Security
of Infocommunication Systems
and Internet of Things

2026 Vol 4, No 1

https://doi.org/10.31861/sisiot2026.1.01015

Received 12 May 2026; revised 10 June 2026; accepted 22 June 2026; published 30 June 2026

Optimization of Entropy Characteristics for Pseudorandom Number Sequences Based on Hybrid Cellular Automata

Serhii Yanushevskyi* **and Yurii Dobrovolsky**

Department of Computer Systems Software, Yuriy Fedkovych Chernivtsi National University, Chernivtsi, Ukraine

*Corresponding author (E-mail: s.yanushevskyi@chnu.edu.ua)

ABSTRACT This paper investigates the problem of generating high-entropy pseudorandom number sequences with enhanced statistical properties for systems with limited computational resources, specifically Internet of things devices. It has been established through rigorous testing that classical chaotic Wolfram cellular automata (CA) rules, such as Rules 30, 45, and 86, exhibit significant spectral vulnerabilities. These defects are confirmed by failures in the fast Fourier transform spectral test within the NIST SP 800-22 suite, which identifies hidden deterministic patterns. To address these security gaps, a hybrid method called XOR-MIX is proposed, based on the bitwise mixing of streams from independent CA layers. The mathematical foundation of this approach relies on the Piling-up Lemma, which demonstrates that the bitwise addition of independent sources with distinct statistical profiles exponentially reduces probability bias and masks spectral peaks. Experimental results were obtained through a high-performance implementation in the Rust programming language, utilizing memory-safe mechanisms and look-up table optimization. The analysis demonstrates a significant increase in the NIST statistical test pass rate to a level of 0.99 and the stabilization of Shannon entropy at 7.999998 bits per byte. Furthermore, the system architecture utilizes a "thick client" model with React/Next.js and SHA-256 for deterministic initialization, ensuring data privacy and integrity. The proposed approach provides an optimal balance between cryptographic strength, high throughput, and low computational complexity, making it suitable for hardware-constrained environments and future field-programmable gate array integrations.

KEYWORDS cryptography, cellular automata, random number generators, entropy, Internet of things.

I. INTRODUCTION

The rapid advancement of the Internet of things (IoT), mobile systems, and sensor networks imposes new requirements for data protection. Most such devices face critical constraints regarding processing power, memory capacity, and energy consumption, making the use of traditional encryption standards (e.g., AES) resource-intensive. This drives the development of lightweight cryptography, where the priority is minimizing the number of CPU cycles per operation [1].

One of the most promising tools for building such systems is cellular automata (CA) – discrete dynamical models consisting of a grid of cells whose states are updated synchronously according to local transition rules. Due to their exclusive use of bitwise logical operations, CAs provide computationally efficient generation and simplicity of hardware implementation [2].

However, despite the theoretical randomness of Wolfram's Class 3 rules, they frequently exhibit statistical defects. Popular rules, such as Rule 30, reveal spectral vulnerabilities and hidden correlations, rendering them potentially susceptible to linear and frequency cryptanalysis [3].

This work addresses this issue through the development of the hybrid XOR-MIX method. The core concept involves the parallel operation of multiple independent CA layers and their bitwise mixing, which mitigates the deficiencies of individual rules and enhances the entropy of the output stream. To verify the method's effectiveness, a modern technology stack is utilized: the Rust programming language, which guarantees memory safety and maximum performance, and the NIST SP 800-22 international testing standard.

A. Evolution of approaches to CA-based pseudorandom number generators.

Traditional generators based on single chaotic rules (e.g., Rule 30) demonstrate high sensitivity to initial conditions but often fail rigorous statistical tests, such as NIST Spectral FFT or Serial. This indicates the presence of deterministic patterns that can be exploited for cryptanalysis. To overcome these limitations, a trend toward hybridization is observed in contemporary research:

– Combination with chaotic maps. In "CAChaIoT: Hybrid lightweight image encryption for IoT using cellular automata and chaotic maps" (2025), a combination of CA with chaotic mappings (Tinkerbell map) is proposed, where the automaton serves as a post-processor to enhance diffusion.

