A study on the effect of shifting on LFSR PRNG

Ankur Kumar Varshney[1], Sushil Kumar Sharma [2] and Rajesh Singh[3]
[1]M.Tech Student, Computer Science & Engineering
B.S.A College of Engineering & Technology, Mathura, India

[2]M.Tech Student, Computer Science & Engineering
B.S.A College of Engineering & Technology, Mathura, India

[3]Faculty Member, Department of Computer Science & Engineering
B.S.A College of Engineering & Technology, Mathura, India

Abstract

There are two kinds of the random number generator (RNG): truly random number generator (TRNG) and pseudorandom number generator (PRNG). The TRNGs are very unpredictable but difficult to handle because it's too sensitive to the changing environment. Random numbers are a fundamental tool in many cryptographic applications like key generation, encryption, masking protocols, or for internet gambling. So require generators which are able to produce large amounts of secure random number. LFSR is the most popular. It advantages depend on simple implementation and high speed performance. However, it has poor security in terms violability. A cryptographically secure PRNG with properties that make it suitable for use in cryptography. So a modified Linear feedback shift register (LFSR) is provided here. The experimental result is compared by so many measures such as time requires to generate the number, brute force attack, entropy, key sensitivity analysis, block frequency etc.

Keyword - *PN sequence, PRNG, Randomness*

1. INTRODUCTION

Random numbers are useful for a variety of purposes such as generating data encryption keys, simulating and modelling complex phenomena and for selecting random samples from larger data sets which are more explain in [1]-[4].Usually, random numbers are generated using software algorithms. Although the sequence of numbers they produce seems random, they are not truly random. It is difficult to program a series of logical steps that produce numbers that do not follow some definite sequence. These random numbers are called Pseudo random numbers.
Four random number generators implemented in are compared in terms of their distribution, independence, speed and period. Each generator is described and its code presented. The tests used to evaluate the generators were simple ones. The objective is to examine pseudo random number generators implemented and select one that exhibited good characteristics in several categories. A random number generator should most importantly generate numbers which are "close" to being uniformly distributed and where subsequent numbers generated "appear" to be independent from previous numbers generated. The generator should also be as fast and efficient as possible. Finally, the generator should have a sufficiently large period for the application in which it will be used. The amount of CPU time a generator uses in order to achieve randomness and the effort expended by the tester to test for randomness depends on what level of statistical randomness is required by the intended application. The random number generators were also timed to determine how fast their code executed and thus determine their average speed. Each generator was timed for 1,000 iterations. Finally, the reported periods of the random number generators are given. A random number generator's period is the length of the longest sequence of numbers generated before the sequence begins to repeat again. These

sequence periods are not independently proven, but simply taken from the documentation and assumed to be correct. With both the speed of a random number generator on a particular computer known and that random number generator's sequence period known, it is possible to calculate how much time it would take that generator to iterate through its period on that particular computer [5].

2. PSEUDORANDOM SEQUENCE GENERATOR

2.1 LFSR-based random number Generator

The most common way to implement a random number generator is LFSR. Codes generated by the LFSR are actually pseudo random sequences (PN) because the sequence repeats itself after a certain number of cycles. It is known as the period of the generator. It is based on the recurrence equation, $\mathbf{X_n=x_{n-1} \oplus x_{n-2} \oplus ....\oplus x_{n-m}}$ **(1)**
The operator $\oplus$ is the exclusive OR (XOR) operator. The equation shows that n^{th} bit can be generated utilizing *m* previous values with XOR operators [2]. The value of *m* determines the period of the generator. The maximum period is 2^m-1. For the 8-bit LFSR, the recurrence equation is, $\mathbf{X_n=x_{n-2} \oplus x_{n-3} \oplus x_{n-4} \oplus x_{n-8}}$ **(2)**
Since new value x_n depends on previous *m* values, it is necessary to store previous *m* values to find the new value. This can be done with *m* single bit shift registers comprised with flip flops. According to the equation (2), XOR feedback tap positions are taken at 0th, 4th, 5th and 6th flip-flops. The maximum period of the generator is 2^8-1 (255). In each clock pulse, generated new bit is inserted to the shift register while the oldest bit shifts out.

