Generating Chaotic Stream Ciphers Using Chaotic Systems

Po-Han Lee,[1] Soo-Chang Pei,[2] and Yih-Yuh Chen[1,3]

[1]*Physics Department, National Taiwan University, Taipei, Taiwan 106, R.O.C.*
[2]*Electrical Engineering Department, National Taiwan University, Taipei, Taiwan 106, R.O.C.*
[3]*Institute of Astrophysics, National Taiwan University, Taipei, Taiwan 106, R.O.C.*
(Received April 23, 2003)

A new scheme for generating good pseudo-random numbers, based on the composition of chaotic maps, is studied. In this method, hereafter called the chaotic stream cipher, one first uses a known chaotic dynamical system to generate a sequence of pseudo-random bytes, then applies certain permutations to them, using the discretized version of another two-dimensional chaotic map. Standard statistical tests of this scheme, as well as other known chaos-based random number generators, are performed and compared. We show that this new scheme can generate a high percentage of usable pseudo-random numbers, while maintaining a large enough key space for potential use in encryptions.

PACS numbers: 05.45.Gg, 07.05.Pj

I. INTRODUCTION

The potential for incorporating chaos into cryptography has been under intensive investigation since Pecora and Carroll demonstrated the possibility of synchronization in chaotic systems [1]. For instance, a very primitive way of utilizing it is to directly hide the data in chaotic signals generated by a chaotic system [2, 3], or to combine standard cryptographic operations with chaos, as was done by He and Vaidya [4]. However, approaches along these lines are more appropriate for covert communications, because the driving signals have been put in the public channel. As it turns out, methods such as those described above are more akin to the generation of pseudo-random numbers, since the only aspect of chaos that is put into use in these systems is the generation of a sequence of presumably random numbers associated with the system variables. In view of the fact that random numbers are indispensable in applications as diverse as simulations, image encryptions, network communications via e-mails, numerical analysis, and decision making, just to name a few, and that a deterministic algorithm is a qualified pseudo-random number generator (PRNG) only if certain statistical properties are met, it is clear that a more thorough analysis of how well the supposedly random sequence generated by a chaotic system stands against the already established standard for reliable random numbers is in order. The aim of this paper is to provide a contribution toward this goal.

In fact, chaotic maps have been utilized in several ways in cryptography; the basic ideas can be classified into three major types: value transformation, position permutation, and hybrid form. Value transformation, which appears to be the most obvious application of chaotic maps, uses one or more one-dimensional maps as a pseudo-random number gen-

 © 2003 THE PHYSICAL SOCIETY OF THE REPUBLIC OF CHINA

erator to produce a binary stream, which is then XOR-ed with the plaintext to produce the ciphertext [5–7]. Although the schemes using chaotic maps to generate a binary stream seem powerful, they nevertheless have been shown to be weak when analyzed using symbolic dynamics [8–10]. The idea of position permutation, on the other hand, is to apply a two-dimensional chaotic map on a torus or a square, to scramble an underlying image, as the so-called symmetric block encryption does [11]. In the third case, one simply combines both position permutation and value transformation to try to achieve a more secure encryption. Such is the case with the algorithm of chaotic mirror-like encryption (CMLIE) [12]. The robustness of the encryption method proposed by Yen *et al.*, or any of the methods mentioned above for that matter, depends on the quality of the randomness of the bits generated by the chaotic systems used. This, again, calls for a closer inspection of how good chaos generated randomness really is.

To quote one more example, one might consider using a chaotic map to generate a random modulator in audio watermarking [13, 14]. In this scheme, the watermarking signal is generated using a key, i.e., a user supplied initial number input to the chaotic system. However, systematic analysis on the reliability of this scheme remains to be done, as of the present writing.

In view of the potential applications and the importance of having a reliable chaos-based pseudo-random number generator, we have set out to investigate its randomness, using the industry-accepted criteria. The result is a new scheme, the chaotic stream cipher, CSC, which is a hybrid form combining the concept of value transformation and the position permutation of the discrete baker map studied by Fridrich. The latter is incorporated into the scheme because, following Fridrich, we will use it as a typical random permutation to protect the random numbers generated by a chaotic system from attacks using symbolic dynamics or return maps.

The organization of this paper is as follows. After this brief introduction, we first review in Sec. II some of the known pseudo-random number generators based on various chaotic maps. Details of our proposed new scheme are also described. Standard statistical tests proposed by the National Institute of Standard and Technology of the USA [15] and a visual test using return maps are then applied to them in Sec. III, followed by a full analysis and comparison of our method and the existing ones. Several conclusions are drawn in Sec. IV.

II. PSEUDO-RANDOM NUMBER GENERATORS USING VARIOUS CHAOTIC MAPS

Utilizing chaos as a random number generator has become an important and exciting field of study in the past decade, since it was realized that one could take advantage of the intrinsic features of a chaotic system and turn them into an aperiodic sequence of random numbers. Among the various nonlinear mappings considered by researchers, the most famous is the so-called logistic map, which is one of the simplest systems exhibiting

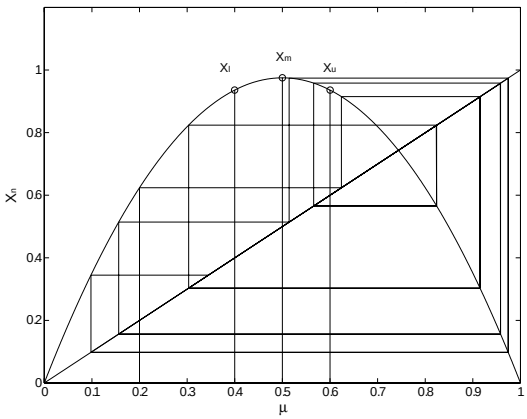

FIG. 1: The bisection method using the logistic map; x_l and x_u stand for the lower and higher cut-off limits, respectively. x_m means the midpoint.

order-to-chaos transitions:

$$x_{n+1} \equiv \mu x_n(1 - x_n), \quad x_n \in (0, 1)\,, \tag{1}$$

where $0 \leq \mu \leq 4$ is a control parameter [16, 17]. When $0 \leq \mu \leq 1$, one can easily show that all orbits converge to the single attracting fixed point at 0. Similarly, for $1 < \mu \leq 3$, all orbits are attracted to the fixed point at $x^* = 1 - \frac{1}{\mu}$. For $3 < \mu \leq \mu_\infty \simeq 3.57$, the logistic map exhibits the phenomenon of periodic doubling. The universality in this regime has been fully investigated by Feigenbaum [18]. For $\mu_\infty < \mu \leq 4$, the orbit diagram of the map reveals an unexpected mixture of order and chaos with periodic windows interspersed between chaotic clouds of dots.

Although mostly chaotic when μ is above $\mu_\infty \approx 3.57$, the logistic map is deterministic, meaning that there is always a strong correlation between successive x_n's. What this implies is that a judicious choice must be adopted, if one is to extract useful numbers from the sequence $\{x_n\}$ as candidates for random numbers. Toward this goal various methods have been proposed, of which we will discuss two below.

