Improvement Pseudo-Random Bit Generator Based on Hybrid Chaotic 1D Logistic Maps and 2D Hénon system

Asst. Prof, Dr. Hala B.Abdul Wahab
Computer Sciences Department
University of Technology
Baghdad, Iraq
hala_bahjat @yahoo.com

Ph.D Student Sundus I. Mahdi
Computer Sciences Department
University of Technology
Baghdad, Iraq
Sundus_mahdi@yahoo.com

Abstract

A pseudo-random number generator (PRNG) or pseudo-random bit generator (PRBG) is a program created a sequence of numbers that can mimic sequences of random numbers. The PRNGs are valued for their practical speed and reproducibility. In this paper PRNG is presented based on combining two chaotic one dimensional (1D) logistic maps and two dimensional (2D) Hénon system has been improved. The algorithm generates at each iteration sequences of 128 bit-sequences based on random initial seeds and control parameter for the hybrid chaotic design. The generator is assessed by a randomness test called Federal Information Processing Standard (FIPS 140-1) which check the randomness of string with length of (20000) bits. The created sequences bits hold a good randomness statistical properties which are demonstrate from the results of the test. These string of bits is suitable for cryptographic applications, therefore it applied on digital speech encryption as a mask key. The results show that it works very well on encrypted the signal through de-correlated the speech samples, and with a high quality for the recovered speech signal through the visualization of the signal.

Index Terms: Chaotic maps, Randomness, Statistical Tests, and Pseudo Random Number Generator.

الخلاصه

مولد شبه الرقم العشوائي او مولد شبه البت العشوائي , هو عباره عن خوارزميه لتوليد سلسله من الاعداد لها خصائص قريبه من الاعداد العشوائيه . يعتبر مولد الاعداد شبه العشوائيه مهما في التطبيق لما يتمتع به من خاصيتين السرعه والتجديد . يعرض في هذا البحث مولد الاعداد العشوائيع وهو مبني على ربط خارطه فوضى ذات بعد واحد Logistic مع خارطه فوضى ذات بعدين Hénon . في كل دوره للخوارزميه التي تولد 128سلسله من البت معتمد على قيمه عشوائيه ادخلت الى المولد كبذره . قيم انجاز المولد من خلال التحليل الاحصائي واوضحت النتائج بان سلسله البت المولده لها عشوائيه جيده ومستوى امني جيد للبيانات مما يؤهلها لتكون مناسبه لاستعمالها بتطبيقات التشفير .وطبقت على تشفير الصوت. واوضحت النتائج بانه كانت فعاله على تفكيك الصوت في عمليه التشفير مع المحافظه على نوعيه الصوت المسترجع.

1. Introduction

The pseudo-random bits or numbers play a role in the gaming industry, statistical mechanics, and cryptography [1]. A generator is a program used to generate those “pseudo-random” numbers. PRNG programs have the ability to generate long sequences of numbers that fit multiple aspects of randomness. While periodical, these numbers can undergo rigorous statistical testing and have the advantage of simple implementation [2].

The chaos theory is one of the most challenge manner for generators design [3]. Chaos theory basically emphases on these systems that are frequently very simple to define, nevertheless whose changing aspects looks to be very disordered. Different pseudo-random sequence will constantly produce from a different seed which include initial condition and the control parameter. The core gains here are speed and reproducibility, in addition to lower requirement for resource intensity.

Furthermore, several pseudo-random number generators have been successfully developed through this last decade [4][5][6]. Still, assessing the randomness level and the global security of the generator is necessary to be hard analysis.

The presented paper improve PRBG by applying a hybrid chaotic system from combining Logistic map 1D with Hénon chaotic system 2D. The two chaotic logistic maps and Hénon are combined to generate of 128 random bits sequences at each iteration. An efficient algorithm based on XOR operation to annihilate the collision problem that may appear while using the function, and permutations among bits produced by the output from the hybrid chaotic maps. Various statistical tests are applied on the generated pseudo-random sequences and have positively accepted. The algorithm of the generator is an efficient since it has: high level of randomness, high initial seed value sensitivity, and good throughput.

This paper is partitioned in multiple sections. Section 2 presents an overview of the used chaotic map. Section 3 discusses all the detailed description of the key generation algorithm. Section 4 demonstrates the randomness tests. Security analysis and possibly vulnerabilities are discussed in Section 5.Conclusion is expressed in section 6.