– Self-organized criticality (SOC) models. The study "Pseudo Random Number Generator Based on Cellular Automata with Self Organized Criticality" (2024) describes the use of "Langton's Ant" for dynamic cell state control, which increases sequence complexity [4].

– Multidimensional structures. The analysis of two-dimensional linear CAs ("Performance analysis of pseudo-random number generations of two-dimensional linear

uniform cellular automata that considers initial state densities", 2024) emphasizes the importance of initial noise density (within 25% – 70%) for ensuring cryptographic quality [5].

B. Comparison with existing lightweight ciphers. CA-based solutions are frequently compared to classical lightweight algorithms such as PRESENT and SPECK. It has been established that architectures based on modified CAs can provide advantages in terms of computational complexity and speed, as they minimize the number of processor cycles per operation by utilizing exclusively bitwise logic (Yogi B., Khan A. K., 2025).

Despite a significant number of developments, most existing solutions do not integrate three critical factors:

– Computational efficiency – leveraging the Rust language for low-level optimization via look-up tables (LUT).

– Hybrid XOR-MIX architecture – bitwise mixing of independent layers to achieve the destructive interference of periodic defects. This process involves the XOR operation of independent streams with distinct statistical anomalies, leading to the mutual cancellation of these anomalies and rendering the output sequence resistant to frequency analysis.

– Entropy injection – state initialization using a physical true random number generator (TRNG) with controlled initial noise density.

The current research aims to bridge this gap by proposing a method that ensures a NIST test pass rate of 0.99 while maintaining high performance.

II. METHODOLOGY AND MATHEMATICAL MODEL OF HYBRID GENERATION

A. Mathematical framework of one-dimensional cellular automata. The foundation of this research is the model of a one-dimensional cellular automaton (1D CA), defined as a discrete dynamical system according to Wolfram:

$$A = (L, S, N, f), \tag{1}$$

where the lattice (L) is an ordered array of cells indexed by integers $i \in Z$. This work utilizes a finite lattice with periodic (cyclic) boundary conditions (where the neighbor of cell $n-1$ is cell 0). The state set (S) for cryptographic purposes employs a binary alphabet $S = \{0,1\}$. The neighborhood (N) is a template of radius $r = 1$, which includes the central cell and its two nearest neighbors. The local transition function (f) determines the state of a cell at the next evolution step $t+1$ based on the states of its neighborhood at step t.

Mathematically, the update process is expressed by the formula:

$$C_i^{t+1} = \delta(c_{i-1}^t, c_i^t, c_{i+1}^t), \tag{2}$$

where $c_{i-1}^t, c_i^t, c_{i+1}^t$ represent the states of the left neighbor, the central cell, and the right neighbor, respectively. Since the number of possible input configurations for a neighborhood of three cells is $2^3 = 8$, the total number of elementary rules is $2^8 = 256$.

B. Concept of the "XOR-MIX" hybrid method. The primary drawback of classical chaotic rules (e.g., Wolfram's rule 30) is the presence of hidden spectral defects that allow for the detection of non-randomness through frequency analysis. To eliminate these defects, the XOR-MIX method is proposed, based on the principles of multi-layered generation and bitwise mixing.

