Efficient chaotic permutations for image encryption algorithms

Abir Awad

Laboratoire de cryptologie et de virologie operationnelles (C +V)^O, esiea, IUT, Laval

awad@esiea-ouest.fr

ABSTRACT

Permutation is widely used in cryptographic algorithm. Recently, a number of candidate instructions have been proposed to efficient compute arbitrary bit permutations. Among these, we present the most attractive methods and having good inherent cryptographic properties. We propose to control it by the perturbed chaotic maps that we studied in [1]. Then, we measure the efficiency of the obtained chaotic permutation methods on a standard image. This study allows choosing a good chaotic permutation method to be used in a chaotic cryptosystem.

1. INTRODUCTION

Recently, chaos has been widely studied in secure communications, and the idea of using digital chaotic systems to construct cryptosystems has been extensively studied since 1990s, and attracts more and more attention in the last years [2, 3]. Chaotic output signal of one dimensional chaotic generator is used for both confusion and diffusion operations in a cryptosystem.
Diffusion spreads the redundant information in the plain text over the cipher text. As a primary method to achieve diffusion, permutation is widely used in cryptographic algorithms. In fact, bit level permutations are particularly the core of any encryption algorithm. Recently, a number of candidate instructions as CROSS method have been proposed for efficient software implementation of arbitrary permutations [4, 5, 6].
Their permutation techniques can be implemented in any programmable processor, whether they are general purpose microprocessor or application-specific cryptography processors.
In another hand, chaotic permutation methods as Socek one were also proposed [7].
In this paper, we propose to control theses methods with perturbed chaotic generators. In order to be used in all applications, chaotic sequences must seem absolutely random. In [1], we study and improve some existing techniques used to generate chaotic signals with desired statistical properties and verifying NIST statistical tests. The perturbing orbit technique used improves the dynamical statistical properties of generated chaotic sequences.
In [8], we proposed this generator to control a cryptographic algorithm but we didn't study the performance of the chaotic permutation method used.
In this paper, we proposed the utilization of the perturbed PWLCM chaotic map to control CROSS method and Socek permutation method. Then, we compare the results obtained

with the two methods controlled by perturbed chaotic PWLCM map and non-perturbed one.
Indeed, it is well known that images are different from texts in many aspects, such as high redundancy and correlation. The main obstacle in designing effective image encryption algorithms is that it is rather difficult to diffuse such image data. In most of the natural images, the value of any given pixel can be reasonably predicted from the values of its neighbors. For that, we choose images like applications to test our propositions.
We implement the two proposed chaotic permutation methods CROSS and Socek using Matlab. Then, we permute the 512x512x3 Mandrill image with these methods. We present the obtained results. And we demonstrate the obtained gain, specifically in the entropy value, using perturbed PWLCM chaotic map instead of simple PWLCM that Socek used in his method.
This paper is organized as follows. Section 2 describes the perturbed PWLCM chaotic map. Section 3 explains the proposed permutation techniques. The simulation results are presented in section 4. And finally, we summarize our conclusions in section 5.

2. PERTURBED PWLCM MAP

A piecewise linear chaotic map (PWLCM) is a map composed of multiple linear segments.

$$x(n) = F[x(n-1)] = \begin{cases} x(n-1)x\dfrac{1}{p} & if\ 0 \le x(n-1) < p \\ [x(n-1)-p]x\dfrac{1}{0.5-p} & if\ p \le x(n-1) < 0.5 \\ F[1-x(n-1)] & if\ 0.5 \le x(n-1) < 1 \end{cases} \quad (1)$$

where the positive control parameter p ϵ (0; 0.5) and x(i) ϵ (0; 1). Since digital chaotic iterations are constrained in a discrete space with 2^N elements, it is obvious that every chaotic orbit will eventually be periodic and will finally go to a cycle with limited length not greater than 2^N. Generally, each digital chaotic orbit includes two connected parts: $x_1, x_2, ..., x_{l,}$ and $x_l, x_{l+1}, ..., x_{l+n,}$, which are respectively called "transient branch" and "cycle". Accordingly, l and $n+1$ are respectively called "transient length" and "cycle period", and $l+n$ is called "orbit length".

To improve the dynamical degradation, a perturbation based algorithm is used [9]. The cycle length is expanded and so good statistical properties are reached.

Here, for computing precision N, each x can be described as:

$$x(n) = 0.x_1(n)x_2(n)...x_i(n)...x_N(n) \qquad x_i(n) \in \{0,1\},\ i = 1,2,...,N \quad (2)$$

A suitable candidate for the perturbing signal generator is the maximal length LFSR because its generated sequences have the following advantages: 1) definite cycle length $(2^k - 1)$ (k is the degree); 2) uniform distribution; 3) delta like autocorrelation function; 4) easy implementation; 5) controllable maximum signal magnitude given by $2^{-N} \times (2^k - 1)$ when used in N-precision system.

