IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON RECENT ADVANCES ON RADIO ACCESS AND SECURITY METHODS IN 5G NETWORKS

Received December 20, 2017, accepted January 19, 2018, date of publication January 31, 2018, date of current version March 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2799682

TRNG (True Random Number Generator) Method Using Visible Spectrum for Secure Communication on 5G Network

KYUNGROUL LEE[1], (Member, IEEE), SUN-YOUNG LEE[2], CHANGHO SEO[3], AND KANGBIN YIM[2]
[1]R&BD Center for Security and Safety Industries, Soonchunhyang University, Asan 31538, South Korea
[2]Department of Information Security Engineering, Soonchunhyang University, Asan 31538, South Korea
[3]Department of the Applied Mathematics, Kongju National University, Gongju 32588, South Korea

Corresponding author: Kangbin Yim (yim@sch.ac.kr)

This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF), Ministry of Education, under Grant NRF-2015R1D1A1A01057300, and in part by the Soonchunhyang University Research Fund.

ABSTRACT The key point of cryptography is cryptographic algorithms and keys. The random number generator is used to generate seeds and keys randomly in many cryptographic systems. For this reason, it is essential to use keys to encrypt and decrypt the transferring information, and the security of these keys is closely related to the security of 5G network. We propose a true random number generator (TRNG) method in this paper, and are using visible spectrum for noise source. We consider that if the cryptography system utilizes data of visible spectrum using the proposed TRNG, and the TRNG generates random numbers with high entropy.

INDEX TERMS 5G security, random number, true random number generator, visible spectrum, random number generation, cryptography, mobile communication.

I. INTRODUCTION

A CRYPTOGRAPHIC system is a system to prevent the leakage and manipulation of the transferred information by an attacker, and the key point of cryptography is cryptographic algorithms and keys. The key is usually generated based on the seed, and the seed must be random. For this reason, it is essential to use keys to encrypt and decrypt the transferred information. Moreover, these random values are utilized not only for keys, but also for various applications. For example, there application are that generation of padding bits, generation of masking to prevent DPA (Differential Power Analysis), hiding operation, generation of OTP (One Time Password), lottery, and statistical simulation [1]. Especially, in 5G network, various information is transmitted between communication entities, and cryptography is required to transmit the information securely. Hence, it is essential to use a key for encryption and decryption of transferring data, and security of these keys is closely related to the security of 5G network.

There are two types of random number generators: PRNG (Pseudo Random Number Generator) and TRNG (True Random Number Generator). PRNG intentionally generates a random number, not a complete random number, and the number is usually generated by a software way. TRNG, on the other hand, generates a true random number and is generated primarily by hardware way [2]. For this reason, the random number generated by TRNG is hard to predict because TRNG is generated based on a physical source that is difficult to predict a random value. Therefore, the random number generated from TRNG is a safe method because it is difficult to generate the same value. However, PRNG is a deterministic system so the generated random number can be guaranteed to be safe when the entropy of initial value has high entropy, but it is impossible to generate the complete random number in the deterministic system. Hence, there is a problem that the attacker conjectures the same key derived from same random number by generating the same initial value such as seed. In addition, it is impossible to generate a complete random number using only mathematical algorithms.

Therefore, this paper proposes a method of TRNG and uses visible spectrum as a noise source. Visible spectrums have difference values even at the same position and the same time, and the range of values is relatively wide. Namely, it is impossible to generate a random value that cannot be predicted even if the environment is the same. In particular, proposed TRNG can provide more secure communication because the TRNG can generate a random value and utilize it as a key for each entity, for example a base band and a station, in 5G network.

This paper is organized as in the following. In section 2, we describe related work, including types of random numbers, generation methods, and existing random number generators. In section 3, we introduce proposed TRNG, including concept, system model, and experiment results. Conclusion and references close the paper.

II. RELATED WORKS

A. TYPES OF RANDOM NUMBERS AND GENERATION METHODS

Random number can be generated in various ways, usually with PRNG and TRNG. The difference between PRNG and TRNG is deterministic, PRNG is a deterministic random number generator, and TRNG is a non-deterministic random number generator.

PRNG generates a long-length random number using algorithms based on a short initial value. The generated random number is called the pseudo random number, which has excellent statistical property [3]. However, the disadvantage is that it is difficult to generate a random initial value and it has a problem with a limited period [4]. For this reason, it is impossible to generate complete random number by using the deterministic random number generator. These generators are LFSR (Linear Feedback Shift Register), LCG (Linear Congruential Generator), multiplication system, and so on [5]–[7].

TRNG generates random number using the randomness of physical phenomena. The generated random number is called a true random number, which is unpredictable, unbiased, and independent. However, the disadvantage is that it is not enough flexibility because it is implemented in hardware, and verification of randomness is required because the randomness is changed according to environment.

The generating method of above random number is classified into two ways; software method and hardware method. A software method is generating random number based on information of a computer, and mainly utilizes information generated during a user-computer interaction. Such information includes network traffic information, hard disk operation information, and so on. Therefore, it is possible to generate a random number based on such information; there is a problem that the same random number is generated when the same environment is configured. For example, when an attacker manipulates information depends on a computer platform such as process, the same random number is generated. This means that the same key can be generated based on the generated same random number. Unlike the software method, the hardware method generates random number based on unpredictable physical phenomena and has been proposed to solve the drawbacks of the software methods [8]–[10]. This method is needed an extra device to collect the random number from physical phenomena, and usually utilizes noise. However, there are disadvantages that the extra device is required and it is difficult to implement.

TABLE 1. Classification of random number generation methods.

Category	Generation methods	Description
Classical method	- Throwing a coin - Throwing a dice	- Cause of occurrence: probability
White noise method	- Thermal noise - Shot noise	- Cause of occurrence: noise affected by the environment - Generation material: register, small AC voltage, polarity semiconductor
Jittered oscillator method	- jitter - clock	- Cause of occurrence: Unstable of oscillator - Generation way: change the fast oscillator signal to the slower oscillator signal
Unstable state method	- Connecting inverter input and output - Disconnecting inverter input and output	- Cause of occurrence: Unstable state of digital circuit - Generation way: Switching from a metastable state to a bitstable state in an actual circuit
Chaotic signal method	- Non-linear shape of the natural world	- Cause of occurrence: Chaotic signal with aperiodic irregularity - Generation way: Sampling values from chaotic circuits

B. EXISTING RANDOM NUMBER GENERATORS

1) GENERATION METHODS OF RANDOM NUMBER

The random number generation methods are classified into five methods as shown in Table 1, a classical method, a white noise method, a jittered oscillator, an unstable state, and a chaotic signal.

The classical method is to generate random number based on probability, and there are throwing a coin and throwing a dice. This method is generated values unexpectedly such as coin and dice. However, this method needs a lot of manpower due to manually generation, so it cannot use actual system [4].

White noise method utilizes noise generated in the supply portion of the power supply to generate random number, using resister, small AC voltage, and polarity semiconductor. Noise is generated by thermal noise and shot noise, and these noises are outpoured week signals [11]–[14]. Therefore, the random number is generated based on the amplified signal because the generated value can be biased when sampling the signal. Generated random number varies irregularly depending on the environment in which when circuit is designed, the error of the machine, and the environment in which when circuit is running.

Jittered oscillator method utilizes the instability of the oscillator to generate random number [14], [15]. Way to generate noise is to affect from a fast oscillator signal to a slower oscillator signal, and the signal output from the oscillator is waved in this case and is called jitter. The reason for this phenomenon is that although the oscillator generates signal with a constant value of 0 and 1, but the part changing from 0 to 1 and the part changing from 1 to 0 are not accurate due to the influence of the process and the circuit driving environment, so this causes an error. Therefore, the fast oscillator signal is sampled with D flip-flop using a slower clock signal to generate a random number with high entropy [11], [13], [15]. Moreover, there is a method of changing the signal of fast oscillator and processing xor operation with several fast oscillators to increase entropy [16]–[19]. Thus, the generated value is influenced by the surrounding environment, so the value cannot be predicted.