The linear feedback shift register is one of the most useful techniques for generating pseudo-random numbers [6]. In short, an LFSR takes a series of bits from a long shift register, XORs them together to come up with a resultant bit, shifts the register along one bit, and then sticks the new bit back into the beginning of the register. If the positions of the bits (the "taps") are chosen carefully, this produces a maximal-length string of bits which is as damn near random as makes no odds to anyone other than mathematicians and cryptographers [7]. Linear feedback shift registers (LFSRs) are used in many of the key stream or bit sequence generators that have been proposed in the literature. There are several reasons for this: LFSRs are well-suited to hardware implementation; they can produce sequences of large period; they can produce sequences with good statistical properties; and Because of their structure, they can be readily analyzed using algebraic techniques.

First, the implementation of most simple and common random number generator, Linear Feedback Shift Register (LFSR) is discussed [8]. This generator tends to fail basic requirement of a RNG due to high correlation in the sequence [9]. The correlation problem that can be reduced by modifying the LFSR random number generator is discussed next

3. PROPOSED SCHEME

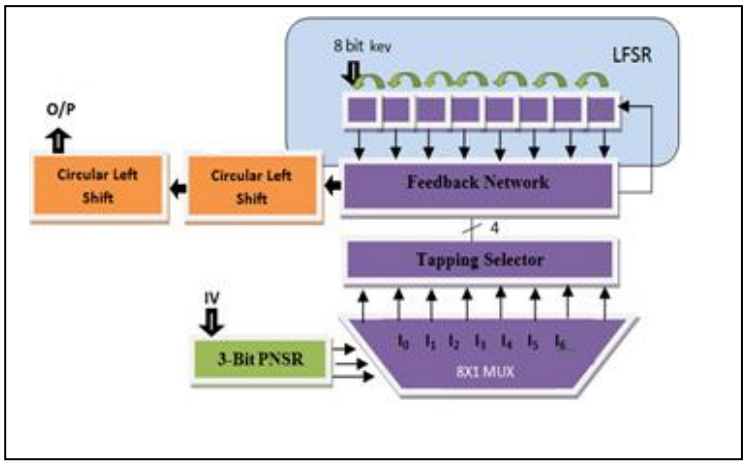
3.1 Modified Linear Feedback Shift Register

A **linear feedback shift register** is a register of bits that performs discrete **step** operations that

- Shift all the bits one position to the left and
- Replace the vacated bit by the XOR of the bit shifted off and the bit at a given ***tap*** position in the register.

A modification in simple LFSR PRNG is proposed here and shown in figure 1(a). The idea is based on increasing the randomness by using 8X1 MUX. Here 7 different tapping are chosen. These tapping are selected by taking certain thing into mind such as redundancy and randomness. These tapings are fed into tapping selector table. A 3bit PRNG is used to randomly select these tapings. The initial vector (i.e. IV) is chosen 001 because it will generate all possible number between 1 to 7. Case 0 is prohibited here because when the all zero condition arises it will generate on zero value onwards. The 3bit PN sequence generator gives a selection

to the tapping selector to choose a particular tapping among 7 selections shown in figure 1(b). This tapping selection is given to the feedback network [10].



(a)

Selection	Tapping Sequence
I_0	[8 7 6 1]
I_1	[8 7 5 3]
I_2	[8 7 3 2]
I_3	[8 6 5 4]
I_4	[8 6 5 3]
I_5	[8 6 5 2]
I_6	[8 7 6 2]

(b)

Figure 1(a) Proposed Technique, (b) Tapping Selector Table

3.2 Feedback Function

In an LFSR, the bits contained in selected positions in the shift register are combined in some sort of function and the result is fed back into the register's input bit. By definition, the selected bit values are collected before the register is clocked and the result of the feedback function is inserted into the shift register during the shift, filling the position that is emptied as a result of the shift.

