An Efficient Image Encryption Technique using Chaotic Logistic Map and RC4 Stream Cipher

Dr. Methaq Talib Gaata\textsuperscript{1}, Fadya Fouad Hantoosh\textsuperscript{2}  
\textsuperscript{1}Computer Science, Al-Mustansiriya University  
\textsuperscript{2}Computer Science, Al-Mustansiriya University

Abstract—In recent years, because of the frequent flow of digital images across the world over the transmission media, image encryption become one of the most substantial topics. In this paper, we propose a new image encryption method based on the well-known Chaotic Logistic Map (CLM) and the Rivest Cipher 4 (RC4) encryption methods. Here, we use the secret key, and the CLM to produce a one-dimensional array of different numbers. Then the RC4 algorithms used to make some sort of random shuffling (relying on the contents of the array created by the CLM) to the array that is created by the RC4 first algorithm. After that, the second algorithm of RC4 used inside a loop to change the value of each color (using the resultant array of the first RC4 algorithm) of a pixel until all the pixels of the image be changed. And by doing that we have produced a cipher image that is completely different and does not reveal any information of the plain image, also we proved that it is really a key sensitive encryption that can’t be decrypted easily by using brute force or any other kind of attacks.

Keywords—image encryption, chaotic logistic map, RC4 stream cipher, image cryptography, chaos based system

I. INTRODUCTION

Encryption has a long history, dating back to when the ancient Greeks and Romans sent secret messages by substituting letters only decipherable with a secret key, although they focused on text on the beginning, but then they worked on the other types of multimedia like images, videos, etc. For now we will focus on images as another example of multimedia protection, on the first years of work they used the same methods that has been used on text (RSA, AES... etc.). However, these encryption schemes appear to be inappropriate for images, due to some special features of images such as bulk capacity, high redundancy, and high correlation among pixels, not to mention that they are usually huge in size [8]. Moreover, these encryption schemes require extra operations on compressed image data, thereby demanding long computational time and high computing power. In real-time communications, because of their low encryption and decryption speeds, they may introduce significant latency. In recent years, the cryptographic schemes have suggested some new and efficient ways to develop secure image encryption. These schemes have typical structure which performed the permutation and the diffusion stages alternatively. However, most of the algorithms faced some problems such as the lack of robustness and security [1].

Chaos-based cryptographic scheme is an efficient encryption first presented in 1989 [2]. It has many brilliant characteristics different from other algorithms such as the sensitive dependence on initial conditions, non-periodicity, non-convergence and control parameters. The one-dimensional chaos system has the advantages of simplicity and high security, many studies were proposed to adopt and improve it.

In this paper, we propose a chaotic encryption scheme based on RC4 stream cipher. RC4 is a simple algorithm, although it has some weakness points as shown in [3] and [6], but it will be combined with chaotic systems to make it almost impossible to break. We used the key generator used by [4] to convert the key to initial value, then use this initial value on Chaotic Logistic Map

DOI:10.21884/IJMTER.2016.3054.PVA2I 213
(CLM) function to generate pseudo random number sequence. Which is then added to the byte streams of plain-image and modulo by 256 when doing an encryption process (OR add 256 to the byte streams of cipher-image and subtract from the stream of pseudo random numbers and take modulo of 256 when doing a decryption process).

The rest of the paper is organized as follows: on section II, we will explain our proposed algorithm with some examples to see how it works, after that we will make some tests to prove that this proposed algorithm has robust security this will be in section III, then the last section will be the conclusion section IV.

II. PROPOSED IMAGE ENCRYPTION SCHEME

In this section, we describe all the steps for encryption and decryption of the image using chaotic logistic map and RC4 Stream cipher (Figure. 1). Where the process is divided into two stages (CLM and RC4 stages):

Stage One: CLM

First, we take the external key “Image Owner key”, a key of 16 characters long, convert it to its hexadecimal form, and then send it to CLM block.

