PSEUDO RANDOM NUMBER GENERATION USING EYE BRIGHTNESS RESPONSE

T. Sivakumar[1] and T. Anusha[2]
[1]Department of Computer Science and Engineering, Dr. Mahalingam College of Engineering and Technology, India
[2]Department of Computer Science and Engineering, PSG College of Technology, India

Abstract:

Random numbers play an important and primary role in the use of Cryptography techniques in real time applications. The cryptographic techniques can be easily compromised if the key can be easily guessed. Therefore it is important that the keys are in random and unpredictable in nature. The operating system uses the random numbers to mask passwords and to offer salt and session identifiers. This paper introduces a new software based pseudo random number generation method based on the eye brightness response formula. This function provides a significant change in sensation for minimum required change in signal intensity. The randomness tests are performed to confirm the randomness of the generated random numbers.

Keywords:

Cryptography, Pseudo Random Number, Eye Brightness Response, Randomness Test

1. INTRODUCTION

The strength of any cryptographic techniques is strongly depends on the randomness of the chosen key. The output of cryptographic techniques is a sequence of random bits which carrying the secret information without revealing no clues about the precious information [9]. Most encryption algorithms require a source of random data, even some symmetric ciphers (where the secret is shared), either to generate new private/public key pairs, for session keys, for padding, or for other reasons. Most computers do not have a hardware based random number generator (RNG), so programmers have had to resort to software based techniques, to generate random numbers[1]. The inadequacies of hardware for hardware based random number generation have created a need for inexpensive and widely available method of generating random numbers with software. Using this method all personal computers can generate cryptographically secure random bits without any specialized hardware.

Depending on the nature of the randomness source, generators are classified in three categories as follows.

- *True Random Number Generators (TRNG)*: For TRNG, the source is a natural physical phenomenon and the properties of independence and unpredictability of the generated values are guaranteed by physical laws. While TRNGs offer the highest level of entropy [3, 4] (meaning measure of uncertainty: number of symbols that have to be known in order to remove uncertainty associated with a random variable, and also information content: number of symbols necessary to encode all possible values of a variable) they do not necessarily present uniform distribution and most of them need to be filtered (post processed) in order to reduce possible bias - tendency towards a particular value, and correlation, and make the output more similar to perfectly random sequence [2,5-7]. In [12], an image encryption method using Knight's Travel path and True Random Number is developed. The true random numbers are generated from the amplitude values of a chosen noise audio file.
- *Unpredictable Random Number Generators (URNG)*: URNGs are based on the unpredictability inherent to human computer interaction and on the indeterminism introduced by the complexity of the underlying phenomenon (e.g. Linux s/dev/random, etc.) [8]. URNGs use easily available devices, like computer components, as entropy sources and provide a high level of randomness [2].
- *Pseudo Random Number Generators (PRNG)*: For PRNG, the source of randomness is a random seed value which is expanded by means of a deterministic recursive formula. As a result, the unpredictability level resumes to the randomness of the seed value and the output is completely determined by the starting state of the generator. The practical features of PRNGs are high generation speed, good statistical results and no need for additional hardware devices. This is a widely used random number generator in cryptographic systems. However, the reduced level of unpredictability is not sufficient for security applications because these can be easily compromised by using a low quality randomness source [2].

In [11], the authors introduced an image encryption method based on pixels position permutation and random key stream. To change the pixel values of the image random key stream is utilized [13]. The necessary amount of random bit stream is constructed by adopting the random bit pattern procedure used in the MD5 hash function.

Human eye is capable of responding to an enormous range of light intensity. Inevitably, eye response to the signal intensity, which determines its apparent intensity, is not linear. That is, it is not determined by the nominal change in physical stimulus, rather by its change relative to its initial level [10]. This paper utilized the eye brightness response formula to generate Pseudo random numbers.

In this paper, a simple and new method for generating Pseudo random number by using the eye brightness response formula is proposed.

The rest of the paper is organized as follows: section 2 provides the proposed random number generator. Section 3 presents the experimental results and analysis. The paper is concluded in section 4.