Unstable state method utilizes the phenomenon that the output of the circuit changes unstably to 0 or 1 to generate a random number. Digital circuit has only bistable state, which is 0 or 1 state. However, in an actual circuit, there is a metastable state which is unstable state, not 0 or 1. Therefore, the output is waved by the noise between 0 and 1, and the output value is determined 0 or 1 randomly when disconnect in a metastable state. This means that the generated information at that time of disconnection is determined by the noise, so it is impossible to predict the generated value [20]–[24].

Chaotic signal utilized the sampling value from the chaotic signal, which is a nonlinear characteristic of a natural world, to analog signal [25], [26]. Even if an ideal system is designed for a chaotic system, the actual system has an error, and such error as noise is sufficient to be used as a random number. For this reason, the generated random number is irregular and difficult to predict. Therefore, if all the conditions such as the initial value and the parameter are not completely same, a completely different value is generated. This means that this method is effective for generating the random number. However, there is a disadvantage that the generated value is limited to a specific range [25], [27].

2) THE SOURCE OF RANDOM NUMBER GENERATOR

The source of random number generator is classified into six sources as shown in Table 2, electrical and electronic circuits, external and auxiliary devices, system information, chaotic signal, natural world system, and others.

Electrical and electronic circuits are classified as semiconductor device, super-luminescent LED, and flip-flop. Semiconductor device is highly affected by the environment, and various noise caused by this environment can be used as a source for a ransom number. These noises include thermal noise, flicker noise, shot noise, avalanche noise, generation/recombination noise, and noise diode/resistance. As with semiconductor device, super luminescent LED emits noise, which is used as the random number source. Flip-flop has metastability state, so noise causes from the state.

External and auxiliary devices are classified as a monitor, disk, MIC, TV/radio, laser, and digital camera. Monitor utilizes noise emitted from a radiation monitor with RS-232 output as a random number source. Disk generates the random number based on air convection inside the disk drive due to rotation speed, space, air flow by heat condition, movement of head and support, etc. MIC utilizes noise of collected signal from MIC or the difference between two microphone signals as the source. TV/radio utilizes noise, including TV/radio broadcasting signal as the source. Laser utilizes the phase noise of the laser and digital camera utilizes noise, including image collected from camera.

TABLE 2. Classification of the random number generator sources.

Category	Source	Description
Electrical and electronic circuits	Semiconductor device [28, 29, 30, 31, 32, 33, 34, 35, 36, 37]	- thermal noise - Flicker noise - shot noise - Avalanche noise - diffusion noise - generation/recombination noise - noise diode/resistance
	Super-luminescent LED [38]	- Noise from LED
	Flip-flop [38]	- Metastability of flip-flop
External and auxiliary devices	Monitor	- Radiation monitor with RS-232 output
	Disk	- Air convection inside the disk drive due to rotation speed, space, air flow by heat condition, movement of head and support, etc.
	MIC (microphone)	- Noise from MIC - Difference between two microphone signals
	TV/Radio	- Broadcasting signal TV/radio
	Laser [38, 39]	- The phase noise of the laser
	Digital camera [40, 41, 42]	- Image with noise from digital camera
System information	Network and process	- Information about network traffic and running processes
	I/O device	- I/O completion timing and statistic
	Time	- System date and time
	RAM [38]	- Block RAM write conflict
Chaotic signal	Laser [43]	- Chaotic laser
	Chaos system [44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54]	- Well-known system: Lorenz system, the Logistic map, the Henonmap, the Rossler system, and the double rod pendulum - Dynamical system: Chua's system, the Tent map, and the Sawtooth map
Natural world system	Light	- Photon polarization detection
	Amount of insolation [40, 41, 42]	- Amount of insolation of measured time
	Atmospheric dynamics [40, 41, 42]	- Atmospheric flow of measured time
	Radioactive material [55]	- the decay of radioactive nucleus
	Single photon [56, 57]	- Irregular split time and arrival time
	Natural emission [58]	- The naturally emitted noise signal
Others	Publication	- Information on publications such as newspaper, magazine, and book
	Recording media	- Information recorded on CD ROMs, Audio CDs, and tapes

System information utilizes unpredictable information during system running as a random number source, and this source is classified into the network and process, I/O device, time, and RAM. Network and process utilize the information about network traffic and running processes as a random number source, and I/O device utilizes I/O completion timing and statistic as the source. Time generated random number based on the system date and time from when the system is booted, and RAM utilizes write conflict caused by block RAM as the source.

Chaotic signal is classified into the laser and chaotic system. Laser utilizes the chaotic signal from the laser as a random number. Chaotic system utilizes information collected from well-known chaotic and dynamical system as the random number.

Natural world system is classified as light, amount of insolation, atmospheric dynamics, radioactive material, single photon, and natural emission. Light utilizes noise emitted from the feature that the light is randomly polarized, and amount of insolation utilizes irregular amount of insolation of measured time as the random number. Atmospheric dynamics utilizes the atmospheric flow of measured time as the random number, and radioactive material utilizes the decay of radioactive nucleus. Single photon generated random number based on irregular split time and arrival time, and natural emission utilized noise signal emitted naturally as the source of the random number. Others are that publication source utilizes information on publications such as newspaper, magazine, and book, and recording media source utilizes information recorded on CD ROMs, Audio CDs, and tapes.

Moreover, the following articles have recently been studied. The fingerprint is collected every time, noise occurs due to the environment, the collecting way, and contact area, and this noise is utilized as a random number [59]. A new method using image collected from camera has been studied, this method after sequential two images are subtracted, if the result is a positive number, it is determined to be 1, and if the result if negative, it is determined to be 0. [60]. In the method using the Moire fringe, a difference in lattice frequency occurs in a state in which lattices having two similar spatial phases are overlapped, and the low-frequency component, among these, is called the Moire interference fringe. These fringes are irregular, so these are used as random number [61].

Color	Violet	Blue	Green	Yellow	Orange	Red
Wavelength	380-410 nm	415-495 nm	495-570 nm	570-590 nm	590-620 nm	620-750 nm
Frequency	668-789 THz	606-668 THz	526-606 THz	508-526 THz	484-508 THz	400-484 THz
Photon energy	2.75-3.26 ev	2.50-2.75 ev	2.17-2.50 ev	2.10-2.17 ev	2.00-2.10 ev	1.65-2.00 ev

FIGURE 1. The visible spectrum.

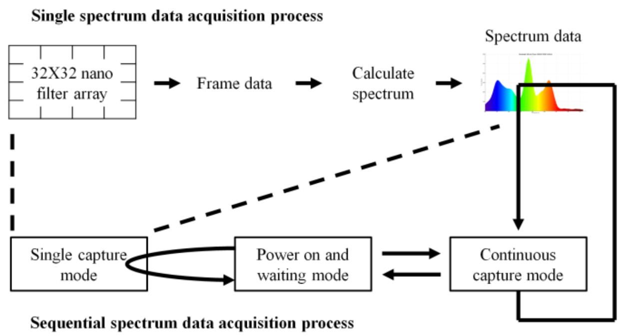

FIGURE 2. Spectrum data acquisition process.

TABLE 3. Part of the collected visible spectrum.

Wavelength	400	401	…	729	730
Spectrum-1	0.0924933	0.10209	…	0.0608392	0.0585175
Spectrum-2	0.108507	0.118519	…	0.0648131	0.0609462
…	…	…	…	…	…
Spectrum-999	0.196799	0.193059	…	0.141342	0.134564
Spectrum-1000	0.193488	0.188438	…	0.156311	0.157035

III. PROPOSED TRNG

A. CONCEPT

Although there are various random number generators as described above, there is a problem that these generators utilize a limited and insufficient data as a random number and a source. We consider that one of the important factors is to generate random numbers based on sufficient data from the source. Therefore, proposed TRNG generates random numbers from sufficient information and its noise, and use visible spectrum that can generate more noise to do this.

