

# Randomness properties of sequence generated using logistic map with novel permutation and substitution techniques

Pushpalatha Gopalakrishna Saraswathy, Ramesh Siddaiah

Department of Electronics and Communication Engineering, Dr. Ambedkar Institute of Technology,  
Affiliated to Visvesvaraya Technological University, Bengaluru, India

## Article Info

### Article history:

Received Jul 4, 2022

Revised Sep 22, 2022

Accepted Oct 1, 2022

### Keywords:

Chaos cryptography

Permutation technique

Random number generation

Statistical tests

Substitution technique

## ABSTRACT

In this paper, a design of a chaos-based keystream generator (KSG) using a novel permutation technique with various two-dimensional patterns and a substitution technique with  $Z_4$  mapping is proposed. Initially, a chaotic function such as a logistic map is used to generate a pseudo-random number. Then these numbers are converted into binary sequences using binary mapping. In order to achieve statistical properties of the resultant binary sequences, a novel method of KSG is developed by considering parameters such as initial value " $x_0$ ", system parameter " $r$ ", novel permutation techniques defined by 2-dimensional patterns, and substitution technique defined over  $Z_4$  transformation. The binary sequences so obtained are subjected to randomness tests by applying the National Institute of Standards and Technology (NIST) SP-800-22 (Revision 1a) test suite for investigation of its randomness properties to obtain suitable sequences which can be used as a key for cryptographic applications. From the results obtained, it is found that the binary sequences exhibit better randomness properties as per the cryptographic requirements.

*This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.*



## Corresponding Author:

Pushpalatha Gopalakrishna Saraswathy

Department of Electronics and Communication Engineering, Dr. Ambedkar Institute of Technology,

Affiliated to Visvesvaraya Technological University

Bengaluru, India

Email: pushpalathags.ec@drait.edu.in

## 1. INTRODUCTION

Multimedia data like audio, video, and image are widely used in present days due to their enormous usage in various applications such as entertainment, military, education, banking, communication, and medical. The data is highly susceptible to threats from different directions. Thus, the protection of such data is essential in order to achieve confidentiality, authenticity integrity, and overall security. Cryptography provides protection significantly to achieve secure communication of data [1], [2]. Traditional block encryption algorithms such as data encryption standard (DES) [3], advanced encryption standard (AES) [4], international data encryption algorithm (IDEA) [5], and Rivest-Shamir-Adleman (RSA) [6] are found to be expensive and slow, hence not suitable for multimedia data in real-time applications, as the size of multimedia data is very large [7], [8].

Stream cipher systems are more suitable for real-time processing of large-sized or bulky multimedia data such as images, speech, and video. Generally, in a stream cipher system, a binary message is encrypted bit by bit modulo-2 in addition to a binary random sequence called a key sequence [9]. The security of a stream cipher system depends on the randomness properties of the key sequence. To meet the demands of the real-time processing and security concerns of multimedia data, chaos-based functions have been in use to generate key streams having desirable randomness properties as required for cryptographic applications. Chaos-based functions are of high interest due to their potential benefits such as high non-linearity, low cost, and large

periodicity [9], [10]. The key sequence is a very important building block of the entire stream cipher system. The important properties of chaos functions such as non-linearity and sensitivity to minutely changed initial conditions have been exploited to obtain randomness characteristics as required for the key sequence for the stream cipher system. Hence the design of a “Chaos-based key stream generator” is a topic of interest. The National Institute of Standards and Technology (NIST) compliance binary sequences which possess the desirable randomness properties to be suitable for cryptographic applications are generated from the proposed generator.

## 2. LITERATURE SURVEY

Chaos-based cryptography is one of the interesting areas for real-time applications such as data, video, and speech encryption for secure communication. Chaotic functions have been finding a place in key generation algorithms due to their nature of exhibiting nonlinearity, which can be exploited to produce a large number of sequences having random properties needed for key sequences. A better chaos-based crypto-system will have some important requirements to meet the demands of cryptography as reported in [11]–[16] such as a well-defined key generation process, large periodicity of key sequence, large key space with uniform distribution, high sensitivity to minute changes, high security, and good resistance to various attacks and threats. Apart from these, the feasibility of implementing in real time without compromising security, cost, and speed is also an appealing feature in applications such as mobile communication and online transactions.

With the increasing level of complexity in the technology, the speed of operation and processing also increased. Hence, the need for a high level of security in the field of hardware technology was more in demand. Chaotic systems are very much sensitive to initial conditions. Various properties of the chaotic systems like ergodicity, mixing, stretching, and folding make them suitable for constructing cryptographic systems [13], [14], [17], [18]. Binary sequence having large linear complexity property is one of the important needs for cryptographic applications and is discussed in matrix recurrence relation over  $Z_4$  [17].

