



Information Network

A survey of random number generator: Approaches, tests, novel applications in block-chain and AI driven industrial networks

Haozhe Chai, Qianqian Pan*, and Jun Wu*

Graduate School of Information, Production and Systems, Waseda University, Fukuoka 808-0135, Japan

Received: 6 April 2025 / Revised: 25 September 2025 / Accepted: 28 October 2025 / Published online: 30 October 2025

Abstract True Random Number Generators (TRNGs) are essential components in industrial systems and security-critical applications, providing non-deterministic randomness based on physical phenomena such as electronic noise, quantum effects, and biological processes. Unlike Pseudo-Random Number Generators (PRNGs), TRNGs offer stronger unpredictability, making them crucial in areas such as industrial control systems (ICS), secure communications, and block-chain protocols. This survey provides a comprehensive review of TRNG technologies. It covers various types of TRNGs and their physical principles, traces their historical development from early hardware to modern implementations, and examines widely used statistical and visual analysis methods for evaluating randomness. We also discuss key challenges in TRNG development, including entropy source reliability, hardware limitations, and scalability for real-world deployment. Furthermore, we explore TRNG applications in block-chain systems, where they support tamper-resistant operations such as consensus, smart contracts, and device authentication. We also highlight the growing integration of TRNGs with machine learning techniques, both to improve randomness generation and to monitor and analyze TRNG output. Overall, this review aims to provide researchers and practitioners with a clear and structured understanding of TRNGs, emphasizing their importance in modern digital and industrial environments.

Keywords TRNG, Block-chain, Machine Learning, Internet of Things, Industry

Citation Chai H, Pan Q and Wu J. A survey of random number generator: Approaches, tests, novel applications in block-chain and AI driven industrial networks. Security and Safety 2025; 4: 2025015. <https://doi.org/10.1051/sands/2025015>

1 Introduction

In today's digital age, randomness plays a crucial role in a multitude of fields including computing, cryptography, simulations, and data security. Random Number Generators (RNGs) are crucial in computing. They are especially important in computer and network security, where they support tasks like encryption and key generation. An RNG is a technique for generating a sequence of unpredictable numbers or symbols. Random number generators (RNGs) are divided into two categories. The first category is pseudo-random number generators (PRNG), also known as software-based random number generators. PRNGs generate numbers using algorithms, which makes them predictable and repeatable. And second one is true random number generator (TRNG), which has physical randomness who generates random

* Corresponding authors (email: qianqian.pan@ieee.org (Qianqian Pan); email: junwu@aoni.waseda.jp (Jun Wu))

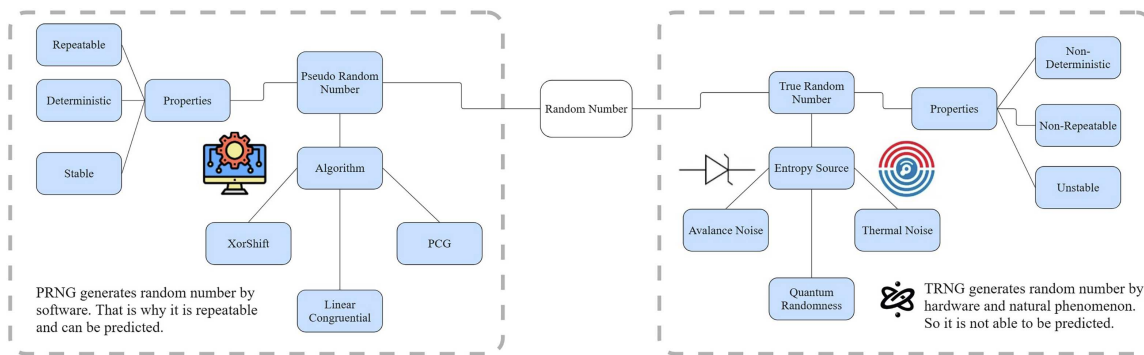


Figure 1. RNG categorization of properties and some examples

number by hardware and natural phenomenon. So, it is non-repeatable and not able to be predicted. With rising security demands, TRNGs have become a research hotspot. Their diversity and strong security features make them especially promising. PRNG is a deterministic system, which means that the generated random numbers are secure as long as the initial value has high entropy [1–15]. The categorization is shown in Figure 1.

However, it's important to note that generating truly random numbers in a deterministic system is not feasible. Consequently, there's a risk that an attacker might deduce the same key from the same random number by recreating the same initial value, like the seed. Moreover, generating entirely random numbers using solely mathematical algorithms are not possible. TRNGs provide high entropy and are suitable for applications where true randomness is critical, such as cryptography, security-sensitive systems, and simulations that require genuine unpredictability. However, TRNGs can be more resource-intensive, slower, and may require specialized hardware to capture physical randomness. They may also have limitations in terms of throughput and availability. The design of true random number generators presents a unique set of challenges as follows, encompassing hardware costs, physical noise stability, and output speed, which must be addressed to create effective TRNG systems [16].

- (1) One of the primary design challenges of TRNGs is managing hardware costs. True random number generation often involves specialized components and precise engineering to capture and amplify physical sources of randomness effectively. These components may include specialized sensors (*e.g.*, electronic noise or radioactive decay detectors), high-quality analogue-to-digital converters (ADCs), and dedicated processing units. As a result, TRNGs can be more expensive to produce than their pseudorandom counterparts. Finding cost-effective solutions to balance the need for true randomness with economic feasibility is a perpetual challenge in TRNG design.
- (2) The stability of physical noise sources presents another challenge. Physical processes, such as electronic noise, can be influenced by environmental factors, temperature fluctuations, and component ageing. Maintaining a consistent and reliable noise source is essential for TRNGs to provide high-quality randomness over time. Engineers must design TRNGs to mitigate these external influences and ensure the long-term stability of the noise source. Advanced error correction and compensation techniques may be required to enhance the robustness of TRNGs against noise source fluctuations [17–33].
- (3) TRNGs tend to generate random numbers at a slower pace compared to pseudo random number generators. This speed gap is a design challenge, especially in applications requiring high-throughput randomness. Striking a balance between true randomness and output speed is essential. Some TRNG designs involve parallelism and efficient algorithms to enhance the output speed without compromising the quality of generated random numbers. However, achieving both high speed and high entropy can be a delicate trade-off in TRNG design [34–45].

While True Random Number Generators (TRNGs) offer high entropy and strong security, their deployment in real-world systems—especially those with strict resource constraints—faces several critical engineering challenges, particularly in power consumption, chip-level integration, and process variability.

- (1) First, power consumption is a key concern, especially in battery-powered or energy-harvesting environments such as IoT nodes, embedded systems, and mobile devices. TRNGs that rely on continuous analog amplification or entropy extraction from high-frequency oscillators may consume non-negligible energy. To address this, recent designs explore duty-cycled entropy harvesting, low-leakage analog circuits, or event-triggered TRNGs that activate only when entropy is needed, thereby reducing power overhead [46].
- (2) Second, integration into chips presents both design and verification challenges. Analog-based entropy sources (*e.g.*, thermal noise, jitter) require sensitive layout and shielding in mixed-signal environments, where digital switching noise can easily interfere. Additionally, integrating TRNGs within modern SoCs demands compatibility with existing IP blocks and clocking domains. Designers must also ensure that the TRNG block passes post-silicon validation and is testable through built-in self-test (BIST) or health monitoring logic, without exposing the entropy source to potential side-channel attacks [47].
- (3) Third, fabrication process variability affects the stability and performance of physical entropy sources. Variations in transistor threshold voltage, channel length, doping concentration, and metal line resistance across wafers or production batches can cause shifts in noise amplitude, jitter behavior, or metastability resolution, directly impacting the entropy quality. This is particularly problematic in sub-10nm nodes, where quantum effects and leakage become more pronounced. To mitigate this, TRNG designers often include calibration logic, post-processing units (*e.g.*, von Neumann correctors, hash functions), and environmental compensation mechanisms to stabilize the output across process, voltage, and temperature (PVT) variations [48].

Collectively, these challenges highlight the importance of co-designing TRNG architecture with system-level constraints in mind. Future directions may involve the use of emerging materials, more efficient entropy extraction algorithms, or machine learning-based real-time stability analysis to ensure robust, scalable, and energy-efficient TRNG integration into diverse industrial platforms.

While several survey papers have given a general overview of the PRNG field [49], existing surveys on TRNGs tend to focus on specific entropy sources [50] or particular ways of implementing them [51]. While these earlier studies are valuable and look closely at certain aspects, they don't provide a full picture of TRNGs across different physical principles, structures, and real-world uses. This review aims to provide a more complete and joined-up view. This work is different. It reviews all types of TRNGs and their entropy sources—electronic noise, quantum effects, and biology. It also tracks their evolution from early prototypes to modern hardware. We compare TRNGs with PRNGs in terms of design, randomness, and security. It also looks at new ways TRNGs are being used, like in block-chain systems (*e.g.*, Proof-of-stake (PoS) protocols and zero-knowledge proofs) and how they can be used with machine learning.

The review explains things in a way that is easy to understand, and it helps readers get to know TRNGs. It also helps readers learn more about other, more advanced research topics. Its overall approach makes it a useful reference for researchers at all levels. It can be used by engineers and industry professionals who want to use TRNGs in new security scenarios. This review fills a gap in literature. It also lays a solid foundation for future innovation in TRNG theory and application. This paper aims to provide a comprehensive review of the development, current challenges, and emerging research directions of true random number generators. We first revisit the types and evolution of TRNGs, covering their physical principles and implementation methods. Subsequently, we discuss the visualization techniques used in the performance testing and analysis of TRNGs, which are crucial for ensuring the randomness and security of the generated values.

Beyond security and cryptography, TRNGs are gaining significant traction in industrial applications, particularly in sectors where reliable and unpredictable randomness is crucial. In industrial automation and control systems, TRNGs help secure communication channels and prevent adversarial attacks on sensitive data streams. In industrial Internet of Things (IIoT) networks, TRNGs are used to generate cryptographic keys for device authentication, ensuring robust protection against cyber threats. Furthermore, smart manufacturing and supply chain security leverage TRNGs to enhance anti-counterfeiting measures and improve the security of industrial-grade embedded systems. As Industry 4.0 continues to evolve, TRNGs are expected to play a vital role in enabling secure edge computing, encrypted sensor data transmission, and protected industrial AI models, ensuring a trustworthy and tamper-resistant environment for intelligent production systems. After the foundational research on TRNGs, we discussed the

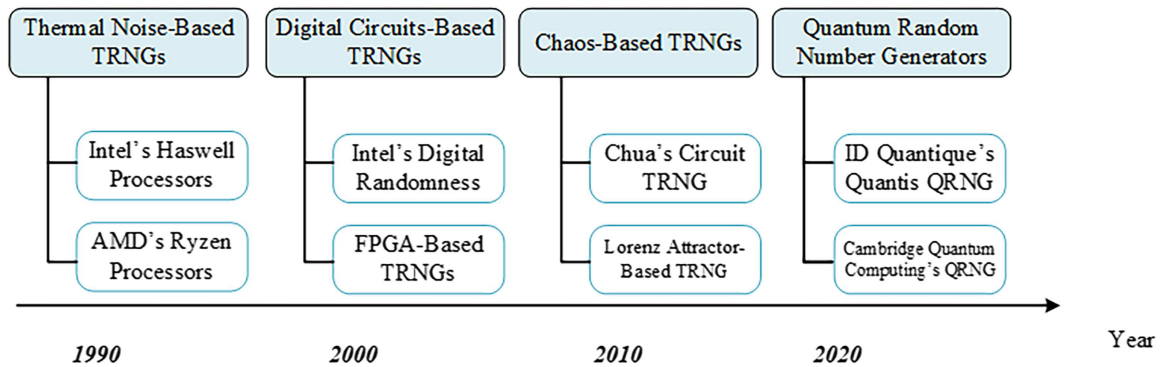


Figure 2. Types of TRNG with Time Goes

application of true random numbers in block-chain technology. With its decentralized and immutable characteristics, block-chain technology represents a revolutionary advancement in modern technology, where true random numbers play a leading role, especially in the execution of smart contracts and the security of cryptocurrencies. We will discuss the feasibility of integrating TRNGs with block-chain technology and how this integration can enhance the security and efficiency of block-chain platforms.

