Cryptanalysis on an image block encryption algorithm based on spatiotemporal chaos*

Wang Xing-Yuan(王兴元)† and He Guo-Xiang(何国祥)‡

Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian 116024, *China*

(Received 9 August 2011; revised manuscript received 11 December 2011)

An image block encryption scheme based on spatiotemporal chaos has been proposed recently. In this paper, we analyse the security weakness of the proposal. The main problem of the original scheme is that the generated keystream remains unchanged for encrypting every image. Based on the flaws, we demonstrate a chosen plaintext attack for revealing the equivalent keys with only 6 pairs of plaintext/ciphertext used. Finally, experimental results show the validity of our attack.

Keywords: encryption, spatiotemporal chaos, chosen plaintext attack, equivalent keys

PACS: 05.45.–a, 05.45.Ac, 05.45.Vx **DOI:** 10.1088/1674-1056/21/6/060502

1. Introduction

Chaos is a definitive and similar random procedure which appears in a nonlinear system.[1−6] A chaotic system is characterized by sensitive dependence on the initial conditions and control parameters. Therefore, it has attracted much attention for cryptology. In 1989, Matthews[7] first proposed the chaos-based encryption scheme, and Fridrich[8] first introduced the chaotic map into image encryption in 1997. Since then, many chaos-based image encryption algorithms have been designed to realize secure communication through public channels.

However, most of these schemes are based on low-dimensional chaotic maps,[9,10] whose drawbacks such as small key space, weak security, and so on are obvious. Hence high-dimensional chaotic systems,[11−13] multiple chaotic systems,[14,15] and spatiotemporal chaotic systems[16−20] have been proposed to improve the security of chaos-based encryption schemes. Unfortunately, many of them have been found to be insecure.

In this paper, we analyse the image block encryption scheme based on spatiotemporal chaos proposed recently in Ref. [19]. The cryptosystem first divides the image into blocks, and then uses the block number as the spatial parameter of coupled map lattice (CML) to iterate the chaos system. The chaos sequence and the former block plaintext co-determine the substitution and diffusion of each block. Although it has many advantages, such as a large key space and high key sensitivity, and so on, unfortunately, we find that this method is vulnerable to the chosen plaintext attack.

The outline of this paper is as follows. In next section, we describe the cryptosystem under study. In Section 3 we demonstrate a chosen plaintext attack that reveals the equivalent keys. In Section 4, we illustrate the success in our break with experimental results. Some security weaknesses and mistakes of the scheme under study are given in Section 5. Finally, we conclude the paper.

2. Wang and Teng's cryptosystem

In Wang and Teng's cryptosystem, a CML is adopted, which can be described as

$$x_{n+1}(i) = (1-\varepsilon)(f(x_n(i))) + \frac{\varepsilon}{2}(f(x_n(i-1)) + f(x_n(i+1))), \quad (1)$$

where n is the time index, i $(i = 1, 2, \ldots, L)$ is the lattice site index, $\varepsilon \in (0, 1)$ is a coupling constant, $x_n(i)$ represents the state variable for the i-th site at

*Project supported by the National Natural Science Foundation of China (Grant Nos. 61173183, 60973152, and 60573172), the Doctoral Program Foundation of Institution of Higher Education of China (Grant No. 20070141014), and the Natural Science Foundation of Liaoning Province, China (Grant No. 20082165).

†Corresponding author. E-mail: wangxy@dlut.edu.cn

‡Corresponding author. E-mail: hegxabc@126.com

© 2012 Chinese Physical Society and IOP Publishing Ltd

time n ($n = 1, 2, \ldots$). The periodic boundary condition $x_n(0) = x_n(L)$ is assumed. The local mapping function $f(x)$ is chosen to be the logistic map and expressed as

$$x_{n+1} = ax_n(1 - x_n), \tag{2}$$

with parameter $a \in (3.5699456, 4]$ and $x \in (0, 1)$, and the system is in chaotic state.

2.1. Preprocess of the plain image and chaotic sequence

Assume that the size of the plain image matrix $\boldsymbol{P}$ is $M \times N$, where $\boldsymbol{P}$ is divided into $m \times n$ ($m, n > 5$) blocks, with the size of each block being $M/m \times N/n$ (M, N are divisible by m, n, respectively). Each block is represented as matrix $\boldsymbol{P}_i$ ($0 \leq i < m \times n$).

The number of CML lattice L is set to be $m \times n$ which is equal to the block number of the divided image. Initialize the system with parameters a, ε and value $\{x_0(1), x_0(2), \ldots, x_0(m \times n - 1)\}$ to iterate the chaotic system $500 + M/m \times N/n$ times, discard the former 500 values to avoid harmful effects.

