Cryptanalysis of a New Signal Security System for Multimedia Data Transmission*

Chengqing Li[1a], Shujun Li[2a], Guanrong Chen[2b], Gang Chen[1b] and Lei Hu[3]

[1]Department of Mathematics, Zhejiang University, Hangzhou 310027, China
[2]Department of Electronic Engineering, City University of Hong Kong, Hong Kong, China
[3]State Key Laboratory of Information Security, Graduate School of Chinese Academy of Sciences, Beijing 100049, China

Abstract

Recently, a new signal security system called TDCEA (two-dimensional circulation encryption algorithm) was proposed for real-time multimedia data transmission. This paper gives a comprehensive analysis on the security of TDCEA. The following security problems are found: 1) there exist some essential security defects in TDCEA; 2) two known-plaintext attacks can break TDCEA; 3) the chosen-plaintext and chosen-ciphertext versions of the aforementioned two known-plaintext attacks can break TDCEA even with a smaller complexity and a better performance. Some experiments are given to show the security defects of TDCEA and the feasibility of the proposed known-plaintext attacks. As a conclusion, TDCEA is not suitable for applications that require a high level of security.

keywords and phrases: TDCEA, image encryption, chaos, cryptanalysis, known/chosen-plaintext attack, multimedia

1 Introduction

In today's digital world, the security of multimedia data, e.g., digital speech, image and video files, becomes more and more important due to their frequent transmission over open networks. In some real applications, such as pay-TV, medical imaging systems, military image/database communications and confidential video conferences, highly secure and reliable storage and transmission of multimedia data are needed. To fulfill such a demand, many encryption schemes have been proposed as possible solutions [1–14]. Meanwhile, cryptanalysis work has also been developed, and some of the proposed schemes have been found to be insecure [9, 10, 15–25]. For a comprehensive survey of the state-of-the-art of image and video encryption, see [26].

The present paper focuses on a new signal security system recently proposed in [1, 2], which is called the two-dimensional circulation encryption algorithm (TDCEA). In fact, TDCEA is an enhanced version of a previous image encryption scheme proposed by the same authors in [3, 4], named BRIE (bit recirculation image encryption), which is the one-dimensional counterpart of TDCEA. The original BRIE scheme has been successfully cryptanalyzed in [19], showing its insecurity against known/chosen-plaintext attacks. Although TDCEA is more complicated than BRIE by using 2-D permutations, this paper will point out that such a 2-D generalization cannot enhance the security of BRIE against known/chosen-plaintext and chosen-ciphertext attacks. In addition, it will be shown that the security of TDCEA against brute-force attack was much overestimated in [1, 2]. Essentially, TDCEA is a permutation-only image cipher, which has been known to be insecure against known/chosen-plaintext attacks [15, 16, 23, 25].

The rest of this paper is organized as follows. The next section briefly introduces TDCEA and its 1-D version BRIE. Section 3 discusses some general security defects of TDCEA. Two known-plaintext and chosen-plaintext attacks are given in Secs. 4 and 5, respectively, with some experimental results for verification. Section 6 briefly discusses the chosen-ciphertext attack, a natural and simple generalization of the chosen-plaintext attack. The last section concludes the paper.

2 TDCEA

The basic idea used in TDCEA is secret bit rotations of every 64 consecutive bits (of 8 consecutive pixels), which are controlled by a chaotic pseudo-random binary sequence (PRBS). BRIE is the simplified version of TDCEA, by rotating only 8 bits in each pixel. To facilitate the description on TDCEA and BRIE, it is assumed that the plain-image has size $M \times N$, where M is the height and N is the width of the image.

2.1 Definitions and Notations

First, some definitions and notations are given in order to introduce TDCEA and BRIE. Assuming two matrices $\boldsymbol{M}$ and $\boldsymbol{M}'$ of size $m \times n$, where m is the height and n is the width, two mapping operations are defined as follows.

- *The horizontal rotation mapping*, $\text{Rotate}X_i^{p,r} : \boldsymbol{M} \to \boldsymbol{M}'$ $(0 \le i \le m-1)$, is defined to circularly rotate the i-th row of $\boldsymbol{M}$, in the left (when $p = 1$) or right (when $p = 0$) direction, by r elements.
- *The vertical rotation mapping*, $\text{Rotate}Y_j^{q,s} : \boldsymbol{M} \to \boldsymbol{M}'$ $(0 \le j \le n-1)$, is defined to circularly rotate the

*This paper has been published in *EURASIP Journal on Applied Signal Processing*, vol. 2005, no. 8, pp. 1277-1288, 2005. The corresponding author is Shujun Li, contact him via `http://www.hooklee.com`.

j-th column of $\boldsymbol{M}$, in the up (when $q=1$) or down (when $q=0$) direction, by s elements.

When $\boldsymbol{M}$ is a $1\times n$ vector, the 1-D version of the above 2-D rotation mapping is denoted by $ROLR_p^q:\boldsymbol{M}\to\boldsymbol{M}'$, which is defined to circularly rotate $\boldsymbol{M}$ in the left (when $p=0$) or right (when $p=1$) direction, by q elements.

2.2 The 1-D version of TDCEA – BRIE

Assuming the plain-image is $f=[f(x,y)]_{x=0,y=0}^{M-1,N-1}$ and the cipher-image is $f'=[f'(x,y)]_{x=0,y=0}^{M-1,N-1}$, BRIE is described as follows [3,4].

- *The secret key*: two integers α,β, and the initial condition $x(0)\in(0,1)$ of the following chaotic Logistic map:
$$x(k+1)=\mu\cdot x(k)\cdot(1-x(k)). \tag{1}$$
- *The Initialization procedure*: run the chaotic Logistic map from $x(0)$ to generate a chaotic sequence, $\{x(k)\}_{k=0}^{\lceil(MN+1)/8\rceil-1}$, where $\lceil a\rceil$ denotes the smallest integer that is not less than a. From the 8-bit binary representation of $x(k)$ as follows,
$$\begin{aligned}x(k) &= \sum_{i=0}^{7} b(8k+i)\cdot 2^{-i-1}\\ &= 0.b(8k+0)b(8k+1)\cdots b(8k+7),\end{aligned}$$
a PRBS is derived: $\{b(k)\}_{k=0}^{MN}$.
- *The encryption procedure*: for the plain-pixel $f(x,y)=\sum_{i=0}^{7}b_i\cdot 2^i$, the corresponding cipher-pixel $f'(x,y)=\sum_{i=0}^{7}b'_i\cdot 2^i$ is determined by the following equation:
$$\boldsymbol{M}'=ROLR_p^q(\boldsymbol{M}),$$
where $p=b(N\cdot x+y)$, $q=\alpha+\beta\cdot b(N\cdot x+y+1)$, and $\boldsymbol{M},\boldsymbol{M}'$ are two 1×8 bit matrices: $\boldsymbol{M}=[b_7,b_6,\cdots,b_0]$, $\boldsymbol{M}'=[b'_7,b'_6,\cdots,b'_0]$.
- *The decryption procedure* is denoted by
$$\boldsymbol{M}=ROLR_{1-p}^{q}(\boldsymbol{M}')=ROLR_p^{8-q}(\boldsymbol{M}').$$

In [19], BRIE was successfully cryptanalyzed and the following security problems were pointed out.

1. The key space is too small and the security against the brute-force attack was much over-estimated;
2. There exist some essential defects, which makes it possible for an attacker to get some visual information of the plain-image by observing the cipher-image;
3. BRIE is not secure against known/chosen-plaintext attacks, since only one known/chosen plain-image is enough to get an equivalent key, a mask array $\boldsymbol{Q}=[q(x,y)]_{x=0,y=0}^{M-1,N-1}$, where $q(x,y)$ satisfies
$$\boldsymbol{M}'=ROLR_0^{q(x,y)}(\boldsymbol{M})$$
and
$$\boldsymbol{M}=ROLR_1^{q(x,y)}(\boldsymbol{M}')=ROLR_0^{8-q(x,y)}(\boldsymbol{M}').$$
4. It is easy to get the sub-keys α, β and the most significant 8-bits of the chaotic state $x(k)$, as a replacement of the sub-key $x(0)$, from the mask array $\boldsymbol{Q}$ obtained above.

