Eur. Phys. J. B (2013) 86: 459
DOI: 10.1140/epjb/e2013-40199-7

THE EUROPEAN PHYSICAL JOURNAL B

Regular Article

A novel stream encryption scheme with avalanche effect

Lequan Min[1,a] and Guanrong Chen[2]

[1] School of Mathematics and Physics, University of Science and Technology Beijing, Beijing 100083, P.R. China
[2] Department of Electronic Engineering, City University of Hong Kong, Hong Kong SAR, P.R. China

Received 12 March 2013 / Received in final form 10 July 2013
Published online 7 November 2013

Abstract. This paper proposes a novel stream encryption scheme with avalanche effect (SESAE). Using this scheme and an ideal pseudorandom number generator (PRNG) to generate d-bit segment binary key streams, one can encrypt a plaintext such that by using any key stream generated from a different seed to decrypt the ciphertext, the decrypted plaintext will become an avalanche-like text which has $2^d - 1$ consecutive one's with a high probability. As a cost, the required bits of the ciphertext are d times those of the plaintext. A corresponding avalanche-type encryption theorem is established. Two chaotic 12-bit segment PRNGs are designed. A generalized FIPS140 test and SESAE test for the two chaotic PRNGs, RC4 12-bit segment PRNG and 12-bit segment Matlab PRNG are implemented. The SESAE tests for 16-bit segment PRNGs are also compared. The results suggest that those PRNGs are able to generate the SESAEs which are similar to those generated via ideal PRNGs.

1 Introduction

Chaos-based encryption techniques has received increasing attention recently (see, for example, [1–3]). An avalanche effect refers to a special and desirable property of cryptographic algorithms. In the case of quality ciphers, the avalanche effect means that a small change in the key or in the plaintext will cause a drastic change in the ciphertext just like an avalanche.

The term avalanche effect was first used in cryptography by Feistel [4]. The original concept dates back to as early as Shannon's diffusion theory [5].

Forre [6] presented a strict avalanche criterion as a generalization of the early concept of avalanche effect. Zhang and Zheng [7] discussed a so-called global avalanche characteristic. Castro et al. [8] introduced a strict avalanche criterion for randomness test and used it to measure the strengths of some well-known PRNGs in the literature. Some detailed discussions on the avalanche criterion of Boolean functions are given in [9].

The avalanche effects in cryptography are classified two types: plaintext avalanche and key avalanche. Strict plaintext avalanche criterion requires that each bit of the ciphertext block should have a change with the probability of one-half whenever any bit of the plaintext is changed [10]. Strict key avalanche criterion requires that each bit of the ciphertext block should have a change with the probability of one-half whenever any bit of the key is changed [10]. This paper will discuss the key avalanche effect.

[a] e-mail: minlequan@gmail.com

If one tosses a fair coin, there is a 1/2 probability that one guesses correctly which side of the coin will be face-up. If one tosses a fair polyhedron with 2^d faces, there will be only a $1/2^d$ probability that one guesses correctly which side of the polyhedron will be face-up.

Similarly to the fair polyhedron tossing game, this paper presents a novel SESAE. In the SESAE, if using any d-bit segment binary key stream generated by a different seed to decrypt the ciphertext, then the decrypted plaintext will become an avalanche-like text, that is, the text has consecutive one's with a probability $(2^d-1)/2^d$ assuming that the used PRNG is able to produce ideal d-segment binary key streams.

The rest of this paper is organized as follows. Section 2 presents a novel SESAE, gives a definition of the ideal PRNG, and shows a criterion theorem on avalanche effect. Section 3 introduces two novel chaotic 12-bit segment PRNGs and compares their randomness with an RC4 12-bit segment PRNG and a Matlab 12-bit segment PRNG via the F140-2 test. SESAE simulation experiments on the four PRNGs are then compared. Furthermore the SESAE tests for 16-bit segment PRNGs similar to the above four PRNGS are also compared. Finally, some concluding remarks are presented in Section 4.

2 SESAE, ideal PRNG and criterion theorem

Two pseudorandom number binary sequences $(\{0, 1\})$ with d-bit segments generated via an ideal PRNG with different seeds or keys should have approximately

$(2^d - 1)/2^d \times 100\%$ different bits. This property motives us to propose a novel SESAE as follows.

Definition 1 (SESAE). *Let $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$ be a binary key stream with d-bit segments generated by a PRNG, $\mathcal{M} = \{m_1, m_2, \ldots, m_n\}$ a binary plaintext stream, and $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$ a ciphertext stream. Then, SESAE is described as follows.*

(1) *The ciphertext $\mathcal{C} = E(\mathcal{M}, \mathcal{P})$ is determined by*

$$c_i = \begin{cases} p_i \text{ if } m_i = 0, \\ \sim p_i \text{ if } m_i = 1, \end{cases} \tag{1}$$

where $\sim p_i$ is defined to be the bit string obtained by replacing all 0s in p_i with 1s, and all 1s in p_i with 0s.

(2) *The corresponding decrypted plaintext $\mathcal{M} = E^{-1}(\mathcal{C}, \mathcal{P})$ is determined by*

$$m_i = \begin{cases} 0 \text{ if } c_i = p_i, \\ 1 \text{ if } c_i \neq p_i. \end{cases} \tag{2}$$

Definition 2. *A PRNG, S, which generates binary d-bit key streams, is called an ideal PRNG, if S has the following properties.*

(1) *The period of any key stream generated by the PRNG is larger than 2^d. Its seed space and key space are larger than 2^{512}.*
(2) *In one period of a pseudorandom key streams generated by the PRNG, the distributions of different d-bit segments in the key stream is homogenous. That is, if the period $p = n \times 2^d$, then the number of each different d-bit segment is equal to n. If the the period p is not an integer multiple of 2^d, then the difference between the numbers of different d-bit segments is at most one.*
(3) *The two key streams $\mathcal{P}_1$, $\mathcal{P}_2$ generated by any two different seeds have $(2^d - 1)/2^d \times 100\%$ different d-bit segments.*

Remark 1.

(1) The assumption on the long period guarantees that the PRNG is suitable for encrypting long bit plaintexts.
(2) The assumption on the large key and seed space guarantees that the PRNG is able to against brute-force attacks.

Theorem 1. *Let S be an ideal PRNG. Then, any encrypted plaintext using SESAE and any key stream generated by PRNG has avalanche effect: if any key stream generated by different seeds is used to decrypt an encrypted plaintext, then consecutive ones will appear in the decrypted text with a probability of $(2^d - 1)/2^d$.*

Proof. Let $\mathcal{P}, \mathcal{M}$ and $\mathcal{C}$ be defined by Definition 1. Denote $\tilde{\mathcal{P}} = \{\tilde{p_1}, \tilde{p_2}, \ldots, \tilde{p_n}\}$ to be another key stream generated by the ideal PRNG with a different seed. Then, by the assumption, one has probabilities

$$Pr[p_j = \tilde{p}_j] = \frac{1}{2^d}, \quad j = 1, 2, \ldots, n.$$

Hence, for a ciphertext $\mathcal{C} = \{c_1, c_2, \ldots, c_n\}$, if $c_j = p_j$, then

$$Pr[c_j = \tilde{p}_j] = \frac{1}{2^d}, \quad j = 1, 2, \ldots, n.$$

Since the distributions of different d-bit segments in $\tilde{\mathcal{P}}$ are homogenous, if $c_j = \sim p_j$ then one still has

$$Pr[c_j = \tilde{p}_j] = \frac{1}{2^d}, \quad j = 1, 2, \ldots, n.$$

Consequently, if one uses the key stream $\tilde{\mathcal{P}}$ to decrypt the ciphertext $\mathcal{C}$, consecutive ones will appear in the decrypted text with a probability of $(2^d - 1)/2^d$. This completes the proof.

Remark 2. If the plaintext is an RGB image, then the decrypted image becomes a pure white image with a probability $\leq (2^d - 1)/2^d$ because one color pixel may have more than one brightness code which is less than 255.

In order to prevent opponents' attacks, one can propose a "one-time-pad" scheme as follows.

Let X be a set in the key and seed space of a PRNG, and assume that Alice and Bob share a one-to-one mapping $f: X \to X$. The sender (Alice or Bob) selects randomly an element $x \in X$ and send it to the receiver. Then, they can use $f(x)$ as key and seed for one-time encryption.

Remark 3. This scheme is easily realized for chaos-based PRNGs because their key and seed spaces are subsets of $\mathbb{R}^n$, and such mappings are easily constructed. Consequently, opponents have to use brute-force attacks to decrypt ciphertexts, which is almost impossible since the key and seed space is lager than 2^{512}. Even if the key size is 2^{128} bits, the time required at 10^6 encryptions/μS is 6.4×10^6 years [11].

3 PRNGs, randomness tests and simulation experiments on SESAE

In the first two subsections, two 12-bit segment chaotic PRNGs based on a discrete Chua circuit and a discrete Chen equation are considered, aiming to illustrate the effectiveness of the above theoretical results in computer simulations, and implement SESAE experiments. In Section 3.3, the RC4 PRNG and the Matlab PRNG (pseudorandom algorithm) are also used with SESAE for comparing avalanche effects using the two chaotic PRNGs. Section 3.4 compares the SESAE tests for 16-bit segment PRNGs similar to the above four PRNGs.

