Computer Engineering and Intelligent Systems
ISSN 2222-1719 (Paper) ISSN 2222-2863 (Online)
Vol.6, No.9, 2015


# Testing for Randomness in Pseudo Random Number Generators Algorithms in a Cryptographic Application

Christopher A. Abilimi[1] Michael Asante[2] Edward Opoku Mensah[3] Francis Ohene Boateng[3]
1.Department of Computer Science, Christian Service University, Box 3110, Kumasi, Ghana
2.Department of Computer Science, Kwame Nkrumah University of Science and Technology, Ghana
3.Department Information Technology Education, University of Education Winneba, Kumasi Campus, Ghana

**Abstract**
The most effective cryptographic algorithm has more randomness in the numbers a generator generates, and the more secured it is to be used for protecting confidential data. Sometimes developers find it difficult to determine which Random Number–Generators (RNGs) can provide a much secured Cryptographic System for secured enterprise application implementations. This research aims to find an effective Pseudo Random Number Generator algorithm among Fibonacci Random Numbers Generator Algorithms, Gaussian Random Numbers Generator Algorithm, Specific Range Random Numbers Generator Algorithms, and Secure Random numbers Generators, which are the most common Pseudo Random Numbers Generators Algorithms, that can be used to improve the security of Cryptographic software systems. The researchers employed Chi-Square test on the first 100 random numbers between 0 to 1000 generated using the above generators and it concluded that, Fibonacci Random Numbers Generator Algorithms can provide a more secured cryptographic application.
**Keywords** Pseudo Random Number Generators, Randomness, Cryptography, Software

## 1. Introduction

Every pseudo random number generator describes an individual series of computer – based generations of random numbers (Henderson, 2009). This is because if the fact that accepted mathematics of modular integer enables a known or countable numbers of random numbers, ultimately there must be a repetition of the series used (Terzidis, 2006). Dichtl and Golić (2007) indicated that a pseudo random numbers are different from numbers correlations, any time there are the repetitions of numbers then they are known as a correlated numbers but not pseudo random numbers, for pseudo random numbers must not have repetitions. It mustn't allow future numbers to be predicted based on previous numbers generated (Aihe, 2008). According to Aihe, when this happens, there is a deficient for accurateness of Monte Carlo Simulation in a pseudo random numbers generations. This therefore calls for procedures or methodologies to be used for the choice of pseudo random number with less or no repetition; this can only be possible if we have knowledge on the random number generator algorithm that produces those random numbers.

According to Menezes, Van Oorschot and Vanstone (2010), for cryptographic purposes, one normally assumes some upper limit on the work an adversary can do (usually this limit is astronomically sized). If one has a pseudo-random number generator whose output is "sufficiently difficult" to predict, one can generate true random numbers to use as the initial value (i.e., the seed), and then use the pseudo-random number generator to produce numbers for use in cryptographic applications (Pachghare, 2008). Such random number generators are called cryptographically secure pseudo-random number generators, and several have been implemented (for example, the /dev/urandom device available on most Unixes, the Yarrow and Fortuna designs, server, and AT&T Bell Laboratories "truerand") (Burrows, Abadi, Manasse, Wobber & Simon, 2006). As with all cryptographic software, Foster, Kesselman and Tuecke(2001), stated that, there are subtle issues beyond those discussed here, so care is certainly indicated in actual practice. In any case, it is sometimes impossible to avoid the need for true (i.e., hardware) random number generators, they indicated.

Kane (2009), explained that ,since a requirement in cryptography is high entropy (i.e., unpredictability to an attacker), any published random sequence is a poor choice, as such sequences such as the digits in an irrational number such as the $\varphi$ or even in transcendental numbers such as $\pi$, or e. All are available to an enterprising attacker. Tomcsanyi and Lueg (2010), put it in another way, that, in cryptography, random bit streams need to be not only random, but also secret and hence unpredictable. Tomcsanyi et al (2010) explained that, public or third-party sources of random values, or random values computed from publicly observable phenomena (weather, sports game results, and stock prices), are almost never cryptographically acceptable, though often tempting and too often used by the unwary. They permit easier attacks than attacking the cryptography.

