MAKÜFEBED

ISSN Online: 1309-2243
http://dergipark.org.tr/makufebed
https://doi.org/10.29048/makufebed.1273073

Mehmet Akif Ersoy Üniversitesi Fen Bilimleri Enstitüsü Dergisi 14(1): 185-195 (2023)
The Journal of Graduate School of Natural and Applied Sciences of Mehmet Akif Ersoy University 14(1): 185-195 (2023)

Araştırma Makalesi / *Research Paper*

Can Tesla Sphere be used for Random Number Generation?

Oğuzhan ARSLAN[1*], İsmail KIRBAŞ[2]

[1]Burdur Mehmet Akif Ersoy University, Graduate School of Natural and Applied Sciences, Burdur, Türkiye
[2]Burdur Mehmet Akif Ersoy University, Faculty of Engineering and Architecture, Burdur, Türkiye

Geliş Tarihi (Received): 29.03.2023, Kabul Tarihi (Accepted): 27.05.2023

✉ *Sorumlu Yazar (Corresponding author*): oguzhan_arslan@outlook.com*
☎ *+90 248 2132751* 📠 *+90 248 2132704*

ABSTRACT

The use of random numbers to represent uncertainty and unpredictability is essential in many industries. This is crucial in disciplines such as computer science, cryptography and statistics, where the use of randomness helps to guarantee the security and reliability of systems and procedures. In computer science, random number generation is used to generate passwords, keys and other security tokens, as well as to add randomness to algorithms and simulations. According to recent research, the hardware random number generators used in billions of IoT devices do not generate enough entropy. This paper describes how raw data collected by IoT system sensors can be used to generate random numbers for cryptography systems and also examines the consequences of these random numbers. Colour, light and camera are used as sensors. Monobit and poker test results are analysed to measure the quality of randomness. Sequences were obtained with the method that gave quality values as a result of the analysis and these sequences were entered into the NIST and FIPS 140-1 randomness test packages. When the results of these two tests were analysed, it was observed that the sequences passed all tests successfully.

Anahtar Kelimeler: Cryptography, internet of things, light sensor, random number generators, webcam sensor

Rastgele Sayı Üretimi için Tesla Küresi Kullanılabilir mi?

ÖZ

Belirsizliği ve öngörülemezliği temsil etmek için rasgele sayıların kullanılması birçok endüstride esastır. Bu, rastgelelik kullanımının sistemlerin ve prosedürlerin güvenliğini ve güvenilirliğini garanti etmeye yardımcı olduğu bilgisayar bilimi, kriptografi ve istatistik gibi disiplinlerde çok önemlidir. Bilgisayar biliminde, rastgele sayı üretimi parolalar, anahtarlar ve diğer güvenlik belirteçleri oluşturmak ve ayrıca algoritmalara ve simülasyonlara rastgelelik eklemek için kullanılır. Son araştırmalara göre milyarlarca Nesnelerin İnterneti cihazında kullanılan donanımsal rastgele sayı üreteçleri yeterli entropi üretmiyor. Bu makale, IoT sistem sensörleri tarafından toplanan ham verilerin kriptografi sistemleri için rastgele sayılar oluşturmak üzere nasıl kullanılabileceğini açıklamakta ve ayrıca bu rastgele sayıların sonuçlarını incelemektedir. Sensör olarak renk, ışık ve kamera kullanılmıştır. Rastgelelik kalitesini ölçmek maksadıyla monobit ve poker test sonuçları analiz edilmiştir. Analiz sonucu kaliteli değerler veren yöntem ile diziler elde edilip NIST ve FIPS 140-1 rastgelelik test paketlerine bu diziler sokulmuştur. Bu iki testin sonuçları irdelendiğinde ise bütün testlerden başarıyla geçtiği gözlemlenmiştir.

Keywords: Kriptografi, nesnelerin interneti, ışık sensörü, rastgele sayı üreteçleri, webcam sensörü

Oğuzhan ARSLAN, https://orcid.org/0000-0002-4399-8910
İsmail KIRBAŞ, https://orcid.org/0000-0002-1206-8294

INTRODUCTION

The term "Internet of Things," or IoT in short, is derived from the phrases "object" and "internet" and is one of the subjects that has been the subject of numerous studies in recent years. There are billions of users worldwide who use the global system of connected computer networks known as the Internet. By facilitating the transmission of information between individuals, this global system has become a crucial component of our daily lives. The Internet of Things is the most used terminology, although it has terminological counterparts such as Internet of Everything (IoE), Web of Things (WoT), Web of Everything (WoE), and Machine to Machine (M2M) (Gözüaçık, 2015). The idea of the Internet of Things (IoT) has come to mean a network of interconnected devices that can interact with one another by connecting to the internet without the help of a third party. IoT devices can access cloud-based resources to collect data and extract the collected data, make authorisation arrangements, and make decisions by analysing the collected data with the help of algorithms (Conti et al., 2018).

Random number generation is critical in many fields because it is used to simulate uncertainty and unpredictability. This is important in fields such as computer science, cryptography, and statistics, where randomness is used to ensure the security and reliability of systems and processes. In computer science, random number generation is used to create randomness in algorithms and simulations, as well as to generate passwords, keys, and other types of security tokens. In cryptography, random numbers are used to generate secure keys for encrypting and decrypting data, as well as to create random challenges in authentication protocols. In statistics, random number generation is used to sample data and to perform statistical tests. Overall, random number generation is a critical component of many systems and processes that rely on uncertainty and unpredictability to function correctly and securely. The raw data needed by random number generators in cryptographic systems can be obtained using the information gathered. This article will describe how these raw data can be utilised to create fixed-length keys that can be incorporated into algorithms that will protect the security of vital communication systems.

