable metric for quantifying the degree of disorder or chaos, evaluating the complexity of compounded processes, and determining the divergence among probability distributions. Entropy is a measurement of the unpredictability of a cryptographic key which is often utilized in cryptanalysis. Utilizing a brute force attack requires (on average) 2^{n-1} (n=number of bit key) to break the key. Entropy fails to capture the requisite number of guesses if the possible keys are not selected at random [19]. Table 3 shows that the entropy values are located at an optimal interval. Consequently, the suggested approach's output image is protected from different statistical attacks.

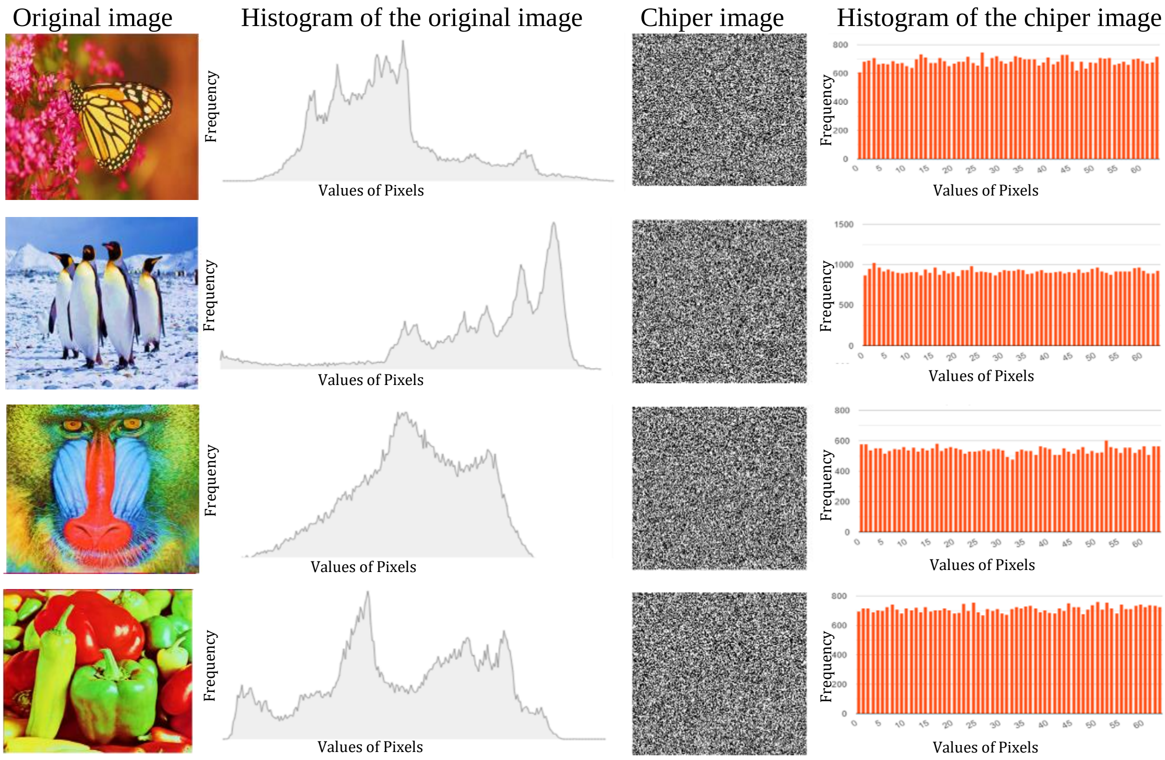


Figure 3. Original images with the histogram for both original and encryption results of images

Table 3. Correlation coefficients and the entropy of different images

Image name	Entropy		Correlation value
	Plain	Cipher	
Butterfly	7.56231	7.652675939	-0.00234697
Penguins	7.31233	7.583061768	-0.014839155
baboon	7.581733458	7.70037	-0.011288981
peppers	7.582755573	7.73271	-0.006632476

4.2.3. Correlation coefficient analysis

A statistical relationship among neighborhood pixels is defined by an image correlation coefficient [20]. An ideal encryption scheme should break the high correlation among neighboring pixels of a plain image and cipher image. We compute the correlation coefficient of pixels from the plain image and the corresponding cipher image using (6) [20], [21]:

$$\text{Cxy} = \frac{\frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))(y_i - E(y))}{\sqrt{D(x)}\sqrt{D(y)}},\ d(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i - E(x))^2,\ E(x) = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{6}$$

where, x_i, y_i are the values of the gray image, N is the total pixels, and E(x), E(y) are the mean values of xi and yi. We can conclude that correlation results have a low correlation (close to zero) between plain image pixels and cipher image pixels as shown in Table 3. Consequently, the proposed approach is effective at protecting against statistical analysis attacks.