The first method we will discuss was proposed by Bianco *et al.* [6, 7]. Because of the nature of how the random bits are derived from the logistic map, we will call this method the "bisection method" for simplicity. In this method, one only retains the middle portion of the unit interval on which the logistic map is defined and further divides it into two subintervals: $[x_l, x_m]$ and $[x_m, x_u]$, where $x_m = 0.5$ is the mid-point, and x_l and x_u are the (user specified) lower and upper bounds, respectively. Upon iteration, the x_n might or might not fall on either interval. If it does fall on the left interval, a random bit of 0 is generated. Likewise, a random bit of 1 is generated if x_n falls on the right interval. However if it falls outside the interval $[x_l, x_u]$, then one simply ignores it and keeps on iterating until the next hit occurs. According to these authors, the method can pass various statistical tests such as the χ^2 test when $\mu = 3.9996$ and $x_0 = 0.1$. Based on this, they also used a two-stage numerical filter process to generate a sequence of floating point numbers, which

are claimed to be irreversible in the sense that the original values used to generate the numbers cannot be recovered.

The other method, where a logistic map is used as a random number generator, can be called the "periodic sampling method" [5]. In this method, one specializes to the case $\mu = 4$ and observes that the change of variable $x_i = (1 - cos\theta_i)/2 = \sin^2 \frac{\theta_i}{2}$ immediately converts the logistic map into the following:

$$\begin{aligned} \theta_{n+1} &\equiv 2\theta_i, \quad (\theta_i < \frac{\pi}{2}) \\ \theta_{n+1} &\equiv 2\pi - 2\theta_i. \quad (\theta_i > \frac{\pi}{2}) \end{aligned} \tag{2}$$

In other words, one is effectively stretching a closed circle by a factor of two then collapsing it back to the original circle. Eq. (2) clearly shows that one gets a periodic series if and only if the seed θ_0 is a rational fraction of π. Thus, starting with an irrational fraction of π, Phatak and Rao [5] observed that the correlation of the sequence θ_i, $\theta_{i+\tau}$,$\theta_{i+2\tau} \ldots$ quickly died out as τ increased from 1 to 14. These authors have therefore suggested that the correlation between successive random numbers are virtually lost if $\tau \geq 23$, remarking that $\tau > 10$ already yields a reasonable result in distribution tests.

Although we will return to the test of the statistical properties of the equivalent variable defined by

$$y_i \equiv \frac{1}{\pi} cos^{-1}(1 - 2x_i), \tag{3}$$

it is worth pointing out at this stage one major shortcoming of this approach, namely, the key space one is left with is rather limited. This is because one has to sacrifice the available degrees of freedom from the parameter μ, which is rigidly set to four at the outset.

Having briefly reviewed two related methods for chaotic random-bit extraction, we next turn to the algorithm we are proposing in this paper. The chaotic stream cipher (CSC), as it will be called hereafter, consists of three parts: (1) The adoption of a chaotic dynamical system with a judicious choice of the system parameters. (For instance, if we use the logistic map for the generation of $\{x_n\}$, then the parameter μ, the initial data x_0, and N, the desired number of iterations, comprise the key space at one's disposal.) (2) Extract the lower bits from the chaotic data $\{x_n\}$. (3) Arrange the random bits obtained in (2) into a two-dimensional gray-scale image, then perform a further permutation of the random bits using a two-dimensional chaotic map. To make the presentation clearer, however, we will only discuss aspects (2) and (3) in this section, leaving the first part for later.

In our method, the extraction of the lower bits for the logistic map is done by chopping off the leading bits after multiplying each x_n of the logistic map by some constant A. Thus, the resulting number R_n can be succinctly expressed as

$$R_n \equiv Ax_n \pmod{S}, \tag{4}$$

where $R_n \in Z^+$ and S is yet another constant.

For example, if we take $A = 10^7$, $S = 256$, $\mu = 3.9$, and $x_0 = 0.1$, then we obtain

$$\begin{aligned}
x_1 &= 3.9x_0(1-x_0) = 0.351, \\
x_2 &= 3.9x_1(1-x_1) = 0.8884161, \\
x_3 &= 3.9x_2(1-x_2) = 0.386618439717, \\
&\cdots \\
x_{99} &= 3.9x_{98}(1-x_{98}) = 0.944128122108, \\
x_{100} &= 3.9x_{99}(1-x_{99}) = 0.205725823497, \\
&\cdots
\end{aligned} \tag{5}$$

using a personal computer with a Pentium III 450 MHz CPU. For reference, the program was compiled using Borland C++ Builder 5.0, with all the variables declared as type *Double* which has a bit-length of 64 bits. When expressed in scientific notation, all the variables have a 15-digit precision. For the sake of concreteness, all the floating point numbers we use in this paper are further downgraded to only twelve digits. The associated random numbers R_n are listed below:

$$\begin{aligned}
R_1 &\equiv 10^7 x_1 \pmod{256} \equiv 240, \\
R_2 &\equiv 10^7 x_2 \pmod{256} \equiv 193, \\
R_3 &\equiv 10^7 x_3 \pmod{256} \equiv 72, \\
&\cdots \\
R_{99} &\equiv 10^7 x_{99} \pmod{256} \equiv 1, \\
R_{100} &\equiv 10^7 x_{100} \pmod{256} \equiv 42, \\
&\cdots
\end{aligned} \tag{6}$$

The reason we have chosen $S = 256$ is simple. The generated random bits will later be organized into a two-dimensional gray-scale image for further processing. Thus, bracketing the bits into bytes is a necessity.

Of course, we do not have to limit ourselves to the logistic map for the sake of generating primitive random bits. A continuous time dynamical system can just as well do the job when suitably discretized. As an example, we may consider the Lorenz equation, which was proposed in 1963 as a highly simplified model for the thermal convection of the atmosphere [19]. The Lorenz equation has the simple form

$$\begin{aligned}
\frac{dx}{dt} &= \sigma(-x+y), \\
\frac{dy}{dt} &= rx - y - xz, \\
\frac{dz}{dt} &= -bz + xy,
\end{aligned} \tag{7}$$

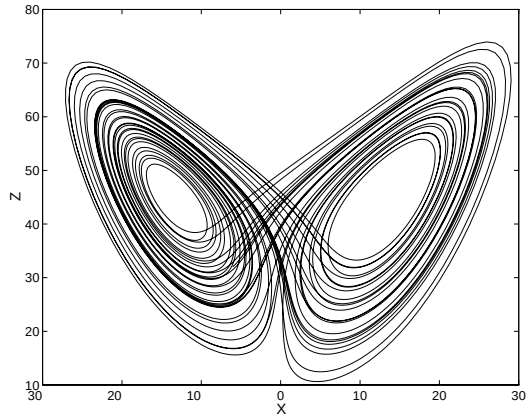

FIG. 2: The projection onto the $x - z$ plane of the Lorenz system, with $\sigma = 16$, $r = 45.6$, $b = 4.0$, $x_0 = 3.5$, $y_0 = 10.5$, $z_0 = -0.5$.

where σ, r, and b are three positive parameters. We now know that the Lorenz system is a continuous-time nonlinear dynamical system, which exhibits chaos within some special parameter regime. A detailed account of the many properties of the Lorenz system can be found in [20]. Here, we will content ourselves with the following simple facts about this system: (1) When the parameter r lies between 0 and 1, the origin (0,0,0) is globally stable. (2) The system bifurcates into two symmetric stationary points, C_1 and C_2, given by $(\pm\sqrt{b(r-1)}, \pm\sqrt{b(r-1)}, r-1)$ at $r = 1$, with C_2 lying in the region $x > 0$. (3) These stationary points remain stable up to a certain $r = r_H = \dfrac{\sigma(\sigma+b+3)}{\sigma-b-1}$. For the case shown in Fig. 2, which is the projection of the phase space trajectory onto the x-z plane for $\sigma = 16$, $r = 45.6$, $b = 4.0$, $x_0 = 3.5$, $y_0 = 10.5$, and $z_0 = -0.5$, we have $r > r_H = 33.45$, so that a strange attractor is present. Here, the numerical algorithm used is a straightforward fourth order Runge-Kutta method with a time step of $t_s = 0.005$.

