International Journal of Security and Its Applications
Vol. 11, No. 4 (2017), pp.1-10
http://dx.doi.org/10.14257/ijsia.2017.11.4.01

DNA and Blum Blum Shub Random Number Generator Based Security Key Generation Algorithm

Gurpreet Kour Sodhi[1] and Gurjot Singh Gaba[2*]

[1, 2]Discipline of Electronics & Communication Engineering,
Lovely Professional University, Jalandhar, India – 144411
[1]gurpreetsodhi123@gmail.com, [2]er.gurjotgaba@gmail.com

Abstract

The communication sector suffers from serious security related threats. The technique presented in this paper is a step towards providing security to the confidential data. The work is based on generating a unique key through DNA and random number generator. Each random sequence produced by Blum Blum Shub random number generator is the result of a seed value which makes the sequence unique and reproducible. The final security key obtained is unique and upon being tested using NIST randomness evaluation tests, it is concluded that the key generated retains its uniqueness to a high level, thus providing efficient security mechanism. Thus, it is possible to integrate it with any security system used in communication sector.

Keywords*: Authentication, Blum Blum Shub random number generator, DNA, security key*

1. Introduction

Security is the major area of consideration when it comes to communication. Integrity refers to trust in the data received. To perform authentication, parties must use unique and secret keys to prevent provision of access to the unintended parties. Biometrics ensures the identification and authentication of an individual on the bases of human body attributes [1]. The introduction of biometrics in data security has gained significance because of the potential threats to the confidential data. There are various security systems evolved in the past, which work on iris [1], audio fingerprint [2], personal signatures [3] and facial features [7] to generate secret keys. The use of the electrocardiogram (ECG) signals for security key generation has also been carried out in the past years [4-6].

Researchers have exploited the uniqueness of DNA and have used this property of DNA in cryptographic algorithms [8]. Blum Blum Shub random number generator (BBSG) has been used to increase the efficiency of the proposed algorithm. This generator works on a seed value which is useful in ensuring the generation of a different key for every seed value used. The work reported in this paper is based on the uniqueness of an individual's DNA and the randomness associated with the output sequence generated by Blum Blum Shub random number generator. The proposed algorithm is tested through NIST tests of randomness as well as the strict avalanche criterion, the results of which are formulated in Table 4.

The paper is organized as follows: Characteristics of DNA & Blum Blum Shub random number generator is described in Section 2. In Section 3, the proposed algorithm for the 256-bit key generation is presented where the DNA values are taken from MIT-BIH database [9]. Section 4, consists of analysis of purposed techniques and its comparison with counterparts. Finally, conclusions and future work are reported in Section 5.

* Corresponding Author

ISSN: 1738-9976 IJSIA

2. Characteristics of DNA and BBSG

The great progress in the field of biotechnology makes the Deoxyribonucleic Acid (DNA) sequencing more effective. Many DNA sequences of various organisms have been successfully sequenced with higher accuracy [9]. Analyzing DNA sequences investigates the biological relationships of different species. However, the analysis of DNA sequences using the biological methods is too slow for processing. Therefore, the assistance of computers is necessary and thus bioinformatics has been extensively developed.

On the other hand, many distributed databases have been constructed and can be easily accessed from the World Wide Web [10-11]. Most of the techniques consider the DNA sequences of an individual as the symbolic data, which is composed of four characters A, G, C, and T corresponding to the four types of nucleic acids: Adenine, Guanine, Cytosine, and Thymine, respectively. However, the bimolecular structures of genomic sequences can be represented in both symbolic as well as numeric form. DNA is made up of two polymeric strands composed of monomers that include a nitrogenous base (A-adenine, C-cytosine, G-guanine, and T-thymine), deoxyribose sugar and a phosphate group. The sugar and phosphate groups, which form the backbone of each strand, are located on the surface of DNA, while the bases are on the inside of the structure. Also, weak hydrogen bonds between complementary bases of each strand (*i.e*., between A and T and between C and G) give rise to pairing of bases, that hold the two strands together [8]. DNA sequences are unique for every individual, even for identical twins. The pattern formed by a DNA sequence specifically represents an individual and its characteristics. Hence, there is no chance of duplicity.