4.2.4. Keyspace and key sensitivity

The total number of variables used in the cryptosystem is referred to as the keyspace. A robust encryption technique should have a large enough keyspace in order to overcome brute-force attacks [22]. Keyspace should be larger than 2^{100} to avoid brute-force attacks [23]. The secret keys are made up of three parts in the suggested approach: a master key, a Chebyshev map, and a logistic map.

The keyspace of the master key is ($10^{15} \approx 2^{50}$) bits, In addition, the keyspace of the Chebyshev map is comprised of control parameter ∈[0, 1] and seeds (b) larger than one. Let the b is a seven-digit decimal number. So, the key length of b will be 2^{56} as well as round number R∈[400, 800]. Consequently, the total keyspace length of the Chebyshev map is $2^{56} \times 400 \approx 2^{65}$. Whereas, the logistic map keys are round number

R∈[400, 800), initial value ∈[0, 1], and control parameter ∈[1, 4]. Consequently, the total keyspace length of the logistic map is about $(10^{15})^2 \times 400 \approx 2^{109}$ bits. Accordingly, the entire keyspace size is $2^{50}+2^{65}+2^{109}=2^{224}$. This is sufficient for any encryption system.

Furthermore, the test of the cipher-text difference rate (CDR) was utilized in order to calculate the difference between the plaintext and the ciphertext. Consequently, we evaluated the sensitivity of the key according to difference results. This test was carried out by slightly altering the key. As described in the formula (7) [24]:

$$\mathrm{Diff(I1,I2)} \begin{cases} 0 \; if \; I1 = I2 \\ 1 \; if \; I1 \; \neq I2 \end{cases}, Diff_{sum}(I1, I2) = \sum Diff(I1, I2)$$

$$CDR = \sum \frac{Diff_{sum}(C1,C2)+diff_{sum}(C1,C3)}{2 \times L} \times 100\% \qquad (7)$$

where, I1, I2 denoted to encrypted text before and after key modification; C1, C2, and C3 are the ciphertext utilizing different encryption keys (K, K+ΔK, and K−ΔK respectively); and L denotes the ciphertext length. As shown in Table 4 all the results are greater than 90%. Also, these results are raising slightly when the size of the input bytes is increased. As a result, the proposed approach results have acceptable key sensitivity in the keys parameters which resist various cryptanalysis.

Table 4. CDR results of different sizes of bytes which applied various encryption keys on it

Number of bytes	CDR result
1024	0.913
2048	0.912
4096	0.926
8192	0.931
16384	0.942

4.3. The proposed approach for diffusion the plain image

The proposed approach is not providing diffusion when it is used to encrypt the images. Because it encrypts identical plaintext blocks into identical ciphertext blocks. Whilst, cipher block chaining (CBC) doesn't generate identical cipher images of repeated plain images [25]. Therefore, the suggested method is used CBC to provide a good diffusion of the image without encrypted it by changing the positions of the image blocks as shown in Figure 4. Algorithm 6 explains the proposed solution for the diffusion of the plain image.