For the generation of random numbers, random sequences were obtained with the data taken from nature with the help of some sensors. The values obtained by using sensors such as temperature, pressure, light, gas, humidity and pH (Rehman et al., 2020), (Sunny et al., 2020), (Üçgün et al., 2020) can be used as seed values for random number generators. Ansari et al. created a real random number generator using ldr and sound sensors connected to an Arduino microcomputer (Ansari et al., 2022). Tuncer and Genç proposed a random number generator based on the GPS sensor in mobile phones and human movement (Genç and Arslan Tuncer, 2019). Yaşar et al. (2021) used the random function of the C programming language and the sha256 summarisation algorithm to generate random integers. In his research, Chen obtained random numbers with video and audio noise with a camera (Chen, 2013). Etem and Kaya created the random number generator for their research without the need for any hardware, using the LCG (Linear Congruential Generator) algorithm with Trivium as the postprocessor (Etem and Kaya, 2020). By raising the electrical voltage, Nikola Tesla, who was born in 1856 in the Serbian village of Similjan, made it possible to transmit electrical power wirelessly with a low output current density (Sezgin, 2021). Raw data from the Tesla sphere in the physical environment will be collected as a noise source using IoT devices or sensors. The obtained values will then be converted into digital data using a Raspberry Pi device, and if necessary, they will be subjected to post processing algorithms to produce fixed length number sequences.

Secure communication system architecture, encryption methods, and random number generators (RNG) are the foundations of cryptography. Private keys and secret keys are generated using the distinctive random numbers produced by the RNG. RNGs are divided into two groups: real and pseudo. Because they are simpler to operate, pseudo random number generators are selected more often. Because the quick generation of random numbers without the need for any hardware is a significant cost benefit. On the other hand, true random number generators, which are crucial for secure communication systems, incorporate non-deterministic numbers as a noise source. Expensive gear is needed to capture the genuine unpredictability of the environment. The random numbers that will be produced must exhibit high statistics, be unpredictable, have a consistent structure, and use hardware rather than pseudo RSUs in terms of confidentiality. Some mathematical conditions (randomness tests) must be satisfied if the produced numbers are used in sensitive contexts, such as cryptography systems. The general encryption structure of a text message between a sender and a recipient is depicted in Figure 1.

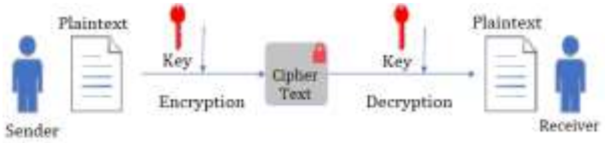

Figure 1. General structure of encryption

As shown in Figure 1, the encryption process is initiated for the text to be encrypted with the help of the key. In this paper, we will talk about how to make the secure key that encryption techniques need.

Previous studies used either internal or external methods to obtain the seed values for random number generators. Post-processing algorithms use input variables like the system clock, mouse movements, CPU data, image or sound data, random functions in programming languages, etc. as seeds. After some time, the numbers generated in this manner begin to repeat and exhibit predictable behaviour. In this project, raw data were obtained with the movements and intensity of the radiations in the Tesla sphere. It has been observed that the raw data obtained do not repeat and continue to be produced unpredictably. The fact that the system is autonomous, that is, it can work without external human intervention, is another point that sets the system apart from other studies. When the literature was analysed, it was seen that a project similar to the Tesla sphere as a noise source was not carried out. If the NIST test suite results are analysed, it will be understood that completely random sequences are generated. In this project, the values obtained from the Tesla sphere (Sunny, 2020), which was used by Nicola Tesla in 1891 to transmit electricity wirelessly as a noise source, will be converted into digital data. After being subjected to a post-processing algorithm with a minicomputer (raspberry pi), fixed-length, unique, unpredictable, and chaos-based number sequences will be obtained. The chaotic environment needed for random number generation is created by gathering information from electrical radiations that are randomly distributed across the sphere, from its centre outward. The input source's chaotic character will guarantee the development of irregular, independent sequences. It is predicted that it will close the knowledge gap in this area and help with the issue of acquiring the seed value of the random number generators used today.

MATERIAL AND METHOD

To make the secret information between two or more communicating points unintelligible, cryptology, which is a cipher science, encrypts it using a variety of techniques. The secret information is subsequently decrypted on the receiving side. It is a collection of approaches and applications built on high-level mathematical ideas (Yalman and Ertürk, 2016).

The phrases "secret" and "writing," which refer to secret writing, are the roots of the word "cryptography." A sender runs the risk of having his communication intercepted and changed when using open networks to convey it to a recipient. Plain text is the message that is in danger here. Encryption is the process of masking a message's content. The plaintext is transformed through this procedure into an encrypted format that is incomprehensible to others. This data could be either encrypted data for storage or a message that is encrypted for transmission. Decryption is the procedure through which the receiving party transforms the cipher text back into plain text (Atar et al., 2017). The process of looking for the ciphertext's solution is known as cryptanalysis. Finding potential flaws in cryptographic systems and information breaches is the fundamental goal of cryptanalysis, which is based on exceedingly complex mathematical calculations.

