A Fast Image Encryption Scheme with Compound Keys

Yong Zhang and Qiong Zhang
School of Software and Communication Engineering, Jiangxi University of Finance and Economics, Nanchang, P. R. China.
Email: zhangyong@jxufe.edu.cn

Abstract. An image encryption scheme with public keys and private keys is proposed. In general, chosen/known plaintext attacks are far more effective than the brute-force attack on an image cryptosystem. Besides employing a 300-bit long secret key, the scheme uses different public keys in each encryption process, which makes the known/chosen plaintext attacks infeasible. And the scheme possesses a structure of diffusion-confusion-diffusion, in which the confusion operation is plaintext-related. The simulation results show that the proposed scheme is faster than AES in encryption/decryption speed, and has good key and plaintext sensitivities.

1. Introduction
In recent years, image encryption has become a research hotspot in the field of information security. Image encryption system falls into the area of symmetric cryptography. Many scholars have proposed multiple excellent image encryption algorithms based on chaotic systems [1-6]. Meanwhile, the cryptanalysts also put forward a large number of deciphering algorithms for image encryption systems [7-10], most of which employ the chosen plaintext or chosen cipher-text attacks.

In order to resist the chosen plaintext or chosen cipher-text attacks, the image encryption system must encrypt the different plain images with different key streams, even if with the same secret key. Therefore, some scholars have proposed the plaintext-related image encryption algorithm [11-12]. On the basis of the plaintext-related image cryptosystem, this paper proposes the image encryption method combining the private key and the public key to better frustrate the chosen/known plaintext or cipher-text attacks.

2. Image Encryption Scheme

2.1. Used Chaos Systems
The piecewise linear chaotic map (PWLCM) in Eq. (1) and Chen’s chaotic system in Eq. (2) are used in this paper.

\[ F(x, p) = \begin{cases} \frac{x}{p}, & 0 < x \leq p \\ \frac{x-p}{0.5-p}, & p < x \leq 0.5 \\ F(1 - x, p), & 0.5 < x \leq 1 \end{cases} \]  

(1)

\[ \begin{align*} x'(t) &= a(y - x) \\ y'(t) &= (c - a)x - xz + cy \\ z'(t) &= xy - bz \end{align*} \]  

(2)

In Eq. (1), let \( p = 0.3 \). And in Eq. (2), let \( a = 35, b = 3, c = 27 \), and the step size be 0.001.
2.2. System Structure

The proposed image cryptosystem possesses the structure as shown in Fig. 1, which includes two covering operations, two plaintext-unrelated diffusion operations, and one plaintext-related confusion operation. The secret key \( K \) is a 300-bit long binary sequence containing nine 32-bit integers and one 12-bit integer, sequentially denoted as \( K_i \), \( i=1, 2, \ldots, 10 \). The public key \( IV \) is a positive decimal less than 1, which is used as the initial value of PWLCM.

![Proposed image cryptosystem diagram](image)

**Figure 1.** Proposed image cryptosystem.

2.3. Key Stream Generator

The proposed image cryptosystem will use four pseudo-random matrices, denoted by \( X, Y, Z \) and \( I \), which are generated by the following steps.

**Step 1.** Generate the initial values of Chen’s system with \( K_1, K_2 \) and \( K_3 \), i.e.

\[
\begin{align*}
  x_{00} & = 44.29 \frac{K_1}{2^{32}} - 23.19 \\
  y_{00} & = 49.67 \frac{K_2}{2^{32}} + 26.19 \\
  z_{00} & = 35.26 \frac{K_3}{2^{32}} + 5.38
\end{align*}
\]

(3) (4) (5)

**Step 2.** Use \( x_{00}, y_{00} \) and \( z_{00} \) as the initial values of Chen’s system, then iterate Chen’s system for 100 times to obtain the state values, denoted by \( x_{01}, y_{01} \) and \( z_{01} \), respectively. Update the values of \( x_{01}, y_{01} \) and \( z_{01} \) with \( K_4, K_5 \) and \( K_6 \), i.e.

\[
\begin{align*}
  x_{01} & = 0.618 \times x_{00} + 0.382 \times (44.29 \frac{K_4}{2^{32}} - 23.19) \\
  y_{01} & = 0.618 \times y_{00} + 0.382 \times (49.67 \frac{K_2}{2^{32}} - 26.19) \\
  z_{01} & = 0.618 \times z_{00} + 0.382 \times (35.26 \frac{K_3}{2^{32}} + 5.38)
\end{align*}
\]

(6) (7) (8)

**Step 3.** Use \( x_{01}, y_{01} \) and \( z_{01} \) as the initial values of Chen’s system, then iterate Chen’s system for 100 times to obtain the state values, denoted by \( x_{02}, y_{02} \) and \( z_{02} \), respectively. Update the values of \( x_{02}, y_{02} \) and \( z_{02} \) with \( K_7, K_8 \) and \( K_9 \), i.e.