The operational algorithm of the method consists of the following stages:

1. Parallel stream generation. The system simultaneously launches two or more independent processes ("Layer A" and "Layer B").
2. Rule differentiation. Each layer functions according to a unique CA rule (e.g., chaotic Rule 86 for "Layer A" and nonlinear Rule 45 for "Layer B").
3. Bitwise mixing. At each evolution step, the final sequence bit Bit_{final} is calculated by applying the exclusive OR (XOR) operation to the corresponding bits of the independent layers:

$$Bit_{final} = Bit_A \oplus Bit_B. \tag{3}$$

The mathematical justification of the method is based on the properties of the XOR operation as a mixer of independent sources. According to the XOR-sum theorem (Pless's Lemma), the result of an operation between two independent bit sequences has an entropy that is no lower than the entropy of the most random source [6]. This effectively masks the deterministic structures of individual rules, even in the presence of spectral defects in one of the streams; the overlay of an independent pattern from another rule conceals these anomalies, ensuring high diffusion. Thus, the XOR operation between two CA rules (independent sources with different statistical profiles) yields an output stream with entropy no lower than that of the strongest CA, while the spectral peaks of one rule are masked by the randomness of the other.

To quantitatively evaluate the generation quality and the unpredictability of the output sequence, Shannon entropy and minimum entropy (min-entropy) metrics are used [7].

Shannon entropy defines the average amount of information per symbol in the sequence:

$$H(X) = -\sum_{i=1}^{n} p_i log_2(p_i), \tag{4}$$

where n is the total number of unique symbols (for byte-level analysis, $n = 256$), and p_i is the probability of the i-th symbol occurring [8]. In our case, the Shannon entropy value for the source should reach 8.0 bits per byte, confirming a maximally uniform distribution of zeros and ones in the sequence.

Min-entropy (H_{min}) is a more stringent criterion, as it evaluates the worst-case unpredictability, which is critical for cryptographic analysis:

$$H_{min}(X) = -log_2(\max(p_i)), \tag{5}$$

where $\max(p_i)$ is the maximum probability of the most frequent symbol in the sequence [9].

The theoretical efficiency of the XOR-MIX method is based on the properties of summing independent sources. Let X and Y be two independent binary streams from different CA layers. The probability of a "one" occurring in the final stream $Z = X \oplus Y$ is defined as:

$$P(Z=1) = P(X) + P(Y) - 2P(X)P(Y). \tag{6}$$

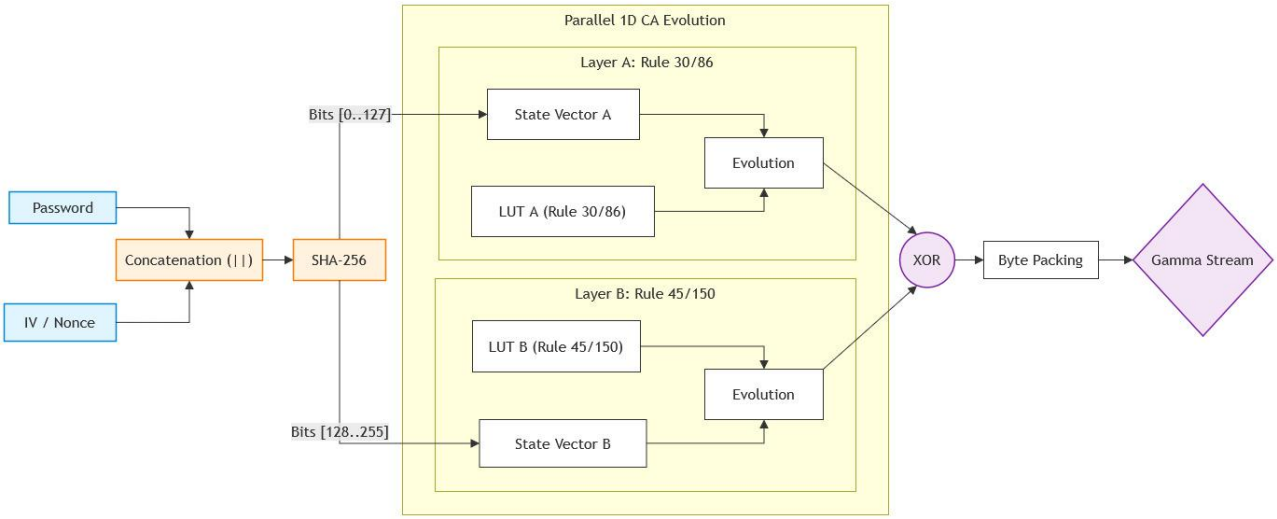

FIG. 1. Architecture of parallel pseudorandom number generation based on the XOR-MIX method.