The encryption algorithms used worldwide and their types, random number generators and their types, the sensors used to obtain raw data in the project and the randomness tests of the sequences obtained after post-processing are explained in the following subheadings.

Encryption Algorithms

Encryption is the process of masking a message's content. The plaintext is transformed through this procedure into an encrypted format that is incomprehensible to others. This data could be either encrypted data for storage or a message that is encrypted for transmission. Decryption is the procedure through which the receiving party transforms the cipher text back into plain text (Atar et al., 2017). In symmetric encryption methods, the encryption algorithm subjects the encrypted message to several procedures before it can be transferred. Symmetric key encryption techniques use the same keys for encryption and decryption (Yılmaz and Ballı, 2016). AES (Advanced Encryption Standard), DES (Data Encryption Standard), and 3DES are popular symmetric encryption techniques today (Triple DES). Public key encryption is another name for asymmetric encryption methods. For encryption and decryption, there is a public key and a private key. Asymmetric encryption techniques boost the computer's processing capability by using very big prime numbers

(Atar et al., 2017). Asymmetric cryptography uses public key infrastructure because long keys and lengthy computations are required (Maqsood et al., 2017). The most popular symmetric encryption technique in use right now is called the Advanced Encryption Standard. AES is a highly effective symmetric key block cipher in terms of both security and performance. The key sizes that can be used for encryption and decryption are 128, 192, and 256 bits (Abood, 2018). Some of the main characteristics of an encryption algorithm are the following: Confidentiality, integrity, irrefutability, accessibility and identity control (Coşkun and Ülker, 2013).

Random Number Generators

Wherever unpredictability is required, such as in computer games, games of chance, and encryption, random number generators can be utilized. Figure 2 illustrates the division of random number generation into "real" and "pseudo" categories. Pseudo RNGs use algorithms to generate their output, therefore after a while the output data starts to repeat itself on a regular basis. The output data is anticipated to be non-periodic since the source of randomness in a true RNG is based on a chaotic source of uncertainty (Tavas, 2011).

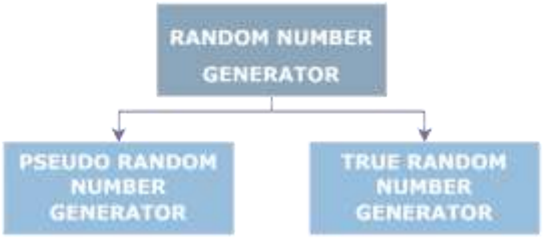

Figure 2. Types of random number generators

Systems that produce random numbers deterministically are known as pseudo RNGs. They have benefits over actual random number generators, including ease of creation and an inexpensive cost. By examining its value at any time when the algorithm is compromised, the subsequent outputs can be anticipated (Demirkol, 2007). In secure communication systems that demand confidentiality, this prediction may result in significant security issues (Huang et al., 2020). True RNGs are systems that employ the chaotic randomness of nature to produce numbers by post-processing with an algorithm. For instance, statistical data gathered by remote monitoring of a plant in an agricultural field or random raw data that cannot be predicted with data obtained from the measurement sensor attached to an animal's foot can be obtained. The numbers exhibiting poor statistical features are post-processed to demonstrate greater statistics after the sampling procedure (Yosunlu and Avaroğlu, 2020).

Testing for randomness ensures that the post-processed datasets from the entropy source are accurate and realistic. Bit sequences obtained using various sensing sources (camera and light sensor) and methodologies (mode method, last bit extraction, and hash algorithms) will be examined in this research paper's monobit and poker test findings. The ratio of ones to zeros in a sequence is compared in the monobit test. If there are more than 9725 ones in a sequence of 20000 bits, the test is successful. If there are fewer than 9725, the test is unsuccessful (Luengo et al., 2022). Post processing algorithms will be employed to refine the raw data and boost unpredictability. One of the most popular post-processing techniques, the XOR algorithm, can be characterized as two-bit inputs producing a one bit output. The hash algorithms utilized in this paper are sha256 and md5.

Experimental Study

Three different sensors were used with Raspberry Pi to obtain data from the physical world. A Tesla sphere, which transmits electricity wirelessly by radiation outside the sphere, was used as a noise source. In the following sub-headings, the system structure designed with colour, camera and light sensors and the raw data obtained are explained.

RGB Colour Sensor

In this section, the raw data from the Tesla sphere utilized as an entropy source that was collected by the TCS34725 colour sensor connected to the Raspberry Pi will be analysed. This sensor additionally measures colour temperature and colour irradiance in addition to colour values. By combining the primary colours of red, green, and blue, colour sensors try to get colour values between 0 and 255. These sensors compare the light from the sensor striking the substance with the light values received by reflecting off the material to arrive at the result. Male-female intermediate cables are used to link the GND, SCL, SDA, and 3V3 pins on the colour sensor to the corresponding pins on the Raspberry Pi device on the breadboard. Figure 3 depicts the overall appearance of the experimental set created with the Raspberry Pi 4, TCS34725 RGB Sensor, Tesla Sphere, Monitor, Keyboard and Mouse.