In addition, we explore the potential integration of TRNGs with machine learning technologies. As a powerful tool for data analysis, machine learning shows vast prospects in optimizing and automating decision processes. TRNGs can improve the robustness of machine learning models during training. Conversely, ML can help monitor and optimize TRNG output, improving randomness assessment. By reviewing these cross-disciplinary applications, we tend to provide a comprehensive perspective on how true random number generators support and advance modern technological development. We aim to provide valuable insights to researchers and developers, and to promote TRNG research and applications.

2 Overview of TRNG technologies

The types of TRNG with time are shown in Figure 2.

2.1 Entropy source

In the realm of TRNG, there exists a diverse array of entropy sources that underpin their operation. These sources harness the inherent unpredictability of physical phenomena to generate genuine randomness. This section provides an in-depth exploration of four distinctive categories of TRNGs, each relying on a different entropy source.

2.2 TRNG with electric-based entropy source

Electric-based TRNGs, unlike their quantum counterparts, draw their entropy from electrical or electronic processes. This category can be further subdivided into two primary types:

2.2.1 Latch-based TRNG

Latch-based TRNGs harness the intriguing property of metastability in digital flip-flops to capture random transitions between voltage levels. The slight timing variations during the transition process introduce entropy into the TRNG. By carefully designing the circuit, latch-based TRNGs can produce random numbers efficiently. They are known for their simplicity, low power consumption, and the ability to be implemented in a wide range of digital systems. This versatility makes them suitable for applications where hardware constraints and energy efficiency are significant considerations.

In X. Wang's thesis, the paper introduces a latch-based TRNG designed to achieve high raw entropy generation (>0.9) over a broad range of voltage and temperature conditions [52]. The TRNG, implemented

in a 130 nm CMOS technology, demonstrates compact size and low energy consumption (0.116 pJ/bit at 0.3 V), inclusive of an on-chip Von Neumann post-processing circuit. The cryptographic security of the generated randomness is verified through NIST SP 800-22 and 800-90B tests. The TRNG's robustness is further confirmed by an accelerated aging test, indicating an equivalent 20-year lifespan.

2.2.2 Sensor-based TRNG

Sensor-based TRNGs employ a wide array of physical sensors, such as photodetectors or radiation detectors, to capture environmental noise or quantum phenomena [20]. These sensors convert the detected physical fluctuations into random bit sequences. Sensor-based TRNGs offer versatility, as they can be adapted to various entropy sources. They are frequently used in applications where specific environmental conditions provide a rich source of entropy. For example, in a radiation sensor-based TRNG, variations in radiation levels can be transformed into random numbers. This adaptability allows sensor-based TRNGs to be tailored to the requirements of specific use cases, such as IoT devices that may operate in diverse environmental conditions.

In A. Degada's thesis [53], the paper introduces a True Random Number Generator (TRNG) prototype utilizing the inherent uncertainty in photoresistor sensors. The proposed prototype stands out by eliminating the need for complex signal preprocessing, leading to reduced delays and lowered hardware costs. A distinctive feature is the adoption of additive scrambling, enabling high-rate sensor sampling. The TRNG prototype achieves an impressive average random bit generation rate of 8 kbps, surpassing recent literature. The contribution is substantiated by the validation of randomness quality through 15 NIST STS test batteries.

2.3 TRNG with photon-based entropy source

Photon-based TRNGs represent the cutting edge of randomness generation. They leverage the fundamental properties of photons, particles of light, to introduce entropy into the generation process. These devices employ quantum principles to exploit the fundamental indeterminacy in the quantum world. Components such as beam splitters and photodetectors are used to capture the quantum properties of photons. By measuring characteristics like the polarization or arrival time of individual photons, these TRNGs create random sequences. Quantum phenomena, such as the uncertainty principle and the no-cloning theorem, ensure the unpredictability of the measurements. Photon-based TRNGs are especially prized in cryptographic applications due to the strong foundations of quantum physics in ensuring the integrity of the generated randomness.

In M. J. Applegate's thesis, the paper introduces an efficient and robust quantum random number generator based on high-rate room temperature photon number detection [54]. The authors propose a novel approach utilizing an electric field-modulated silicon avalanche photodiode designed for high-rate photon number detection with exceptional resolution. The device detects up to 4 photons from optical pulses emitted by a laser without applying a dead-time constraint. Through measurement and modeling, optimal illumination conditions are identified, exhibiting robustness against variations in photon flux. The authors achieve a 99 percent efficiency in extracting random bits from detected photon numbers, resulting in 1.97 bits per detected photon and a bit rate of 143 Mbit/s. The extracted bits successfully pass stringent statistical tests for randomness, and the proposed scheme demonstrates scalability with the potential for multi-Gbit/s bit rates.

2.4 TRNG with thermal-based entropy source

Thermal-based TRNGs, in contrast, delve into the world of thermal and electronic noise in semiconductor materials. These TRNGs often feature diodes or resistors as noise sources. The thermal fluctuations in the materials result in random voltage or current variations, which are then digitized to create random sequences. While not as esoteric as quantum-based TRNGs, thermal-based TRNGs have gained recognition for their simplicity and cost-effectiveness. They find widespread use in scenarios where hardware constraints or budget considerations limit the use of more complex entropy sources.

In Y. Yao's thesis, the paper introduces a novel contribution in the form of a True Random Number Generator (TRNG) that utilizes continuous skyrmion thermal Brownian motion in a confined geometry at room temperature [55]. The proposed TRNG achieves the generation of a random bitstream, with an approximately equal probability of ~ 50 percent for both bits "0" and "1." Notably, this is achieved through the periodic detection of the relative position of the skyrmion without the necessity for any additional activations.

2.5 TRNG with quantum-based entropy source

Quantum-based TRNGs represent the frontier of random number generation, leveraging the fundamental indeterminacy of quantum mechanics to produce genuine, irreducible randomness. Unlike classical noise-based TRNGs that are often susceptible to environmental interference and hardware instability, quantum-based TRNGs derive entropy from intrinsic quantum phenomena, which are inherently unpredictable and theoretically uncorrelated with any classical variable. This makes quantum entropy sources ideal for cryptographic and security-critical applications. Quantum can also be further subdivided as an entropy source, such as the OQRNG proposed by M. Lavich, which provides a faster and more stable entropy source based on optics.

The two most widely utilized quantum phenomena in TRNG design are: Quantum photon behavior, which includes photon arrival time, phase, or polarization. And Quantum tunneling effects, which are as observed in electron transitions in tunnel diodes or other nanoscale structures. One of the most common implementations involves single-photon detection. In such systems, a laser emits photons that pass through a beam splitter. Detectors placed at the outputs of the splitter register the photon's path. Due to the inherent uncertainty in quantum measurement, the photon's path (and thus the detector that clicks) is fundamentally random. This binary outcome can be directly mapped to a random bit (*e.g.*, detector $A = 0$, detector $B = 1$). This approach guarantees randomness at the quantum level without reliance on classical post-processing.

Another class of quantum entropy source exploits vacuum fluctuations, where quantum noise is amplified through homodyne detection or phase noise analysis. These systems can achieve extremely high throughput, with recent implementations (*e.g.*, ID Quantique, Toshiba) reaching speeds of 1 Gbps or higher, while remaining provably secure under quantum mechanical assumptions.

In K. Witek's thesis, the paper outlines the development of a Quantum-True Random Number Generation (QTRNG) platform by Random Power [56]. The platform generates unpredictable bit streams by analyzing the time series of self-amplified endogenous pulses resulting from stochastically generated charge carriers in a dedicated silicon device. To enhance the quality of randomness and address potential biases, the authors incorporate a SHA256 conditioning function, following NIST recommendations. The primary contribution lies in the creation of a robust QTRNG platform that ensures the required level of entropy for secure random number generation.

Each of these TRNG categories offers distinct advantages and trade-offs, making them suitable for various use cases. Table 1 below shows different entropy sources of TRNG and their key characteristics, advantages, limitations, and typical applications. The choice of a particular TRNG depends on the specific requirements of the application, ranging from high-security cryptographic systems to resource-constrained IoT devices. This rich tapestry of TRNG options underscores the need for a deeper exploration of design choices and challenges in the creation of an ideal TRNG, as discussed in the subsequent sections of this paper.

3 Statistical test for true random number generators

Evaluating TRNGs is crucial in fields like cryptography, simulations, and stochastic modeling, where high-quality randomness is essential. TRNGs generate numbers using inherently unpredictable physical processes, but their outputs still need thorough testing to ensure they meet the desired randomness properties. Statistical tests for TRNGs play a vital role in verifying the absence of patterns, biases, and correlations in the generated sequences. Basic methods for analyzing TRNG outputs include: Frequency Analysis, Autocorrelation, Entropy Measurement, Distribution Tests [57–67]. Let us have a view at the randomness tests of TRNG with time goes as Figure 3.

Here are a few random number tests that have been widely used over the years.

Table 1. Different Entropy Sources of TRNG

Entropy source	Key characteristics	Advantages	Limitations	Typical applications
Electric-based	Using electrical noise	Easy to integrate; low cost; compatible with CMOS	Susceptible to EMI, aging; lower entropy if not conditioned	Embedded systems, smart cards, general-purpose chips
Photon-based	Based on photon arrival time, path, polarization, etc.	High entropy; inherently random; supports high bit rates	Requires optical components; bulky or expensive	Quantum key distribution (QKD), high-security modules
Thermal-based	Exploits thermal noise (Johnson-Nyquist) in resistors or diodes	Simple hardware; widely available	Entropy rate is low; may need amplification and post-processing	Low-power IoT devices, analog sensors
Quantum-based	Leverages quantum phenomena	Provably random; highest entropy; strong against modeling	Complex; costly; sensitive to implementation quality	Cryptography, blockchain, secure industrial networks

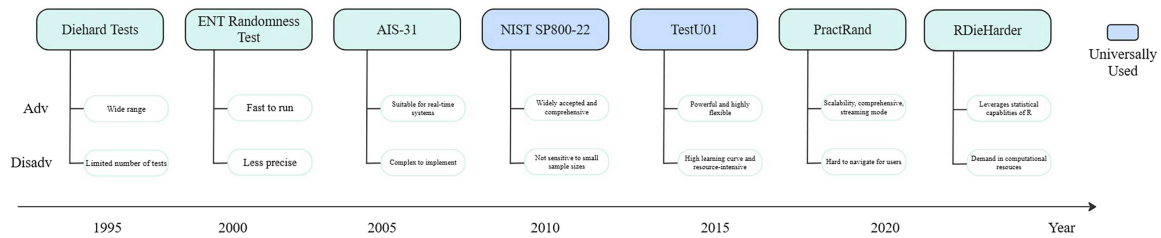


Figure 3. Randomness Tests of TRNG with Time Goes

3.1 Detailed introduction of randomness tests

3.1.1 Diehard tests

Diehard tests was created in 1995. Developed by George Marsaglia, the Diehard tests are among the earliest and most widely recognized test suites for RNGs. It includes tests like the Birthday Spacings test, Overlapping Permutations test, Ranks of 31×31 and 32×32 matrices, and more.

Diehard Tests pioneered the field of randomness testing. It covers a variety of non-randomness characteristics, widely accepted in both academic and applied settings. However, some tests may not detect newer forms of non-randomness, and it can be computationally demanding for large data sets. What's more, some tests require a deep understanding of statistical concepts to interpret results effectively.

The Diehard test suite comprises a diverse set of statistical tests designed to examine different aspects of randomness. Notable examples include the Birthday Spacings test, which checks for uniform spacing among numbers in a sequence; the Overlapping Permutations test, which evaluates the frequency of specific permutations in subsequences; and the Ranks of Matrices test, which analyzes the rank of randomly generated binary matrices. Other tests include the Monkey Test, which counts the occurrences of overlapping m-grams (*e.g.*, simulated word patterns); the Count-the-1's test, which measures the number of 1s in designated byte positions; the Parking Lot Test, which simulates randomly parking cars in a limited area to assess space usage; and the Minimum Distance Test, which calculates the distances between randomly placed points within a unit square. Together, these tests provide a comprehensive assessment of the statistical properties of random number sequences.

3.1.2 ENT randomness test

ENT randomness test was created in 1998. ENT provides a quick and comprehensive evaluation of random sequences using simple statistical tests. It has simple implementation and quick execution, providing essential randomness metrics and suitable for rapid assessment of randomness quality. But ENT is less comprehensive compared to more recent suites, may not detect subtle non-random patterns. And the results are straightforward but might miss complex forms of non-randomness.

