Optik 124 (2013) 5411–5415

Contents lists available at ScienceDirect

Optik

journal homepage: www.elsevier.de/ijleo

Cryptanalysis of a color image encryption algorithm based on chaos

Guangyou Tu [a,b,*], Xiaofeng Liao [b], Tao Xiang [b]

[a] Center of Information and Network, Chongqing University, Chongqing 400044, China
[b] College of Computer Science, Chongqing University, Chongqing 400044, China

ARTICLE INFO

Article history:
Received 27 October 2012
Accepted 22 March 2013

Keywords:
Image encryption
Logistic map
Chosen-plaintext attack
Cryptanalysis

ABSTRACT

A novel image encryption algorithm based on logistic map is proposed recently. In this paper, a chosen plaintext attack on this algorithm is presented and some other flaws of the algorithm are pointed out. Theoretical analysis and experimental simulation indicate that the plain image can be recovered exactly from the cipher image without secret key. Therefore, this algorithm is not secure enough for practical applications. An improvement is proposed to enhance the security of the original algorithm. Simulation results and theoretical analysis show that the improved scheme has expected cryptographic properties and is more secure than the original algorithm.

1. Introduction

Chaotic system is characterized by sensitive dependence on initial conditions and control parameters, pseudo-randomness and ergodicity, which meet the classic Shannon requirements of confusion and diffusion. The use of chaotic systems for secure communications has been an active area of research in the past few years.

Recently, many chaos-based image encryption algorithms have been developed [1–11]. Some of them have been cryptanalyzed [12–14] and have been found to be not secure. A novel color image encryption algorithm based on logistic map is proposed by Wang and Teng [15] recently. The cryptosystem uses the logistic chaotic system to generate shuffling sequences, permute and diffusion the RGB component of color image at the same time and make the three components affect each other. In this paper, we analyzed Wang and Teng's algorithm, present a chosen plaintext attack on this algorithm and give an improving method.

The rest of this paper is organized as follows. In the next section, an image encryption algorithm proposed in Ref. [15] is introduced. In Section 3, the cryptanalysis and the chosen plaintext attack of the algorithm under study are described. Simulations and experimental results are given in this section too. Section 4 analyzes some other flaws in the original algorithm, gives an improving method and evaluates the security of the method. The last section concludes the paper.

2. Review of the cryptosystem

The cryptosystem proposed in Ref. [15] permutes and diffuses the plaintext image pixels by logistic map. First, it converts the $M \times N$ true-color plain-image P into its RGB components; each color's (R, G, B) matrix is $M \times N$, and the pixels' values range from 0 to 255. The logistic map is described as

$$x_{n+1} = ax_n(1 - x_n)x_n \in (0, 1) \quad (1)$$

The system is chaotic when $a \in (3.5699456,4)$.

The encryption scheme is composed of two processes: pixels permutation and pixels diffusion. Firstly, the permutation algorithm is described as follows:

Step 1 Reorganize the RGB matrixes of plain-image P and get matrix P_1 with $3M$ rows and N columns. Select initial parameter a and x_0 to iterate the logistic map $m + 3M$ times. Sort the last $3M$ values according to the sort rule, and rearrange the rows of P_1 to transformed matrix P'_1.

Step 2 Partition P'_1 into three $M \times N$ matrixes from the top to bottom, and combine them horizontally and get matrix P_2 with M rows and $3N$ columns. Select another initial parameter a_1 and y_0, and iterate the logistic map n+$3MN$ times. Sort the last $3MN$ values according the sort rule, and rearrange the columns of P_2 and get the transformed matrix P'_2. The shuffle rule is similar to that of Step 1.

Secondly, the pixels diffusion procedure is shown as below:

* Corresponding author at: No. 174 Shazhengjie, Shapingba District, Chongqing 400044, China.
E-mail address: lefour@cqu.edu.cn (G. Tu).

0030-4026/$ – see front matter © 2013 Elsevier GmbH. All rights reserved.
http://dx.doi.org/10.1016/j.ijleo.2013.03.113

Step 1: Use the last 3MN values of the Step 2 in the permutation algorithm y_n, and compute the sequences:

$$z1_n = \mathrm{mod}(y_n \times 10^{14}, 3) \quad (n = 1, 2, \ldots 3MN) \tag{2}$$

$$z2_n = \mathrm{mod}(y^n \times 10^{14}, 256) \quad (n = 1, 2, \ldots 3MN) \tag{3}$$

Step 2: Partition the P'_2 into three matrixes R_2, G_2, B_2 from left to right and convert them into vector which has the length of $L = M \times N$. $z1_n$ decides what vector will be diffused, "0" for R vector, "1" for G vector, and "2" for B vector. $z2_n$ is used to diffuse the proper pixel values. The diffusion formula is

$$C_{\mathrm{now}} = (P_{\mathrm{now}} + z2_{\mathrm{n}} + C_{\mathrm{pre}} + P_{\mathrm{pre}})\mathrm{mod}256 \ (P_0 = C_0 = 0) \tag{4}$$

3. The proposed cryptanalysis

According to Kerchoff's principle [16], there are four types of attack: cipher text only attack, known plaintext attack, chosen plaintext attack, and chosen cipher text attack. If an encryption algorithm is not able to resist anyone attack mentioned above, it is considered to be unsecure.

In the original proposal, there are 4 parameters used as secret key. The key space can reach to 10^{56}, it is greater than 2^{128}. Obviously, such a key space can resist brute-force attack. However, as we will show, attacker can reveal $z1_n$, $z2_n$ and the shuffle rule instead of reveal the secret key. This is very easy compared to attacking the secret initial conditions.