2. PROPOSED RANDOM NUMBER GENERATOR

The proposed random number generation uses the brightness response of the eye to produce random number (based on

Brightness perception in complex fields, Bartleson and Breneman) which is given by Eq.(1) and Eq.(2).

$$\log B = 2.037 + 0.1401\log I - a\exp(b\log I) \tag{1}$$

$$F = \log(\log B) \tag{2}$$

where, *F* is proposed the pseudo random function, and logB is Eye response to photographic image under varying image and surround luminance, *I* is the intensity, *a* and *b* are constants varying with the luminance level. The Fig.1 shows the eye response as a function of luminance [10].

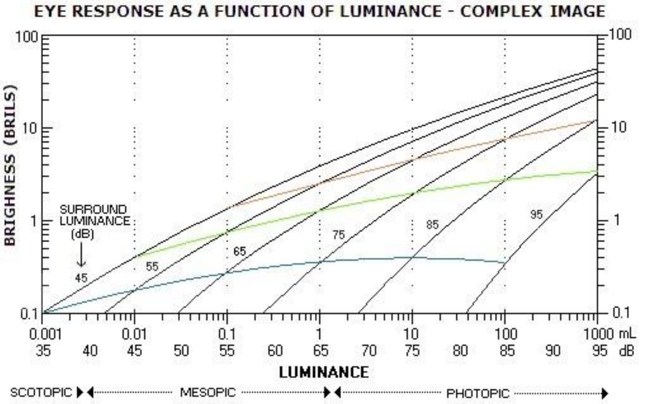

Fig.1. Eye Response as a function of luminance

2.1 ALGORITHM

The following are the sequence of steps used to generate a random numbers.

Input: Intensity (*I*) and constants varying with the luminance level values *a* and *b* and the range of required random numbers

Output: Random numbers

Step 1: Take the intensity 0.1 or any other value, the value *a* and *b* as 0.6 and 4.5 respectively (seed value is stored or can be subject to change).

Step 2: The output for the input intensity calculated. For Example, input *i* = 0.001, the following is the sample output of the pseudo random function.

0.3306091223544712

0.34015730427728313

0.3487952142313704

0.35667596584697386

Step 3: Two positions after the decimal point is taken and converted into binary string of length 64 bits. For instance,

1000010111110001011001100011011010101010010111111001110001000000 0

1000011001011001000100010010010001100110101000100100011011100 10

1000010111110111011110010100011011000011100001010001110100000 00

10000110011010000100010101001100010100100101001001111100010101010000110000000100010101010011011001010101001111101010101 10011000

Step 4: The 64-bit binary array is separated into two halves of size 32 bits and XOR-ed with each other to ensure randomness and further the 32-bit array is converted into 16-bit array which in turn is converted into 8-bit array in the same manner.

Then the obtained random numbers are,

00101110 (48)

01000110 (70)

00001011 (11)

01100111 (103)

00001000 (8)

Step 5: The binary array (8 bit binary number array) is converted as integer and is stored as random numbers.

Step 6: Repeat steps 2 to 5 until sufficient amount of random numbers are generated.

Step 7: Stop the process

3. EXPERIMENTAL RESULTS AND RANDOMNESS TESTING

The proposed method is experimented using Java language and the system configuration is Processor Intel i5-5200U CPU, Clock speed 2.2GHz, RAM 4GB and the operating system is Ubuntu 4.1. The random numbers generation by using the proposed method is shown in the form of images in Fig.2 and Fig.3 respectively.

Fig.2. Random Numbers (128×128)

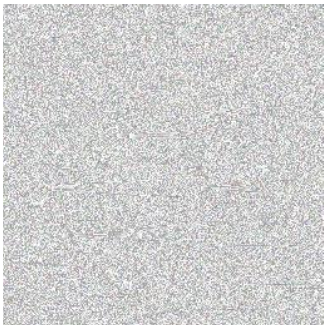

Fig.3. Random Numbers (256×256)

The statistical analysis of the random sequences is very important. The statistical tests assess the outcome of a randomness generator [9].

3.1 FREQUENCY TEST (MONO BIT TEST)

This test determines the proportion of the number of ones and zeros in a bit sequence. In the bit sequence 1 is taken as +1 and 0 is taken as -1 and the sobs is calculated by Eq.(3) and Eq.(4) [14].

$$S_{obs} = S_n/root(n) \tag{3}$$

where, $S_n = b_1 + b_2 + b_3 +...+ b_n$

The p-value is then given by,

$$P\text{-value} = S_{obs}/1.414 \tag{4}$$

If the P-value is greater than or equal to 0.01, the sequence is said to be random, otherwise it is rejected as not random.