The ENT test suite includes several simple, yet insightful statistical analyses aimed at evaluating the quality of random sequences. Key components include Entropy, which measures the unpredictability of a sequence, and the Chi-Square Test, which evaluates the uniformity of value distribution. The Arithmetic Mean test checks whether the average of the sequence aligns with theoretical expectations, while the Monte Carlo pi Value estimates π by simulating random sampling in a geometric space. Additionally, Serial Correlation is used to determine the degree of dependence between successive values, and the Compression Test evaluates how compressible the data is—since truly random data should be largely incompressible. Together, these tests offer a fast and lightweight snapshot of randomness characteristics.

3.1.3 AIS-31

AIS-31 was created in 2001. Developed by the German Federal Office for Information Security (BSI), AIS-31 is tailored for evaluating both deterministic and non-deterministic RNGs in cryptographic applications. AIS-31 is comprehensive and rigorous, making it particularly suitable for cryptographic applications, and includes online tests for continuous performance evaluation. However, it is complex to implement, especially for online tests, tailored to certain standards limiting broader applicability, and resource-intensive, requiring significant computational power and time for thorough testing.

The AIS-31 test suite encompasses a range of statistical evaluations designed to rigorously assess the quality of both deterministic and non-deterministic RNGs. Among its core components are the Entropy Test, which validates the quality of the entropy source, and the Chi-Square Test, which examines the uniformity of output distribution. The suite also includes the Poker Test, which analyzes the frequency of specific patterns, and the Runs Test, used to measure the number and length of consecutive identical bits. The Autocorrelation Test further assesses dependency across bit positions, while the Adaptive Proportion Test adjusts statistical thresholds based on sample size to ensure reliable assessments. Finally, the Longest Run Test evaluates the maximum length of consecutive occurrences of the same digit within a sequence. These tests collectively offer a robust framework for identifying both subtle and overt deviations from true randomness.

3.1.4 NIST SP800-22

NIST SP800-22 was created in 2001. Developed by the National Institute of Standards and Technology (NIST), SP800-22 is a widely used suite for evaluating RNGs, especially in cryptographic contexts. NIST SP800-22 is standardized and widely accepted, providing a high level of trust and covering a broad range of statistical tests. It is computationally intensive, especially for large data sets, with some tests not being effective on small sample sizes, and requires significant statistical knowledge to implement and interpret results.

The NIST SP800-22 test suite consists of a comprehensive collection of statistical tests aimed at rigorously evaluating the randomness of binary sequences, particularly in cryptographic applications. Key components include the Frequency Test, which verifies whether 0s and 1s appear with equal likelihood, and the Block Frequency Test, which assesses bit distribution within subdivided blocks. The Runs Test and Longest Run of Ones in a Block Test examine the occurrence and length of consecutive identical bits, while the Rank Test evaluates the linear dependence in matrices constructed from the sequence. More advanced tests such as the FFT Test apply Fourier analysis to uncover periodic features, and the Non-overlapping and Overlapping Template Matching Tests scan for predefined patterns with or without overlapping. Additionally, the Universal Test detects repetition distances to gauge unpredictability, and the Approximate Entropy Test measures the frequency of all possible overlapping patterns of a certain length. Together, these tests form a robust and widely recognized framework for assessing statistical randomness.

3.1.5 *TestU01*

TestU01 was created in 2007. Developed by Pierre L'Ecuyer and Richard Simard, TestU01 is a comprehensive library for testing RNGs, including extensive batteries like SmallCrush, Crush, and BigCrush. TestU01 is highly comprehensive and suitable for both research and practical applications, with advanced tests for deep analysis and versatility for a wide range of RNGs. However, it has a high learning curve, requiring significant statistical expertise, is resource-intensive demanding considerable computational power, and its detailed setup and interpretation can be challenging.

The TestU01 library offers one of the most extensive and rigorous frameworks for testing random number generators, incorporating multiple test batteries such as SmallCrush, Crush, and BigCrush. Among its core tests are the Birthday Spacings Test, which identifies spacing anomalies similar to Diehard, and the Collision Test, which tracks repeated values in generated sequences. The Gap Test analyzes the intervals between repeated occurrences, while the Permutation Test checks for uniform distribution of value arrangements within subsequences. The suite also includes tests for deeper structural properties, such as Linear Complexity, which assesses how complex the sequence is to predict, and the Matrix Rank Test, which evaluates the rank of matrices composed from bit sequences. Additionally, the Multinomial Test expands upon the traditional chi-square approach to accommodate multi-category outcomes, and the Spectral Test investigates the sequence's frequency components. These robust tools make TestU01 particularly valuable for both academic research and industrial-grade RNG evaluation, despite its steep learning curve and computational intensity.

3.1.6 *PractRand*

PractRand was created in 2012. PractRand (Practical Random Number Generator Test Suite) is designed to handle very large data sets and detect subtle patterns and correlations. PractRand is scalable for very large data sets and provides a wide range of tests covering various types of non-randomness, with the ability to process data in real-time. However, it is complex to use with a broad range of tests and options that can be difficult to navigate, demanding significant computational resources for large data sets, and requires deep statistical knowledge to understand and interpret results.

PractRand (Practical Random Number Generator Test Suite) was introduced in 2012 and is tailored for handling extremely large data sets while detecting subtle statistical anomalies and patterns. Its key tests include BRank, which examines the rank of binary matrices, and GCD, which analyzes the greatest common divisor properties in number sequences. The suite also employs FPMulti to detect potential biases via floating-point operations and DistC6 to evaluate distribution behavior across six dimensions. Other specialized tests include BCFN (Bias-Corrected Frequency-N), which enhances frequency testing accuracy, and DC6-9x1Bytes, designed to reveal bias in sequences of bytes. Lastly, Mod3 investigates how values behave under modulo 3 operations. This broad and nuanced set of tools enables PractRand to assess randomness with remarkable sensitivity and scale, although the suite's complexity, high computational demand, and steep interpretive requirements pose challenges for general use.

3.1.7 *RDieHarder*

RDieHarder was created in 2018 as an R interface for the Dieharder suite, providing accessibility to users who are familiar with the R programming environment. By integrating with R, it leverages R's robust statistical capabilities, making it a convenient tool for researchers and practitioners who prefer R for data analysis. RDieHarder inherits a comprehensive array of statistical tests from the original Dieharder suite, including the classic Diehard tests, GNU Scientific Library (GSL) tests, and additional advanced randomness assessments. This combination offers extensive functionality and flexibility, especially for exploratory statistical testing and automation within R workflows. However, like its predecessor, RDieHarder can be computationally demanding and assumes a working knowledge of R and its statistical tools, which may pose challenges for users unfamiliar with the environment.

3.1.8 *Industrial standardization of random number*

In industrial environments, especially in security-critical systems such as encrypted communication modules, industrial control systems (ICS), and IoT devices, the generation of high-quality random numbers

must meet strict standardization requirements. Several well-established standards have been introduced to ensure the reliability, unpredictability, and compliance of both pseudo and true random number generators (PRNGs and TRNGs).

One of the most widely used frameworks is NIST SP 800-22, which offers a suite of 15 statistical tests for evaluating the randomness of binary sequences. It is complemented by the SP 800-90 series, which defines entropy source modeling and requirements for deterministic and non-deterministic RNGs. Together, they form the basis for FIPS 140-3, the U.S. federal standard for cryptographic module certification, which is extensively adopted in industrial-grade security devices.

In Europe, AIS-31, released by Germany's BSI, plays a similar role by providing evaluation methods for TRNGs, including continuous online testing and classification profiles. It is commonly used in secure chips, smart cards, and embedded authentication modules. On a global scale, ISO/IEC 18031 defines principles and procedures for random number generation, supporting both TRNGs and PRNGs. It promotes interoperability in international industrial systems and aligns with widely accepted cryptographic practices.

Overall, these standards guide the design and validation of RNGs in industrial contexts, ensuring devices meet regulatory, safety, and performance expectations across diverse application domains.

3.2 Summary

Evaluating the output of TRNGs is essential to ensure their reliability and suitability for critical applications. Table 2 below shows the universally used test suites and their characteristics, advantages and limitations. Various statistical test suites have been developed over the years, each with their unique strengths and weaknesses. Early test suites like Diehard and ENT laid the groundwork, providing essential tools for randomness assessment. AIS-31 and NIST SP800-22 introduced more rigorous and standardized approaches, especially for cryptographic applications. TestU01 expanded the scope with comprehensive and advanced tests, while newer suites like PractRand and RDieHarder offer scalability and integration with modern statistical environments.

Selecting the appropriate test suite depends on the specific requirements of the application, the size of the data set, and the available computational resources. Combining multiple test suites can provide a more robust and thorough evaluation of TRNG outputs, ensuring the highest standards of randomness and security.

4 Result and visualizing analysis

The criteria for evaluating the TRNG output can be categorized as follows: independence, distribution, periodicity, randomness, trend detection, *etc.* [68–75].

4.1 Independence

Used to check whether there is a dependency between data points, *i.e.* whether the random numbers are generated independently of each other.

4.1.1 Scatter plot

A scatter plot is a graph that displays individual data points in two dimensions, plotting one variable on the x -axis and another on the y -axis. A scatter plot of successive TRNG values can reveal whether the values are independent and uniformly distributed. If the points are randomly distributed without any discernible patterns, the TRNG output is likely random. Patterns, clusters, or gaps could indicate dependencies or biases in the data.

Scatter plots provide a quick visual check for correlations between successive values. However, they can be limited when visualizing subtle higher-order dependencies or patterns.

Table 2. Test Suites Which are Universally Used

Test Suite	Year Introduced	Key Characteristics	Advantages	Limitations
Diehard tests	1995	One of the earliest RNG test suites; includes 15 classic tests	Historically influential; simple to run; still widely referenced	Limited coverage; outdated for modern cryptographic needs
ENT	1998	Provides fast statistical tests: entropy, chi-square, mean, <i>etc</i>	Lightweight and easy to use; quick overview of randomness	Shallow coverage; cannot detect complex or subtle flaws
AIS-31	2001	German BSI standard for TRNG and DRNG evaluation	Strong in continuous (online) testing; cryptographically oriented	Implementation complexity; test focus narrower than others
NIST SP800-22	2001	U.S. government standard; 15 statistical tests for binary data	Well-documented, standardized, widely accepted in industry	Requires long sequences; some tests lack statistical robustness
TestU01	2007	Large-scale empirical library; includes Crush and BigCrush	Very comprehensive; advanced statistical rigor	High learning curve; computationally intensive
PractRand	2012	Adaptive, real-time testing; suitable for large data streams	Scalable; detects subtle patterns; good for long sequences	Complex to interpret; fewer formal documents
RDieharder	2018	R interface for Dieharder + GSL tests; user-friendly for R users	Combines many tests; integrates with statistical analysis tools	Depends on R knowledge; inherits Dieharder's limits

4.1.2 Autocorrelation plot

Autocorrelation measures how a value in a sequence is related to previous values in the same sequence. The plot shows autocorrelation values for different time lags. Ideally, a TRNG sequence should show no significant autocorrelations at any lag, meaning each value is independent of previous ones. Peaks or periodic behavior in the plot may indicate correlations, periodicities, or weaknesses in the generator.

Autocorrelation plots are highly effective for detecting hidden patterns and periodicities, making them particularly useful for evaluating independence between values. However, it is computationally expensive for large datasets.

We generated a stream of random numbers using my own TRNG and used them to draw an autocorrelation plot as Figure 4. The autocorrelation plot shows how each bit in the sequence correlates with its lagged values. The x -axis represents the lag (distance between bits), and the y -axis shows the correlation coefficient. For a high-quality TRNG, all values except lag 0 should be close to zero. High peaks suggest periodicity or predictability, which reduces randomness.

4.2 Distribution

Used to verify that the random numbers generated conform to a desired probability distribution (*e.g.*, uniform distribution).