The output sequence of CML is transformed into an integer sequence as follows:

$$x'_n(i) = x_n(i) \times 10^{14} \bmod 256. \tag{3}$$

Then arrange the integer sequence $x'_n(i)$ generated by each lattice i from left to right, from top to bottom to form a matrix $\boldsymbol{X}_i$ ($0 \leq i < m \times n$) of size $M/m \times N/n$.

2.2. Substitution of the image blocks

Shuffle the plain image matrix $\boldsymbol{P}$ by vector $\boldsymbol{T}$ as $\boldsymbol{P'} = \boldsymbol{T}(\boldsymbol{P})$, where $\boldsymbol{T} = \{t_0, t_1, \ldots, t_{m \times n-1}\}$. For block $\boldsymbol{P}_i$, the shuffling process can be described as $\boldsymbol{P}'_i = t_i(\boldsymbol{P}_i)$. If t_i is row transformation, $t_i = \{t_{i,1}, t_{i,2}, \ldots, t_{i,M/m}\}$, otherwise, t_i is column transformation, $t_i = \{t_{i,1}, t_{i,2}, \ldots, t_{i,N/n}\}$.

The last output value $x_{M/m \times N/n}(i)$ of each lattice i and the last pixel value of the $(i-1)$-th block are combined to determine the substitution of the i-th block. Use

$$\theta = \left(x_{M/m \times N/n}(i) + \frac{P_{i-1}[M/m - 1][N/n - 1]}{256}\right) \bmod 1 \tag{4}$$

as reference, if $\theta > 0.5$ then corresponding block $\boldsymbol{P}_i$ will be permuted by row transformation, otherwise $\boldsymbol{P}_i$ is permuted by column transformation. After being permuted, the block $\boldsymbol{P}_i$ is changed into $\boldsymbol{P}'_i$.

(i) Row transformation: we take out the M/m values $\{x_{501}(i), x_{502}(i), \ldots, x_{500+M/m}(i)\}$ from the output sequence of the i-th lattice, and arrange these M/m values in an order and obtain $\{\bar{x}_{501}(i), \bar{x}_{502}(i), \ldots, \bar{x}_{500+M/m}(i)\}$. We find the position of values $\{\bar{x}_{501}(i), \bar{x}_{502}(i), \ldots, \bar{x}_{500+M/m}(i)\}$ in $\{x_{501}(i), x_{502}(i), \ldots, x_{500+M/m}(i)\}$ and mark down the transform position $TM = \{t_1, t_2, \ldots, t_{M/m}\}$. Then rearrange the row of the i-th image block according to $\boldsymbol{TM}$.

(ii) Column transformation: we take out the N/n values $\{x_{501}(i), x_{502}(i), \ldots, x_{500+N/n}(i)\}$ from the output sequence of the i-th lattice, and arrange these N/n values in an order and obtain $\{\bar{x}_{501}(i), \bar{x}_{502}(i), \ldots, \bar{x}_{500+N/n}(i)\}$. We find the position of values $\{\bar{x}_{501}(i), \bar{x}_{502}(i), \ldots, \bar{x}_{500+N/n}(i)\}$ in $\{x_{501}(i), x_{502}(i), \ldots, x_{500+N/n}(i)\}$ and mark down the transform position $TN = \{p_1, p_2, \ldots, p_{N/n}\}$. Then rearrange the column of the i-th image block according to $\boldsymbol{TN}$.

2.3. Diffusion of image blocks

We use the transformed image matrix $\boldsymbol{X}_i$ to diffuse the permuted block $\boldsymbol{P}'_i$. The corresponding elements of these matrixes are processed. $\boldsymbol{C}_i$ is the cipher image block matrix. The encryption transformation is

$$\begin{aligned} &\boldsymbol{C}_i = (\boldsymbol{P}'_i + \boldsymbol{X}_i + \boldsymbol{C}_{i-1} + \boldsymbol{P}'_{i-1}) \bmod 256, \\ &\boldsymbol{C}_{-1} = 0,\ \boldsymbol{P}'_{-1} = 0. \end{aligned} \tag{5}$$

The decryption proceeds in a way similar to encryption but in reverse order. For more details about the encryption/decryption procedures, readers may refer to Ref. [19].

3. Chosen plaintext attack

In the chosen plaintext attack, some plain images are attacked and the corresponding ciphered images are obtained. By analysing the plaintext/ciphertext pairs, the attack can reveal the secret parameters.

For the purpose of the attack, we assume that the plain image matrix is $\boldsymbol{P}$, and the ciphered image matrix is $\boldsymbol{C}$. If the row number M/m and the column number N/n of each block are known to be attacked, the ciphered image matrix $\boldsymbol{C}$ can be divided into $\{\boldsymbol{C}_0, \boldsymbol{C}_1, \ldots, \boldsymbol{C}_{m \times n-1}\}$, with the size of each block $\boldsymbol{C}_i$ being $M/m \times N/n$. If the the matrix $\boldsymbol{X}$ is known to be attacked, the output $\boldsymbol{P'}$ of the shuffling procedure can be easily obtained by using the reversed operation of Eq. (5). If shuffling vector $\boldsymbol{T}$ is known, the plain image matrix $\boldsymbol{P}$ can be recovered from the shuffled image $\boldsymbol{P'}$. Therefore, there are only three secret parameters to be extracted, and their blocks each have a

size $M/m \times N/n$, thus matrix $\boldsymbol{X}$ is obtained from the spatiotemporal chaos system and the shuffling vector $\boldsymbol{T}$. The attack consists of three steps, which are shown below in detail.