Figure 1. Proposed Image Encryption Model

A chaotic logistic map is employed to achieve the goal of image encryption
Throughout the algorithm, we will keep the value of the system parameter of the logistic map as a constant (i.e. \( \lambda = 4 \)), which corresponds to a highly chaotic case. While the initial condition \( X_0 \) for the logistic map is calculated using the equation below:

\[
X_{0,2} = \lambda X_{0,1} (1 - X_{0,1}) \tag{1}
\]

To find \((X_{01} \text{ and } X_{02})\) we need to split the external key into two groups G0 and G1. By using these two groups, the real numbers \(X_{01}\) and \(X_{02}\) can be calculated using equations (3) and (4):

\[
X_{01} = (X_0 * 2^p + \ldots + X_{25} * 2^{25}) / 2^{25} \tag{3}
\]

\[
X_{02} = (X_0 * 2^p + \ldots + X_{25} * 2^{25}) / 2^{25} \tag{4}
\]

The initial value \(X_0\) will be utilized by the CLM equation (2.18) to generate an array of pseudo random numbers. Generally, the chaotic process uses initial value \(X_0\) to get \(X_1\), then \(X_1\) value will be used to get \(X_2\), and so on.

After obtaining each \(X_n\) value, we multiply it by image width and divide it by width to get the result by width. Then the result would be converted into integer, which is done by taking eight points starting after decimal point of the real numbers. For example, assume the value of \(X_n\) after multiplication and division is 0.1234567890 the result after conversion is 1234567890.

These integer numbers would be stored in an array (called “U” of size 256); the process is repeated until the array is filled. By finishing this step, we leave the CLM block, take the array as an output, and go to RC4 block.

**Stage Two: RC4 stream cipher**

1) As a first step in RC4 algorithm, an array called “\(S\)” is created. Where the content of it are set equal to the values from 0 through 255 in ascending order; such as, \(S[0]=0, S[1]=1, \ldots, S[255]=255\).

2) Next step, array \(U\) is used to produce the initial permutation of array \(S\). For each \(S[index]\), swap \(S[index]\) with another byte in \(S\) according to the content of \(U[index]\), and other parameters (for better clarification, see algorithm 1). This will cause the content of \(S\) still contains all the numbers from zero through 255.

3) Streams generation is done by using PRGA, which swaps \(S[index]\) with another byte of \(S\) according to a scheme dictated by the current configuration of \(S\) (details in algorithm 2).

---

**Algorithm 1: Initial Permutation**

**INPUT:** Array \(S\) of integers in ascending order, Array \(U\) produced by CLM

**OUTPUT:** Permuted array \(S\)

1. \(j \leftarrow 0\);
2. **For** \(i \leftarrow 0\) **to** \(255\)
3. \(j \leftarrow (j + S[i] + U[i]) \mod 256\);
4. **SWAP(S[i], S[j]);**
5. **End for**

---

**Algorithm 2: Pseudo-Random Generation**

**Algorithm (PRGA)**

**INPUT:** Permuted Array \(S\)

**OUTPUT:** Integer value between 0 and 255

1. \(i \leftarrow 0; \ j \leftarrow 0;\)
2. **While ()**
3. \(i \leftarrow (i + 1) \mod 256;\)
4. \(j \leftarrow (j + S[i]) \mod 256;\)
5. **SWAP (S[i], S[j]);**
6. \(t \leftarrow [(S[i] + S[j]) \mod 256];\)
7. **return** \(t;\)

---

After finishing the last step, the encryption process

---

1 Choosing eight points is not a rule; it can be any number of points. As long as it gives numbers that it’s modulo by 256 will range between 0 and 255.

---

@IJMTER-2016, All rights Reserved 215
(algorithm 3) for each RGB channel is done by adding the output of the last step to one of the RGB channels value, and then takes modulo by 256. This process is repeated for the other channels, and to the rest of the image pixels values.