The perturbing bit sequence can be generated every n clock as follows:

$$Q_{k-1}^{+}(n) = Q_k(n) = g_0 Q_0(n) \oplus g_1 Q_1(n) \oplus ... \oplus g_{k-1} Q_{k-1}(n) \quad with\ n = 0,1,2,... \quad (3)$$

Where $\oplus$ represents 'exclusive or', $g = [g_0\ g_1 ... g_{k-1}]$ is the tap sequence of the primitive polynomial generator, and $Q_0, Q_1, ..., Q_{k-1}$ are the initial register values of which at least one is non zero.

The perturbation begins at $n = 0$, and the next ones occur periodically every Δ iterations (Δ is a positive integer), with $n = l \times \Delta$, $l = 1,2,...$, The perturbed sequence is given by the equation (4):

$$x_i(n) = \begin{cases} F[x_i(n-1)] & 1 \le i \le N-k \\ F[x_i(n-1)] \oplus Q_{N-i}(n) & N-k+1 \le i \le N \end{cases} \quad (4)$$

Where $F[x_i(n)]$ represents the ith bit of $F[x(n)]$.

The perturbation is applied on the last k bits of $F[x(n)]$.

When $n \neq l \times \Delta$, no perturbation occurs, so $x(n) = F[x(n-1)]$. The system cycle length is given by the following relation

$$T = \sigma \times \Delta \times \left(2^k - 1\right) \quad (5)$$

where σ is a positive integer. The lower bound of the system cycle length is

$$T_{\min} = \Delta \times \left(2^k - 1\right) \quad (6)$$

3. CHAOTIC PERMUTATION METHODS

We used a perturbed PWLCM chaotic map to control the chosen permutation techniques, CROSS and Socek ones. The chaotic values *x* are real. Then, a discretization method is applied to transform it to unsigned integer on 32 bits using the following formulas:

$$y = round(x.2^{32}) \tag{7}$$

where *x* is a chaotic real value and *y* is the discretized one. The last one is used to control the chosen permutation methods that we will explain in the following paragraph.

3.1. Cross permutation

CROSS instruction is defined as follows:

R3=CROSS(m1, m2, R1, R2)

R1 is the source register which contains the bits to be permuted, *R2* is the configuration register and R3 is the destination register for the permuted bits. One CROSS instruction performs two basic operations on the source according to the contents of the configuration register and the values of *m1* and *m2*. We propose to control this method by chaotic values. Then, in each iteration, the control register *R2* is filled by the chaotic binary suite (8 bits). We said that the chaotic value is on 32 bits. Then, the chaotic value (32 bits) is generated each 4 iterations. The chaotic suite is decomposed on 4 parts. Each chaotic byte is utilized to control a byte from the image. Fig. 1 shows how the CROSS instruction works on 8-bit systems.

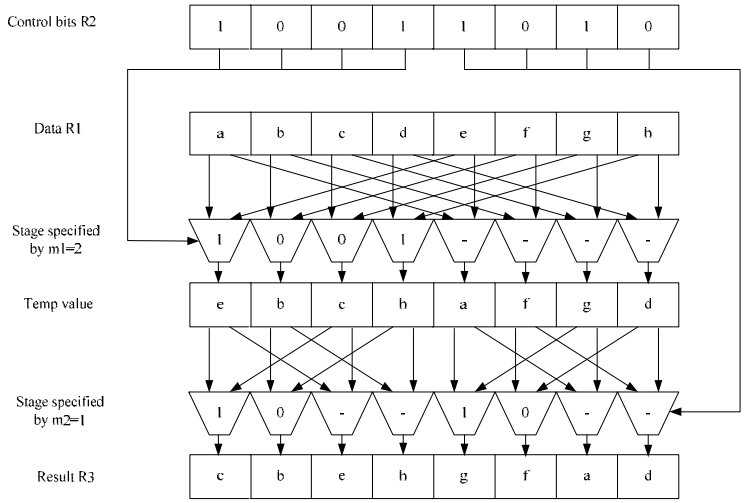


Figure 1. The CROSS permutation method performed on 8-bit

The inverse CROSS instruction is the same as that used for the encryption process but the contents of the configuration register *m1* and *m2* are exchanged.

3.2. Socek permutation

The permutation method proposed by Socek is to permute the indices of bits *p= [1, 2, 3, 4, 5, 6, 7, 8]* of each pixel using the chaotic value. Then the bits are rearranged according to the new array indices *q*. Fig. 2 presents the algorithm of Socek method applied on a 8 bits.