Visible spectrum is generally the region of electromagnetic waves in the rage of 400nm to 700nm, as shown in figure 1, and this area is called spectrum [62]. A spectrum has information of more than 300 electromagnetic waves, and the range of the information also has a wide range of values corresponding to the range of electromagnetic waves. Therefore, the visible spectrum contains a lot of information that can be used as a random number, so this is suitable for use as the random number because the information is originated from electromagnetic waves.

Detailed process and procedure for acquiring visible spectrum are shown in Fig. 2. The inputted illumination is converted into frame data by 32X32 nano filter array, and the data is calculated to represent the wavelength as a spectrum. Through this process, spectrum data corresponding to inputted illumination is acquired. This process extracts a single spectrum, which operates in a single capture mode. To generate continuous random numbers, spectrum data must be continuously collected, so the spectrum data is collected the number of times in continuous mode when specific number of times is set. In this mode, parameters such as shutter speed, average, and number of times required for acquisition are set.

As shown in Figs. 1 and 2, visible spectrum has various information in wavelength, frequency, and photon energy. In this paper, we collect data of visible spectrum based on wavelength, and some of the results are shown in Table 3 and Fig. 3.

As shown in Fig. 3, the collected spectrum has not distributed pattern. In order to improve the randomness of the collected data, we take two decimal places, and the distribution is shown in Fig. 4.

As shown in Fig. 4, the two decimal places are processed as data to generate a random number, and the data are appropriately distributed.

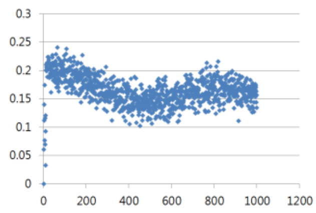

FIGURE 3. Collected spectrum data.

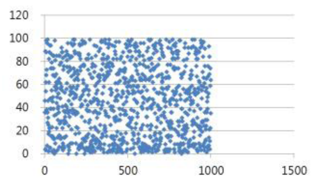

FIGURE 4. Distribution of two decimal places of spectrum data.

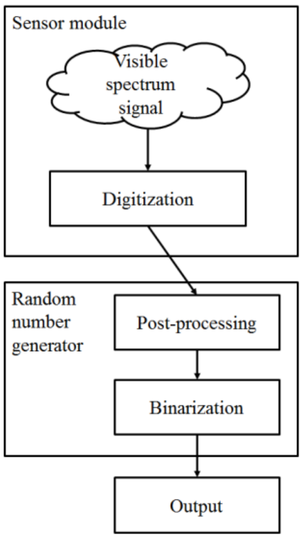

FIGURE 5. TRNG configuration (model).

B. SYSTEM MODEL

The proposed TRNG model is shown in Fig. 5, and the module consists of a collection module, a generation module, and an output module based on NIST 800-90b [3].

The collection module is a sensor module. This module consists of a sensor that collects data of the visible spectrum and an interface to transfer collected information to a computer. Among them, the sensor collecting visible spectrum information is a noise source, and the sensor converts the analog signal to digital information and transmits it. The collected information is passed to a generation module, which performs post-processing to generate a random number. The post-processing process increases the distribution as described in Section III-A, reduces the correlation between each spectrum, so the result improves the randomness. The post-processing result is binarized to measure the entropy, so the result value has 0 or 1. The binarized result from the generation module transfers to the output module, the module saves all collected results as a file in a format for measuring the entropy.

Based on the system model, the procedure for acquiring spectrum data is described as follows:

Step 1: The sensor module collects the analog signal of illumination from a 32X32 nano filter array. This signal is not a digital signal, but an analog signal with a lot of noise.

Step 2: In the sensor module, the analog signal collected in step 1 is converted into frame data, which is a digital signal. After calculating the spectrum based on the converted digital signal, the calculation result is composed of spectrum data. This is the digitization step.

Step 3: When spectrum data is collected, the data is performed a post-processing process for using random number by random number generator. The post-processing process enhances the distribution of the collected spectrum data, reduces the correlation between the spectrum data and improves the randomness. In this paper, we utilize two decimal places as random numbers to improve the randomness of collected data.

Step 4: In this paper, we decide the size of the generated random number to be 1 bit, and perform binarization with 0 and 1 based on the result processed in step 3. The result of the binarization is transferred to the output module and stored as a file. Finally, a random number having a size of 1 bit is generated using the stored file.

C. EXPERIMENT RESULT

In this paper, we collected 1,000 spectrums for the experiment, and collected the output results through processing described in section 3.2. The data collected in one spectrum collects a total of 331 spectrum data from 400nm to 730nm in wavelength, so total number of collected data is 331,000 because total 1,000 spectrums are collected.

There are various methods to measure the randomness of collected random data. In this paper, we use the entropy estimation tool provided by NIST. NIST has released a tool for measuring the entropy of generated random numbers, and now 800-90b has been released [3]. 800-90b estimates the randomness by other estimation methods according to the properties of the random number, and the estimation methods are classified into independent and identically distributed (IID) and non-IID. IID estimates when the generated random number is independent, while non-IID estimates when the generated random number is not independent. In this paper, we estimate the entropy by non-IID estimation method because the collected random number is based on the continuously collected spectrums. Entropy was estimated based on four consecutive spectrum data, and the results are shown in Table 4 and Fig. 6.

Non-IID means that there is a dependency between data. Namely, it is possible to predict the value that will appear later through the previous value. Therefore, various entropy estimation methods are needed for dependent values having sequence. The methods provided by NIST obtain the probability of correctly estimating next value using the dependencies between the data in various ways, and the entropy is calculated according to the probability obtained.

TABLE 4. Min-entropy of first estimation result.

Estimation method		TEST 1	TEST 2	TEST 3	TEST 4	NIST sample
dRunning entropic statistic	Most Common Value	P(max): 0.502621 Min-entropy: 0.992457	P(max): 0.502756 Min-entropy: 0.992043	P(max): 0.503695 Min-entropy: 0.989377	P(max): 0.504939 Min-entropy: 0.985819	P(max): 0.501721 Min-entropy: 0.995043
	Collision	P(max): 0.604492 Min-entropy: 0.726204	P(max): 0.604004 Min-entropy: 0.72737	P(max): 0.600586 Min-entropy: 0.735557	P(max): 0.606201 Min-entropy: 0.722131	P(max): 0.535156 Min-entropy: 0.901968
	Markov	P(max): 6.02073e-39 Min-entropy: 0.991916	P(max): 6.32058e-39 Min-entropy: 0.991368	P(max): 7.14917e-39 Min-entropy: 0.98998	P(max): 9.97228e-39 Min-entropy: 0.986229	P(max): 4.00013e-39 Min-entropy: 0.996525
	Compression	P(max): 0.553284 Min-entropy: 0.853909	P(max): 0.552856 Min-entropy: 0.855023	P(max): 0.552307 Min-entropy: 0.856457	P(max): 0.554565 Min-entropy: 0.85057	P(max): 0.5 Min-entropy: 1
	t-Tuple	P(max): 0.609642 Min-entropy: 0.713966	P(max): 0.586789 Min-entropy: 0.769086	P(max): 0.603412 Min-entropy: 0.728786	P(max): 0.59119 Min-entropy: 0.758306	P(max): 0.529343 Min-entropy: 0.917726
	LRS	P(max): 0.520513 Min-entropy: 0.941995	P(max): 0.517699 Min-entropy: 0.949815	P(max): 0.573474 Min-entropy: 0.8022	P(max): 0.537049 Min-entropy: 0.896874	P(max): 0.503651 Min-entropy: 0.989503
Running predictor	MultiMCW Prediction	P(max): 0.556115 Min-entropy: 0.846544	P(max): 0.50115 Min-entropy: 0.996684	P(max): 0.516113 Min-entropy: 0.95424	P(max): 0.526367 Min-entropy: 0.925859	P(max): 0.501051 Min-entropy: 0.996972
	Lag Prediction	P(max): 0.503077 Min-entropy: 0.991149	P(max): 0.501833 Min-entropy: 0.99472	P(max): 0.501747 Min-entropy: 0.994969	P(max): 0.502816 Min-entropy: 0.991897	P(max): 0.500592 Min-entropy: 0.998291
	MultiMMC Prediction	P(max): 0.578522 Min-entropy: 0.789556	P(max): 0.579763 Min-entropy: 0.786466	P(max): 0.58071 Min-entropy: 0.784111	P(max): 0.582543 Min-entropy: 0.779563	P(max): 0.501159 Min-entropy: 0.99666
	LZ78Y Prediction	P(max): 0.568014 Min-entropy: 0.816001	P(max): 0.502777 Min-entropy: 0.99201	P(max): 0.542969 Min-entropy: 0.881059	P(max): 0.503899 Min-entropy: 0.988792	P(max): 0.501024 Min-entropy: 0.99705
Min-entropy		0.713966		0.728786	0.722131	0.901968