3.1. Extracting $M/m \times N/n$

Choose two plain images each with a size of $M \times N$ to encrypt, whose pixel matrixes are donated as $\boldsymbol{PA}$ and $\boldsymbol{PB}$ respectively and the corresponding ciphered image matrixes are $\boldsymbol{CA}$ and $\boldsymbol{CB}$, respectively. Let $\boldsymbol{PA}$ be composed of all "a" and $\boldsymbol{PB}$ of all "b", where $0 \leq a \leq 255$, $b = (a+1) \bmod 256$. In this way, the shuffling procedure does not work for the chosen plain images, $\boldsymbol{PA}$ and $\boldsymbol{PB}$, that is to say, $\boldsymbol{PA}' = \boldsymbol{PA}$ and $\boldsymbol{PB}' = \boldsymbol{PB}$. Although the size of each image block is unknown, we can compute $(\boldsymbol{CB} - \boldsymbol{CA}) \bmod 256$ as follows:

$$
\begin{aligned}
&\boldsymbol{CA}_0 = (a + \boldsymbol{X}_0) \bmod 256,\\
&\boldsymbol{CB}_0 = (b + \boldsymbol{X}_0) \bmod 256,\\
&(\boldsymbol{CB}_0 - \boldsymbol{CA}_0) \bmod 256\\
&\quad = (b + \boldsymbol{X}_0 - a - \boldsymbol{X}_0) \bmod 256 = 1,\\
&\boldsymbol{CA}_1 = (2a + \boldsymbol{X}_1 + \boldsymbol{CA}_0) \bmod 256,\\
&\boldsymbol{CB}_1 = (2b + \boldsymbol{X}_1 + \boldsymbol{CB}_0) \bmod 256,\\
&(\boldsymbol{CB}_1 - \boldsymbol{CA}_1) \bmod 256\\
&\quad = (2b + \boldsymbol{CB}_0 - 2a - \boldsymbol{CA}_0) \bmod 256 = 3,\\
&\boldsymbol{CA}_2 = (2a + \boldsymbol{X}_2 + \boldsymbol{CA}_1) \bmod 256,\\
&\boldsymbol{CB}_2 = (2b + \boldsymbol{X}_2 + \boldsymbol{CB}_1) \bmod 256,\\
&(\boldsymbol{CB}_2 - \boldsymbol{CA}_2) \bmod 256\\
&\quad = (2b + \boldsymbol{CB}_1 - 2a - \boldsymbol{CA}_1) \bmod 256 = 5,\\
&\cdots\\
&\boldsymbol{CA}_{m\times n-1} = (2a + \boldsymbol{X}_{m\times n-1} + \boldsymbol{CA}_{m\times n-2}) \bmod 256,\\
&\boldsymbol{CB}_{m\times n-1} = (2b + \boldsymbol{X}_{m\times n-1} + \boldsymbol{CB}_{m\times n-2}) \bmod 256,\\
&(\boldsymbol{CB}_{m\times n-1} - \boldsymbol{CA}_{m\times n-1}) \bmod 256\\
&\quad = (2b + \boldsymbol{CB}_{m\times n-2} - 2a - \boldsymbol{CA}_{m\times n-2}) \bmod 256\\
&\quad = (2mn - 1) \bmod 256. \qquad (6)
\end{aligned}
$$

From Eq. (6), we can see that the result of $(\boldsymbol{CB} - \boldsymbol{CA}) \bmod 256$ is also divided into $m \times n$ blocks, as are the plain images. The pixel values in each block are the same, and the difference between two adjacent blocks is equal to a constant 2.



Fig. 1. Result of $(\boldsymbol{CB} - \boldsymbol{CA}) \bmod 256$ when $a = 0$ and $b = 1$.

To give a better demonstration, here we set a= 0 and b= 1, the result of $(\boldsymbol{CB} - \boldsymbol{CA}) \bmod 256$ is shown in Fig. 1. By analysing the result of $(\boldsymbol{CB} - \boldsymbol{CA}) \bmod 256$, we can easily obtain the row number M/m and column number N/n of each block.