Further to strengthen the bond of security a random sequence is generated by Blum Blum Shub random number generator, using a secret value given by the user. This pseudo random generator takes the form:

$$\text{Random number} = \text{mod}((X_o)^2,\ M) \tag{1}$$

Where,

M: it's the product of two large primes 'p' and 'q'

X_0: is the seed value

The seed should be an integer that is co-prime to M (*i.e*. 'p' and 'q' are not factors of seed) and not equal to 0 or 1. The values taken for p & q are 383 and 503 respectively.

The random number from generator along with DNA sequence forms a very strong 256 bit key which is not only less susceptible to attacks but also provides a higher level of security.

3. The Working Principle

The suggested key is prepared by integrating the DNA sequence of an individual and BBSG random sequence.

3.1. DNA Sequence Formulation

The working principle of the suggested algorithm is explained in three subsequent subsections:

1) Obtaining a DNA sequence of 1024 characters from the DNA database. The DNA sequence consists of base pairs 'agct'.
2) Obtaining the binary sequence from DNA characters: Each character of the DNA sequence is represented by 8-bit ASCII code. Hence, resulting in a DNA sequence of length 8192 bits.
3) Framing a DNA sequence of 256 bits:

(i) The DNA sequence is then divided into equal halves.

(ii) Apply exclusive-or operation on the obtained sequences.

(iii) The result is further divided into two equal parts and exclusive-or operation is applied again.

The step (iii) is repeated till a sequence is obtained whose length is 256 bits. The whole procedure is summarized in the flow chart (Figure 1).