\[
\begin{align*}
  x_{02} & = 0.618 \times x_{01} + 0.382 \times (44.29 \frac{K_4}{2^{32}} - 23.19) \\
  y_{02} & = 0.618 \times y_{01} + 0.382 \times (49.67 \frac{K_2}{2^{32}} - 26.19) \\
  z_{02} & = 0.618 \times z_{01} + 0.382 \times (35.26 \frac{K_3}{2^{32}} + 5.38)
\end{align*}
\]

(9) (10) (11)

**Step 4.** Use \( x_{02}, y_{02} \) and \( z_{02} \) as the initial values of Chen’s system, then iterate Chen’s system for 100 times to bypass the transient states. Then, continue to iterate Chen’s system for \( MN \) times to get three...
sequences, denoted by \{x_i\}, \{y_i\} and \{z_i\}, i=1,2,...,MN, respectively. Separately generate the matrices \(S_x, S_y, S_z\) of size \(M\times N\) with \(S_x(i,j)=x_{i,j}\), \(S_y(i,j)=y_{i,j}\), \(S_z(i,j)=z_{i,j}\), \(i=1,2,...,M,\ j=1,2,...,N\). Finally, produce the pseudo-random matrices \(X, Y, Z\) and \(S_x, S_y, S_z\) by the following equations.

Where, \(i=1, 2,...,M, j=1, 2,..., N\), and floor(x) return the largest integer that is not greater than x.

\[
X(i,j) = \begin{cases} 
1, & \left( 100 + S_x(i,j) \right) \text{mod} \ 1 < 0.5 \\
0, & \left( 100 + S_x(i,j) \right) \text{mod} \ 1 \geq 0.5
\end{cases}
\]

\[
Y(i,j) = \text{floor}(2^{16} \times \left[ \left( 100 + S_y(i,j) \right) \text{mod} \ 1 \right]) \text{mod} \ 256
\]

\[
Z(i,j) = \text{floor}(2^{16} \times \left[ S_z(i,j) \text{mod} \ 1 \right]) \text{mod} \ 256
\]

**Step 5.** Use \(IV\) as the initial value of PWLCM, and then iterate PWLCM for \(K_{10}\) times. Continue to iterate PWLCM for \(MN\) times to get a sequence. And convert the sequence into a matrix \(J\) of size \(M\times N\) in line order. Finally, transform \(J\) into an integer matrix \(I\) with the following equation.

\[
I(i,j) = \text{floor}(2^{16} \times J(i,j)) \text{mod} \ 256, \ i=1,2,...,M, \ j=1,2,...,N
\]

With the above steps, the pseudo-random matrices \(X, Y, Z\) and \(I\) are generated with the secret key \(K\) and public key \(IV\).

### 2.4. **Covering Operation**

As shown in Fig. 1a, the covering-I operation converts the plain image \(P\) into the matrix \(A\) with \(X\) and \(I\), i.e.

\[
A(i,j) = \begin{cases} 
P(i,j), & \text{if } X(i,j) = 0 \\
P(i,j) \text{ XOR } I(i,j), & \text{if } X(i,j) = 1, \ i=1,2,...,M, \ j=1,2,...,N
\end{cases}
\]

The covering-II operation converts the matrix \(E\) into the cipher image \(C\) with \(X\) and \(I\), i.e.

\[
C(i,j) = \begin{cases} 
E(i,j) \text{ XOR } I(i,j), & \text{if } X(i,j) = 0 \\
E(i,j), & \text{if } X(i,j) = 1, \ i=1,2,...,M, \ j=1,2,...,N
\end{cases}
\]

### 2.5. **Diffusion Operation**

As shown in Fig. 1a, the diffusion-I operation converts the matrix \(A\) into the matrix \(B\) with \(Y\), i.e.

\[
B(1,1) = (A(1,1) + Y(1,1)) \text{ mod} \ 256
\]

\[
B(1,j) = (A(1,j) + Y(1,j) + B(j-1,j)) \text{ mod} \ 256, \ j=2,3,...,N
\]

\[
B(i,1) = (A(i,1) + Y(i,1) + B(i-1,1) + B(i-1,N)) \text{ mod} \ 256, \ i=2,3,...,M
\]

\[
B(i,j) = (A(i,j) + Y(i,j) + B(i-1,j) + B(i,j-1)) \text{ mod} \ 256, \ i=2,3,...,M, \ j=2,3,...,N
\]

The diffusion-II operation converts the matrix \(D\) into the matrix \(E\) with \(Z\), i.e.