While the decryption (algorithm 4) process is done by adding 256 to each byte of the RGB and subtracts it from the output of step three (PRGA), and then the result would be modulated by 256, this is also repeated to the other channels and to the rest of the image pixels values.

III. EXPERIMENTAL RESULTS

In this section, results of the proposed encryption system are presented. Our encryption system works on both color and grey images with different sizes and image formats, however, the speed might vary with image size. By applying the algorithms above, we get the results in Table 1 and Figure. 2, which shows the key stream generated by chaotic logistic map, and some values of the initial condition.

\[ \begin{array}{|c|c|c|c|} 
\hline 
X & U & S & S' \\
\hline 
0.2061767578125 & 20617676 & 0 & 154 \\
0.654671609401703 & 65467161 & 1 & 67 \\
0.904306772980348 & 90430677 & 2 & 232 \\
0.346144133288869 & 34614413 & 3 & 229 \\
0.905313489114266 & 90531349 & 4 & 92 \\
0.342883902168078 & 34288390 & 5 & 11 \\
0.90125812720828 & 90125813 & 6 & 141 \\
0.355967661397216 & 35596766 & 7 & 176 \\
0.917018741746452 & 91701874 & 8 & 174 \\
0.304381476128822 & 30438148 & 9 & 207 \\
0.846933572473846 & 84693357 & 10 & 6 \\
0.518548385162139 & 51854839 & 11 & 19 \\
0.998623829631508 & 99862383 & 12 & 177 \\
0.00549710609443652 & 549711 & 13 & 34 \\
0.0218675516760921 & 218675 & 14 & 79 \\
\hline 
\end{array} \]

Table 1. Data analysis for the encryption system
Where X is the initial value generated by using the secret key and chaotic logistic map, U is an array produced from the initial value, S is the initial array generated by (KSA), and S’ is the initial permutation also generated by (KSA) using U array.

**IV. SECURITY ANALYSIS**

A good encryption system should be robust against all kinds of attacks. In this section, to demonstrate the efficiency of the proposed method different tests were done to prove its robustness.

**A. Key sensitivity analysis**

A typical image encryption process should be critical with respect to the secret key i.e. the alteration of a single bit in the secret key should produce an entirely different encrypted image [5],[7]. We made a key sensitivity test using a key that is one digit different from the original key to decrypt the encrypted image. We have encrypted plain-image in Figure 3 using key “qawsedrfigyhujik” and then decrypted the cipher-image using: wrong key “pawsedrfigyhuik”, and correct key ” qawsedrfigyhujik”. The resulting image is totally different from the original image (as shown in Figure 3. Clearly, the decryption with a slightly different key fails completely, and hence the proposed image encryption procedure is a highly key sensitive.

**B. Key space analysis**

The robustness of a cryptosystem depends entirely on the secret keys. For a secure image cipher, the key space should be large enough to make the brute force attack hardly possible. The key space size tells us the number of different keys that can be used for the encryption of an image. Our proposed image cipher has $2^{128}$ different combinations of the secret key; hence, the brute-force attack is difficult in our case.

**C. Histogram analysis**

An image-histogram explicates how pixels in an image are distributed by graphing the number of pixels at each color intensity level. In order to have an idealistic ciphered image in histogram’s sight of view, the histogram of the image must have parallel distribution of pixels with the color intensity value. Based on the histogram analysis for the images in Figure. 4a, and Figure. 4b, we can see from that the histogram of each RGB channel is uniform and significantly different from the respective
histograms of the original image, and so it does not give any hint to employ whatever attack on the proposed image encryption procedure.

V. CONCLUSION

In this paper, we propose a new image encryption method based on the well-known Chaotic Logistic Map (CLM) and the Rivest Cipher 4 (RC4) encryption methods. We tried to find a way that is fast, robust and completely secure just to make sure that our pictures can be sent safely without worrying about revealed information, also we proved that it is highly key sensitive encryption that can’t be decrypted easily by using brute force or any other types of attack.

REFERENCES