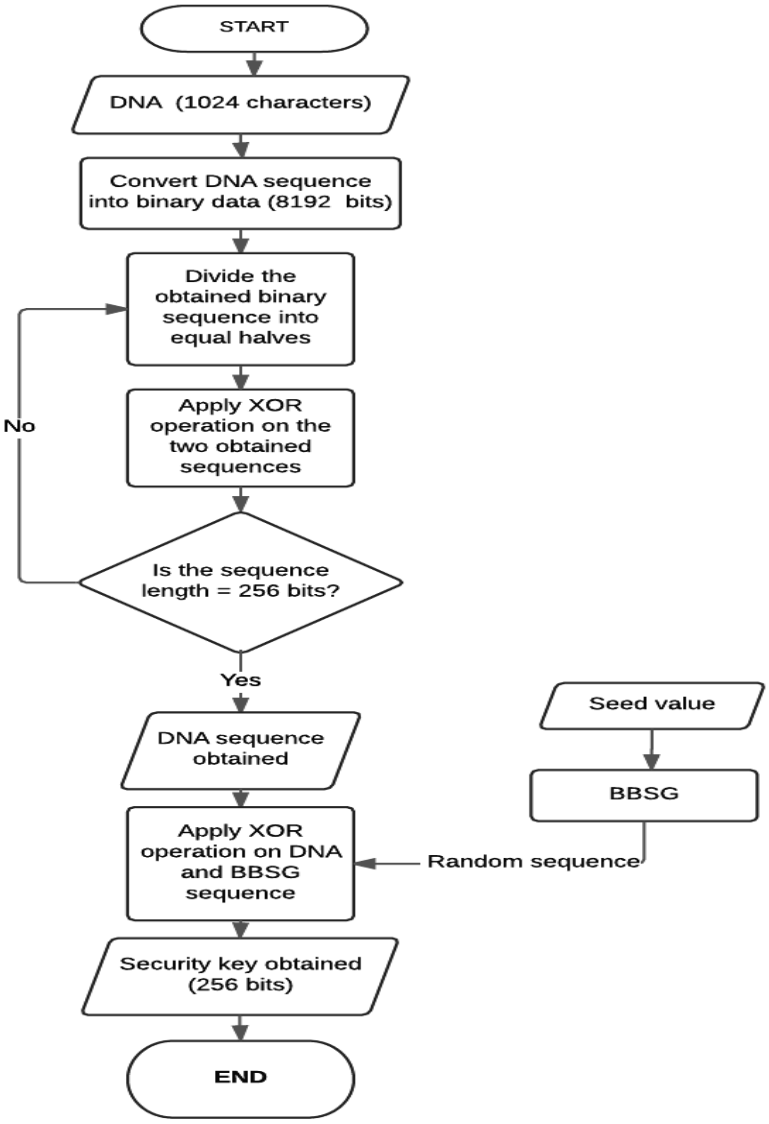

Figure 1. Key Generation Process

The algorithm is followed for three DNA data sequences and the results are provided in Table 1.

Table 1. DNA Sequence Formulation

	DNA data (1024 aracters)	Binary sequence (8192 bits)	DNA sequence (256-bits)
D_1	gcacaatcagaagcaggcggaggaga cggcggccttcgaggaggtcatgaag gacctgagcctgac..... ....gcacagaggca aggcgtcagcaggcatcgcccaccctg tctccgctgtcacccatcactcaggctgt agccatg[12]	011001110110001101100001011 000110110000101100001011101 0001100011…… ……..…… ……..…… ……..……..…… ……..……0111 010001100001011001110110001 101100011011000010111010001 100111	000001000000000000010101 000101110000001000000110 000101010000010000000110 000001100000011000000110 000100110000000000000000 000100110000000000000010 000001100000000000000110 000001100000010000000010 000100110000011000000010 000000000000010000000000 0001001100000000
D_2	ccacgcgtccgggcgagaagatggcg acttcgaacaatccgcggaaattcagcg agaagatcgcgct……..…… …… …… …… ……..... ....ggcgtcagccccct gtccctgagcacagaggcaaggcgtc agcaggcatcgcccgccctgtccccgc tgtcacccat [13]	011000110110001101100001011 000110110011101100011011001 1101110100.....…… …… ……..…… ……..…… ……..……011101000 110001101100001011000101100 011011000110110000101110100	000000100000001000000010 000000100000011000010111 000001000001011100000100 000000000000000000000000 000001100000001000000110 000101010001001100000010 000000000000000000000010 000100110000011000000000 000000100000011000000110 000000100000000000010111 0000010000000100
D_3	agcccttaggggaagagtcctgctctg gctgttgatgctccagctccagaaatcc cagtacctgcaactg…… …… …… …… ……..……..…… …… ……...……tctggagc agcagctgccctacgccttcttcaccca ggcgggctcccagcagccaccgccgc agccccagcccccgccg[14]	011000010110011101100011011 000110110001101110100011101 0001100001…… …… …… ……..…… ……..…… ……..……10001101 100011011000110110001101100 111011000110110000011 01100111	000101010000000000000010 000101110000010000010111 000000000000001000000100 000001100000000000000100 000101010000001000000100 000000000000000000010101 000000100000001000000000 000001000000000000010111 000000100001001100010011 000001100000000000010111 0001001100010011

***D_1, D_2, D_3:** DNA sequences

3.2. Generation of Random Sequence through Blum Blum Shub Generator (BBSG)

Three random sequences are generated using three respective seed values. The generated data is summarized in Table 2.

Table 2. Obtained Random Sequences from BBSG

	Seed value	Random sequences from BBSG (256 bits)
B_1	101355	110010100001001110111100100010000111010011110010110111001100111100 101010011011100101011000110010010001000010100101000000100000101000011 111010011000111010100110111010111011001011001100100110010000111001111 100110001101101110110010010110111011101110001001100110
B_2	101357	101110100001100110001000100111111010110101010001011111101110111110 001101101111110110011010100011111011111011110000011000110101111110 001000101010001011111101110100000001011111000000101111110000110001010 011001101000011100010110011111101010001110110001101100 1
B_3	101359	111000110010001010011101101011000000111011101011001110101110011101 001001110101101110110011011110111011110100000100101110000010111110 101011000000011011001010101111101111000111111111100010001001101111 001110001111010101010001011000110101101110100101100 1

***B_1, B_2, B_3:** Three random sequences.

Different seed values ensure the uniqueness of the sequences generated.