In keeping with the spirit described above, it is natural to ask if we may make use of the chaos inherent in this system and adopt a similar strategy to extract useful random bits from the Lorenz equation. Thus, we consider $A|x|$, $A|y|$, $A|z|$ (mod S), where A might take on the values of 10^7 (as in the logistic map) or 65537 (as a variant to test the usefulness of the idea) and $S = 256$. This, then, is the continuous-time counterpart to our proposed bit-extraction scheme for the logistic map.

For the sake of comparison, we will also study the statistical properties of the random bits derived from other more complicated systems. Specifically, we will focus on the He-Vaidya cryptosystem [4], which was originally proposed as a master-slave type synchronous chaotic system, where the slave can be different from the master, so that a given degree of security can be achieved. A schematic representation of the system is illustrated in Fig. 3. Here, the sender uses a Lorenz system as the master. Thus, the variables of the master are governed by

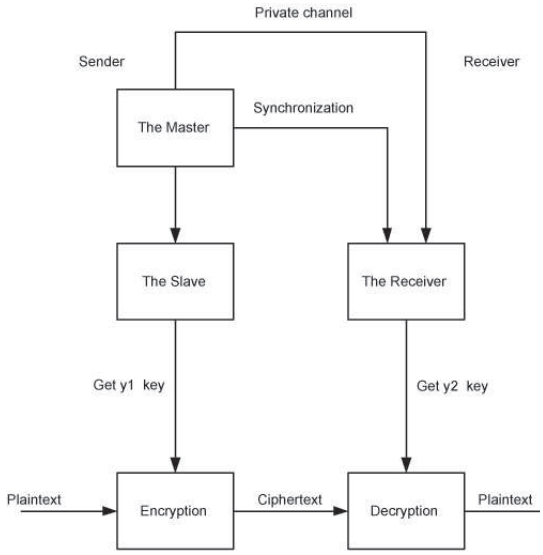

FIG. 3: The scheme of the He-Vaidya cryptosystem.

$$\begin{aligned}\frac{dx}{dt} &= \sigma(-x+y),\\ \frac{dy}{dt} &= rx-y-xz,\\ \frac{dz}{dt} &= -bz+xy,\end{aligned} \tag{8}$$

whereas the variables of the slave are

$$\begin{aligned}\frac{dx_1}{dt} &= -{x_1}^3+y_1,\\ \frac{dy_1}{dt} &= -x-x_1-8y_1.\end{aligned} \tag{9}$$

That is, the slave is driven by the master through the variable x. The secret encryption key y_1 is derived from the slave. The same driving signal is sent to the receiver, which is an exact replica of the slave on the sender's end:

$$\begin{aligned}\frac{dx_2}{dt} &= -{x_2}^3+y_2,\\ \frac{dy_2}{dt} &= -x-x_2-8y_2.\end{aligned} \tag{10}$$

This allows the extraction of the encryption key y_2, which ideally will be the same as

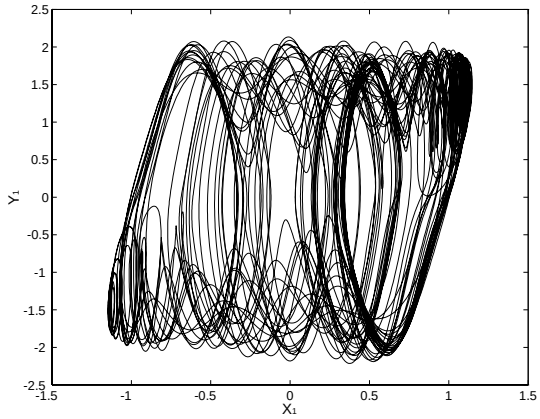

FIG. 4: The He-Vaidya chaotic map.

FIG. 5: An image of Lena and the resultant image after applying the discrete baker map once.

y_1 if the idea of chaotic synchronization works perfectly. Figure 4 shows that indeed the correlation between x_1 and y_1 is not obvious. However, our interest in this system here emphasizes more its potential as a candidate for a good random-number generator. This is because data security during the encryption process (using y_1) is an issue separate from the chaotic synchronism of this cryptosystem, and the former is intimately related to our present work. Thus, we again use modulo arithmetic to chop off the high bits of the numerical value of the variables for comparison.

Next, we proceed to describe in detail the third part of CSC, which is the further permutation of the random bits obtained from the chaotic systems described above, using a two-dimensional chaotic map. For this purpose, we adopt the baker map, which in its simplest form is defined on the unit square $[0,1]\times[0,1]$ and is described by the following formulas:

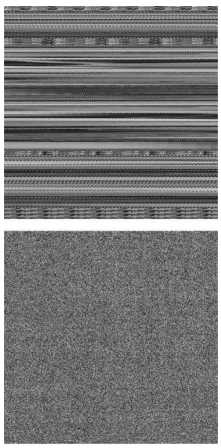

FIG. 6: Lena after two and eight successive applications of the discrete baker map.

$$\begin{aligned} B(x,y) &= (2x, y/2) \qquad \text{(when } 0 \le x < 1/2), \\ B(x,y) &= (2x-1, y/2+1/2) \qquad \text{(when } 1/2 \le x \le 1). \end{aligned} \tag{11}$$

However, to effectively scramble a given image, a more general and discrete version is needed. One method was suggested by Pichler and Scharinger [21] and discussed more fully by Fridrich [11]. In this map, one wishes to partition a digital image of $N \times N$ pixels. The way to do it is find k integers $n_1, n_2, \cdots, n_k$ such that each integer n_i divides N, while simultaneously satisfying $n_1 + n_2 + \cdots n_k = N$. Defining $N_0 = 0$, $N_i = N_{i-1} + n_i$, one can partition the original square image into k^2 rectangles. Any given pixel (r,s), with $N_{i-1} \le r < N_i$ and $0 \le s < N$, is mapped to

$$B_{n_1,n_2,\cdots,n_k}(r,s) = (q_i(r - N_i) + s \mod q_i \quad , \quad (s - s \mod q_i)/q_i + N_i), \tag{12}$$

where $q_i = N/n_i$.

For instance, the 472×472 "Lena" image can be scrambled using the ciphering key below:

$$(8 \quad 8 \quad 8 \quad 59 \quad 59 \quad 4 \quad 4 \quad 118 \quad 118 \quad 4 \quad 2 \quad 4 \quad 4 \quad 59 \quad 8 \quad 4 \quad 1). \tag{13}$$

In the above, the 17 numbers in the sequence are the divisors of 472 randomly arranged according to some given prescription. Figure 5 shows the original image and the result a simple application of the associated discrete baker map once. Figure 6 shows the results of applying the discrete baker map twice and eight times, respectively. In doing so, one can effectively take advantage of the diffusion caused by the map and avoid attacks using a return map. Specifically, we first generate the random bytes using the methods described in the previous paragraphs and then place them sequentially into each pixel of the square, to construct a grey-scale image. Then we exploit a discrete baker map to scramble the image. This permutation constitutes the last step of the CSC.