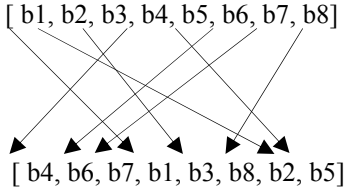


Figure 2. Socek method on 8 bits

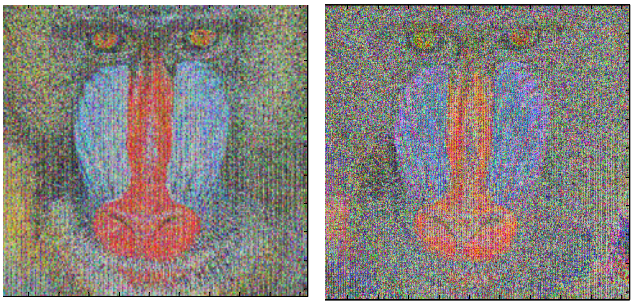
In this example, the new array of indices q= [4, 6, 7, 1, 3, 8, 2, 5].
To perform the inverse of Socek method, the bits are rearranged according to the array indices *(8-q(i))* instead of *q(i)*.

4. SIMULATION RESULTS

Some experimental results are given in this section to demonstrate the efficiency of proposed chaotic permutation methods. We implemented in *Matlab* the Cross and Socek methods. The plain image used is 'MANDRILL.BMP' with the size 512x512x3 (Fig. 3).

Figure 3. Mandrill image

Fig. 4 presents the permuted images.



(a) (b)

Figure 4. Permuted image of Mandrill with (a) CROSS and (b) Socek method controlled by perturbed PWLCM map.

As we can see, the bit permutation method doesn't hide the texture of the image. Then, a substitution method is necessary to construct a chaotic cryptosystem.

4.1 Difference between the original and the encrypted image

Common measures like *NPCR* (Number of pixels change rate) and *UACI* (Unified Average Changing Intensity) are used to test the difference between the original image *P1* and the permuted one *C1*.
NPCR stands for the number of pixel change rate. Then, if *D* is a matrix with the same size as images *P1* and *C1*, *D (i,j)* is determined as follows (8):

$$D(i,j)=\begin{cases}1 & if\ \ P_1(i,j)\neq C_1(i,j)\\ 0 & else\end{cases} \tag{8}$$

NPCR is defined by the following formula (9):

$$NPCR=\frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}D(i,j)}{M\times N}\times 100 \tag{9}$$

where *M* and *N* are the width and height of P_1 and C_1.

The *UACI* measures the average intensity of differences between the plain image and the permuted one.
UACI is defined by the following formula (10):

$$UACI=\frac{1}{MxN}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\frac{\left|P_1(i,j)-C_1(i,j)\right|}{255}x100 \tag{10}$$

The optimal values of these parameters are respectively: $NPCR_{opt}$= 99.61% and $UACI_{opt}$=33.46%.

In table I, we summarize the mean values of NPCR and UACI obtained between the original image and the permuted one.

Table 1. NPCR and UACI between the original image and the permuted one.

	Cross	Socek
NPCR	80.9926	98.5208
UACI	20.0257	27.1395

As, we can see, Socek method gives better results than CROSS method.

4.2. Correlation of adjacent pixels in the permuted image

Statistical analysis on large amounts of images shows that averagely adjacent 8 to 16 pixels are correlative. To test the correlation between horizontally, vertically and diagonally adjacent pixels from the image, we calculate the correlation coefficient of a sequence of adjacent pixels by using the formulas (18), (19) and (20), (21).
We calculate the correlation coefficient *r* of original and encrypted image by using the following formulas (11), (12) and (13), (14):

$$E(x) = \frac{1}{MxN}\sum_{i=1}^{M}\sum_{j=1}^{N} P_1(i,j) \tag{11}$$

$$D(P_1) = \frac{1}{MxN}\sum_{i=1}^{M}\sum_{j=1}^{N} [P_1(i,j) - E(P_1(i,j))]^2 \tag{12}$$

$$\mathrm{cov}(P_1,C_1) = \frac{1}{MxN}\sum_{i=1}^{M}\sum_{j=1}^{N} [P_1(i,j) - E(P_1(i,j))][C_1(i,j) - E(C_1(i,j))] \tag{13}$$

$$r_{P_1C_1} = \frac{\mathrm{cov}(P_1,C_1)}{\sqrt{D(P_1)}\sqrt{D(C_1)}} \tag{14}$$

where $P_1(i,j)$ and $C_1(i,j)$ are gray values of the original pixel and the encrypted one.

Fig. 5 and 6 show the correlation distributions of horizontally adjacent pixels in the original and permuted images. And, table II gives the coefficients correlation of horizontal, vertical and diagonal pixels.