According to Schneier (2007), since most cryptographic applications require a few thousand bits at most, slow random number generators serve well—if they are actually random. This use of random generators is important; many software developers believe every computer should have a way to generate true random numbers (Knuth, 2014).

Drimer (2008), Schneier (2007) and Standaert, Peeters, Rouvroy & Quisquater (2006) , explained that

 

the greatest vulnerability of any cryptographic algorithms is the existence of correlations in the random numbers used in the process, since that meant that it is possible for the opponent or the attacker to predict the trends in the data or information encrypted and that might like increase vulnerabilities rate of the systems. Statistically whether or not a peculiar pseudo random number generator creates a series good enough statistically random may be cracked in some susceptible systems (Panjwani & Cutrell, 2010). It is a neat initiative to carry out a Monte Carlo Simulation experiment with a series of pseudo random number generators to see if it is a sensitivity biased to a system in particular (Wall & Jenkins, 2012).

The researchers seek to establish among four pseudo random number generators namely; Fibonacci Random Numbers Generator Algorithms, Gaussian Random Numbers Generator Algorithm, Specific Range Random Numbers Generator Algorithms, and Secure Random numbers Generators, the most secured one that can be used in cryptographic software application in order to reduce vulnerability to the applications by attacker.

## 2. Methodology

### 2.1 Tests for Randomness of Random Number Generators (RNGs).

The analysis of the trend of the distributions of numbers in a random numbers generator is called the Quantitative tests of Randomness and is done by Chi-Square Test (Inglot & Janic-Wróblewska, 2003). According to Spall (2005), the application areas for random numbers such as computer simulation and in stochastic modeling the experimental outputs of random number generators are confirmed to establish that experimental tests of randomness were done by means of data that are truly random.

#### 2.2.1 Chi-square Test

According to Opoku-Mensah, Abilimi and Boateng (2013), one of the most popular tests for random numbers is the Ch-Square Test, which has conformed to the errors of pseudo random series generators sensitivity. Opoku-Mensah et al (2013), explained that, the distribution of the Chi-Square is a non-negative value and a proportion which shows how often real random series should surpass a computed number or value determined as a result for collections of zeroes and ones in a certain data file. The procedures used are as follows:

i. Java Programming codes is written for each of Fibonacci Random Numbers Generators, Gaussian Random Numbers Generators, Secure Random Numbers Generators and Specific Range Random Numbers Generators
ii. This test produces sequences of 100 random integers between 0 and 1000 using the Java codes for each of Fibonacci Random Numbers Generators, Gaussian Random Numbers Generators, Secure Random Numbers Generators and Specific Range Random Numbers Generators.
iii. The generated random numbers in (ii) above is coded in Statistical Package for Social Science (SPSS) software to test of randomness of the numbers in the generators. The procedures include:
    a. Go to and click on the **Analyse Menu** on the SPSS bar after the data has been coded.
    b. Choose **Nonparametric Test** on the submenu.
    c. Go to and then Click on **Chi-Square Test**.
    d. Select all the four random numbers generators and move them to the **variable list.**
    e. The results generated for the Chi-Square Test are interpreted.
iv. For uniform distributions of random numbers from the generators, then the expectations is that there should be 1 appearance or occurrences for the number 1 or 2 or 3 and so on and so forth to the end of the dimension of the numbers, 100.
v. Then how often each of the numbers appears will be expected to have a frequency of 1.0 or 1.1 averagely for all rhe numbers from 1 to 100 ranges.
vi. The observed frequency is the real frequency for every one of the numbers generated by the generator with the ranges of numbers specified.
vii. The Chi-Square value or statistics is calculated from the difference between the observed frequency of the test and that of the expected frequency of every one of the numbers generated as follows:

$$\chi^2 = \sum_{i=1}^{R} \frac{(O_i - E_i)^2}{E_i}$$

- $R$ is the distinct individual random numbers likely ($R = 100$), the number of the observed frequencies of happenings or occurrences for the random numbers ($i$) is $Oi$ and where the frequency of expectation or the expected frequency is $Ei$ for the random integer or number $i$.
- Since the expectation is that the integer distribution must uniformly spread, then the occurrences for every random number or integer must have equal expected frequencies.
- $Ei$ can then be computed with $N$ as the total number of observations, using the equation in the

 

next section.

- The Chi-Square values for the four random numbers generators obtained are shown below:

## 3. Results

### 3.1. The Test for randomness in Random numbers

The analysis of the factors responsible for randomness in a random number generator showed that factors tend to give many considerations to Fibonacci Random Numbers Generator than the other Generators compared with. This is because the Chi-Square analysis for the repetition of numbers in the generators showed that numbers were less or not repeated at all in the Fibonacci Random Numbers Generator (Chi-Square Value = 0.000),and this accounts for the reason for lowest Chi-square value, than the other generators under studied. The lowest the chi-square value the lesser the repetitions of numbers in the random number generator and vice versa. That is because repetition of numbers in the random number generators increases with the decrease in Chi-Square Value and vice versa. In that order, the second less repeated random number generator is Specific Range Random Numbers Generator (Chi-Square value = 4.500), followed by Secure Random Numbers Generator (Chi-Square value = 7.380) and then Gaussian Random Numbers Generator (Chi-Square value = 14.400) as shown in ***Table 1.***

This therefore makes Gaussian Random Numbers Generator (Chi-Square value = 14.400) as the pseudo random number generator with the most repeated numbers and that is why it has got the highest Chi-Square value of 14.400, as shown in ***Table 1***. This also means that numbers in the GRNG were more likely to be repeated than all the others random numbers generators. ***Table 1*** represents the descriptive statistics of the analysis (Standard Deviation, Mean, etc.). Standard deviation of numbers increases with decreases normality and vice versa. This means the higher the standard deviation value the higher the deviation from the normal and vice versa. Therefore Fibonacci Random Number Generator is more deviated from the normal distribution than all the other generators while Gaussian Random Number Generator produces numbers that obeys the normal distribution than all the generators compared with as shown in ***Table 1.***

**Table 1**
**The Test for randomness of PRNGs (Pseudo Nandom Number Generators)**

| Statistics | | Specific Range Random Numbers Generator | Secure Random Numbers Generator | Gaussian Random Numbers Generator | Fibonacci Random Numbers Generator |
|---|---|---|---|---|---|
| Chi-Square | | 4.500 | 7.380 | 14.400 | 0.000 |
| Df | | 94 | 90 | 87 | 99 |
| Asymp.Sig. | | 1.000 | 1.000 | 1.000 | 1.000 |
| Monte carlo | Sig. | 1.000 | 1.000 | 1.000 | 1.000 |
| 90%Confidence Interval | Lower Bound | 0.977 | 0.977 | 0.977 | 0.977 |
| | Upper Bound | 1.000 | 1.000 | 1.000 | 1.000 |