3.3. Fusion of DNA Sequences and BBSG Produced Random Sequences

Further Exclusive-or logic is applied between every DNA sequence and the Blum Blum Shub Generator produced random sequence. This is done to frame a final 256-bit key which can be used for providing security. This is repeated for two other sequences also. Finally, three 256-bit keys are obtained as shown in Table 3. These keys are represented in the paper as KEY_1, KEY_2, KEY_3.

Table 3. Security Keys

KEY_1	1100101000010011101111001000100001110100111100101101110011001111100 1 01010011011100101011000110010010001000010100101000000010000010100001 1 10100110001110101001101101011101100101100110010011001000011100111100 11000110110111011001001011011110111011100010011 00110
KEY_2	10111010000110011000100010011111101011010101000101111110111011111100 01101101111110110011010100011111011111011110000011000110101111110100 10001010100010111111011101000000010111111000000101111110000110001010 11 00110100001110001011001111110101000111011000110 11001
KEY_3	11100011001000101001110110101100000011101110101100111010111001111010 01001110101101110110011011110111101111010000010010111000001011111010 10110000000110110010101011111101111000111111111100010001001101111001 11000111101010101000101100011010110111010010110 01100

The keys are unique and random and thus can be useful in providing security in a data sensitive environment.

4. Results and Evaluation

The efficiency of a security key is based on two of its features, randomness and uniqueness. The National Institute of Standards and Technology (NIST) tests discuss some aspects of selecting and testing random number generators. The outputs of such generators can be used in many security related applications, such as the generation of security keys. For a random number generator to be used for security applications its output sequences must be unpredictable in the absence of the knowledge of the input. NIST tests are useful for determining if a generator is suitable to be used for a particular security application or not. The randomness of a key is evaluated on the bases of the P-value obtained, which is desired to be greater than 0.01 for a random sequence [15].

4.1. NIST Tests

A brief overview of the NIST tests used for evaluating the produced sequences is given as:

1) Runs test

This test is based on the calculation of the number of runs in the entire sequence, where a run specifies the number of uninterrupted sequence of identical bits [15].The results in Table 4 depict that the sequences generated using the proposed algorithm have higher rate of interruptions, and thus they are random in nature.

2) Frequency Test

Frequency test analyses the proportion of number of ones and zeros in the entire sequence. It checks the closeness between the number of ones and number of zeros. A sequence is said to be random if the proportion of both is close to each other [15]. The results in Table 4 show that the proposed algorithm produces better proximity between the count of ones and zeros as compared to the previously used techniques.

3) Approximate Entropy Test

The aim of this test is finding the frequency of all the overlapping bit patterns across the entire sequence. The purpose here is to compare the frequency of overlapping blocks of two consecutive / adjacent lengths with the expected result for a random sequence.

4) Discrete Fourier Transform Test (DFT)

The purpose of this test is to find the peak heights in the Discrete Fourier Transform of a sequence. It detects the periodic features (*i.e*. repetitive patterns) in the sequence which further indicates a deviation from the assumed randomness. The focus is to detect if the number of peaks exceeding the 95 % threshold are significantly different than 5%.

5) Binary Derivative Test

The Binary Derivative Test is performed using exclusive-or operation between successive bits until only one bit is left. Next, the ratio of number of ones to the length of entire sequence in each case is calculated. Finally, the average of the ratio of all the sequences is calculated, if the value lies near to 0.5, then the sequence is considered to be a random sequence [15]. The results in Table 4 indicate that the output of the proposed algorithm is random.

6) Maurer's "Universal Statistical" Test

This test emphasizes to detect if a sequence can be significantly compressed without any loss of information. The number of bits between matching patterns are calculated. A sequence which is significantly compressible is considered to be a non-random sequence. This test is also known as Universal test [15].