Chaos-based algorithms [19]–[23] uses chaotic maps such as logistic maps, Lorenz maps, Baker maps, and Chen maps to generate random numbers to be suitable for cryptographic needs. Some of the one and two-dimensional chaotic cryptosystems found in [14], [15], [23] are promising generators for their suitability for cryptographic applications. The better the randomness properties of the key sequence, the better the security for the stream cipher system. Maximum-length sequences called m-sequences generated by n-stage linear feedback shift registers (LFSRs) have very good randomness properties. However, m-sequences have low linear complexity and hence are not suitable for cryptographic applications [24]–[26]. In practical stream cipher designs, a large linear complexity of the key stream is obtained by a nonlinear transformation of the LFSR sequences [27], [28].

Some of the examples of cryptographically secure pseudorandom bit generators that present a quarter-rate pseudo-random binary sequence generator (PRBSG) are RSA pseudorandom bit generator [29], Micali-Schnorr pseudorandom bit generator [30], [31], Blum-Blum-Shub generator ( $x^2 \bmod N$  generator) [27], [28], [32]. For cryptographic applications, random number sequences having desirable statistical properties may be used as a key. There are many standard test suites like NIST [33], DIEHARD, and Crypt-XS. are recommended to determine the randomness properties of sequences and their suitability for cryptographic applications [2].

## 3. PROPOSED METHOD OF CHAOS BASED KEY STREAM GENERATOR

In this work, a novel method for generating cryptographically secure random binary sequences is proposed. The process of binary sequence generation in the proposed model chaos based key stream generator (CBKSG) is as shown in Figure 1. Initially, a chaotic function namely a logistic map is used to generate pseudorandom numbers (PRNs) and which are then converted into binary sequences. This does not possess the randomness properties as required to be suitable for cryptographic key sequences. Thus, the proposed work aims at generating binary sequences which possess randomness properties as suitable for cryptographic key sequences. The overall process involves applying permutation and substitution techniques to the resultant sequences of logistic maps to ensure they possess good randomness properties as suitable for the key sequences for cryptographic applications.

In this work, a design of a novel permutation technique by defining various two-dimensional patterns and a substitution technique by defining  $Z_4$  mapping is proposed. In the permutation technique, the defined patterns are made to visit randomly on the resultant binary sequence of the logistic map which is arranged in two-dimensional space. The random visit made by the 2-dimensional pattern is considered to construct new binary sequences and they are tested for their randomness properties. In the substitution technique, binary sequences obtained after the permutation are subjected to nonlinear mapping  $Z_4$  transformation to obtain new binary sequences and they are tested for their randomness properties.

The standard statistical test suite NIST SP 800-22 (Rev-1a) is used to test the randomness properties of the sequences generated from the proposed keystream generator. The permutation and substitution techniques are explained in the following section, it is also shown that the sequences so obtained exhibit randomness properties suitable as keys for cryptographic applications. The design process involves three stages. They are explained in the following section.