3.2. Extracting matrix $\boldsymbol{X}$

The diffusion step in the cryptosystem under study is designed in the propagating cipher-block chaining (PCBC) model[21] by using the previous ciphertext block $\boldsymbol{C}_{i-1}$, the previous shuffled plaintext block $\boldsymbol{P}'_{i-1}$ and the keystream $\boldsymbol{X}_i$ to encrypt the present shuffled plaintext block $\boldsymbol{P}'_i$. The flowchart of the encryption in PCBC mode is given in Fig. 2.

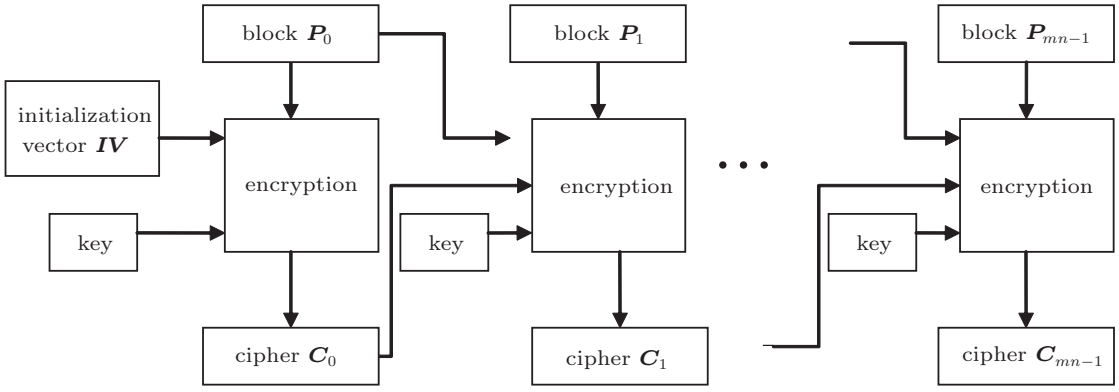


Fig. 2. Propagating PCBC model encryption.

We invoke a plaintext set $\boldsymbol{PD} = \{\boldsymbol{PD}_0, \boldsymbol{PD}_1, \ldots, \boldsymbol{PD}_{mn-1}\}$, whose element $\boldsymbol{PD}_i$ is of size $M/m \times N/n$ and each element $\boldsymbol{PD}_i$ is set to be 0. Encrypt $\boldsymbol{PD}$ and represent the corresponding ciphertext as $\boldsymbol{CD} = \{\boldsymbol{CD}_0, \boldsymbol{CD}_1, \ldots, \boldsymbol{CD}_{mn-1}\}$. It is obvious that the shuffled image matrix $\boldsymbol{PD}' = \boldsymbol{PD} = \{0, 0, \ldots, 0\}$.

According to Eq. (5), the obtained ciphertext $\boldsymbol{CD}$ can be further represented as follows:

$$\begin{aligned}
&\boldsymbol{CD}_0 = (\boldsymbol{PD}'_0 + \boldsymbol{X}_0 + \boldsymbol{IV}) \bmod 256,\\
&\boldsymbol{IV} = C_{-1} + \boldsymbol{PD}_{-1},\\
&\boldsymbol{CD}_1 = (\boldsymbol{PD}'_1 + \boldsymbol{X}_1 + \boldsymbol{CD}_0 + \boldsymbol{PD}'_0) \bmod 256,\\
&\boldsymbol{CD}_2 = (\boldsymbol{PD}'_2 + \boldsymbol{X}_2 + \boldsymbol{CD}_1 + \boldsymbol{PD}'_1) \bmod 256,\\
&\cdots
\end{aligned} \tag{7}$$

Because $\boldsymbol{PD}'_i = \boldsymbol{PD}_i = 0$, the obtained ciphertext $\boldsymbol{CD}$ can be rearranged as follows:

$$\begin{aligned}
&\boldsymbol{CD}_0 = (0 + \boldsymbol{X}_0 + 0) \bmod 256 = \boldsymbol{X}_0,\\
&\boldsymbol{IV} = 0 + 0 = 0,\\
&\boldsymbol{CD}_1 = (0 + \boldsymbol{X}_1 + \boldsymbol{X}_0 + 0) \bmod 256\\
&\qquad = (\boldsymbol{X}_1 + \boldsymbol{X}_0) \bmod 256,\\
&\boldsymbol{CD}_2 = (0 + \boldsymbol{X}_2 + \boldsymbol{X}_1 + \boldsymbol{X}_0 + 0) \bmod 256\\
&\qquad = (\boldsymbol{X}_2 + \boldsymbol{X}_1 + \boldsymbol{X}_0) \bmod 256,\\
&\cdots
\end{aligned} \tag{8}$$

The matrix $\boldsymbol{X} = \{\boldsymbol{X}_0, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_{mn-1}\}$ can be easily recovered as follows:

$$\begin{aligned}
&\boldsymbol{X}_0 = \boldsymbol{CD}_0,\\
&\boldsymbol{X}_1 = (\boldsymbol{CD}_1 - \boldsymbol{CD}_0) \bmod 256,\\
&\boldsymbol{X}_2 = (\boldsymbol{CD}_2 - \boldsymbol{CD}_1 - \boldsymbol{CD}_0) \bmod 256,\\
&\cdots
\end{aligned} \tag{9}$$