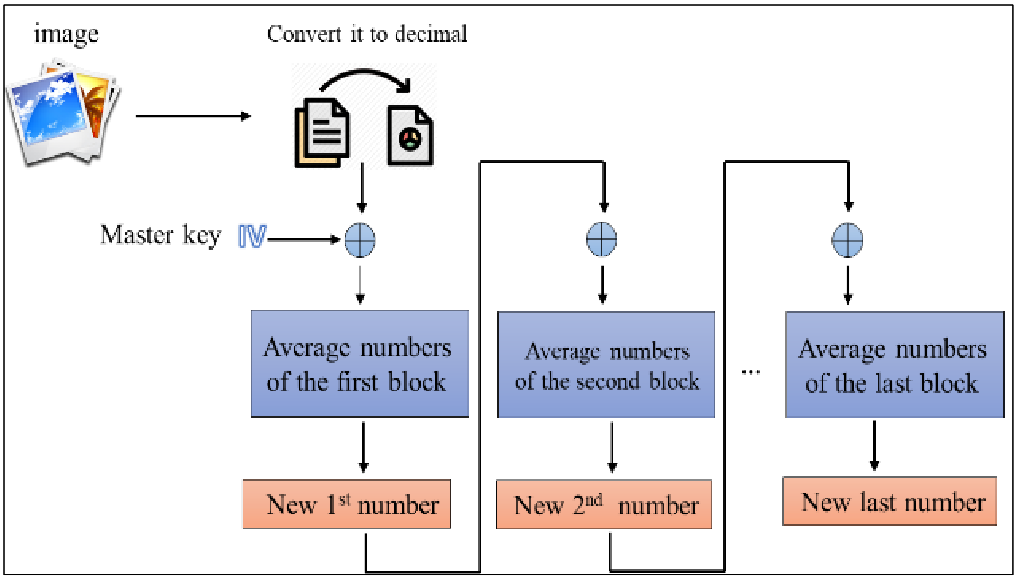


image
Convert it to decimal
Master key
IV
Average numbers of the first block
Average numbers of the second block
...
Average numbers of the last block
New 1st number
New 2nd number
New last number

Figure 4. Proposed CBC for diffusion plain image

Algorithm 6: Produce a new sequence to image blocks

```
1: Input: plain image with size 256*256.
2: Output: new sequence numbers of image blocks (used for confusion plain image).
3: Begin
4: Divide the image into 32*32 blocks. Each block is (8*8).
5: Find the average numbers of each block and round it to integer, then mod the result by 1024.
6: The results (32*32 numbers) of step: 5 are used as a new index number for the plain image.
7: The results of step:6 are XOR with the IV (here, we used the same input master key of subsection 3.2. as an IV). $C_1=E(P_1\oplus IV)$.
8: XOR of the result of step:7 with the average of the second block, and so on.
   $C_i=E(P_i\oplus C_{i-1})$, where: i=2,..,N.
9: Store the results of step:8 in the array (arr[]).
10: Replace the duplicated numbers in arr[] with non-founded numbers in decreasing way.
11: Increasingly reorder the result of step:10.
12: End.
```

To reconstruction the image after decrypted it to the original sequence, we must store the index sequence (the result of step: 10 in the Algorithm 6) at the end of the header of the image (at the beginning of the image data). The cipher image should be reordered by using the index sequence after decrypted, to get the original image.

4.3.1. Plain image sensitivity (analysis of differential attack)

Another common form of attack is the differential attack. In this attack, the attackers choose a simple image and modify it by making minor changes to it, such as a one-bit change. After that, they use the cryptosystem to encrypt the two images. Consequently, they attempt to break the scheme by tracing the differences. So, a cryptosystem must generate significantly different cipher images from similar two plain images with slightly different [16]. In general, the invulnerability of differential attacks is measured using two metrics: unified average changing intensity (UACI) and the number of pixel change rates (NPCR). The following are the formulas for calculating these two metrics [1]:

$$UACI = \frac{1}{W\times H}\sum_{i=0}^{W-1}\sum_{j=0}^{H-1}\frac{|C_1(i,j)-C_2(i,j)|}{255}\times 100\% \tag{8}$$

where, C1 and C2 are the cipher images resulted from the proposed approach.

$$NPCR = \frac{\sum_{i,j} D(i,j)}{W\times H}\times 100\%, Where: D(i,j) = \begin{cases}1, & if\ (C_1(i,j) \neq C_2(i,j) \\ 0, & if (C_1(i,j) = C_2(i,j)\end{cases} \tag{9}$$

The results of the proposed approach are shown in Table 5 all scores of UACI are less than the theoretical normal values (33.2255%). Also, the scores of NPCR are close to the theoretical normal values (99.5693%) [1]. Accordantly, we concluded the proposed approach is passed UACI and NPCR tests. Consequently, the suggested method provides sufficient diffusion operations, and it is resistant to differential attack. Whilst, the peak signal to noise ratio (PSNR) measurement is commonly utilized to assess the quality of restored images. The formulas (10) and (11) are utilized to compute this measure [26]:

$$MSE = \frac{1}{W*H}\sum_{i=0}^{W-1}\sum_{j=0}^{H-1}(I_p(i,j) - I_D(i,j))^2 \tag{10}$$

$$PSNR = 10 * log\frac{255^2}{MSE}(dB) \tag{11}$$

where, I_P is the original plain image, and I_D is the decrypted image. The PSNR is expressed in decibels. The higher of decibels value (greater than 35 dB), means the better in image quality. Table 5 illustrated the scores of PSNR. All results are excellent because the proposed approach does not utilize compression or filter operations. Consequently, the suggested method exhibits better performance at the image quality test.