2.3 TDCEA

TDCEA [1,2] is an enhanced version of BRIE, by extending the bit rotation operations from one pixel to 8 consecutive pixels, and from two directions (left and right) to four directions (left, right, up and down).

TDCEA encrypts a plain-image block by block, where each block contains 8 consecutive pixels. To simplify the following description, without loss of generality, assume that MN can be divided by 8. Consider the 2-D plain-image $\{f(x,y)\}_{x=0,y=0}^{M-1,N-1}$ as a 1-D signal $\{f(l)\}_{l=0}^{MN-1}$ by scanning it in raster order[1]. Then, the plain-image can be divided into $MN/8$ blocks:
$$\{f^{(8)}(0),\cdots,f^{(8)}(k),\cdots,f^{(8)}(MN/8-1)\},$$
where
$$f^{(8)}(k)=\{f(8k+0),\cdots,f(8k+i),\cdots,f(8k+7)\}.$$
Rewrite each block $f^{(8)}(k)$ as an 8×8 bit matrix $\boldsymbol{M}_k=[M_k(i,j)]_{i=0,j=0}^{7,7}$, by assigning the 64 bits in the current block in the raster order: $f(8k+i)=\sum_{j=0}^{7}M_k(i,j)\cdot 2^j$. In the same way, the 8 pixels of each block of the cipher-image can be represented by an 8×8 bit matrix, $\boldsymbol{M}'_k=[M'_k(i,j)]_{i=0,j=0}^{7,7}$, where $f'(8k+i)=\sum_{j=0}^{7}M'_k(i,j)\cdot 2^j$. Based on the matrix-representations of the plain/cipher-images, the working mechanism of TDCEA can be described as follows.

- *The secret key*: two integers α,β, the initial condition $x(0)$, and the control parameter μ of the Logistic map (1), where $0<\alpha<8$, $0\le\beta<8$ and $0<\alpha+\beta<8$.
- *The initialization procedure*: run the Logistic map starting from $x(0)$ to generate a chaotic sequence, $\{x(k)\}_{k=0}^{MN/8-1}$, and then extract the 17-bit representation of $x(k)$ to yield a PRBS, $\{b(i)\}_{i=0}^{17MN/8-1}$. In the hardware implementation given in [1,2], the Logistic map is realized in 17-bit fixed-point arithmetic.
- *The encryption procedure*:
  - *Step 1 – horizontal rotations*: for $i=0\sim 7$ (i.e., for each value of i from 0 to 7, the same hereinafter) do $\boldsymbol{M}_k^*=\text{Rotate}X_i^{p,r}(\boldsymbol{M}_k)$, where $p=b(17k+i)$ and $r=\alpha+\beta\cdot b(17k+i+1)$;
  - *Step 2 – vertical rotations*: for $j=0\sim 7$ do $\boldsymbol{M}'_k=\text{Rotate}Y_j^{q,s}(\boldsymbol{M}_k^*)$, where $q=b(17k+8+j)$ and $s=\alpha+\beta\cdot b(17k+9+j)$.
- *The decryption procedure* is a simple reversion of the above encryption procedure, as follows:
  - *Step 1 – vertical rotations*: for $j=0\sim 7$ do $\boldsymbol{M}_k^*=\text{Rotate}Y_j^{q,s}(\boldsymbol{M}'_k)$, where $q=1-b(17k+8+j)$ and $s=\alpha+\beta\cdot b(17k+9+j)$;
  - *Step 2 – horizontal rotations*: for $i=0\sim 7$ do $\boldsymbol{M}_k=\text{Rotate}X_i^{p,r}(\boldsymbol{M}_k^*)$, where $p=1-b(17k+i)$ and $r=\alpha+\beta\cdot b(17k+i+1)$.

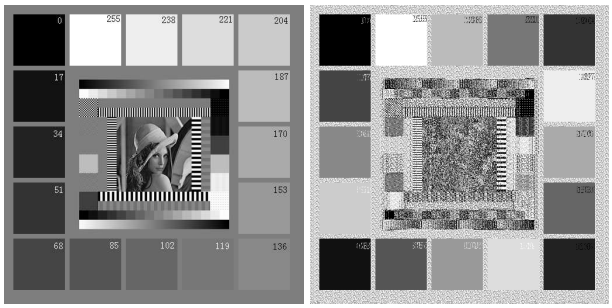
[1] Note that in [1,2] TDCEA is described directly for 1-D signals. In this paper, we prefer to explicitly mention the transform from 2-D images to 1-D signals, so as to emphasize the relation between BRIE and TDCEA (which is not mentioned in [1,2]).

3 Some Security Defects of TDCEA

3.1 Essential defects of circulations

In [19], some essential defects of the *ROLR* operation were found: 1) some plain-pixels may keep unchanged after encryption, so the plain-image will roughly emerge if there are too many such pixels; 2) for a sub-region in the plain-image with a fixed gray value, at most eight gray values[2] will be contained in the corresponding sub-region of the cipher-image, which will lead the edge of this sub-region to appear in the cipher-image. The second fact is also true for sub-regions with close pixel values.

Although TDCEA extends the shift operation to two dimensions, the above defects of *ROLR* cannot be completely removed. As an extreme example, when all elements in M_k are 0-bits or 1-bits, it is obvious that $M'_k \equiv M_k$, which means TDCEA cannot encrypt blocks with fixed pixel value 0 (black) or 255 (white) at all. To test the performance of TDCEA compared with BRIE, we have encrypted the same test image used in [19] for BRIE, with the following parameters: $(\alpha,\beta)=(2,4), x(0)=34816/2^{17}\approx 0.2656$, $\mu = 128317/2^{15}\approx 3.9159$. The encryption result is shown in Fig. 1, from which one can see that the 16 squares in the plain-image remain fixed in the cipher-image, though the fixed gray values have been changed for most squares. Comparing this result with those given in [19, Figure 1], it is obvious that the security defects of BRIE is not enhanced by TDCEA.



a) the plain-image b) the cipher-image

Figure 1: A special test image, "Test_pattern", encrypted by TDCEA

As a second example to test the possible enhancement of TDCEA on the BRIE security, we also tested the encryption performance of TDCEA on some general natural images containing many smooth areas. As known, the pixels within a smooth area generally have close pixel values, which are found similar to the squares with fixed gray values shown in Fig. 1 when TDCEA is applied for encryption. Two images, "House" and "Cameraman", are selected for testing. The experimental results are shown in Fig. 2, from which one can see many important edges of the plain-images emerging in the cipher-images. In this experiment, the parameters of TDCEA are as follows: $(\alpha,\beta)=(5,1)$, $x(0)=33578/2^{17}\approx 0.2562$ and $\mu = 129518/2^{15}\approx 3.9526$.

[2]For some pixel values, the number of different cipher pixel-values is even smaller, which may be 1, 2, or 4 [19, Sec. 3.1].

a) "House" b) Encrypted "House"

c) "Cameraman" d) Encrypted "Cameraman"

Figure 2: Two natural images, "House" and "Cameraman", encrypted by TDCEA, with $(\alpha,\beta)=(5,1)$, $x(0)=33578/2^{17}\approx 0.2562$ and $\mu = 129518/2^{15}\approx 3.9526$

3.2 Security Problem of α,β

In [1, 2], the values of α and β are constrained by $0<\alpha<8$, $0\le\beta<8$ and $0<\alpha+\beta<8$. Thus, the number of all possible values of (α,β) is $7+6+\cdots+2+1=28$. However, similar to the case of BRIE, α and β should also obey the following rule pointed out in [19]: $\alpha\neq 1,7$ or $\alpha+\beta\neq 1,7$. If this rule is not satisfied, then there only exists 1-bit circular rotations, since $\text{Rotate}X_i^{p,1}=\text{Rotate}X_i^{p,7}$ and $\text{Rotate}Y_j^{q,1}=\text{Rotate}Y_j^{q,7}$. Generally speaking, 1-bit circular rotations are not good enough to effectively encrypt the plain-image, and some visual information may leak from the cipher-image. When $(\alpha,\beta)=(1,6)$, $x(0)=33578/2^{17}\approx 0.2562$, $\mu=129518/2^{15}\approx 3.9526$, the encryption results of two plain-images, "House" and "Cameraman", are shown in Fig. 3. It can be seen that the visual information containing in the cipher-images is so much (even more than that in Fig. 2) that the plain-images can be obviously guessed. Excluding the three values of (α,β) that violate the above rule, $(1,0),(1,6),(7,0)$, the number of all "good" values of (α,β) is only 25 $(=28-3)$.