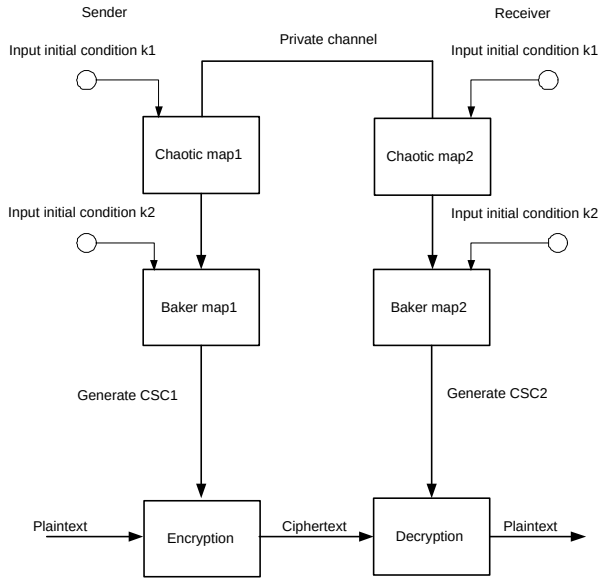

FIG. 7: The CSC, chaotic stream cipher, cryptosystem scheme.

In Fig. 7, we have shown an easy implementation of how the CSC is used to encrypt a plaintext: The sender simply uses a chaotic system to generate the random numbers, and this is followed by an application of the baker map for further permutations. The generated chaotic stream ciphers can then be used as an encryption key to encode the plaintext. The decryption on the receiver's end follows the same path.

III. TESTS ON RANDOM NUMBER GENERATORS

In order to test the *CSC* method described in Sec. II, we have performed certain statistical tests for various chaotic systems. The tests we use are the standard criteria specified in FIPS PUB 140-2 tests [15], which consist of four tests, totaling a number of 16 items. In the FIPS PUB 140-2 statistical tests of random numbers one considers a single bit stream of 20,000 consecutive bits output from the generator. The bits are then subjected to each of the tests below. Failure to meet any of the specified criteria means that the sequence must be rejected. The four tests, termed the monobit test, the poker test, the runs test, and the long run test, are briefly described below for completeness.

(1). Monobit Test:

Count the number of ones in the 20,000 bit stream. Denote this quantity by $\mathbf{X}$. The test is passed if $9,725 < \mathbf{X} < 10,275$.

(2). Poker Test:

Divide the 20,000 bit stream into 5,000 contiguous 4-bit segments. Count and store the number of occurrences of each of the 16 possible 4-bit values. Denote $f(i)$ as the number

TABLE I: The required interval for runs test in the FIPS PUB 140-2 statistical tests.

Length of Run	Required Interval
1	2315–2685
2	1114–1386
3	527–723
4	240-384
5	103–209
6+	103–209

of each 4-bit value i where $0 \leq i \leq 15$. Evaluate the following:

$$\mathbf{X} = \frac{16}{5000}\left(\sum_{i=0}^{15}[f(i)]^2\right) - 5000\,. \tag{14}$$

The test is passed if $2.16 < \mathbf{X} < 46.17$.

(3). Runs Test:

A run is defined as the maximal sequence of consecutive bits of either all ones or all zeros, which is part of a 20,000 bit sample stream. The incidences of all runs (for both consecutive zeros and consecutive ones) of all lengths(≥ 1) in the sample stream should be counted and stored. The test is passed if the number of runs that occur (of lengths 1 through 6) of each type is within the corresponding interval specified in Table 1. This must hold for both the zeros and ones; that is, all 12 counts must lie in the specified interval. For the purpose of this test, runs of greater than 6 are considered to be of length 6.

(4). Long Run Test:

A long run test is defined to be a run of length 26 or more (of either zeros or ones). For the sample of 20,000 bits, the test is passed if there are *no* long runs.

To test the quality of the random bits generated, we will have to check a total of sixteen items (one for the monobit test, one for the poker test, twelve for the runs test, and two for the long run test). Before describing the test results below, we note that a visual test of the random numbers can also serve as an aid in discriminating bad generators, even though it is hardly a quantifiable way of justifying whether a sequence of random numbers is a good candidate. This is especially true when dealing with chaotic systems, since it is known that the return map or its like (such as the Lorenz "map"), can sometimes reveal the underlying structure of the time sequence, and these features can be easily identified by a casual look at the associated pictures! Thus, besides the official tests mentioned above, we will also construct the corresponding return map to see if there is any undisclosed structure in the data set.

Now, let us try to analyze the various chaos-based random number generators. For any variable involved, the criteria are met if the total number of passing items is 16, as mentioned before; and a passing number smaller than 16 reflects the fact that the sequence has failed at least one item in the test. The test results are summarized below:

For the bisection method and the periodic sampling method, we have shown in Fig. 8

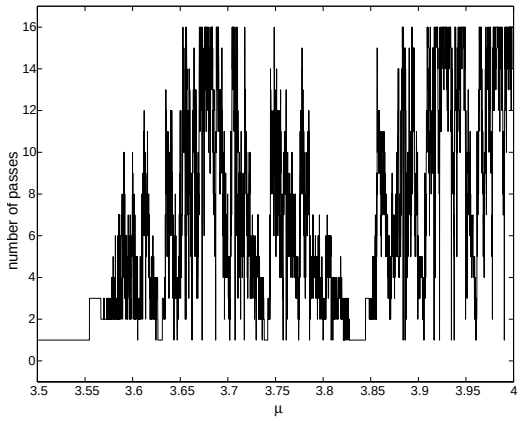

FIG. 8: The statistical test for the bisection method using the FIPS PUB 140-2 tests, in which $x_l = 0.4$, $x_m = 0.5$, $x_u = 0.6$, and the initial value $x_0 = 0.1$. The test is passed when the total number of passes is 16.

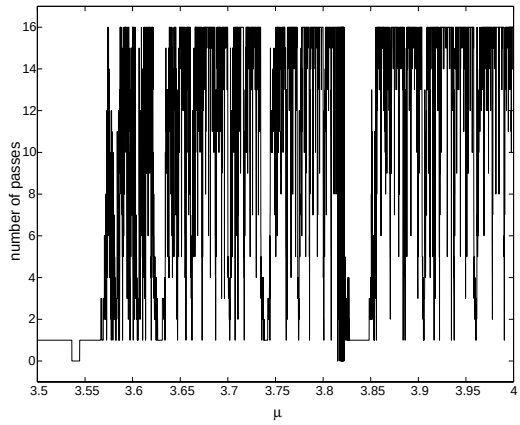

FIG. 9: The statistical test for the bisection method using the FIPS PUB 140-2 tests, with $x_l = 0.48$, $x_m = 0.5$, $x_u = 0.52$, and an initial value of $x_0 = 0.1$.