3.2 RUNS TEST

A run of length k consists of k identical bits bound before and after with a bit of opposite value. For example, the bit sequence 1100101011011100 consists of the runs 11, 00, 1, 0, 1, 0, 11, 0, 111, 00. The purpose of the runs tests is to determine whether the number of runs of ones and zeros of various lengths is as expected for a random sequence [14]. The Eq.(5) and Eq.(6) give the formula to compute and verify the runs test [14].

$$V_{obs}=\sum R(k) + 1 \tag{5}$$

where, $k = 0$ to n, $R(k) = 0$ if $\varepsilon_k = \varepsilon_k + 1$ and $R(k) = 1$,

The P-value is given by

$$P\text{-value} = \text{erfc}(nV^2_{obs} \times 3.3598 \times \text{sqrt}(2n) \times 6.7196) \tag{6}$$

If the P-value is greater than or equal to 0.01, the sequence is said to be random, otherwise it is rejected as not random. The randomness test results on the random numbers generated by using the proposed method is given in Table.1.

Table.1. Statistical Test Result

Random Numbers	Runs Test	Frequency Test	Result
512×512	0.4759	0.8026	Pass
256×256	0.2045	0.6171	Pass
128×128	0.3327	0.3173	Pass
64×64	0.0455	0.0455	Pass

4. CONCLUSION

Cryptographic techniques can be easily broken if the key is vulnerable to predict or easily guessed. Therefore it is important that the keys are random and so cryptography desire for random numbers. In this paper, a new software based pseudo random number generation method based on the eye brightness response is developed. The result obtained with the proposed method is tested with two randomness test cases. From, the result it is found that the proposed method can be utilized to generate random numbers for practical applications.

REFERENCES

[1] Kurt Seifried, “Kurt Seifried Information Security”, Available at: https://seifried.org/security/.

[2] Kinga Marton, Alin Suciu and Iosif Ignat, “Randomness in Digital Cryptography A Survey”, Romanian Journal of Information Science and Technology, Vol. 13, No. 3, pp. 219-240, 2010.

[3] C.E. Shannon, “A Mathematical Theory of Communication”, *Bell System Technical Journal*, Vol. 27, pp. 623-656, 1948.

[4] J. Kelsey, “Entropy and Entropy Sources in x9.82”, Available at: https://pdfs.semanticscholar.org/presentation/e762/977134551a72edf19d8be13fef075de97b38.pdf.

[5] J. Von Neumann, “Various Techniques used in Connection with Random Digits”, *National Bureau of Standards, Applied Mathematics Series*, Vol. 12, pp. 36-38, 1951.

[6] M. Blum, “Independent Unbiased Coin Flips from a Correlated biased Source- a Finite State Markov Chain”, *Combinatorica*, Vol. 6, No. 2, pp. 97-108, 1986.

[7] Z. Gutterman B. Pinkas and T. Reinman, “Analysis of the Linux Random Number Generator”, *Proceedings of IEEE Symposium on Security and Privacy*, pp. 21-24, 2006.

[8] A. Seznec and N. Sendrier N., “Hardware Volatile Entropy Gathering and Expansion: Generating Unpredictable Random Number at User Level”, Research Report, Department of Computer Science, INRIA, 2002

[9] Kinga Marton, Alin Suciu, Christian Sacarea and Octavian Cret, “Generation and Testing of Random Numbers for Cryptographic Applications”, *Proceedings of the Romanian Academy, Series A*, Vol. 13, No. 3, pp. 368-377, 2012.

[10] Eye Intensity Response, Available at: http://www.telescopeoptics.net/eye_intensity_response.htm, Accessed on 2016.

[11] T. Sivakumar and R. Venkatesan, “Image Encryption Method based on Pixel Shuffling and Random Key Stream”, *International Journal of Computer and Information Technology*, Vol. 3, No. 6, pp. 1468-1476, 2014.

[12] T. Sivakumar and R. Venkatesan, “A New Image Encryption Method Based on Knight’s Travel Path and True Random Number”, *Journal of Information Science and Engineering*, Vol. 32, No. 1, pp. 133-152, 2016.

[13] William Stallings, “*Cryptography and Network Security-Principles and Practice*”, Pearson Education, 2015.

[14] Andrew Rukhin et al., “A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications”, *National Institute of Standards and Technology*, pp. 1-131, 2001.