3.3 Low practical security against brute-force attacks

In [1, 2], it was claimed that the complexity of TDCEA against brute-force attack is $O\left(2^{17MN/8}\right)$ since $17MN/8$ secret bits are used in the encryption/decryption procedures. However, this statement is not true due to the following reason: all $17MN/8$ bits are uniquely determined by the initial condition $x(0)$ and the control parameter μ of the Logistic map (1), which have only 34 secret bits. Moreover, not all values of μ can ensure the chaoticity of the Logistic map [27], so we can assure that the number of possible different chaotic bit sequences is smaller than 2^{34}.

a) Encrypted "House" b) Encrypted "Cameraman"

Figure 3: Two natural images, "House" and "Cameraman", encrypted by TDCEA, when $(\alpha,\beta)=(1,6)$, $x(0)=33578/2^{17}\approx 0.2562$ and $\mu=129518/2^{15}\approx 3.9526$

Considering that the computational complexity of TDCEA is $O(MN)$, i.e., $49MN$ operations of all kinds [1, Sec.2.5], and the number of all possible values of (α,β) is 25, the total complexity against the brute-force attack is $O(2^{34}\cdot 25\cdot 49MN)\approx O(2^{44}MN)$. For a typical image of size 256×256, the complexity is about $O(2^{60})$, which is much smaller than $O(2^{17MN/8})=O(2^{139264})$, the complexity claimed in [1, 2]. Obviously, the security of TDCEA against brute-force attacks was over-estimated too much in [1, 2].

4 Known-Plaintext Attacks

The known-plaintext attack is the attack of reconstructing the secret key or its equivalent with some known plain-texts and their corresponding ciphertexts, which is practical and occurs more and more frequently in today's networked world [28]. Although it was claimed that TDCEA can efficiently resist this kind of attacks [1, Sec.2.6], we propose two different known-plaintext attacks in this section to effectively break TDCEA. One attack requires a few number of known plain-texts, and another requires only one.

4.1 Known-plaintext attack 1: Getting permutation matrices as an equivalent key

The insecurity of BRIE against known/chosen-plaintext attacks are caused by the fact that the *ROLR* operation is actually composed of secret permutations of all 8 bits of each pixel value. As shown in [25], all permutation-only ciphers are not secure against known/chosen-plaintext attacks. Apparently, TDCEA falls into the category of permutation-only ciphers, since the circulation rotations are actually secret permutations of all 64 bits of each 8-pixel block. As a result, if an attacker knows (or chooses) a number of plain-blocks and cipher-blocks at the same position, k, it is possible for him to partially (or even completely) reconstruct the bit permutation by comparing $\boldsymbol{M}_k$ and $\boldsymbol{M}'_k$. This is the basic principle of the first type of known/chosen-plaintext attacks to be discussed below.

Apparently, for the k-th pixel-block $f^{(8)}(k)$ and its cipher-block $f'^{(8)}(k)$, the encryption transformation can be represented by an 8×8 permutation matrix, $\boldsymbol{W}_k=[W_k(i,j)]_{i=0,j=0}^{7,7}$, where $W_k(i,j)=(i',j')$ denotes the secret position of the plain-bit $M_k(i,j)$ in $\boldsymbol{M}'_k$. Since there are $MN/8$ different blocks, the encryption of f can be represented by $MN/8$ permutation matrices: $\{\boldsymbol{W}_k\}_{k=0}^{MN/8-1}$. Once the attacker gets the $MN/8$ permutation matrices and their inverses, $\{\boldsymbol{W}_k^{-1}\}_{k=0}^{MN/8-1}$, he can use these matrices as an equivalent key to decrypt any cipher-image encrypted with the same key.

In [25], a general algorithm was proposed for deriving the secret permutations (i.e., the permutation matrices) from a number of known plain-images and the corresponding cipher-images. This algorithm depends on the fact that the secret permutations do not change the values of the permuted elements. As a result, one can compare the values of the elements of the plain-images and the cipher-images to reveal the secret permutations. Here, we show how to optimally realize the general algorithm for TDCEA and discuss the breaking performance.

Given n known plain-images $f_0\sim f_{n-1}$ and the corresponding cipher-images $f'_0\sim f'_{n-1}$, denoting the k-th 8×8 bit matrix of the l-th plain-image and cipher-image by $\boldsymbol{M}_{l,k}=[M_{l,k}(i,j)]_{i=0,j=0}^{7,7}$, $\boldsymbol{M}'_{l,k}=[M'_{l,k}(i,j)]_{i=0,j=0}^{7,7}$, respectively, the algorithm of deriving the permutation matrix $\boldsymbol{W}_k$ is described as follows.

- *Step 1a – calculate a generalized bit matrix* $\widetilde{\boldsymbol{M}}_k=\left[\widetilde{M}_k(i,j)\right]_{i=0,j=0}^{7,7}$, where $\widetilde{M}_k(i,j)=\sum_{l=0}^{n-1}M_{l,k}(i,j)\cdot 2^l$. Apparently, $\widetilde{M}_k(i,j)$ is an n-bit integer.

  Note: when n is larger than the word-length of the longest integer (which is 32 or 64 for most computers), it may be impossible to store $\widetilde{M}_k(i,j)$ as a normal integer in a computer. In this case, one has to divide $\widetilde{M}_k(i,j)$ into multiple short integers for storage and computation (i.e., to use long-integer techniques). Since the long-integer technique is easy for implementations and n is generally smaller than 32 in most attacking scenarios[3], here we do not pay special attention on this issue.

- *Step 1b – calculate a generalized bit matrix* $\widetilde{\boldsymbol{M}}'_k=\left[\widetilde{M}'_k(i,j)\right]_{i=0,j=0}^{7,7}$, in the same way as *Step 1a*.

- *Step 2 – get multi-valued permutation matrix*, $\widehat{\boldsymbol{W}}_k=\left[\widehat{W}_k(i,j)\right]_{i=0,j=0}^{7,7}$, where $\widehat{W}_k(i,j)=\left\{(i',j')\mid \widetilde{M}_k(i,j)=\widetilde{M}'_k(i',j')\right\}$.

- *Step 3 – derive an estimation of the permutation matrix* $\boldsymbol{W}_k$ *from* $\widehat{\boldsymbol{W}}_k$.

Apparently, if and only if each element of $\widehat{\boldsymbol{W}}_k$ contains only one pixel position, i.e., the measure of every element of $\widehat{\boldsymbol{W}}_k$ is 1, one can uniquely get the permutation matrix $\boldsymbol{W}_k$; otherwise, only an estimated version, $\widetilde{\boldsymbol{W}}_k$, can be derived. In other words, $\widetilde{\boldsymbol{W}}_k=\boldsymbol{W}_k$ holds if and only if the cardinality of $\widehat{\mathbb{W}}_k=\left\{\widehat{W}_k(0,0),\cdots,\widehat{W}_k(7,7)\right\}$ is 64, i.e., $\#\left(\widehat{\mathbb{W}}_k\right)=64$. When $\#\left(\widehat{\mathbb{W}}_k\right)=P<64$, with n_i $(i=1\sim P)$ denoting the measure of the P different elements in $\widehat{\boldsymbol{W}}_k$,

[3] As discussed below, the breaking performance is rather good when $n\le 32$ (see Fig. 5), so one can simply set $n=32$ even when $n>32$.

one can easily deduce that there are $\prod_{i=1}^{P}(n_i!)$ possible estimations of $\boldsymbol{W}_k$ in total. Thus, the task of *Step 3* is to determine one estimated permutation matrix from all $\prod_{i=1}^{P}(n_i!)$ possible ones. Although many different methods can be used to realize *Step 3*, the following simple algorithm is enough in most cases to achieve an acceptable performance:

- Initialize all elements of an 8×8 flag matrix, $\boldsymbol{F}_k = [F_k(i,j)]_{i=0,j=0}^{7,7}$, to zeros.
- For $i = 0 \sim 7$ and $j = 0 \sim 7$, determine the value of $\widetilde{W}_k(i,j)$ as follows:
  1. find the first position (i',j') satisfying $M_k(i,j) = M'_k(i',j')$ and $F_k(i',j') = 0$;
  2. set $\widetilde{W}_k(i,j) = (i',j')$ and $F_k(i',j') = 1$.