3.1 Chua circuit based PRNG, F140 test and experiments on SESAE

3.1.1 Discrete Chua circuit

Based on the third-order Chua circuit [12], and a smooth Chua circuit [13], a six-order discrete Chua circuit is introduced with a generalized synchronization property.

The driving part of the discrete Chua circuit has the following form:

$$\boldsymbol{X}(k+1)=\begin{bmatrix}x(k+1)\\ y(k+1)\\ z(k+1)\end{bmatrix}=\begin{cases}x(k)+\epsilon\alpha[y(k)-x(k)-bx(k)\\ \quad -\dfrac{a-b}{\pi}\arctan(ux(k))]\\ y(k)+\epsilon[x(k)-y(k)+z(k)]\\ z(k)+\epsilon[-\beta y(k)-\gamma z(k)],\end{cases}\tag{3}$$

where $\alpha=10$, $\beta=15$, $\gamma=0.01$, $a=-2$, $b=-0.78$, $\epsilon=0.05$, $u=\sqrt{27}$.

The driven part has the following form:

$$\mathbf{Y}(k+1)=\begin{bmatrix}y_1(k+1)\\ y_2(k+1)\\ y_3(k+1)\end{bmatrix}=\boldsymbol{A}\boldsymbol{X}(k+1)-\frac{1}{8}(\boldsymbol{A}\boldsymbol{X}(k)-\mathbf{Y}(k)),\tag{4}$$

where

$$\boldsymbol{A}=\begin{bmatrix}-3.7972 & 0.015274 & 0.41865\\ 0.19872 & -3.2532 & 0.84622\\ 0.60379 & 0.4451 & -3.4748\end{bmatrix}\tag{5}$$

is an invertible matrix. From Theorem 1 in [14], it follows that systems (3) and (4) are in generalized synchronization with respect to the transformation $H(\cdot)=(h_1(\cdot),h_2(\cdot),h_3(\cdot))^{\mathrm{T}}=\boldsymbol{A}:\mathbb{R}^3\to\mathbb{R}^3$.

The calculated Lyapunov exponents of system (3) are 0.00042, 0, 0.0030. Therefore, systems (3) is chaotic. Since systems (3) and (4) are in generalized synchronization with the transformation H, and H is invertible, system (4) is also chaotic. Furthermore, the orbits of a chaotic system are sensitive to the initial conditions of the system, and the invertible transformation H is continuous, the orbits of system (4) are also sensitive to the initial conditions of the system.

Now, select the following initial conditions:

$$x(0)=0.2,\ y(0)=0.2,\ z(0)=0.2;\tag{6}$$

$$\mathbf{Y}(0)=\boldsymbol{A}\boldsymbol{X}(0)+1.\tag{7}$$

The chaotic orbits of the state variables $\boldsymbol{X}(k)$ and $\boldsymbol{Y}(k)$ for the first 10 000 iterations are shown in Figures 1a and 1b. Observe that the dynamic orbits of the driving system are similar to those of the third-order Chua circuit [12].

Figures 2a–2c show that although the initial condition (7) has a perturbation, $\boldsymbol{X}(k)$ and $\boldsymbol{Y}(k)$ are rapidly conversing into generalized synchronization as the theory (Theorem 1 in [14]) predicts.

3.1.2 Generalized FIPS140-2 test

The FIPS 140-2 test issued by the American National Institute of Standards and Technology [15] has been widely used for verifying the statistical properties of the randomness of the pseudo-random numbers generated by PRNGs.

The FIPS 140-2 test consists of four sub-tests: Monobit test, Poker test, Run test and Long Run test. Each test needs a single stream of 20 000 one and zero bits from the keystream generator. Any failure in the first three tests means that the corresponding quantity of the sequences falls out the required intervals listed in the second column in Table 1. The Long Run test is passed if there are no runs of length 26 or more.

Based on the FIPS 140-2 test and the Golomb's three postulates on the randomness that pseudorandom sequences should satisfy [16], a conference paper [17]

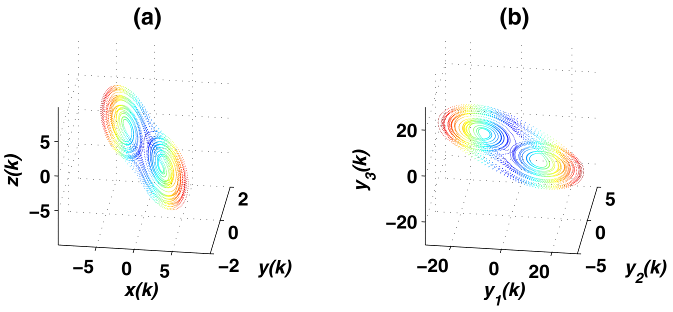


Fig. 1. (Color online) Chaotic orbits of the state variables: (a) $x(k)$-$y(k)$-$z(k)$, (b) $y_1(k)$-$y_2(k)$-$y_3(k)$.

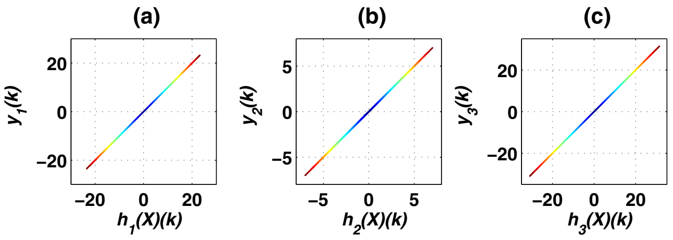


Fig. 2. (Color online) The state vectors $\boldsymbol{X}$ and $\boldsymbol{Y}$ are in generalized synchronization with respect to the transformation H. (a) $h_1(\mathbf{X})(k)$-$y_1(k)$, (b) $h_2(\mathbf{X})(k)$-$y_2(k)$, (c) $h_3(\mathbf{X})(k)$-$y_3(k)$.

Table 1. The required intervals of the FIPS 140-2 Monobit test, Pork tests, Run test. Here, MT, PT, and LT represent the Monobit test, the Pork test and the Long Run test, respectively. k represents the length of the run of a tested sequence. χ^2 DT represents χ^2 distribution.

Test Item	FIPS 140-2 Required intervals	$\alpha=10^{-4}$ Required intervals	Golomb's Postulates
MT	9725–10 275	9725–10 275	10 000
PT	2.16–46.17	2.16–46.17	χ^2 DT
LT	<26	<26	–
k	Run Test	Run test	
1	2315–2685	2362–2638	2500
2	1114–1386	1153–1347	1250
3	527–723	556–694	625
4	240–384	264–361	313
5	103–209	122–191	156
6+	103–209	122–191	156

has proposed the Monobit test, Pork test, Run test and Longest Run test for d-bit segment binary sequence case:

(1) Change the d-bit binary sequence ϵ with length n into its corresponding decimal number sequence $\{0, 1, \ldots, 2^d - 1\}^n$. Denote the sequence by:

$$\epsilon = \epsilon_1 \epsilon_2 \ldots \epsilon_n. \tag{8}$$

(2) For any $\epsilon_i \in \epsilon$, denote $2^d - 1 - \epsilon_i$ by $\sim \epsilon_i$.
(3) For each fixed i, taking ϵ_i and $\sim \epsilon_i$ consecutively from ϵ, we obtain $2^d/2$ new sequences and denote them by:

$$E_i = E_{n_1} E_{n_2} \ldots E_{n_i}, \quad i = 1, 2, \ldots, 2^d/2. \tag{9}$$

(4) For any fixed i and E_i, replace $\epsilon_i \in E_i$ by 0, and $\sim \epsilon_i$ by 1. Denote the new sequences by:

$$\tilde{\epsilon}_i = \tilde{\epsilon}_{n_1} \tilde{\epsilon}_{n_2} \ldots \tilde{\epsilon}_{n_i}, \quad i = 1, \ldots, 2^d/2. \tag{10}$$

(5) For the case of $n = 10\,000 \times 2^d$, one can use the FIPS 140-2 suite to test the $\{0, 1\}$ sequence $\tilde{\epsilon}_i's$. If taken significant level $\alpha = 0.0001$, the required intervals for the tests are listed in the third column in Table 1 (see [17] for detail).

According to Golomb's three postulates [16], the ideal values of the Monobit test and the Run test are listed in the 4th column in Table 1.

3.1.3 A novel PRNG and generalized FIPS140-2 test

Let

$$y_1(k), y_2(k), y_3(k), k = 1, \ldots, N$$

be a chaotic stream generated by the discrete Chua circuit (4). Denote

$$Y_N = \{y_1(i) + y_2(i) + y_3(i) | i = 1, 2, \ldots, N\}, \tag{11}$$

$$normal(\mathcal{Y}_N) = \frac{Y_N - \min(Y_N)}{\max(Y_N) - \min(Y_N)}. \tag{12}$$

Let $L = 12$ and T be a transformation form $\mathbb{Z}^+$ to binary bits. Then, one has a binary sequence with $L \times N$ bits:

$$\begin{aligned} \mathcal{P} &= T(normal(\mathcal{Y}_N)) \\ &= \mathrm{mod}(round(10^{16} \times normal(\mathcal{Y}_N)), 2^L). \end{aligned} \tag{13}$$

Consequently, a novel chaotic PRNG (called PRNG I) is constructed via equations (3)–(7) and (11)–(13).

