Dynamic Random Number Generator based on User Seed(s)

Saleh N. Abdullah, Ph.D
Assistant Prof., Khawlan College, Sana'a University, Yemen.

Sharaf A. Alhomdy , Ph.D
Assistant Prof. & Vice-Dean, Faculty of Computer and Information Technology, Sana'a University,Yemen.

ABSTRACT
Random number generators (RNGs) is an underlying technology to accomplish highly secure systems. Therefore, for any security or simulation, systems should be associated with RNGs. Many of RNGs are currently in use, but the main defects in the available RNGs are the short period of its repeat cycle length and the predefined values of static factors as well. In this paper, we will try to suggest a method to extend the periodic cycle of the repetition and to use dynamic factors instead of static factors based on the seed values for the sake of security enhancement.

Keywords
Seed, Period, Static Factors, Dynamic Factors, RNGs, Security, Simulation, LCM, Linear Congruential, Uniformity, Independence.

1. INTRODUCTION
The random number generators (RNGs) are useful for a variety of purposes, such as [1]:

a) Generating data encryption keys.

b) Simulating and modeling complex phenomena.

c) Selecting random samples from larger data sets.

d) Testing problem generation for the performance evaluation of computer algorithms.

e) Statistically sampling, and so on.

Moreover, nowadays, many of daily activities in security need RNGs to perform their tasks. For example, secret and public-key generation and challenge-response authentication require unpredictable random numbers. Therefore, despite the large amount of theoretical research already done on this subject, many of the generators currently in use, for example the RNGs are an underlying technology to accomplish highly secure systems. But the main defects in the available RNGs are [1, 2]:

The short periods of its repeat cycle length.

1) Sometimes such available RNGs do not satisfy the desire needs for specific applications.

2) The predefined values of static factors may reduce the associated security.

In this paper, we will try to suggest a new method to extend the periods of repeat cycle length to generate random numbers. The method can be used for the specific applications by determining the number of digits. It also uses dynamic factors based on the seed value(s) to enhance the associated security. The paper has been organized in a flexible manner. Section II focuses on the background related to the previous work. Section III explains a proposal method to enhance the security produced. Section IV discusses the analysis & complexity for the proposed method and the experimental result shown in section IV. Conclusion & future research assignments will be highlighted in section V.

2. BACKGROUND
Indeed, all RNGs are based upon specific mathematical algorithms, which are repeatable and sequential. As such, it would be satisfying to generate random numbers from a process that is according to well-established understanding. So, to be useful in simulation, a sequence of random numbers R_1, R_2 must have two important properties: uniformity and independence. That is, each random number R_i is an independent sample drawn from a continuous uniform distribution between 0 and 1(mean 1/2, standard deviation). Some consequences of the uniformity and independence properties are as follows [2, 3]:

a) Uniformity: if the interval [0, 1] is divided into n sub-intervals of equal length, the expected number of observations in each interval N/n, where N is the total number of observations. The distribution of numbers in the sequence should be uniform; that is the frequency of occurrence of each number should be the same.

b) Independence: the probability of observing a value in a particular interval is independent of the previous values drawn. This means that no value in the sequence can be inferred from the others.

Mainly there are many methods used to generate random numbers. In the most methods the modulus (m) should be as large as possible, because a small set of numbers make the outcome easier to predict. We will mention some of them as follows. [2, 4, 5, 6]:

2.1 Linear Congruential Method (LCM)
This method is used to generate a sequence of integers X_1, X_2,…X_n between 0 and m-1 followed by a recursive relationship as in Eq. (1):

$$X_i = (aX_{i-1} + c) \bmod m \qquad (1)$$

Such that the parameters as:

X_0 =seed (or starting value)

m =modulus(or divisor)

a =multiplier

c =increment

Where m > 0 and a < m, c < m, X_0< m

The selection of the values for a, c, m, and X_0 drastically affects the statistical properties and the cycle length. The random integers X_i are being generated in the interval [0, m-1]. The random numbers generated are then calculated as in Eq. (2):

$$R_i = \frac{X_i}{m}, \; for \qquad i = 1,2,....,k \qquad (2)$$

Therefore, the criteria of good RNG is to satisfy maximum density and it leaves no large gaps on [0, m-1]. Such that maximum density means that all numbers between [0, m-1]

should appear in predictable manner. According to [2] to achieve the maximum density and avoid cycling we have to choose the suitable values for the factors a, c, m and $X_{0.}$

There are main deficiencies in the LCM, if an opponent knows that LCM is being used and if the parameters are known, and once a single number is discovered, then all subsequent numbers can be easy to know. Even if the opponent knows only that LCM is being used, then the knowledge of a small part of sequence is sufficient to determine the parameters of the algorithm. Suppose that the opponent is able to determine values for $X_{0,}$ $X_{1,}$ X_2, and X_3. Then

$$X_1 = (aX_0 + c) \bmod m \quad (3)$$

$$X_2 = (aX_1 + c) \bmod m \quad (4)$$

$$X_3 = (aX_2 + c) \bmod m \quad (5)$$

Those equations (3), (4) and (5) can be solved to get the parameters a, c, and m.

