A Chaotic Neural Network Based Cryptographic Pseudo-Random Sequence Design

Prateek Singla, Payal Sachdeva
Department of Computer Engineering,
Faculty of Engineering and Technology,
Jamia Millia Islamia, New Delhi-110025, India

Musheer Ahmad
Department of Computer Engineering,
Faculty of Engineering and Technology,
Jamia Millia Islamia, New Delhi-110025, India

***Abstract*—Efficient random sequence generators are significant in the application areas of cryptographic stream cipher design, statistical sampling and simulation, direct spread spectrum, etc. A cryptographically efficient pseudo-random sequence should have the characteristics of high randomness and encryption effect. The statistical quality of pseudo-random sequences determines the strength of cryptographic system. The generation of pseudo-random sequences with high randomness and encryption effect is a key challenge. A sequence with poor randomness threatens the security of cryptographic system. In this paper, the features and strengths of chaos and neural network are combined to design a novel pseudo-random binary sequence generator for cryptographic applications. The statistical performance of the proposed chaotic neural network based pseudo random sequence generator is examined against the NIST SP800-22 randomness tests and multimedia image encryption. The results of investigations are promising and depict its relevance for cryptographic applications.**

Keywords— Chaotic, neural network, pseudo-random sequence generator, cryptography, image encryption.

I. INTRODUCTION

In today's world of technological advancements, the information is distributed, shared and exchanged at a very fast rate over the open wired/wireless networks. Therefore, it demands the need of methods to ensure confidentiality and privacy of sensitive information asset before transmitting it through communication channel. Cryptographic techniques are used to achieve this goal and make the information unintelligible to unauthorized parties except the intended recipient. An efficient cryptosystem must be able to generate high quality random sequences to extract cryptographic keys. The usage and significance of pseudo-random number generators is not limited to cryptographic techniques. They also form an essential component in modern communication systems, statistical simulation and testing, etc [1-6]. Pseudo random number generators are deterministic processes which are used to generate a series of outputs from an initial state [7-9]. Pseudo-random number generators can be implemented either in software or in hardware. For hardware implementations, linear feedback shift registers based sequence generators are popularly used [4, 6, 10-12]. They are simple to implement and have high speed performance. However, they have an inherent disadvantage that their feedback tapings can be determined under Berleykamp Massey attack. The strength of the cryptosystem largely depends on the randomness of the sequences generated by pseudo-random generator, and thus this security flaw makes the LFSR-based generators insecure.

The extensive studies of chaotic systems dynamics in last two decades have revealed the interesting relationship between chaos and cryptography. According to the chaos theory, chaotic systems are the dynamical systems whose state evolves with time. Chaotic systems possess characteristics which are suitable for cryptographic use such as their sensitivity to initial conditions/system parameters, ergodicity, mixing properties, unstable periodic orbits with long periods and random-like behavior [13]. Due to these features, chaotic systems are extensively incorporated into the encryption systems to ensure secure transmission of text, images, audio or video over insecure networks [14-18]. Chaos based cryptosystems are known to have yielded better results and provide high levels of security than the conventional encryption techniques.

The interconnected structure of the neurons in a neural network introduces complexity in the system. The mixing of neuron outputs from one layer into a single neuron of the next layer makes the system sensitive to the plaintext and produces a seemingly random output. This property of neural networks makes them suitable for generating cryptographic keys and the inputs at various points. Furthermore, the structure of the neural network viz. the weights, biases and transfer function parameters are generated by using chaotic maps to make the system more random and sensitive to the plaintext. In this paper, the features of both chaotic systems and neural networks are exploited to build an efficient cryptographic pseudo-random sequence based on chaotic neural network. The analyses of the proposed generator against the statistical, randomness and encryption analyses demonstrate its excellent characteristics and improved performance.

The structure of rest of this paper is as follows: the proposed methodology for the chaotic neural network based cryptographic pseudo random sequences is described in subsequent section. The performance of proposed generator against the statistical tests, randomness tests and encryption test is discussed Section III, while the conclusion of the work is drawn in Section IV.

II. PROPOSED GENERATOR

The proposed chaotic neural network based pseudo-random binary sequence generator is illustrated in the following subsections.

