Avrupa Bilim ve Teknoloji Dergisi
Özel Sayı 24, S. 157-160, Nisan 2021

Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
Special Issue 24, pp. 157-160, April 2021

Research Article

Security Analysis of Java SecureRandom Library

Kenan İnce

İnönü University, Faculty of Engineering, Departmant of Computer Engineering, Malatya, Turkey, (ORCID: 0000-0003-4709-9557), kenanince@gmail.com

(2nd International Conference on Access to Recent Advances in Engineering and Digitalization (ARACONF)-10–12 March 2021)

(**DOI:** 10.31590/ejosat.900956)

ATIF/REFERENCE: İnce, K. (2021). Security Analysis of Java SecureRandom Library. *Avrupa Bilim ve Teknoloji Dergisi*, (24), 157-160.

Abstract

Java is one of the most used programming languages. Developers use java language in all of their projects, embedded systems or as a background service provider for different frontend applications. In today's world where security gains importance day by day, the reliability of security libraries of programming languages is also gaining importance.
One of the common research area of computer security is random number generation. Most of the cryptographic applications require random numbers. Many different approaches exist for secure random number generation. However, most of them are academic for today. For this reason, it is more common to use libraries that are available in programming languages. In this study, a comprehensive analysis of Java SecureRandom library by means of security is presented. NIST 800-22 test suit is used for randomness tests.

Keywords: SecureRandom, Java Security, NIST 800-22, Randomness Tests.

Java SecureRandom Kütüphanesinin Güvenlik Analizi

Öz

Java en çok kullanılan programlama dillerinden biridir. Geliştiriciler java dilini projelerinin tamamında, gömülü sistemlerde veya farklı arayüz projeleri için servis katmanında kullanmaktadırlar. Güvenliğin her geçen gün önem kazandığı günümüzde, programlama dillerinin güvenliğinin bütünlüğü önem kazanmaktadır.
Rasgele sayı üretimi, bilgisayar güvenliğinin en önemli araştırma alanlarından biridir. Bir çok kriptografik uygulama rasgele sayılara ihtiyaç duyar. Güvenli rasgele sayı üretimi konusunda bir çok çalışma yapılmıştır. Fakat bunların bir çoğu günümüz için akademik seviyede kalmaktadır. Bu sebeple programlama dillerinin içerisinde hazır bulunan kütüphanelerin kullanımı daha yaygındır. Bu çalışmada, Java SecureRandom kütüphanesinin güvenlik anlamında detaylı bir analizi sunulmuştur. Rassallık testleri için NIST 800-22 Rev1a test ortamı kullanılmıştır.

Anahtar Kelimeler: SecureRandom, Java Güvenliği, NIST 800-22, Rassallık Testleri.

1. Introduction

According to Tiobe Index of January 2021, Java is the second highest rated programming language. Also, same study shows that java always in the top three programimng language after the year 2001 (TIOBE 2021). It is expected that such a widely used rogramming language will be able to meet the needs of time. Security is an important requirement as well as reliability and platform independence. For his reason, various studies carried out on security analysis of java and java related applications.

Feng et al (2011), in his article, he presents a new approach by java byte-level flow analysis. By doing this, he claims this method can be used as an assistant to reveal byte code vurnelabities.

Martínez et al (2017) investigate Java EE Access control mechanism on web security due to misconfiguration. They propose a reverse engineering model for analyzing anomalies and they share this application on Github.

Paul & Evans (2006) compare two major platform by means of security which are Java and .NET. Basically, they show how .NET prevent vulnerabilities that are exists in Java. They also mentioned that .NET benefited from past experiences of Java. By shielding some details from developers, they prevent mis configuration on policies.

Another study on Java security is done by Herzog & Shahmehri in 2005. They explore the slowness of Java security manager. Because, especially thinking the time they investigate the performance of Java security manager, time and space complexity of programming language mechanism is important by any means. They present 20 execution times in a table format. They find out that the when the resource Access done under security manager, the execution time increase approximately 100% by comparing to resource access without security manager.