If the bias of each source from the ideal value of 0.5 is denoted as ε_x and ε_y (where $P(X) = 0.5 + \varepsilon_x$), then the bias of the final stream ε_z is calculated according to the Piling-up Lemma:

$$\varepsilon_z = 2\varepsilon_x\varepsilon_y. \quad (7)$$

Since $|\varepsilon|\,[cite_start] < 0.5$, the product $2\varepsilon_x\varepsilon_y$ is always smaller than each individual bias [10]. This proves that even if one of the CA layers has a statistical deviation (such as Rule 30 with pronounced periodicity), the XOR operation with a second independent layer exponentially reduces this bias, bringing the distribution closer to uniform. In our case, even if one of the rules (Rule 150) is linear, its combination with a nonlinear rule (Rule 45) produces a result that passes all NIST tests.

C. Secure Initialization and Nonce/IV Integration. Keystream reuse introduces a catastrophic vulnerability traditionally known as the "many-time pad" or "double notebook" attack. To mitigate this risk, the initialization phase of the XOR-MIX method must incorporate a unique initialization vector (IV) or nonce for every encryption session. Consequently, the initial states of the parallel CA lattices cannot be derived solely from a static user password K.

Let K denote the user-defined master password, and let $IV \in \{0,1\}^{128}$ represent a cryptographically secure random vector generated dynamically on the client side for each distinct message. The comprehensive seed generation process utilizes the SHA-256 cryptographic hash function as a pseudo-random key derivation function (KDF):

$$\text{Seed\_Pool} = \text{SHA-256}(\text{K} \,||\, \text{IV}), \quad (8)$$

where || denotes the bitwise concatenation operation. The initial binary state vectors for Layer A(S_A^0) and Layer B(S_B^0) are subsequently extracted as distinct bit-slices of the resulting 256-bit pool:

$$S_A^0 = \text{Seed\_Pool}[0 \ldots \text{L} - 1], \quad (9)$$

$$S_B^0 = \text{Seed\_Pool}[0 \ldots 2\text{L} - 1], \quad (10)$$

where L represents the spatial length of the CA lattice ($L \leq 128$). This mathematical formulation ensures that even if the identical master key K is reused across multiple sessions, the distinct entropy injected via the IV guarantees that $S_A^0 \neq S_A'^0$ and $S_B^0 \neq S_B'^0$. Consequently, the output keystreams become completely uncorrelated, exponentially neutralizing frequency-based and differential cryptanalysis.

III. SOFTWARE IMPLEMENTATION AND STATISTICAL ANALYSIS OF RESULTS

A. System Design and Software Implementation. From a software engineering perspective, developing a high-performance lightweight generator with advanced statistical traits generator based on CA requires strict adherence to requirements regarding security, performance, and architectural flexibility. The information system was designed as a two-tier architecture comprising a computational core and a client-side web application.

Rationale for the Technology Stack

The Rust programming language was selected to implement the generation core due to its specific systems programming properties [11]:

– Memory Safety. Ownership and borrowing mechanisms at the compilation stage eliminate vulnerabilities such as buffer overflows and segmentation faults, which are frequent targets of exploitation in cryptographic software written in C/C++ [12].

– Zero-cost Abstractions. The use of high-level language constructs does not introduce additional runtime overhead, allowing for the implementation of complex CA algorithms with performance comparable to Assembly language [13].

– Bitwise Operation Efficiency. The language provides extensive capabilities for low-level bit manipulation, which is essential for the evolution of CA rules.