Remark 4. The perturbed initial conditions ((6) and (7)) and the perturbed matrix $\boldsymbol{A}$ defined by (5) can be used as keys of PRNG I. The perturbed initial conditions can also be used as seeds of PRNG I.

Table 2. The tested Mean ± SD of all $E_i's$ defined by (9) for the 100 key streams with length $10\,000 \times 2^{12}$ generated by PRNG I and RC4 PRNG, respectively. Here, MT, PT, and LT represent the Monobit test, Pork test and Long Run test, respectively; k represents the length of the run of a tested sequence.

Test item	Bits $\{\varepsilon_i, \sim\varepsilon_i\}$	PRNG I Mean ± SD	RC4 Mean ± SD
MT	ε_i	$10\,000 \pm 247.8$	$10\,000 \pm 100.00$
	$\sim\varepsilon_i$	9999.8 ± 247.55	$10\,000 \pm 99.80$
PT	–	15.01 ± 5.49	25.352 ± 8.47
LT	ε_i	13.65 ± 1.9312	13.62 ± 1.87
	$\sim\varepsilon_i$	13.66 ± 1.9264	13.62 ± 1.87
k		Run test	
1	ε_i	2490.3 ± 74.968	2499.9 ± 50.3
	$\sim\varepsilon_i$	2490.4 ± 75.2381	2499.9 ± 50.05
2	ε_i	1246.7 ± 32.974	1250.1 ± 33.06
	$\sim\varepsilon_i$	1246.8 ± 32.991	1250.0 ± 33.11
3	ε_i	624.54 ± 27.236	624.87 ± 23.37
	$\sim\varepsilon_i$	624.45 ± 27.21	624.91 ± 23.45
4	ε_i	312.85 ± 21.938	312.44 ± 16.84
	$\sim\varepsilon_i$	312.80 ± 21.918	312.48 ± 16.81
5	ε_i	156.84 ± 16.154	156.23 ± 12.12
	$\sim\varepsilon_i$	156.82 ± 16.138	156.21 ± 12.10
6^+	ε_i	157.84 ± 21.404	156.21 ± 11.88
	$\sim\varepsilon_i$	157.82 ± 21.4332	156.13 ± 11.819

Now, select the following initial conditions:

$$[x(0), y(0), z(0)]^{\mathrm{T}} = [1.2658, 0.7821, 0.0867]^{\mathrm{T}} \tag{14}$$

$$\mathbf{Y}(0) = \boldsymbol{A}\mathbf{X}(0) + 1. \tag{15}$$

The FIPS 140-2 test is used to check 100 12-bit segment binary keystreams with length $10\,000 \times 2^{12}$ generated by PRNG I with perturbed randomly initial conditions (14) and (15), and the elements of matrix (5) in the range $|\epsilon| \in [10^{-16}, 10^{-13}]$.

Calculate the mean value (denoted by Mean) and the standard deviation (denoted by SD) of the FIPS 140-2 tests to all $E_i's$ defined by (9) for the 100 key streams. The results are shown in the 3th column in Table 2. Observe that the mean values of all tested items are very close to the corresponding ideal values listed in the 4th column in Table 1.

3.1.4 RC4 algorithm based PRNG and FIPS140-2 test

The RC4 was designed by Rivest of the RSA Security in 1987, which has been widely used in popular protocols such as Secure Sockets. The RC4 Algorithm based 12-bit segment PRNG can be designed via Matlab commands:

```
L=12; N = 10000*2^L;
K=randint(1,2^L,[0 2^L-1]);
S=[0:2^L-1];j=0;
for i=1:2^L
        j=mod(j+S(i)+K(i),2^L);
            Sk=S(j+1);
               S(j+1)=S(i);
                   S(i)=Sk;
end
      C=zeros(1,N); j=0;i=0; k=1;
for l=1:N
         i=mod(i+1,2^L);
            j=mod(j+S(i+1),2^L);
               Sk=S(j+1);
                  S(j+1)=S(i+1);
                     S(i+1)=Sk;
 C(l)=S(mod(S(j+1)+S(i+1),2^L)+1);
 end
```

Here, "2^L" represents 2^L; "randint(1,2^L,[0 2^L-1])" generates a vector of uniformly distributed random integers $\{0, 1, \ldots, 2^L - 1\}$ of dimension 2^L; "mod" means modulus after division; "zeros(1,N)" is a zero raw vector of dimension N.

Let T be a transformation form $\mathbb{Z}^+$ to binary bits. Then, one has a binary sequence with $L \times N$ bits:

$$\mathcal{P} = T(C).$$

Consequently, the RC4 Algorithm based 12-bit segment PRNG is designed.

Remark 5. The randomly chosen K=randint(1,2^L,[0 2^L-1]) can be used as keys of RC PRNG.

The FIPS 140-2 test is used to check 100 12-bit segment binary keystreams with length $N = 10\,000 \times 2^{12}$ generated by RC4 PRNG with perturbed randomly key K.

Calculate the mean value and the standard deviation of the FIPS 140-2 tests to all $E_i's$ defined by (9) for the 100 key streams. The results are shown in the 4th column in Table 2.

Observe that the mean values of all tested items are very close to the corresponding ideal values listed in the 4th column in Table 1. The results for the Monobit test, the one-run test and the "large than 5-run" test for RC4 PRNG are better than those of PRNG I.

3.1.5 Simulations on SESAE

Now, the avalanche effect of SESAE is investigated, which is used to encrypt and decrypt an RGB image Flowers with 167×121 pixels as shown in Figure 3a based on PRNG I.

The simulation is implemented via the Matlab 7.1 platform on a PC computer.

(1) Transform the image Flowers to a binary plaintext steam $\mathcal{M} = \{m_1, m_2, \ldots, m_n\}$, where $n = 167 \times 121 \times 3 \times 8$.
(2) Use PRNG I with initial conditions (6) and (7) to generate a 12-bit segment key stream with length n: $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$.
(3) Use formula (1), after dropping the first 10 000 iterative values of the key stream $\mathcal{P}$, to encrypt the plaintext steam $\mathcal{M}$, and obtain a ciphertext $\mathcal{C} = E(\mathcal{M}, \mathcal{P})$.
(4) Drop the first 10 000 iterative values (see (3) and (4)). Use formula (2) to decrypt the ciphertext and obtain a decrypted plaintext image $\bar{\mathcal{M}} = E^{-1}(\mathcal{C}, \mathcal{P})$ without errors.
(5) Randomly disturb the initial conditions (14) and (15), and matrix (5), for 1000 times in the range $|\epsilon| \in [10^{-16}, 10^{-13}]$, to obtain key streams (dropping the first 10,000 iterative values):

$$\mathcal{P}_i, \quad i = 1, 2, \ldots, 1000.$$

(6) Use $\{\mathcal{P}_1, \ldots, \mathcal{P}_{1000}\}$ to decrypt the ciphertext and obtain decrypted plaintext, respectively:

$$\bar{\mathcal{M}}_i = E^{-1}(\mathcal{C}, \mathcal{P}_i), \quad i = 1, \ldots, 1000.$$

After changing $\bar{\mathcal{M}}_i$ to RGB images, all images become almost pure white images. There are total of 484 968 $\{0, 1\}$ codes in each decrypted image. Among the decrypted images, the minimum number of '0's in the decrypted images is 88, and the maximum one is 152. The two decrypted images (denoted by $I_{88,1}$ and $I_{152,1}$) are shown in Figure 3[1].

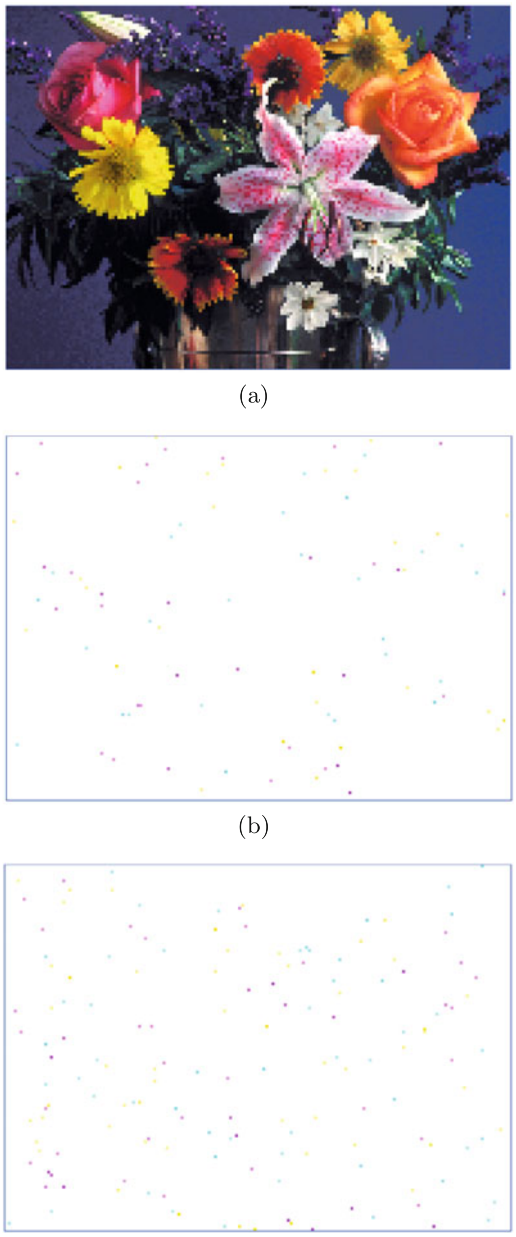
Denote $I_{i,j}$ to be the jth image with number i of 0 codes. Table 3 lists the static data of the first 10 images with minimum '0's codes and the last 10 decrypted images with maximum '0's codes. Here $\mathcal{N}_1$ and $\mathcal{N}_2$ represent the number of '0's and the number of the color pixels with brightness less than 255, respectively; Mean represents the average brightness of color pixels in the decrypted images; Percentage represents the percentage of the 1 codes in the decrypted image. Observe that the percentages of the numbers of "1" codes are in the rang [99.9687, 99.9819], which is very close to the ideal value $(2^{12} - 1)/2^{12} = 99.9756$.

