Random Number Generators and their Applications: A Review

Priyanka[1], Imran Hussain[1], Aqeel Khalique[1]
[1] *Department of CSE, SEST, Jamia Hamdard, New Delhi 110062, INDIA*
priyanka18293@gmail.com
ihussain@jamiahamdard.ac.in
aqeelkhalique@gmail.com

***Abstract*—** In cryptography there are many types of algorithms which use random numbers in network security. To make the encryption and decryption process more robust, a certain level of randomness should be used to make it unpredictable. Random numbers are the sequence of numbers placed in such a way that the values present in the sequence should be uniformly distributed and they should be independent from each other. Random number generators are used in several industries to generate random values. These industries are not limited but include gaming, financial institutions, secure communication, cyber security and others. Now, random number generators are also finding their space in big data analytics. Random numbers can be classified as true random numbers and pseudorandom numbers. In this paper, we discuss the basic properties of random number generators, true-random numbers and pseudorandom numbers. Further, the paper also includes the comparative analysis of different random numbers generators and the algorithms used in them and a brief study about their characteristics, features, advantages and their usage.

***Keywords*—** *Random Number, Random Number Generator, TRNG, PRNG, True Random, Pseudorandom, Big Data Analysis*

I. INTRODUCTION

Random numbers are numbers whose sequence is uniformly distributed, and it is impossible to predict further upcoming values based on present sequence. These numbers are generated using some kind of mathematical algorithms which uniformly distributes all numbers preset in the sequence [1]. Random numbers have many applications but most important application of random numbers is in cryptography, where they are main part in encryption keys. The quality of the random numbers used in cryptography decides the security toughness of the system. If the quality of random number generator is good then it would be very difficult for anyone to break into the system. It has been proven difficult to break into the algorithms such as AES, RSA and ECC. Randomness and unpredictability are two major requirements for a sequence of random numbers [2]. Random numbers in cryptography are used in following applications:

- Private keys for digital signature algorithms
- PIN code and various password generation
- Values that are be used in protocols such as key establishment

In cryptography it is important to ensure that the secret keys are random, and are totally unpredictable, or in short they may follow the rules of randomness [3].

Random numbers are classified in two categories true random numbers and pseudorandom numbers. Random numbers are generated through different random number generators (RNGs). RNGs are just the application of randomness devices such as dice, shuffles cards and flipping coins. Nowadays, random number generators are implemented through programming based on deterministic computation, but this is not really taken as true random because the output that we get can be predicted if all seed values are known, so this is called pseudorandom number generation . An example of a RNG is a object or a device which reads radio noise and then extracts that value and hand it over to the user [4]. Applications that are based on true randomness are games such as bingo, card games, the lottery and similar games. Random number generators are classified as true random number generators (TRNG) and pseudorandom number generators (PRNG).

II. TYPES OF RANDOM NUMBER GENERATORS

Random numbers generators are classified into two categories true random number generators (TRNGs) and pseudorandom number generators (PRNGs).

A. True Random Numbers Generators

A true random number generator uses seed values or takes entropy source that are already present in ht environment it does not invent them. Entropy is known as the amount of unpredictability of the result. We can get entropy source from

the physical environment of the computer such as keystroke timing sample, disk electrical activity, movement of the mouse, and immediate values of the system clock and many more [5]. The single source, or the combination of these sources, is given as input to the algorithm which produces random binary output.

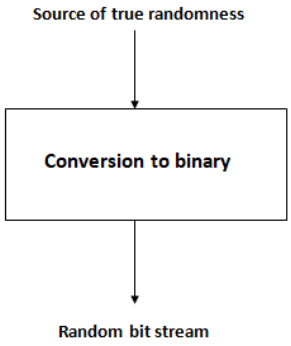

Fig. 1: True Random Number Generator

1) Feature of True Random Numbers

- It is generated by taking seed values from the entropy sources that are present in the physical environment [6].
- The defined set or the sequence is random. It has randomness.
    - It consists of uniformity that means that bits are distributed uniformly overall in the sequence, the rate of occurrence of ones and zeros are roughly equal.
    - It is independent which means no one subset in the defined set is deduced from the others.
- It is unpredictable; in a sequence each number is independent of other numbers in the sequence.
- Uses – Gambling games such as cards, security related algorithms.
- Examples - keystroke timing sequence, movement of the mouse.