7) Random Excursion Variant Test

The test is used to find out the total number of times a particular state occurs in a cumulative sum random walk. The P-value specifies if the sequence is random or not. This test considers successive sums of the binary bits as a one-dimensional random walk [15]. The P-value for Random Excursion Variant Test is calculated using Equation (2).

$$P_{value} = erfc \times \frac{(|\xi(x) - j|)}{\sqrt{(2 \times j \times ((4 \times |x|) - 2))}} \quad (2)$$

Where,
erfc : the error function
ξ : the total number of times the state *x* occurs
x : the state occurred
j : the total number of cycles
The efficiency of the proposed technique is evaluated by comparing it with other traditional techniques used in the field of authentication and security. The tests have been performed on Key_1 and the results are compiled in Table 4.

Table 4. Comparison of Proposed & Traditional Techniques

S.No	Input Source of random number generator	Key Length (bits)	Runs Test	Frequency Test	Approximate Entropy Test	DFT Test	Binary Derivative Test	Maurer's Test	Random Excursion Variant Test
			P-value	P-value	P-value	P-value	P-value	P-value	
1	ECG [16]	128	0.1262	0.2487	0.5468	0.0294	0.0039	0.9428	Random
2	Image [17]	256	0.0809	0.8026	0.9759	0.4220	0.0113	0.9780	Random
3	Iris sequence[18]	128	0.1254	0.3768	0.9409	0.3304	0.0021	0.9062	Random
4	Fingerprint [19]	128	0.3345	0.3041	0.3345	0.7597	-	0.2757	Random
5	DNA & BBSG	256	0.0809	0.8026	0.8540	0.4220	0.4995	0.9601	Random

It is observed that the P-value obtained for the keys generated by the proposed algorithm, in all the seven tests, have significant values as compared to other techniques. Thus it can be concluded that the keys generated are random in nature and thus, fulfill the basic criteria for being used as security keys in a data sensitive environment.

Avalanche test has also been performed on these keys. The purpose of this test is to check the avalanche effect, which is a desirable property of the security keys. Wherein, if the input value is altered slightly, the output obtained changes significantly. It gives the percentage of bits flipped with the change in input. This is a significant property of security keys.

The Avalanche Test is performed on three sets of DNA and BBSG sequences:

Case 1: In the initial set, two security keys are generated through two DNA sequences while keeping the same BBSG sequence.

Case 2: The second set involves generation of two security keys through the same DNA sequence and two BBSG sequences.

Case 3: In the third set, two security keys are generated through two DNA and BBSG sequences.

Further, the avalanche effect is calculated for each of the three sets, this is done to know the amount of randomness the proposed technique produces on changing the input. The avalanche effect can be calculated using the formula given in equation (3).

$$Avalanche\ effect = \frac{No.of\ bits\ flipped\ in\ the\ sequence}{Total\ no.of\ bits\ in\ the\ sequence}\ x\ 100 \quad (3)$$

The result of avalanche effect on changing the inputs is summarized in tabular form, where D_1, D_2, D_3 represent the DNA sequences as taken from Table 1 and B_1, B_2, B_3 represent the BBSG sequences characterized by the seed values, as taken from Table 2. The result is summarized in Table 5 to Table 7.

Table 5. Avalanche Test Analysis: Case 1

DNA Sequences (D_n)	**Seed value**	**Blum Blum Shu Random sequences (B_n)**	**Key Generated K= D_n xor B_n**	**Avalanche Result of Key (K_n)**	
				No. of Bits Flipped	**Avalanche Effect**
D_1	101355	B_1	1100101000010011101111100100 0100001110100111100101101111 0011001111100101010011011100 0101011000110010010001000010 1001010000000100000101000011 1101001100011101010011011010 0111011001011001100100110010 0000111001111001100011011011 1101100100101101111011101110 0001001100110	58	22.65 %
D_2			1100110000010001101010111001 1110101110000111000111100110 1110111001001011100110001001 0110100011111001101110001011 0101001100100011101010000101 0011000111100100110110101100 1100000110110011011001010011 0001011001100011010011101101 1001010011111100111110101011 00010		