Table 5. UACI, NPCR, and PSNR scores of the tested images

The tested image	Size of Image (W.H)	Score of UACI (%)	Score of NPCR (%)	Score of PSNR (dB)
Butterfly	256 * 256	33.1354	99.57643	60.0533
Penguins	256 * 256	33.1456	99.50634	59.5016
baboon	256 * 256	33.06358	99.53645	59.4439
peppers	256 * 256	33.0354	99.58542	60.0447

5. CONCLUSION

This work has presented a new lightweight encryption approach that is based on permutation between Chebyshev and logistic maps and which is destined for secure data transfers in limited environments like IoT. The proposed solution utilizes to solve the key distribution problem by generating a random key (PRNG). The suggested method was compared with three algorithms by utilizing the same plaintext and master key as input parameters. Despite the failure of some test results for some comparative algorithms, all allowed NIST test results of the proposed approach are passed successfully. Also, the histogram test showed the cipher histogram is more uniform than the plain histogram. Furthermore, the suggested approach's output is resisting various cryptanalysis and protected from different statistical attacks such as Shannon entropy, correlation coefficient, and keyspace and key sensitivity.

REFERENCES

[1] E. Yavuz, “A novel chaotic image encryption algorithm based on content-sensitive dynamic function switching scheme,” *Optics and Laser Technology*, vol. 114, pp. 224–239, Jun. 2019, doi: 10.1016/j.optlastec.2019.01.043.

[2] S. A. Jassim and W. K. Awad, “Searching over encrypted shared data via cloud data storage,” *Journal of Theoretical and Applied Information Technology*, vol. 96, no. 12, pp. 3707–3716, Jun. 2018.

[3] S. Q. A. Al-Rahman, S. A. Jassim, and A. M. Sagheer, “Design a mobile application for vehicles managing of a transportation issue,” *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 4, pp. 2263–2272, Aug. 2021, doi: 10.11591/eei.v10i4.2918.

[4] A. S. Alshammari, “Comparison of a chaotic cryptosystem with other cryptography systems,” *Engineering, Technology and Applied Science Research*, vol. 10, no. 5, pp. 6187–6190, Oct. 2020, doi: 10.48084/etasr.3745.

[5] H. A. Khan, R. Abdulla, S. K. Selvaperumal, and A. Bathich, “IoT based on secure personal healthcare using RFID technology and steganography,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11, no. 4, pp. 3300–3309, 2021, doi: 10.11591/ijece.v11i4.pp3300-3309.

[6] M. Saber and M. M. Eid, “Low power pseudo-random number generator based on lemniscate chaotic map,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11, no. 1, pp. 863–871, Feb. 2021, doi: 10.11591/ijece.v11i1.pp863-871.

[7] H. Ali-Pacha, N. Hadj-Said, A. Ali-Pacha, M. A. Mohamed, and M. Mamat, “Cryptographic adaptation of the middle square generator,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 6, pp. 5615–5627, Dec. 2019, doi: 10.11591/ijece.v9i6.pp5615-5627.

[8] L. Ding, C. Liu, Y. Zhang, and Q. Ding, “A new lightweight stream cipher based on chaos,” *Symmetry*, vol. 11, no. 7, Jul. 2019, Art. no. 853, doi: 10.3390/sym11070853.

[9] Z. Hua, Y. Zhou, and H. Huang, “Cosine-transform-based chaotic system for image encryption,” *Information Sciences*, vol. 480, pp. 403–419, Apr. 2019, doi: 10.1016/j.ins.2018.12.048.

[10] A. Kadhim, “New image encryption based on pixel mixing and generating chaos system,” *Al-Qadisiyah Journal Of Pure Science*, vol. 25, no. 4, pp. 1–14, 2020.

[11] Y. Luo, Y. Liu, J. Liu, X. Ouyang, Y. Cao, and X. Ding, “ECM-IBS: A Chebyshev map-based broadcast authentication for wireless sensor networks,” *International Journal of Bifurcation and Chaos*, vol. 29, no. 09, Aug. 2019, Art. no. 1950118, doi: 10.1142/S0218127419501189.

[12] A. T. Sadiq, A. K. Farhan, and S. A. Hassan, “A proposal to improve RC4 algorithm based on hybrid chaotic maps,” *Journal of Advanced Computer Science and Technology Research*, vol. 6, no. 4, pp. 74–81, 2016.

[13] M. S. Fadhil, A. K. Farhan, and M. N. Fadhil, “Designing substitution box based on the 1D logistic map chaotic system,” *IOP Conference Series: Materials Science and Engineering*, vol. 1076, no. 1, Art. no. 012041, Feb. 2021, doi: 10.1088/1757-899X/1076/1/012041.

[14] Y. Chen, Q. Cao, Z. Zhu, Z. Wang, and Z. Zhao, “Switched fuzzy sampled-data control of chaotic systems with input constraints,” *IEEE Access*, vol. 9, pp. 44402–44410, 2021, doi: 10.1109/ACCESS.2021.3066402.