978-1-4799-4910-6/14 $31.00 © 2014 IEEE
DOI 10.1109/ACCT.2014.38

A. The Neural Network

In the proposed generator, the neural network shown in Fig.1 is used, which is composed of four layers: the input layer, the first hidden layer, the second hidden layer and the output layer. The input to the neural network is a 64-bit key $\boldsymbol{P} = [P_1\ P_2\ ....P_{64}]$. The output of the input layer is defined as:

$$C=F^{n0}(\Sigma W_0 P + B_0, Q_0) \quad (1)$$

Where n_0 is a random number ($1 \le n_0 \le 10$) generated by the key generator and $F(.)$ is the transfer function, which is the piecewise linear chaotic map (PWLCM) [19]. The PWLCM is among the most studied chaotic systems. Its system equation is described as follows:

$$x(k+1) = F(x(k), q) = \begin{cases} \dfrac{x(k)}{q} & 0 < x(k) \le q \\ \dfrac{1-x(k)}{1-q} & q < x(k) < 1 \end{cases} \quad (2)$$

Where $x(k)$ is state of the map and q is the control parameter and satisfies $0 < x(k) < 1$, $0 < q < 1$. The lyapunov exponent of a dynamical system specifies the rate of separation of minutely close trajectories. The PWLCM has highest lyapunov exponent at $q = 0.5$. The input weight matrix is $W_0=[w_{0,0},w_{0,1}...w_{0,7},w_{1,0}...w_{7,7}]$ of size 8×8, the bias matrix is $B_0=[b_0,b_1...b_7]$ of size 8×1, and the control parameter matrix $Q_0=[q_0,q_1...q_7]$ is of size 8×1. To calculate the output of the input layer C, first the inputs $P_1,P_2\ ....\ P_{64}$ are multiplied by their respective weights $(w_{0,0},w_{0,1}...w_{0,7},w_{1,0}...w_{7,7})$ and added with the respective biases $(b_0,b_1...b_7)$ of neurons. Then this value is used as current state x, and generated control parameter q to iterate the PWLCM map for n_0 times, as described in eqn (1). Similarly, the matrices D, E and Op are computed as:

$$D = F^{n1}(\Sigma W_1 C + B_1, Q_1)$$

$$E = F^{n2}(\Sigma W_2 D + B_2, Q_2)$$

$$Op = F^{n3}(\Sigma W_3 E + B_3, Q_3)$$

Where the matrix W_1 is of size 4×8, W_2 of 2×4, W_3 of 1×2, B_1 of 4×1, B_2 of 2×1, B_3 of 1×1, Q_1 of 4×1, Q_2 of 2×1 and Q_3 of 1×1. Initially, the matrices W_0, B_0, Q_0, W_1, B_1, Q_1, W_2, B_2, Q_2, W_3, B_3, Q_3 received the values generated from the key generator. Like n_0, the random numbers n_1, n_2, n_3 are generated by the key generator ($1 \le n_1, n_2, n_3 \le 10$) and define the number of iterations of the transfer function at each layer. The random iterations of the transfer functions improves the randomness of relation between the input and output of each layer.