Note that *Step 2* is also incorporated into the above algorithm, which is very useful in reducing the total complexity.

Next, let us see how many known plain-images are enough to achieve an acceptable breaking performance. Roughly, the larger the n is, the less the $\prod_{i=1}^{P}(n_i!)$, the more accurate the estimated permutation matrix $\widetilde{\boldsymbol{W}}_k$, and so the better the breaking performance will be. As a result, by estimating the mathematical expectation of n_i, one can conceptually derive a lower bound for n. To simplify the following analyses, let us assume that each element in $\boldsymbol{M}_{l,k}$ distributes uniformly over $\{0,1\}$ and any two elements are independent of each other. Then, one can see that there are two types of elements in each $\widehat{W}_k(i,j)$:

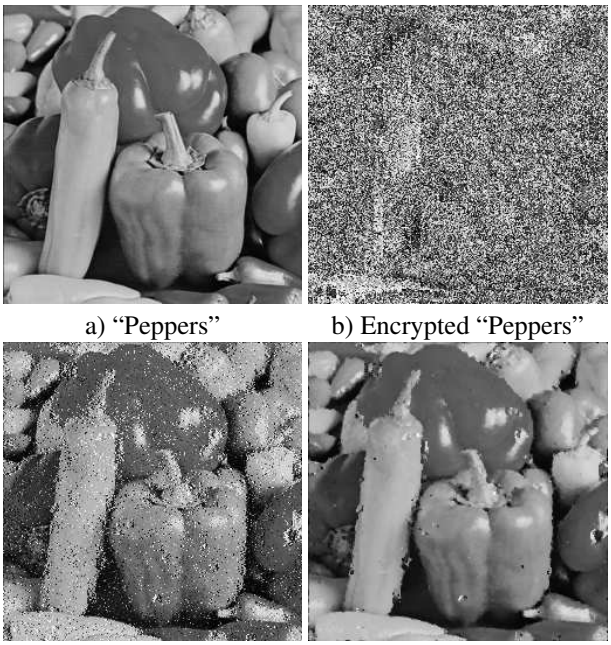
- *the only real position*, which absolutely occurs;
- *other fake positions*, each of which occurs in $\widehat{W}_k(i,j)$ with a probability of $1/2^n$, since any two bits in a bit matrix are identical with a probability of $1/2$.

Thus, it follows that the average cardinality of $\widehat{W}_k(i,j)$ is $\overline{n}_i = 1+(64-1)/2^n = 1+63/2^n$, which approaches 1 exponentially as n increases. Generally speaking, when $1+63/2^n < 1.5$, i.e., about half elements in $\widetilde{\boldsymbol{W}}_k$ are correct, the decryption performance will be acceptable[4]. Solving this inequality, one has

$$n \geq 1+\lceil \log_2 63 \rceil = 1+\lceil 5.9773 \rceil = 7.$$

This theoretical result has been verified by experiments as shown in Figs. 4 and 5. Note that the above analysis can also be derived from the general result given in [25]. Though the above result is deduced under the assumption that $\{\boldsymbol{M}_{l,k}\}$ is an i.i.d. sequence, it can be qualitatively generalized to other distributions of $\{\boldsymbol{M}_{l,k}\}$. Our experiments show that the above theoretical result essentially holds for most natural images.

For a randomly selected key, $(\alpha,\beta) = (2,2)$, $x(0) = 33578/2^{17} \approx 0.2562$, $\mu = 129518/2^{15} \approx 3.9526$, a set of known plain-images (all natural images) are randomly selected for testing. When $n = 8$, the plain-image "Peppers" (Fig. 4a) and its cipher-image (Fig. 4b) are used to verify the breaking performance based on $MN/8$ estimated permutation matrices, $\{\widetilde{\boldsymbol{W}}_k\}_{k=0}^{MN/8-1}$. The recovered plain-image is shown in Fig. 4c. It is found that almost all visual information contained in the original plain-image has been successfully recovered, though only $38012/65536 = 58\%$ of plain-pixels are correct in value. With some noise reduction algorithms, one can further enhance the recovered plain-image. One enhanced result with a 3×3 median filter is shown in Fig. 4d.



a) "Peppers" b) Encrypted "Peppers"

c) Recovered "Peppers" via $\{\widetilde{\boldsymbol{W}}_k\}_{k=0}^{MN/8-1}$ d) Enhanced "Peppers" by a 3×3 median filter

Figure 4: The image "Peppers" recovered by the first known-plaintext attack

Figure 5 shows the percentage of correctly-recovered plain-pixels with respect to n, the number of known plain-images. One can see that the breaking performance is good when $n \geq 8$. Also, it is found that the breaking performance of the natural image is better than the noisy image under the same condition, which is attributed to the correlation existing in the natural image for decryption as discussed in [25]. It can also be observed that the slope of the two lines in Fig. 5 are very flat when $n \geq 16$, this is also due to the correlation of the known-images (e.g., the MSBs of adjacent pixels are the same with a high probability).

The complexity of this attack is rather small. For each block, the time complexity consumed in *Step 1a* and *Step 1b* is $O(2\cdot 64\cdot(n-1))$, and the average complexity in *Step 2* is $O(64\cdot 32)$, so the total attack complexity is only $O((2\cdot 64\cdot(n-1)+64\cdot 32)\cdot MN/8) = O(16(n+15)MN)$.

This known-plaintext attack has two disadvantages: 1) the number of required known plain-images is somewhat large; 2) with n known plain-images of size $M\times N$, this attack can only decrypt cipher-images of size not greater than $M\times N$. In the following subsection, we will introduce another known-plaintext attack, by which we can get the secret keys with only one known plain-image (but with a larger complexity).

[4] It is an empirical result drawn from our experiments, which can be qualitatively explained by the fact that human eyes have a good capability of rejecting noises in natural images.

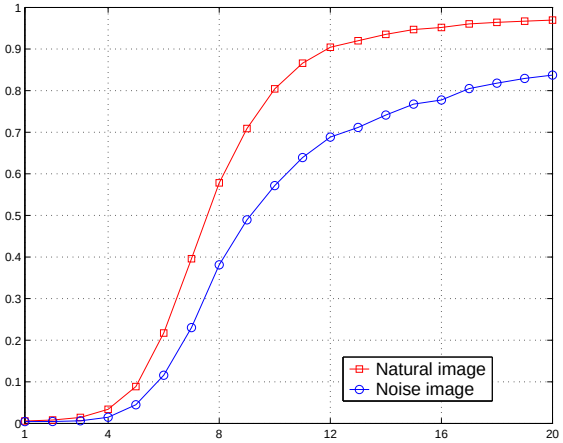


Figure 5: The percentage of correctly-recovered pixels with respect to the number of known plain-images

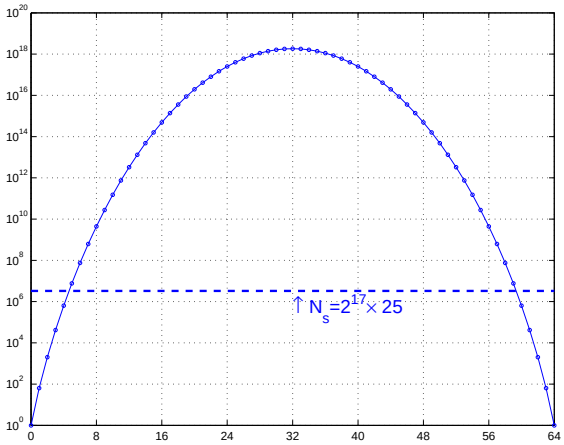


Figure 6: $C(t)=\binom{64}{t}=\frac{64!}{t!(64-t)!}$ with respect to t

4.2 Known-plaintext attack 2: Getting the secret key from one known plain-image

The known-plaintext attack introduced in this subsection is actually an optimized brute-force attack. By utilizing the correlation information existing between two consecutive chaotic states and the control parameter μ, the multiplicative search of the two sub-keys $x(0)$ and μ can be reduced to be the additive search of two chaotic states $x(k)$ and $x(k+1)$. This can dramatically reduce the attack complexity. Also, since each guessed chaotic state can be verified by a few number of 8-pixel blocks, not by the whole known plain-image, the attack complexity can be further reduced.