the number of passes as a function of the parameter μ, when $x_l = 0.4$, $x_m = 0.5$, $x_u = 0.6$ and the initial value of x_0 is 0.1. For the bisection method the fraction that successfully passes all tests is a poor 7.86%. If we change the values of x_l and x_u to $x_l = 0.48$ and x_u= 0.52, we get the results shown in Fig. 9. The success rate is seen to have increased to about 39.79%, but at the price of severely narrowing down the available key space. In the two experiments above, the system is arbitrarily set to a fail-state if the total number of iterations exceeds 100000 without having passed all the tests. For comparison purposes, we also tested the success rate as a function of τ for the τ shifted method, when $X_l = 0.0$, $X_m = 0.5$, $X_u = 1.0$, and the initial value $X_0 = 0.1$. ($\mu = 4.0$ is fixed in this method.) Interestingly, the random numbers thus generated all passed the tests, even when τ was as small as five. Again, however we note that this method has the narrowest key space,

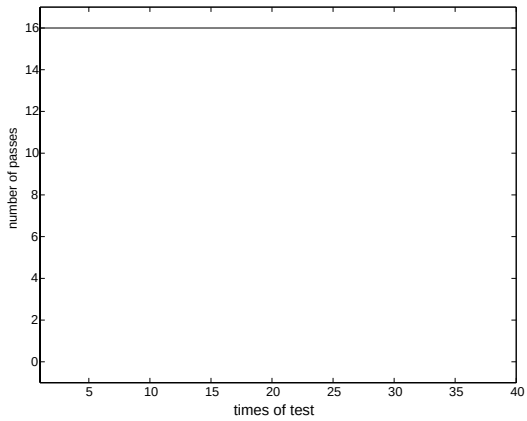

FIG. 10: The statistical test for the periodic sampling method using the FIPS PUB 140-2 tests, with $\mu = 4.0$, $S = 256$, and an initial value of $x_0 = 0.1$.

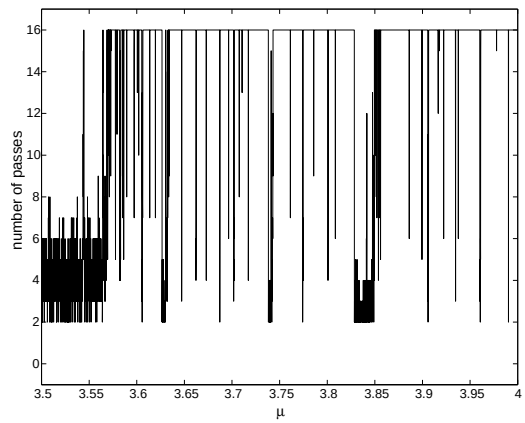

FIG. 11: The statistical test for the logistic map using the FIPS PUB 140-2 tests, in which $A = 10^7$, $S = 256$, and an initial value of $x_0 = 0.1$.

making it less suitable for encryption purpose.

However for the test of the random bits extracted from our primitive algorithm (without permutations), the results appear to be much better. For instance, the logistic map yields a passing fraction of 77.44% for $A = 10^7$, $\mu = 3.5 \sim 4.0$, $S = 256$, the initial value of $X_0 = 0.1$. Using the same parameters, but with A arbitrarily set to $A = 1801982953$, the success rate is not much altered: it is 78.14%. This is plotted in Fig. 12. As a twist, we also tried out $A = 10^{10}$, $S = 2$, which still yields a 77.28% success rate, as is shown in Fig. 13. All this suggests that our algorithm has a more or less uniform success rate (irrespective of the choice of A for the parameter regimes explored) which is at the same time significantly higher than that for the bisection method.

Employing the same tests on the Lorenz system and the He-Vaidya chaotic system, we have considered $A|x|$, $A|y|$, $A|z|$, $A|x_1|$, $A|y_1|$ (mod S). Figure 14 shows the result as

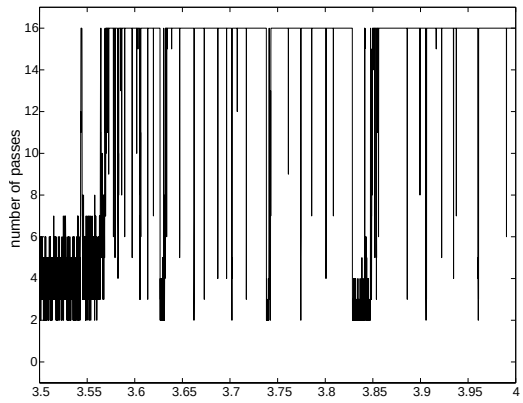

FIG. 12: Same as in Fig. 11, but with $A = 1801982953$.

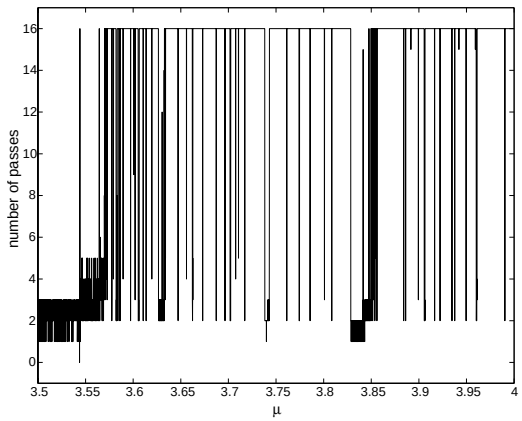

FIG. 13: Same as in Fig. 11, but with $A = 10^{10}$, $S = 2$, initial value $x_0 = 0.1$.

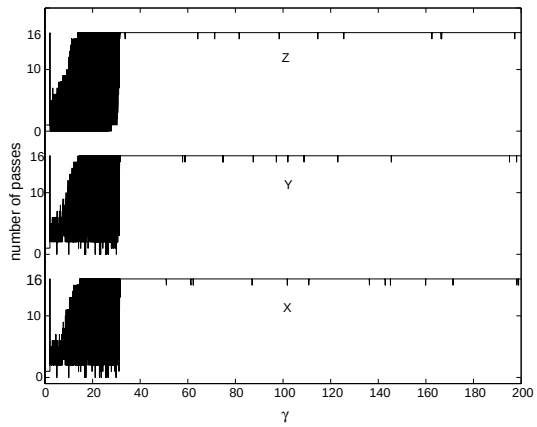

FIG. 14: The statistical test for x, y, and z of the Lorenz system using the FIPS PUB 140-2 tests, with $A = 10^7$, $S = 256$, $\sigma = 16$, $b = 4.0$, $x_0 = 3.5$, $y_0 = 10.5$, and $z_0 = -0.5$.

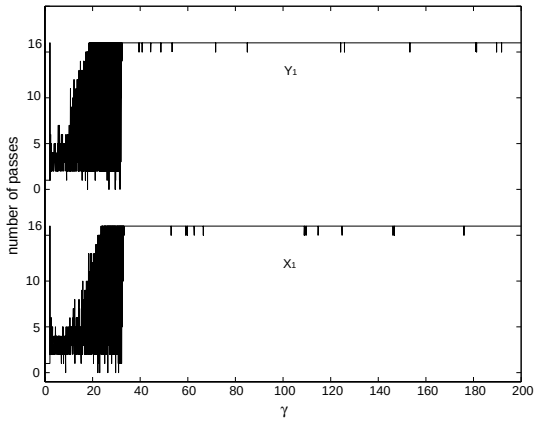

FIG. 15: The statistical test for x_1 and y_1 of the He-Vaidya system using the FIPS PUB 140-2 tests, with $A = 10^7$, $S = 256$, $\sigma = 16$, $b = 4.0$, $x_0 = 3.5$, $y_0 = 10.5$, $z_0 = -0.5$, $X_{10} = -1$, and $y_{10} = 2.0$.