4.2.1 Histogram

A histogram is a graphical representation of the distribution of numerical data, dividing the data into bins and showing the frequency of values within each bin. For a uniformly random sequence, the histogram should show an approximately flat distribution, meaning each bin should contain a similar number of

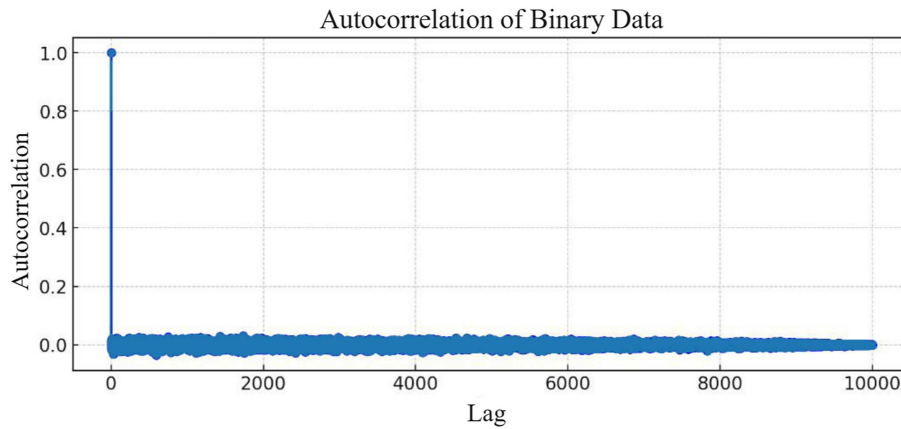


Figure 4. Auto-correlation Plot

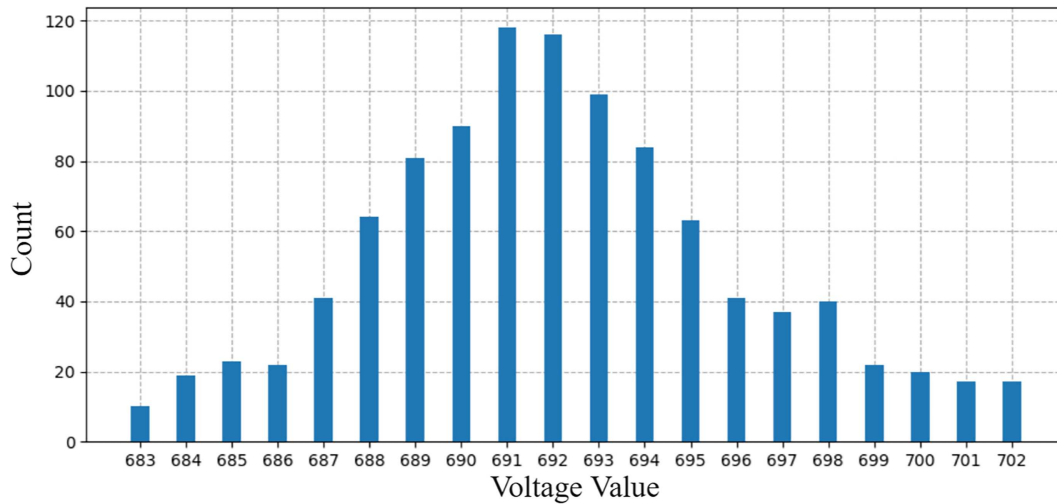


Figure 5. Histogram of Voltage Value of Sensors

values. Deviations from this indicate that certain values are being favored more than others, which can suggest non-random behavior.

Histograms are effective for detecting non-uniform distributions, but they are less informative about correlations or temporal dependencies in the sequence. In the following histogram Figure 5, we tested the randomness of LDR sensor. We collected 1000 data whose voltage values exceeding 659 among 679 via Arduino Uno under normal indoor lighting conditions. The collected data presents an ideal normal distribution, proving that the LDR sensor can be used as a good entropy source for the design of TRNG.

4.2.2 Q-Q plot

A Q-Q plot compares the quantiles of a dataset with the quantiles of a theoretical distribution (*e.g.*, a uniform or normal distribution). In TRNG evaluation, a Q-Q plot can be used to compare the distribution of the generated values with a uniform distribution. If the points closely follow the reference line, the data matches the expected distribution. Deviations from the line suggest skewness or biases in the TRNG output.

Q-Q plots are particularly good at detecting deviations from the expected distribution, especially in the tails, but they do not capture correlations or dependencies between values.

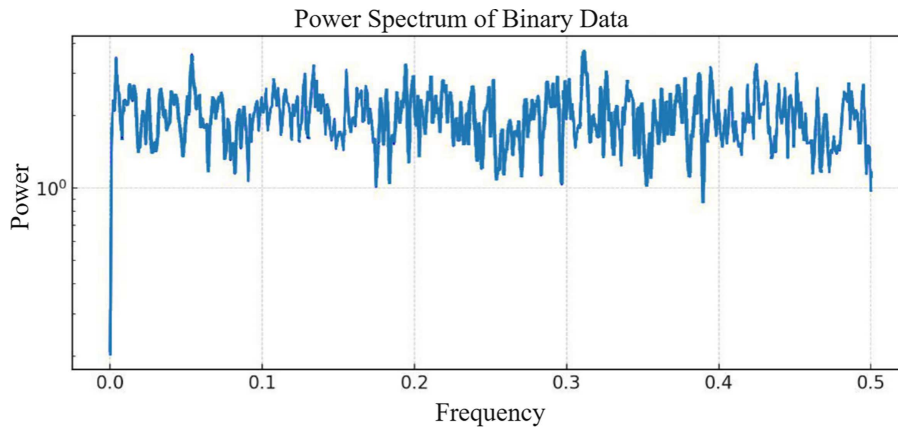


Figure 6. Frequency Spectrum

4.3 Periodicity

Used to detect the presence of repetitive patterns or periodic behavior.

4.3.1 Power spectrum/frequency spectrum

The power spectrum displays how the power of a signal is distributed across different frequency components. It is often calculated using a Fourier transform. For a truly random sequence, the power spectrum should be flat, indicating that no frequency dominates. Peaks in the spectrum suggest periodicities or repeat patterns within the data, which is a sign of non-randomness. The frequency spectrum is useful for detecting periodic components that are not easily visible in time-domain plots. However, it is less intuitive to interpret compared to other visualizations.

As in Figure 6, the x -axis denotes frequency, and the y -axis indicates amplitude. A truly random sequence should produce a flat or noisy spectrum without sharp peaks. Peaks at specific frequencies suggest patterns or cycles, which weaken the security of the TRNG.

4.4 Randomness

Used to check whether the sequence is characterized by randomness, *i.e.*, there is no predictable pattern.

4.4.1 Heatmap

A heatmap visualizes data in a matrix format, using colors to represent different values or frequencies of occurrence. In TRNG analysis, a heatmap can be used to show the joint distribution of pairs of successive values. If the values are truly random, the heatmap should exhibit no discernible patterns. Any concentration of color in certain areas can indicate non-randomness or dependencies between values.

Heatmaps are effective for visualizing relationships between pairs of values or the joint distribution of multiple variables. However, they can be difficult to interpret without a clear color scale or if the dataset is too large. The following Figure 7 is a heatmap we drew using random number streams generated by my own TRNG. The heatmap shows the transition frequency between consecutive bits. The x -axis represents the current bit, and the y -axis represents the next bit. Color intensity indicates how often each transition occurs. Ideally, all four transitions ($0 \rightarrow 0$, $0 \rightarrow 1$, $1 \rightarrow 0$, $1 \rightarrow 1$) should have similar frequencies. Uneven color distribution implies correlation or bias between bits, which may affect randomness quality.

4.4.2 Time series plot

A time series plot displays data points in the order in which they were generated, showing how the values change over time. A time series plot of TRNG output should look completely random, with no trends,

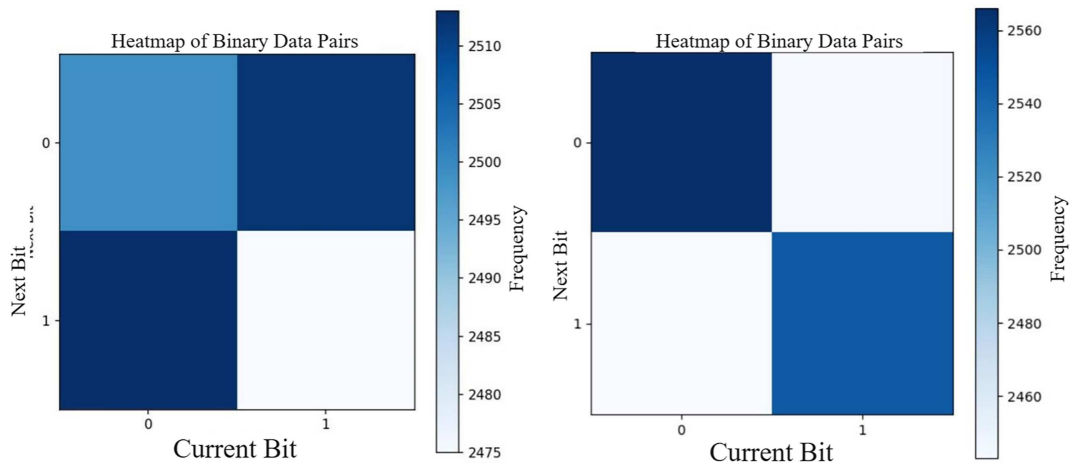


Figure 7. Heatmap of Output of TRNG and PRNG

cycles, or repeat patterns. If trends or periodicity are observed, this could suggest a bias or weakness in the TRNG.

Time series plots are useful for identifying trends, drifts, or long-term dependencies. However, they are less effective for large datasets, where random fluctuations can make visual interpretation difficult.

4.5 Trend detection

Used to check for long term trends or shifts in the random numbers generated.

4.5.1 Box-and-whisker plot

A box plot shows the distribution of a dataset based on its quartiles, highlighting the median, interquartile range, and potential outliers. A box plot of TRNG output should ideally show a symmetric distribution with few or no outliers. If the distribution is skewed or contains significant outliers, it could indicate non-randomness.

Box plots provide a concise summary of the central tendency, dispersion, and outliers. They are good for detecting anomalies but are less informative about the internal structure of the data or higher-order dependencies.

4.5.2 Cumulative distribution function plot

A CDF plot shows the cumulative probability of a dataset, mapping each value to its corresponding cumulative probability. When analyzing TRNG output, the CDF should closely follow the CDF of the expected distribution (usually uniform). Deviations indicate that the data is not following the expected random distribution.

CDF plots are useful for detecting global deviations in distribution, but they provide less information about local variations or dependencies between data points.

4.6 Summary and analysis

Visualizing TRNG output is an essential step in randomness testing, complementing statistical tests by providing an intuitive, graphical way to detect patterns, biases, or other forms of non-randomness that might not be evident in numerical analysis. Each visualization method has its strengths and limitations:

- Scatter plots and autocorrelation plots are excellent for detecting correlations and dependencies between successive values.

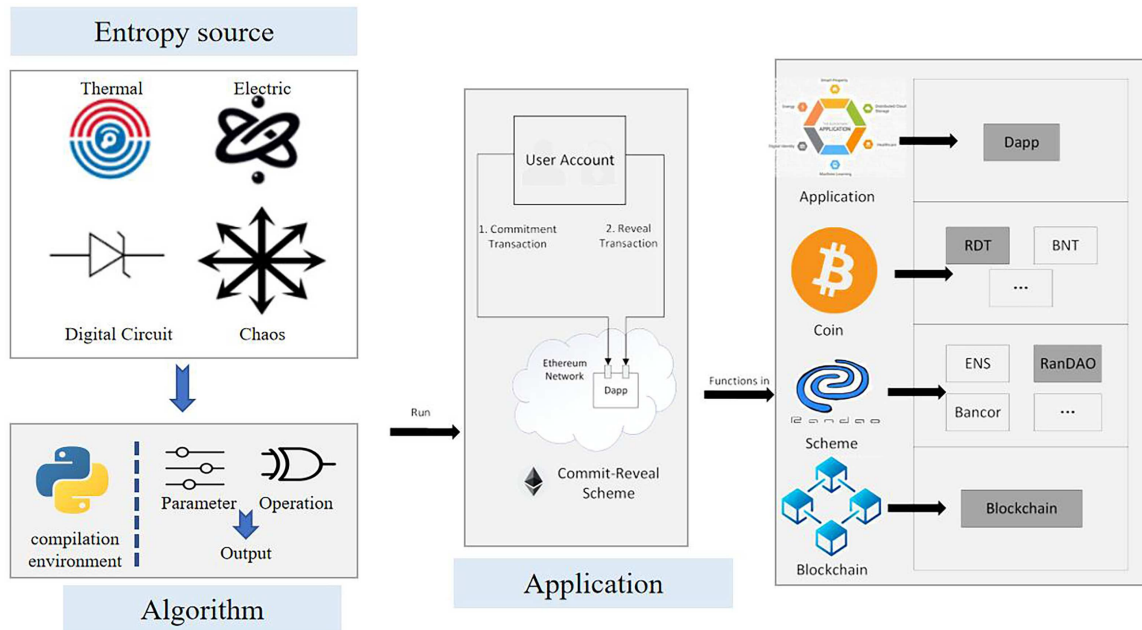


Figure 8. How TRNG Works in Block-Chain

- Histograms and Q-Q plots are effective for identifying distributional deviations, ensuring the TRNG generates uniformly distributed values.
- Power spectrum analysis reveals hidden periodicities in the data that might otherwise go unnoticed.
- Box plots and CDF plots provide a summary of the overall distribution and can help identify outliers or skewness.
- Heatmaps and time series plots are more specialized, useful for identifying complex dependencies or trends over time.