2.2 Combined Linear Congruential Generators

This method obtains the longer period generator because it combines two or more multiplicative congruential generators [2, 7].

- Let $X_{i,1}$, $X_{i,2}$, …, $X_{i,k}$ be the i^{th} output from k different multiplicative congruential generators.
- The j^{th} generator $X_{0,j}$:
  - $X_{i+1,j} = (a_i X_i + c_i) \bmod m_i$ (6)
- Such that m_j is a prime modulus, a_j is multiplier, and m_j -1 is a period.
- Produces integers $X_{i,j}$ approximate ~ Uniform in [0, m_j – 1].
- $W_{i,j} = X_{i,j-1}$ approximate ~ Uniform on integers in [0, m_j - 2].

$$X_i = \left(\sum_{j=1}^{k} (-1)^{j-1} X_{i,j} \right) \bmod m_i - 1 \quad (7)$$

Then the maximum possible period shown in Eq (8) as follows:

$$P = \frac{(m_1 - 1)(m_2 - 1)\ldots(m_k - 1)}{2^{k-1}} \quad (8).$$

2.3 RNG using Cipher Text

This method uses any cipher text to generate random numbers by converting the cipher text to binary digits and selects suitable numbers of binary digits to be converted to decimal digits. But the main defects are that it needs more calculation [3].

3. PROPOSED METHOD

The main idea of the suggested method is to use combined linear congruential generators based on generating the parameters a_i, c_i, and m_i dynamically depending on the user seed(s) named as ($X_{0,j}$). It also uses the concatenation of several generators numbers dynamically (CONCATENAT (X_i) such that i=1,..k). Hence, started point i changes periodically.

The algorithm becomes as follows:

1- Select seeds ($X_{0,j}$) such that j=1 to k.

2- Compute

a_j = geta ($X_{0,j}$ + n)

where n is the number required & a_j is prime number ≈=1+4k, and k is an integer.

3- Compute the

c_j = prime (a_j)

4- Compute

m_j = getm (a_j+ n) , $m_j = 2^b$

such that b is integer number.

5- For j= 1 to k,

$$X_j = (a_j X_{i,j} + c_j) \bmod m_j$$

$$X_{i+1,j} = X_j$$

where i = 0 to n, and k is an integer number depends on security required.

6- Compute

X_{i+1} = CONCATENATE (X_j) for k's values, such that j position change dynamically.

7- Set i= i+1

8- If i<=n go to step 5.

9- Stop.

4. ANALYSIS AND COMPLEXITY STUDY

In linear congruential method, the Eq. (1) is used to generate the RNGs. Thus the selection of the values for the parameters a, c, and m drastically affect the statistical properties and the cycle length.

Meanwhile, in combined linear congruential generators, using the Eq. (6) with the same parameters a, c, and m which are used as a partial part and the selection of the values for the parameters drastically affects the statistical properties and the cycle length.

Whereas, the concatenation or addition of two or more random numbers is a method to produce a new random number with the same statistical properties and to increase the cycle length. The time complexity to concatenate two or more random numbers always less than the addition of the same numbers. Therefore, it can be said that the time complexity is less than the current combined method.

5. EXPERIMENTAL RESULTS

This section explains some examples for experimental results for different essential seeds using our generated simulation software according to the proposed algorithm mentioned in section III. As a result, three different experiments have been done with different user seeds, it gives different outputs. The result shows that, there are no reputation value and the cycle period is better than the other algorithms. Table (1) shows the result for 50 values using four essential seeds (12, 129, 512, 985). Table (2) shows the result for 50 values using four essential seeds (55, 278, 985, 1299); whereas, Table (3) shows the result for 50 values using four essential seeds (980, 1200, 2440, 3600).

Table (1): Data Generated for 4 values of Essential Seeds (12, 129, 512, 985) & 50 Output

Generated value	Generated value	Generated value
9723257448	3223719403657	2910812053664
845710962745	5714411692272	4035713082345
53841741368	2836918082457	171846172048
4834918283081	7718816373216	10041600121
1053524493872	5621314521769	1014208934016
449715683929	651366653648	644611802505
12526815572768	116256161593	254812651376
7269601193	2124415813568	6033718401305
113882011152	806110921929	453489652320
4911443065	7325615692976	884371228617
696817573120	76655762777	334012732752
961734681353	9342819091872	20505136441
1214641361480	10429386041	11740414212672
923051872153	815041833256	112285356777
135082931424	3648916881913	411606412080
12014910043561	372285732224	1083316321625
145618811856	37011777342051	

Table (2): Data Generated for 4 values of Essential Seeds (55, 278, 985, 1299) & 50 Output.