Another Java related security study is evaluation of Java Scure Socket Extention (JSSE) usage. They point out due to the complexity of the application programming interface (API) of transport level security (TLS) leads developers to mis use of security mechanism and this result in vulnerabilities in their application. They study with 11 developers to identify usability issues of JSSE and they show that the abbsraction layer is the main reason of misusage (Wijayarathna & Arachchilage, 2019).

The main motivation of this study withstands to following facts:

1. Random number generation is a crucial task in cryptography.
2. Although there exist many secure random number generators (SRNGs), they recuire extra investigation and implementation.
3. Java is the second highest rated programming language today.

Considering the above acceptances, it is evaluated that security anaylsis of SecureRandom library (which extends default Java.util.Random library) of Java programming language is very important and required. Furthermore, according to out investigation, no study exists on SecureRandom library.

2. Material and Method

2.1. Random Number Generation

Random number generators (RNG) categorized in to two main classes, deterministic and non-deterministic. Determinism means that is it possible or not to reproduce same sequence of random numbers which is generated previously. As a result, if an RNG does not depends on physical events the randomness of the generator must be tested.

Nondeterministic methods also can be divided into two main categories which are physical and computational RNGs (Saldamli & Koc 2009). Java SecureRandom library is in the computational non-deterministic RNG category. In theory SecureRandom library is cryptographically strong RNG (Oracle JavaSE-8, 2021).

In literature there exists many RNG studies focus on chaos theory (Katz et al, 2008; Stojanovski and Kocarev, 2001), FPGA (Thomas and Lok, 2013; Akçay et al, 2017), electron transistor (Uchida et al, 2007) etc. All these studies focus on more secure and reliable random number generation. However, while all programming languages have random libraries, many developers rely their applications' security, if needed, on these standart libraries. Because, it is hard to implement thecniques on many academic studies for developers. Besides, random libraries are ready and easy to use.

2.2. Randomness Test Suites

There exist some statistical test suites for testing a sequence is random or not. Most commonly used test suites are NIST 800-22 (Lawrence et al, 2010), Diehard and Dieharder (Brown, 2021), ENT Utility (Walker, 2008) and TestU01 (L'ecuyer & Simard, 2007). The most prefered test suite in literature is NIST. NIST test suite consist of 15 different statistical test which is shown in Table 1. Table 1 also shows relative minimum bit length requirement to be able to produce meaningfull results according to suite documentation.

Table 1: NIST Tests and Relative Minimum Bit Length Recommendation

#	Test Name	Min Len
1	Frequency (Monobit)	100
2	Block Frequency	100
3	Runs	100
4	Longest Run of Ones	128
5	Binary Matrix Rank	38912
6	Discrete Fourier	1000
7	Non-Overlapping Template Matching	1000000
8	Overlapping Template Matching	106
9	Universal Statistical	387840
10	Linear complexity	1000000
11	Serial	32
12	Approximate Entropy	127
13	Cumulative Sums	100
14	Random Excursions	1000000
15	Random Excursions Variant	1000000

2.1. Random Number Generation Algorithm

It is known that standard random number generation libraries in widely used programming languages are use system time. As a result, if an attacker finds out the generation time of random number, he/she may generate same random number or sequence. To overcome this problem, standard RNG libraries in programming languages uses seed. A seed is the initial starting point of generation. If a complex seed is given, more secure number generation will be acquired.

In addition to standard random number libraries, SecureRandom library gives developers to select some generation predefined algorithms. Table 2 shows algorithm that are present in Java Cryptography Architecture Standards.

Algorithm Name	**Platform**
NativePRNG	Linux, Mac
NativePRNGBlocking	Linux, Mac
NativePRNGNonBlocking	Linux, Mac
PKCS11	-
SHA1PRNG	Linux, Mac, Windows
Windows-PRNG	Windows

In this study, it is preferred to use SHA1PRNG due to platform independence. PKCS11 library is dependent on installing the related libraries separately. Other algorithms run under stated platforms without any other requirement except JDK 8 or higher.