The feedback function in an LFSR has several names: XOR, odd parity, sum modulo 2. Whatever the name, the function is simple: 1) Add the selected bit values, 2) If the sum is odd, the output of the function is one; otherwise the output is zero.

The bit positions selected for use in the feedback function are called "taps". The list of the taps is known as the "tap sequence". By convention, the output bit of an LFSR that is n bits long, the feedback tapping are kept changing which make the generated code quite complex[6].

4. EXPERIMENTAL RESULTS

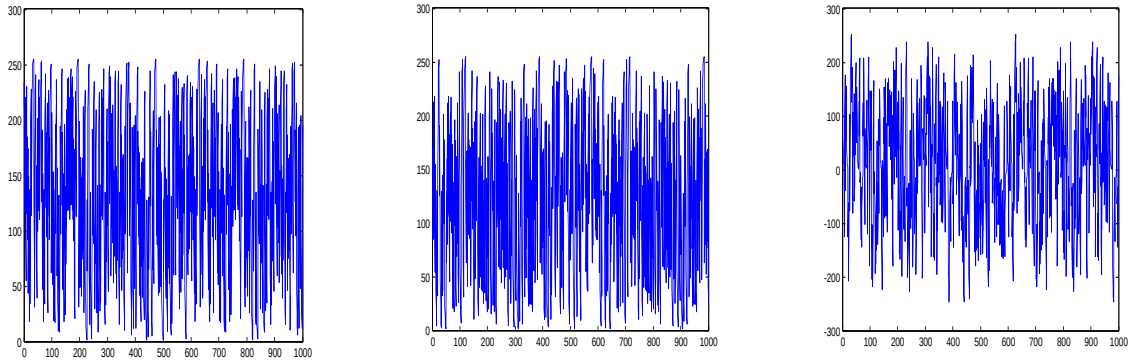
4.1 Brute force analysis

The key length used in the encryption determines the practical feasibility of performing a brute-force attack, with longer keys exponentially more difficult to crack than shorter ones. Brute-force attacks can be made less effective by obfuscating the data to be encoded, something that makes it more difficult for an attacker to recognize when he/she has cracked the code. One of the measures of the strength of an encryption system is how long it would theoretically take an attacker to mount a successful brute-force attack against it. The simple LFSR PRNG uses 8-bit key i.e. 2^8=256 combination must be used to break the key, while modified LFSR uses 8-bit key with 7 different taps which will raise the permutation to 256X256=65536.

4.2 Key Sensitivity Analysis

Sensitivity analysis is the study of how the uncertainty in the output of a model can be apportioned to different sources of uncertainty in the model input [11]. A related practice is uncertainty analysis which focuses rather on quantifying uncertainty in model output. Ideally, uncertainty and sensitivity analysis should be run in tandem. In this test the one bit of the key is changed and analysis on the output on the values of LFSR with 3 circular shifts is done. 1000

samples are generated by keeping a single bit change in the key. The fig 2 shows the difference between the values generating by changing the one bit in the key.



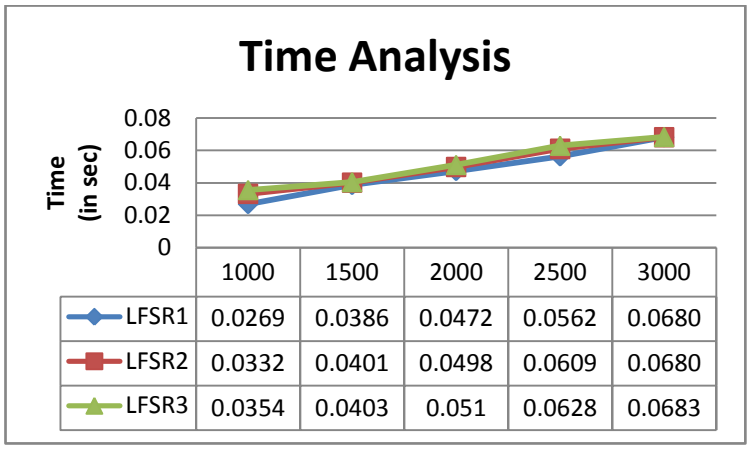
(a) (b) (c)