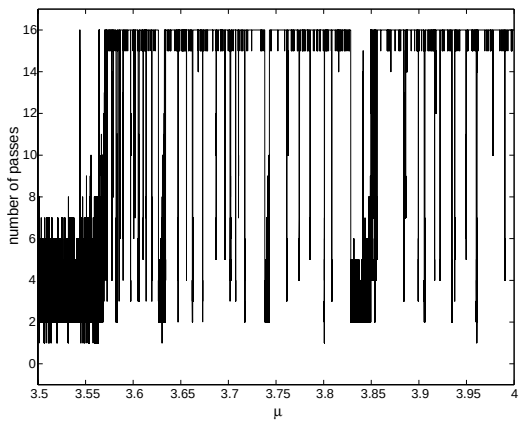

FIG. 16: The statistical test for CSC using the logistic map, with $A = 10^7$, $S = 256$, and initial value $x_0 = 0.1$. The baker map is applied only once.

a function of r for the Lorenz system, using $A = 10^7$ and $S = 256$. (For reference, the parameter values we used are: $\sigma = 16$, $b = 4.0$, $x_0 = 3.5$, $y_0 = 10.5$, $z_0 = -0.5$, $x_{10} = -1$, $y_{10} = 2.0$.) The success rate is seen to be about 86.07%, 86.18%, and 86.20% for the three variables involved, respectively. Similarly, plotted in Fig. 15 are $10^7|x_1|$, $10^7|y_1|$ (mod 256) for the He-Vaidya system. The passing fraction is also high: about 84.31% and 85.28%, respectively for the r regime considered. This implies that chopping off the higher bits is indeed a good strategy.

The same tests have been performed on the CSC. Fig. 16 shows the CSC applied to the logistic map with $A = 10^7$, $S = 256$, and the initial value of $x_0 = 0.1$. For this case, the passing fraction is about 77.10%. Here, the discrete baker map is applied only once. The passing fraction remains at the same 77.10% level if the baker map is applied five or eight times, as shown in Figs. 17–18. Moreover the passing fraction is compromised by the

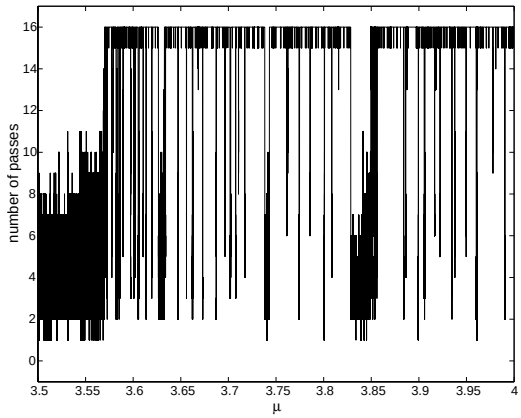

FIG. 17: Same as in Fig. 16, but with the baker map applied five times.

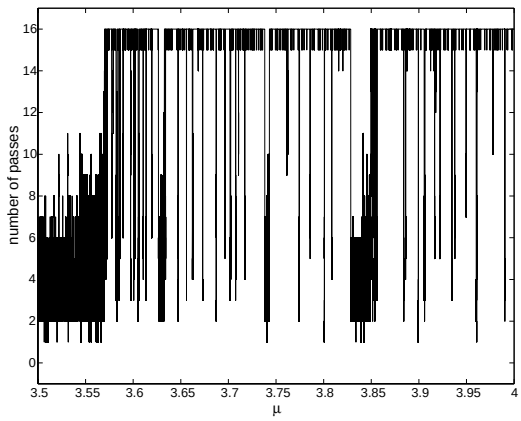

FIG. 18: Same as in Fig. 16, but with the baker map applied eight times.

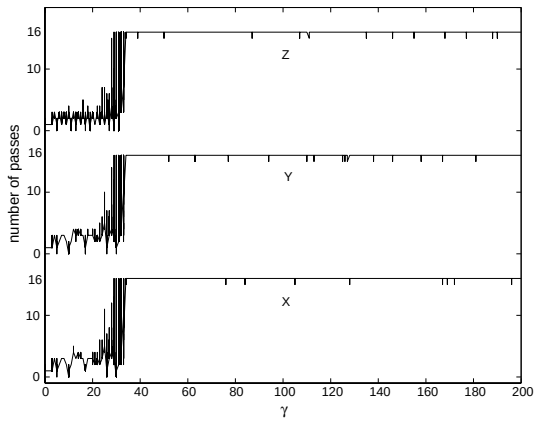

FIG. 19: The statistical test for the CSC method using the Lorenz equation, with $A = 10^7$, $S = 256$, $\sigma = 16$, $b = 4.0$, $x_0 = 3.5$, $y_0 = 10.5$, and $z_0 = -0.5$. The baker map is applied only once.

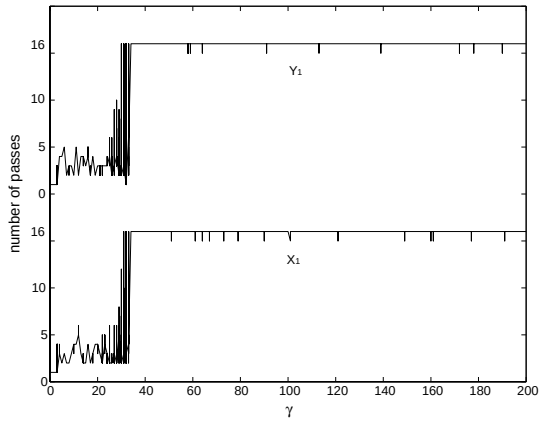

FIG. 20: The statistical test for the CSC method using the He-Vaidya system, with $A = 10^7$, $S = 256$, $\sigma = 16$, $b = 4.0$, $x_0 = 3.5$, $y_0 = 10.5$, $z_0 = -0.5$, $x_{10} = -1$, and $y_{10} = 2.0$. The baker map is applied only once.

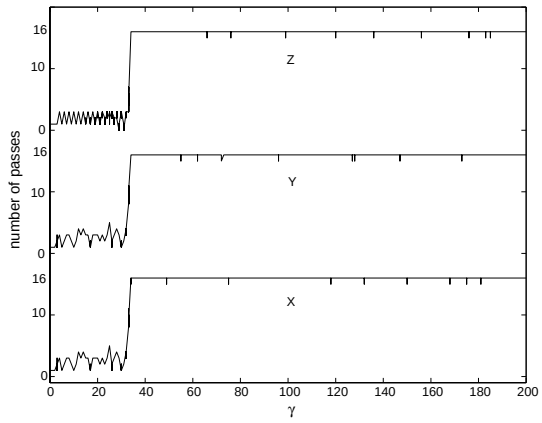

FIG. 21: Same as in Fig. 19, but with the baker map applied five times.

advantage one supposedly gains from making it less susceptible to return map-like attacks, although the reduction in the passing fraction is by no means significant.