Table 4 lists some statistical data of the perturbed keys corresponding to the above ten decrypted images. The results may suggest that there are no significant correlations between the perturbed parameters and the decrypted images if the perturbation $|\epsilon|$ is larger than 10^{-16}.

In summary, the simulation shows that using PRNG I and SESAE to encrypt RGB images is able to generate encrypted images with significant avalanche effects. The key space of PRNG I may be larger than $10^{15\times15} > 2^{747}$ [2].

[1] In order to make the pixels whose brightness is less than 255 be visible in the decrypted images, hereafter we reduce the brightness of the three color planes of these pixels by 100, respectively.

[2] It can be proved that if perturbing the elements of matrix (5) in the range $|\epsilon| \in [10^{-16}, 10^{-1}]$, the perturbed matrices are still invertible.


(a)

(b)

(c)

Fig. 3. (Color online) (a) Original image Flowers. Two decrypted images via key streams generated with slightly perturbed initial conditions and system parameters in the range $[10^{-16}, 10^{-13}]$: (b) $I_{88,1}$, and (c) $I_{152,1}$.

Table 3. The static data of the first 10 images with minimum '0's codes and the last 10 decrypted images with maximum '0's codes. Here $\mathcal{N}_1$ and $\mathcal{N}_2$ represent the number of '0's and the number of the color pixels with brightness less than 255, respectively; Mean represents the average brightness of color pixels in the decrypted images; Percentage represents the percentage of the 1 codes in the decrypted image.

Images	$\mathcal{N}_1$	$\mathcal{N}_1$	Mean	Percentage
$I_{88,1}$	88	88	254.9503	99.9819
$I_{89,1}$	89	89	254.9516	99.9816
$I_{89,2}$	89	88	254.9493	99.9816
$I_{91,1}$	91	91	254.9555	99.9812
$I_{92,1}$	92	92	254.9417	99.9810
$I_{94,1}$	94	94	254.9485	99.9806
$I_{94,2}$	94	93	254.9566	99.9806
$I_{95,1}$	95	95	254.9438	99.9804
$I_{95,2}$	95	95	254.9531	99.9804
$I_{95,3}$	95	95	254.9377	99.9804
$I_{145,1}$	145	145	254.9187	99.9701
$I_{146,1}$	146	146	254.9215	99.9699
$I_{146,2}$	146	146	254.9207	99.9699
$I_{147,1}$	147	147	254.9325	99.9697
$I_{147,2}$	147	147	254.9146	99.9697
$I_{147,3}$	147	147	254.9179	99.9697
$I_{148,1}$	148	148	254.9416	99.9695
$I_{148,2}$	148	148	254.9306	99.9695
$I_{149,1}$	149	148	254.9380	99.9693
$I_{152,1}$	152	152	254.9154	99.9687

3.2 Chen equation based PRNG, FIPS140-2 test and experiments on SESAE

3.2.1 Discrete Chen system

Based on the third-order Chen system [18], a six-order discrete Chen system is introduced through generalized synchronization.

The driving part of the discrete Chen system has the following form:

$$\mathbf{X}(k+1) = \begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ x_3(k+1) \end{bmatrix} = \begin{cases} x_1(k) + \epsilon a[x_2(k) - x_1(k)] \\ x_2(k) + \epsilon[(c-a)x_1(k) - x_1(k)x_3(k) \\ \quad + cx_2(k)] \\ x_3(k) + \epsilon[x_1(k)x_2(k) - bx_3(k)], \end{cases} \tag{16}$$

where $a = 35$, $b = 3$, $c = 28$, $\epsilon = 0.01$.

The driven system has the following form:

$$\mathbf{Y}(k+1) = \begin{bmatrix} y_1(k+1) \\ y_2(k+1) \\ y_3(k+1) \end{bmatrix} = \boldsymbol{A}\mathbf{X}(k+1) - \frac{1}{8}(\boldsymbol{A}\mathbf{X}(k) - \mathbf{Y}(k)), \tag{17}$$

$$\boldsymbol{A} = \begin{bmatrix} -3.7972 & 0.015274 & -1.58135 \\ 1.1987 & -2.2532 & -1.15378 \\ -1.39621 & -1.5549 & -4.4748 \end{bmatrix} \tag{18}$$

is an invertible matrix. The calculated Lyapunov exponents of system (16) are $\{0.0017, 0, -1.0115\}$. From Theorem 1 given in [14], one knows that systems (16) and (17) can achieve generalized synchronization with respect to the transformation $H = \boldsymbol{A}\colon \mathbb{R}^3 \to \mathbb{R}^3$. Hence systems (16) and (17) are chaotic and sensitive to the initial conditions of the two systems.

Table 4. The minimums, means, and maximums in absolute values of the 15 perturbed parameters corresponding to 20 chaotic steams used for decrypted images $I'_{i,j}s$.

Images	Min ($\times 10^{-13}$)	Mean ($\times 10^{-13}$)	Max ($\times 10^{-13}$)
$I_{88,1}$	0.0787	0.6513	0.9963
$I_{89,1}$	0.0314	0.4850	0.9660
$I_{89,2}$	0.0842	0.5566	0.9511
$I_{91,1}$	0.0088	0.4661	0.9805
$I_{92,1}$	0.0231	0.4298	0.9923
$I_{94,1}$	0.0186	0.4020	0.9091
$I_{94,2}$	0.0826	0.4652	0.7953
$I_{95,1}$	0.0000	0.4862	0.9389
$I_{95,2}$	0.0411	0.4202	0.8797
$I_{95,3}$	0.0055	0.4992	0.8460
$I_{145,1}$	0.0157	0.5407	0.8511
$I_{146,1}$	0.0016	0.4604	0.89539
$I_{146,2}$	0.0703	0.5006	0.9767
$I_{147,1}$	0.0775	0.4451	0.9990
$I_{147,2}$	0.0420	0.4211	0.9773
$I_{147,3}$	0.0065	0.5470	0.9780
$I_{148,1}$	0.0866	0.4526	0.9151
$I_{148,2}$	0.0038	0.4329	0.8624
$I_{149,1}$	0.0145	0.4500	0.9310
$I_{152,1}$	0.0202	0.4429	0.99637

Here, select the same initial conditions (6) and (7) given in the discrete Chua circuit. The chaotic orbits of the state vectors $\boldsymbol{X}(k)$ and $\boldsymbol{Y}(k)$ for the first 10 000 iterations are shown in Figures 4a and 4b, respectively. Observe that the dynamic orbits of the driving system are similar to those of the third-order Chen system.

Figures 5a–5c show that, although the initial condition (6) has a perturbation, $\boldsymbol{X}(k)$ and $\boldsymbol{Y}(k)$ are rapidly converging into generalized synchronization as the theory (Theorem 1 in [14]) predicts.

3.2.2 New PRNG and FIPS140-2 test

Let

$$y_1(k), y_2(k), y_3(k), k = 1, \ldots, N$$

be chaotic streams generated by the discrete Chen system (17). Using formulas (12) and (13), one can design a chaotic PRNG (called PRNG II).

Select the following initial conditions:

$$[x(0), y(0), z(0)]^{\mathrm{T}} = [-15.871, 2.9297, 13.913]^{\mathrm{T}} \tag{19}$$

$$\boldsymbol{Y}(0) = \boldsymbol{A}\boldsymbol{X}(0) + 1. \tag{20}$$

The perturbed initial conditions (19), (20) and the perturbed matrix $\boldsymbol{A}$ defined by (18) can be used as keys of PRNG II. The perturbed initial conditions can also be used as seeds of PRNG II.

The FIPS 140-2 test is used to check 100 12-bit segment binary keystreams with length $10\,000 \times 2^{12}$ generated by PRNG II with perturbed randomly initial conditions (19) and (20), and matrix (18) in the range $|\epsilon| \in [10^{-16}, 10^{-13}]$.