\[
E(M,N) = (D(M,N) + Z(M,N)) \text{ mod} \ 256
\]

\[
E(M,j) = (D(M,j) + Z(M,j) + E(M,j+1)) \text{ mod} \ 256, \ j=N-1,N-2,...,2,1
\]

\[
E(i,N) = (D(i,N) + Z(i,N) + E(i+1,N) + E(i+1,1)) \text{ mod} \ 256, \ i=M-1,M-2,...,2,1
\]

\[
E(i,j) = (D(i,j) + Z(i,j) + E(i+1,j) + E(i+1,j)) \text{ mod} \ 256, \ i=M-1,M-2,...,2,1, \ j=N-1,N-2,...,2,1
\]

### 2.6. **Confusion Operation**

The plaintext-related confusion operation converts \(B\) into \(D\) with \(I\) by the following steps.

**Step 1.** For each coordinate \((i,j)\) of \(B\), generate a new coordinate \((m,n)\) by the following equations.

\[
m = (\text{sum}(B(i,1 \text{ to } N)) - B(i,j) + \text{sum}(I(i,1 \text{ to } N)) + I(i,j)) \text{ mod} \ M
\]

\[
n = (\text{sum}(B(1 \text{ to } M,j)) - B(i,j) + \text{sum}(I(1 \text{ to } M,j)) + I(i,j)) \text{ mod} \ N
\]

If \(m \neq i\) and \(n \neq j\), then exchange \(B(i,j)\) and \(B(m,n)\); otherwise, do nothing.
Step 2. Perform the Step 1 for each pixel in $B$ according the line scanning order to obtain a new matrix, denoted by $D$.

3. Simulation Test
The computer used is configured with Intel Core i7-4720HQ, 8GB DDR3L Memory, Windows 10 (64-bit), Visual Studio 2017 (Community) and C# language. The typical encryption/decryption results are as shown in Fig. 2. Where, the secret key $K=\{545404223,3922919431,2715962281,418932849,1196140742,2348838239,4112460543,4144164702,676943031,509\}$ (in decimal format), and the public key $IV=0.9058$. The images of Lena and Baboon are both of size 256×256.

![Image](image.png)

Figure 2. Simulation results. (a) Lena; (b) Baboon; (c-d) Cipher images of (a-b), respectively; (e-f) Recovered images of (c-d), respectively.

As shown in Fig. 2, the cipher images are noise-like ones, and the recovered images are identical to the original ones.

4. Security Analysis
To save space, the statistical characteristics, such as information entropy, correlation and histogram analysis, are not analyzed here. This section mainly discusses the following security features: key space, encryption /decryption speed, private/public key sensitivities and plain/cipher image sensitivities.

4.1. Key Space
Due to the key length of 300-bit, the key space size of proposed system is $2^{300}$, which is much larger than AES (for the key space size of AES is no more than $2^{256}$).

4.2. Encryption/decryption Speed
Without loss of generality, the secret key and public key are the same with those used in Sect. 3, and the Lena of size 256×256 is used as the test image. The encryption/decryption speed is listed in Table 1. It can be seen from Table 1, the proposed system is faster than AES. In the proposed system, the key stream generator can produce the pseudo-random matrices in advance, so the proposed system (excluding key stream generator) is more than triple the speed of AES.

|                         | Encryption speed | Decryption speed |
|-------------------------|------------------|------------------|
| Proposed system         | 14.0589          | 13.8998          |
| Proposed (excluding key stream generator) | 24.3845          | 24.0865          |
| AES                     | 7.2496           | 6.6223           |

4.3. Secret Key Sensitivity
In the secret key sensitivity test, the public key keeps unchanged, and the images of Lena, Baboon, Pepper, Plane, all-black and all-white images are used. To save space, here only the secret key sensitivity in encryption process is discussed. Firstly, random generate 100 private keys. Secondly, slightly change one bit of each key to get two secret keys. Thirdly, encrypt the tested images with the two keys to get two corresponding cipher images. Fourthly, calculate the values of NPCR and UACI.
between these two cipher images [13]. Finally, calculate the average values of NPCR and UACI of 100 trials, then list the results in Table 2.

Table 2. Private key sensitivity test results (%).

|       | Lena       | Baboon     | Pepper     | Plane      | All-black  | All-white  | Theoretical |
|-------|------------|------------|------------|------------|------------|------------|-------------|
| NPCR  | 99.6089    | 99.6085    | 99.6059    | 99.6120    | 99.6074    | 99.6120    | 99.6094     |
| UACI  | 33.4675    | 33.4551    | 33.4483    | 33.4649    | 33.4481    | 33.4565    | 33.4635     |

It can be seen from Table 2, the calculated results are fairly close to the theoretical values, demonstrating that the proposed system has strong secret key sensitivity.