The CSC method applied to the three variables in the Lorenz system yields a passing fraction of 83.36%, 83.35% and 83.38%, for x, y, and z, respectively. Here, the pertinent parameters are $A = 10^7$, $S = 256$, $\sigma = 16$, $b = 4.0$, $x_0 = 3.5$, $y_0 = 10.5$, and $z_0 = -0.5$, and we have applied the baker map only once. This is shown in Fig. 19.

The results for the He-Vaidya system, with $A = 10^7$, $S = 256$, $\sigma = 16$, $b = 4.0$, $x_0 = 3.5$, $y_0 = 10.5$, $z_0 = -0.5$, $x_{10} = -1$, and $y_{10} = 2.0$ yield a passing fraction of 83.13% and 83.26% for x_1 and y_1, as plotted in Fig. 20. Again, the baker map is applied only once.

Further tests, with different numbers of applied baker maps are summarized in Tables 2 through 4 and shown in Figs. 21–24. The one feature of note is that they all have basically the same passing fraction, above 83%. In other words, the passing fraction is not

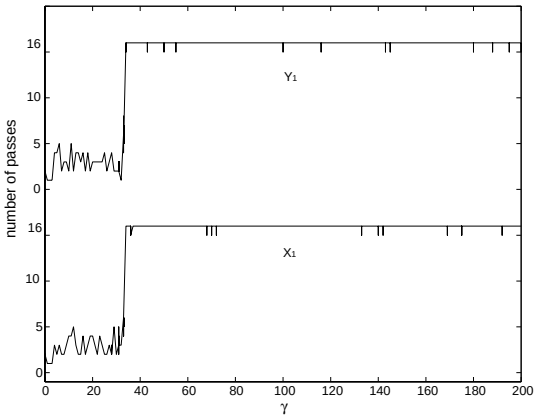

FIG. 22: Same as in Fig. 20, but with the baker map applied five times.

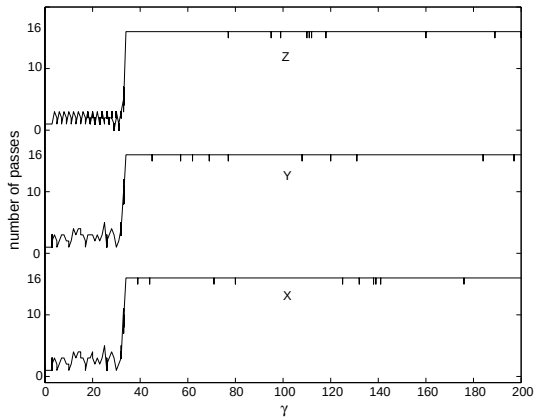

FIG. 23: Same as in Fig. 19, but with the baker map applied eight times.

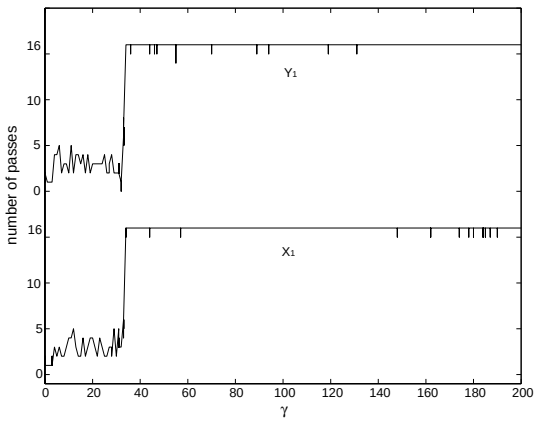

FIG. 24: Same as in Fig. 20, but with the baker map applied eight times.

TABLE II: The statistical test for various random number generators using a logistic map.

A chaotic system: the logistic map		
Method of generating random numbers	Initial conditions	Percentage of parameter regimes passing all 16 items
bisection method $3.5 \leq \mu \leq 4.0$	$x_l = 0.4$, $x_m = 0.5$, $x_u = 0.52$, $x_0 = 0.1$ $x_l = 0.48$, $x_m = 0.5$, $x_u = 0.6$, $x_0 = 0.1$	7.86 % 39.79 %
periodic sampling method $\mu = 4.0$, $\tau \geq 5$	$x_0 = 0.1$, test times = 40	100%
CSC without bit permutations $3.5 \leq \mu \leq 4.0$	$x_0 = 0.1$, $A = 10^7$, $S = 256$ $x_0 = 0.1$, $A = 1801982953$, $S = 256$ $x_0 = 0.1$, $A = 10^{10}$, $S = 2$	77.44% 78.14% 77.28%
CSC method $3.5 \leq \mu \leq 4.0$ N=number of times baker map is applied	$x_0 = 0.1$, $A = 10^7$, $S = 256$, $N = 1$ $x_0 = 0.1$, $A = 10^7$, $S = 256$, $N = 5$ $x_0 = 0.1$, $A = 10^7$, $S = 256$, $N = 8$	77.10% 77.10% 77.10%

TABLE III: The statistical test for various random number generators using the Lorenz equation.

A chaotic system: the Lorenz system				
Method of generating random numbers	Initial conditions			Percentage of parameter regimes passing all 16 items
CSC without bit permutations	$A = 10^7$, $S = 256$ $x_0 = 3.5$, $y_0 = 10.5$, $z_0 = -0.5$ $0 \leq r \leq 200$		x	86.07%
			y	86.18%
			z	86.20%
CSC method N=number of times baker map is applied	$A = 10^7$, $S = 256$ $x_0 = 3.5$, $y_0 = 10.5$, $z_0 = -0.5$ $0 \leq r \leq 200$	N=1	x	83.36 %
			y	83.35%
			z	83.38%
		N=5	x	83.03%
			y	83.04%
			z	83.03%
		N=8	x	83.03%
			y	83.03%
			z	83.03%

only high but seems rather uniform and robust for the parameter regimes we have explored.

Finally, we also did a return map analysis for visual testing. Because simple chaotic systems, such as the logistic map, are deterministic and can be defined by simple formulas, the correlation between successive iterates is strong. That is why a return map of the kind x_n versus x_{n+1} usually yields useful information about the structure of the system.

TABLE IV: The statistical test for various random number generators using a He-Vaidya map.

A chaotic system: the He-Vaidya system				
Method of generating random numbers	Initial conditions		Percentage of parameter regimes passing all 16 items	
CSC without bit permutations	$A = 10^7$, $S = 256$ $x_0 = 3.5$, $y_0 = 10.5$, $z_0 = -0.5$ $x_1 = -1$, $y_1 = 2$ $0 \leq r \leq 200$		x_1	84.31%
			y_1	85.28%
CSC method N=number of times baker map is applied	$A = 10^7$, $S = 256$ $x_0 = 3.5$, $y_0 = 10.5$, $z_0 = -0.5$ $x_1 = -1$, $y_1 = 2$ $0 \leq r \leq 200$	N=1	x_1	83.13%
			y_1	83.26%
		N=5	x_1	83.02%
			y_1	83.02%
		N=8	x_1	83.00%
			y_1	83.03%

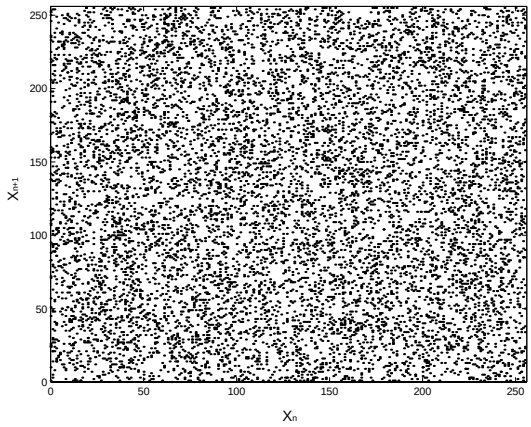

FIG. 25: The return map of the low bit of z_n extracted from the logistic map, with $A = 10^7, \mu = 3.9$, $S = 256$.