The basic idea of this attack is based on the following facts: 1) each permutation matrix $\boldsymbol{W}_k$ is uniquely determined by the current chaotic state $x(k)$ and the two sub-keys α, β; 2) two consecutive chaotic states $x(k)$ and $x(k+1)$ satisfy $x(k+1)\approx\mu\cdot x(k)\cdot(1-x(k))$. Once an attacker gets the right values of any two consecutive chaotic states, he can immediately get an estimation of μ, and then completely break TDCEA if α and β are also known.

To get the right value of a chaotic state $x(k)$ corresponding to the k-th bit matrix $\boldsymbol{M}_k$, one can use the permutation information existing in $\boldsymbol{M}_k$ and $\boldsymbol{M}'_k$. When there are t 0-bits and $(64-t)$ 1-bits in $\boldsymbol{M}_k$, one can calculate that the number of all possible values of $\boldsymbol{M}'_k$ is $C(t)=\binom{64}{t}=\frac{64!}{t!(64-t)!}$. In comparison, the number of all possibilities of each permutation matrix is equal to the number of all possible values of the 3-tuple data $(x(k),\alpha,\beta)$, which is less than $N_s=2^{17}\cdot 25$. When $5\le t\le 59$, one has $C(t)\gg N_s$ (see Fig. 6). This means that the probability that a wrong value of $(x(k),\alpha,\beta)$ coincides with $\boldsymbol{W}'_k$ is close to zero, i.e., one can exhaustively search all possible values of $(x(k),\alpha,\beta)$ to find a few number of candidates of the right value. Apparently, such an exhaustive searching procedure is optimized when $t=32$.

Carrying out the above procedure on two consecutive bit matrices, one can find some candidates of two consecutive chaotic states, $x(k)=0.b(17k+0)\cdots b(17k+16)$ and $x(k+1)=0.b(17k+18)\cdots b(17k+33)$. Then, an estimated value of the sub-key μ can be derived as

$$\widetilde{\mu}=\frac{x(k+1)}{x(k)\cdot(1-x(k))}. \tag{2}$$

Due to the quantization errors introduced in the finite-precision arithmetic, generally $x(k+1)\neq\mu\cdot x(k)\cdot(1-x(k))$, so $\widetilde{\mu}\neq\mu$. Fortunately, following the error analysis of $\widetilde{\mu}$ given in the Appendix of [22], it has been shown that when $x(k+1)\ge 2^{-n}$ $(n=1\sim 17)$, $|\widetilde{\mu}-\mu|<2^{n+3}\cdot 2^{-17}$. For example, when $x(k+1)\ge 2^{-1}=0.5$, one can exhaustively search $2^4=16$ values in the neighborhood of $\widetilde{\mu}$ to find the right value of μ. To verify whether $\widetilde{\mu}=\mu$, one can iterate the Logistic map from $x(k+1)$ until $x(MN/8-1)$ and then check the coincidence between each bit matrix $\boldsymbol{M}_i$ and $\boldsymbol{M}'_i$, $i=k+2,\cdots,MN/8-1$. Once a mismatch occurs, the current guessed value is discarded, and the next value is tested. To minimize the verification complexity, one can check only a number of chaotic states sufficiently far from $x(k+1)$ to eliminate most (if not all) wrong values of $\widetilde{\mu}$, and verify a few left ones by checking all chaotic states from $x(k+2)$ to $x(MN/8-1)$.

The proposed known-plaintext attack can be concretized step by step as follows.

- *Step 1*: Find the first two consecutive plain-blocks, $f^{(8)}(k)$ and $f^{(8)}(k+1)$, whose corresponding bit matrices $\boldsymbol{M}_k$ and $\boldsymbol{M}_{k+1}$ both have about 32 0-bits.

  Note: assuming that each bit in $\boldsymbol{M}_k$ distributes uniformly and independently, one can deduce that

  $$P_s=Prob\left[|t-32|\le s\right]=\frac{\sum_{i=32-s}^{32+s}\binom{64}{i}}{2^{64}}, \tag{3}$$

  where t is the number of nonzero elements of $\boldsymbol{M}_k$ and $0\le s\le 32$. When $s=4$, $P_s\approx 0.7396$, which is sufficiently large for an attacker to find valid plain-blocks within all the $MN/8$ blocks.

- *Step 2*: Exhaustively search all possible values of $(x(k),\alpha,\beta)$, and record those coinciding with $\boldsymbol{M}_k$ and $\boldsymbol{M}'_k$. Assume that m_1 candidates are recorded in total: $\{x_i(k),\alpha_i^*,\beta_i^*\}_{i=0}^{m_1-1}$.

- *Step 3*: Search all possible values of $x(k+1)$ and all values of (α,β) in $\{\alpha_i^*,\beta_i^*\}_{i=0}^{m_1-1}$, and record

those coinciding with $\boldsymbol{M}_{k+1}$ and $\boldsymbol{M}'_{k+1}$. Assume that m_2 candidates are recorded in total: $\{x_j(k+1), \alpha_j^{**}, \beta_j^{**}\}_{j=0}^{m_2-1}$.

- *Step 4*: For $i = 0 \sim m_1 - 1$ and $j = 0 \sim m_2 - 1$, do the following operations.
  - *Step 4a*: If $\alpha_i^* = \alpha_j^{**}$ and $\beta_i^* = \beta_j^{**}$, then calculate $\widetilde{\mu} = \dfrac{x_j(k+1)}{x_i(k) \cdot (1 - x_i(k))}$ and continue to execute *Step 4b*; otherwise, go to the next loop.
  - *Step 4b*: Assuming that $x_j(k+1) \geq 2^{-n}$, exhaustively search all possible 2^{n+3} values of μ within the neighborhood of $\widetilde{\mu}$. For each searched value, iterate the Logistic map from $x_i(k+1)$ to $x_i(MN/8-1)$. If every chaotic state $x_i(l)$ and (α_i^*, β_i^*) agree with $\boldsymbol{M}_l$ and $\boldsymbol{M}'_l$ $(l = k+2 \sim MN/8-1)$, then the attack completes.

The time complexity of this attack can be calculated as follows.

- The average complexity of *Step 2* is $2^{17} \cdot 25 \cdot (14 \cdot 8 + \frac{1}{2} \cdot 8 \cdot 8) < 2^{29}$.
- The complexity of *Step 3* is obviously less than that of *Step 2*.
- The average number of exhaustive searching loops in *Step 4* is $(m_1 \cdot m_2 \cdot C_x)$, where

$$C_x = \sum_{n=1}^{17} 2^{n+3} \cdot Prob\left[2^{-n} \leq x_j(k+1) < 2^{-(n-1)}\right],$$

  which is the mathematical expectation of the space size of the searching neighborhood of $\widetilde{\mu}$. Considering the computational complexity for each searching loop, the average complexity of *Step 4* is of order $\frac{m_1 \cdot m_2 \cdot C_x}{2} \cdot 49MN$. Without loss of generality, assume that $x_j(k+1)$ distributes uniformly over the interval [0,1], i.e., $Prob\left[2^{-n} \leq x_j(k+1) < 2^{-(n-1)}\right] = 2^{-n}$. Thus, $C_x = \sum_{n=1}^{17} 2^{n+3} \cdot 2^{-n} = 2^3 \cdot 17 = 136$. Then, the average complexity becomes $O(\frac{833 m_1 m_2 MN}{2})$. Since, in almost cases, $MN \leq 4096 \cdot 4096 = 2^{24}$ and m_1, m_2 are generally very small, the complexity is generally not greater than $O(2^{36})$.