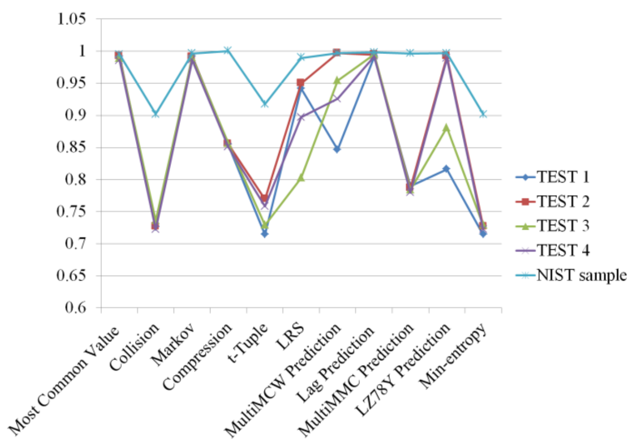

FIGURE 6. First estimation result.

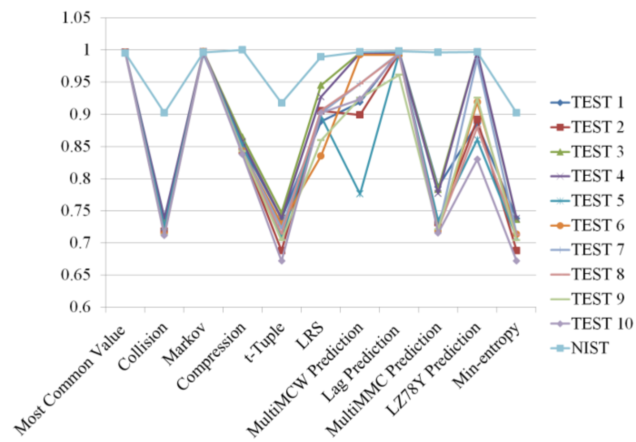

FIGURE 7. Results of a second experiment based on 1,000,000 data.

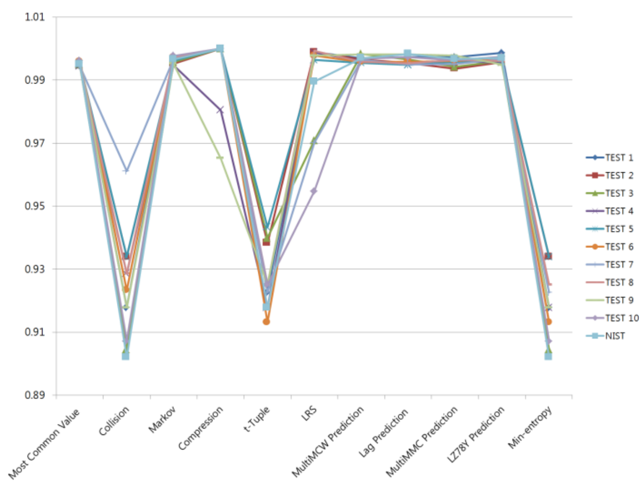

FIGURE 8. Experiment results after XOR operation with the sample provided by NIST.

The performance of TRNG is estimated by the calculated entropy. NIST recommends the following ten entropy estimation methods:

The Most Common Value Estimate: Calculate the entropy using the ratio of the most frequently appearing values from the input dataset.

The Collision Estimate: If the collision is an arbitrary repetitive value, the probability of the most frequently appearing output value is estimated based on the time until the collision occurs. When the collision time is longer, the entropy is calculated the high value.

The Markov Estimate: The min-entropy is calculated by measuring the dependency between successive values from the input dataset.

The Compression Estimate: The entropy of the dataset is calculated based on how much the dataset can be compressed.

The t-Tuple Estimate: The entropy is calculated per sample based on the frequency of the t-tuple appearing in the input dataset.

The Longest Repeated Substring (LRS) Estimate: The collision entropy of the source is calculated based on the number

TABLE 5. Results of a second experiment based on 1,000,000 data.

Estimation method		TEST 1	TEST 2	TEST 3	TEST 4	TEST 5
dRunning entropic statistic	Most Common Value	P(max): 0. 501446	P(max): 0. 50124	P(max): 0. 501545	P(max): 0. 501238	P(max): 0. 501515
		Min-entropy: 0. 995833	Min-entropy: 0. 996426	Min-entropy: 0. 99555	Min-entropy: 0. 996432	Min-entropy: 0. 995635
	Collision	P(max): 0. 599121	P(max): 0. 60791	P(max): 0. 600098	P(max): 0. 599609	P(max): 0. 60498
		Min-entropy: 0. 73908	Min-entropy: 0. 71807	Min-entropy: 0. 736731	Min-entropy: 0. 737905	Min-entropy: 0. 72504
	Markov	P(max): 4.9635e-39	P(max): 3.90025e-39	P(max): 3.70529e-39	P(max): 3.70032e-39	P(max): 4.33277e-39
		Min-entropy: 0. 994092	Min-entropy: 0. 99681	Min-entropy: 0. 997388	Min-entropy: 0. 997403	Min-entropy: 0. 995624
	Compression	P(max): 0. 554138	P(max): 0. 557373	P(max): 0. 549194	P(max): 0. 552002	P(max): 0. 553955
		Min-entropy: 0. 851682	Min-entropy: 0. 843285	Min-entropy: 0. 864611	Min-entropy: 0. 857255	Min-entropy: 0. 852159
	t-Tuple	P(max): 0. 599777	P(max): 0. 620778	P(max): 0. 595942	P(max): 0. 591182	P(max): 0. 611909
		Min-entropy: 0. 737502	Min-entropy: 0. 68785	Min-entropy: 0. 746757	Min-entropy: 0. 738326	Min-entropy: 0. 708611
	LRS	P(max): 0. 540092	P(max): 0. 533942	P(max): 0. 519429	P(max): 0. 525879	P(max): 0. 536994
		Min-entropy: 0. 888724	Min-entropy: 0. 905244	Min-entropy: 0. 945001	Min-entropy: 0. 927197	Min-entropy: 0. 897022
Running predictor	MultiMCW Prediction	P(max): 0. 528984	P(max): 0. 53879	P(max): 0. 501408	P(max): 0. 501671	P(max): 0. 583984
		Min-entropy: 0. 918705	Min-entropy: 0. 8992205	Min-entropy: 0. 995942	Min-entropy: 0. 995186	Min-entropy: 0. 775998
	Lag Prediction	P(max): 0. 500772	P(max): 0. 501648	P(max): 0. 501372	P(max): 0. 501212	P(max): 0. 501953
		Min-entropy: 0. 997773	Min-entropy: 0. 995253	Min-entropy: 0. 996047	Min-entropy: 0. 996507	Min-entropy: 0. 994375
	MultiMMC Prediction	P(max): 0. 579471	P(max): 0. 602237	P(max): 0. 581058	P(max): 0. 583623	P(max): 0. 601066
		Min-entropy: 0. 787191	Min-entropy: 0. 731598	Min-entropy: 0. 783247	Min-entropy: 0. 776891	Min-entropy: 0. 734404
	LZ78Y Prediction	P(max): 0. 541219	P(max): 0. 53879	P(max): 0. 501036	P(max): 0. 501077	P(max): 0. 550781
		Min-entropy: 0. 885716	Min-entropy: 0. 892205	Min-entropy: 0. 997014	Min-entropy: 0. 996894	Min-entropy: 0. 860449
Min-entropy		0. 737502	0. 68785	0. 736731	0. 737905	0. 708611