2. Chaotic Theory

The control parameters and initial conditions for the formula of the chaotic systems are very sensitive to any slight change in their values, therefore the chaotic system are characterized by their high sensitivity and some properties like ergodicity, pseudo-random behavior. Chaotic systems are suitable for PRNG or PRBG due to their sensitive initial conditions.

Chaos can be defined by some special characteristics [7][8]:

- **Unpredictable and nonlinear:** this means they are sensitive to the preliminary state of affairs, where two randomly closed initial with slight change in the starting point can lead to significant different trajectories.
- **Topological mixing:** system will progress in time so that any certain range of the state will represent as mixing colored dyes, states are always converted or overlaps with any other certain range.
- **Randomness**: This means that the system seems to be random and disorderly but in fact it is not. Beneath the random behavior, there is a sense of order and pattern.
- **Dense periodic orbits:** It explains that the chaotic system never repeats itself in an orbit, and does not ever have the same orbit.

- **Ergodicity:** This means that the dynamics system gives the same statistics when computed over time or space.

2.1 Logistic map

A chaotic logistic map is defined by:

$$F(X) = r.\mathrm{X}(1 - X) \qquad 2.1$$

where r is the control parameter and it's value is between 3.57 and 4.0 [9], X is the value for the initial condition, and F(X) is the value for the next state of X. Many generator based on Logistic map to generate the pseudo-random number. The highly chaotic case can obtained where the value of r is close to 4.0. The chaotic logistic map is used under the iterative form:

$$X_{n+1} = r.X_n.(1 - X_n) \quad \forall \mathrm{n} \geq 0 \qquad 2.2$$

where X_0 the initial seed, is a real number ϵ [0, 1], and at the same time all the elements which are obtained as output X_{n+1} is limited by the same interval [0,1], n is any integer number greater than zero.

- In the case when $r\epsilon[0, 3)$ takes this range, the output sequences come back to the same output after several iterations without any chaotic behavior.
- In the case when $r\epsilon[3, 3.57)$ the system appears periodicity.
- In the case when $r\epsilon[3.57, 4)$, it becomes chaotic with no periodicity.
- It has small key space.

2.2 Hénon Chaotic System

The Hénon map has a discrete-time scale n=1, 2,…(i.e. it is a map) system. The Hénon is two dimensional real planes and has two control parameters (a, b). Here, chaotic behavior is due to a dynamic system. The Hénon map takes a point (X_n, Y_n) in the plane and maps to a new point X_{n+1}, Y_{n+1}. Changing parameters a, b can result in many possibilities. Where a = 0.3, b ∊ [1.07, 1.4], the system is chaotic [5][10]. Hénon Chaotic equation system is :

$$Y_{n+1} = 1 - a.Y_n^2 + Z_n \qquad 2.3$$

$$Z_{n+1} = b \cdot Y_n \qquad 2.4$$

3. Key Generation Proposal (KGP)

The main indication of the proposed PRBG is to combine two different chaotic which are logistic maps and Hénon system. From this hybrid chaotic the combination between logistic maps and Hénon will generate a 128 sequences bits at each iteration. The algorithm based on xor operation to annihilate the collision problem that may appear while using the hybrid chaotic, and permutations among bits produced by the output from the hybrid chaotic system. Figure (1) shows the block diagram for the generator.

All the details for the proposed algorithm is expressed in pseudo code 1 as shown in Figure (2).