Combining the above results, one concludes that the total complexity is $O(2^{36})$, which is practically small even for a PC and much smaller than $O(2^{60})$, the complexity of the simple brute-force attack shown in Sec. 3.3.

Figure 7 shows an experimental result of the recovered plain-image "Peppers", where the 5-th and 6-th pixel-blocks are chosen to exhaustively search the secret key. As a result, all chaotic states from $x(5)$ are successfully derived and only $(5 \cdot 8 = 40)$ leading plain-pixels at the left-bottom corner are not recovered correctly.

5 Chosen-Plaintext Attacks

Chosen-plaintext attacks are enhanced (and generally stronger) versions of known-plaintext attacks, with some intentionally chosen plaintexts and the corresponding ciphertexts [28]. In these attacks, the two known-plaintext attacks introduced in the previous section can be significantly enhanced.

Figure 7: The recovered plain-image "Peppers" by the second known-plaintext attack

5.1 Chosen-Plaintext Attack 1: Getting permutation matrices as an equivalent key

As discussed in Sec. 4.1, if $\#(\widetilde{\mathbb{M}}_k) = 64$, the permutation matrix $\boldsymbol{W}_k$ can be uniquely determined. Apparently, it is easy to ensure $\#(\widetilde{\mathbb{M}}_k) = 64$ by choosing the following six plain-images: $\forall\, k = 0 \sim MN/8-1, i = 0 \sim 7, j = 0 \sim 7$,

$$\begin{aligned}
f_0 &: M_{0,k}(i,j) = \lfloor (8i+j)/32 \rfloor \bmod 2;\\
f_1 &: M_{1,k}(i,j) = \lfloor (8i+j)/16 \rfloor \bmod 2;\\
f_2 &: M_{2,k}(i,j) = \lfloor (8i+j)/8 \rfloor \bmod 2;\\
f_3 &: M_{3,k}(i,j) = \lfloor (8i+j)/4 \rfloor \bmod 2;\\
f_4 &: M_{4,k}(i,j) = \lfloor (8i+j)/2 \rfloor \bmod 2;\\
f_5 &: M_{5,k}(i,j) = (8i+j) \bmod 2.
\end{aligned}$$

With the above six chosen plain-images, $\#(\widetilde{\mathbb{M}}_k) = 64$ holds so all $MN/8$ permutation matrices can be uniquely determined, which can then be used to decrypt any cipher-image of size not greater than MN.

The time complexity of such an attack is of the same order as the known-plaintext attack with $n = 6$ known plain-images, i.e., $O(16(6+15)MN) = O(336MN)$.

In fact, due to a special weakness of TDCEA, even two chosen plain-images are enough to completely reconstruct each 8×8 permutation matrix. Recalling the encryption procedure of TDCEA, one can see that 2-D secret rotations are merely a simple combination of 1-D rotations in two directions: 8 horizontal rotations followed by 8 vertical rotations. Such a property makes the division of the 2-D secret rotations possible in chosen-plaintext attacks with only two plain-images. In cryptanalysis, we call such attacks *divide-and-conquer* (DAC) attacks. The DAC chosen-plaintext attack can be described as follows.

- *Break the 8 vertical secret rotations*: Choose a plain-image f_0 as follows: $\forall\, k = 0 \sim MN/8-1$, $f_0^{(8)}(k) =$

$\{255,0,0,0,0,0,0,0\}$, i.e.,

$$\boldsymbol{M}_{0,k}=\begin{bmatrix}1&1&1&1&1&1&1&1\\0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\end{bmatrix}.$$

It is obvious that the 8 horizontal secret rotations have no influence on the above plain-image. That is, the 2-D TDCEA is reduced to the 1-D BRIE in the vertical direction. Since each column of $\boldsymbol{M}_{0,k}$ has only one 1-bit, by comparing $\boldsymbol{M}_{0,k}$ and $\boldsymbol{M}'_{0,k}$ one can uniquely get 8 values, $s_k(j)$ $(j=0\sim 7)$, which satisfy $\boldsymbol{M}'_{0,k}=\text{Rotate}\boldsymbol{Y}_j^{0,s_k(j)}(\boldsymbol{M}_{0,k})$ and serves as the equivalent rotation parameter of the j-th column.

- *Break the 8 horizontal secret rotations*: Choose a plain-image f_1 as follows: $\forall\ k=0\sim MN/8-1$, $f_1^{(8)}(k)=\{1,1,1,1,1,1,1,1\}$, i.e.,

$$\boldsymbol{M}_{1,k}=\begin{bmatrix}1&0&0&0&0&0&0&0\\1&0&0&0&0&0&0&0\\1&0&0&0&0&0&0&0\\1&0&0&0&0&0&0&0\\1&0&0&0&0&0&0&0\\1&0&0&0&0&0&0&0\\1&0&0&0&0&0&0&0\\1&0&0&0&0&0&0&0\end{bmatrix}.$$

Since the 8 vertical secret rotations have been obtained via f_0, one can remove all the 8 vertical rotations from $\boldsymbol{M}'_{1,k}$ to get the intermediate bit matrix $\boldsymbol{M}^*_{1,k}$. Then, by comparing $\boldsymbol{M}^*_{1,k}$ and $\boldsymbol{M}_{1,k}$, one can similarly get another 8 values, $r_k(i)$ $(i=0\sim 7)$, where $\boldsymbol{M}^*_{1,k}=\text{Rotate}\boldsymbol{X}_i^{0,r_k(i)}(\boldsymbol{M}_{1,k})$. Here, $r_k(0)\sim r_k(7)$ are the equivalent rotation parameter of the i-th line.

Apparently, after revealing the horizontal and vertical secret rotations, the permutation matrix $\boldsymbol{W}_k$ can be immediately reconstructed by simply combing the 16 rotations. In this case, the time complexity is only $O((4+1+4+8)MN)=O(17MN)$.

5.2 Chosen-plaintext attack 2: Getting the secret key

In the first chosen-plaintext attack, one can get 16 values, $s_k(0)\sim s_k(7)$ and $r_k(0)\sim r_k(7)$, for each pixel block $f^{(8)}(k)$. Based on the 16 values, the second known-plaintext attack discussed in Sec. 4.2 can be dramatically enhanced in most cases by introducing a much more effective way of deriving the 17 secret bits, $b(17k+0)\sim b(17k+16)$, of the chaotic state $x(k)$.

To simplify the following discussions, create a new vector, $rs_k(i)$ $(i=0\sim 15)$, which satisfies that $\forall\ i=0\sim 7$, $rs_k(i)=r_k(i)$ and $\forall\ i=8\sim 15$, $rs_k(i)=s_k(i-8)$.

Recalling the encryption procedure of TDCEA, it is obvious that the 16 values $\{rs_k(i)\}_{i=0}^{15}$ have a deterministic relation with the 17 secret bits $b(17k+0)\sim b(17k+16)$. Such a relation can be used to facilitate an exhaustive search of the 17 secret bits, i.e., the search of the k-th chaotic state $x(k)=0.b(17k+0)\cdots b(17k+16)$.

Considering the fact that $\text{Rotate}\boldsymbol{X}_i^{0,r}=\text{Rotate}\boldsymbol{X}_i^{1,8-r}$, $\text{Rotate}\boldsymbol{Y}_j^{0,s}=\text{Rotate}\boldsymbol{Y}_j^{1,8-s}$, one can see that $\forall\ i=0\sim 15$, $k=0\sim MN/8-1$, $rs_k(i)$ must be a value in the set $\mathbb{S}=\{\alpha,\alpha+\beta,8-\alpha,8-(\alpha+\beta)\}$.