Estimation method		TEST 6	TEST 7	TEST 8	TEST 9	TEST 10
dRunning entropic statistic	Most Common Value	P(max): 0. 501236	P(max): 0. 501388	P(max): 0. 501287	P(max): 0. 50136	P(max): 0. 501157
		Min-entropy: 0. 996439	Min-entropy: 0. 996	Min-entropy: 0. 996292	Min-entropy: 0. 996082	Min-entropy: 0. 996666
	Collision	P(max): 0. 609863	P(max): 0. 609863	P(max): 0. 610352	P(max): 0. 610352	P(max): 0. 61084
		Min-entropy: 0. 713442	Min-entropy: 0. 713442	Min-entropy: 0. 712288	Min-entropy: 0. 712288	Min-entropy: 0. 71134
	Markov	P(max): 3.58016e-39	P(max): 3.67234e-39	P(max): 3.82145e-39	P(max): 3.81106e-39	P(max): 3.73808e-39
		Min-entropy: 0. 997775	Min-entropy: 0. 997488	Min-entropy: 0. 99704	Min-entropy: 0. 99707	Min-entropy: 0. 997288
	Compression	P(max): 0. 557373	P(max): 0. 55719	P(max): 0. 557861	P(max): 0. 557739	P(max): 0. 559509
		Min-entropy: 0. 843285	Min-entropy: 0. 843759	Min-entropy: 0. 842022	Min-entropy: 0. 842337	Min-entropy: 0. 837766
	t-Tuple	P(max): 0. 603062	P(max): 0. 606176	P(max): 0. 609549	P(max): 0. 613911	P(max): 0. 627768
		Min-entropy: 0. 729621	Min-entropy: 0. 722192	Min-entropy: 0. 714185	Min-entropy: 0. 703898	Min-entropy: 0. 671697
	LRS	P(max): 0. 560548	P(max): 0. 53501	P(max): 0. 533554	P(max): 0. 551183	P(max): 0. 535142
		Min-entropy: 0. 835091	Min-entropy: 0. 902361	Min-entropy: 0. 906294	Min-entropy: 0. 859396	Min-entropy: 0. 902006
Running predictor	MultiMCW Prediction	P(max): 0. 502643	P(max): 0. 518358	P(max): 0. 518358	P(max): 0. 526526	P(max): 0. 527344
		Min-entropy: 0. 992394	Min-entropy: 0. 94798	Min-entropy: 0. 94798	Min-entropy: 0. 925422	Min-entropy: 0. 923184
	Lag Prediction	P(max): 0. 502643	P(max): 0. 501685	P(max): 0. 502081	P(max): 0. 513654	P(max): 0. 501646
		Min-entropy: 0. 992394	Min-entropy: 0. 995147	Min-entropy: 0. 994008	Min-entropy: 0. 96113	Min-entropy: 0. 995258
	MultiMMC Prediction	P(max): 0. 60773	P(max): 0. 606887	P(max): 0. 606433	P(max): 0. 606995	P(max): 0. 60897
		Min-entropy: 0. 718498	Min-entropy: 0. 7205	Min-entropy: 0. 72158	Min-entropy: 0. 720244	Min-entropy: 0. 715556
	LZ78Y Prediction	P(max): 0. 528984	P(max): 0. 504886	P(max): 0. 543387	P(max): 0. 526526	P(max): 0. 5625
		Min-entropy: 0. 918705	Min-entropy: 0. 985972	Min-entropy: 0. 879947	Min-entropy: 0. 925422	Min-entropy: 0. 830075
Min-entropy		0. 713442	0. 713442	0. 712288	0. 703898	0. 671697

of substrings that are repeated within the input dataset. This is a complementary estimation method that the size of the tuple can adjust when the size of the tuple is too long in the T-tuple estimate.

The Multi Most Common in Window Prediction Estimate: Based on the last w output, the next output is predicted, and the most common sample value is predicted. This method is designed for cases when the most common value changes over time.

The Lag Prediction Estimate: The entropy is calculated by obtaining the probability of predicting the next output based on the specified lag.

The Multi MMC Prediction Estimate: This constructs Multiple Markov Model with Counting (MMC). The entropy is calculated to predict the most frequently observed transitions from current output by observing the transition frequency of successive next outputs from one output.

The LZ78Y Prediction Estimate: This is used a method of adding a string to a dictionary based on LZ78 code. Predictor adds the new string to the dictionary until the string reaches the maximum capacity of the dictionary. The entropy of the prediction success probability is calculated by using a dictionary to predict the next value.

The Multi Most Common in Window Prediction Estimate, The Lag Prediction Estimate, and The Multi MMC Prediction Estimate are calculated the entropy using a predictor selected within the highest probability of prediction success among several subpredictors. Otherwise, The LZ78Y Prediction Estimate is calculated as a probability to correctly predict the next value using values in the dictionary.

NIST recommends estimating entropy based on 1,000,000 data. However, the data collected from one test were 331,000, so the entropy was estimated based on insufficient data. Therefore, we estimate the entropy based on collected 1,000,000 data, the result is shown in Table 5 and Fig. 7.

As shown in Fig. 6, three experiment results have lower entropy than the sample provided by NIST. The reason is that the collected random number is not generated with high entropy having a certain probability, but has low entropy because the data is generated according to the visible

TABLE 6. Experiment results after XOR operation with the sample provided by NIST.

Estimation method		TEST 1	TEST 2	TEST 3	TEST 4	TEST 5
dRunning entropic statistic	Most Common Value	P(max): 0. 501343 Min-entropy: 0.99613	P(max): 0. 501783 Min-entropy: 0.994865	P(max): 0. 50183 Min-entropy: 0.994729	P(max): 0. 501953 Min-entropy: 0.994376	P(max): 0. 501697 Min-entropy: 0.995112
	Collision	P(max): 0. 529297 Min-entropy: 0.917851	P(max): 0. 523438 Min-entropy: 0.933911	P(max): 0. 53418 Min-entropy: 0.904603	P(max): 0. 525391 Min-entropy: 0.928538	P(max): 0. 523438 Min-entropy: 0.933911
	Markov	P(max): 3.71261e-39 Min-entropy: 0.997365	P(max): 4.49904e-39 Min-entropy: 0.9952	P(max): 4.32946e-39 Min-entropy: 0.995633	P(max): 4.70339e-39 Min-entropy: 0.994699	P(max): 4.12048e-39 Min-entropy: 0.99619
	Compression	P(max): 0. 5 Min-entropy: 1	P(max): 0. 5 Min-entropy: 1	P(max): 0. 5 Min-entropy: 1	P(max): 0. 506836 Min-entropy: 0.980409	P(max): 0. 5 Min-entropy: 1
	t-Tuple	P(max): 0. 527557 Min-entropy: 0.922601	P(max): 0. 52187 Min-entropy: 0.938238	P(max): 0. 521265 Min-entropy: 0.939911	P(max): 0. 529343 Min-entropy: 0.917726	P(max): 0. 520025 Min-entropy: 0.943346
	LRS	P(max): 0. 500408 Min-entropy: 0.998824	P(max): 0. 50039 Min-entropy: 0.998874	P(max): 0. 510206 Min-entropy: 0.970847	P(max): 0. 500434 Min-entropy: 0.998747	P(max): 0. 501251 Min-entropy: 0.996395
Running predictor	MultiMCW Prediction	P(max): 0. 501079 Min-entropy: 0.996891	P(max): 0. 501194 Min-entropy: 0.99656	P(max): 0. 500588 Min-entropy: 0.998303	P(max): 0. 501355 Min-entropy: 0.996097	P(max): 0. 501576 Min-entropy: 0.995461
	Lag Prediction	P(max): 0. 500944 Min-entropy: 0.99728	P(max): 0. 501568 Min-entropy: 0.995484	P(max): 0. 501205 Min-entropy: 0.996529	P(max): 0. 501783 Min-entropy: 0.994866	P(max): 0. 501787 Min-entropy: 0.994854
	MultiMMC Prediction	P(max): 0. 500929 Min-entropy: 0.997322	P(max): 0. 502208 Min-entropy: 0.993643	P(max): 0. 502017 Min-entropy: 0.994192	P(max): 0. 501359 Min-entropy: 0.996084	P(max): 0. 501436 Min-entropy: 0.995863
	LZ78Y Prediction	P(max): 0. 50047 Min-entropy: 0.998643	P(max): 0. 501519 Min-entropy: 0.995625	P(max): 0. 501358 Min-entropy: 0.996088	P(max): 0. 501597 Min-entropy: 0.995401	P(max): 0. 501334 Min-entropy: 0.996157
Min-entropy		0.917851	0.933911	0.904603	0.917726	0.933911