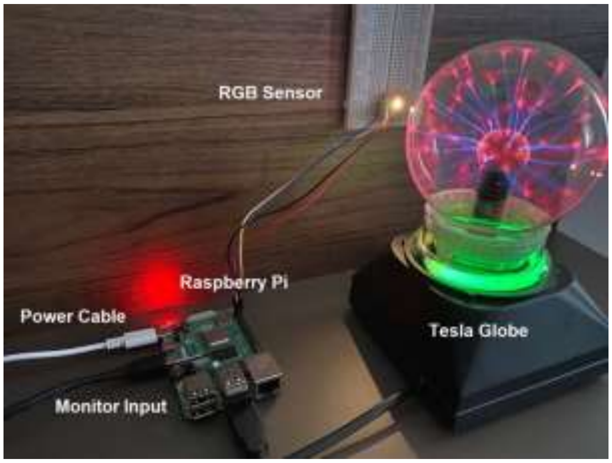

Figure 3. General view of the RGB sensor system

A small sphere in the Tesla sphere's centre randomly emits electrical radiations of various colours in the direction of the glass sphere outside. To extract three red, green, and blue values between 0 and 255 from the colour sensor, a Python coding procedure was used. Raw data were gathered from the Tesla sphere in the real world using a colour sensor as a noise source, and the values obtained were then transformed using a Raspberry Pi device into the numerical colour values in Table 1.

Table 1. Raw data from the RGB sensor

NU.	COLOUR HEXADECİMAL CODE	R-G-B VALUES
1	FFFFFF	(255,255,255)
2	2D2D2D	(45,45,45)
3	FFFFFF	(255,255,255)
4	FFFFFF	(255,255,255)
5	5C5C5C	(92,92,92)
6	5C1010	(92,16,16)
7	5C1010	(92,16,16)
8	5C1010	(92,16,16)

Raw data including RGB values of (45,45,45), colour temperature of 1391.0K, and colour light intensity of 17.566 lux were evaluated. The raw data collected at any given time was noted to be high-quality numbers, but as time went on, the data produced correlated outcomes and the same values overlapped. The (92,16,16) values acquired from the colour sensor were thought to be unsuitable for use as random number generator seeds because they overlap, are not changeable, and have a relationship to one another. As a result, the RGB sensor test results are not listed under the heading "Analysis Results."

WEBCAM Sensor

In this section, the raw data from the Tesla sphere utilized as an entropy source that was collected by the webcam sensor attached to the Raspberry Pi computer will be analysed. The movements of the electrical radiations in the Tesla sphere were detected using the webcam attached to the Raspberry Pi through a USB port, and raw data with x and y coordinate values were collected. Figure 4 depicts the overall perspective of the experimental setup created using a Raspberry Pi 4, Webcam, Tesla Globe, Monitor, Keyboard and mouse.

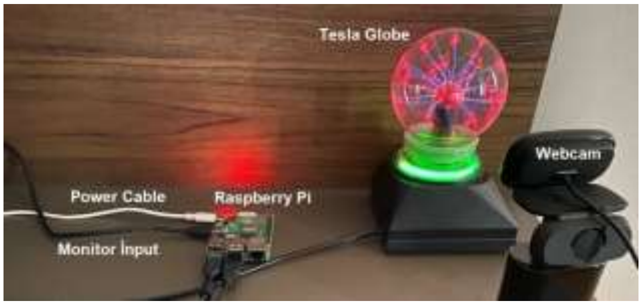

Figure 4. General view of the webcam sensor system

The sphere's radiations are identified using OpenCV, a Python computer language package, and the raw data collected from the moving area's x and y coordinates is then examined. Intel introduced OpenCV, an open-source visual library, in 1999. On the Raspberry Pi computer, the necessary installation processes for the OpenCV library, which is utilized in both academic work and commercial applications, were carried out. Following the library's installation, a program in the Python programming language was created that locates moving areas, grids them in, and outputs the weight point's x and y coordinates, as shown in Table 2.

Table 2. Raw data from the Webcam

NU.	X COORDINATE	Y COORDINATE	ELAPSED TIME (sec)
1	324	305	0.10
2	282	302	0.091
3	197	121	0.078
4	193	82	0.088
5	212	108	0.082
6	260	329	0.078
7	364	133	0.067
8	354	151	0.096

The information gathered in the table above serves as the random number generator's seed values. These variables were used to generate outputs of fixed length using 4 distinct techniques. The first approach entails translating the remainder (Mod 16) into the hexadecimal number system after dividing the x and y coordinate values by 16, respectively. In Table 2, the remainders that were produced after applying the Mod 16 method to the numbers in the second row (282 and 302) correspond to the hexadecimal values "10" and "e," respectively. According to the residual values obtained using this method, the x and y coordinates produced when the webcam sensor detects movement provide an eight-bit output (1010, 1110). Until the specified fixed key length is reached, the motion detection cycle is repeated. The second method involves converting the coordinate values to a binary number system and taking the last bit.

The third-row values (197 and 121) in Table 2 have last bit values of "1" for both coordinate data (after conversion to binary by the last bit method). According to the final bit values discovered using this method, the x and y coordinates formed when the webcam sensor detects movement generate a two-bit long output. Until the specified fixed key length is reached, the motion detection cycle is repeated. The coordinate values are entered into the Md5 and Sha256 hash algorithms to complete the third and fourth methods. After using XOR post processing, the output is obtained by independently summing the x and y coordinate values. These techniques produced 1024-bit outputs, which were then submitted to a monobit randomness test to ensure their randomness. The section under "Analysis Results" will assess the test results.