3.3. Extracting shuffling vector $\boldsymbol{T}$

We first obtain the range of the last output value $\boldsymbol{X}_{M/m\times N/n}(i)$ of each lattice i, we invoke a plaintext set $\boldsymbol{PE} = \{\boldsymbol{PE}_0, \boldsymbol{PE}_1, \ldots, \boldsymbol{PE}_{mn-1}\}$, whose element $\boldsymbol{PE}_i$ is of size $M/m \times N/n$, each element $\boldsymbol{PE}_i$ is set to be as follows:

$$\boldsymbol{PE}_i = \begin{pmatrix} M/m-1 & M/m-1 & \cdots & M/m-1 \\ M/m-2 & M/m-2 & \cdots & M/m-2 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}.$$

Encrypt $\boldsymbol{PE}$ and obtain the corresponding ciphertext represented as $\boldsymbol{CE} = \{\boldsymbol{CE}_0, \boldsymbol{CE}_1, \ldots, \boldsymbol{CE}_{mn-1}\}$. Now that we have known the matrix $\boldsymbol{X}$, we can invert the diffusion procedure. Hence we can easily obtain the output $\boldsymbol{PE}'$ of the shuffling procedure by using the reversed operation of Eq. (5). If $\boldsymbol{PE}'_i = \boldsymbol{PE}_i$, we know that the i-th block $\boldsymbol{PE}_i$ is implemented by column transformation, which indicates that θ of the i-th block obtained by Eq. (5) is less than or equal to 0.5. Otherwise, $\boldsymbol{PE}_i$ is implemented by row transformation, which indicates that $0.5 < \theta < 1$. With the help of Eq. (4), we can obtain $\boldsymbol{X}_{M/m\times N/n}(i)$ as

$$\begin{aligned}
&\boldsymbol{X}_{M/m\times N/n}(i)\\
&= (\theta - \boldsymbol{PE}_{i-1}[M/m-1][N/n-1]/256) \bmod 1\\
&= (\theta - 0) \bmod 1 = \theta.
\end{aligned} \tag{10}$$

So we can see that the range of $\boldsymbol{X}_{M/m\times N/n}(i)$ is the same as that of θ.

Next, we can recover the shuffling vector $\boldsymbol{T}$. The detailed procedure is described below. The flowchart of attacking the item t_i of the shuffling vector $\boldsymbol{T}$ is given in Fig. 3.

Step 1 Construct two plain images $\boldsymbol{PH}$ and $\boldsymbol{PV}$;

Step 2 Set i= 0;

Step 3 According to the range of $\boldsymbol{X}_{M/m\times N/n}(i)$ obtained above, we set the values of blocks $\boldsymbol{PH}_i$ and $\boldsymbol{PV}_i$:

$$\boldsymbol{PH}_i = \begin{pmatrix} M/m-1 & M/m-1 & \cdots & M/m-1 \\ M/m-2 & M/m-2 & \cdots & M/m-2 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix},$$

and

$$\boldsymbol{PV}_i = \begin{pmatrix} 0 & 1 & \cdots & N/n-1 & N/n \\ 0 & 1 & \cdots & N/n-1 & N/n \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 1 & \cdots & N/n-1 & N/n \\ 0 & 1 & \cdots & N/n-1 & 128 \end{pmatrix},$$

if $0.5 < \boldsymbol{X}_{M/m\times N/n}(i) < 1$;

$$\boldsymbol{PH}_i = \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & \cdots & 1 & 1 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ M/m-2 & M/m-2 & \cdots & M/m-2 & M/m-2 \\ M/m-1 & M/m-1 & \cdots & M/m-1 & 128 \end{pmatrix},$$