* *Refer* Table 1 for **D_1**, **D_2**, & Table 2 for **B_1**,
**Different DNA sequences- Same BBSG Sequence

Table 6. Avalanche Test Analysis: Case 2

DNA Sequences (D_n)	**Seed value**	**Blum Blum Shub Random sequence (B_n)**	**Key Generated K= D_n xor B_n**	**Avalanche Result of Key (K_n)**	
				No. of Bits Flipped	**Avalanche Effect**
D_1	101355	B_1	11001010000100111011110 01000100001110100111100 10110111001100111110010 10100110111001010110001 10010010001000010100101 00000010000010100001110 10011000111010100110110 10111011001011001100100 11001000011100111100110 00110110111011001001011 01111011101110001001100 110	118	46.09 %
	101357	B_2	10111100000110111001111 11000101010101001010000 00011011111111110011000 10011011001101101010101 01111110001011011100000 01010011011011110011110 00101010001101111101110 10001000100101000000111 11111010011100111011001 10100011110001001001110 11010001101100111111011 101		

* *Refer* Table 1 for **D_1** & Table 2 for **B_1**, B_2
**Same DNA sequences- Different BBSG Sequence

Table 7. Avalanche Test Analysis: Case 3

DNA Sequences (D_n)	**Seed value**	**Blum Blum Shub Random sequences (B_n)**	**Key Generated K= D_n xor B_n**	**Avalanche Result of Key (K_n)**	
				No. of Bits Flipped	**Avalanche Effect**
D_1	101355	B_1	11001010000100111011111001 00010000111010011110010111 01110011001111100101010011 01111001010110001100100100 01000001010010100000010000 01010000111010011000111010 10011011010111011001011001 10010011001000011100111100 11000110110111011001001011 01101111011101110001001100 110	120	46.87 %
D_2	101357	B_2	1011101000011001100010001 0011111101011010101000101 1111101110111111000110110 1111110110011010100011111 0111110111100000110001101 0111111010010001010100010 1111110111010000000101111 1000001011111100001100010 1011001101000011100010110 0111111010100011101100011 011001		

* *Refer* Table 1 for **D_1**,**D_2** & Table 2 for **B_1**, B_2
**Different DNA sequences- Different BBSG Sequence

The avalanche test results clearly conclude that a slight change in the inputs leads to a significant change in the output.

5. Conclusion

In the work proposed, a unique algorithm is presented to generate a security key, using DNA sequence of an individual and a random sequence generated through Blum Blum Shub generator. DNA is a unique characteristic of an individual and when it is collaborated with the Blum Blum Shub generator's output the resultant key becomes complex and efficient. DNA sequence, if used individually may result in a weak authentication technique as the DNA of an individual can be obtained even without making the person aware of it. Thus, the integration of Blum Blum Shub sequence with the DNA makes the security key a significant and powerful tool. The proposed security key finds its application in the security systems used in highly secure areas like nuclear plants, banks, military base, *etc.* The performance analyzed in NIST Tests concludes that the security keys are random and are highly recommended to be used. As a future work, other signals like audio, video *etc.* can be used as an input to the algorithm. There are other random number generators that can be used which may lead to more efficient results. Apart from that, the algorithm can be extended for longer security keys to prevent Brute force attacks.