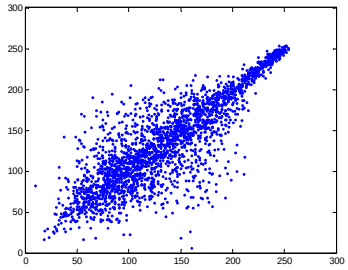
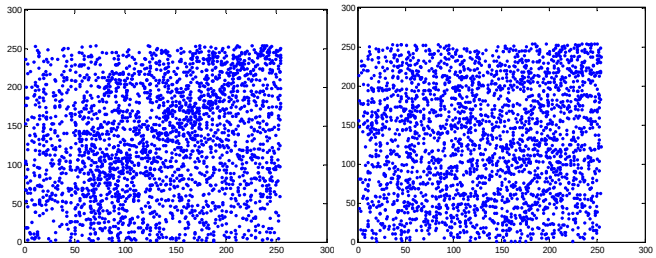


Figure 5. Correlation distributions of horizontally adjacent pixels in Mandrill image



(a) (b)

Figure 6. Correlation distributions of horizontally adjacent pixels in the permuted images using the proposed chaotic (a) CROSS and (b) Socek methods.

Table 2. Mean values of the coefficients correlation of horizontal, vertical and diagonal pixels.

Model	Mandrill image	Cross	Socek
Horizontal	0.9203	0.1938	0.1132
Vertical	0.8631	0.1644	0.1276
Diagonal	0.8494	0.1703	0.0993

4.3. Information entropy analysis

Entropy is a statistical measure of randomness that can be used to characterize the texture of an image. It is well known that the entropy *H(m)* of a message source m can be calculated as :

$$H(m) = \sum_{i=0}^{2^N-1} p(m_i) \log_2 \frac{1}{p(m_i)} \tag{15}$$

Where $p(m_i)$ represents the probability of message m_i.
When an image is encrypted, its entropy should ideally be 8. If it is less than this value, there exists a certain degree of predictability which threatens its security.
In table 3, we list the entropy of the images permuted by the proposed chaotic permutation methods. The values obtained are very close to the theoretical value 8. This means that information leakage in the permutation process is negligible.

Table 3. Entropy value for the images permuted with CROSS and Socek method controlled by PWLCM and perturbed one.

	Original Image	PWLCM		Perturbed PWLCM	
		Cross	Socek	Cross	Socek
Entropy	7.762	7.881	7.888	7.913	7.950

5. CONCLUSION

In this paper, we propose a novel chaotic permutation techniques based on the previously proposed method by Lee and Socek [5, 7]. A much higher security level can be obtained with our approach. First, the length of the chaotic orbit cycle is increased. Then, the entropy value is improved. So, we can say that our chaotic permutation methods are more secure and suitable for chaotic image encryption schemes.

6. REFERENCES

[1] A. Awad, S. E. Assad, Q. Wang, C. Vlădeanu, B. Bakhache, “Comparative Study of 1-D Chaotic Generators for Digital Data Encryption,” IAENG International Journal of Computer Science, vol. 35, no. 4, 2008.

[2] G. Millérioux, J. M. Amigo, J. Daafouz, “A connection between chaotic and conventional cryptography,” IEEE Trans. Circuits and Systems, vol. 55, no. 6, pp. 1695-1703, Jul. 2008.

[3] G. Alvarez, S. Li, “Some Basic Cryptographic Requirements for Chaos Based Cryptosystems,” International Journal of Bifurcation and Chaos, vol. 16, no. 8, pp. 2129-2151, 2006.

[4] Z. Shi, R. Lee, "Bit Permutation Instructions for Accelerating Software Cryptography," IEEE, Application-specific Systems, Architectures and Processors, pp. 138-148, 2000.

[5] R. B. Lee, Z. Shi, X. Yang, “Efficient Permutation Instructions for Fast Software Cryptography,” IEEE Micro, vol. 21, no. 6, pp. 56-69, 2001.

[6] Y. Hilewitz, Z. J. Shi, R. B. Lee, “Comparing Fast Implementations of Bit Permutation Instruction,” IEEE, Signals Systems and Computers, vol.2, 1856 – 1863, 2004.

[7] D. Socek, S. Li, S. S. Magliveras, B. Furht, “Enhanced 1-D Chaotic Key Based Algorithm for Image Encryption,” IEEE, Security and Privacy for Emerging Areas in Communications Networks, 2005.

[8] A. Awad, S. E. Assad, D. Carragata, “A Robust Cryptosystem Based Chaos for Secure Data,” IEEE, Image/Video Communications over fixed and mobile networks, Bilbao Spain, 2008.

[9] T. Yang, C. W. Wu, L. O. Chua, “Cryptography Based on Chaotic Systems,” IEEE Trans. Circuits and Systems, vol. 44, no. 5, pp. 469–472, 1997.