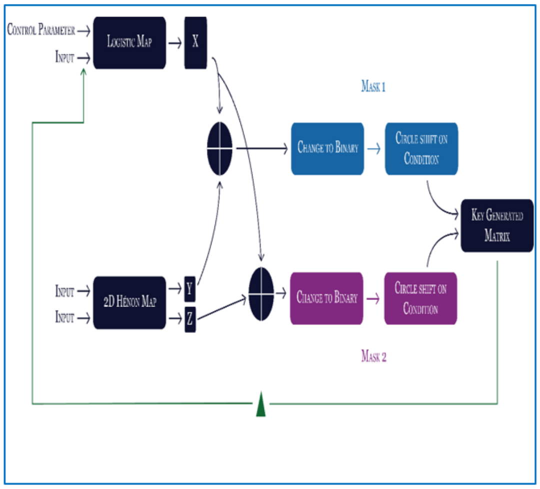
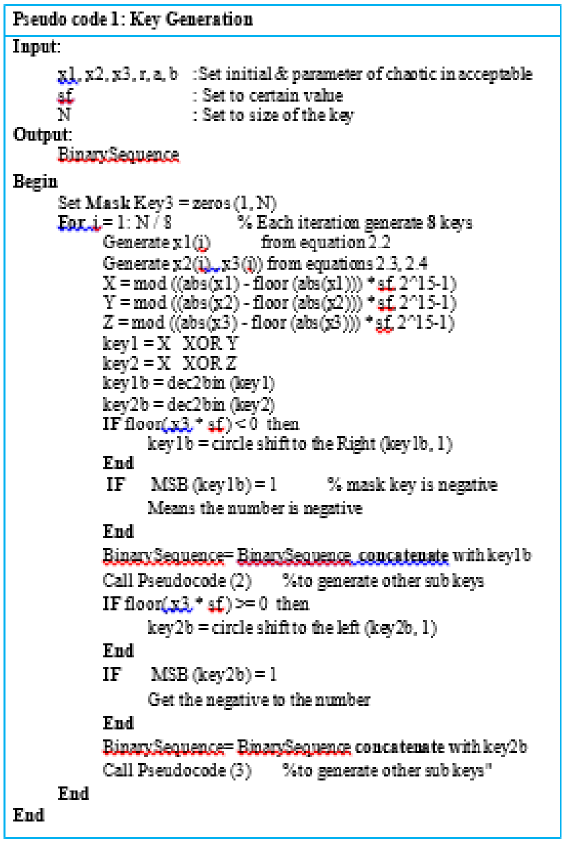


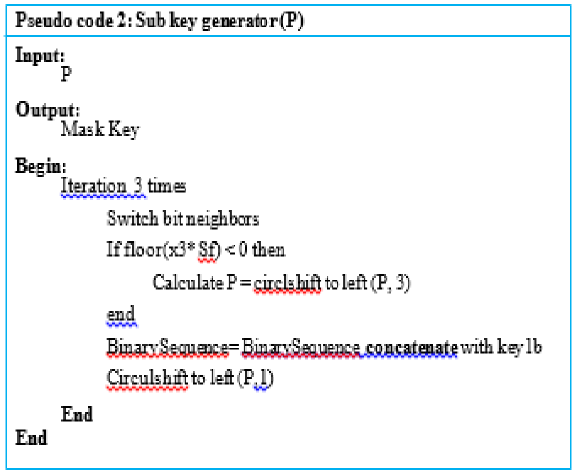
Figure (1) Mask Key Generation Design

Where X, Y, Z are the output from the chaotic, which mixed by using xor operation to generate the Mask1 and Mask2. Then each Mask1 is converted to binary and permuted according to certain condition to generate different sixteen bits sequences.



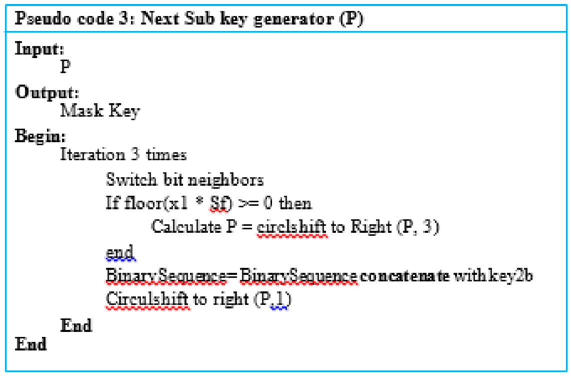
```
Pseudo code 1: Key Generation
Input:
        x1, x2, x3, r, a, b   :Set initial & parameter of chaotic in acceptable
        sf                    : Set to certain value
        N                     : Set to size of the key
Output:
        BinarySequence
Begin
        Set Mask Key3 = zeros (1, N)
        For i = 1: N / 8            % Each iteration generate 8 keys
                Generate x1(i)          from equation 2.2
                Generate x2(i) , x3(i)) from equations 2.3, 2.4
                X = mod ((abs(x1) - floor (abs(x1))) * sf, 2^15-1)
                Y = mod ((abs(x2) - floor (abs(x2))) * sf, 2^15-1)
                Z = mod ((abs(x3) - floor (abs(x3))) * sf, 2^15-1)
                key1 = X   XOR Y
                key2 = X   XOR Z
                key1b = dec2bin (key1)
                key2b = dec2bin (key2)
                IF floor( x3 * sf ) < 0  then
                        key1b = circle shift to the Right (key1b, 1)
                End
                 IF    MSB (key1b) = 1          % mask key is negative
                        Means the number is negative
                End
                BinarySequence= BinarySequence  concatenate with key1b
                Call Pseudocode (2)      %to generate other sub keys
                IF floor( x3 * sf ) >= 0  then
                        key2b = circle shift to the left (key2b, 1)
                End
                IF     MSB (key2b) = 1
                        Get the negative to the number
                End
                BinarySequence= BinarySequence concatenate with key2b
                Call Pseudocode (3)      %to generate other sub keys"
        End
End
```