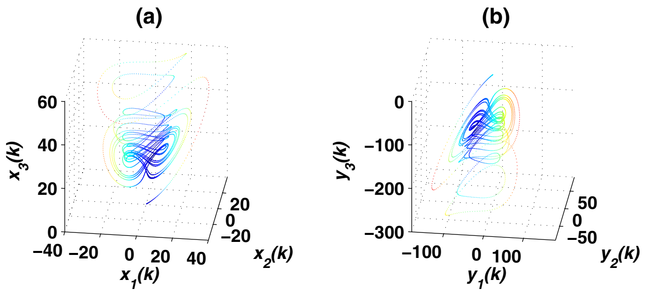


Fig. 4. (Color online) Chaotic orbits of the components of the state vectors: (a) $x(k)$-$y(k)$-$z(k)$, (b) $y_1(k)$-$y_2(k)$-$y_3(k)$.

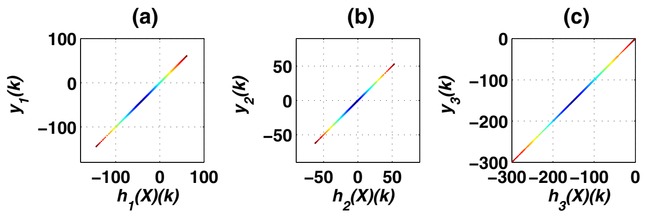


Fig. 5. (Color online) The state vectors $\boldsymbol{X}$ and $\boldsymbol{Y}$ are in generalized synchronization with respect to the transformation H. (a) $h_1(\boldsymbol{X})(k)$-$y_1(k)$, (b) $h_2(\boldsymbol{X})(k)$-$y_2(k)$, and (c) $h_3(\boldsymbol{X})(k)$-$y_3(k)$.

Calculate the mean value and the standard deviation of the FIPS 140-2 tests to all $E'_i s$ defined by (9) for the 100 key streams. The results are shown in the 3th column in Table 5. Observe that the mean values of all tested items are very close to the corresponding ideal values listed in the 4th column in Table 1.

3.2.3 Matlab algorithm based PRNG and FIPS140-2 test

Use matlab command randint(1,10000*2^L,[0 2^L-1]) where L = 12 generates 100 12-bit segment steams $\mathcal{C}_1, \ldots, \mathcal{C}_{100}$.

Let T be a transformation form $\mathbb{Z}^+$ to binary bits. Then, one has a binary sequence with $L \times N$ bits:

$$\mathcal{P} = T(C).$$

Consequently, the Matlab Algorithm based 12-bit PRNG is designed.

Then, use the FIPS 140-2 test to measure the 100 key streams. Calculate the mean value and the standard deviation of the FIPS 140-2 tests to all $E'_i s$ defined by (9) for the 100 key streams. The results are shown in the 4th column in Table 5. Observe that the means of all tested items are very close to the corresponding ideal values listed in the 4th column in Table 1. The results for the Monobit test and the one-run test are better than those of PRNG II.

3.2.4 Simulations on SESAE

Now, the avalanche effect of SESAE is investigated, which encrypts and decrypts the image Lina with 128×128 pixels shown in Figure 6a, using PRNG II. The simulation is implemented by Matlab 7.1 on a PC.

Table 5. The tested Mean ± SD of all $E_i^{\prime}s$ defined by (9) for the 100 key streams with length $10\,000 \times 2^{12}$ generated by PRNG I, and Matlab PRNG, respectively. Here, MT, PT, and LT represent the Monobit test, Pork test and Long Run test, respectively; k represents the length of the run of a tested sequence.

Test item	bits $\{\varepsilon_i, \sim\varepsilon_i\}$	PRNG II Mean ± SD	Matlab Mean ± SD
MT	ε_i	9992.9 ± 283.53	9999.5 ± 100.02
	$\sim\varepsilon_i$	9993.1 ± 283.38	$10,000 \pm 100.27$
PT	–	15.013 ± 5.5039	15.045 ± 5.4757
LT	ε_i	13.616 ± 1.9149	13.60 ± 1.8675
	$\sim\varepsilon_i$	13.621 ± 1.9176	13.63 ± 1.8783
k		Run test	
1	ε_i	2498.1 ± 83.056	2499.8 ± 49.765
	$\sim\varepsilon_i$	2498.1 ± 83.059	2499.7 ± 49.742
2	ε_i	1249 ± 46.733	1249.8 ± 33.313
	$\sim\varepsilon_i$	1249 ± 46.821	1249.8 ± 33.587
3	ε_i	624.43 ± 28.709	625.13 ± 23.499
	$\sim\varepsilon_i$	624.55 ± 28.665	624.8 ± 23.237
4	ε_i	312.26 ± 18.755	312.22 ± 16.907
	$\sim\varepsilon_i$	312.26 ± 18.762	312.52 ± 16.891
5	ε_i	156.12 ± 12.794	156.2 ± 12.092
	$\sim\varepsilon_i$	156.12 ± 12.782	156.2 ± 12.092
6^+	ε_i	156.11 ± 12.676	156.34 ± 11.906
	$\sim\varepsilon_i$	156.09 ± 12.676	156.38 ± 11.982

(1) Transfer the image Lina to a binary plaintext steam $\mathcal{M} = \{m_1, m_2, \ldots, m_n\}$, where $n = 128 \times 128 \times 3 \times 8$.
(2) Use PRNG II with initial conditions (19) and (20) to generate a 12-bit segment key stream $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$.
(3) Use formula (1) and the key stream $\mathcal{P}$ to encrypt the plaintext steam $\mathcal{M}$, and obtain a ciphertext $\mathcal{C} = E(\mathcal{M}, \mathcal{P})$.
(4) Drop the first 10 000 iterative values (see (16) and (17)). Using formula (2) and the key stream to decrypt the ciphertext, and obtain a decrypted plaintext image $\bar{\mathcal{M}} = E^{-1}(\mathcal{C}, \mathcal{P})$ without errors.
(5) Randomly disturb the initial conditions (19) and (20), and matrix (18) for 1000 times in the range $|\epsilon| \in [10^{-16}, 10^{-14}]$, and obtain key streams (dropping the first 10 000 iterative values):

$$\mathcal{P}_i, \quad i = 1, 2, \ldots, 1000.$$

(6) Use $\{\mathcal{P}_1, \ldots, \mathcal{P}_{1000}\}$ to decrypt the ciphertext, and obtain the decrypted plaintext:

$$\bar{\mathcal{M}}_i = E^{-1}(\mathcal{C}, \mathcal{P}_i),\ i = 1, \ldots, 1000.$$

(7) Change $\bar{\mathcal{M}}_i$ to RGB images.

After changing $\bar{\mathcal{M}}_i$ to RGB images, all images become almost pure white images. There are total of 393 216 $\{0, 1\}$ codes in each decrypted image. Among the decrypted images, the minimum number of '0's in the decrypted is 69, and the maximum one is 131. The two decrypted images (denoted by $I_{69,1}$ and $I_{131,1}$) are shown in Figures 6a and 6b, respectively.

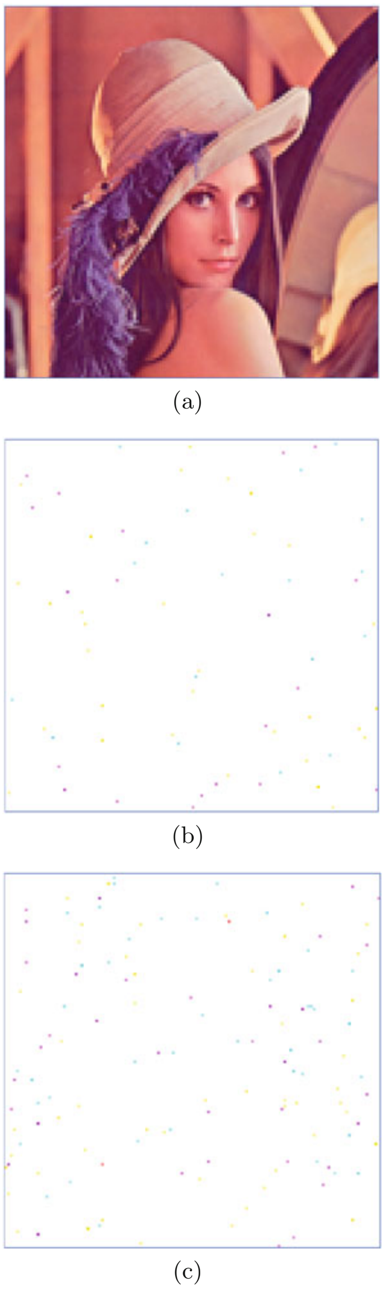


Fig. 6. (Color online) (a) Original image Lena. Two decrypted images via key streams generated with slightly perturbed initial conditions and system parameters in the range $[10^{-16}, 10^{-13}]$: (b) $I_{69,1}$, and (c) $I_{131,1}$.

Denote $I_{i,j}$ to be the jth image with number i of 0 codes. Table 6 lists the static data of the first 10 images with minimum '0's codes and the last 10 decrypted images with maximum '0's codes. Here $\mathcal{N}_1$ and $\mathcal{N}_2$ represent the number of '0's and the number of the color pixels with brightness less than 255, respectively; Mean represents the average brightness of color pixels in the decrypted images; Percentage represents the percentage the 1 codes in the decrypted image. Observe that the percentages of the numbers of "1" codes are in the rang

Table 6. The static data of the first 10 images with minimum '0's codes and the last 10 decrypted images with maximum '0's codes. Here $\mathcal{N}_1$ and $\mathcal{N}_2$ represent the number of '0's and the number of the color pixels with brightness less than 255, respectively; Mean represents the average brightness of color pixels in the decrypted images; Mercentage represents the percentage of the 1 codes in the decrypted image.

