Creation of New Chaotic Super Random Number Generators for Improving the Security of Chaotic Cryptography

[1]Rakesh Kumar Rai* [2]Rakesh Kumar Prajapati*
I.T.S Engineering College, Greater Noida, India*

***Abstract*-The importance of random number generator is very huge in the case of cryptography especially in the case of chaotic cryptography where we are using chaotic random numbers. We are generating random number and apply those numbers on the encryption key. While in this proposed paper we are trying to implements an extra parameter so that this will improve the random number generation process and with this procedure we are able to increase the security of chaotic cryptography. The introduction of this extra parameter will change our logistic equation of random number generator and after that a series of new random numbers will generate and that will enhance the security of chaotic cryptography**.

Key Words: RNG, PRNG, TRNG, DBRG, Cryptography.

I.Introduction

Random numbers are sets of digits (i.e., 0, 1, 2, 3, 4, 5, 6, 7, 8, 9) arranged in random order. Because they are randomly ordered, no individual digit can be predicted from knowledge of any other digit or group of digits.

A random number generator is a process that produces random numbers. Any random process (e.g., a flip of a coin or the toss of a die) can be used to generate random numbers. Stat Trek's Random Number Generator uses a statistical algorithm to produce random numbers. Random number generators (RNGs) which have been used for only military cryptographic applications in the past got expanding usage for typical digital communication equipment.

Almost all cryptographic systems require unpredictable values; therefore RNG is a fundamental component for cryptographic mechanisms. Generation of public/private key pairs for asymmetric algorithms and keys for symmetric and hybrid cryptosystems require random numbers. The onetime pad, challenges, nonces, padding bytes and blinding values are created by using truly random number generators (TRNGs) [1]. Pseudo-random number generators (PRNGs) generate bits in a deterministic manner. In order to appear to be generated by a TRNG, The pseudorandom sequences must be seeded from a shorter truly random sequence [2].

A general way to design a chaotic stream cipher is to generate a random bit stream using chaotic system. In this paper, we propose a novel random bit generator through the cross coupling of two chaotic systems which can be used in the design of a new chaotic stream cipher as well as in other engineering applications, where random bit sequences are required[3].

In recent years, a class of algorithms called chaotic PRNGs has appeared in the literature. Those chaotic PRNGs originate from physics. Their state s is a real in the interval [0 : 1], their output bit is computed as a function of the state. In the simplest case, if $s \geq 0.5$, then the output is 1, else the output is 0. The state transition function is a function $f : [0 : 1] \rightarrow [0 : 1]$ that exhibits chaotic behavior. Because of this chaotic behavior, the output is assumed to have desirable statistical properties.

We have implemented several chaotic PRNGs and computed their period experimentally. We found that in all cases, the period lengths are much shorter than expected. Thus, despite the desirable mathematical properties of the chaotic functions, their use in floating point based implementations of PRNGs cannot be recommended. Our analysis of the logistic map partly explains the experimental results.

The seed is a number that controls whether the Random Number Generator produces a new set of random numbers or repeats a particular sequence of random numbers. If the text box labeled "Seed" is blank, the Random Number Generator will produce a different set of random numbers each time a random number table is created. On the other hand, if a number is entered in the "Seed" text box, the Random Number Generator will produce a set of random numbers based on the value of the Seed. Each time a random number table is created, the Random Number Generator will produce the same set of random numbers, until the Seed value is changed.

A random sequence of numbers can be viewed as a sequence of independent uniform stochastic

variables. There are many ways to generate random numbers (RNGs). The most common methods use thermal actuations as a source. These systems typically have man degrees of freedom. Computers can generate pseudo-random numbers (PRNGs), but most of these algorithms require a seed (a starting value). This will eventually lead to repetition and has the disadvantage that the same sequences will always be generated from the same seed[4].

II. Importance of random number generators

The technique of cryptography is always very important in data authentications, entity authentication, data integrity and confidentiality. In recent years, the cryptographic schemes have suggested some new and efficient ways to develop secure encryption. These schemes have typical structure which performed the permutation and the diffusion stages alternatively. However, most of algorithms be faced with some problems such as the lack of robustness and security. The random number generators are intransitive in cryptography for generation of cryptographic keys, allegorically, secret keys utilized in symmetric cryptosystems and large numbers is intransitive in asymmetric cryptosystems because of unpredictable, should better be generated randomly.