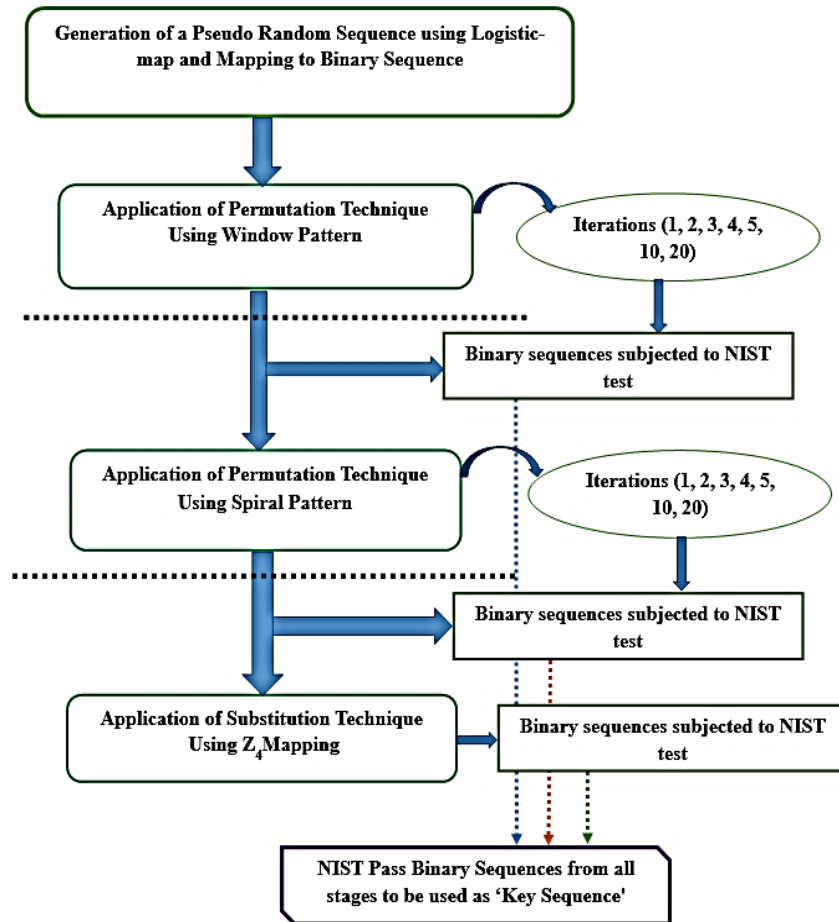


Figure 1. Proposed CBKSG

**3.1. Stage-1: logistic map**

The model is designed as a three-step process during the first stage. In step 1, the floating-point PRNs are generated using a logistic map with predefined initial conditions. In step 2, the floating values are converted into integer values by the decimal conversion method. In step 3, the integer values will be converted into binary values. The steps are explained in detail as follows.

- a. Generation of the floating-point sequence using a logistic map

A pseudo-random sequence (floating in nature) is generated using a logistic map as defined in (1),

$$x_{n+1} = rx_n(1 - x_n) \tag{1}$$

where  $0 \leq x_0 < 1$  and  $0 \leq r < 4$ ;  $x_0$  the initial value,  $x_{n+1}$ , the next value. The bifurcation diagram of the logistic map is shown in Figure 2. It exhibits chaotic behavior in the range  $3.99 \leq r < 4$ . The output of the logistic map is a floating-point sequence that is considered in the range of 0.000000 to 0.999999 and mapped to an integer of six digits. Then it is converted into a binary stream of length  $N=10^6$ .

- b. Conversion of floating values into integer values

The floating numbers are converted into integer values by multiplying each value by  $N=10^6$  and stored in column vector 'A' of size  $N \times 1$  as defined in (2),

$$A^T = [I_1, I_2, I_3, \dots, I_N] \quad (2)$$

where  $I_n$  is an integer value with  $1 \leq n \leq N$  and  $0 \leq I_n \leq N-1$

c. Conversion of integer values into Binary stream ( $B$ )

The integer values  $I_n$  are then converted into a binary stream of length  $N$  by using the simple rule as defined in (3).

$$B = \begin{cases} 1, & I_n = \text{Even} \\ 0, & I_n = \text{Odd} \end{cases} \quad (3)$$

Thus,  $10^6$  bits are generated and saved into a square matrix ' $C$ ' of size  $10^3 \times 10^3$ .

In this way, a set of about 1,000 binary sequences are generated each of length  $10^6$  bits. These sequences need to be scrambled in order to kill their periodicity and linearity to obtain randomness properties. Hence permutation techniques are developed and applied. The model is developed to the next stage called stage-2 by designing and applying permutation techniques.

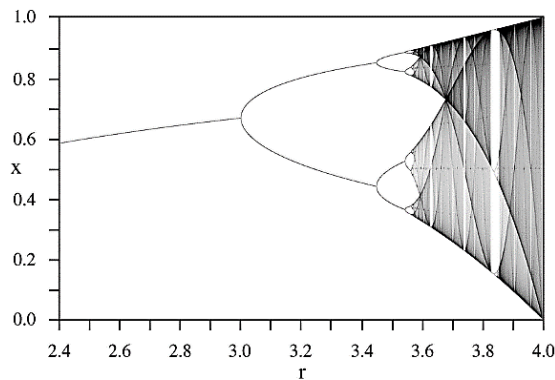


Figure 2. Bifurcation diagram of logistic map

### 3.2. Stage-2: permutation techniques

In the second stage, two-dimensional permutation patterns called 'Window Pattern' denoted by ' $P_w$ ' and 'Spiral Pattern' denoted by ' $P_s$ ' are designed. They are used as moving windows on the binary sequences obtained from the previous stage which are arranged in a two-dimensional space. These patterns are explained in detail in the following section.

#### 3.2.1. Window pattern: $P_w$

$P_w$  is a combination of two sub patterns namely  $P_1$  and  $P_1'$ . Thus,  $P_w = (P_1 \cup P_1')$ . The sub patterns  $P_1$ ,  $P_1'$  and main pattern ' $P_w$ ' are shown in Figures 3(a) to (c), respectively. The sub pattern  $P_1$  is defined for the square matrix of size  $3 \times 3$  as shown in Figure 3(a). The trace of the nodes  $o_{(1)}$ ,  $o_{(2)}$ ,  $o_{(3)}$ , and  $o_{(4)}$  depicts the path to visit by the pattern  $P_1$ . The binary values stored at these points are read in the said order.