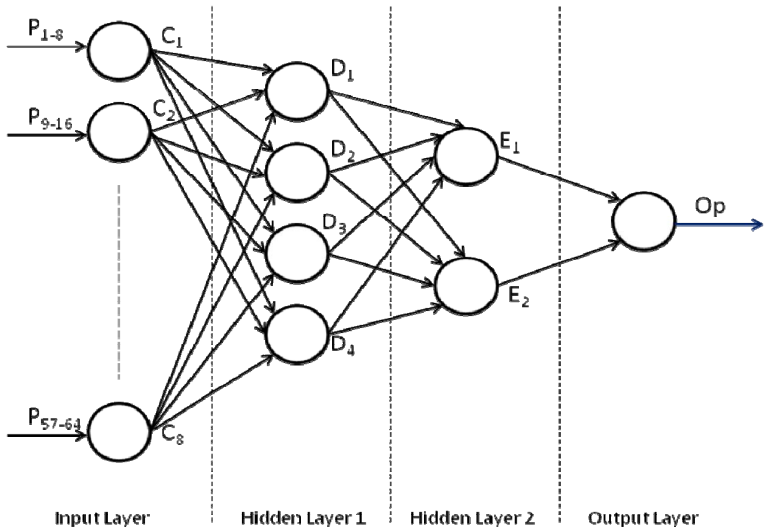

Figure 1. Synoptic of proposed four-layer chaotic neural network

After one complete operation of the chaotic neural network, the output of each neuron of the Input Layer directly becomes the input of that neuron for the next operation, apart from becoming the input to the first hidden layer for the same operation. The control parameter matrices Q_i are transformed using the networks outputs in such a way that limit their range in [04, 0.6] as:

$$Q_0 = (0.2\times C) + 0.4$$

$$Q_1 = (0.2\times D) + 0.4$$

$$Q_2 = (0.2\times E) + 0.4$$

$$Q_3 = (0.2\times Op) + 0.4$$

This transformation is done to keep the value of the control parameters close to 0.5 in order to obtain the best chaotic behaviour from the chaotic transfer function. *Op(j)* is the output of the neural network after *j*-th operation and satisfies $0 < Op < 1$. Op is normalized in the range 0-255 as:

$$w(j) = (Op(j) \times 10^{10}) mod(256)$$

If it crosses the threshold value of 127, then the next member of the pseudo-random binary sequence is taken as 1, else 0.

$$\psi(j) = \begin{cases} 0 & if \quad w(j) \le 127 \\ 1 & if \quad w(j) \ge 128 \end{cases}$$

B. The Key Generator

The key generator uses the 1-D chaotic cubic map and accepts a 64-bit key $\boldsymbol{K} = K_1K_2K_3K_4$, where K_i is a 16-bit component of key $\boldsymbol{K}$, to calculate the initial condition $y(0) = (\Sigma(K_i/2^{16}))mod(1)$ of cubic map and returns the values $y(n)$ on iterations. The state of the cubic map is governed as:

$$y(n+1) = \lambda.y(n).(1 - y(n).\, y(n))$$

Where $\lambda = 2.59$ is map's control parameter and the state of the map $y(n)$ satisfies $0 \le y(n) \le 1$.

C. The Proposed Algorithm

The steps of the proposed chaotic neural network based pseudo-random binary sequence generation are as follows.

1. Input key $\boldsymbol{K}$ to the key generator and iterate it 50 times and discard the values.
2. Again, iterate the key generator to initialize $W_0, W_1, W_2, W_3, B_0, B_1, B_2, B_3, Q_0, Q_1, Q_2, Q_3, n_0, n_1, n_2, n_3$.
3. Provide the 64-bit seed $\boldsymbol{P}$ to the neural network.
4. Operate the neural network to obtain the output *Op*
5. Normalize the output *Op* in the range 0-255.
6. Apply thresholding with 127 to extract next member $\psi(j)$ of the random binary sequence.
7. Assign Q_0 to C, Q_1 to D, Q_2 to E, and Q_3 to Op after applying the transformation.
8. Repeat steps 3 to 7 to obtain the pseudo-random sequence of desired length.

III. Performance Investigations

In this section, the statistical performance against the auto-correlation function, 0/1 balancedness, randomness and encryption efficiency of the proposed pseudo-random sequence generator is investigated. The two keys of 64-bits used for simulation are $\boldsymbol{K}$ and $\boldsymbol{P}$. The key space of proposed generator is $(2^{16})^4 \times (2^{64}) = 2^{128}$. One of the statistical parameters used to measure the quality of a sequence is the auto-correlation function. According to the first postulate of Golomb, the auto-correlation function has a delta-function form for a perfectly noise-like sequence. The auto-correlation function of the proposed pseudo-random sequence is shown in Fig. 2. It is evident from the figure that sequence has good auto-correlation function form and the function has a maximum value of 0.004462 for non-zero shift.