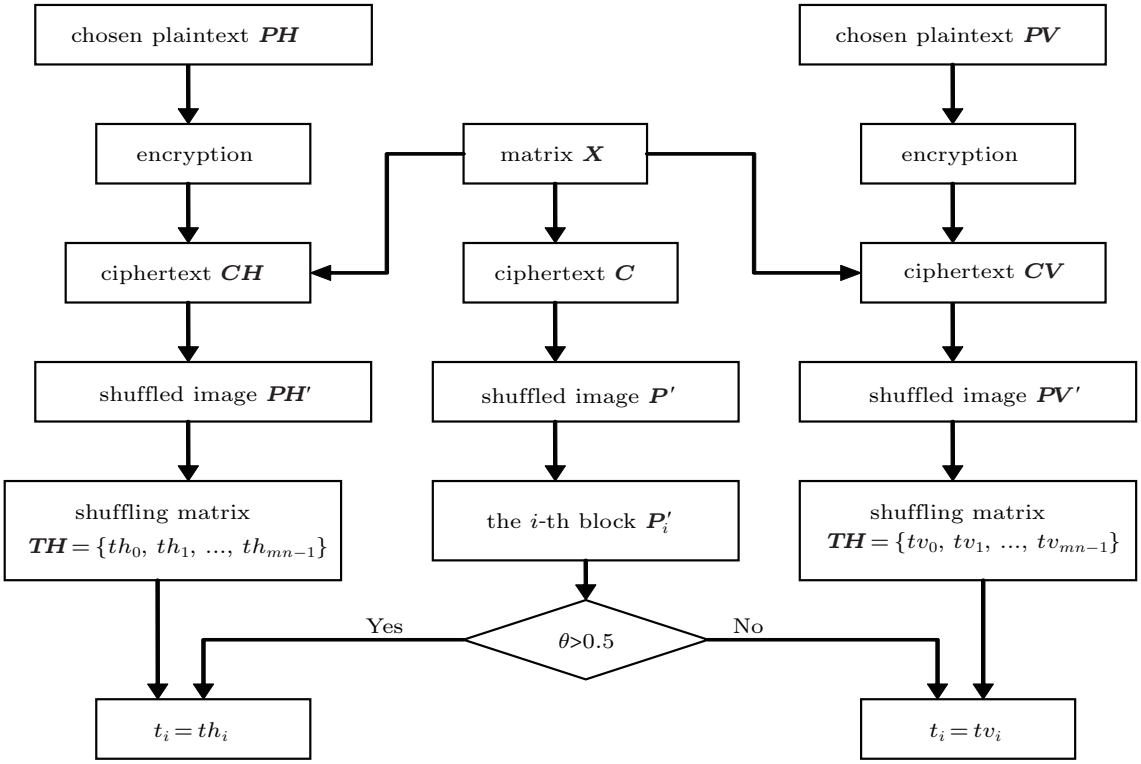


Fig. 3. Flowchart of attacking item t_i of the shuffling vector $\boldsymbol{T}$.

and

$$\boldsymbol{PV}_i = \begin{pmatrix} N/n & N/n-1 & \cdots & 0 \\ N/n & N/n-1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ N/n & N/n-1 & \cdots & 0 \end{pmatrix},$$

if $\boldsymbol{X}_{M/m\times N/n}(i) \le 0.5$.

In this way, all blocks of $\boldsymbol{PH}$ are implemented by row transformation and all blocks of $\boldsymbol{PV}$ are implemented by column transform.

Step 4 If $i \ge m \times n$, go to Step 5; otherwise, set $i = i + 1$, go to Step 3;

Step 5 Encrypt $\boldsymbol{PH}$ and $\boldsymbol{PV}$ and obtain the corresponding ciphertext $\boldsymbol{CH}$ and $\boldsymbol{CV}$ respectively. Now that the matrix $\boldsymbol{X}$ is obtained in Subsection 3.2, we can easily obtain the output $\boldsymbol{PH}'$ and $\boldsymbol{PV}'$ of the shuffling procedure, using the reversed operation of Eq. (5);

Step 6 Set $j = 0$;

Step 7 The row transformation vector th_j permutes the j-th block of $\boldsymbol{PH}$, since each entry in the first column is distinct, the location of the first column of $\boldsymbol{PH}'_j$ reveals the permutation th_j. For example, if $\boldsymbol{X}_{M/m\times N/n}(j) \le 0.5$, $th_{j,k} = \boldsymbol{PH}'_{j_{K,0}} + 1$, where $1 \le k \le M/m$, and $\boldsymbol{PH}'_{j_{K,0}}$ represents the k-th row first column of $\boldsymbol{PH}'_j$; Otherwise, $0.5 < \boldsymbol{X}_{M/m\times N/n}(j) < 1$, $th_{j,k} = M/m - \boldsymbol{PH}'_{j_{K,0}}$.

Step 8 The column transformation vector tv_j permutes the i-th block of $\boldsymbol{PV}$, since each entry in the first row is distinct, the location of the first row of $\boldsymbol{PV}'_j$ reveals the permutation tv_j. For example, if $0.5 < \boldsymbol{X}_{M/m\times N/n}(j) < 1$, $tv_{j,k} = \boldsymbol{PV}'_{j_{0,k}} + 1$, where $1 \le k \le N/n$, and $\boldsymbol{PV}'_{j_{0,k}}$ represents the k-th column first row of $\boldsymbol{PV}'_j$; otherwise, $\boldsymbol{X}_{M/m\times N/n}(j) \le 0.5$, and $th_{j,k} = N/n - \boldsymbol{PV}'_{j_{0,k}}$.