For example, if  $o_{(1)}$ ,  $o_{(2)}$ ,  $o_{(3)}$ , and  $o_{(4)}$  are 1, 0, 1, and 1, then the read value along the trace forms binary string 1011. The sub pattern  $P_1'$  is defined for the square matrix of size  $3 \times 3$  is shown in Figure 3(b). The trace of the nodes  $*_{(1)}$ ,  $*_{(2)}$ ,  $*_{(3)}$ ,  $*_{(4)}$ , and  $*_{(5)}$  depicts the path to visit by the pattern  $P_1'$ . The binary values stored at these points are read in the said order. For example, if  $*_{(1)}$ ,  $*_{(2)}$ ,  $*_{(3)}$ ,  $*_{(4)}$ , and  $*_{(5)}$  are 1, 0, 0, 1, and 1 then the read value along the trace forms a binary string 10011.

The window pattern is denoted by  $P_w$ . The window pattern  $P_w$  is defined for the square matrix of size  $3 \times 3$  is shown in Figure 3(c), which is the union of patterns  $P_1$  and  $P_1'$ . The trace of the nodes  $o_{(1)}$ ,  $o_{(2)}$ ,  $o_{(3)}$ ,  $o_{(4)}$  followed by  $*_{(5)}$ ,  $*_{(6)}$ ,  $*_{(7)}$ ,  $*_{(8)}$ ,  $*_{(9)}$  depicts the path to visit by the pattern  $P_w$ . For example, if the  $3 \times 3$  matrix has 9 binary values along the nodes  $o_{(1)}$ ,  $o_{(2)}$ ,  $o_{(3)}$ ,  $o_{(4)}$  are 1, 0, 1, 1 respectively and  $*_{(5)}$ ,  $*_{(6)}$ ,  $*_{(7)}$ ,  $*_{(8)}$ ,  $*_{(9)}$  are 1, 0, 0, 1, 1 respectively, then the read value along the trace of the window pattern is 101110011 in the order of their positions.

This way the window pattern will be moved from left to right along the binary values stored in the square matrix ' $C$ ' from the top left position to the bottom right position without overlapping. The random visit made by the window pattern on the binary values of matrix  $C$  is considered to construct a new binary sequence. In this manner, a permuted binary sequence is generated. The uncovered values are considered row-wise and

appended at the end of the generated sequence. With this technique, the original binary sequence of matrix C is scrambled and thus loses its originality and periodicity.

The illustration of the application of the window pattern is shown for the 6×6 square matrix in Figure 4. The matrix has 36 binary values obtained from the logistic map arranged row-wise. These binary values have to be permuted using the window pattern. Thus, the window pattern will be moved from left to right along the binary values stored in the 6×6 square matrix from the top left position to the bottom right position without overlapping. The random visit made by the window pattern on the binary values of the matrix is considered to construct a new binary sequence. The values are read along the trace of the window pattern in the order from 1 to 36. Let the permuted binary sequence after applying the window pattern be denoted by  $B_w$  as shown in (4).

In this way, the window pattern is applied to the sequences of matrix C which is of size 1000×1000 in the manner as illustrated in Figure 4. After the application of the window pattern to matrix C, we obtain the first permuted binary sequence of length  $10^6$ . In this manner, a set of about 10,000 sequences are generated each of length  $10^6$  for various initial values ‘ $x_0$ ’ and system parameter ‘ $r$ ’. These sequences have to be tested for randomness properties.

$$B_w = \{O_1, O_2, O_3, O_4, *5, *6, *7, *8, *9, O_{10}, O_{11}, \dots \dots *_{14}, *_{15}, \dots \dots, O_{31}, \dots, O_{36}\} \tag{4}$$

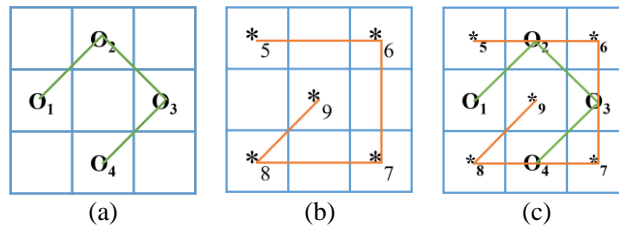


Figure 3. Sub patterns and window pattern; (a)  $P_1$ , (b)  $P_1$ , and (c)  $P_w$