Images	$\mathcal{N}_1$	$\mathcal{N}_1$	Mean	Percentage
$I_{69,1}$	69	69	254.9636	99.9825
$I_{71,1}$	71	71	254.9600	99.9819
$I_{71,2}$	71	71	254.9402	99.9819
$I_{71,3}$	71	71	254.9535	99.9819
$I_{73,1}$	73	73	254.9536	99.9814
$I_{73,2}$	73	73	254.9578	99.9814
$I_{75,1}$	75	75	254.9499	99.9809
$I_{75,2}$	75	75	254.9441	99.9809
$I_{75,3}$	75	75	254.9501	99.9809
$I_{76,1}$	76	76	254.9543	99.9807
$I_{121,1}$	121	119	254.9024	99.9692
$I_{122,1}$	122	122	254.9130	99.9690
$I_{122,2}$	122	121	254.9184	99.9690
$I_{122,3}$	122	121	254.9152	99.9690
$I_{125,1}$	125	125	254.9249	99.9682
$I_{125,2}$	125	125	254.9120	99.9682
$I_{125,3}$	125	125	254.9300	99.9682
$I_{127,1}$	127	127	254.9094	99.9677
$I_{128,1}$	128	128	254.9189	99.9674
$I_{131,1}$	131	131	254.9229	99.9667

[99.9667, 99.9825], which is very close to the ideal value $(2^{12}-1)/2^{12} = 99.9756$.

Table 7 lists some statistical data of the perturbed keys corresponding to the above ten decrypted images. The results may suggest that there are no significant correlations between the perturbed parameters and the decrypted images if the perturbation $|\epsilon|$ is larger than 10^{-16}.

In summary, the simulation shows that using PRNG II and SESAE to encrypt RGB images is able to generate encrypted images with significant avalanche effects. The key space of PRNG II may be larger than $10^{15\times15} > 2^{747}$ [3].

3.3 Simulations on other PRNGs on SESAE

3.3.1 Simulation using RC4 PRNG on SESAE

Now the RC4 PRNG (algorithm) described in Section 3.1.4 is used with SESAE to encrypt and decrypt the image Lina.

(1) In the RC4 algorithm, select a seed $K_0 = K =$ randint (1,2^L,[0 2^L-1]) where $L = 12$, and let $l = 167 \times 121 \times 3 \times 8$, to obtain a chaotic stream C.
(2) Change C to a 12-bit segment binary key steam $\mathcal{P}$.
(3) Use formula (1) and the key stream $\mathcal{P}$ to encrypt the plaintext steam $\mathcal{M}$, generated by the image Flowers, and obtain a ciphertext $\mathcal{C} = E(\mathcal{M}, \mathcal{P})$.

[3] It can be proved that if perturbing the elements of matrix (18) in the range $|\epsilon| \in [10^{-16}, 10^{-1}]$, the perturbed matrices are still invertible.

Table 7. The minimums, means, and maximums in absolute values of the 15 perturbed parameters corresponding to 20 chaotic steams used for decrypted images $I'_{i,j}s$.

Images	Min $(\times10^{-13})$	Mean $(\times10^{-13})$	Max $(\times10^{-13})$
$I_{69,1}$	0.1502	0.5710	0.9800
$I_{71,1}$	0.3604	0.6789	0.9437
$I_{71,2}$	0.0264	0.4867	0.9998
$I_{71,3}$	0.0291	0.4556	0.8914
$I_{73,1}$	0.1402	0.4652	0.9608
$I_{73,2}$	0.0334	0.5669	0.9993
$I_{75,1}$	0.0208	0.4474	0.8797
$I_{75,2}$	0.0053	0.4624	0.9503
$I_{75,3}$	0.0304	0.4545	0.9907
$I_{76,1}$	0.0821	0.4417	0.9051
$I_{121,1}$	0.0108	0.4592	0.9973
$I_{122,1}$	0.0613	0.2772	0.6648
$I_{122,2}$	0.0325	0.5335	0.8152
$I_{122,3}$	0.0393	0.5174	0.8720
$I_{125,1}$	0.0410	0.4574	0.9815
$I_{125,2}$	0.0166	0.6666	0.9675
$I_{125,3}$	0.0131	0.5171	0.9796
$I_{127,1}$	0.0924	0.5303	0.8916
$I_{128,1}$	0.0327	0.4638	0.9672
$I_{131,1}$	0.0295	0.4522	0.9776

(4) Again, use Matlab command $K =$ randint(1,2^L,[0 2^L-1]) to generate 1000 12-bit segment binary key steams:

$$\mathcal{P}_i, \quad i = 1, 2, \ldots, 1000.$$

(5) Use $\{\mathcal{P}_1, \ldots, \mathcal{P}_{1000}\}$ to decrypt the ciphertext $\mathcal{C}$, and obtaining the decrypted plaintext:

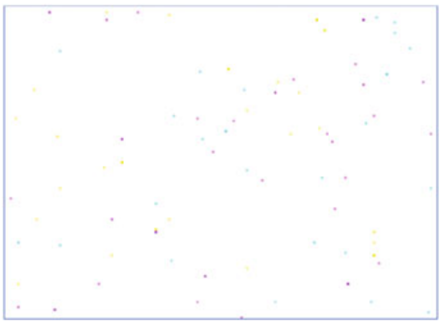
$$\bar{\mathcal{M}}_i = E^{-1}(\mathcal{C}, \mathcal{P}_i),\ i = 1, \ldots, 1000.$$

(6) Change $\bar{\mathcal{M}}_i$ to RGB images.

Totally 1000 images become almost pure white colored images. There are total of 484 968 $\{0, 1\}$ codes in each decrypted image. Among the decrypted images, the minimum number of '0's in the decrypted is 69, and the maximum one is 131. The two decrypted images (denoted by $I_{80,1}$ and $I_{149,1}$) are shown in Figures 7a and 7b, respectively.

Denote $I_{i,j}$ to be the jth image with number i of 0 codes. Table 8 lists the static data of the first 10 images with minimum '0's codes and the last 10 decrypted images with maximum '0's codes. Here $\mathcal{N}_1$ and $\mathcal{N}_2$ represent the number of '0's and the number of the color pixels with brightness less than 255, respectively; Mean represents the average brightness of color pixels in the decrypted images; Percentage represents the percentage of the 1 codes in the decrypted image. Observe that the percentages of the numbers of "1" codes are in the rang [99.9693, 99.9835], which is very close to the ideal value $(2^{12}-1)/2^{12} = 99.9756$.

Table 9 lists some statistical data of the perturbed keys corresponding to the above ten decrypted images. The results may suggest that there are no significant correlations between the perturbed K values and the decrypted images.


(a)

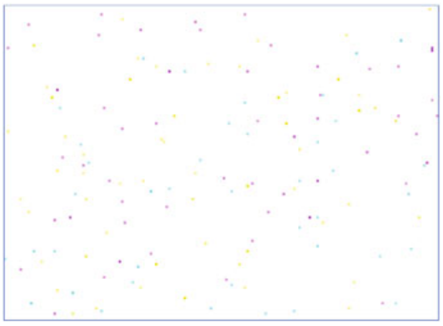

(b)

Fig. 7. (Color online) Two decrypted images via key streams generated perturbed $K_{80,1}$ and $K_{149,1}$: (a) $I_{80,1}$, and (b) $I_{149,1}$.

The above results suggest that using different key stream to decrypt images do not provide useful information on both the original image Flowers and the original seed K_0.

In summary, the simulation shows that using RC4 PRNG with SESAE to encrypt RGB images is able to generate images with significant avalanche effects.

Remark 6. Comparing Table 3 with Table 8 shows that the SESAEs generated by PRNG I and RC4 PRNG do not have significant differences.

3.3.2 Simulation using Matlab PRNG on SESAE

For comparison, the Matlab PRNG is used with SESAE to perform the same simulations.

(1) Use Matlab command randint$(128 * 128 * 3 * 8, 12)$ to generate a 12-bit segment binary key steam $\mathcal{P}$.
(2) Use formula (1) and the key stream $\mathcal{P}$ to encrypt the plaintext steam $\mathcal{M}$ generated by the image Lina, and obtain a ciphertext $\mathcal{C} = E(\mathcal{M}, \mathcal{P})$.
(3) Again use Matlab command randint$(128{*}128{*}3{*}8, 12)$ to generate 1000 12-bit segment binary key steam:

$$\mathcal{P}_i, \quad i = 1, 2, \ldots, 1000.$$

(4) Use $\{\mathcal{P}_1, \ldots, \mathcal{P}_{1000}\}$ to decrypt the ciphertext $\mathcal{C}$, and obtain the decrypted plaintext:

$$\bar{\mathcal{M}}_i = E^{-1}(\mathcal{C}, \mathcal{P}_i),\ i = 1, \ldots, 1000.$$

Table 8. The static data of the first 10 images with minimum '0's codes and the last 10 decrypted images with maximum '0's codes. Here $\mathcal{N}_1$ and $\mathcal{N}_2$ represent the number of '0's and the number of the color pixels with brightness less than 255, respectively; Mean represents the average brightness of color pixels in the decrypted images; Percentage represents the percentage of the 1 codes in the decrypted image.