Generated value	Generated value	Generated value
1061954483924	1873703657741	15095136642246
3175027452551	9877922723960	991382345585
2069593682922	1353262457987	21872320481180
1393430811709	1343273216782	11130121703
2102738723776	5158176917	1907940164018
21588639291699	7473936483556	9121025053701
1182152768342	19184615933975	668111376520
34901193473	17473535682042	3993413053435
18624311522860	4389819292621	1026152320926
15126306579	1785929762384	2114106171953
15887931201090	11947027772099	4225927523188
2515013532565	8650318722534	959424411303
348434801176	16384241361	14251126721162
1995181531787	154787256444	2037387773533
7090314241070	17522219133551	146912080992
11576235613889	12665522242258	235416252499
1080318562820	33136665364821	

Table (3): Data Generated for 4 values of Essential Seeds (980, 1200, 2440, 3600) & 50 Output.

Generated value	Generated value	Generated value
379762568013337	4060169224487676	781155722497981
3428362438641672	45392175532097	11321620544692
765102142814373	331654427285504	3765211367696873
3836321242722540	3309240547777805	644220072244856
1445215344493329	398032454402980	221366137373941
360424489685872	168530895931705	1564319640488156
3789274111613021	400433363446504	389287333934049
6841588967828	1725169352417093	1844281653524704
1653355346574217	1340164417768140	2221256577851821
1220293678006616	1381308143854257	353213168967044
118133096015637	2132164876882000	3669350525776217
3164265256482492	27011877734205	484100050165192
325277774253953	23636047840628	268531762017765
372304079765952	3637294525457705	2940212474727596
113367726011981	179616241842232	1317196143213137
308426045444964	214190956575285	66028966216176
155718734561489	75136665364812	

6. CONCLUSION AND FUTURE WORK

This paper suggests Dynamic Random Number Generator Based on User Seed(s). The suggested method uses the idea of combined linear congruential generators based on generating the parameters a_i, c_i, and m_i dynamically depending on the user seed(s). It is difficult to detect three factors used to generate random numbers if the seeds change periodically, also it is difficult to detect the next generated random number based on the first one. Using user seeds to generate factors which are used to generate random numbers will enhance the security and accuracy. The result shows that there are no reputation value and the cycle period is better than the other algorithms. In the future the method needs software application can be compared with other methods, also we intend to use this method in reducing the risk in key management research.

7. REFERENCES

[1] William Stallings 2009. Cryptography and Network Security: Principles and Practice. 3rd Ed. India Reprint. Agrawal-M IETE-Technical-Review.

[2] Jerry Banks, etc. 2001. Discrete-Event System Simulation. 3rd Ed. Pearson Education. Singapore.

[3] Bruce Schneier 2010. Applied Cryptography. 3rd Ed. John Wiley & Sons. (ASIA) Pvt. Ltd. 2 Clementi Loop # 02-01. Singapore 129809.

[4] Borosh. S. and Niederreiter H. 1983. "Optimal Multipliers For Pseudo-Random Number Generation By The Linear Congruential Method", BIT 23,65-74.

[5] Figiel, K.D. and Sule. D.R. 1985. "New Lagged Product Test for Random Number Generators", Comput. Ind. Eng. Vol. 9, 287-296.

[6] S. Japertas and et al. 2007. "Unpredictable Cryptographic Pseudo-Random Number Generator based on Non-linear Dynamic Chaotic System", Electronics and Electrical Engineering ISSN 1392 – 1215, pp 29 -32.

[7] P. L'Ecuyer 1988. "Efficient and Portable Combined Random Number Generators", Communications of the ACM 31 June 1988, Volume 31 No. 6, USA.

[8] Douglas W. Mitchell 1993. "A Nonlinear Random Number Generator with Known, Long Cycle Length", Cryptologia, Volume 17 Issue 1, pp 55- 62, USA.

8. AUTHOR's PROFILE

Dr. Saleh Noman Abdullah Alasaly, born in 1969, Gabel Habashee, Taiz, Republic of Yemen. Ph.D. in Infromation Securty, SRTMU, India, 2005. Assistant Prof., Khawlan College, Sana'a University, Yemen.

E-mail: saleh.alasali97@yahoo.com

Dr. Sharaf A. Alhomdy, born in 20/01/1971, Alsena, Taiz, Republic of Yemen. Ph.D. in Computer Scince, Pune University, India, 2009. Asst. Prof. & Vice-Dean, Faculty of Computer and Information Technology, Sana'a University, Yemen.

E-mail: sharafalhomdy@gmail.com.