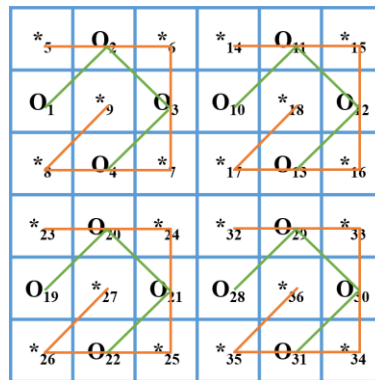


Figure 4. Illustration of the use of window pattern for 6×6 matrix

**3.2.2.  $B_w$  sequences to NIST test**

The output binary sequences ( $B_w$ ) obtained from the use of window patterns are subjected to the statistical test suite NIST-SP 800-22 (Revision 1a) for testing randomness properties. The permutation process using the window pattern is repeated for several iterations 1, 2, 3, 4, 5, 10, and 20 considering the output sequences of the previous iteration as input for the next iteration. After the completion of every iteration, the resultant sequences will undergo statistical tests for randomness properties. As the observations made from their randomness tests at every iteration, it is found that the sequences are not satisfying the required properties for the cryptographic key. To improve their statistical properties further, a permutation pattern called spiral pattern ‘ $P_s$ ’ is designed and used.

**3.2.3. Spiral pattern:  $P_s$**

Two-dimensional permutation pattern ‘ $P_s$ ’ is defined as shown in Figure 5. Spiral pattern can be used for any square matrix of size  $N \times N$  (for  $N$  is even). The generation of spiral pattern for square matrix of size  $N \times N$  ( $N$  is even) is explained in the following steps.

- a. Choosing the initial position of the spiral. Initial position of row and column is chosen as  $N/2, N/2$ .
- b. Trace of a spiral with spiral movement index ( $n$ ). The trace of the spiral for the first round begins at initial position ( $N/2, N/2$ ). The movement of the spiral for one round is defined as the function  $\{Trace(P_s)\}$  which is defined as follows.

$\{Trace(P_s)\}$  for one round of spiral: shift right by  $n$  position, shift down by  $n$  position, shift left by  $(n+1)$  position, shift up by  $(n+1)$  position where, one round of a spiral is traced with initial value of  $n=1$ , after completion of one round of spiral, for the second round ' $n$ ' is incremented twice as  $n+2$  and the trace will be continued from the same position, the process will continue by incrementing  $n$  value for every round until covering all the values in the matrix. The trace of the spiral can be stopped when the last value is covered in the matrix.

For example, if a  $4 \times 4$  matrix has 16 binary values arranged row-wise as 1011001101001100, then the initial row and column position is (2,2), and the trace of the spiral pattern is as depicted in Figure 6. The random visit made by the spiral pattern on the binary values of the matrix is considered to construct a new binary sequence. The permuted binary sequence obtained along the trace of the spiral is denoted as  $B_s$  are read as 0101001011100011.

In this way, the spiral pattern is applied to the resultant sequences of stage-1 which are stored in the square matrix ' $D$ ' which is of size  $1000 \times 1000$  in the manner as illustrated in Figure 6. After the application of the spiral pattern to matrix  $D$ , we obtain the permuted binary sequence of length  $10^6$ . In this manner, a set of about 10,000 sequences are generated each of length  $10^6$  for various initial values ' $x_0$ ' and system parameter ' $r$ '. These sequences will be tested again for randomness properties.



Figure 5. Spiral pattern

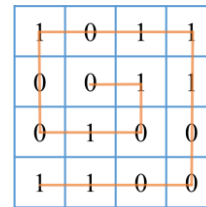


Figure 6. Illustration of the spiral pattern

### 3.2.4. $B_s$ sequences to NIST test

The output binary sequences ( $B_s$ ) obtained from the use of window patterns are subjected to the statistical test suite NIST-SP 800-22 (Revision 1a) for testing randomness properties. The permutation process using a spiral pattern is repeated for several iterations 1, 2, 3, 4, 5, 10, and 20 considering the output sequences of the previous iteration as input for the next iteration. After the completion of every iteration, the resultant sequences will undergo statistical tests for randomness properties. From the observations made of their randomness tests from every iteration, they are improved, however, the percentage of sequences passing the NIST tests is not sufficient to satisfy the required properties for the cryptographic key. To improve their statistical properties further the model is developed to the next stage.

### 3.3. Stage-3: substitution technique

In the third stage, a substitution process called  $Z_4$  mapping is designed. This technique is used in addition to the permutation techniques performed in the previous stages in order to increase non-linearity. As per the requirements for a cryptographic key, the randomness of the sequence can be improved by using the permutation and substitution processes together. The  $Z_4$  mapping is explained as follows.

#### 3.3.1. $Z_4$ mapping process

$Z_4$  mapping is a substitution technique, which is applied to the output sequences of stage 2. This results in final binary sequences denoted as  $B$  of length  $10^6$ . The steps involved in  $Z_4$  substitution technique are explained as follows.