## 4. Discussions

A good PRNG will produce a sequence of numbers that cannot be easily guessed or determined by an adversary. The general assumption is that the opponent knows the algorithm being used. This is usually referred to as Kerckhoff's principle (Yehuda, 2006). This accessing is evidenced in Fibonacci Random Numbers Generator than the other Generators compared with. This is because the Chi-Square analysis for the repetition of numbers in the generators showed that numbers were less repeated or no repetition in the Fibonacci Random Numbers Generator than the other generators under studied. Also Dichtl and Golić (2007), indicated that a pseudo random numbers are different from numbers correlations, any time there are the repetition of numbers then there known as a correlated numbers but not pseudo random numbers, for pseudo random numbers must not have repetitions. The accessing by Dichtl and Golić (2007), is more evident in Fibonacci Random Numbers Generator (Chi-Square Value = 0.000), followed by Specific Range Random Number Generator (Chi-Square value = 4.500), followed by Secure Random Numbers Generator (Chi-Square value = 7.380) and the least of them to be considered as pseudo random number generator is Gaussian Random Numbers Generator (Chi-Square value = 14.400), because the highest repetition of the numbers in the generator and hence will make a cryptographic algorithm less secured. Again, based on the accessing of Kane (2009),that a requirement in cryptography is high entropy (i.e., unpredictability to an attacker), any published random sequence is a poor choice, as are such sequences as the digits in an irrational number such as the φ or even in transcendental numbers such as π, or e and this requires that every seed(random number) to be used for the cryptographic systems have no pattern of predictability (like more repletion of the numbers in the random number generators). This means that a highly

 

secured cryptographic application has less or no repeated numbers in the generator and this is seen in Fibonacci Random Numbers Generator.

On the other hand, Gaussian Random Numbers Generator (Chi-Square value = 14.400), is show to have the greatest vulnerability features as it is shown to have the highest repeated numbers among the factors analysed as seen in Drimer (2008), Schneier (2007) and Standaert, Peeters, Rouvroy & Quisquater (2006) , who explained in their various papers indicating that the greatest vulnerability of any cryptographic algorithms is the existence of correlations in the random numbers used in the process, since that meant that it is possible for the opponent or the attacker to predict the trends in the data or information encrypted and that might likely increase vulnerabilities rate of the systems. There it can be said that Gaussian Random Numbers Generator (Chi-Square value = 14.400) is more vulnerable followed by Secure Random Numbers Generator (Chi-Square value = 7.380), followed by Specific Range Random Number Generator (Chi-Square value = 4.500) and finally the least vulnerable in cryptographic security terms is Fibonacci Random Numbers Generator (Chi-Square value = 0.000). This is because according to Schneier (2007), vulnerability of random number generators increases with decreasing Chi-Square values and vice versa, and this makes the generator with the highest Chi-Square value the most vulnerable and vice versa.

## 5. Conclusions

The test for repetition of PRNGs (Pseudo Random Number Generators Algorithms) also revealed that numbers in the Fibonacci Random Numbers Generator produce lesser or no repeated numbers in the generator (**Section 3.1**) than the other Generators compared with. However, the worst generator in terms of repetition of numbers in the generator is Gaussian Random Numbers Generator (GRNG) (Chi-Square value = 14.400) (**section 3. 1 & Table 1**). This means that numbers in the GRNG were more likely to depend on each other's than all the others random numbers generators. It can therefore be concluded that, Fibonacci Random Numbers Generator will provide more secure cryptographic system, followed by Specific Range Random Number Generator, followed by Secure Random Numbers Generator and then Gaussian Random Numbers Generator (**Section 3.1 & Table 1**).

## References

Aihe, D. O. (2008). A reinforcement learning technique for enhancing human behavior models in a context-based architecture. ProQuest.

Andrew, R., Juan, S., James, N., Miles, S., Elaine, B., & Stefan, L. (2010). A statistical test suite for random and pseudorandom number generators for cryptographic application, Revision1a,USA.Special Publication 800-22 (http://csrc.nist.gov/publications/nistpubs /800-22-rev1a/SP800-22rev1a.pdf. Accessed 2012, September 1).

Burrows, M., Abadi, M., Manasse, M. S., Wobber, E. P., & Simon, D. R. (2006). U.S. Patent No. 7,149,801. Washington, DC: U.S. Patent and Trademark Office.

Dichtl, M., & Golić, J. D. (2007). High-speed true random number generation with logic gates only (pp. 45-62). Springer Berlin Heidelberg.

Drimer, S. (2008). Volatile FPGA design security–a survey. IEEE Computer Society Annual Volume, 292-297.