A combination of these visualizations, along with statistical tests, gives a comprehensive picture of the randomness quality of TRNG outputs. While visual analysis is more subjective and qualitative, it often serves as an early warning system, flagging potential issues for deeper investigation. For thorough TRNG evaluation, both visual and statistical approaches should be used in tandem to ensure the highest standards of randomness are met.

5 TRNGs for block-chain in industrial networks

Block-chain technology has become popular because of its decentralized design, permanent records and ability to allow secure direct transactions without the need for a centrally trusted party. In industrial networks, where it is important to be able to communicate securely, prove that devices are who they say they are, and share data that you can trust, block-chain is being used to make everything more transparent, trackable and strong. All of these block-chain-based applications use cryptographic protocols that depend on high-quality random numbers for tasks such as key generation, digital signatures, consensus algorithms, and smart contract execution. TRNGs get randomness from unpredictable physical processes. This makes TRNGs particularly well-suited for block-chain-enhanced industrial environments. For example, TRNGs can be used to create secure cryptographic keys for verifying devices in industrial IoT networks, make randomness more reliable in consensus mechanisms like Proof-of-Stake (PoS), and ensure the correct execution of smart contracts across distributed control systems. As industrial systems continue to use decentralized architectures to reduce attack surfaces and enhance operational security, TRNGs are set to become a basic component. TRNGs help to connect physical randomness with block-chain security mechanisms, which makes them a key part of making industrial networks more secure [76–81]. TRNG can work in Block-Chain like in Figure 8.

The integration of TRNGs in block-chain can enhance security and functionality in several critical areas: First, in block-chain systems, the security of private keys are paramount, as private keys are used to authorize transactions. If these keys are generated using a weak or predictable source of randomness, the entire block-chain network could be at risk. TRNGs provide a high-entropy source for generating cryptographic keys, significantly reducing the likelihood of key prediction or compromise. By introducing truly random keys, block-chain platforms can ensure the safety of their users' digital assets and transaction records. Second, many block-chain networks rely on consensus protocols, such as Proof of Work (PoW), Proof of Stake (PoS), or newer algorithms like Proof of Authority (PoA) and Delegated Proof of Stake (DPoS), to verify transactions and add new blocks to the chain. TRNGs can improve these protocols by making sure that the selection processes within the consensus mechanism are truly unpredictable. For example, in PoS-based systems, where validators are selected at random, a TRNG-based approach would stop any malicious actors from predicting or manipulating the selection process, making the network fairer and safer. Thirdly, smart contracts (self-executing contracts with predefined rules) often rely on random number generation for certain operations, such as lotteries, auctions, and randomized rewards. If a random number generator can be predicted, then an adversary could manipulate the outcome. [82–86]

One concrete application of TRNGs in block-chain systems is in the generation of secure and unpredictable nonces during the mining or block proposal process. In Proof-of-Work (PoW) and certain Proof-of-Stake (PoS) protocols, miners or validators must generate a nonce that satisfies a cryptographic condition (*e.g.*, resulting in a hash with a certain number of leading zeros). If this nonce can be predicted or manipulated, attackers may gain an unfair advantage, compromising fairness and decentralization. By using hardware-based TRNGs to generate these nonces, systems can ensure that the values are genuinely unpredictable and derived from physical entropy sources such as thermal noise or quantum fluctuations. This prevents attackers from precomputing favorable nonces or reversing the nonce-generation process. For example, in a mining rig or validator node, an embedded TRNG can generate a high-entropy nonce for each new block, which is then combined with block data for hashing. Because the nonce cannot be reproduced or anticipated, this enhances resistance against pre-image and replay attacks, ensuring greater fairness in block competition and immutability in block-chains. Additionally, TRNGs can be used to record node behavior unpredictably for consensus integrity, or in smart contract logic where deterministic randomness from PRNGs would be vulnerable to manipulation by insiders. Therefore, TRNGs contribute to both protocol-level randomness and execution-level integrity, making them a key enabler for secure, decentralized block-chain infrastructures.

Integrating true random number generators (TRNGs) into smart contracts ensures the randomness is truly unpredictable, strengthening the fairness and integrity of decentralized applications (dApps) operating on block-chain. In PoW-based block-chains like Bitcoin, miners compete to solve complex cryptographic puzzles to validate transactions and earn rewards. The difficulty of the puzzles can be adjusted based on network conditions, but they must still be random to prevent miners from optimizing and potentially controlling the process. TRNGs can make the mining process more unpredictable, which would make it harder for attackers to manipulate block creation or fake transactions [87, 88].

The future of TRNGs in block-chain is exciting. There are several areas that we can research. For example, we need to find ways to make TRNGs work better in block-chain. This means making TRNGs more efficient and making sure they can work well with block-chain systems without adding too much delay or extra computing steps. Researchers will need to focus on balancing high-quality randomness with low energy consumption and fast generation speeds. This is important for block-chain's distributed and resource-constrained nodes. Secondly, block-chain technology has been praised for its transparency, but this can sometimes conflict with the need for privacy. TRNGs could be added to advanced privacy protocols like Zero-Knowledge Proofs (ZKPs) or Ring Signatures, which would make block-chain's privacy features even better. Using TRNGs in privacy-focused block-chain platforms can make sure that transactions are more secure and anonymous.

Also, as block-chain networks grow, so does the number of sophisticated attacks aimed at compromising them. Research should look at how TRNGs can help defend against advanced block-chain threats such as Sybil attacks, 51 percent attacks, or front-running attacks in decentralized finance (DeFi). Adding TRNGs to the main block-chain security protocols could reduce these risks by making it harder for attackers to break in. Another good idea for future research is to explore hybrid consensus mechanisms that combine TRNG-based randomness with existing block-chain protocols. For example, a hybrid PoW/PoS

model enhanced by TRNG could lead to more secure and efficient consensus algorithms that are resistant to manipulation while preserving decentralization [89, 90].

TRNGs have the potential to improve the security, fairness and reliability of block-chain technologies. They can make a really random number that is really good quality, which makes them perfect for making cryptographic keys, agreeing on the block-chain, and running smart contracts. More research is needed to deal with the practical integration challenges and to make the most of the potential of TRNGs in this evolving technological landscape.

6 TRNGs with machine-learning in industrial networks

TRNGs are very important in cryptography, secure communications and complex simulations where high-quality randomness is essential. They are different from PRNGs, which are algorithmically determined and potentially predictable. TRNGs get randomness from unpredictable physical things like electronic noise, thermal fluctuations, or quantum effects. But they can be affected by things like temperature and electrical noise, which can make them unreliable and inconsistent. This can be a problem in places like industrial environments where there is a lot of noise and things are always changing. However, the use of machine learning (ML) and artificial intelligence (AI) could improve TRNG design, operation, and deployment in industrial networks. Using algorithms to monitor, filter and calibrate entropy sources in real-time can improve stability, randomness quality and fault tolerance of TRNGs in harsh industrial conditions. TRNGs can also help to protect ML systems by providing secure keys and random numbers, and by protecting models from being changed by hackers. In industrial networks, where lots of devices, sensors, and controllers use AI for automation, anomaly detection, and predictive maintenance, adding TRNGs to ML pipelines makes them more secure and reliable. This combination creates new opportunities for industrial systems that are secure, adaptable and resilient, and which are driven by reliable data and strong randomness. [91, 92].

There are two ways that TRNGs and machine learning can interact.

Firstly, using machine learning for TRNGs: The main goal here is to use ML to improve TRNG performance by making it more robust, stable and efficient. Machine learning models can help in noise filtering, error correction, and anomaly detection in the TRNG's output, thus increasing the quality of the random numbers produced. For example, machine learning algorithms can predict and deal with the effects of outside factors (like temperature or voltage changes) on the entropy source. Additionally, machine learning can be used to detect when the TRNG starts to produce random numbers that deviate from the ideal, flagging potential vulnerabilities before they compromise security [93].

Secondly, TRNGs for machine learning: TRNGs offer highly unpredictable, non-deterministic data, which can benefit machine learning models in several ways. One potential application is in randomness injection, where TRNG-generated data could be used to introduce randomness into the training process, thereby reducing overfitting and improving model generalization. Moreover, TRNGs could play a role in securing machine learning pipelines by generating secure encryption keys for data transmission and model parameters, making it more difficult for adversaries to perform attacks such as model inversion or adversarial attacks. Furthermore, TRNG-generated noise can be introduced into training data to create more robust models that can resist such attacks, as well as prevent data leakage in privacy-sensitive applications [94]. Figure 9 shows the interaction between TRNG and ML.

Given the potential of combining TRNGs with machine learning, several future research directions are worth exploring. Firstly, a significant area of interest is the design of adaptive TRNG systems that use ML algorithms to dynamically adjust the entropy source based on real-time environmental feedback. This could involve a closed-loop system where machine learning (ML) models monitor TRNG and make immediate corrections to improve performance under different conditions. Secondly, adding TRNGs to machine learning pipelines could make ML applications much more secure. Research in this area could look at how TRNG-generated cryptographic keys could secure both model weights and data transmission, reducing the risk of model tampering, data poisoning, or adversarial attacks. Thirdly, random initialization is crucial for setting the model's parameters, especially the weights of neural networks and other machine learning models. Good random initialization can help the model to find the best solution quickly, avoid getting stuck in a local optimum, and improve the final performance of the model. We also need to explore more ways to use TRNGs to add randomness to machine learning models during training and inference.

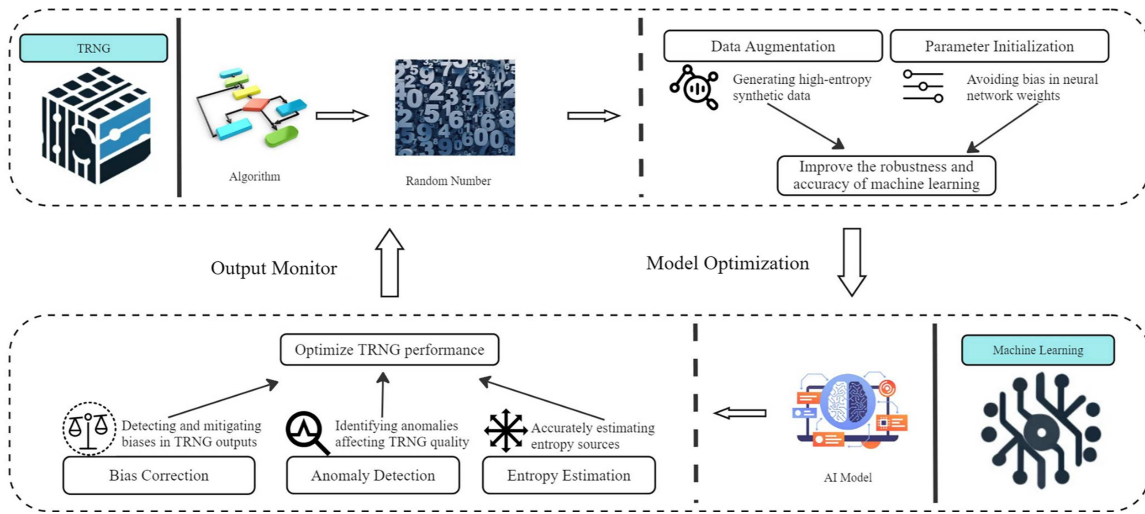


Figure 9. Interaction between TRNG and Machine-Learning

This could be particularly valuable for deep learning models that require a certain degree of randomness to achieve better performance and reliability [95–97].

In a word, the combination of TRNGs and machine learning is a field that has a lot of potential but has not yet been fully explored. If we can combine the strengths of both, we can make TRNGs better and more secure while also making machine learning models stronger and more private. This area of research has the potential to lead to significant advances in secure computing and AI security.

7 How TRNGs work with software systems or APIs

Although TRNGs are fundamentally hardware-based, their effectiveness depends heavily on how they interface with software systems. In modern computing environments, TRNGs serve as the root source of entropy for operating systems, cloud infrastructures, and embedded platforms. Their integration into software stacks typically occurs through low-level drivers, cryptographic libraries, or API interfaces provided by the underlying hardware platform.