Step 9 If $j \ge m \times n$, we obtain two vectors, $\boldsymbol{TH} = \{th_0, th_1, \ldots, th_{mn-1}\}$ and $\boldsymbol{TV} = \{tv_0, tv_1, \ldots, tv_{mn-1}\}$, then go to Step 10; otherwise, set $j = j + 1$, go to Step 7;

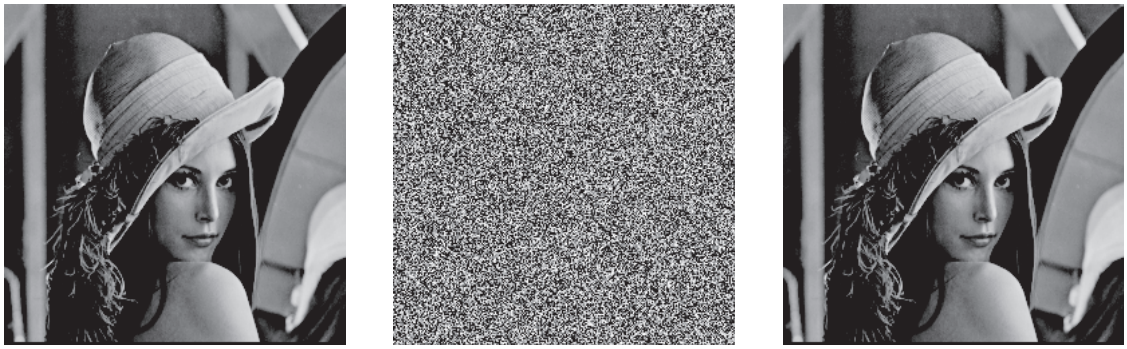
Step 10 Set $q = 0$;

Step 11 Since the range of $\boldsymbol{X}_{M/m\times N/n}(q)$ is acquired above and the last pixel value of $(q-1)$-th block $\boldsymbol{P}_{q-1}[M/m-1][N/n-1]/256$ is known, the range of θ of current plain block $\boldsymbol{P}_q$ can be obtained using Eq. (6). If $\theta > 0.5$, the current plain block $\boldsymbol{P}_q$ is implemented by row transformation, so $t_q = th_q$; otherwise, the current plain block $\boldsymbol{P}_q$ is implemented by column transformation, hence $t_q = tv_q$;

Step 12 If $q \ge m \times n$, the attack procedure is ended, and the shuffling vector $\boldsymbol{T} = \{t_0, t_1, \ldots, t_{m\times n-1}\}$; otherwise, set $q = q + 1$, go to Step 11.

4. Experimental results

We encrypt an image of a lady called Lena, with a size of 256×256, by using the Wang and Teng's algorithm, and then break it with our chosen plaintext attack described above. The experiment is performed using MATLAB on a personal computer with a Pentium 4 CPU 1.70 GHz, 768 MB memory and 80 GB hard-disk capacity, and the operation system is Microsoft Windows XP. The original image, the cipher image and the image recovered are shown in Figs. 4(a), 4(b), and 4(c), respectively. Obviously, the recovered image is identical with the original one, which implies that the original algorithm is too feasible to be applied in image encryption.



(a) The plain image of Lena (b) The cipher image (c) The recovered image

Fig. 4. Experimental result of the chosen plaintext attack.

5. Security flaws and mistakes of the original scheme

In this section, we discuss some defects and mistakes of the scheme under study. Firstly, we give four defects of the scheme.

(I) As claimed in Ref. [19], row block number m and column block number n must be divisible by row number M and column number N of the image respectively, moreover, $m, n > 5$. These constraints led to the encryption failure of many images using the scheme proposed in Ref. [19], such as an image of size 256×5. Even though some images can be successfully encrypted using it, there are few partition approaches. For example, when the size of an image is 256×281, there are only 6 kinds of partition methods. Especially, when the row number and the column number of an image are both prime numbers, there is only one kind of partition approach, where $m = M$ and $n = N$, that is to say, a pixel is a block. All in all, there are a lot of restrictions on the size of the image.

(II) Matrix $\boldsymbol{X}$ generated from the spatiotemporal chaos system and used in the diffusion step is the same for each plaintext. That is to say, the generated keystream depends on neither the plaintext nor the ciphertext. The reuse of the keystream more than once makes it weak against chosen plaintext, chosen ciphertext and known plaintext attacks. In Subsection 3.3 of this paper, we recovered the keystream matrix $\boldsymbol{X}$ by a chosen plaintext attack and only a pair of plaintext/ciphertext is used.

(III) In the diffusion step of original scheme, because $\boldsymbol{C}_{-1} = 0$, $\boldsymbol{P}'_{-1} = 0$, and $\boldsymbol{IV} = \boldsymbol{C}_{-1} \oplus \boldsymbol{P}'_{-1}$, there are many pairs of $\boldsymbol{C}_{-1}/\boldsymbol{P}'_{-1}$ which satisfy the requirement $\boldsymbol{C}_{-1} \oplus \boldsymbol{P}'_{-1} = 0$. Hence it is not secure enough. To prevent some attacks, $\boldsymbol{IV} = C_{-1} \oplus \boldsymbol{P}'_{-1}$ should be generated randomly each time and should not be reused. In this way, $\boldsymbol{IV}$ is not predictable by anyone but except the sender and the receiver.