2.3. Application

The application developed in Mac Big Sur operation system. Also, relative percentages tested on Windows 10 machine. The presented results are mean of both platforms. Simple activity diagram of the application presented in Figure 1.

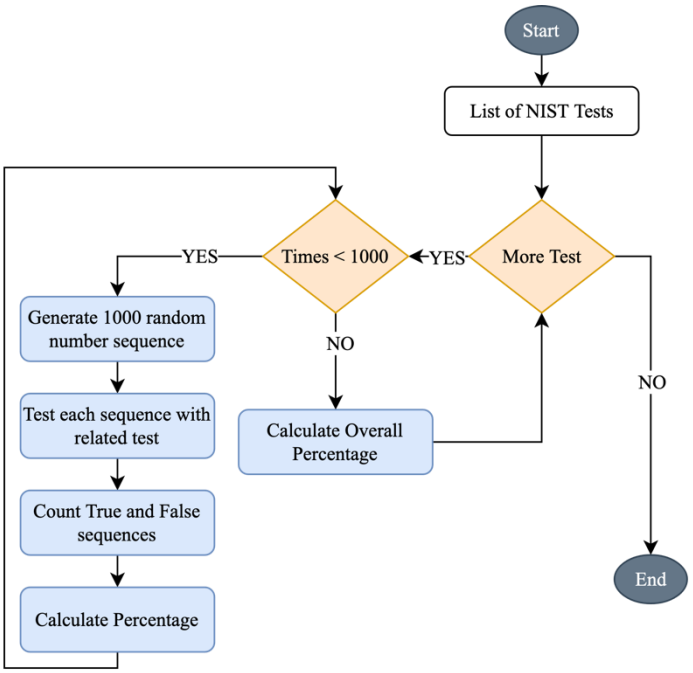

Figure 1: Block diagram of the test algorithm

In order for the application results to be valid, a random number sequence consisting of 1000 samples was tested 1000 times and the average rate was obtained. Obtained results presented in Table 3.

Table 3: Application result percentages

#	Test Name	True Percentage
1	Frequency (Monobit)	98.9
2	Block Frequency	99.51
3	Runs	98.81
4	Longest Run of Ones	99.23
5	Binary Matrix Rank	99.19
6	Discrete Fourier	98.55
7	Non-Overlapping Template Matching	24.4
8	Overlapping Template Matching	84.98
9	Universal Statistical	98.82
10	Linear complexity	99.02
11	Serial	97.77
12	Approximate Entropy	98.9
13	Cumulative Sums	99.03
14	Random Excursions	57.64
15	Random Excursions Variant	57.77

3. Results and Discussion

According to the results, the library generates cryptographically secure sequences. However, while the sequence length increase, the reliability percentage drops.

Both “Random excursions” and “Random excursions variant” tests rely on cumulative sum random walk. They test some arbitrary fixed length sequence cumulative sums repeats or not. Also “Non-Overlapping template matching test” rely on aperiodic patterns. It analyzes the existing of these patters. As a result, it can be said that SecureRandom library shows weak security requirements with pattern tests.

5. Acknowledge

This work was supported by the projects of the İnönü University Scientific Research Projects Depart-ment (SRPD) numbered FBG-2018-1107 and FBG-2020-2143. The author would like to thank İnönüUniversity SRPD for their valuable feedback.

References

TIOBE 2021, TIOBE Index for January 2021, https://www.tiobe.com/tiobe-index/, Last accessed: Jan 17 2021.

Z. L. Feng, T. Hong, H. M. Huan, K. X. Hui and J. Qi (2011), "Checking Java Bugs by Data Propagation Analysis," *2011 First International Conference on Instrumentation, Measurement, Computer, Communication and Control*, Beijing, 2011, pp. 861-864, doi: 10.1109/IMCCC.2011.217.