Foster, I., Kesselman, C., & Tuecke, S. (2001). The anatomy of the grid: Enabling scalable virtual organizations. International journal of high performance computing applications, 15(3), 200-222.

Henderson, H. (2009). Encyclopedia of computer science and technology. Infobase Publishing.

Inglot, T., & Janic-Wróblewska, A. (2003). Data driven chi-square test for uniformity with unequal cells. Journal of Statistical Computation and Simulation,73(8), 545-561.

James, A. H., Abdissa N., Michael D. , Edwardes, B., & Janet E. F., (2003). Statistical Analysis of Correlated Data Using Generalized Estimating Equations: An Orientation, American Journal of Epidemiology, USA,Volume 157, Issue 4 , pp. 364-375.

Kane, A. M. (2009). On the use of Continued Fractions for Stream Ciphers. InSecurity and Management (pp. 583-589).

Knuth, D. E. (2014). Art of Computer Programming, Volume 2: Seminumerical Algorithms, The. Addison-Wesley Professional.

Menezes, A. J., Van Oorschot, P. C., & Vanstone, S. A. (2010). Handbook of applied cryptography. CRC press.

Opoku-Mensah, E., Abilimi, C. A., & Boateng, F. O. (2013). Comparative Analysis of Efficiency of Fibonacci Random Number Generator Algorithm and Gaussian Random Number Generator Algorithm in a Cryptographic System.Computer Engineering and Intelligent Systems, 4(10), 50-57.

Pachghare, V. K. (2008). Cryptography and information security. PHI Learning Pvt. Ltd..

Panjwani, S., & Cutrell, E. (2010, July). Usably secure, low-cost authentication for mobile banking. In Proceedings of the Sixth Symposium on Usable Privacy and Security (p. 4). ACM.

Terzidis, K. (2006). Algorithmic architecture. Routledge.

Tomcsanyi, D. P., & Lueg, L. (2010). CCMP known-plain-text attack.

 

Schneier, B. (2007). Applied cryptography: protocols, algorithms, and source code in C. john wiley & sons.

Spall, J. C. (2005). Introduction to stochastic search and optimization: estimation, simulation, and control (Vol. 65). John Wiley & Sons.

Standaert, O. X., Peeters, E., Rouvroy, G., & Quisquater, J. J. (2006). An overview of power analysis attacks against field programmable gate arrays.Proceedings of the IEEE, 94(2), 383-394.

Wall, J. V., & Jenkins, C. R. (2012). Practical statistics for astronomers. Cambridge University Press.

Yehuda, L. (2006). Introduction to cryptography, 89-656. (http://u.cs.biu.ac.il/~lindell/89-656/Intro-to-crypto-89-656.pdf. Accessed 2012, August 1).

The IISTE is a pioneer in the Open-Access hosting service and academic event management. The aim of the firm is Accelerating Global Knowledge Sharing.

More information about the firm can be found on the homepage:
http://www.iiste.org

## CALL FOR JOURNAL PAPERS

There are more than 30 peer-reviewed academic journals hosted under the hosting platform.

**Prospective authors of journals can find the submission instruction on the following page:** http://www.iiste.org/journals/ All the journals articles are available online to the readers all over the world without financial, legal, or technical barriers other than those inseparable from gaining access to the internet itself. Paper version of the journals is also available upon request of readers and authors.

## MORE RESOURCES

Book publication information: http://www.iiste.org/book/

Academic conference: http://www.iiste.org/conference/upcoming-conferences-call-for-paper/

## IISTE Knowledge Sharing Partners

EBSCO, Index Copernicus, Ulrich's Periodicals Directory, JournalTOCS, PKP Open Archives Harvester, Bielefeld Academic Search Engine, Elektronische Zeitschriftenbibliothek EZB, Open J-Gate, OCLC WorldCat, Universe Digtial Library , NewJour, Google Scholar