Random number generators can be classified into three classes which are pseudorandom number generators (PRNGs), true random number generators (TRNGs) and hybrid random number generators (HRNGs). PRNGs use deterministic processes to generate a series of outputs from an initial seed state. TRNGs use of non-deterministic source (i.e., the entropy source), along with some processing function (i.e., the entropy distillation process) to generate the random bit sequence.

Random number generators (RNGs) are useful in every scientific area which uses Monte Carlo Methods [5].It is difficult to imagine a scientific area where Monte Carlo methods and RNGs are not used. Extremely important is the application of RNGs in cryptography for generation of cryptographic keys and random initialization of certain variables in cryptographic protocol.

III.Randomness in Cryptography

A pseudorandom number generator (PRNG), also known as a deterministic random bit generator (DRBG) is an algorithm for generating a sequence of numbers that approximates the properties of random numbers. The sequence is not truly random in that it is completely determined by a relatively small set of initial values, called the PRNG's state, which includes a truly random seed. Although sequences that are closer to truly random can be generated using hardware random number generators, pseudorandom numbers are important in practice for their speed in number generation and their reproducibility, and they are thus central in applications such as simulations (e.g., of physical systems with the Monte Carlo method), in cryptography, and in procedural generation. Good statistical properties are a central requirement for the output of a PRNG, and common classes of suitable algorithms include linear congruential generators, lagged Fibonacci generators, and linear feedback shift registers. Cryptographic applications require the output to also be unpredictable, and more elaborate designs, which do not inherit the linearity of simpler solutions, are needed. More recent instances of PRNGs with strong randomness guarantees are based on computational hardness assumptions, and include the Blum Blum Shub, Fortuna, and Mersenne Twister algorithms.

One tries to apply this notion to generate pseudo-random number generators (PRNGs) because random numbers are widely used not only in cryptography and Monte Carlo applications but also in less obvious applications (Pecora et al. 1990; L'Ecuyer 1994; Kocarev & Parlitz 1995; Hidalgo et al. 2001; Fernandez et al. 2003). We mention just a couple of them. (i) In spread spectrum techniques, a binary signal is mixed with a random number sequence, to spread the spectrum over a wider frequency range.

Using different random number sequences, it is possible to share a communication channel among several users (Mazzini et al. 1997; Dinan & Jabbari 1998; Shan et al. 2006; De Micco et al. 2007). Reduction of electromagnetic interference is another important benefit of the spread spectrum effect (Setti et al. 2000; Callegari et al. 2002). (ii) Consider a low-frequency signal immersed in a high frequency digital noise. Sampling at time intervals defined by a

random number sequence, the resultant signal becomes filtered without using any coil or capacitors that are expensive, especially in power systems.

Truly random numbers are not attainable from computers, and it is unlikely that we will ever be able to get them from 'natural' sources, since one commonly assumes that any system is governed by underlying physical rules and consequently it is deterministic. A successful strategy to build up a PRNG is to start with the time series of a simple nonlinear chaotic map and to apply to it an adequate randomizing procedure so as to 'heighten/boost' its stochastic nature. Such strategy requires a quantitative evaluation of the improvement achieved after effecting the procedure. González et al. (2005) used the statistical complexity measure originally proposed by López-Ruiz et al. (1995) and later modified by Lamberti et al. (2004) to quantify the effectiveness of such randomizing modus operandi when applied to a Lorenzian chaotic system. It was also shown there that a widely employed course of action—the mixing of two chaotic signals—is not effective in this respect, contrary to what one might expect.

A new pseudo-random number generator (PRNG) based on a modified logistic map was proposed by Liu [6]. Based on this PRNG, a chaotic stream cipher was designed. Further, a chaotic random number generator was developed by Wang et al. [7] and realized it by an analog circuit. In 2006, Wang et al. [8] proposed a pseudo-random number generator based on z-logistic map, where the binary sequence through the chaotic orbit was realized under finite computing precision.

IV.Super Random Number Generator

In the number generation process initially we are generating numbers using logistic map function and this Logistic map is perhaps the simplest example of a discrete chaotic dynamical system that exhibits chaotic behavior. This map models the population growth [9] and can be expressed as:

$$X_{n+1} = \lambda X_n(1- X_n)$$

Where Xn and λ are the initial condition and system parameters respectively. Where λ varies from 1 to 4.