Light Sensor

This part will analyse the unprocessed data collected by the LDR sensor attached to the Raspberry Pi from the Tesla sphere used as an entropy source. Utilized by the Raspberry Pi device, the LDR is a sensor that gauges light intensity in proportion to the amount of light that strikes it. The amount of light hitting the LDR will determine how much energy the capacitor receives. The time until logic 1 will provide the light intensity since the Raspberry Pi will identify the capacitor charging as logic 1 when it happens. Male-female intermediate wires on the breadboard are used to link the light sensor and capacitor to the Raspberry Pi device's GND, GPIO3, and 3V3 pins. The light values displayed in Table 5 were collected from the Tesla sphere, which serves as the noise source. Figure 5 shows how the experiment set made with the Raspberry Pi 4, LDR Sensor, Tesla Globe, Monitor, Keyboard and Mouse looks as a whole.

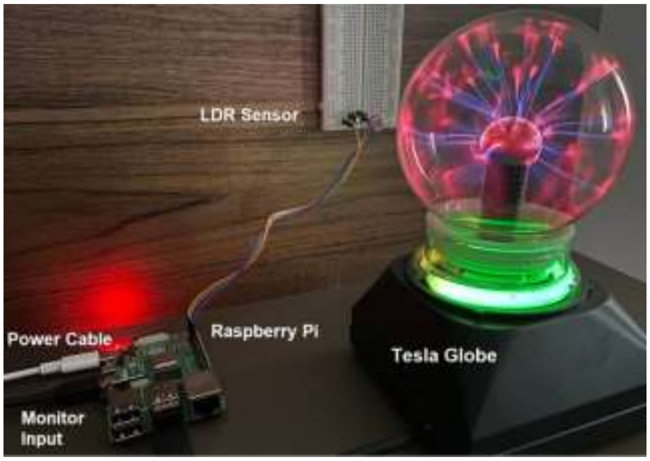

Figure 5. General view of the LDR sensor system

Table 3. Raw data from the LDR sensor

NU.	MEASURED LIGHT INTENSITY	ELAPSED TIME (sec)
1	1090	0.1021
2	1067	0.1019
3	1098	0.1017
4	1094	0.1019
5	638	0.1023
6	654	0.1023
7	1102	0.1020
8	1061	0.1021

The data obtained in Table 3 serves as the random number generator's seed values. Using these data, four distinct strategies, were used to produce outputs of fixed length. The first technique is the remainder (mod 16), which is achieved by dividing the light values by 16. The second approach involves converting the light values to binary and obtaining the final bit. The third and fourth methods are obtained by using the Md5 and Sha256 hash algorithms, respectively. These techniques led to the creation of 1024 bit outputs, similar to those used in the webcam sensor section, which were then subjected to a monobit randomness test in order to verify the unpredictability. The section under "Analysis Results" will assess the test results.

Monobit Test Analysis Results

The monobit test results from three different sources (Pseudo, Webcam, and LDR Sensor) are compared in this study. The frequency test, sometimes referred to as the monobit test, is discovered by counting the occurrences of the integers 0 and 1 in the sequence. 512-bit values should be one- and 512-bit values should be zero in the 1024-bit long outputs acquired from the sensors in the preceding section. The 1024-bit sequence's monobit test result, which was produced using the Random function in the Python programming language to produce pseudorandom numbers, is shown in Table 4 and shown graphically in Figure 6. The distance between one and zero for the 1024-bit sequence is 34.

Table 4. Monobit test of pseudo random number generation

NUMBER	EXPECTED	OBSERVED
Number of 1's	512	495
Number of 0's	512	529

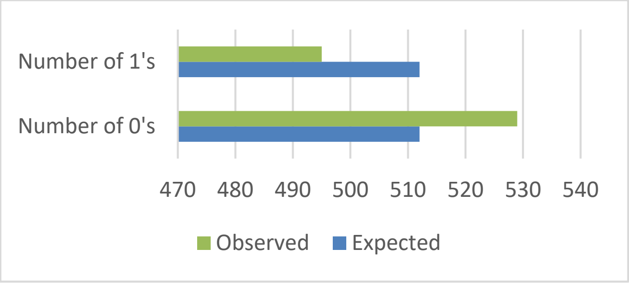

Figure 6. Monobit test chart of pseudo random number generation

The monobit test results of the output sequences produced by four different techniques using a length of 1024 bits are given in Table 5. The graphical representation is shown in Figure 7, and raw data with x and y coordinate values were obtained by detecting the movements of the electrical radiations in the Tesla sphere using the Webcam sensor. The difference between one and zero for the 1024-bit sequence using the Mod 16 approach is 10, the Md5 method is 20, the Sha256 method is 18, and the sequence created by omitting the last bits has a difference of 24. The sequence acquired using the Mod 16 approach was found to be the most similar to the expected values, while the sequence obtained using the last bit method was found to be the furthest from them. In this test, it was found that, in comparison to the pseudorandom number produced by the computer, the numbers generated by all techniques employing the Webcam sensor produced good results.