Figure:2 (a) PN sequence by using 8bit key, (b) PN sequence by using 8bit key with 1bit change, (c) difference of fig(a) and (b).

4.3 Time Analysis

The quality assessment of the technique is done to measure the amount of the time required to generate the data. Lower the time required means better technique. Figure 3 shows the time required to generate the PN sequence with simple LFSR, LFSR with 1bit circular shift (i.e. LFSR2) and LFSR with 2 bit circular shift (i.e. LFSR3).

.



	1000	1500	2000	2500	3000
LFSR1	0.0269	0.0386	0.0472	0.0562	0.0680
LFSR2	0.0332	0.0401	0.0498	0.0609	0.0680
LFSR3	0.0354	0.0403	0.051	0.0628	0.0683

Figure 3 time analysis of PN sequence generate by simple LFSR, LFSR with 1bit circular shift (i.e. LFSR2) and LFSR with 2 bit circular shift (i.e. LFSR3).

4.4 Test for the Longest Run of Ones in a Block

The focus of the test is the longest run of one's within M-bit blocks. The purpose of this test is to determine whether the length of the longest run of one's within the tested sequence is consistent with the length of the longest run of ones that would be expected in a random sequence. This test is performing on the 1000 sample of random numbers. Note that an irregularity in the expected length of the longest run of one's implies that there is also an irregularity in the expected length of the longest run of zeroes. Long runs of zeroes were not evaluated separately due to a concern about statistical independence among the tests.

4.5 Non-overlapping (A periodic) Template Matching Test

The focus of this test is the number of occurrences of pre-defined target substrings. The purpose of this test is to reject sequences that exhibit too many occurrences of a given non-periodic (a periodic) pattern. For this test and for the Overlapping Template Matching test, an m-bit window is used to search for a specific m-bit pattern. If the pattern is not found, the window slides one bit position. For this test, when the pattern is found, the window is reset to the bit after the found pattern, and the search resumes. This test is performing on the 1000 sample of random numbers.

4.6 Lempel-Ziv Complexity Test

The focus of this test is the number of cumulatively distinct patterns (words) in the sequence. The purpose of the test is to determine how far the tested sequence can be compressed. The sequence is considered to be nonrandom if it can be significantly compressed. A random sequence will have a characteristic number of distinct patterns. This test is performing on the 1000 sample of random numbers.

4.7 Test for frequency within a block

The focus of the test is the proportion of zeroes and ones within M bit blocks. The purpose of this test is to determine whether the frequency of one's is an M-bit block is approximately M/2. This test is performing on the 1000 sample of random numbers.

4.8 Random variant test

The focus of this test is the number of times that a particular state occurs in a cumulative sum random walk. The purpose of this test is to detect deviations from the expected number of occurrences of various states in the random walk. This test is performing on the 1000 sample of random numbers.