References

[1] F. Hao, R. Anderson and J. Daugman, "Combining Cryptography with Biometrics Effectively", Technical Report No.640, UCAM-CL-TR-640, vol. 55, no. 9, **(2006)**, pp. 1081-1088.
[2] M. Covell and Baluja S, "Known-Audio Detection using Waveprint: Spectrogram Fingerprinting by Wavelet Hashing", ICASSP, vol. 1, **(2007)**, pp. 237-240.
[3] M. Covell and S. Baluja, "Audio Fingerprinting: Combining Computer Vision & Data Stream Processing", ICASSP, vol. 2, **(2007)**, pp. 213-216.
[4] S. Ktata, K. Ouni and N. Ellouze, "A Novel Compression Algorithm for Electrocardiogram Signals based on Wavelet Transform and SPIHT", International Journal of Signal Processing, vol. 5, no. 11, **(2009)**, pp. 253.
[5] S. A. Chouakri, Bereksi-R. F, S. Ahmaidi and O. Fokapu, "Wavelet denoising of the electrocardiogram signal based on the corrupted noise estimation", Univ. Djillali Liabes, Sidi Bel Abbes, Computers in Cardiology, **(2005)**, pp. 1021-1024.
[6] L. Brown and J. Seberry, "On the design of permutation P in DES type cryptosystems", Proceedings of the workshop on the theory and application of cryptographic techniques on Advances in cryptology, Springer-Verlag, vol. 434, **(2001)**, pp. 696-705.
[7] B. Chen and V. Chandran, "Biometric Based Cryptographic Key Generation from Faces", Queensland University of Technology, **(2007)**, pp. 394-401.
[8] H. T. Chang, C. J. Kuo, Y. D. Yunlin, N.-W. Lo, D. Taichung, Z. Wei and L. Hsinchu, "DNA Sequence Representation and Comparison Based on Quaternion Number System", vol. 3, no. 11, **(2012)**, pp. 39-46.
[9] A. L. Goldberger, L. Amaral, L. Glass, J. M. Hausdorff, P. Ch. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C. K. Peng, H. E. Stanley, "Physio Bank, Physio Toolkit and Physio Net: Components of a New Research Resource for Complex Physiologic Signals Circulation", **(2000).**
[10] Ensembl Genome Browser. http://www.ensembl.org/index.html.
[11] NCBI databases. http://www.ncbi.nlm.nih.gov/Entrez.
[12] NCBI Genbank, http://www.ncbi.nlm.nih.gov/nuccore/3327045?report=genbank.
[13] NCBI Genbank, http://www.ncbi.nlm.nih.gov/nuccore/20380066?report=genbank.
[14] NCBI Genbank, http://www.ncbi.nlm.nih.gov/nuccore/33874586?report=genbank.
[15] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenso, M. L. Vange, D. Banks, A. Heckert, J. Dray, S. Vo and L. E. Bassham III, "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications", **(2010)**.
[16] B. H. A. Garcia, A. V. Alarcon and O. Starostenko, "A Wavelet-Based 128-bit Key Generator Using Electrocardiogram Signals", **(2009)**, pp. 644-647.
[17] S. Hedayatpour and S. Chuprat, "Hash Functions-based Random Number Generator with Image Data Source", **(2011)**, pp. 69-73.
[18] W. Wei and Z. Jun, "Image encryption algorithm Based on the key extracted from iris characteristics", **(2013)**, pp. 169-172.
[19] L. Ying, Y. Jing, W. Shu and L. Xiao, "Design of A Random Number Generator from Fingerprint", vol. **(2010),** pp. 278-280.

Authors

Gurpreet Kour Sodhi, is currently pursuing her Masters in Electronics and Communication Engineering from Lovely Professional University. Her research area includes - 'Enhancing and Maintaining Security in Wireless Communication Systems' and 'Networks'. She is working in this field since 2015 and has potential to resolve several problems of industry through her expertise.

Gurjot Singh Gaba, is currently pursuing Ph.D. in Electronics & Electrical Engineering with Spl. in Cryptography and Network Security of WSN and IoT. He is working as an Asst. Prof. in Lovely Professional University, India since 2011. His research interest includes Wireless Sensor Networks, Computer Networks, Optical Communications and Cryptography. He is a reviewer of SCIE and Scopus Indexed Journals. He has recently been appointed as Editor of IJEEE journal. He is a member of many technical bodies including ISCA, IAENG, IACSIT, CSI, and ISTE. He is an author of six International books and more than two dozen research papers.