Table 5. Monobit test with webcam

NUMBER	EXPECTED	OBSERVED			
		Mod 16	Md5	Sha256	End Bits
Number of 1's	512	507	522	521	524
Number of 0's	512	517	502	503	500

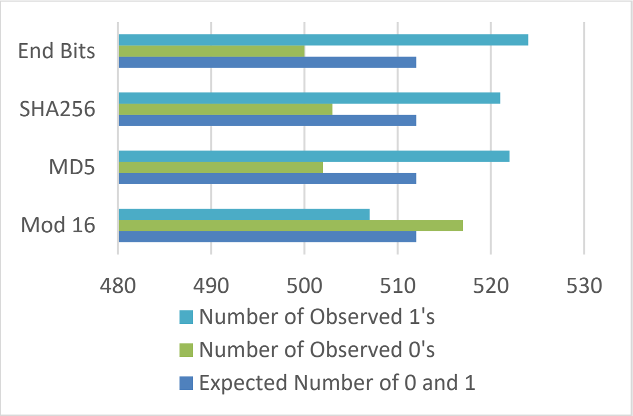

Figure 7. Monobit test chart with webcam

Table 6 lists the findings of the 1024-bit long arrays' monobit tests, which were conducted using 4 different techniques to gauge the radiation strength in the Tesla sphere using an LDR sensor. Figure 8 shows a graphical representation of the data. When using the Mod 16 method, the difference between one and zero for the 1024-bit array is seen to be 22, when using the Md5 method it is 12 and the Sha256 method to be 32, and when using the array created by eliminating the final few bits it is 16. The sequence acquired using the Md5 method was found to be the most similar to the expected values, while the sequence obtained using the Sha256 approach was found to be the furthest from them. In this experiment, it was found that the numbers generated using any of the LDR sensor's methods performed better than the pseudorandom numbers produced by the computer.

Table 6. Monobit test with LDR sensor

NUMBER	EXPECTED	OBSERVED			
		Mod 16	Md5	Sha256	End Bits
Number of 1's	512	501	506	528	520
Number of 0's	512	523	518	496	504

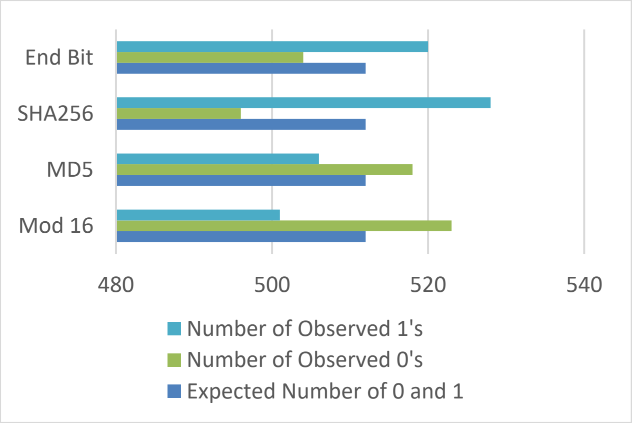

Figure 8. Monobit test chart with LDR sensor

Poker Test Analysis Results

The poker test results from three different sources (Pseudo, Webcam, and LDR Sensor) are compared in this study. In this test, 5000 numbers are produced by dividing a random sequence of 20000 bits into blocks of four bits. Numbers are expressed in the hexadecimal base using these four bits. The computed poker value must fall between 1.03 and 57.4 in order to pass the test.

Table 7 and Figure 9 show the poker test outcome for the 20000-bit sequence produced by the Random function in the Python computer language, which generates pseudo-random numbers. The poker value for a 20000-bit sequence was found to be 13.9904.

Table 7. Poker test of pseudo random number generation

NUMBER	EXPECTED	OBSERVED
Poker Values (X)	1.03 < X < 57.4	13.9904

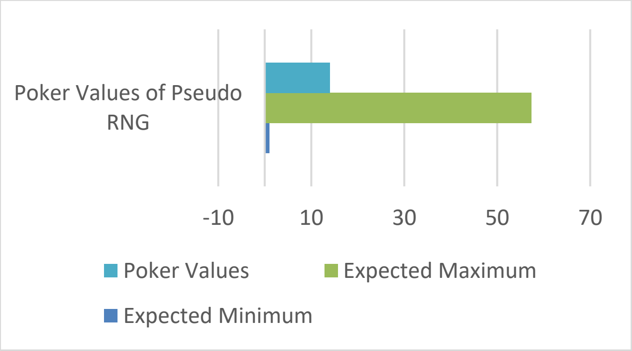

Figure 9. Poker test chart of pseudo random number generation

After using a webcam sensor to monitor the movement of electrical radiations in the Tesla sphere and obtaining raw data with x and y coordinate values, the results of the poker test for 20000-bit output sequences generated by four different methods are shown in Table 8 and the graphical representation is shown in Figure 10. The poker value of the 20000-bit sequence is 21.4720 for the Mod 16 technique, 0.8256 for the Md5 method, -5.1647 for the Sha256 approach and 10.1504 for the sequence produced by skipping the last bits. It is observed that Mod16 and the last bits method passed the test successfully.

Table 8. Poker test with Webcam

	EXPECTED	OBSERVED			
		Mod 16	Md5	Sha256	End Bits
Poker Value (X)	1.03 < X < 57.4	21.472	0.8256	-5.1647	10.1504

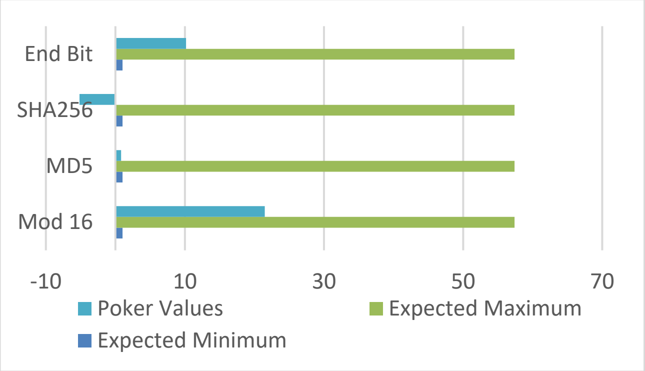

Figure 10. Poker test chart with Webcam

The poker test results for 20000-bit long sequences obtained using 4 different techniques by measuring the radiation intensity in the Tesla sphere with the LDR sensor are given in Table 9 and the graphical representation is given in Figure 11. The poker value of the 20000-bit sequence is 16.3392 for the Mod 16 technique, -54.4511 for the Md5 method, -5.8944 for the Sha256 approach and 20.3136 for the sequence produced by taking the last bits. It is observed that the last bits and mod 16 method passed the test successfully.