2) Types of True Random Numbers Generators:

Four major true random generators:

a) Random.org: Random.org is the website which hosts widely used true random number generator. The random sequences generated by Random.org are freely distributed, leading to a varied user base. The entropy is collected from atmospheric noise by the random number generator from this site. Radio devices present in them picks up the noise and run it through a postprocessor which is then converted and results into a stream of binary ones and zeroes. It has been certified by several third parties that the number sequence on this site pass the industry-standard test suites, making it a free and feasible option for not regular or permanent users of random numbers [7].

b) HotBits: It is another popular random number generator based on internet. It generates the random number sequences based on radioactive decay. Just like random.org, random numbers we obtain from this generator are sent over the internet, so there is always the possibility that a third person has knowledge of the sequence. This makes it inappropriate for security purpose, but HotBits is useful when random data which cannot be doubted is needed.

c) Lasers: The use of TRNGs that overcome the problem of slow production. In this method entropy can be obtained by several different means. Having two photons race to a line of end is one method which is used nowadays. Another technique is measuring the varying intensity of a chaotic laser. Laser generators are capable of increased speeds, but they are difficult to install and it is expensive too. In practical applications it is not easy to imagine the use of laser-based generators.

d) Oscillators: Basic hardware is used in oscillators, which makes it more suitable for installation. It is a simple circuit which is obtained by placing an odd number of inverter gates in a loop. The final output of this layout is undefined, as the current oscillates in a sine wave pattern over time.

B. Pseudorandom Numbers Generators

A random number generator that does not depend on real world activities to produce their sequence is referred to as pseudorandom number generators. The sequences of numbers generated by PRNGs are similar to the properties of random numbers. This process is evaluated by a small group of initial values. Using seed state pseudorandom number generator starts from a random starting state. In a short interval of time many numbers are generated which can be generated later on, if the starting point in the defined set is known [8]. Thus, the numbers generated are efficient and deterministic. True random number generators convert entropy sources directly into sequences; a pseudorandom number generator needs to find entropy to keep itself unpredictable . By taking the time of day, the location or position of the mouse, or the activity on the keyboard we can achieve the entropy required for the PRNGs. By using the human interaction as entropy we can achieve this explanation of sources. Since there may be a probability that an attacker could manipulate into the system purposely to bias it. Hence, we glared out this approach in secure setting [9].

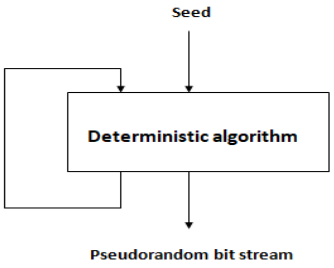

Fig. 2: Pseudorandom Number Generator

1) Features of Pseudorandom Numbers:

- It is generated through some mathematical algorithms.
- The defined set or the sequence is random. It has randomness.
  - It consists of uniformity which means that the frequency of occurrence of ones and zeros are equal approximately.
  - It is scalable i.e., if a particular sequence is random, then the subsequence extracted form is also random.
  - It is consistent.
- It is unpredictable.
  - Forward unpredictability – If the seed is not known to us then the next sequence should be unpredictable even if the previous sequences are known.
  - Backward unpredictability – The seed should not be easily determined from any generated sets or sequences.
- Seed can be generated from entropy source of any physical environment so it must be secure.
- Uses – Cryptography, security algorithms.

2) Types of Pseudorandom Numbers Generators:

In this section, we cover the types of algorithms for PRNGs.

- *Linear Congruential Generators (LCG)*

  Liner congruential generator is a simple example of pseudorandom number generator. It was developed by D. H. Lehmer in 1949 [10]. The formula for algorithm it uses is:

  $$X_{n+1} = (aX_n + c) \bmod m$$

  Where, X_o is the starting value $0 \leq Xo < m$
  m the modulus $m > 0$
  a the multiplier $0 < a < m$
  c the increment $0 \leq c < m$

  If suitable values are given to the parameters then it can produce a long random-like sequence. This algorithm tells the point that pseudorandom generators are deterministic. Initial X_o value is entropy for
  this generator, and≡ other chosen constants a, c and m likely remains same. An attacker can reconstruct the sequence if small number of values is given (knowing a, c, m).