In secure operating systems, TRNGs are commonly embedded in security modules such as TPMs, cryptographic accelerators, or SoCs. These hardware components feed entropy into the system’s random pool, which is then made available to user-space applications. For instance, Linux systems use daemons like `rngd` to transfer entropy from TRNG hardware into `/dev/random` or `/dev/urandom`, while modern mobile systems like iOS or Android access hardware TRNGs within Secure Enclave or TrustZone environments to generate keys, authenticate users, or protect secure storage. Additionally, some firmware-level components, including BIOS and UEFI, leverage TRNGs during system initialization to establish trusted identities and secure boot processes.

In cloud computing and virtualization environments, TRNGs play a key role in ensuring entropy availability across isolated and containerized systems. Since virtual machines and containers often lack physical access to entropy sources, cloud platforms use host-level TRNGs or external hardware security modules (HSMs) to provide randomness. Services such as AWS CloudHSM or Azure Key Vault expose high-entropy randomness to clients for secure key generation and cryptographic operations. These TRNGs are typically accessed through standard APIs such as `function getrandom` or through integration with cryptographic libraries like OpenSSL. Hypervisors may also pass TRNG entropy to guest systems via virtual hardware interfaces to mitigate entropy starvation in multi-tenant environments [98].

In embedded systems and IoT devices, TRNGs are typically integrated directly into microcontrollers or system-on-chips (SoCs) as compact hardware blocks. These TRNGs often draw entropy from sources like thermal noise or oscillator jitter, and are accessed via hardware abstraction layers (HAL) provided by the chip manufacturer. Software development kits (SDKs) for platforms such as STM32, ESP32, or Nordic chips allow developers to read entropy values from TRNG peripherals through simple API calls.

Furthermore, lightweight cryptographic libraries like mbedTLS or wolfSSL offer interfaces that directly accept TRNG output for use in symmetric and asymmetric encryption protocols. These devices often implement built-in health checks or self-tests to validate the quality of entropy before use, ensuring compliance with cryptographic security standards.

In summary, TRNGs interface with software systems through a wide range of mechanisms—from operating system entropy pools to cryptographic APIs and hardware-specific SDKs. Their integration ensures that upper-layer software components, including encryption protocols, authentication services, and secure communication frameworks, can reliably access high-entropy randomness. This close coupling between hardware entropy and software application ensures robust security foundations across industrial, cloud, and embedded computing environments.

8 Threats against TRNG

TRNGs play a crucial role in cryptographic systems, providing genuine randomness essential for generating secure keys, nonces, and other critical cryptographic elements. However, TRNGs are not immune to attacks, particularly in environments where adversaries have access to their physical properties or outputs. Various attack vectors aim to compromise the unpredictability of TRNGs by either revealing patterns in their outputs or manipulating the randomness generation process. This section outlines the primary attack strategies targeting TRNGs, including traditional methods and emerging machine learning (ML)/deep learning (DL)-based techniques [94].

8.1 Traditional attacks on TRNGs

8.1.1 Side-channel attacks (SCAs)

Side-channel attacks exploit unintended physical emissions (*e.g.*, power consumption, electromagnetic radiation, timing) generated during the operation of TRNGs. Attackers leverage this leakage to extract useful information about the internal state of the TRNG, which can then be used to predict future random numbers [99, 100].

- Power Analysis Attacks: Power analysis is a well-known method where attackers measure the power consumed by the TRNG during its operation. Techniques such as Differential Power Analysis (DPA) or Correlation Power Analysis (CPA) allow attackers to detect subtle variations in power that correlate with specific output bits of the random number generator. By performing statistical analysis on these measurements, attackers can reduce the uncertainty of future TRNG outputs.
- Electromagnetic (EM) Attacks: Similar to power analysis, electromagnetic attacks rely on detecting the electromagnetic radiation emitted by TRNG circuitry during random number generation. By measuring and analyzing these emissions, attackers may infer patterns in the generated random numbers, especially if the TRNG exhibits weaknesses under specific environmental conditions or design flaws.
- Timing Attacks: Attackers may measure the time it takes for a TRNG to generate a random number. Any dependency between timing and the randomness generation process can be exploited to infer the output, particularly in cases where timing discrepancies are correlated with internal states of the TRNG.

8.1.2 Fault injection attacks

Fault injection attacks involve deliberately disturbing the TRNG's operation to reduce its randomness or cause it to generate predictable output. This can be done through physical means, such as injecting laser pulses, altering voltage supply, or introducing extreme environmental changes like temperature variations.

- Laser Injection: By shining laser pulses on specific areas of a TRNG circuit, attackers can alter the behavior of transistors or logic gates, leading to faults in the random number generation process. These faults can either cause the TRNG to output repeated sequences or reduce the entropy of the generated numbers.
- Voltage/Temperature Manipulation: TRNGs are sensitive to process, voltage, and temperature (PVT) variations. Attackers can manipulate these environmental factors to induce faulty behavior in TRNGs. For instance, reducing the voltage below operational thresholds or increasing temperature may cause the TRNG to generate biased or correlated outputs, making them vulnerable to cryptographic analysis.

8.1.3 Algorithmic attacks

TRNGs typically include post-processing algorithms such as whitening or entropy extraction to ensure uniformity in their output distribution. Attackers may reverse-engineer or exploit vulnerabilities in these algorithms to predict the final random number outputs.

- Weakness in Entropy Extraction: If the TRNG relies on a suboptimal entropy extraction algorithm, it may introduce patterns in the output that an attacker can exploit. A compromised entropy extraction process can provide an adversary with the ability to partially predict the output of the TRNG by reducing the overall entropy in the final random numbers.

8.2 Machine learning and deep learning-based attacks

In recent years, ML and DL have emerged as powerful tools to enhance traditional attack techniques. These models can automatically learn intricate patterns in complex datasets, which has significant implications for attacking TRNGs [101].

8.2.1 ML-augmented side-channel attacks

Traditional side-channel attacks, such as power analysis and electromagnetic attacks, have been enhanced using machine learning algorithms. These attacks leverage ML/DL to automate and improve pattern recognition from the vast amounts of side-channel data collected.

- Deep Learning-Based Power Analysis: By training deep neural networks (*e.g.*, Convolutional Neural Networks, CNNs) on power traces collected from the TRNG, attackers can learn hidden patterns in the power consumption related to specific random number outputs. These models can outperform traditional statistical techniques by finding non-linear correlations and subtle dependencies in the side-channel data, making attacks more efficient and effective.
- Electromagnetic Side-Channel Attacks with ML: Similarly, ML models can process electromagnetic emission data from TRNGs to detect correlations between signal characteristics and the random numbers being generated. With sufficient training data, a neural network can potentially predict future random outputs by learning how the internal states of the TRNG influence the side-channel emissions.

8.2.2 Generative model attacks

Generative models, such as Generative Adversarial Networks (GANs), can be used to simulate the behavior of TRNGs. Attackers can train a GAN using legitimate TRNG output data to generate synthetic random numbers that closely resemble the TRNG's actual output.

- GAN Attacks: By replicating the TRNG's output characteristics, an attacker may replace the TRNG's legitimate output with synthetic numbers generated by the GAN. If these numbers are indistinguishable from the real TRNG output, this could compromise the integrity of cryptographic systems that rely on TRNG for randomness.

8.2.3 Pattern recognition in TRNG output

Even in cases where TRNGs are functioning correctly, imperfections in the design or environmental influences may introduce subtle biases in the output. Machine learning models can be trained to detect and exploit these biases.

- Machine Learning-Based Randomness Detection: A machine learning model could analyse large sets of TRNG output and detect small statistical anomalies or patterns. These models have the capacity to reveal non-random behaviour that may not be detected by traditional randomness tests, thereby providing attackers with a means to predict future outputs with greater accuracy than brute-force approaches.

8.3 Mitigation strategies and research directions

As attack techniques against TRNGs become more sophisticated, including ML/DL-based methods, countermeasures need to evolve. Some potential directions for improving TRNG security include:

- Real-Time Entropy Monitoring: Machine learning models can be used to continuously monitor the entropy level of TRNG output, ensuring that any anomalies are detected in real-time, and appropriate corrective actions are taken.
- Adaptive Fault Detection: Machine learning can also be applied to detect and mitigate fault injection attacks by identifying abnormal operational states or patterns that deviate from expected TRNG behavior.
- Designing TRNGs Resilient to ML Attacks: Future TRNG designs may need to include features specifically intended to thwart ML/DL-based attacks, such as adding more unpredictability through adaptive hardware designs or introducing random noise that confounds ML models.

8.4 Ethical and regulatory concerns in TRNG deployment

While the technical design and statistical testing of TRNGs are well studied, ethical and regulatory concerns remain under-discussed yet highly relevant in real-world deployment. One of the most critical ethical risks lies in the possibility of “backdoored” TRNGs—devices that appear statistically random but have been intentionally engineered to produce predictable outputs. Such vulnerabilities may stem from manipulated entropy sources, biased post-processing algorithms, or hidden control mechanisms. A notable parallel is the controversy surrounding Dual_EC_DRBG, a standardized PRNG later suspected of containing a backdoor exploitable by its designers. Similar risks exist in hardware TRNGs if the physical entropy source or its digital interface is not transparent or auditable. To mitigate these risks, the design and implementation of TRNGs—particularly those used in critical infrastructures or cryptographic modules—should be subject to rigorous third-party verification and, where possible, open design principles that allow for public scrutiny [102]

Regulatory frameworks have begun to address these issues in part. Standards such as NIST SP 800-90B and FIPS 140-3 require entropy sources to undergo health testing, continuous monitoring, and statistical validation to ensure the unpredictability and robustness of TRNGs. In Europe, AIS-31 from Germany’s BSI goes further by introducing runtime health tests and classification levels for evaluating TRNG quality and suitability for secure applications such as smart cards and eIDs. Though less explicit, GDPR and ISO/IEC 15408 (Common Criteria) also imply strong requirements for cryptographic randomness, especially in privacy-critical systems. However, there is still a lack of unified global standards that address TRNG deployment from an ethical and national security perspective. As a result, the use of uncertified or proprietary TRNGs in sensitive applications—such as digital identity systems, election platforms, or financial cryptography—poses significant risks.

Ensuring transparency and trust in TRNGs thus becomes not only a technical challenge but also an ethical imperative. Deployments in secure operating systems, industrial controllers, or cloud-based cryptographic services should disclose the source and certification status of their randomness. Where TRNGs are embedded into user-facing systems, users should be informed of their security guarantees and any known limitations. These concerns underline the need for further research and regulation in the intersection of hardware-based security, randomness assurance, and public accountability.

9 Future direction

Despite a lot of progress in TRNG research, there are still several challenges. In the future, most improvements will focus on two areas: making the randomness better and more stable, and making it more useful in more situations.

First, the industry is still looking for better ways to make TRNGs more random, stable and cheap. In future, we will try to make new physical mechanisms for generating entropy, improve the robustness of existing sources, and deal with environmental biases. At the same time, the way TRNGs are made will get better at using less power, making them smaller so they can be used in more devices, and making it

easier to make lots of TRNGs in large factories. Another important thing is making sure they are secure. This will need more rigorous ways to check randomness, certification frameworks, and resistance against new attacks to make people trust security solutions based on TRNGs.

Second, TRNGs will be used in more areas as they are added to new technologies. In block-chain, for example, TRNG modules in IoT devices can be used to make PoS consensus protocols, zero-knowledge proofs and decentralized applications more secure by adding an element of randomness. Finally, TRNGs and machine learning working together is an exciting area to research. TRNGs can make machine learning models more random, which can help them to generalize better and be more secure when faced with challenges. In the same way, machine learning can help analyse the random numbers produced by TRNGs, spot any biases, and measure how random the numbers are. This can make TRNGs more reliable and secure.

As research in this area continues, TRNGs will become more and more important in protecting digital systems and supporting new cryptographic and computational systems. Solving these problems and finding new uses for TRNGs will make them an even more important technology in industry and security.