- Sequence over  $\{Z_4\}$  is defined as  $\{Z_4\} = \{0,1,2,3\}$
- Binary sequences ( $B_s$ ) obtained from the stage-2 are mapped to  $Z_4$  elements using the following mapping
  - ‘00’ is mapped to 0
  - ‘01’ is mapped to 1
  - ‘10’ is mapped to 2
  - ‘11’ is mapped to 3



It is observed from the results of intermediate sequences  $\{B_w\}$  from Tables 1 and 2 that, the average percentage of sequences passing the NIST tests is 25 (without iteration) and 51 (with iteration). It is also observed from the results of intermediate sequences  $\{B_s\}$  from Tables 3 and 4 that, the average percentage of sequences passing the NIST tests is 61 (without iteration) and 72 (with iteration). The implication of the NIST results of intermediate sequences is that the average percentage of sequences passing NIST is improved to 72 by using the permutation techniques. Further improving the results, the substitution technique is used.

#### 4.2. NIST results of final binary sequences

The final binary sequences obtained from stage-3 by the process of substitution seem to exhibit good randomness properties. NIST test results of final binary sequences  $\{B\}$  are shown in Table 5. Some typical cases with results for all tests of NIST are shown in Table 6. It is observed from the results of final binary sequences  $\{B\}$ , that the average percentage of sequences passing the NIST tests is 97. The results are improved further from the previous stage by an additional substitution technique.

Table 5. Percentage of sequences  $\{B\}$  which pass NIST tests

No of sequence considered = 10.000										
System parameter ( $r$ ): 3.9000 to 3.9999, varied in steps of .0001										
Initial value ( $x_0$ )	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	
NIST Test result in percentage (%) [Iteration ( $n=0$ )]	97	98	97	98	98	98	96	97	98	

Table 6. NIST test results for typical cases

Statistical test	Case 1: $r=3.9865, x_0=0.1$		Case 2: $r=3.9865, x_0=0.5$		Case 3: $r=3.9865, x_0=0.9$	
	P-value	Result	P-value	Result	P-value	Result
Frequency	0.9662	Success	0.9775	Success	0.9775	Success
Block frequency	0.9887	Success	0.9887	Success	0.9662	Success
Cumulative sums	0.9662	Success	0.9775	Success	0.9887	Success
Runs	0.9887	Success	0.9775	Success	0.9887	Success
Longest run	0.9887	Success	0.9662	Success	0.9887	Success
Rank	0.9775	Success	0.9662	Success	0.9887	Success
FFT	0.9887	Success	0.9775	Success	0.9775	Success
Non overlapping templates	0.9775	Success	0.9775	Success	0.9775	Success
Overlapping templates	0.9887	Success	0.9775	Success	0.9550	Success
Universal	0.9887	Success	0.9550	Success	0.9887	Success
Approximate entropy	0.9775	Success	0.9775	Success	0.9887	Success
Random excursions	0.9811	Success	0.9887	Success	0.9800	Success
Random excursions variant	0.9811	Success	0.9791	Success	0.9800	Success
Serial	0.9775	Success	0.9662	Success	0.9662	Success
Linear complexity	0.9887	Success	0.9887	Success	0.9887	Success

Overall, comparing the NIST results of the sequences obtained from all stages as shown in Tables 1 to 5, it is evident that the percentage of sequences which passes NIST tests have been improved stage-wise. Hence, the proposed model has the process of permutation with window pattern, spiral pattern and, substitution with  $Z_4$  technique with satisfactory results in terms of efficiency, speed, simplicity, and novelty. Also, the final stage binary sequence of length  $10^6$  may be generated in less than a second. Hence, the proposed model is an efficient design for binary key stream generation for cryptographic applications.

## 5. CONCLUSION

With an extensive survey of cryptographic requirements and chaos-based encryption algorithms, it is found that binary sequences with good randomness properties are suitable as the key sequences for cryptographic applications. Hence a novel model namely 'chaos-based key stream generator' is developed to generate NIST compliance pseudo-random binary sequences. In this work Logistic map is used to generate PRNs initially by considering various parameters like initial value ( $x_0$ ) and system parameter function ( $r$ ). Novel permutation and substitution techniques have been developed and used to improve the statistical properties of the sequences obtained from the logistic map. Using the proposed model, large-length ( $10^6$ ) binary sequences are generated in large volumes (10,000) and are subjected to NIST statistical tests. NIST SP-800-22 (Revision 1a) statistical tests suite is used to determine the randomness properties of these sequences. From the NIST test results, it is evident that the sequences obtained from the model exhibit good randomness properties and meet the requirements of the cryptographic key. Hence, the binary sequences generated from the proposed model 'chaos-based key stream generator' using a novel permutation technique with various two-



dimensional patterns and substitution technique with  $Z_4$  mapping prove to be an efficient ‘cryptographically secure random binary sequence generator’.