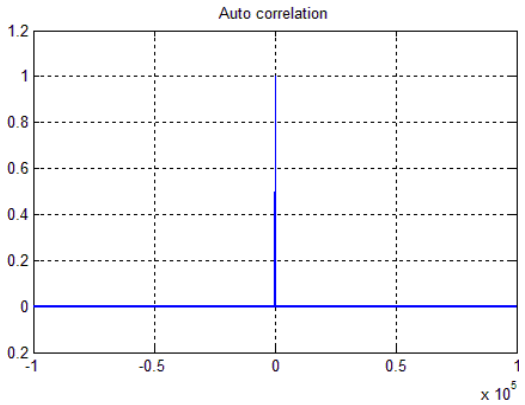

Figure 2. Auto-correlation function of pseudo-random sequence

A. 0/1 Balancedness Test

According to the second postulate of Golomb, the noise-like sequence has an equality distribution. An efficient pseudo-random binary sequence should have equal numbers of 0's and 1's. To check the equality distribution of the proposed pseudo-random binary sequence, the number of 1's are counted and listed in Table I. The equality distribution measures provided in Table I and Fig. 3 are found close to 50% showing that the proposed generator satisfy the equality distribution property.

TABLE I. Equality Distribution of Proposed PRBS

Sequence length	Count of 1's	%age
1000	512	51.2
10000	5040	50.4
20000	9997	49.98
50000	24957	49.91
100000	50090	50.09
200000	100233	50.12
500000	250055	50.01

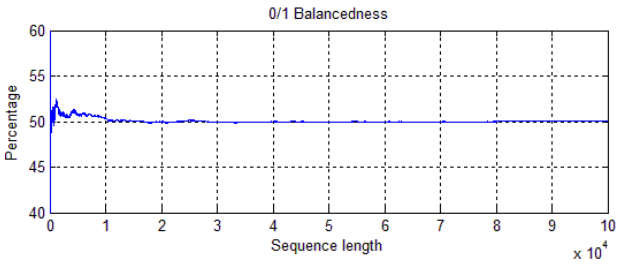

Figure 3. Equality distribution showing percentage occurrence of 1's

B. NIST Randomness Tests

To verify the randomness performance of proposed pseudo-random sequence generator, the randomness tests of Statistical Test Suite (STS) by NIST [20] are performed. The tests such as Frequency, Block frequency, Cusum-forward, Cusum-reverse, Runs, Longest runs, Rank, FFT, Linear complexity, Serial, Approximate entropy, Lempel-Ziv compression and Overlapping template tests are performed on the generated binary sequence. These randomness tests are used to calculate a *p*-value. If a *p*-value for a test is determined to be equal to 1, then the sequence appears to have perfect randomness. A *p*-value of zero indicates that the sequence failed the test, and appears to be completely non-random [12]. A significance level (α) is chosen for the tests. If *p*-value ≥ α, then the sequence appears to be random. Else if *p*-value < α, then the sequence appears to be non-random. Typically, α is chosen in the range [0.001, 0.01]. An α = 0.01 indicates that one would expect 1 out of 100 sequences to be rejected. A *p*-value ≥ 0.01 would mean that the sequence is random with a confidence of 99%. A *p*-value < 0.01 would mean that the sequence is non-random with a confidence of 99%. The STS randomness tests are applied using α = 0.01. The randomness results of a proposed pseudo-random sequence consisting of first 100,000 bits are listed in the Table II. It is clear that the sequence under test has passed all above mentioned randomness tests. The *p*-value obtained in each tests is extensively greater than chosen α, which ensures the high randomness of the generated sequence. Moreover, it can also be seen from the comparison made in Table II that the proposed chaotic neural network based generator has better randomness performance than the existing LFSR based generator investigated in [12].