4.4. Public Key Sensitivity

In the public key sensitivity test, the secret key keeps unchanged, and the images of Lena, Baboon, Pepper, Plane, all-black and all-white images are used. To save space, here only the public key sensitivity in encryption process is discussed. Firstly, random generate 100 public keys. Secondly, slightly change each public key by $10^{-14}$ to get two public keys. Thirdly, encrypt the tested images with the two public keys and the secret key to get two corresponding cipher images. Fourthly, calculate the values of NPCR and UACI between these two cipher images. Finally, calculate the average values of NPCR and UACI of 100 trials, then list the results in Table 3.

Table 3. Public key sensitivity test results (%).

|       | Lena       | Baboon     | Pepper     | Plane      | All-black  | All-white  | Theoretical |
|-------|------------|------------|------------|------------|------------|------------|-------------|
| NPCR  | 99.6095    | 99.6129    | 99.6097    | 99.6096    | 99.6117    | 99.6107    | 99.6094     |
| UACI  | 33.4746    | 33.4666    | 33.4441    | 33.4731    | 33.4680    | 33.4638    | 33.4635     |

From Table 3, the calculated results are very close to the theoretical values, demonstrating that the proposed system has strong public key sensitivity.

4.5. Plaintext Sensitivity

In plaintext sensitivity test, the secret key and public key keep unchanged. When slightly change one pixel of the plain image, one uses the indicators of NPCR and UACI to measure the difference between the two resultant cipher images. The test results are listed in Table 4.

As shown in Table 4, the calculated results are far close to the theoretical values, demonstrating that the proposed system has strong plaintext sensitivity.

Table 4. Plaintext sensitivity test results (%).

|       | Lena       | Baboon     | Pepper     | Plane      | All-black  | All-white  | Theoretical |
|-------|------------|------------|------------|------------|------------|------------|-------------|
| NPCR  | 99.6103    | 99.6114    | 99.6148    | 99.6082    | 99.6095    | 99.6074    | 99.6094     |
| UACI  | 33.4625    | 33.4538    | 33.4816    | 33.4666    | 33.4578    | 33.4581    | 33.4635     |

4.6. Cipher-text Sensitivity

In cipher-text sensitivity test, the secret key and public key keep unchanged. When slightly change one pixel of the cipher image, one uses the indicators of NPCR and UACI to measure the difference between the two recovered images. The test results are listed in Table 5. As shown in Table 5, the calculated results are way close to the theoretical values, demonstrating that the proposed system has strong cipher-text sensitivity.
Table 5. Cipher-text sensitivity test results (%).

|          | Lena         | Baboon       | Pepper       |
|----------|--------------|--------------|--------------|
|          | Calculated  | Theoretical  | Calculated  | Theoretical  | Calculated  | Theoretical  |
| NPCR     | 99.6063     | 99.6094      | 99.6108     | 99.6094      | 99.6100     | 99.6094      |
| UACI     | 28.6894     | 28.6850      | 27.9195     | 27.9209      | 30.9148     | 30.9134      |

5. Conclusions
This paper proposed a new image cryptosystem combined with the secret key and public key. Each encryption uses a new public key. And the public key and the cipher image are transferred to the receiver through public channel. Even the enemy has the public key; he cannot generate the pseudo-random matrix $I$. The tests show that the proposed cryptosystem is highly sensitive to the secret key and public key, so it can frustrate the chosen/known plain or cipher image attacks.

6. Acknowledgments
This work was fully supported by the National Natural Science Foundation of China (Nos. 61762043 and 61562035), the Science and Technology Project of Education Department of Jiangxi Province, China (No. GJJ160426), and the Graduate Innovative Special Fund Project of Jiangxi Provence, China (No. YC2017-S226).

7. References
[1] Wang X and Wang Q 2014 Nonlinear Dyn. 75 567
[2] Hua Z and Zhou Y 2016 Information Sciences 339C 237
[3] Zhang Y 2017 Chinese J. Electronics 26 1022
[4] Ye G, Zhao H and Chai H 2016 Nonlinear Dyn. 83 2067
[5] Li C, Luo G, Qin K and Li C 2017 Nonlinear Dyn. 87 127
[6] Khan M and Shah T 2015 Neural Comput. Appl. 26 1137
[7] Zhang Y 2015 Optik 126 223
[8] Wang H, Xiao D, Chen X and Huang H 2018 Signal Processing 144 444
[9] Su X, Li W and Hu H 2017 Multimedia Tools Appl. 76 14021
[10] Norouzi B and Mirzakuchaki S 2017 Multimedia Tools Appl. 76 1817
[11] Zhang Y 2018 Information Sciences 450C 361
[12] Zhang Y and Tang Y 2018 Multimedia Tools Appl. 77 6647
[13] Chen G, Mao Y and Chui C 2004 Chaos Solitons Fractals 21 749