Figure (2) Key Generation



```
Pseudo code 2: Sub key generator (P)
Input:
        P
Output:
        Mask Key
Begin:
        Iteration 3 times
                Switch bit neighbors
                If floor(x3* Sf) < 0 then
                        Calculate P = circlshift to left (P, 3)
                end
                BinarySequence= BinarySequence concatenate with key1b
                Circulshift to left (P,1)
        End
End
```

Figure (3) Sub Key Generation



```
Pseudo code 3: Next Sub key generator (P)
Input:
        P
Output:
        Mask Key
Begin:
        Iteration 3 times
                Switch bit neighbors
                If floor(x1 * Sf) >= 0 then
                        Calculate P = circlshift to Right (P, 3)
                end
                BinarySequence=BinarySequence concatenate with key2b
                Circulshift to right (P,1)
        End
End
```

Figure (4) Next Sub Key Generation

The basic intent is to take a binary sequence of length 16 and permutated it to a new sequence in such a way that an adversary cannot efficiently distinguish between output sequence of PRBG and truly random sequence.

4. Randomness basic tests

Let S = S_0, S_1, S_2,... , S_{n-1} be a binary sequence of length n. Here are some statistical tests to determine whether the sequence carries truly random characteristics.

FIPS 140-1 [11] is a multiple statistical tests were executed on PRBG and outcomes were recorded in Table 1.

FIPS 140-1 recommends monobit, poker, runs and long-run tests. Each test is a two-sided test where a test-statistic is required to lie within an interval.

The expected results and obtained results are demonstrated in the table below.

Table 1: Results of standard test on PRBG 20000 bit stream

Test	Expected Results	Obtained Results
Monobit Test (Number on ones in 20000 bit stream)	9654- 10346	10022
Poker Test (a fraction of the number of occurrences of 16 bit 4-bits numbers in 20000 bit stream	1.03- 57.4	5.85
Run Test (Successive occurrences of 1s and 0s in 20000 bit stream)	Length 1: 2267- 2733 2: 1079- 1421 3: 502- 748 4: 223- 402 5: 90- 223 6: 90- 223	2350 1231 511 306 150 93
Long Run Test (Runs longer than 34)	0	0

The produced pseudo-random sequences have successfully passed the various statistical tests as monobit, poker, run, and long run test.

5. Experimental test

The PRNG which correspond to the PRBG has a chaotic behavior as shown in Figure (5).

The PRBG has good statistical randomness so is being applied in speech encrypting system [10] and the experimental results for the whole encryption system is expressed in Figure (6) for speech signal using both waveform and spectrogram .

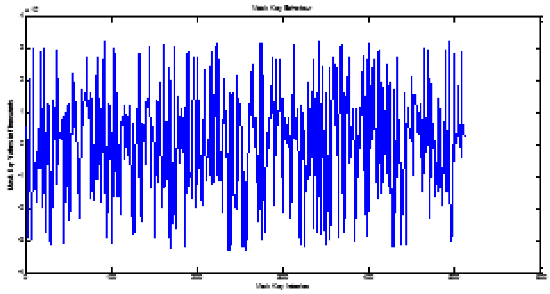


Figure (5) PRNG Behavior

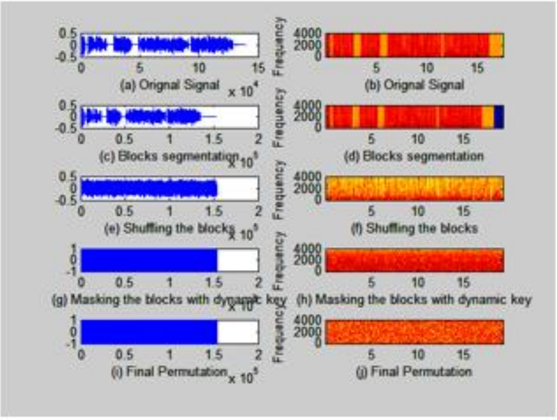


Figure (6) Encryption system for Speech signal using the generated key

It is clear that the PRBG is very effective so the encrypted signal is like a noise, by listening to it, there is nothing to understand. Since the speech encryption system is symmetric so, the same PRBG are applied for the decryption and the quality for the recovered speech is very clear as the original speech as express in Figure (7).