(IV) Given two plain images with only one pixel different at the position (x, y) of the i-th block, if (x, y) is the last position of a block, because all plaintexts before (x, y) are identical for the two plaintext images, the ciphertexts before (x, y) will also be identical; otherwise, there are only $m \times n - i$ different values between the two ciphertexts, where $m \times n$ represents the number of blocks.

There are two mistakes in Ref. [19], which are corrected in our paper.

i) The last sentence in Subsection 2.3 of Ref. [19] says moving t_1 column to the first column, t_2 to the second column. However, the authors also previously said the rearrangement of column of the i-th image block according to $\boldsymbol{TN}$, and $\boldsymbol{TN} = \{p_1, p_2, \ldots, p_{N/n}\}$. Hence, the sentence can correctly be written as move p_1 column to the first column, p_2 to the second column.

ii) Equation (4) in our paper, whose out-

put value decides the substitution of the i-th block, is incorrectly defined as $\theta = (x_{M\times N}(i) + \boldsymbol{P}_{i-1}[M/m-1][N/n-1]/256) \bmod 1$ (see Subsection 2.3 in Ref. [19]). As claimed in Ref. [19], $x(i)$ is generated from the CML lattice and each lattice generates $M/m \times N/n$ values (see Subsection 2.2 in Ref. [19]), so the last output value of each lattice i is $\boldsymbol{X}_{M/m\times N/n}(i)$, not $x_{M\times N}(i)$ any more.

6. Conclusion

In this paper, we gave a complete break of an image block encryption algorithm based on spatiotemporal chaos. The main weakness of the encryption scheme is that the keystream generated from the spatiotemporal chaos system depends on neither the plaintext nor the ciphertext, which makes it fixed and reused in encrypting every image. The security weakness reveals that the proposed encrypting algorithm in Ref. [19] is not secure against the chosen plaintext attack. In this paper, particularly, we use 6 pairs of plaintext/ciphertext for revealing the equivalent keys. The experimental results demonstrate that our attack can successfully decrypt the cipher image. To avoid the flaws in application, the encrypting algorithm needs some improvements.

References

[1] Zhang H G, Ma T D, Huang G B and Wang Z L 2010 *IEEE Trans. Syst. Man Cybern.* B **40** 831

[2] Zhang H G, Fu J, Ma T D and Tong S C 2009 *Chin. Phys.* B **18** 969

[3] Liu Y J, Chen C L P and Tong S C 2011 *IEEE Trans. Neural Network* **22** 1162

[4] Zhang H G, Xie Y, Wang Z and Zheng C 2007 *IEEE Trans. Neural Network* **18** 1841

[5] Wang K, Pei W J, Zhou J T, Zhang Y F and Zhou S Y 2011 *Acta Phys. Sin.* **60** 070503 (in Chinese)

[6] Jin J X and Qiu S S 2010 *Acta Phys. Sia.* **59** 792 (in Chinese)

[7] Matthews R 1989 *Cryptologia* **13** 29

[8] Fridrich J 1997 *The 1997 IEEE International Conference on Systems, Man, and Cybernetics,* October 12–15, 1997 Hyatt Orlando, Florida, USA, p. 1105

[9] Xiang T, Liao X F, Tang G, Chen Y and Wong K W 2006 *Phys. Lett.* A **349** 109

[10] Chen G R, Mao Y B and Chui C K 2004 *Chaos, Solitons & Fractals* **21** 749

[11] Sun F Y, Liu S T and Lü Z W 2007 *Chin. Phys.* **16** 3616

[12] Gao T G and Chen Z Q 2008 *Phys. Lett.* A **372** 394

[13] Guan Z H, Huang F J and Guan W J 2005 *Phys. Lett.* A **346** 153

[14] Huang C K and Nien H H 2009 *Opt. Commun.* **282** 2123

[15] Wang X Y and Yu Q 2009 *Commun. Nonlinear Sci. Number. Simul.* **14** 574

[16] Lian S G 2009 *Chaos, Solitons & Fractals* **40** 2509

[17] Sun F Y and Lü Z W 2011 *Chin. Phys.* B **20** 040506

[18] Li P, Li Z, Hulang W A and Chen G R 2007 *Chaos, Solitons & Fractals* **32** 1867

[19] Wang X Y and Teng L 2011 *Nonlinear Dynam.* **67** 365

[20] Sun F Y and Lü Z W 2011 *Acta Phys. Sin.* **60** 040503 (in Chinese)

[21] Rhouma R and Belghith S 2008 *Phys. Lett.* A **372** 5790