10 Conclusion

To summarize, this survey has outlined the key aspects of TRNGs, including their definition, types, challenges, and potential applications. A comparison of TRNGs with PRNGs was conducted, highlighting TRNGs' superior security but noting practical limitations. The survey also described several TRNG testing and visualization methods that are currently in use. The survey emphasized the growing role of TRNGs in block-chain, where they enhance cryptographic security, and in machine learning, where they improve robustness by introducing genuine randomness. However, challenges such as the reliability of entropy sources, hardware constraints, and system integration persist. Despite these challenges, TRNGs remain indispensable for applications that demand high-quality randomness. Their integration with technologies such as block-chain and machine learning holds considerable promise for future advancements in security and efficiency. Continued research in this area is therefore essential to address current challenges and expand the practical applications of TRNGs.

Acknowledgments

We would like to express our sincere gratitude to Lab of Network Intelligence and Security, IPS, Waseda University for the invaluable guidance and support throughout this research

Funding

This work was supported in part by the JSPS KAKENHI under Grants 23K11072 and 24KF0259, and in part by the Telecommunications Advancement Foundation.

Conflicts of interest

The authors declare no conflicts of interest.

Data availability statement

No data are associated with this article.

Author contribution statement

Haozhe Chai: He studied the conception (formulation of research question), literature organization, framework construction, data analysis and interpretation. He also did the writing/manuscript preparation (writing initial draft), and final approval of the version to be published. Qianqian Pan: He studied conception (formulation of research question), guided framework supplement and guided data analysis. He also did the writing/manuscript preparation (commentary and revision), and final approval of the version to be published. Jun Wu: He studied the conception (formulation of research question), guided figure drawing and guided result interpretation. He also did the writing/manuscript preparation (critical review and revision), and final approval of the version to be published.

References

- [1] Priyanka IH and Khalique A. Random number generators and their applications: a review 2019; **7**: 1777–81.
- [2] Du M, Chen Q and Liu L et al. A blockchain-based random number generation algorithm and the application in blockchain games. In: 2019 IEEE Int. Conf. Syst. Man Cybern. (SMC), Bari, Italy, 2019, 3498–503, <http://doi.org/10.1109/SMC.2019.8914618>.
- [3] Ashraf A and Elmedany W. Authentication in IoT devices using blockchain technology: A review. In: 4th Smart Cities Symp. (SCS 2021), Online Conference, Bahrain, 2021, 545–51, <http://doi.org/10.1049/icp.2022.0398>.
- [4] Choi J, Shin W and Kim J et al. Random seed generation for IoT key generation and key management system using blockchain. In: 2020 Int. Conf. Inform. Networking (ICOIN), Barcelona, Spain, 2020, 663–5, <http://doi.org/10.1109/ICOIN48656.2020.9016518>.
- [5] Ansari U, Chaudhary AK and Verma S. True random number generator (TRNG) using sensors for low cost IoT applications. In: 2022 Int. Conf. Commun. Comput. Internet Things (IC3IoT), Chennai, India, 2022, 1–6, <http://doi.org/10.1109/IC3IoT53935.2022.9767999>.
- [6] Pan Q, Wu J and Zheng X et al. Differential privacy and IRS empowered intelligent energy harvesting for 6G internet of things. *IEEE Internet Things J* 2022; **9**, 22109–22, <http://doi.org/10.1109/JIOT.2021.3104833>.
- [7] Pan Q, Wu J and Bashir AK et al. Joint protection of energy security and information privacy for energy harvesting: an incentive federated learning approach. *IEEE Trans Indust Inf* 2022; **18**: 3473–83, <http://doi.org/10.1109/TII.2021.3105492>.
- [8] Lee K, Lee, S-Y and Seo C et al. TRNG (True Random Number Generator) Method using visible spectrum for secure communication on 5G network. *IEEE Access* 2018; **6**: 12838–47, <http://doi.org/10.1109/ACCESS.2018.2799682>.
- [9] Priyanka IH and Aqeel K. Random number generators and their applications: a review 2019; **7**: 1777–81.
- [10] Arciuolo T and Elleithy KM. Parallel, true random number generator (P-TRNG): using parallelism for fast true random number generation in hardware. In: 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC), NV, USA, 2021, 0987–92, <http://doi.org/10.1109/CCWC51732.2021.9375939>.
- [11] Yang J et al. A low cost and high reliability true random number generator based on resistive random access memory. In: 2015 IEEE 11th International Conference on ASIC (ASICON), Chengdu, China, 2015, 1–4, <http://doi.org/10.1109/ASICON.2015.7516996>.
- [12] Stipčević M and Koç Ç K. True random number generators. *Open Problems in Mathematics and Computational Science*. Cham: Springer International Publishing, 2014, 275–315.
- [13] Hong SL and Liu C. Sensor-based random number generator seeding. *IEEE Access* 2015; **3**: 562–8.
- [14] Fischer V and Drutarovský M. True random number generator embedded in reconfigurable hardware. In: *Int. Workshop Cryptogr. Hardware Embedded Syst.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, 415–30.
- [15] Kumkar V, Tiwari A and Tiwari P et al. Vulnerabilities of wireless security protocols (WEP and WPA2). *Int J Adv Res Comput Eng Technol (IJARCET)* 2012; **1**: 34–8.
- [16] Ansari U, Chaudhary AK and Verma S. Enhanced true random number generator (TRNG) using sensors for IoT security applications. In: 2022 Third Int. Conf. Intell. Comput. Instrum. Control Technol. (ICICICT), 2022, 1593–97.
- [17] Ghafoor I, Jattala I and Durrani S. et al. Analysis of openssl heartbleed vulnerability for embedded systems. In: 17th IEEE Int. Multi Topic Conf. 2014. IEEE, 2014, 314–9.
- [18] Bernstein DJ, Lange T and Niederhagen R. Dual EC: A standardized back door. *The New Codebreakers: Essays Dedicated to David Kahn on the Occasion of His 85th Birthday*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, 256–81.
- [19] Cho S, Hong E and Seo S, Random number generator using sensors for drone. *IEEE Access* 2020; **8**: 30343–54.
- [20] Applegate MJ, Thomas O and Dynes JF et al. Efficient and robust quantum random number generation by photon number detection. *Appl Phys Lett* 2015; **107**: 071106. <http://doi.org/10.1063/1.4928732>.
- [21] Witek K, Caccia M and Kucewicz W et al. A method for implementing a SHA256 hardware accelerator inside an quantum true random number generator (QTRNG). In: 2023 30th Int. Conf. Mixed Design Integ. Circuits Syst. (MIXDES), Krakow, Poland, 2023, 251–6, <http://doi.org/10.23919/MIXDES58562.2023.10203268>.
- [22] Gimenez G, Cherkaoui A and Frisch R et al. Self-timed ring based true random number generator: threat model and countermeasures. In: 2017 IEEE 2nd Int. Verif. Secur. Workshop (IVSW), Thessaloniki, Greece, 2017, 31–8, <http://doi.org/10.1109/IVSW.2017.8031541>.
- [23] Pdv-p8104 datasheet.
- [24] Degada A and Thapliyal H. Harnessing uncertainty in photoresistor sensor for true random number generation in IoT devices. In: 2020 IEEE Int. Conf. Consumer Electron. (ICCE), 2020, 1–5, <http://doi.org/10.1109/ICCE46568.2020.9042967>.

- [25] Mller-Olm M and Seidl H. Analysis of modular arithmetic. In: Programming Languages and Systems: 14th European Symp. Programming, ESOP 2005, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2005, Edinburgh, UK, April 4-8, 2005. Proceedings 14, Springer Berlin Heidelberg, 2005, 46–60.
- [26] Georgescu C, Simion E and Nita AP et al. A view on NIST randomness tests (in) dependence. In: 2017 9th Int. Conf. Electron. Comput. Artif. Intell. (ECAI), IEEE, 2017, 1–4.
- [27] Randao Provably Fair Random Number White Paper, randao.org, September, 11, 2017.
- [28] Crocetti L, Nannipieri P and Di Matteo S et al. Review of methodologies and metrics for assessing the quality of random number generators. *Electronics* 2023; **12**: 723.
- [29] Wang Y, Yu W and Wu S et al. Flash memory for ubiquitous hardware security functions: true random number generation and device fingerprints. In: 2012 IEEE Symp. Secur. Privacy, IEEE, 2012, 33–47.
- [30] Kalai YT and Raz R. Interactive pcp. International Colloquium on Automata, Languages, and Programming. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, 536–47.
- [31] Alturki MA and Roşu G. Statistical model checking of RANDAOs resilience to pre-computed reveal strategies. In: Formal Methods. FM 2019 International Workshops: Porto, Portugal, October 7–11, 2019, Revised Selected Papers, Part I 3. Springer International Publishing, 2020, 337–49.
- [32] Han R, Lin H and Yu J. Randchain: a scalable and fair decentralized randomness beacon. *Cryptology ePrint Archive*, 2020.
- [33] Alturki MA and Roşu G. Statistical model checking of RANDAOs resilience to pre-computed reveal strategies. In: Formal Methods. FM 2019 Int. Workshops: Porto, Portugal, October 7–11, 2019, Revised Selected Papers, Part I 3. Springer International Publishing, 2020, 337–49.
- [34] Jia Z, Chen R and Li J. Delottery: a novel decentralized lottery system based on blockchain technology. in: Proc. 2019 2nd Int. Conf. Blockchain Technol. Appl. 2019, 20–5.
- [35] Rahman A, Nasir MK and Rahman Z et al. Distblockbuilding: a distributed blockchain-based sdn-iot network for smart building management. *IEEE Access* 2020; **8**: 140008–18.
- [36] Han R, Yu J and Lin H. RandChain: decentralised randomness beacon from sequential proof-of-work. *IACR Cryptol. ePrint Arch* 2020; **2020**: 1033.
- [37] Komargodski I and Tamir Y. On distributed randomness generation in blockchains. International Symposium on Cyber Security, Cryptology, and Machine Learning, Cham: Springer Nature Switzerland, 2023, 49–64.
- [38] Lokshina IV, Gregu M and Thomas WL. Application of integrated building information modeling, IoT and blockchain technologies in system design of a smart building. *Procedia Comput Sci* 2019; **160**: 497–502.
- [39] Van Cutsem O, Dac DH and Boudou P et al. Cooperative energy management of a community of smart-buildings: A Blockchain approach. *Int J Electr Power Energy Syst* 2020; **117**: 105643.
- [40] Liu Z, Chi Z and Osmani M et al. Blockchain and building information management (BIM) for sustainable building development within the context of smart cities. *Sustainability* 2021; **13**: 2090.
- [41] Frustaci F, Spagnolo F and Corsonello P et al. A high-speed and low-power DSP-based TRNG for FPGA implementations. *IEEE Trans Circuits Syst II: Express Briefs* 2024;
- [42] Chen, T, Ma Y and Lin J et al. A lightweight full entropy TRNG with on-chip entropy assurance. *IEEE Trans Comput-Aided Design Integr Circuits Syst* 2021; **40**: 2431–44.
- [43] Morsali M, Moaiyeri MH and Rajaei R. A process variation resilient spintronic true random number generator for highly reliable hardware security applications. *Microelectron J* 2022; **129**: 105606.
- [44] Williams S, Khalil K, Bayoumi M. A Novel 0.04 pJ/bit dynamic TRNG using bit reconfigurable ring oscillators. In: SoutheastCon 2024, 2024, 1546–52.
- [45] McCullough BD. A review of TESTU01., 2006, 677–82.
- [46] Suci A, Toma RA and Marton K. Parallel implementation of the TestU01 statistical test suite. In: 2012 IEEE 8th Int. Conf. Intell. Comput. Commun. Process., Cluj-Napoca, Romania, 2012, 317–22, <http://doi.org/10.1109/ICCP.2012.6356206>.
- [47] Barak B, Shaltiel R and Tromer E. True random number generators secure in a changing environment. In: Cryptographic Hardware and Embedded Systems-CHES 2003: 5th Int. Workshop, Cologne, Germany, September 8–10, 2003. Proceedings 5, Springer Berlin Heidelberg, 2003, 166–80.
- [48] Wang X, Zhang R and Wang Y et al. A 0.116pJ/bit latch-based true random number generator with static inverter selection and noise enhancement. In: 2022 Int. Symp. VLSI Design Automation Test (VLSI-DAT), Hsinchu, Taiwan, 2022, 1–4, <http://doi.org/10.1109/VLSI-DAT54769.2022.9768078>.
- [49] Kohlbrenner P and Gaj K. An embedded true random number generator for FPGAs. In: Proc. 2004 ACM/SIGDA 12th Int. Symp. Field Programmable Gate Arrays, 2004, 71–8.
- [50] Roić V and Verbauwhede I. Hardware-efficient post-processing architectures for true random number generators. *IEEE Trans Circuits Syst II: Express Briefs* 2019; **66**: 1242–6.