Estimation method		TEST 6	TEST 7	TEST 8	TEST 9	TEST 10
dRunning entropic statistic	Most Common Value	P(max): 0. 501431 Min-entropy: 0.995877	P(max): 0. 50196 Min-entropy: 0.994356	P(max): 0. 501595 Min-entropy: 0.995405	P(max): 0. 501387 Min-entropy: 0.996004	P(max): 0. 501372 Min-entropy: 0.996047
	Collision	P(max): 0. 527344 Min-entropy: 0.923184	P(max): 0. 513672 Min-entropy: 0.961081	P(max): 0. 525391 Min-entropy: 0.928538	P(max): 0. 529297 Min-entropy: 0.917851	P(max): 0. 533203 Min-entropy: 0.907243
	Markov	P(max): 3.72164e-39 Min-entropy: 0.997338	P(max): 3.95212e-39 Min-entropy: 0.996661	P(max): 3.7549e-39 Min-entropy: 0.997238	P(max): 4.48173e-39 Min-entropy: 0.995243	P(max): 3.61062e-39 Min-entropy: 0.997679
	Compression	P(max): 0. 5 Min-entropy: 1	P(max): 0. 5 Min-entropy: 1	P(max): 0. 5 Min-entropy: 1	P(max): 0. 512207 Min-entropy: 0.965201	P(max): 0. 5 Min-entropy: 1
	t-Tuple	P(max): 0. 531042 Min-entropy: 0.913101	P(max): 0. 527557 Min-entropy: 0.922601	P(max): 0. 526629 Min-entropy: 0.925141	P(max): 0. 526629 Min-entropy: 0.925141	P(max): 0. 526629 Min-entropy: 0.925141
	LRS	P(max): 0. 500792 Min-entropy: 0.997717	P(max): 0. 510478 Min-entropy: 0.97008	P(max): 0. 500274 Min-entropy: 0.99921	P(max): 0. 500765 Min-entropy: 0.997794	P(max): 0. 515954 Min-entropy: 0.954686
Running predictor	MultiMCW Prediction	P(max): 0. 501517 Min-entropy: 0.995631	P(max): 0. 501231 Min-entropy: 0.996454	P(max): 0. 501473 Min-entropy: 0.995757	P(max): 0. 500649 Min-entropy: 0.998127	P(max): 0. 501132 Min-entropy: 0.996739
	Lag Prediction	P(max): 0. 501453 Min-entropy: 0.995815	P(max): 0. 501756 Min-entropy: 0.994944	P(max): 0. 501612 Min-entropy: 0.995358	P(max): 0. 500627 Min-entropy: 0.998191	P(max): 0. 500962 Min-entropy: 0.997228
	MultiMMC Prediction	P(max): 0. 501373 Min-entropy: 0.996044	P(max): 0. 501691 Min-entropy: 0.995129	P(max): 0. 501251 Min-entropy: 0.996395	P(max): 0. 500784 Min-entropy: 0.99774	P(max): 0. 501273 Min-entropy: 0.996332
	LZ78Y Prediction	P(max): 0. 501056 Min-entropy: 0.996958	P(max): 0. 50114 Min-entropy: 0.996716	P(max): 0. 501368 Min-entropy: 0.99606	P(max): 0. 501767 Min-entropy: 0.994912	P(max): 0. 500907 Min-entropy: 0.997387
Min-entropy		0.913101	0.922601	0.925141	0.917851	0.907243

TABLE 7. Comparison with existing random number generators.

Entropy source	Estimated entropy	Sample Size	Entropy source	Reference
Wireless (LQI)	0.47	8 bits	Wireless (LQI)	Hennebert et al. [63]
Accelerometer X	0.22	9 bits	Accelerometer X	Hennebert et al. [64]
Accelerometer Y	0.42	9 bits	Accelerometer Y	
Accelerometer Z	0.36	9 bits	Accelerometer Z	
Vibration sensor	0.17	16 bits	Vibration sensor	
Magnetic sensor	0.62	16 bits	Magnetic sensor	
GTX 690	0.50	4 bits	GTX 690	Yoo et al. [65]
GTX 780	0.60	4 bits	GTX 780	
Only visible spectrum	0.72	1 bit	Only visible spectrum	Our results
Visible spectrum with XOR operation	0.93	1 bit	Visible spectrum with XOR operation	

spectrum. Therefore, in order to improve the entropy higher, we estimate processed data that the operation result after the XOR operation with the sample provided by NIST. The estimation results are shown in Table 6 and Fig. 8.

As a result, the entropy of the NIST sample is 0.901968, and the results of our three tests are 0.917851, 0.933911, and 0. 904603, respectively. This means that the entropy can be increased by performing the XOR operation based on the generated random number including the existing PRNG, rather than using the data of the visible spectrum independently. The increase rate of entropy was increased from 0.29% to 3.54% at maximum. Consequently, proposed TRNG can be expected to generate random numbers with high entropy.

Finally, we compare and evaluate proposed random number generator with existing random number generators, and the results are shown in Table 7 [63]–[65]. As a result, most existing generators have low entropy and the highest entropy is 0.62. However, the proposed generator has minimum entropy of 0.72 and a maximum of 0.93, which is increased from at least 86% to a maximum of 423%. This means that the proposed method is practically excellent.

IV. CONCLUSION

In this paper, we proposed a TRNG using the visible spectrum as a source. The visible spectrum contains a lot of information that can be used as a random number. The spectrum includes a lot of noises because the information is originated from electromagnetic waves. The collected visible spectrum information performs post-processing. Through this process, generated random numbers are increasing the distribution, reducing the correlation between each spectrum and increasing the randomness. We collected 1,000 spectrums for the experiment and estimated the entropy using tool provided by NIST. As a result of the estimation, the data collected from one test were 331,000, so the entropy was estimated based on insufficient data, which has low entropy. The reason is that the collected random number is not generated with high entropy having a certain probability but has low entropy because the data is randomly generated according to the visible spectrum. Therefore, in order to increase the entropy, we estimated based on the operation result after the XOR operation with the sample provided by NIST. As a result, three test results increased the entropy. This means that the entropy can be increased by performing the XOR operation based on the generated random number including the existing PRNG, rather than using the data of the visible spectrum independently. Consequently, proposed TRNG can be expected to generate random numbers with high entropy.