Table 9. Monobit test with LDR sensor

	EXPECTED	OBSERVED			
		Mod 16	Md5	Sha256	End Bits
Poker Value (x)	1.03 < X < 57.4	16.3392	-54.4511	-5.8944	20.3136

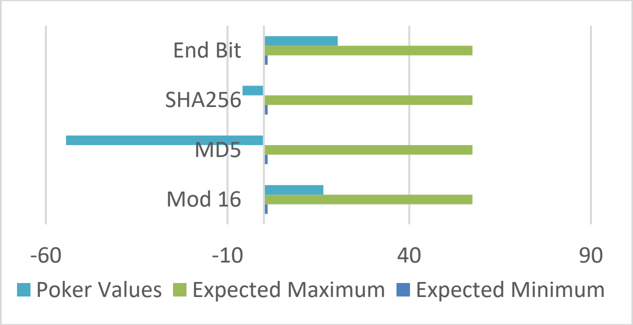

Figure 11. Monobit test chart with LDR sensor

When the data from the RGB, camera, and LDR sensors are analysed, it is seen that the RGB values are the same and the output values are similar, even if the LDR sensor values are different. When compared to the other two sensors, the camera sensor produced raw data of higher quality since it collected two distinct x and y values in each cycle that were unrelated to one another. When the approaches were analysed, it was found that the Mod 16 method produced better results and that the Md5 and Sha256 methods failed some tests.

Statistical Test Results

On a 20000 long produced sequence, the FIPS 140-1 test suite checks for randomness. There are tests for monobit, poker, runs, and long runs in this test suite. The system clock was used as a post-processing step to add more randomization to the Mod 16 method's raw value results. The positive outcomes of this test using the camera sensor and Mod 16 technique are displayed in Table 10 below.

Table 10. FIPS test results

Test	Expected		Result	
Monobit	9654< X <10346		10037	
Poker	1.03< X <57.4		7.98	
	Block Length	**Block Number Range**	**0's Number**	**1's Number**
Run	1	2267-2733	2494	2520
	2	1079-1421	1247	1191
	3	502-748	662	649
	4	223-402	297	328
	5	90-223	153	150
	6	90-223	148	163

Long Run	<= 34	Passed

One million long produced sequences are used in the NIST 800-22 test set to evaluate randomness. This test suite consists of 16 different tests, and for each test to be deemed successful, the P value must be less than 0.01 at that stage. Table 11 displays the productive outcomes of this test using the camera sensor and Mod 16 technique.

Table 11. NIST test results

No.	**Test Name**	**P Value**	**Result**
1	Frequency	0.7634	Successful
2	Block Frequency	0.1559	Successful
3	Run	0.8625	Successful
4	Test for the Longest Run of Ones in a Block	0.7775	Successful
5	Binary Matrix Rank	0.4399	Successful
6	Discrete Fourier Transform	0.6872	Successful
7	Non-Overlapping Template Mathing	0.7312	Successful
8	Overlapping Template Mathing	0.0478	Successful
9	Maurer's Universal Statistical	0.3666	Successful
10	Linear Complexity	0.8760	Successful
11	Serial - 1	0.8307	Successful
12	Serial - 2	0.8601	Successful
13	Approximate Entropy	0.8676	Successful
14	Cumulative Sums	0.9532	Successful
15	Random Excursions (x=+1)	0.3708	Successful
16	Random Excursions Variant (x=-1)	0.6782	Successful

The arrays, which give quality results with the Mod 16 technique, have been put into the NIST test suite, which is the most difficult test suite that controls randomness in the world, and also into the FIPS 140 test suite. One million sequences were produced for the NIST suite and twenty thousand sequences were produced for the FIPS test suite. It was observed that the sequences produced successfully passed all the tests in these two test packages.

CONCLUSION

In cryptographic applications, randomness is the most crucial element of security and confidentiality. Therefore, the security of the entire system is significantly impacted by the quality of random number generators utilized in various communication contexts. Because these two can be combined, random numbers can be generated as actual, fake, or hybrid. To assess the quality of these generated numbers, several statistical tests are performed. The entropy of the noise source is intimately related to the security of RNGs. By getting the seed value from the physical world using IoT sensors, this study aims to improve entropy levels. The sensors created in the Raspberry Pi environment were used to collect raw data from the Tesla sphere, the source of the noise. Eight different readings were collected using these two sensors, and they were then examined using the Mod 16, Last Bits, Sha256, and Md5 techniques. The raw data obtained with the Mod 16 approach and the camera sensor produced superior results than the other methods, according to the assessments with the Monobit and Poker tests.