[15] X. Li and Y. Ling, “Research and application of pseudorandom sequence based on Xor,” *Procedia Computer Science*, vol. 183, pp. 814–819, 2021, doi: 10.1016/j.procs.2021.03.003.

[16] Y. Liu, Z. Qin, X. Liao, and J. Wu, “A chaotic image encryption scheme based on Hénon-Chebyshev modulation map and genetic operations,” *International Journal of Bifurcation and Chaos*, vol. 30, no. 06, May 2020, Art. no. 2050090, doi: 10.1142/S021812742050090X.

[17] E. Avaroğlu, “The implementation of ring oscillator based PUF designs in field programmable gate arrays using of different challenge,” *Physica A: Statistical Mechanics and its Applications*, vol. 546, May 2020, Art. no. 124291, doi: 10.1016/j.physa.2020.124291.

[18] A. A. Shah, S. A. Parah, M. Rashid, and M. Elhoseny, “Efficient image encryption scheme based on generalized logistic map for real time image processing,” *Journal of Real-Time Image Processing*, vol. 17, no. 6, pp. 2139–2151, Dec. 2020, doi: 10.1007/s11554-020-01008-4.

[19] P. Nannipieri *et al.*, “True random number generator based on fibonacci-galois ring oscillators for FPGA,” *Applied Sciences*, vol. 11, no. 8, Apr. 2021, Art. no. 3330, doi: 10.3390/app11083330.

[20] F. Yang, J. Mou, Y. Cao, and R. Chu, “An image encryption algorithm based on BP neural network and hyperchaotic system,” *China Communications*, vol. 17, no. 5, pp. 21–28, May 2020, doi: 10.23919/JCC.2020.05.003.

[21] K. Shravanraj, R. G. Rejith, and M. Sundararajan, “Evaluation of heavy metals in coastal aquifers and seawater,” in *Remote Sensing of Ocean and Coastal Environments*, Elsevier, 2021, pp. 155–176.

[22] A. A. A. El-Latif *et al.*, “Providing end-to-end security using quantum walks in iot networks,” *IEEE Access*, vol. 8, pp. 92687–92696, 2020, doi: 10.1109/ACCESS.2020.2992820.

[23] S. A. Banday, M. K. Pandit, and A. R. Khan, “Securing medical images via a texture and chaotic key framework,” in *Multimedia Security*, Springer, 2021, pp. 3–24.

[24] S. Jing, Y. Guo, and W. Chen, “Meaningful ciphertext encryption algorithm based on bit scrambling, discrete wavelet transform, and improved chaos,” *IET Image Processing*, vol. 15, no. 5, pp. 1053–1071, Apr. 2021, doi: 10.1049/ipr2.12085.

[25] B. Idrees, S. Zafar, T. Rashid, and W. Gao, “Image encryption algorithm using S-box and dynamic Hénon bit level permutation,”

Multimedia Tools and Applications, vol. 79, no. 9–10, pp. 6135–6162, Mar. 2020, doi: 10.1007/s11042-019-08282-w.

[26] R. M. Neamah, J. A. Abed, and E. A. Abbood, "Hide text depending on the three channels of pixels in color images using the modified LSB algorithm," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 1, pp. 809–815, Feb. 2020, doi: 10.11591/ijece.v10i1.pp809-815.

BIOGRAPHIES OF AUTHORS

Sameeh Abdulghafour Jassim received the bachelor's degree and the master's degree in Computer Science from University of Anbar, Iraq in 2006–2007 and 2012–2013 respectively. Currently, he is a Ph.D. student at the University of Technology department of computer sciencea as well as an assistant instructor in the Ministry of Education, Iraq. His research interests include Cloud Computing, Cryptography, Security, IoT, Chaos theory, and Lightweight Cryptography. He can be contacted at email: prog85sameeh@gmail.com.

Alaa Kadhim Farhan is Professor in the Department of Computer Sciences, University of Technology-Baghdad-Iraq. He completed her Bachelor of computer Science and Master of Science degrees in information security, from Department of computer Sciences-University of Technology, Baghdad, Iraq, in 2003, and 2005, respectively. He received her Ph.D. degree in information security from University of Technology, Baghdad, Iraq 2009. In 2005 he joined the Department of Computer Sciences, University of Technology, as an academic staff member. Prof. Dr. Alaa is the author of numerous technical papers since 2008, her research interests include: Cryptography, programming languages, Chaos theory, cloud computing. He can be contacted at email: Alaa.K.Farhan@uotechnology.edu.iq.