- *Blum BlumShub Generator (BBS)*

  Blum BlumShub is a pseudorandom number generator proposed by Lenore Blum, Manuel Blum and Michael Shub in 1986 [11]. The BBS generator produces a sequence of bits using following expression:

  $$X_{n+1} = Xn^2 \bmod M$$

  Where, M = p*q, product of two large primes p and q.

  Xo should be an integer
  $Xo \neq 0$ and $Xo \neq 1$
  $P \equiv 3 \pmod 4$
  $q \equiv 3 \pmod 4$

- *Linear Feedback shift Register (LFSR)*

  In this shift register the input bit that is given is a linear function of the previous state [12]. It provides a easy and simple means for generating no sequential lists of numbers. Right- shift operation and XOR operation are only required for generating pseudorandom numbers. It can be implemented in hardware very easily and it also produces the sequence which remains for longer period. It consists of three elements: a shift register, a linear feedback function and a clock. Sequences of bits generated at that time are shift registers. Whenever we need output bit, the generator is shifted 1 position to the right by shifting all the other remaining bits. The function of other bits present in the register is generated from the new left most bits available. The output bit is in stage 0. XOR operation is the feedback function of bits present in the registers.

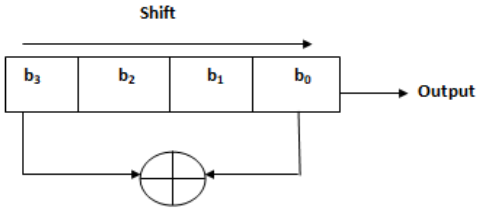

Fig. 3: Linear Feedback Shift Register

III. TEST AND RESULTS

This section consists of several test suites which are done on random binary numbers [13].

Frequency Test: This test checks the occurrence of ones and zeros in a sequence. For passing this test the occurrence of zeros and ones in a sequence should be approximately equal.

Frequency Test within a Block: This test checks the frequency of ones in an M-bit block. This test evaluates that for M-bit block the number of occurrence ones should be nearly M/2.

Runs Test: This test checks the overall runs in the defined set of the sequence. It evaluates the oscillation of ones and zeros in the defined set.

Test for the Longest Run of Ones in a Block: It checks for the longest run of ones in M-bit block.

Non-Overlapping Template Matching Test: It checks occurrence of already specified target sequence.

Maurer's "Universal Statistical" Test: It checks the number of bits between matching patterns.

Cumulative Sums Test: It checks the cumulative sum of bits in the sequence.

Overlapping Template Matching Test: It determines the number of occurrences of pre-specified target strings.

Binary Matrix Rank Test: It checks for linear dependence among fixed length substrings of the original sequence.

Discrete Fourier Transform Test: It detects periodic features in the tested sequence that would indicate a deviation from the assumption of randomness.
Linear Complexity Test: It determines whether or not the sequence is complex enough to be considered random.

Serial Test: The focus of this test is the frequency of all possible overlapping m-bit patterns across the entire sequence.

Approximate Entropy Test: The focus of this test is the frequency of all possible overlapping m-bit patterns across the entire sequence.

Random Excursions Test: The focus of this test is the number of cycles having exactly K visits in a cumulative sum random walk.

Random Excursions Variant Test: the focus of this test is the total number of times that a particular state is visited(i.e, occurs) in a cumulative sum random walk.

TABLE 1: NUMERICAL ANALYSIS OF TEST RESULTS

Binary Random Sequence	Frequency Test	Frequency Test Within a Block	Runs Test	Test for the Longest Run of Ones in a Block	Non-Overlapping Template Machine Test	Maurer's "Universal Statistical" Test	Cumulative Sums Test
10101010010101010010	✓	✓	✓	✓	✓	✓	✓
00010010100100101010	✓	✓	✓	✓	✓	✓	✓
10101010000010010101	✓	✓	✓	✓	✓	✓	X
10100010000100101010	✓	✓	✓	✓	✓	✓	X
01001010101001010000	✓	✓	✓	✓	✓	✓	✓
10100101010101010101	✓	✓	✓	✓	✓	✓	✓