- [51] Pan Q, Wu J and Zheng X et al. Differential privacy and IRS empowered intelligent energy harvesting for 6G internet of things. *IEEE Int Things J* 2022; **9**: 22109–22, <http://doi.org/10.1109/JIOT.2021.3104833>.
- [52] Yao Y, Chen X and Kang W et al. Thermal brownian motion of skyrmion for true random number generation. *IEEE Trans Electron Devices* 2020; **67**: 2553–8, <http://doi.org/10.1109/TED.2020.2989420>.
- [53] Mezher AE. Enhanced RSA cryptosystem based on multiplicity of public and private keys. *Int J Electr Comput Eng* 2018; **8**: 3949.
- [54] Schuba CL, Krsul IV and Kuhn MG et al. Analysis of a denial of service attack on TCP. *Proc. 1997 IEEE Symp. Secur. Privacy (Cat. No. 97CB36097)*, IEEE, 1997, 208–23.
- [55] Liu P. Public-key encryption secure against related randomness attacks for improved end-to-end security of cloud/edge computing. *IEEE Access* 2020; **8**: 16750–9, <http://doi.org/10.1109/ACCESS.2020.2967457>.
- [56] Peng Y, Zhao H and Sun X et al. A side-channel attack resistant AES with 500Mbps, 1.92pJ/Bit PVT variation tolerant true random number generator. In: 2017 IEEE Comput. Soc. Ann. Symposium VLSI (ISVLSI), Bochum, Germany, 2017, 249–54, <http://doi.org/10.1109/ISVLSI.2017.51>.
- [57] Teng X, Liu X and Zhang Y. A UHF passive RFID tag front-end design with a novel true random number generator. In: *IEEE Access*, <http://doi.org/10.1109/ACCESS.2024.3378128>.
- [58] Guy J, Ambrosi E and Wu C-H et al. Ultrafast ~7 Mbps true random number generator based on SNGCT selector. *IEEE Trans Electron Devices* 2024; **71**: 2794–800, <http://doi.org/10.1109/TED.2024.3371424>.
- [59] Zhang Z et al. Ultra-fast true random number generator based on ill-posedness nucleation of skyrmion bags in ferrimagnets. In: *IEEE Electron Device Lett* 2024; **45**: 917–20, <http://doi.org/10.1109/LED.2024.3368210>.
- [60] Pandey SK and Jenef R. A comparative study and analysis of quantum random number generator with true random number generator, In: 2024 16th Int. Conf. COMMun. Syst. NETWORKS (COMSNETS), Bengaluru, India, 2024, 1000–5, <http://doi.org/10.1109/COMSNETS59351.2024.10426934>.
- [61] Chen Y, Tian Y and Zhou R et al. NDSTRNG: non-deterministic sampling-based true random number generator on SoC FPGA systems. *IEEE Trans Comput* 2024; **73**: 1313–26, <http://doi.org/10.1109/TC.2024.3365955>.
- [62] Yang S et al. Lightweight hybrid entropy source true random number generator based On jitter and metastability. In: *IEEE Trans. Circuits Syst. II: Express Briefs*, <http://doi.org/10.1109/TCSII.2024.3363015>.
- [63] Kim J and Chae H. A 10-Gb/s True random number generator using ML-resistant middle square method. *IEEE J Solid-State Circuits*, <http://doi.org/10.1109/JSSC.2023.3346428>.
- [64] Wang X, Zhang R and Liu K et al. A 0.116 pJ/bit latch-based true random number generator featuring static inverter selection and noise enhancement. *IEEE Trans Very Large Scale Integr (VLSI) Syst* 2024; **32**: 564–72, <http://doi.org/10.1109/TVLSI.2023.3328602>.
- [65] Zanotti T et al. Reliability analysis of random telegraph noisebased true random number generators. In: 2023 IEEE Int. Integr. Reliab. Workshop (IIRW), South Lake Tahoe, CA, USA, 2023, 1–6, <http://doi.org/10.1109/IIRW59383.2023.10477697>.
- [66] Anagnostopoulos NA et al. A method to construct efficient carbon-nanotube-based physical unclonable functions and true random number generators. In: 2023 26th Euromicro Conf. Digital Syst. Design (DSD), Golem, Albania, 2023, 61–9, <http://doi.org/10.1109/DSD60849.2023.00019>.
- [67] Petura O, Mureddu U and Bochar N et al. A survey of AIS-20/31 compliant TRNG cores suitable for FPGA devices. In: 2016 26th Int. Conf. Field Programmable Logic Appl. (FPL), IEEE, 2016, 1–10.
- [68] Rasheed AF, Zarkoosh M and Abbas SF. Comprehensive evaluation of encryption algorithms: a study of 22 performance tests. In: 2023 Sixth Int. Conf. Vocational Educ. Electr. Eng. (ICVEE), Surabaya, Indonesia, 2023, 191–4, <http://doi.org/10.1109/ICVEE59738.2023.10348240>.
- [69] Luo Y, Wang W and Best S et al. A high-performance and secure TRNG based on chaotic cellular automata topology, *IEEE Trans Circuits Syst I: Regular Papers* 2020; **67**: 4970–83, <http://doi.org/10.1109/TCSI.2020.3019030>.
- [70] Pareschi F, Rovatti R and Setti G. On statistical tests for randomness included in the NIST SP800-22 test suite and based on the binomial distribution. *IEEE Trans Inf Forensics Secur* 2012; **7**: 491–505.
- [71] Iwasaki A. Analysis of NIST SP800-22 focusing on randomness of each sequence. *JSIAM Lett* 2018; **10**: 1–4.
- [72] Dong L, Zeng Y and Ji L et al. Study on the pass rate of NIST SP800-22 statistical test suite. In: 2014 Tenth International Conference on Computational Intelligence and Security, IEEE, 2014, 402–4.
- [73] Chen D, Chen H and Fan L et al. Error analysis of NIST SP 800-22 test suite. *IEEE Trans Inf Forensics Secur* 2023.
- [74] Adesina NO, Wang B and Morell W et al. Experimental analyses of a noise-based true random number generator. In: 2023 IEEE 13th Ann. Comput. Commun. Workshop Conf. (CCWC), Las Vegas, NV, USA, 2023, 0915–8, <http://doi.org/10.1109/CCWC57344.2023.10099239>.
- [75] L'ecuyer P and Richard S. TestU01: AC library for empirical testing of random number generators. *ACM Trans Math Software (TOMS)* 2007; **33**: 1–40.

- [76] Susanti BH, Jimmy J and Mareta WA. ENT randomness test on DM-PRESENT-80 and DM-PRESENT-128-based pseudorandom number generator. In: 2021 4th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), IEEE, 2021.
- [77] Walker J, ENT-A pseudorandom number sequence test program, Fourmilab, 06. 2009.
- [78] NIST Statistical Test Suite, 06. 2009.
- [79] Marsaglia G, The diehard test suite, 06. 2009.
- [80] Alani MM. Testing randomness in ciphertext of block-ciphers using DieHard tests. *Int J Comput Sci Netw Secur* 2010; **10**: 53–7.
- [81] Brown RG, Eddelbuettel D and Bauer D. Dieharder, Duke University Physics Department Durham, NC, 2018, 27708-0305.
- [82] Abdulhameed HA et al. A lightweight hybrid cryptographic algorithm for WSNs tested by the diehard tests and the raspberry Pi. In: 2022 Int. Conf. Comput. Sci. Software Eng. (CSASE), IEEE, 2022.
- [83] Andrew Rukhin (NIST), Juan Soto (NIST), James Nechvatal (NIST), Miles Smid (NIST), Elaine Barker (NIST), Stefan Leigh (NIST), Mark Levenson (NIST), Mark Vangel (NIST), David Banks (NIST), N. Heckert (NIST), James Dray (NIST), San Vo (NIST), Lawrence Bassham (NIST), A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, NIST SP 800-22 Rev. 1
- [84] Marek S et al. A bad day to die hard: correcting the dieharder battery. *J Cryptol* 2022; **35**: 1–20.
- [85] Nagarajan KK. Randomness Analysis of YUGAM-128 Using Diehard Test Suite, 2020.
- [86] George M and Tsang WW. Some difficult-to-pass tests of randomness. *J Stat Software* 2002; **7**: 1–9.
- [87] Anastasios B, Nastou PE and Petroudis G et al. Random number generators: principles and applications. *Cryptography* 2023; **7**: 54, <http://doi.org/10.3390/cryptography7040054>.
- [88] Fei Y, Lixiang L and Qiang T et al. A survey on true random number generators based on chaos. *Discr Dyn Nat Soc* 2019; **2545123**: 10, <http://doi.org/10.1155/2019/2545123>.
- [89] Bakiri M et al. Survey on hardware implementation of random number generators on FPGA: Theory and experimental analyses. *Comput Sci Rev* 2018; **27**: 135–53.
- [90] Wen Y and Weize Y. Machine learning resistant pseudorandom number generator. *Electron Lett* 2019; **55**: 515–7.
- [91] Truong ND et al. Machine learning cryptanalysis of a quantum random number generator. *IEEE Trans Inf Forensics Secur* 2018; **14**: 403–14.
- [92] Ash-Saki A, Alam M and Ghosh S. Improving reliability of quantum true random number generator using machine learning. In: 2020 21st Int. Symp. Quality Electron. Design (ISQED). IEEE, 2020.
- [93] Wali A, Ravichandran H and Das S. A machine learning attack resilient true random number generator based on stochastic programming of atomically thin transistors. *ACS nano* 2021; **15**: 17804–12.
- [94] Kumar S. Analyzing efficiency of Pseudo-Random Number Generators using Machine Learning. In: Proc. 4th Int. Conf. Inf. Network Secur. 2016.
- [95] Li C et al. Deep learning-based security verification for a random number generator using white chaos. *Entropy* 2020; **22**: 1134.
- [96] Yu Y, Moraitis M and Dubrova E. Can deep learning break a true random number generator?. *IEEE Trans Circ Syst II: Express Briefs* 2021; **68**: 1710–4.
- [97] Barakbayeva T, Zhuo C and Amir G. SRNG: An efficient decentralized approach for secret random number generation. In: IEEE Int. Conf. Blockchain Cryptocurrency (ICBC), 2024.
- [98] Mar'ah L, Mulyawan R and Suksmono AB. Quantum random number generation for secure communications using hybrid hadamard controlled-NOT gates with XOR bitmask. In: 2024 IEEE Int. Conf. Adv. Telecommun. Networking Technol. (ATNT), Johor Bahru, Malaysia, 2024, 1–4, <http://doi.org/10.1109/ATNT61688.2024.10719212>.
- [99] Nguyen-Van T et al. A system for scalable decentralized random number generation. In: 2019 IEEE 23rd Int. Enterprise Distributed Object Comput. Workshop (EDOCW), IEEE, 2019
- [100] Son, DH, Tran Thi TQ and Le Quang M. RANDAO-based RNG: last revealer attacks in ethereum 2.0 randomness and a potential solution. arXiv preprint arXiv:2403.09541, 2024.
- [101] Hsieh C-H et al. BCsRNG: a secure random number generator based on blockchain. *IEEE Access* 2022; **10**: 98117–26.
- [102] Bernstein DJ, Lange T and Niederhagen R. Dual EC: A Standardized Back Door. In *The New Codebreakers*, Springer, 2016.



Haozhe Chai is currently a Ph.D. student of Waseda University, Japan. He got the Bachelor's Degree at Zhejiang University in 2019 and Master's Degree at Waseda University in 2024. His research interests include true random number generator and integrated circuit.



Qianqian Pan is currently the JSPS International Research Fellow, Graduate School of Information, Production and Systems, Waseda University, Japan. She received the B.S. and M.S. from the School of Information Science and Engineering, Southeast University, Nanjing, China, in 2015 and 2018, respectively. She received the Ph.D. degree from the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2023. Her research interests include blockchain, privacy protection, and next-generation networks.



Jun Wu received the Ph.D. degree in information and telecommunication studies from Waseda University, Japan, in 2011, where he is a professor. He was a Post-Doctoral Researcher with the Research Institute for Secure Systems, National Institute of Advanced Industrial Science and Technology (AIST), Japan, from 2011 to 2012. He was a Researcher with the Global Information and Telecommunication Institute, Waseda University, Japan, from 2011 to 2013. His research interests include the intelligence and security techniques of artificial intelligence, digital twin, Internet of Things (IoT), 5G/6G, molecular communication, *etc.*