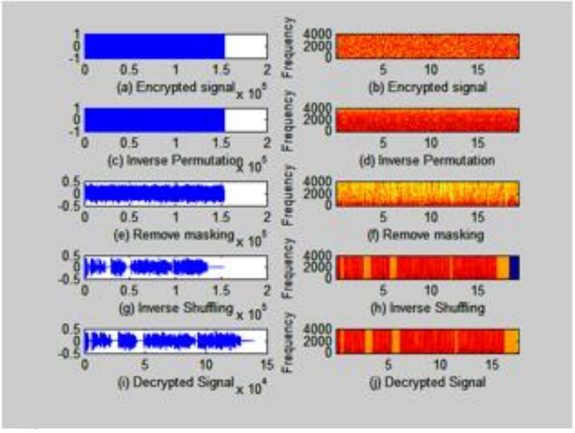


Figure (7) Decryption system for Speech signal using the generated key

6. Conclusion

A pseudo-random bit generator based on the combination of two chaotic 1D logistic map and 2D Hénon system was presented. For three given initial seeds and two control parameters, the generator produces a sequence formed of 16-bit sequences. The treatment combines xor operation and permutations on the 16 bits of the elements obtained by the generator. Such a PRBG has shown its ability to produce a very large number of pseudo-random sequences which can be useful in several cryptographic applications. The advantages of the generator are: a high sensitivity related to the initial seed values, a high randomness level, the resistance against several attacks and the rapidity of the algorithm.

Chaotic sequences have several good properties including ease of key generation, sensitive dependence on the initial condition and parameter values. Applying chaos plays an active role in the improvement of the quality of PRNGs and, gives a great contribution to improve the security of data due to the excellent properties of chaotic sequences as disordered behavior and its unpredictability.

References:

1. Pareek, Narendra K and Vinod Patidar, Krishan K Sud. "A Random Bit Generator Using Chaotic Maps". International Journal of Network Security, Vol. 10, No. 1, pp. 32–38, 2010.

2. Rütti, Mario "A Random Number Generator Test Suite for the C++ Standard", Institute for theoretical physics ETH Zurich, Diploma Thesis March 10, 2004.

3. Alvarez, Gonzalo and Shujun Li. "Some Basic Cryptographic Requirements for Chaos-Based Cryptosystems". International Journal of Bifurcation and Chaos, Vol. 16, No. 8, pp. 2129-2151, 2006.

4. Elsherbeny, Mohamed Nageb and Mahmud Rahal. "Pseudo – Random Number Generator Using Deterministic Chaotic System". International Journal of Scientific & Technology Research Vol. 1, No. 9, Oct. 2012.

5. Patidar, Vinod and K. K. Sud "A Pseudo Random Bit Generator Based on Chaotic Logistic Map and its Statistical Testing". Informatics 33, pp. 441–452, 2009.

6. Sadkhan, Sattar B and Rana Saad Mohammed. "Proposed random unified chaotic map as PRBG for voice encryption in wireless communication". Procedia Computer Science 65, pp. 314 – 323, May 2015.

7. Carmen, Pellicer-Lostao and López-Ruiz Ricardo. ″ Notions of Chaotic Cryptography: Sketch of a Chaos based Cryptosystem″ University of Zaragoza, Spain, 2012.

8. Su, Zhaopin , Guofu Zhang, and Jianguo Jiang. "Multimedia Security: A Survey of Chaos-Based Encryption Technology" China, 2012.

9. Al-saad, Saad N. and Eman H. Hashim, ″A Speech Scrambler Algorithm Based on chaotic system" Vol. 24, No 5, 2013.

10. Abdul Wahab, Hala Bahjat and Sundus I. Mahdi, "Speech Encryption Based on Wavelet Transformation and Chaotic Map" Eng. & Tech. Journal, Vol.34, Part (B), No.5, 2016.

11. **F**ederal **I**nformation **P**rocessing **S**tandards **P**ublication 140-1, "Security requirements for cryptographic modules", 1995.