TABLE II. RESULTS OF NIST STS RANDOMNESS TESTS

Randomness Test	***p-value* [In Proposed]**	**Comparison**	
		In Proposed	**In [21]**
Frequency Test	0.827429	Pass	Pass
Block Frequency Test	0.809163	Pass	Pass
Cusum-Forward Test	0.964088	Pass	Pass
Cusum-Reverse Test	0.809385	Pass	Pass
Runs Test	0.070599	Pass	Pass
Longest Runs Test	0.951598	Pass	**Fail**
Rank Test	0.143839	Pass	Pass
FFT Test	0.409891	Pass	**Fail**
Linear Complexity Test	0.626052	Pass	**Fail**
Serial Test	0.200756	Pass	**Fail**
Approx Entropy Test	0.600440	Pass	**Fail**
Lempel-Ziv Compression	0.398475	Pass	Pass
Overlapping Template	0.626269	Pass	**Fail**

C. Encryption Test

To test the encryption performance of the generated sequence, it is experimented with same image chosen in [12], i.e. standard 8-bits gray-scale *Lena* image of same size 256×256, shown in Fig. 4. The generated pseudo-random sequence is XORed with the plain-image bitstream to obtain encrypted image shown in Fig. 6. The histogram of the plain image is shown in Fig. 5, and that of the encrypted image is shown in Fig. 7. The average of the gray-level intensities of the encrypted image (Fig. 4) comes out as 127.5689. In [12], the gray-level intensities average is 127.2324. The average value obtained by encryption with our generator is more close to the ideal value = 127.5.

Figure 4. Plain-image of *Lena*

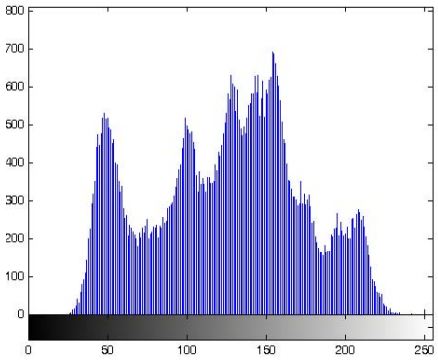

Figure 5. Pixels gray value distribution in plain-image of *Lena*

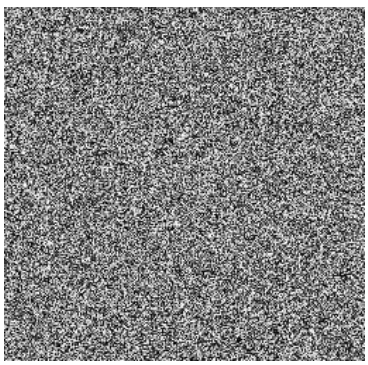

Figure 6. Encrypted *Lena* image

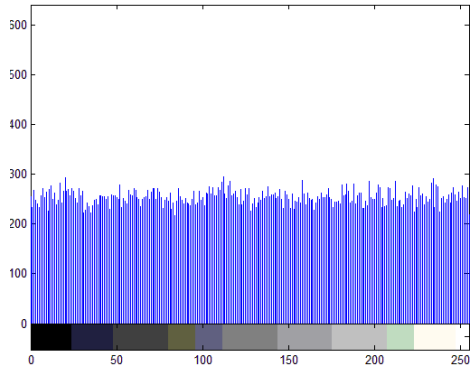

Figure 7. Pixels gray value distribution in encrypted *Lena* image

The standard deviation in frequencies of pixel intensities is an indication of the flatness/uniformity of the histogram (distribution of pixels gray-level intensities) of an image. A lower score of this deviation means that the histogram is more flat/uniform and it is more close to the histogram of a perfect white noise image. The score of this deviation is found out to be 14.9209 for the Fig. (6)-(7) , which is lower than the value 15.7191 obtained in [12]. Thus, it can be concluded that the histogram shown in Fig. 7 is more flat and the pixels are more uniformly distributed in the encrypted image.