## 6. FUTURE SCOPE

The variants of the proposed model may be designed further considering changes in any one or all the parameters. The variants can be obtained from initial values, permutation patterns, substitution patterns, and iterations. With each different parameter, there is a scope for developing a new model for pseudo-random number generation.

## ACKNOWLEDGMENT

This research work is funded by Vision Group of Science and Technology (VGST), Govt. of Karnataka (GRD 575). The authors acknowledge the support for the research work by VGST. The authors would also like to thank Dr. Ambedkar Institute of Technology, Bengaluru for providing the research support.




## REFERENCES

- [1] B. Schneier, *Applied cryptography: Protocols, algorithms, and source code in C*. Wiley; 2nd edition, 1996.
- [2] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*. CRC Press, New York, 1996.
- [3] A. Biryukov and C. Cannière, “Data encryption standard (DES),” in *Encyclopedia of Cryptography and Security*, Springer US, 2005, pp. 129–135.
- [4] NIST, “The advanced encryption standard (AES),” *Federal Information Processing Standards Publication 197*. pp. 1–51, 2001.
- [5] X. Lai and J. L. Massey, “A proposal for a new block encryption standard,” in *EUROCRYPT 1990: Advances in Cryptology — EUROCRYPT '90*, 1991, pp. 389–404.
- [6] R. L. Rivest, M. J. B. Robshaw, R. Sidney, and Y. L. Yin, “The RC6 block cipher,” in *First advanced encryption standard (AES) conference*, 1998, vol. 1–21.
- [7] P. Patil, P. Narayankar, N. D. G., and M. S. M., “A comprehensive evaluation of cryptographic algorithms: DES, 3DES, AES, RSA and blowfish,” *Procedia Computer Science*, vol. 78, pp. 617–624, 2016, doi: 10.1016/j.procs.2016.02.108.
- [8] W. Stallings, *Cryptography and network security principles and practices*. Prentice Hall, 2005.
- [9] R. A. Rueppel, *Stream ciphers in contemporary cryptology*. G. J. Simmons ed, IEEE Press, 1992.
- [10] B. Preneel, V. Rijmen, and A. Bosselaers, “Recent developments in the design of conventional cryptographic algorithms,” in *State of the Art in Applied Cryptography*, 1998, pp. 105–130.
- [11] C. E. Shannon, “Communication theory of secrecy systems,” *Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, Oct. 1949, doi: 10.1002/j.1538-7305.1949.tb00928.x.
- [12] G. Alvarez and S. Li, “Some basic cryptographic requirements for chaos-based cryptosystems,” *International Journal of Bifurcation and Chaos*, vol. 16, no. 08, pp. 2129–2151, Aug. 2006, doi: 10.1142/S0218127406015970.
- [13] L. Shujun, M. Xuanqin, and C. Yuanlong, “Pseudo-random bit generator based on couple chaotic systems and its applications in stream-cipher cryptography,” in *INDOCRYPT 2001: Progress in Cryptology — INDOCRYPT 2001*, 2001, pp. 316–329.
- [14] S. Ramesh, K. N. Haribhat, and R. Murali, “On linear complexity of binary sequences generated using matrix recurrence relation defined over  $Z_4$ ,” *International Journal of Distributed and Parallel systems*, vol. 1, no. 2, pp. 68–85, Nov. 2010, doi: 10.5121/ijdps.2010.1207.
- [15] S. Li, G. Chen, K.-W. Wong, X. Mou, and Y. Cai, “Baptista-type chaotic cryptosystems: problems and countermeasures,” *Physics Letters A*, vol. 332, no. 5–6, pp. 368–375, Nov. 2004, doi: 10.1016/j.physleta.2004.09.028.
- [16] P. G. S., R. S., and A. Raganna, “Voice encryption with watermarking for secure speech communication,” *International Journal of Emerging Technologies and Innovative Research*, vol. 6, no. 1, pp. 406–414, 2019.
- [17] Z. Su, J. Jiang, S. Lian, D. Hu, C. Liang, and G. Zhang, “Selective encryption for G.729 speech using chaotic maps,” in *2009 International Conference on Multimedia Information Networking and Security*, 2009, pp. 488–492, doi: 10.1109/MINES.2009.89.
- [18] M. Ashtiyani, P. M. Birgani, and S. S. K. Madahi, “Speech signal encryption using chaotic symmetric cryptography,” *Journal of Basic and Applied Scientific Research*, vol. 2, no. 2, pp. 1678–1684, 2012.
- [19] S. NajimAlSaad and E. Hato, “A speech encryption based on chaotic maps,” *International Journal of Computer Applications*, vol. 93, no. 4, pp. 19–28, May 2014, doi: 10.5120/16203-5488.
- [20] H. Kohad, V. R. Ingle, and M. A. Gaikwad, “Security level enhancement in speech encryption using kasami sequence,” *International Journal of Engineering Research and Applications (IJERA)*, vol. 2, no. 4, pp. 1518–1523, 2012.
- [21] A. Servetti and J. C. De Martin, “Perception-based partial encryption of compressed speech,” *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 8, pp. 637–643, Nov. 2002, doi: 10.1109/TSA.2002.804300.
- [22] Q.-H. Lin, F.-L. Yin, T.-M. Mei, and H. Liang, “A blind source separation based method for speech encryption,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 53, no. 6, pp. 1320–1328, Jun. 2006, doi: 10.1109/TCSI.2006.875164.
- [23] E. Mosa, N. W. Messiha, and O. Zahran, “Chaotic encryption of speech signals in transform domains,” in *2009 International Conference on Computer Engineering & Systems*, Dec. 2009, pp. 300–305, doi: 10.1109/ICCES.2009.5383252.
- [24] E. Mosa, N. W. Messiha, O. Zahran, and F. E. Abd El-Samie, “Chaotic encryption of speech signals,” *International Journal of Speech Technology*, vol. 14, no. 4, pp. 285–296, Dec. 2011, doi: 10.1007/s10772-011-9103-7.
- [25] L. J. Sheu, “A speech encryption using fractional chaotic systems,” *Nonlinear Dynamics*, vol. 65, no. 1–2, pp. 103–108, Jul. 2011, doi: 10.1007/s11071-010-9877-1.
- [26] M. Ahmad, B. Alam, and O. Farooq, “Chaos based mixed keystream generation for voice data encryption,” *International Journal on Cryptography and Information Security*, vol. 2, no. 1, pp. 39–48, Mar. 2012, doi: 10.5121/ijcis.2012.2104.
- [27] P. Sathiyamurthi and S. Ramakrishnan, “Speech encryption algorithm using FFT and 3D-Lorenz–logistic chaotic map,” *Multimedia Tools and Applications*, vol. 79, no. 25–26, pp. 17817–17835, Jul. 2020, doi: 10.1007/s11042-020-08729-5.
- [28] L. E. Bassham et al., *A statistical test suite for random and pseudorandom number generators for cryptographic applications*. National Institute of Standards and Technology, Gaithersburg, MD, 2010.