REFERENCES

[1] B. Cha, K. Kim, and H. Na, "Random password generation of OTP system using changed location and angle of fingerprint features," in *Proc. IEEE Int. Conf. Comput. Inf. Technol.*, Jul. 2008, pp. 420–425.
[2] NIST. *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf
[3] NIST. *Recommendation for the Entropy Sources Used for Random Bit Generation*. [Online]. Available: https://csrc.nist.gov/csrc/media/publications/sp/800-90b/draft/documents/sp800-90b_second_draft.pdf
[4] D. W. Davies and W. L. Price, *Security for Computer Networks: An Introduction to Data Security in Teleprocessing and Electronic Funds Transfer*. Hoboken, NJ, USA: Wiley, 1989.
[5] B. Jansson, *Random Number Generators*. Stockholm, Sweden: Almqvist & Wiksell, 1966.
[6] T. Ritter, "The efficient generation of cryptographic confusion sequences," *Cryptologia*, vol. 15, no. 2, pp. 81–131, Apr. 1991.
[7] M. Blum and S. Micali, "How to generate cryptographically strong sequences of pseudo-random bits," *SIAM J. Comput.*, vol. 13, no. 4, pp. 850–864, 1984.
[8] G. B. Agnew, "Random sources for cryptographic systems," in *Proc. Workshop Theory Appl. Cryptograph. Techn.*, 1987, pp. 77–81.
[9] D. Davis, R. Ihaka, and P. Fenstermacher, "Cryptographic randomness from air turbulence in disk drives," in *Proc. Adv. Cryptol.*, 1994, vol. 94. pp. 120–144.
[10] M. Jakobsson, E. Shriver, B. K. Hillyer, and A. Juels, "A practical secure physical random bit generator," in *Proc. ACM Conf. Comput. Commun. Security*, 1998, pp. 103–111.
[11] B. Jun and P. Kocher, "The intel random number generator," Cryptography Res., San Francisco, CA, USA, White Paper, 1999.
[12] C. S. Petrie and J. A. Connelly, "A noise-based IC random number generator for applications in cryptography," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 47, no. 5, pp. 615–621, May 2000.
[13] M. Bucci, L. Germani, R. Luzzi, A. Trifiletti, and M. Varanonuovo, "A high-speed oscillator-based truly random number source for cryptographic applications on a smart card IC," *IEEE Trans. Comput.*, vol. 52, no. 4, pp. 403–409, Apr. 2003.
[14] V. Bagini and M. Bucci, "A design of reliable true random number generator for cryptographic applications," in *Proc. Conf. Cryptograph. Hardw. Embedded Syst.*, 1999, p. 728.
[15] M. Dichtl and N. Janssen, "A high quality physical random number generator," in *Proc. Sophia Antipolis Forum Microelectron.*, 2000, pp. 48–53.
[16] U. Güler, S. Ergün, and G. Dündar, "A digital IC random number generator with logic gates only," in *Proc. IEEE Int. Conf. Electron., Circuits, Syst.*, Dec. 2010, pp. 239–242.
[17] B. Sunar, W. J. Martin, and D. R. Stinson, "A provably secure true random number generator with built-in tolerance to active attacks," *IEEE Trans. Comput.*, vol. 56, no. 1, pp. 109–119, Jan. 2007.
[18] M. Dichtl and J. D. Golic, "High speed true random number generation with logic gates only," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.*, 2007, pp. 45–62.
[19] K. Wold and S. Petrović, "Behavioral Model of TRNG Based on Oscillator Rings Implemented in FPGA," in *Proc. IEEE Int. Symp. Design Diagnostic Electron. Circuits Syst.*, Apr. 2011, pp. 163–166.
[20] M. J. Bellido, A. J. Acosta, M. Valencia, A. Barriga, and J. L. Huertas, "Simple binary random number generator," *Electron. Lett.*, vol. 28, no. 7, pp. 617–618, Mar. 1992.
[21] M. Epstein, L. Hars, R. Krasinski, M. Rosner, and H. Zheng, "Design and implementation of a true random number generator based on digital circuit artifacts," in *Proc. Cryptograph. Hardw. Embedded Syst.*, 2003, pp. 152–165.
[22] I. Vasyltsov, E. Hambardzumyan, Y.-S. Kim, and B. Karpinskyy, *Fast Digital TRNG Based on Metastable Ring Oscillator* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2008, pp. 164–180.
[23] V. B. Suresh and W. P. Burleson, "Entropy extraction in metastability-based TRNG," In *Proc. IEEE Int. Symp. Hardw.-Oriented Security Trust*, Jun. 2010, pp. 135–140.
[24] J. Qu and M. O'Neill, "Ultra-lightweight true random number generators," *Electron. Lett.*, vol. 46, no. 14, pp. 988–990, Jul. 2010.
[25] S. Callegari, R. Rovatti, and G. Setti, "Embeddable ADC-based true random number generator for cryptographic applications exploiting nonlinear signal processing and chaos," *IEEE Trans. Signal Process.*, vol. 53, no. 2, pp. 793–805, Feb. 2005.
[26] T. Stojanovski and L. Kocarev, "Chaos-based random number generators—Part I: Analysis [cryptography]," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 48, no. 3, pp. 288–291, 2001.
[27] S. Ergun and S. Ozog, "Truly random number generators based on a non-autonomous chaotic oscillator," *J. Int. J. Electron. Commun.*, vol. 61, no. 4, pp. 235–242, 2007.
[28] C. D. Motchenbacher and J. A. Connelly, *Low-Noise Electronic System Design*. Hoboken, NJ, USA: Wiley, 1993.
[29] P. R. Gray, P. Hurst, R. G. Meyer, and S. Lewis, *Analysis and Design of Analog Integrated Circuits*. Hoboken, NJ, USA: Wiley, 1977.
[30] T. Yamazaki and A. Uchida, "Performance of random number generators using noise-based superluminescent diode and chaos-based semiconductor lasers," *J. IEEE J. Sel. Topics Quantum Electron.*, vol. 19, no. 4, Apr. 2013, Art. no. 0600309.
[31] T.-K. Kuan, Y.-H. Chiang, and S.-I. Liu, "A 0.43 pJ/bit true random number generator," in *Proc. IEEE Asian Solid-State Circuits Conf.*, Nov. 2014, pp. 33–36.
[32] S. Yasuda, H. Satake, T. Tanamoto, R. Ohba, K. Uchida, and S. Fujita, "Physical random number generator based on MOS structure after soft breakdown," *IEEE J. Solid-State Circuits*, vol. 39, no. 8, pp. 1375–1377, Aug. 2004.
[33] J. Holleman, S. Bridges, B. P. Otis, and C. Diorio, "A 3 μW CMOS true random number generator with adaptive floating-gate offset cancellation," *IEEE J. Solid-State Circuits*, vol. 43, no. 5, pp. 1324–1336, May 2008.
[34] M. Peric, P. Milicevic, Z. Banjac, V. Orlic, and S. Milicevic, "High speed random number generator for section key generation in encryption devices," in *Proc. Telecommun. Forum*, 2013, pp. 117–120.
[35] C. De Roover and M. Steyaert, "A 500 mV 650 pW random number generator in 130 nm CMOS for a UWB localization system," in *Proc. ESSCIRC*, 2010, pp. 278–281.
[36] H. Zhun and C. Hongyi, "A truly random number generator based on thermal noise," in *Proc. Int. Conf. ASIC*, 2001, pp. 862–864.
[37] A. Khanmohammadi, R. Enne, M. Hofbauer, and H. Zimmermanna, "A monolithic silicon quantum random number generator based on measurement of photon detection time," *IEEE Photon. J.*, vol. 7, no. 5, Oct. 2015, Art. no. 7500113.