Images	$\mathcal{N}_1$	$\mathcal{N}_1$	Mean	Percentage
$I_{80,1}$	80	80	254.9642	99.9835
$I_{91,1}$	91	91	254.9510	99.9812
$I_{94,1}$	94	94	254.9642	99.9806
$I_{95,1}$	95	95	254.9573	99.9804
$I_{96,1}$	96	96	254.9437	99.9802
$I_{96,2}$	96	95	254.9444	99.9802
$I_{97,1}$	97	97	254.9445	99.9800
$I_{97,2}$	97	97	254.9555	99.9800
$I_{97,3}$	97	97	254.9576	99.9800
$I_{98,1}$	98	98	254.9502	99.9798
$I_{143,1}$	143	143	254.9194	99.9705
$I_{144,1}$	144	144	254.9097	99.9703
$I_{144,2}$	144	144	254.9172	99.9703
$I_{144,3}$	144	144	254.9092	99.9703
$I_{145,1}$	145	145	254.9254	99.9701
$I_{146,1}$	146	146	254.9063	99.9699
$I_{147,1}$	147	147	254.9289	99.9697
$I_{147,2}$	147	147	254.9204	99.9697
$I_{148,1}$	148	148	254.9256	99.9695
$I_{149,1}$	149	149	254.9236	99.9693

Table 9. The differences between the original seed K_0 and the seeds $K_{j,i}$ are measured by norm $\|K_0 - K_{j,i}\|$.

	$\|K_0 - K_{j,i}\|$ $(\times 10^5)$				
	$K_{80,1}$	$K_{91,1}$	$K_{94,1}$	$K_{95,1}$	$K_{96,1}$
K_0	1.0743	1.0555	1.0844	1.0577	1.0782
	$K_{96,2}$	$K_{97,1}$	$K_{97,2}$	$K_{97,3}$	$K_{98,1}$
K_0	1.0701	1.0652	1.0801	1.0764	1.0543
	$K_{143,1}$	$K_{144,1}$	$K_{144,2}$	$K_{143,3}$	$K_{145,1}$
K_0	1.0791	1.0715	1.0588	1.0742	1.0803
	$K_{146,2}$	$K_{147,1}$	$K_{147,2}$	$K_{148,1}$	$K_{149,1}$
K_0	1.0865	1.0657	1.0561	1.0758	1.0841

(5) Change $\bar{\mathcal{M}}_i$ to RGB images.

Totally 1000 images become almost pure white colored images. There are total of 393 216 $\{0, 1\}$ codes in each decrypted image. Among the decrypted images, the minimum number of '0's in the decrypted images is 66, and the maximum one is 124. The two decrypted images (denoted by $I_{66,1}$ and $I_{124,1}$) are shown in Figures 8a and 8b, respectively. The decrypted images show avalanche effects.

Denote $I_{i,j}$ to be the jth image with number i of 0 codes. Table 10 lists the static data of the first 10 images with minimum '0's codes and the last 10 decrypted images with maximum '0's codes. Here $\mathcal{N}_1$ and $\mathcal{N}_2$ represent the number of '0's and the number of the color pixels with brightness less than 255, respectively; Mean represents the average brightness of color pixels in the decrypted images; Percentage represents the percentage the 1 codes in the decrypted image. Observe that the percentages of the numbers of "1" codes are in the rang

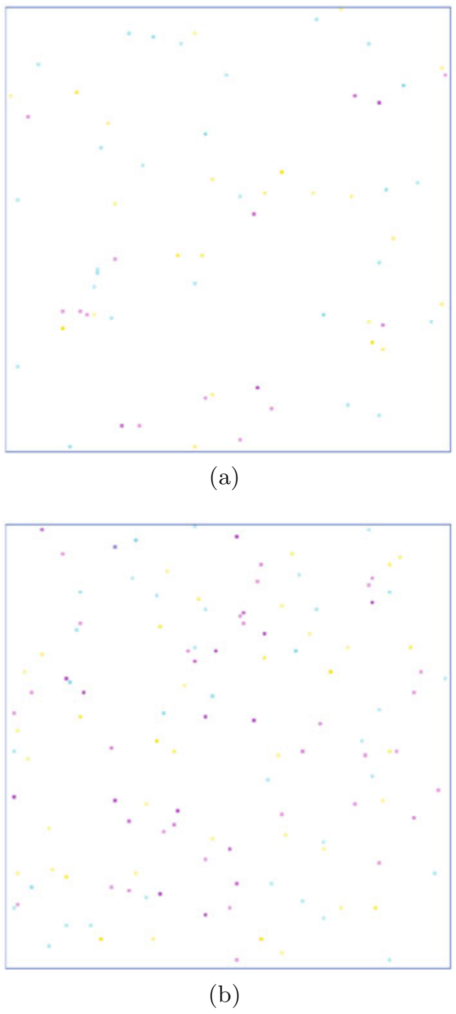


Fig. 8. (Color online) Two decrypted images via key streams generated with different key streams generated by the Matlab PRNG: (a) $I_{66,1}$, and (b) $I_{124,1}$.

[99.9685, 99.9832], which is very close to the ideal value $(2^{12}-1)/2^{12} = 99.9756$.

Table 11 lists some statistical data of the norms between the original key stream S_0 and the key stream $S_{i,j}$ used in the above ten decrypted images, respectively. The results may suggest that there are no significant correlations between the norm and the corresponding decrypted image.

In summary, the simulations show that using Matlab PRNG with SESAE to encrypt RGB images is able to generate images with significant avalanche effects.

Remark 7. Comparing Table 6 with Table 10 shows that the SESAEs generated by PRNG II and Matlab PRNG do not have significant differences.

3.4 SESAE experiments for 16-bit segment PRNGs

This subsection implements some SESAE experiments for 16-bit segment PRNGs.

Table 10. The static data of the first 10 images with minimum '0's codes and the last 10 decrypted images with maximum '0's codes. Here $\mathcal{N}_1$ and $\mathcal{N}_2$ represent the number of '0's and the number of the color pixels with brightness less than 255, respectively; Mean represents the average brightness of color pixels in the decrypted images; Percentage represents the percentage of the 1 codes in the decrypted image.

Images	$\mathcal{N}_1$	$\mathcal{N}_1$	Mean	Percentage
$I_{66,1}$	66	66	254.9565	99.9832
$I_{67,1}$	67	67	254.9606	99.9830
$I_{70,1}$	70	70	254.9349	99.9822
$I_{70,2}$	70	70	254.9552	99.9822
$I_{70,3}$	70	70	254.9633	99.9822
$I_{71,1}$	71	71	254.9550	99.9819
$I_{71,2}$	71	71	254.9606	99.9819
$I_{71,3}$	71	71	254.9503	99.9819
$I_{73,1}$	73	73	254.9504	99.9814
$I_{73,2}$	73	73	254.9610	99.9814
$I_{118,1}$	118	118	254.9347	99.9700
$I_{119,1}$	119	119	254.9260	99.9697
$I_{119,2}$	119	119	254.9139	99.9697
$I_{120,1}$	120	120	254.9154	99.9695
$I_{120,2}$	120	120	254.9328	99.9695
$I_{120,3}$	120	120	254.9395	99.9695
$I_{121,1}$	121	121	254.9283	99.9692
$I_{122,1}$	122	122	254.9305	99.9687
$I_{123,2}$	123	123	254.9286	99.9687
$I_{124,1}$	124	124	254.9113	99.9685

Table 11. The differences between the original keystream $\mathcal{S}_0$ and the keystreams $\mathcal{S}'_{j,i}s$ are measured by norm $\lVert\mathcal{S}_0 - \mathcal{S}_{j,i}\rVert$.

	$\lVert\mathcal{S}_0 - \mathcal{S}_{j,i}\rVert$ $(\times 10^6)$				
	$\mathcal{S}_{66,1}$	$\mathcal{S}_{67,1}$	$\mathcal{S}_{70,1}$	$\mathcal{S}_{70,2}$	$\mathcal{S}_{70,3}$
$\mathcal{S}_0$	1.0743	1.0555	1.0844	1.0577	1.0782
	$\mathcal{S}_{71,1}$	$\mathcal{S}_{71,2}$	$\mathcal{S}_{71,3}$	$\mathcal{S}_{73,1}$	$\mathcal{S}_{73,2}$
$\mathcal{S}_0$	1.0701	1.0652	1.0801	1.0764	1.0543
	$\mathcal{S}_{118,1}$	$\mathcal{S}_{119,1}$	$\mathcal{S}_{119,2}$	$\mathcal{S}_{120,1}$	$\mathcal{S}_{120,2}$
$\mathcal{S}_0$	1.0791	1.0715	1.0588	1.0742	1.0803
	$\mathcal{S}_{120,3}$	$\mathcal{S}_{121,1}$	$\mathcal{S}_{123,1}$	$\mathcal{S}_{123,2}$	$\mathcal{S}_{124,1}$
$\mathcal{S}_0$	1.0865	1.0657	1.0561	1.0758	1.0841

Now let the parameter L in the four PRNGs described in Sections 3.1.3, 3.1.4, 3.2.2 and 3.2.3 be equal to 16. Then one obtains four 16-bit segment PRNSs.