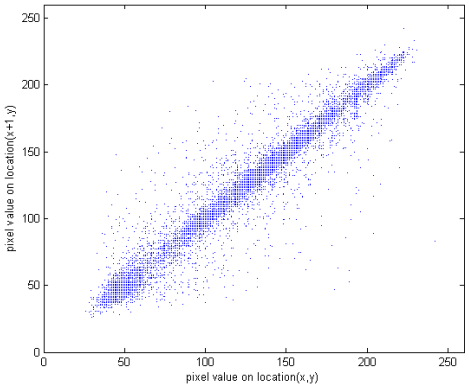

Figure 8. Correlation of adjacent pixels in plain-image

The adjacent pixels correlation analysis of encrypted image shown in Fig. 6 is done and the values of correlation coefficients for randomly chosen 10,000 vertically, horizontally and diagonally adjacent pixels in the encrypted image come out to be -0.001943, 0.002661 and 0.000713, respectively. These values are very close to their ideal value = 0. Thus, it is evident that the pixels of encrypted image are highly uncorrelated to each other. Fig. 8-9 depicts the relation of the adjacent pixels of plain-image and encrypted image. Fig. 9 demonstrates the high uncorrelation of adjacent pixels in encrypted image. The cross-correlation between two images data shown in Fig. 10, largely confines towards zero line, this ensures that the two data don't have any correlation and they are highly deviated from each other. The pixels percentage difference between the plain-image and encrypted image is found out to be 99.62%, which further confirms that the encrypted image is totally distinct from plain-image. The information entropy analysis of the encrypted image shows that the image shown in Fig. 6 has an entropy of 7.9976, which is close to its ideal value=8 (for an 8-bits grey-scale perfect white noise image); the value 8 corresponds to a perfect random source. The entropy of encrypted image indicates that the pseudo-random binary sequence generated with the proposed chaotic neural network based generator is well suited for encryption applications.

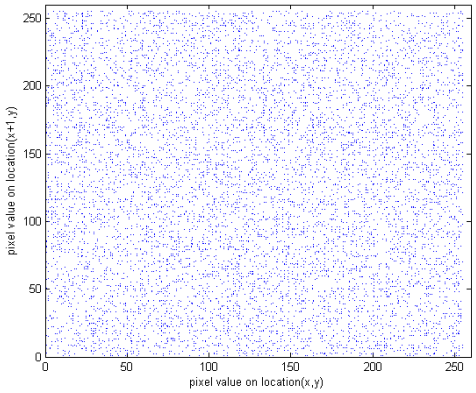

Figure 9. Correlation of adjacent pixels in encrypted image

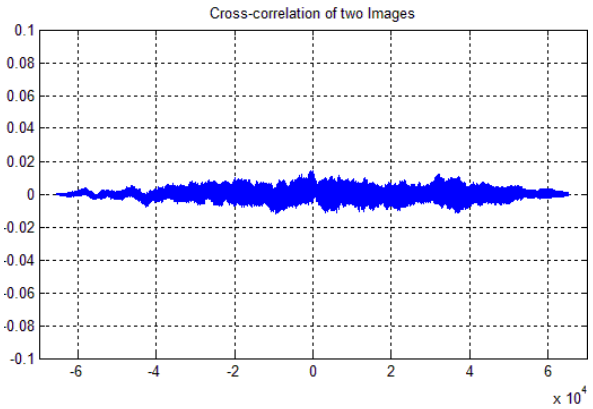

Figure 10. Cross-correlation of plain-image and encrypted image data

TABLE III. COMPARISON OF PROPOSED AND LFSR BASED GENERATOR

Parameter	**In Proposed**	**In [12]**
Randomness	Highly Random	Less Random
Avg. of gray-values	127.5689	127.2324
Std. dev. in freq. of gray-values	14.9209	15.7191
Key space	2^{128}	$\approx 2^n \times 2^k = 2^{12}$ (for n=8 and k=4)

IV. CONCLUSIONS

A chaotic neural network based pseudo-random sequence generator is proposed. The generator utilises the high sensitivity and randomness property of chaotic maps such as the cubic map and the piece-wise linear chaotic map, coupled with the nonlinear complexity of a four-layer neural network. The statistical, randomness and encryption performance of the generated pseudo-random sequence is tested and compared with a recent LFSR-based generator. The statistical and comparative analyses show the proposed chaotic neural network based generator outperforms the existing LFSR-based generator. The investigations also confirm that the characteristics of generated pseudo-random binary sequence are close to that of perfect noise sequence. Hence, the proposed chaotic neural network based generator is strong and suitable for cryptographic applications.

REFERENCES

[1] E. Cruselles, M. Soriano, and J. L. Melus, "Uncorrelated PN sequences generator for spreading codes in CDMA systems", Sixth IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, vol. 3, pp. 1335--1340, 1995.