- [29] V. Patidar, K. K. Sud, and N. K. Pareek, "A pseudo random bit generator based on chaotic logistic map and its statistical testing," *Informatica*, vol. 33, no. 4, pp. 441–452, 2009.
- [30] M. François and D. Defour, "A pseudo-random bit generator using three chaotic logistic maps," Research Report, LIRMM (UM, CNRS), 2013.
- [31] S. B. Sadkhan and R. S. Mohammed, "A proposed voice encryption based on random Lorenz map with DCT permutation," *International Journal of Advancements in Computing Technology*, vol. 7, no. 3, 2015.
- [32] C. Camara, H. Martín, P. Peris-Lopez, and M. Aldalaien, "Design and analysis of a true random number generator based on GSR signals for body sensor networks," *Sensors*, vol. 19, no. 9, Apr. 2019, doi: 10.3390/s19092033.
- [33] NIST, "Security requirements for cryptographic modules," FIPS, PUB 140-2, 2001.

## BIOGRAPHIES OF AUTHORS



**Pushpalatha Gopalakrishna Saraswathy**    is a research scholar, doing her research in the field of cryptography and network security in the Department of Electronics and Communication Engineering at the research center Dr Ambedkar Institute of Technology, Visvesvaraya Technological University. She has 15 years of experience in teaching. She received her B.E degree in telecommunication engineering from Sri Siddhartha Institute of Technology, Karnataka, India in 2006, and her M.Tech. degree in VLSI and embedded systems from Visvesvaraya Technological University, Belgaum, India in 2014. Her research areas also include signal processing, chaos cryptography, and digital communication. She can be contacted at pushpalathags.ec@drait.edu.in.



**Ramesh Siddaiah**    is a professor and head of the Department of Electronics and Communication Engineering, Dr Ambedkar Institute of Technology, Bengaluru, India. He has more than 3 decades of experience in teaching. He received his B.E degree in electronics and communication engineering from Gulbarga University, Karnataka, India in 1990, M.Tech. degree in industrial electronics from Visvesvaraya Technological University, Belgaum, India in 2001, and Ph.D. degree from Dr. MGR Educational and Research Institute, Chennai, India in 2013. His research areas include analog communication, digital communication, cryptography, and VLSI design. He has contributed to his research areas by publishing more than 25 journals. He can be contacted at rameshs.ec@drait.edu.in.