However, this is no longer the case once we have chopped off the higher bits. This is illustrated in Fig. 25. The same is true with the Lorenz system and the He-Vaidya system, as Figs. 26-27 show.

Of course it is not unreasonable to turn this around and assume that the CSC can be used similarly to encrypt a meaningful image by itself. As an example, we use the chaotic bits generated by the Lorenz system to encode the image of Lena and then apply the baker map on the resultant image for further scrambling. The result is shown in Fig. 28, which does not seem to possess any visible trace of the objects in the original picture.

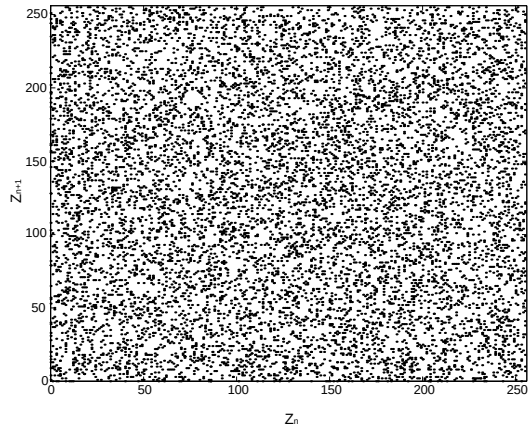

FIG. 26: The return map of the low bit of z_n extracted from the Lorenz equation, with $A = 10^7$, $S = 256$, $\sigma = 16$, $b = 4.0$, $\gamma = 40.0$, $x_0 = 3.5$, $y_0 = 10.5$, and $z_0 = -0.5$.

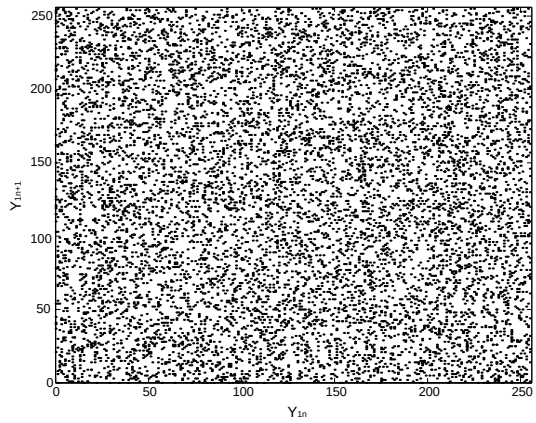

FIG. 27: The return map of the low bit of y_{1n} extracted from the He-Vaidya system, with $A = 10^7$, $S = 256$, $\sigma = 16$, $b = 4.0$, $\gamma = 40.0$, $x_0 = 3.5$, $y_0 = 10.5$, $z_0 = -0.5$, and $x_{10} = -1$.

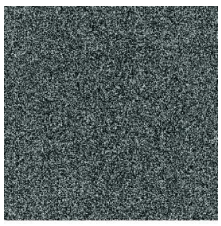

FIG. 28: The image of Lena after applying CSC with the Lorenz equation to encode the original image, with $A = 10^7$, $S = 256$, $\sigma = 16$, $b = 4.0$, $\gamma = 40.0$, $x_0 = 3.5$, $y_0 = 10.5$, and $z_0 = -0.5$.

IV. CONCLUSION

We have shown, using standard criteria for admissible random numbers, that the chaotic stream cipher proposed in the present work has the advantage of being able to generate a high percentage of usable random numbers while maintaining a large enough key space. It is also fast, in the sense that it can effectively exploit the intrinsic chaos of simple deterministic systems. However we have to admit that our method still suffers from a drawback that is also inherent in most chaotic systems, namely, the distribution of the admissible random numbers as a function of the parameters involved exhibits a fractal structure, and thus cannot be foretold from one simple glance at the chosen values for the parameters. Issues such as this still remain to be answered.

Acknowledgments

This work was supported by the National Science Council of the Republic of China under grant numbers NSC90-2112-M-002-020 and NSC91-2112-M-002-021.

References

[1] L. M. Pecora and T. Caroll, Phys. Rev. Lett. **64**, 821 (1990).
[2] G. Perez and H.A. Cerdeira, Phys. Rev. Lett. **74**, 1970 (1995).
[3] Yih-Yuh Chen, Europhys. Lett. **34**, 245 (1996).
[4] R. He and P. G. Vaidya, Phys. Rev. E, **57**, 1532 (1998).
[5] S. C. Phatak and S. S. Rao, Phys. Rev. E **51**, 3670 (1995).
[6] M. E. Bianco and D. A. Reed, *Encryption System Based on Chaos theory*, US Patent No. 5048086, Sept. 10.A, 1991.
[7] M. E. Bianco and G. L. Mayhew, *High Speed Encryption System and Method*, US Patent No. 5365588, Nov.15, 1994.
[8] J. Fridrich and J. Geer, J. Appl. Math. Comp. **71**, 227 (1995).
[9] J. Fridrich and J. Geer, J. Appl. Math. Comp. **80**, 129 (1995).
[10] J. Fridrich, J. Appl. Math. Comp. **83**, 181 (1997).
[11] J. Fridrich, Int. J. Bifurcation and Chaos **8**, 1259 (1998).
[12] J. C. Yen and J. I. Guo, Pattern Recognition and Image Analysis **10**, 236 (2000).
[13] L. Boney, A. H. Tewfik, and K. N. Hamdy, IEEE Proceedings of Multimedia, 473-480 (1996).
[14] P. Bassia, I. Pitas, and N. Nikolaidis, IEEE Transactions on multimedia **3**, 232 (2001).
[15] National Institute of Standard and Technology and Communication Security Establishment, *Derived Test Requirement (DTR) for FIPS PUB 140-2, Security Requirements for Cryptographic Modules*, available at URL:http//www.nist.gov/cmvp.
[16] T. Habutsu, Y. Nishio, I. Sasase, and S. Mori, "A secret key cryptosystem by iterating a chaotic map ," in *Advances in Cryptology* - EUROCRYPT'91, ed. Davies, D. W., LNCS 547 (Springer-Verlag, Berlin), pp. 127-140 (1991).
[17] M. Andrecut, Int. J. of Modern Phys. B **12**, 921 (1998).
[18] M. J. Feigenbaum, J. Stat. Phys. **19**, 25 (1978).
[19] E. N. Lorenz, J. Atoms. Sci. **20**, 130 (1963).

[20] C. Sparrow, "*The Lorenz Equations: Bifurcations, Chaos, and Strange Attractors,*" Springer–Verlag, 1982.

[21] F. Pichler and J. Scharinger, "Ciphering by Bernoulli shifts in finite Abelian groups," in *Contributions to General Algebra, Proc. Linz-Conference*, pp. 465-476 (1994).