4.9 Entropy Analysis

Entropy is the randomness collected by an operating system or application for use in cryptography or other uses that require random data. This randomness is often collected from hardware sources, either pre-existing ones such as mouse movements or specially provided randomness generators. It provides 7.9917 entropy

5. CONCLUSION

PRNG is widely used method improving the controllability of random number. The PRNG can be implemented through software or hardware. For hardware implementation, LFSR is the most popular. It advantages depend on simple implementation and high speed performance. However, it has poor security in terms violability. The "quality" of the randomness required for these applications varies. For example creating a nonce in some protocols needs only uniqueness. On the other hand, generation of a master key requires a higher quality, such as more entropy. And in the case of one-time pads, the information-theoretic guarantee of perfect secrecy only holds if the key material is obtained from a true random source with high entropy with 7.9917. However, it has poor security in terms violability. For software implementation, most of users applied built-in random generator for their applications which is simple and fast. However, the randomness of generated pseudorandom numbers with simple LFSR approach is claimed not secure enough to be used for cryptography but The PRNG generated by using modified LFSR provides a good randomness. The brute force analysis is far much better than simple LFSR approach.

References

[1] M. Luby, Pseudorandomness and Cryptographic Applications, Princeton University Press, 1996.

[2] Random Number Generator (2011). Wikipedia website [Online]. Available:http://en.wikipedia.org/wiki/Hardware_random_number_generator.

[3] Jiang Hao, Li Zheying, "On the Production of Pseudo-random Numbers in Cryptography" in Journal Of Changzhou Teachers College of Technology, Vo1. 7 , No. 4, Dec. 2001.

[4] D. E. Knuth, "The Art of Computer Programming", Vol. 2: Seminumerical Algorithms. Reading, MA: Addison-Wesley, 1969.

[5] William N. Graham, "A Comparison of Four Pseudo Random Number Generators Implemented in Ada*", Applied Research Laboratories The University of Texas at Austin.

[6] Lala krikor, sami baba at al , "Image Encryption Using DCT and Stream Cipher", European Journal of Scientific Research ISSN 1450-216X Vol.32 No.1 (2009), pp.47-57.

[7] Ismet Ozturk, Ibrahim Sogukpinar, "Analysis and Comparison of Image Encryption Algorithms", World Academy of Science, Engineering and Technology 3 2005.

[8] Horowitz P., Hill W., "The Art of Electronics", 2nd Edition, Cambridge University Press (2001).

[9] Peter Martin, "An Analysis of Random Number Generators for a Hardware implementation of Genetic Programming using FPGAs and Handle-C", Technical Report, CMS-358 (2002).

[10] Shaziya Parveen, Shradha Parashar, Izharuddin," Security Techniques for Medical Signals", International Journal of VLSI and Signal Processing Applications, Vol. 1, Issue 2 , May 2011,(74-79) ,ISSN 2231-3133.

[11] Ascough II, J.C., T.R. Green, L. Ma, and L.R. Ahjua, "Key Criteria and Selection of Sensitivity Analysis Methods Applied to Natural Resource Models", 1USDA-ARS, Great Plains Systems Research Unit, Fort Collins, CO 80526.

Author bibliography

Authors:

Ankur Kumar Varshney: *was born in India in August, 1986. He has completed B.Tech in Computer Engineering from Aligarh college of Engineering and Technology, Aligarh India in 2008.Currently he is pursuing M.Tech in Computer Engineering from B.S.A College of Engineering and Technology, Mathura, India.*
His interest areas are Advanced Digital Signal Processing, Security Application, and Digital Image Processing.

Sushil Sharma: *was born in India in Jan, 1981. He has completed MCA from Kalicharan Nigam Institute of Technology, Banda India in 2005.Currently he is pursuing M.Tech in Computer Engineering from B.S.A College of Engineering and Technology, Mathura, India.*
His interest areas are Advanced Digital Signal Processing, Security Application, and Digital Image Processing.

Rajesh Singh: *was born in Uttar Pradesh, India in Oct, 1977.He has Completed B.Tech and M.Tech from Dr.B.R.A.U, Agra and K.S.O.U, Mysore, respectively. Currently he is Assistant Professor in the Department of Computer Engineering, B.S.A College of Engineering and Technology,Mathura, 281004, India.*
He has published more than 5 papers in National and International Conferences and Journals.
His area of research interests Data Mining, Digital Signal Processing.