Using the same conditions given in the four subsections encrypts the images Flowers and Lina, and then using 1000 key streams generated by different keys decrypts the encrypted images by the original key stream, respectively. The results are given as follows.

(1) The total 1000 images (Flowers) decrypted by the key stream generated by 16-bit segment PRNG I become almost pure ones. Among the decrypted images, the minimum number of '0's in the decrypted images is 1, and the maximum one is 11. Therefore the percentages of the numbers of "1" codes are in the rang [0.99998 0.99999], which is very close to the ideal value $(2^{16} - 1)/2^{16} = 0.99998$. The two decrypted images (denoted

Fig. 9. (Color online) Two decrypted images via key streams generated with different key streams generated by the 16-bit segment PRNG I: (a) $I_{1,1}$, and (b) $I_{11,1}$.

by $I_{1,1}$ and $I_{11,1}$) are shown in Figure 9. The decrypted images show strong avalanche effects.

(2) The total 1000 images (Flowers) decrypted by the key stream generated by RC4 16-bit segment PRNG become almost pure ones. Among the decrypted images, the minimum number of '0's in the decrypted images is 1, and the maximum one number of '0's in the decrypted images is 17. Therefore the percentages of the numbers of "1" codes are in the rang $[0.99996, 0.99999]$, which is very close to the ideal value $(2^{16} - 1)/2^{16} = 0.99998$. The two decrypted images (denoted by $I_{1,1}$ and $I_{17,1}$) are shown in Figure 10. The decrypted images show strong avalanche effects.

(3) The total 1000 images (Lina) decrypted by the key stream generated by 16-bit segment PRNG II become almost pure ones. Among the decrypted images, there are ten images to become pure white ones. The maximum number of '0's in the decrypted images is 10. This image is denoted by $I_{10,1}$. Therefore the percentages of the numbers of "1" codes are in the rang [0.99998 1], which is in good agreement with the ideal value $(2^{16} - 1)/2^{16} = 0.99998$. The image $I_{10,1}$ is shown in Figure 11. In summary, the strong avalanche effects appeared in the decrypted images.

(4) The total 1000 images decrypted by the key stream generated by Matlab 16-bit segment PRNG become almost pure ones. Among the decrypted images, there are three images to become pure white ones. The maximum number of '0's in the decrypted images is 16. Therefore the percentages of the numbers of "1" codes are in the rang [0.99996, 1], which is very close to the ideal value $(2^{16} - 1)/2^{16} = 0.99998$. One decrypted image with thirteen zero codes (denoted by $I_{16,1}$) is shown in Figure 12. The decrypted images show strong avalanche effects.

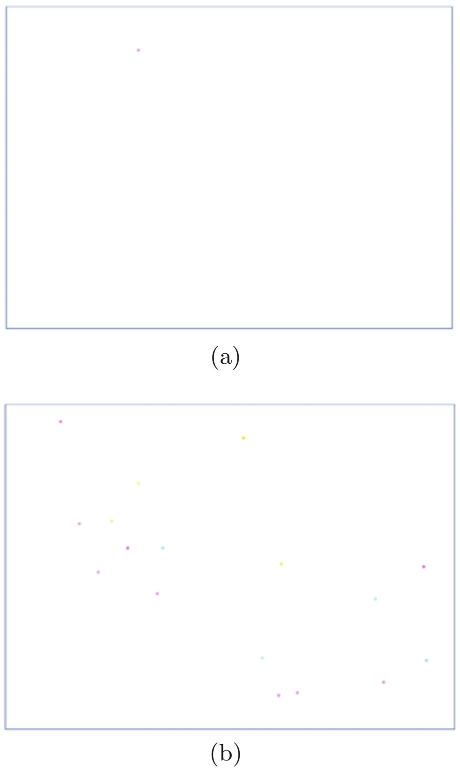


Fig. 10. (Color online) Two decrypted images via key streams generated with different key streams generated by the 16-bit segment RC4 PRNG (a) $I_{1,1}$, and (b) $I_{17,1}$.

Fig. 11. (Color online) Decrypted image $I_{10,1}$ via key streams generated with different key streams generated by the 16-bit segment PRNG II.

Fig. 12. (Color online) Decrypted image $I_{16,1}$ via key streams generated with different key stream generated by the 16-bit segment Matlab PRNG.

4 Conclusions

This paper presents a novel d-bit segment stream encryption scheme with avalanche effect (SESAE). The main feature of SESAE is to make each bit of the decrypted plaintext changed to "1" with the probability of $(2^d - 1)/2^d$ when any seed of the key stream is changed, rather than changed one-half as required by the traditional strict key avalanche criterion [10]. As a cost, the required bits of the ciphertext are d times those of the plaintext.

The definition of the ideal PRNG is introduced. An ideal PRNG should generate homogenous and different d-bit segment binary key streams with probability $(2^d - 1)/2^d$ when using different seeds.

The main advantage of SESAE is that the decrypted ciphertext will become a monotone "white" text if a key stream is used with different seeds generated by an ideal PRNG to decrypt the ciphertext. Therefore, the avalanche effect of SESAE is stronger than that of the traditional ones.

The generalized FIPS140-2 tests have demonstrated that the random performances of the two chaotic-based PRNGs are not as good as those of RC4 PRNG and Matlab PRNG. However the simulations have shown that the encrypted RGB images via the two new chaotic PRNGs, RC4 PRNG, and Maltab PRNG have the similar significant avalanche effects, which are very close to the ideal value $(2^d - 1)/2^d\%$. The results suggest that the four PRNGs are qualified candidates for SESAE.

The simulation results suggest that the SESAE requirements for PRNGs are not as strict as those under the traditional avalanche criteria. In fact, simulations suggest that SESAE needs only those PRNGs that are able to generate different key streams with approximately $(2^d - 1)/2^d\%$ bits.

The simulations also suggest that the key spaces of the two chaotic PRNGs are both larger than $10^{15\times15} > 2^{747}$, which are large enough to against brute-force attacks. Furthermore, the chaotic PRNGs are good to be used as the "one-time-pad" scheme (see Remark 3). Consequently, they are secure enough for practical applications.

The authors would like to thank the anonymous referee for valuable comments. This project is supported by the National Natural Science Foundation of China (Grant Nos. 61074192, 61170037), and the Hong Kong Research Grants Council under GRF Grant CityU 1109/12E.

References

1. A. Skrobek, Phys. Lett. A **363**, 84 (2007)
2. D. Arroyo et al., Phys. Lett. A **372**, 1034 (2008)
3. S.J. Xu et al., Phys. Lett. A **376**, 1003 (2012)
4. H. Feistel, Sci. Am. **228**, 15 (1973)
5. C.E. Shannon, Bell Syst. Tech. J **28**, 656 (1949)
6. F. Forre, The strict avalanche criterion: spectral properties of booleans functions and an extended definition, in *Advances in cryptology*, edited by S. Goldwasser, Crypto88, Lecture Notes in Computer Science, (Springer-Verlag, Berlin, 1990), Vol. 403
7. X.-M. Zhang, Y. Zheng, J. Univers. Comput. Sci. **1**, 320 (1995)
8. J.C. Castro et al., Math. Comput. Simulat. **68**, 1 (2005)
9. O.A. Logachev, A.A. Salnikov, V.V. Yashchenko, *Boolean Functions in Coding Theory and Cryptography (Translation of Mathematical Monogaphs)* (American Mathematical Society, Providence, Rhode Island, 2012), Vol. 241
10. R.J. Spillman, *Classical and Contemporary Cryptology* (Pearson Education INC, Upper Saddle River, 2005)
11. W. Stallings, *Cryptography and Network Security: Principle and Practice*, 2nd edn. (Prentice-Hall Inc, Upper Saddle River, 1999)
12. L.O. Chua, C.W. Wu, A.S. Huang, G.Q. Zhong, *IEEE Trans. Circ. Syst.* I **49**, 732 (1994)
13. L. Min, K.R. Crounse, L.O. Chua, Int. J. Bifurc. Chaos **10**, 1295 (2000)
14. H. Zang, L. Min, G. Zhao, A generalized synchronization theorem for discrete-time chaos system with application in data encryption scheme, in *Proceedings of the 2007 Int. Conf. on Communications, Circuits and Systems Kokura, Fukuoka, 2007*, Vol. II, pp. 1325–1329
15. NIST, *FIPS PUB 140-2, Security Requirements for Cryptographic Modules* (NIST, Gaithersburg, 2001)
16. S.W. Golomb, *Shift Register Sequences* (Laguna Hills, Aegean Park, 1982)
17. L. Min, L. Hao, L. Zhang, Study on the statistical test for string pseudorandom number generators, in *Advances in Brain Inspired Cognitive Systems* edited by Liu et al. (Springer-Verlag, Berlin, 2013), Vol. 7888, pp. 278–287
18. G. Chen, T. Ueta, J. Bifurc. Chaos **9**, 1465 (1999)