[38] S. Burri, D. Stucki, Y. Maruyama, C. Bruschini, E. Charbon, and F. Regazzoni, ''SPADs for quantum random number generators and beyond,'' in *Proc. IEEE Asia South Pacific Design Autom. Conf.*, Jan. 2014, pp. 788–794.
[39] B. Qi, Y. M. Chi, H.-K. Lo, and L. Qian, ''High-speed quantum random number generation by measuring phase noise of a single-mode laser,'' *Opt. Lett.*, vol. 35, no. 3, pp. 312–314, 2010.
[40] R. Li, ''A true random number generator algorithm from digital camera image noise for varying lighting conditions,'' in *Proc. IEEE SoutheastCon*, Apr. 2015, pp. 1–8.
[41] S. G. Tanyer, K. D. Atalay, and S. C. Inam, ''Goodness-of-fit and randomness tests for the sun's emissions true random number generator,'' in *Proc. Int. Conf. Math. Comput. Sci. Ind.*, 2014, pp. 216–218.
[42] C. Hennebert, H. Hossayni, and C. Lauradoux, ''Entropy harvesting from physical sensors,'' in *Proc. ACM Conf. Security Privacy Wireless Mobile Netw.*, 2013, pp. 149–154.
[43] W. Wei and H. Guo, ''Bias-free true random-number generator,'' *Opt. Lett.*, vol. 34, no. 12, pp. 1876–1878, 2009.
[44] S. Hirsch, M. W. Smale, and R. Devaney, *Differential Equations, Dynamical Systems, and an Introduction to Chaos*, 3rd ed. San Francisco, CA, USA: Academic, 2013.
[45] A. Lasota and M. C. Mackey, *Chaos, Fractals and Noise—Stochastic Aspects of Dynamics*, 2nd ed. Berlin, Germany: Springer, 1994.
[46] M. Delgado-Restituto and A. Rodriguez-Vazquez, ''Integrated chaos generators,'' *Proc. IEEE*, vol. 90, no. 5, pp. 747–767, May 2002.
[47] T. Addabbo, A. Fort, S. Rocchi, and V. Vignoli, ''Chaos based generation of true random bits,'' in *Intelligent Computing Based on Chaos*. Berlin, Germany: Springer, 2009, pp. 355–377.
[48] A. Rodriguez-Vazquez, J. Huertas, and L. Chua, ''Chaos in switched-capacitor circuit,'' *IEEE Trans. Circuits Syst.*, vol. CS-32, no. 10, pp. 1083–1085, Oct. 1985.
[49] A. Rodriguez-Vazquez, A. Rueda, B. Perez-Verdu, and J. L. Huertas, ''Chaos via a piecewise-linear switched-capacitor circuit,'' *Electron. Lett.*, vol. 23, no. 12, pp. 662–663, Jun. 1987.
[50] A. Rodriguez-Vazquez, J. L. Huertas, A. Rueda, B. Perez-Verdu, and L. O. Chua, ''Chaos from switched-capacitor circuits,'' in *Proc. IEEE*, vol. 75, no. 8, pp. 1090–1106, Aug. 1987.
[51] A. Rodriguez-Vazquez, M. Delgado, S. Espejo, and J. L. Huertas, ''Switched-capacitor broadband noise generator for CMOS VLSI,'' *Electron. Lett.*, vol. 27, no. 21, pp. 1913–1915, Oct. 1991.
[52] M. Delgado-Restituto, A. Rodriguez-Vasquez, S. Espejo, and J. L. Huertas, ''A chaotic switched-capacitor circuit for 1/f noise generation,'' *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl*, vol. 39, no. 4, pp. 325–328, Apr. 1992.
[53] A. Rodriguez-Vázquez, R. Domínguez-Castro, F. Medeiro, and M. Delgado-Restituto, ''High resolution CMOS current comparators: Design and applications to current-mode function generation,'' *Analog Integr. Circuits Signal Process.*, vol. 7, no. 2, pp. 149–165, 1995.
[54] M. Delgado-Restituto and A. Rodriguez-Vazquez, ''Mixed-signal map-configurable integrated chaos generator for chaotic communications,'' *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 48, no. 12, pp. 1462–1474, Dec. 2001.
[55] Y. Yoshizawa, H. Kimura, H. Inoue, K. Fujita, M. Toyama, and O. Miyatake, ''Physical random numbers generated by radioactivity,'' *J. Jpn. Soc. Comput. Statist.*, vol. 12, no. 1, pp. 67–81, 1999.
[56] T. Jennewein, U. Achleitner, G. Weihs, H. Weinfurter, and A. Zeilinger, ''A fast and compact quantum random number generator,'' *Rev. Sci. Instrum.*, vol. 71, no. 4, pp. 1675–1680, 2000.
[57] M. Wahl, M. Leiggen, M. Berlin, T. Rohlicke, H. J. Rahn, and O. Benson, ''An ultrafast quantum random number generator with provably bounded output bias based on photon arrival time measurements,'' *Appl. Phys. Lett.*, vol. 98, no. 17, 2011, Art. no. 171105.
[58] C. R. S. Williams, J. C. Salevan, X. Li, R. Roy, and T. E. Murphy, ''Fast physical random number generator using amplified spontaneous emission,'' *Opt. Exp.*, vol. 18, no. 23, pp. 23584–23597, 2010.
[59] S. Park, N. Joo, and M. Kang, ''Practically secure and efficient random bit generator using digital fingerprint image for the source of random,'' *J. Korea Inf. Process. Soc.*, vol. 10D, no. 3, pp. 541–546, 2003.
[60] X. Zhang, L. Qi, Z. Tang, and Y. Zhang, ''Portable true random number generator for personal encryption application based on smartphone camera,'' *Electron. Lett.*, vol. 50, no. 24, pp. 1841–1843, 2014.
[61] H. Kang and K. Lee, ''True random number generation method by using the Moire Fringe,'' In *J. Korea Internet Things Soc.*, vol. 2, no. 1, pp. 23–27, 2016.
[62] Wikipedia. *Visible Spectrum*. Accessed: Feb. 14, 2018. [Online]. Available: https://en.wikipedia.org/wiki/Visible_spectrum
[63] C. Hennebert, H. Hicham, and L. Cedric, ''The entropy of wireless statistics,'' in *Proc. Eur. Conf. Netw. Commun. (EuCNC)*, 2014, pp. 1–5.
[64] C. Hennebert, H. Hossayni, and C. Lauradoux, ''Entropy harvesting from physical sensors,'' in *Proc. ACM Conf. Security Privacy Wireless Mobile Netw.*, 2013, pp. 149–154.
[65] T. Yoo and Y. Yeom, ''Using GPU as hardware random number generator,'' in *Proc. GPU Technol. Conf.*, 2015, paper P5292. [Online]. Available: http://on-demand.gputechconf.com/gtc/2015/posters/GTC_2015_Developer_Algorithms_12_P5292_WEB.pdf

KYUNGROUL LEE received the B.S., M.S., and Ph.D. degrees from Soonchunhyang University, South Korea, in 2008, 2010, and 2015, respectively.

He has been with the R&BD Center for Security and Safety Industries, Soonchunhyang University, as a Research Professor since 2016, where he is currently with the Laboratory of Information Systems Security Assurance.

His research interests include security protocols, vulnerability analysis, hardware security, reverse engineering, platform security, and offensive security analysis. Related to these topics, he has involved in over twenty research projects and published over a hundred research papers. He has served as a committee chair of the international conferences and workshops and an Editorial Member of the IJITST.

SUN-YOUNG LEE received the B.S. and M.S. degrees from the Department of Computer Science, Pukyong National University, Busan, South Korea, in 1993 and 1995, respectively, and the Ph.D. degree from the Department of Communication and Information Engineering, The University of Tokyo, in 2001.

She is currently a Professor with the Department of Information Security Engineering, Soonchunhyang University. She is interested in contents security, cryptography, information theory, and information security.

CHANGHO SEO received the B.S., M.S., and Ph.D. degrees from the Department of Mathematics, Korea University, South Korea, in 1990, 1992, and 1996, respectively. His research interests include cryptography, information security, and system security.

KANGBIN YIM received the B.S., M.S., and Ph.D. degrees from the Department of Electronics Engineering, Ajou University, Suwon, South Korea, in 1992, 1994, and 2001, respectively.

He is currently a Professor with the Department of Information Security Engineering, Soonchunhyang University. His research interests include vulnerability assessment, code obfuscation, malware analysis, leakage prevention, secure platform architecture, and mobile security. Related to these topics, he has involved in over sixty research projects and published over a hundred research papers. Prof. Yim has served as an Executive Board Member of the Korea Institute of Information Security and Cryptology, Korean Society for Internet Information, and The Institute of Electronics Engineers of Korea. He also has served as a committee chair of the international conferences and workshops and the Guest Editor of the journals, such as JIT, MIS, JCPS, JISIS, and JoWUA.