For each guessed value $(\widetilde{\alpha},\widetilde{\beta})$, one can determine 16 bits, denoted by $\widetilde{b}(17k+1)\sim\widetilde{b}(17k+16)$, as estimations of $b(17k+1)\sim b(17k+16)$, as follows: $\forall\ i=1\sim 16$,

$$\widetilde{b}(17k+i)=\begin{cases}0, & rs_k(i-1)\in\{\widetilde{\alpha},8-\widetilde{\alpha}\},\\ 1, & rs_k(i-1)\in\{\widetilde{\alpha}+\widetilde{\beta},8-(\widetilde{\alpha}+\widetilde{\beta})\}.\end{cases}\tag{4}$$

Note that the above equation is invalid when $\widetilde{\alpha}=\widetilde{\alpha}+\widetilde{\beta}$ or $\widetilde{\alpha}=8-(\widetilde{\alpha}+\widetilde{\beta})$, i.e., $\widetilde{\beta}=0$ or $2\widetilde{\alpha}+\widetilde{\beta}=8$. Similarly, one has another equation for estimating the values of $b(17k+0)\sim b(17k+15)$: $\forall\ i=0\sim 15$,

$$\widetilde{b}(17k+i)=\begin{cases}0, & rs_k(i)\in\{\widetilde{\alpha},\widetilde{\alpha}+\widetilde{\beta}\},\\ 1, & rs_k(i)\in\{8-\widetilde{\alpha},8-(\widetilde{\alpha}+\widetilde{\beta})\}.\end{cases}\tag{5}$$

The above equation is invalid when $\widetilde{\alpha}=4$, $\widetilde{\alpha}+\widetilde{\beta}=4$ or $2\widetilde{\alpha}+\widetilde{\beta}=8$.

According to how much information that one can get from $\{rs_k(i)\}_{i=0}^{15}$, all values of $(\widetilde{\alpha},\widetilde{\beta})$ can be divided into the following classes in the chosen-plaintext attack.

- *C1)* $\widetilde{\alpha}\neq 4$, $\widetilde{\alpha}+\widetilde{\beta}\neq 4$, $\widetilde{\beta}\neq 0$ *and* $2\widetilde{\alpha}+\widetilde{\beta}\neq 8$: $\widetilde{b}(17k+1)\sim\widetilde{b}(17k+16)$ and $\widetilde{b}(17k+0)\sim\widetilde{b}(17k+15)$ can be uniquely determined by Eq. (4) and Eq. (5), respectively, so all the 17 bits, $\widetilde{b}(17k+0)\sim\widetilde{b}(17k+16)$, can be uniquely recovered.
  - There are 12 *C1*-values of $(\widetilde{\alpha},\widetilde{\beta})$, as follows: $(1,1)$, $(1,2)$, $(1,4)$, $(1,5)$, $(2,1)$, $(2,3)$, $(2,5)$, $(3,3)$, $(3,4)$, $(5,1)$, $(5,2)$, $(6,1)$.
- *C2)* $4\in\{\widetilde{\alpha},\widetilde{\alpha}+\widetilde{\beta}\}$ *and* $\widetilde{\beta}\neq 0$ *(which ensures* $2\widetilde{\alpha}+\widetilde{\beta}\neq 8$*)*: $\widetilde{b}(17k+1)\sim\widetilde{b}(17k+16)$ can be uniquely determined by Eq. (4), but $\widetilde{b}(17k+0)$ has to be guessed[5].
  - There are 6 *C2*-values of $(\widetilde{\alpha},\widetilde{\beta})$, as follows: $(1,3)$, $(2,2)$, $(3,1)$, $(4,1)$, $(4,2)$, $(4,3)$.
- *C3)* $\widetilde{\alpha}\neq 4$ *and* $\widetilde{\beta}=0$ *(which ensures* $2\widetilde{\alpha}+\widetilde{\beta}\neq 8$*)*: $\widetilde{b}(17k+0)\sim\widetilde{b}(17k+15)$ can be uniquely determined by Eq. (5), but $\widetilde{b}(17k+16)$ has to be guessed.
  - There are 6 *C3*-values of $(\widetilde{\alpha},\widetilde{\beta})$, as follows: $(1,0)$, $(2,0)$, $(3,0)$, $(5,0)$, $(6,0)$, $(7,0)$.

[5]Note that $\widetilde{b}(17k+0)$ can be **uniquely** determined in the following two sub-cases: a) when $\widetilde{\alpha}=4$ and $\widetilde{b}(17k+1)=1$, one can uniquely determine $\widetilde{b}(17k+0)$ by Eq. (5) since $\widetilde{\alpha}+\widetilde{\beta}\neq 4$; b) when $\widetilde{\alpha}\neq 4$ and $\widetilde{b}(17k+1)=0$, one can also uniquely determine $\widetilde{b}(17k+0)$ by Eq. (5). The two sub-cases occur with a probability of 0.5 when $\{b(i)\}$ distributes uniformly over $\{0,1\}$.

- *C4)* $2\widetilde{\alpha}+\widetilde{\beta}=8$: all the 17 bits has to be exhaustively guessed, as in the second known-plaintext attack discussed in Sec. 4.2.
  - There are 4 *C4*-values of $(\widetilde{\alpha},\widetilde{\beta})$, as follows: $(1,6)$, $(2,4)$, $(3,2)$, $(4,0)$.

The above four different cases correspond to different values of $\#(\mathbb{S})$ as follows:

- $\#(\mathbb{S})=4$: $(\widetilde{\alpha},\widetilde{\beta})$ is one of the 12 *C1*-values;
- $\#(\mathbb{S})=3$: $(\widetilde{\alpha},\widetilde{\beta})$ is one of the 6 *C2*-values;
- $\#(\mathbb{S})=2$: $(\widetilde{\alpha},\widetilde{\beta})$ is one of the 6 *C3*-values and the following *C4*-values: $\{(1,6),(2,4),(3,2)\}$;
- $\#(\mathbb{S})=1$: $(\widetilde{\alpha},\widetilde{\beta})=(4,0)$ (a *C4*-value).

Since one can guess the value of $\#(\mathbb{S})$ by observing the cardinality of the set $\{rs_k(0),\cdots,rs_k(15)\}\subseteq\mathbb{S}$, it is possible to search (α,β) in part of all possible values to reduce the attack complexity. Apparently, the success probability of such a guess is $P_e=Prob[\mathbb{S}=\{rs_k(0),\cdots,rs_k(15)\}]$. Since the theoretical deduction of P_e is rather difficult, experiments are performed to test all 2^{17} possible values of $b(17k+0)\sim b(17k+16)$. It results in that $P_e=122684/2^{17}\approx 0.936$, which is sufficiently large. Note that it is easy to further increase the success probability of the guess, by observing $n>1$ blocks at the same time. In doing so, the success probability will be *greater* than $P_e^{(n)}=1-(1-P_e)^n$ under the assumption that the chaotic bits for different blocks distribute uniformly and independently. As n increases, $P_e^{(n)}$ will approach 1 exponentially. In real attacks, even $n=2$ is enough in almost all cases, since $P_e^{(2)}\approx 0.996$. If all guessed values determined by $\#(\{rs_k(0),\cdots,rs_k(15)\})$ fail to pass the verification, it means that the rare event $\{rs_k(0),\cdots,rs_k(15)\}\subset\mathbb{S}$ occurs[6]. In this case, one has to continue to exhaustively search all other values of (α,β).

When the real value of (α,β) belongs to *C1*, *C2*, *C3* classes, the complexity of the chosen-plaintext attack will be much smaller than the complexity of its known-plaintext counterpart, due to the following reasons:

- the exhaustive searching procedure for the 17 bits of each chaotic state is simplified to be a deterministic calculation procedure dominated by Eqs. (4) and/or (5);
- the number of guessed values of (α,β) is reduced from 28 to 12 for *C1*, 6 for *C2* and *C3*;
- some values of (α,β) can be verified by checking whether or not $\{rs_k(0),\cdots,rs_k(15)\}\subseteq\{\widetilde{\alpha},\widetilde{\alpha}+\widetilde{\beta},8-\widetilde{\alpha},8-(\widetilde{\alpha}+\widetilde{\beta})\}$;
- one can intentionally choose the second chaotic state to ensure $x(k+1)\geq 0.5$, i.e., $b(17(k+1)+0)=1$, so as to reduce C_x, the average searching complexity of μ, from 136 to $2^{1+3}=16$.
- the exhaustive search of μ can be validated by just comparing the calculated chaotic state with the bits derived by Eqs. (4) and/or (5).