[2] V. R. Gonzalez-Diaz, E. Bonizzoni, and F. Maloberti, "Pseudorandom sequence generation for mismatch analog compensation of ADCs", IEEE International Conference on Electronics, Circuits, and Systems, pp. 118--121, 2010.

[3] S. Y. Hwang, G. Y. Park, D. H. Kim and K. S. Jhang, "Efficient implementation of a pseudorandom sequence generator for high-speed communications", ETRI Journal, vol. 32, no. 2, pp. 222--229, 2010.

[4] H. Zhou, H. C. Card and G. E. Bridges, "Parallel pseudorandom number generation in GaAs cellular automata for high speed circuit testing", Journal of Electronic Testing, vol. 6, no. 3, pp. 325-330, 1995.

[5] H. Niederreiter and A. Winterhof, "On a new Class of inversive pseudorandom numbers for parallelized simulation methods", Periodica Mathematica Hungarica, vol. 42, no. 1 - 2, pp. 77--87, 2001.

[6] R. Latif and M. Hussain, "Hardware-based random number generation in wireless sensor networks", Advances in Information Security and Assurance, Lecture Notes in Computer Science, vol. 5576, pp. 732--740, 2009.

[7] R. M. DSouza, Y. Bar-Yam and M. Kardar, Sensitivity of ballistic deposition to pseudorandom number generators, Physical Review E, vol. 57, no. 5, pp. 5044–5052, 1998.

[8] J. F. Fernandez and C. Criado, Algorithm for normal random numbers, Physical Review E, vol. 60, no. 3, pp. 3361–3365, 1999.

[9] I. Vattulainen, T. Ala-Nissila and K. Kankaala, Physical models as tests of randomness, Physical Review E, vol. 52, no. 3, pp. 3205–3214, 1995.

[10] R. Mita, G. Palumbo and M. Poli, "Pseudo-random sequence generators with improved inviolability performance", IEE Proceedings- Circuits, Devices and Systems, vol. 153, no. 4, pp. 375--382, 2006.

[11] M. Ahmad and Izharuddin, "Security enhancements in GSM cellular standard", IEEE International Conference on Wireless Communication and Sensor Networks, pp. 116-1119, 2008.

[12] F. Maqsood, O. Farooq and W. Ahmad, "LFSR and PLA based complex code generator for stream cipher", International Conference on Multimedia, Signal Processing and Communication Technologies, pp. 269--272, 2009.

[13] G. Alvarez and S. Li, "Some basic cryptographic requirements for chaos based cryptosystems", International Journal of Bifurcation and Chaos, vol. 16, no. 8, pp. 2129-2151, 2006.

[14] G. Y. Chen, Y. B. Mao and C. K. Chui, "A symmetric image encryption scheme based on 3D chaotic cat maps", Chaos Solitons & Fractals, vol. 21, no. 3, pp. 749--761, 2004.

[15] M. Ahmad, C. Gupta and A. Varshney, "Digital Image Encryption Based on Chaotic Map for Secure Transmission", International Conference on Multimedia, Signal Processing and Communication Technologies, pp. 289--292, 2009.

[16] K. W. Tang and W. K. S. Tang, "A chaos-based secure voice communication system", IEEE International Conference on Industrial Technology, pp. 571--576, 2005.

[17] S. Lian, J. Sun, J. Wang and Z. Wang, "A chaotic stream cipher and the usage in video protection", Chaos, Solitons and Fractals, vol. 34, no. 3, pp. 851--859, 2007.

[18] Y. Tang, Z. Wang and J. Fang, "Image encryption using chaotic coupled map lattices with time varying delays", Communication in Nonlinear Science and Numerical Simulation, vol. 15, no. 9, pp. 2456--2468, 2009.

[19] S. Li, G. Chen and X. Mou, "On the dynamical degradation of digital piecewise linear chaotic maps", International Journal of Bifurcation and Chaos, vol. 15, no. 10, pp. 3119–3151, 2005.

[20] A. Rukhin, et al. "A Statistical Test Suite for Random and Pseudo-random Number Generators for Cryptographic Applications", NIST Special Publication 800-22, 2001.