As a result, it has been discovered through this project that seed values for random number generation can be derived from the sensors of IoT systems, which are currently evolving quickly and which we will encounter more frequently in the future in our daily lives. In the study of (Genç and Arslan Tuncer, 2019), which obtains random numbers with position values consisting of human movements, in order to generate numbers, the human must move, provided that it is not too small, and no data can be obtained while in a stationary position. Similarly, in the study of (Zhang et al., 2014), it is necessary to intervene in the camera with external human intervention in order to generate seed values. In contrast to the studies in the literature, using the Tesla sphere as the noise source creates a chaotic environment where random, non-repeating data are collected in the next step. It has been shown that the keys that come from the electrical radiations in the centre of the sphere are more accurate than the keys that come from pseudo-random number generators. The one million long bit string generated for the NIST randomness tests passed all of these tests. The low cost of the sensor and noise source sphere is another point that sets the project apart.

ACKNOWLEDGMENT

This study was supported by Burdur Mehmet Akif Ersoy University within the scope of Scientific Research Project No. 0800-YL-21.

REFERENCES

Abood, O.G., Guirguis S, Guirguis, S.K. (2018). A survey on cryptography algorithms. *International Journal of Scientific and Research Publications*, 8(7): 495-516.

Ansari, U., Chaudhary, A.K., Verma S. (2022). True random number generator (TRNG) using sensors for low cost IoT applications. In 2022 International Conference on Communication, Computing and Internet of Things (IC3IoT), March 10-11, 2022, Chennai, India, 1-6.

Atar, E., Ersoy, O.K., Özyılmaz, L. (2017). Hybrid data compression and optical cryptography with steep matching search method. *Journal of the Faculty of Engineering and Architecture of Gazi University*, 32(1): 139–147.

Chen, I-Te. (2013) Random numbers generated from audio and video sources. *Mathematical problems in engineering;* DOI:10.1155/2013/285373.

Conti, M., Dehghantanha, A., Franke K., Watson, S. (2018). Internet of things security and forensics: Challenges and opportunities. *Future Generation Computer Systems*, 78: 544–546.

Coşkun, A., Ülker, Ü. (2013). Development of a cryptography algorithm for national information security and reliability determination against letter frequency analysis. *Journal of Information Technologies*, 6(2): 31.

Demirkol, A.Ş. (2007). Adc based random number generator with chaotic oscillator input. PhD Thesis, Istanbul Technical University, Istanbul, Turkey.

Etem, T., Kaya, T. (2020). Trivium-linear conjugate generator based bit generation for image encryption. *Fırat University Journal of Engineering Science*, 32(1): 287–294.

Genç, Y., Arslan Tuncer, S. (2019). Human movements based true random number generation. *Bitlis Eren University Journal of Science and Technology*, 8(1): 261–269.

Gözüaçık, N. (2015). Parent based routing algorithm for rpl used in IoT networks. MSc Thesis, Istanbul Technical University, İstanbul, Türkiye.

Huang, M., Chen, Z., Zhang, Y., Guo, H. (2020). A phase fluctuation based practical quantum random number generator scheme with delay-free structure. *Applied Sciences,* 10(7): 2431.

Luengo, E.A., Cerna, M.B.L., Villalba, L.J.G., Hurley-Smith D, Hernandez-Castro J. (2022). Critical analysis of hypothesis tests in federal ınformation processing standard (140-2). *Entropy*, 24(5): 613.

Maqsood, F., Ahmed, M., Mumtaz Ali, M., Ali Shah, M. (2017). Cryptography: A comparative analysis for modern techniques. *International Journal of Advanced Computer Science and Applications*, 8(6).

Rehman, A.U., Hussain, M., Munawar, A., Attique, M., Idress, M., Anwar, F., Ahmad, M. (2020). E-cultivation using the IoT with adafruit cloud. *International Journal of Advance and Applied Sciences,* 7(9): 75–82.

Sezgin, Z.E. (2021). Tesla coil. Master Project, Maltepe University, Istanbul, Turkey.

Sunny, A.I., Zhao, A., Li, L., Kanteh Sakiliba, S. (2020). Low-cost IoT-based sensor system: A case study on harsh environmental monitoring. *Sensors*, 21(1): 214.

Tavas, V. (2011). Random number generators suitable for integration. PhD Thesis, Istanbul Technical University, Istanbul, Turkey.

Üçgün, H., Gömbeci, F., Yüzgeç, U., Yalçin, N. (2020). Real-time indoor air quality monitoring system with IoT based platform. *Bilecik Şeyh Edebali University Journal of Science and Technology,* 7(1): 370–381.

Yalman, Y., Ertürk, İ. (2016). The use of steganography in ensuring personal information security. ÜNAK Existence in the Information Age "Opportunities and Threats" Symposium, 2(2): 215.

Yaşar, S.N., Ceren Dikici, F., Tanyildizi, E., Karaköse, E. (2021). Design of a generator based on middle square and SHA3 algorithm for randomisation requirements in science and engineering studies. *Fırat University Journal of Science and Technology,* 33(1): 81–91.

Yılmaz, M., Ballı, S. (2016). Development of an intelligent selection system for the use of data encryption algorithms. *International Journal of Information Security Engineering*, 2(2): 18–28.

Yosunlu, D., Avaroğlu, E. (2020). Investigation of post processing algorithms. *Journal of Computer Science and Technology,* 1(2): 66–73.

Zhang, X., Qi, L., Tang, Z. ve Zhang, Y. (2014). Portable true random number generator for personal encryption application based on smartphone camera. *Electronics Letters,* 50(24): 1841–1843.