But after that we want to increase the security of our encryption algorithm for that purpose we are introducing an extra parameter and this parameter will be applied in random number generations and these chaotic random numbers will applied in chaotic cryptography to enhance our security at one level, so that our encryption algorithm will be more secure as compared to previously given logistic map random number generation process.

So the introduction of this extra parameter can generate new series of random numbers this series of new numbers are known as chaotic random numbers and the process of this number generation is known as random number generation. So the logistic equation in chaos theory is given below

$$f(x_n)=4*x_{n-1}*(1-x_{n-1})$$

and this logistic equation can generate a series of numbers from $x_0,x_1,x_2\ldots\ldots\ldots\ldots\ldots\ldots,x_n$
$x_1 =f(x_0)$ Similarly $f(x_n)=f(x_{n-1})$
here x_0 is the first number and x_n is the nth number

But we want to modify this equation and we generate a new series of numbers these numbers are known as series of super random numbers and the technique for this purpose is known as super chaotic number generators and for that purpose the equation will modify for the next round. So modified for nth random number will be given below

$$f(x_n)=f(x_{n-1})*b+(x_{n-1})*b$$

x_n is the nth number and b is an extra parameter
So now below is the detail of the random numbers which are generated from both the equation

1. Results from unmodified equation

$$\text{i.e } f(x_n) =f(x_{n-1})$$

For input $x_0=0.1$ $x_1,x_2,\ldots\ldots\ldots\ldots..,x_{10}$ are following

0.09000000000000001
0.08190000000000001
0.07519239000000001
0.06953849448608791

0.06470289227069623
0.060516428002502905
0.05685418994432079
0.05362179103009588
0.05074649455682061
0.04817128784701519

For input x_0=0.2 $x_1,x_2,$…….....x_{10} are following

0.16000000000000003
0.13440000000000002
0.11633664
0.1028024261935104
0.09223408736223825
0.08372696049069325
0.07671675657768315
0.07083129583788365
0.06581422336780986
0.06148271137030189

And the result from modified equation

i.e $f(x_n)=f(x_{n-1})*b+(x_{n-1})*b$

For x_0=0.1 and parameter b=0.5 x_1, $x_2,x_3,$……,x_{10} are following

0.09000000000000001
0.08597500000000001
0.08229951234374999
0.07892966826320519
0.0758286492162171
0.07296532945567154
0.07031321874110555
0.06784963120699389
0.0655550262809923
0.06341248170939905

For x_0=0.2 and b=0.5 $x_1,x_2,x_3,$……..,x_{10} are following

0.16000000000000003
0.1476
0.13696956000000002
0.1277691979443516
0.11973458011060908
0.11265992746861916
0.10638390377227734
0.10077878423174194
0.09574239616709067
0.09119216227441952

For x=0.1 and b=1 we will get the same result as like the previous equation which is a unmodified equation so we can say that it this equation a general condition for the above given equation.

V.Conclusion

We have proposed a new super chaotic random number generator that will generate new set of random numbers those numbers are created by applying extra parameter so that the key will modified by those numbers and this key will provide more better encryption and decryption technique so that the security of chaotic cryptography will increase. So in this paper we are able to get our objective as previously we have proposed about an extra parameter.

References

[1] Jun B, Kocher P. 1999.The Intel random number generator. Cryptography Research, Inc. white paper prepared for Inter Corp.

[2] Menezes A.Van Oorschot P & Vanstone S. 1996. Handbook of applied cryptology. Boca Raton, FL: CRC Press.

[3] Sandhu G. S. & Berber S. 2009.Theoretical model, simulation results and performances of a secure Chaos-based multi-user communication system. International Journal of Network Security, 8(1): 25-30.

[4] Wichmann B. & Hill I. 2006. Generating good pseudorandom numbers. Computational Statistics &
Data Analysis, 51 (3): 1614 -1622.

[5] Niederreiter H. 1992. Random Number Generation and Quasi-Monte Carlo Methods. Philadelphia, PA: SIAM.

[6] Liu J. 2005.Design of a chaotic random sequence and its application. Computer Engineering, 31(18):150-152.

[7] Wang Y. Shen H& Yan X. 2005.Design of a chaotic random number generator. Chinese Journal of Semiconductors,26(12): 2433-2439.

[8] Wang L.Wang F. P& Wang Z. J. 2006.Novel chaos- based pseudo-random number generator. Acta Physica Sinica, 55(8):3964-3968.

[9] May R.M. Nature 261 (1976): 459.