3.1. Chosen plaintext attack on the diffusion rule

Since the security of the cryptosystem relies on the position changing rule of pixels and pixels diffusion. So we want to reveal the diffusion rule first. In this cryptosystem, as the diffusion formula is mentioned above, $z1_n$ and $z2_n$ are the key factors of the diffusion. If we know $z1_n$ and $z2_n$, we can decrypt the shuffled pixels' values easily. According to the diffusion formula (4), we can obtain that:

$$z2_n = (C_{\mathrm{now}} + 768 - P_{\mathrm{now}} - C_{\mathrm{pre}} - P_{\mathrm{pre}})\mathrm{mod}256 \quad (P_0 = C_0 = 0) \tag{5}$$

768 is added to ensure the module is positive number.

Step 1 Construct a true color image P with size $M \times N$, whose three base color pixels matrix R, G and B are composed of all zero. After permutation procedure and diffusion, we get three ciphered vectors R^c, G^c and B^c.

Step 2 Construct another true color image P1 with size $M \times N$, whose three base color pixels matrix R1, G1 and B1 are composed of all ones. After permutation procedure and diffusion, we get three ciphered vectors $R1^c$, $G1^c$ and $B1^c$.

Step 3 Let $i = 1$, $P_0 = C_0 = 0$ and $P1_0 = C1_0 = 0$, the decrypt flag of three vectors equal to "1".

Step 4 Let $Z1_i = 0$, $P_{\mathrm{now}} = 0$, *match_num* = 0, C_{now} equal to R^c's unit value. According to the formula (5) we can count the value of $z2_i$, we define it as $z2_i\_0\_p$.

Let $P_{\mathrm{now}} = 1$, C_{now} equal to $R1^c$'s unit value. According to the formula (5) we can count another value of $z2_i$, we define it as $z2_i\_0\_p1$.

If $z2_i\_0\_p = z2_i\_0\_p1$, then *match_num* = *match_num* + 1.

Step 5 Repeat the similar operation to Step 4 respect to remained two possible values of $z1_i$. If $z1_i = 1$, C_{now} equal to the G cipher vector's unit value, and if $z1_i = 2$, C_{now} equal to the B cipher vector's unit value.

Step 6 If *match_nu* = 1, that unique value that satisfied $z2i_{-*}\_p = z2i_{-*}\_p_1$ is the actual value of $z2_i$, and the corresponding value of $z1_i$ is the actual value of $z1_i$ too. Mark down the vector and its pixel value. Move the decrypt flag to the next unit of the vector, other vector's decrypt flag remain unchanged.

Step 7 Let $P_{\mathrm{pre}} = 0$, $P1_{\mathrm{pre}} = 1$, C_{pre} of the R^c, G^c, B^c and $R1^c$, $G1^c$, $B1^c$ equal the value of the corresponding pixels of the Step 6 respectively. Repeat Step 4 and Step 6 until all the pixels have been processed. We can obtain all the values of $z1_n$ and $z2_n$.

If *match_num* > 1, we must input another plain-image to reveal remain values of $z1_n$ and $z2_n$.

3.2. Chosen plaintext attack on shuffling rule

Since the cryptosystem use the same secret key encrypt the plain-image, the permutation algorithm and the pixels diffusion has no relationship with the change of plain-image. So we can choose two plain images having only a pixel difference to encrypt, by compare the cipher images, we can find some rule of the permutation algorithm. But this method is inaccurate and slow.

As we have revealed $z1_n$ and $z2_n$ in Section 3.1, we can obtain the shuffled matrixes R_2, G_2, B_2 of any plain image by the following formula.

$$P_{\mathrm{now}} = (C_{\mathrm{now}} + 768 - z2_n - C_{\mathrm{pre}} - P_{\mathrm{pre}})\mathrm{mod}256 \quad (P_0 = C_0 = 0) \tag{6}$$

We can find the permutation rule by comparing the plain image's matrix with the shuffled matrix. For a normal 256×256 true color image, it has 196608 pixels in image matrix and only 256 possible values. Almost all the pixels' values are duplicated many times. It is impossible to obtain the position changing rule by comparing one normal plain image with the transformed one. So we must choose some special images to make it.

In this section, we treat the plain image and the cipher image as vectors. First, we convert the $M \times N$ true color plain image into three vectors for each base color, then we combine the three vectors into one vector V which has a length of $L = 3M \times N$orderly.

Step 1 Construct a true color image of size $M \times N$, whose first 255 elements are 255, 254, 253, ... 1 and other elements are zero. After permutation procedure and diffusion, we get three ciphered vectors R^c, G^c, B^c.

Step 2 By using (6) and the values of $z1_n$ and $z2_n$ revealed in Section 3.1, we can obtain the undiffused three base color vectors. Combine the three vectors into one vector V_c orderly.

Step 3 Sort V_c in descending order, mark down the position changing rule, the first 255th elements' position is corresponding the V's first 225th element's position change rule.

Step 4 Repeat Step1 to Step 3 by move back the position of unequal zero pixels orderly until all the pixels' position changing rule have been revealed.

3.3. Experimental results

In our practice, we select same keys as those in Ref. [15]. Before getting the 3*MN* values of column scrambling, we iterate the chaotic system 500 times to avoid harmful effects. By inputting two plain images with all "0" and "1" values in pixels. We reveal all the values of $z1_n$ and $z2_n$ as follow:

$z1_n = \{2, 1, 0, 0, 2, 2, 0, 0, 2, 1, \ldots\}$

$z2_n = \{122, 195, 226, 184, 189, 72, 184, 255, 119, 142, \ldots\}$

The values are same with the compute result.