When the real value of (α,β) belongs to *C4* class, the average complexity of the chosen-plaintext attack is also smaller than the one of its known-plaintext counterpart, since the value of (α,β) can be immediately determined[7] with a sufficiently high probability, $P_e^{(n)}\approx 1$, that is, only when the rare event $\{rs_k(0),\cdots,rs_k(15)\}\subset\mathbb{S}$ occurs, one needs to exhaustively search the value of (α,β).

6 Chosen-Ciphertext Attacks

Chosen-ciphertext attacks are mirror versions of chosen-plaintext attacks, in which a cryptanalyst attempts to determine the secret key from knowledge of plaintexts that correspond to ciphertexts chosen by the attacker [28]. For TDCEA, due to the symmetry of the encryption and decryption procedures, one can carry out chosen-ciphertext attacks, in very much the same way as the chosen-plaintext attacks discussed in Sec. 5.

7 Conclusions

In this paper, the security of the recently-proposed encryption scheme for multimedia transmission, called TDCEA [1, 2], has been analyzed carefully. Some defects existing in TDCEA have been found and diagnosed. Two methods of known-plaintext attacks and their chosen-plaintext attack counterparts have been proposed to break the scheme. In addition, chosen-ciphertext attack has been mentioned briefly. Both theoretical and experimental analyses have been given to demonstrate the defects of TDCEA and to verify the feasibility of the proposed known-plaintext attacks. In conclusion, TDCEA is not suggested for applications that require a high level of security level.

8 Acknowledgement

This research was partially supported by the National Natural Science Foundation, China, under grants no. 60373041 and no. 60202002, and by the Applied R&D Centers of the City University of Hong Kong under grants no. 9410011 and no. 9620004.

References

[1] H.-C. Chen, J.-I. Guo, L.-C. Huang, and J.-C. Yen, "Design and realization of a new signal security system for multimedia data transmission," *EURASIP J. Applied Signal Processing*, vol. 2003, no. 13, pp. 1291–1305, 2003.

[2] J.-C. Yen and J.-I. Guo, "Design of a new signal security system," in *Proc. IEEE Int. Symposium on Circuits and Systems*, vol. 4, 2002, pp. 121–124.

[6] Note that the occurrence probability is not zero, though it is very close to zero when n is sufficiently large.

[7] If $\#(\mathbb{S})=1$, then $(\alpha,\beta)\equiv(4,0)$; otherwise, one can determine the value of (α,β) quickly by checking the following three candidates: $(1,6),(2,4),(3,2)$.

[3] ——, "A new image encryption algorithm and its VLSI architecture," in *Proc. IEEE Workshop on Signal Processing Systems*, 1999, pp. 430–437.

[4] ——, "A new MPEG/encryption system and its VLSI architecture," in *Proc. Int. Symposium on Communications*, 1999, pp. 215–219.

[5] K.-L. Chung and L.-C. Chang, "Large encrypting binary images with higher security," *Pattern Recognition Letters*, vol. 19, no. 5-6, pp. 461–468, 1998.

[6] N. Bourbakis and C. Alexopoulos, "Picture data encryption using scan patterns," *Pattern Recognition*, vol. 25, no. 6, pp. 567–581, 1992.

[7] Alexopoulos, N. Bourbakis, and N. Ioannou, "Image encryption method using a class of fractals," *J. Electronic Imaging*, vol. 4, no. 3, pp. 251–259, 1995.

[8] L. Tang, "Methods for encrypting and decrypting MPEG video data efficiently," in *Proc. 4th ACM Int. Conference on Multimedia*, 1996, pp. 219–229.

[9] H. C. H. Cheng, "Partial encryption for image and video communication." Master's thesis, Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada, Fall 1998.

[10] L. Qiao, "Multimedia security and copyright protection," Ph.D. dissertation, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, 1998.

[11] S. U. Shin, K. S. Sim, and K. H. Rhee, "A secrecy scheme for MPEG video data using the joint of compression and encryption," in *Information Security: Second Int. Workshop (ISW'99) Proc.*, ser. Lecture Notes in Computer Science, vol. 1729, 1999, pp. 191–201.

[12] J.-C. Yen and J.-I. Guo, "Efficient hierarchical chaotic image encryption algorithm and its VLSI realisation," *IEE Proc. – Vis. Image Signal Process.*, vol. 147, no. 2, pp. 167–175, 2000.

[13] W. Zeng and S. Lei, "Efficient frequency domain selective scrambling of digital video," *IEEE Trans. Multimedia*, vol. 5, no. 1, pp. 118–129, 2003.

[14] B. Bhargava, C. Shi, and S.-Y. Wang, "MPEG video encryption algorithms," *Multimedia Tools and Applications*, vol. 24, no. 1, pp. 57–79, 2004.

[15] J.-K. Jan and Y.-M. Tseng, "On the security of image encryption method," *Information Processing Letters*, vol. 60, no. 5, pp. 261–265, 1996.

[16] L. Qiao, K. Nahrstedt, and M.-C. Tam, "Is MPEG encryption by using random list instead of ZigZag order secure?" in *Proc. IEEE Int. Symposium on Consumer Electronics (ISCE'97)*, 1997, pp. 226–229.

[17] L. Qiao and K. Nahrsted, "Comparison of MPEG encryption algorithms," *Computers & Graphics*, vol. 22, no. 4, pp. 437–448, 1998.

[18] T. Uehara and R. Safavi-Naini, "Chosen DCT coefficients attack on MPEG encryption schemes," in *Proc. IEEE Pacific-Rim Conference on Multimedia (IEEE-PCM'2000)*, 2000, pp. 316–319.

[19] S. Li and X. Zheng, "On the security of an image encryption method," in *Proc. IEEE Int. Conference on Image Processing*, vol. 2, 2002, pp. 925–928, a refined preprint is available at http://www.hooklee.com/pub.html.

[20] ——, "Cryptanalysis of a chaotic image encryption method," in *Proc. IEEE Int. Symposium on Circuits and Systems*, vol. II, 2002, pp. 708–711, a refined preprint is available at http://www.hooklee.com/pub.html.

[21] C.-C. Chang and T.-X. Yu, "Cryptanalysis of an encryption scheme for binary images," *Pattern Recognition Letters*, vol. 23, no. 14, pp. 1847–1852, 2002.

[22] C. Li, S. Li, D. Zhang, and G. Chen, "Cryptanalysis of a chaotic neural network based multimedia encryption scheme," in *Advances in Multimedia Information Processing - PCM 2004 Proceedings, Part III*, ser. Lecture Notes in Computer Science, vol. 3333. Springer-Verlag, 2004, pp. 418–425.

[23] X.-Y. Zhao, G. Chen, D. Zhang, X.-H. Wang, and G.-C. Dong, "Decryption of pure-position permutation algorithms," *Journal of Zhejiang University SCIENCE*, vol. 5, no. 7, pp. 803–809, 2004.

[24] S. Li, C. Li, G. Chen, and X. Mou, "Cryptanalysis of the RCES/RSES image encryption scheme," Cryptology ePrint Archive: Report 2004/376, available online at http://eprint.iacr.org/2004/376, 2004.

[25] S. Li, C. Li, G. Chen, D. Zhang, and N. G. Bourbakis, "A general cryptanalysis of permutation-only multimedia encryption algorithms," Cryptology ePrint Archive: Report 2004/374, available online at http://eprint.iacr.org/2004/374, 2004.

[26] S. Li, G. Chen, and X. Zheng, "Chaos-based encryption for digital images and videos," in *Multimedia Security Handbook*, B. Furht and D. Kirovski, Eds. CRC Press, LLC, 2004, ch. 4, with preprint available at http://www.hooklee.com/pub.html.

[27] Hao Bai-Lin, *Starting with Parabolas: An Introduction to Chaotic Dynamics*. Shanghai, China: Shanghai Scientific and Technological Education Publishing House, 1993, (In Chinese).

[28] B. Schneier, *Applied Cryptography – Protocols, Algorithms, and Souce Code in C*, 2nd ed. New York: John Wiley & Sons, Inc., 1996.