Salvador Martínez, Valerio Cosentino, Jordi Cabot (2017), Model-based analysis of Java EE web security misconfigurations, Computer Languages, Systems & Structures, Volume 49, 2017, Pages 36-61, ISSN 1477-8424, https://doi.org/10.1016/j.cl.2017.02.001.

Nathanael Paul, David Evans (2006), Comparing Java and .NET security: Lessons learned and missed, Computers & Security, Volume 25, Issue 5, 2006, Pages 338-350, ISSN 0167-4048, https://doi.org/10.1016/j.cose.2006.02.003.

Almut Herzog, Nahid Shahmehri (2005), Performance of the Java security manager, Computers & Security, Volume 24, Issue 3, 2005, Pages 192-207, ISSN 0167-4048, https://doi.org/10.1016/j.cose.2004.08.006.

Chamila Wijayarathna, Nalin Asanka Gamagedara Arachchilage (2019), Why Johnny can't develop a secure application? A usability analysis of Java Secure Socket Extension API, Computers & Security, Volume 80, 2019, Pages 54-73, ISSN 0167-4048, https://doi.org/10.1016/j.cose.2018.09.007.

Saldamli G. and Koc C. K. (2009), Random Number Generators for Cryptographic Applications, in Cryptographic Engineering, Springer.

Oracle JavaSE-8 (2021), Class SecureRandom, https://docs.oracle.com/javase/8/docs/api/java/security/SecureRandom.html, Last Accessed: Jan 17 2021.

Lawrence E. Bassham, Andrew L. Rukhin, Juan Soto, James R. Nechvatal, Miles E. Smid, Elaine B. Barker, Stefan D. Leigh, Mark Levenson, Mark Vangel, David L. Banks, Nathanael Alan Heckert, James F. Dray, and San Vo. (2010). *SP 800-22 Rev. 1a. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*. Technical Report. National Institute of Standards & Technology, Gaithersburg, MD, USA.

Robert G. Brown (2021), Robert G. Brown's General Tools Page, https://webhome.phy.duke.edu/~rgb/General/dieharder.php, Last Accessed: Jan 17 2021.

John Walker (2008), A Pseudorandom Number Sequence Test Program, https://www.fourmilab.ch/random/, Last Accessed: Jan 17 2021.

L'ecuyer, P. and Simard, R. (2007). TestU01: A C library for empirical testing of random number gen-erators. ACM Trans. Math. Softw. 33, 4, Article 22 (August 2007), 40 pages. DOI=10.1145/1268776.1268777 http://doi.acm.org/10.1145/1268776.1268777

O. Katz, D. A. Ramon and I. A. Wagner, (2008), "A Robust Random Number Generator Based on a Differential Current-Mode Chaos," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 16, no. 12, pp. 1677-1686, Dec. 2008, doi: 10.1109/TVLSI.2008.2001731.

T. Stojanovski and L. Kocarev, "Chaos-based random number generators-part I: analysis [cryptography]," in *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 48, no. 3, pp. 281-288, March 2001, doi: 10.1109/81.915385.

D. B. Thomas and W. Luk, "The LUT-SR Family of Uniform Random Number Generators for FPGA Architectures," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 4, pp. 761-770, April 2013, doi: 10.1109/TVLSI.2012.2194171.

L. Akçay, E. Çil, A. Vardar, İ. Yaman, R. Yeniçeri and M. E. Yalçın, "Implementation of a chaotic time-delay RNG based secure communication system on FPGA," *2017 10th International Conference on Electrical and Electronics Engineering (ELECO)*, Bursa, 2017, pp. 1277-1280.

Ken Uchida, Tetsufumi Tanamoto, Shinobu Fujita, Single-electron random-number generator (RNG) for highly secure ubiquitous computing applications, Solid-State Electronics, Volume 51, Issues 11–12, 2007, Pages 1552-1557, ISSN 0038-1101, https://doi.org/10.1016/j.sse.2007.09.015.