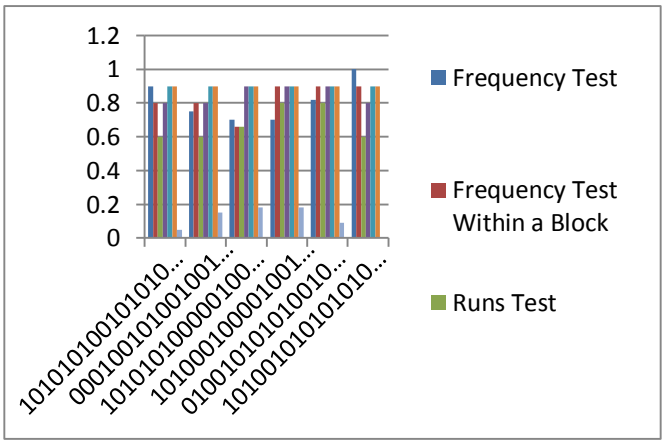

Fig. 4: Numerical Analysis of Test Results

IV. CONCLUSION

Random numbers are widely used in gaming field and security purpose. In this paper we introduced the types of random number generators and the basic properties, features and working of true random numbers and pseudorandom number generators. We also analyzed the algorithms through which random numbers are generated. The algorithms discussed in this paper can be easily implemented in various kind of computing devices. These algorithms are very simple and generate sequences in binary form and consist of randomness and unpredictability in it. Pseudorandom number generators are easy to implement and provide best results. The scope of this research can be utilized in various number of aspects of a random number generators.

ACKNOWLEDGEMENT

We thank our friends for so much of motivation and support for completing it successfully and presenting this paper.

REFERENCES

[1] Kumar, I. Cryptology. Laguna Hills, CA: Aegean Park Pres, 1997.
[2] Schneier, B. Applied Cryptography. John Wiley & Sons, 2nd Edition, 1996.
[3] Eastlake, D., Schiller, J., and Crocker, S. Randomness Requirements for Security. RFC 4086, June 2005.
[4] Knuth, D. The Art of Computer Programming, Volume 2: Seminumerical Algorithms. Reading, MA: Addison-Wesley,1998.
[5] Maxim integrated, https://www.maximintegrated.com/en/app-notes/index.mvp/id/4400, last accessed 2019/02/11.
[6] Stalling, W. Cryptography And Netwrok Security, 5th Edition.
[7]Random.org, https://www.random.org, last accessed 2019/02/11.
[8]Law, A., Kelton, D. Simulation modeling and analysis, 3rd Edition. AZ: McGrawHill Higher Education. 2000.
[9]Technopedia, https://www.technopedia.com/definiton/25842/pseudo-random-number-generator-prng, last accessed 2019/02/11.
[10] Lehmer, D. H. 1949. Mathematical methods in large-scale computing units. 2nd Symposium on Large-Scale Digital Calculating Machinery. pp. 141-146.
[11] Blum, L., Blum, M. and Shub M. 1986. A simple unpredictable pseudo-random number generator. SIAM Journal on Computing. Vol. 15.
[12] Klein, A. 2013. Stream Ciphers. Chapter 2: Liner Feedback Shift Registers.
[13] Rukhin, A., Soto, J., Smid, M., Barker, E., Leigh, S., Levenson, M., Vangel, M., Banks, D., Heckert, A., Dray, J. And Vo S. 2010. A statistical test suite for random and pseudorandom number generators for cryptographic applications. NIST Speical Publication 800-22.

Priyanka is a student of B.Tech. (Final Year) in Computer Science & Engineering at Jamia Hamdard, New Delhi. Priyanka has done many Java programming based projects. Priyanka has research interest in Big Data, Cloud Computing.

Dr. Imran is working as an Assistant Professor in the Jamia Hamdard. Dr. Imran major research is in the field of e-learning technologies which includes designing, development, implementation and administration of e-learning courses and its integration with open source e-learning tools.

Aqeel Khalique is Assistant Professor in Jamia Hamdard, New Delhi. Aqeel has done several researches in the area of Information Security, Pervasive Computing, Cloud Computing & Cryptography. Aqeel has completed his M.Tech. from IIT Roorkee and worked in IT and Software Development Companies.