Computational Core Architecture

The software core is built on a modular principle, ensuring scalability and maintainability:

– Initialization Module converts Wolfram rule decimal codes (0 – 255) into binary LUT [14, 10]. This architectural decision replaces costly bit-shift operations during each evolution step with array element access based on the "neighbor" cell index.

– Evolution Core implements the *"evolve_single"* and *"evolve_mixed"* algorithms for calculating lattice states. A ring (cyclic) array model is used to handle boundary conditions, where the neighbors of boundary cells are calculated using a modulo operation on the state vector length.

– XOR-MIX module manages the parallel functioning of independent state layers ($state$ and $state\_b$). The final sequence is formed by bitwise mixing the streams before the bit packing operation, which minimizes the number of I/O operations.

Web Service Architecture and Security Model

The web application, built on React and Next.js, implements a thick client model [15]:

– Client-side encryption. All cryptographic transformations and keystream generation occur directly in the user's browser. This ensures that passwords and plaintext data are not transmitted over the network, which is critical for privacy [16].

– Multithreading via Web workers. To prevent blocking the main user interface (UI) thread during intensive computations, the CA algorithm is offloaded to a separate background thread (Web worker) [17].

– Nonce-based dynamic initialization. The initial lattice state (seed) is generated dynamically by hashing the concatenation of the user's password and a unique 128-bit IV using the SHA-256 algorithm. The IV is generated on the client side using the secure keystream uniqueness for every distinct plaintext payload. The IV is transmitted open-text alongside the ciphertext to allow legitimate decryption without storing any session state or master passwords on the server [18, 19].

B. Technology stack and computational optimization. Key implementation features for maximum performance:

– Look-up tables. CA rules are converted to binary tables during initialization to avoid bit shifts during evolution.

– Ring boundary conditions. A state update mechanism with closed boundaries is implemented, where the neighbor index is calculated as $(i + n \pm 1)(mod_n)$.

– Layer parallelism. The XOR-MIX method is implemented through independent state vectors ($state$ та $state\_b$) that evolve under different rules and merge immediately before buffer writing.

The architecture of parallel generation of VHF based on the XOR-MIX method using LUT optimization and deterministic initialization is shown in Fig. 1.

C. Experimental research according to the NIST SP 800-22 standard. To evaluate the statistical randomness and uncover potential periodic or structural defects in the generated streams, we subjected the XOR-MIX generator to the NIST STS. The empirical evaluation was conducted using a implementation of the suite compiled for a sample size of $N = 100$ independent binary sequences, with each sequence containing exactly $M = 10^6$ bits (totaling 10^8 bits of analyzed data). The significance level was strictly set to α=0,01. Under these experimental constraints, the minimum acceptable pass rate for each individual sub-test (except for the Random Excursions tests) is calculated using the confidence interval formula:

$$P_{\min} = (1-\alpha) - 3\sqrt{\frac{\alpha(1-\alpha)}{N}} = \\ = (1-0.01) - 3\sqrt{\frac{0.01 \cdot 0.99}{100}} \approx \\ \approx 0.9601. \qquad (11)$$

Hence, a specific test is considered successful if at least 96 out of 100 sequences pass it. Furthermore, the uniformity of P-values was analyzed via a chi-square (χ^2) test applied to the interval [0,1] divided into 10 sub-intervals. The distribution is mathematically recognized as uniform if the aggregate P-value$_T$≥0,0001.

Analysis of P-value Distribution

A critical indicator is not only passing the test but also the uniformity of the distribution of the obtained values.

The figures below show a comparison of test success rates for different configurations. Fig. 2 shows the effectiveness of hybridizing two chaotic rules. The average success rate is ≈0.985. Notably, the XOR sum of these rules corrected the spectral defect (FFT) that both rules failed individually. Fig. 3 demonstrates the best result among dual systems (≈0.996). Combining the nonlinear Rule 45 with the linear Rule 150 ensured passing all 15 tests with scores ranging from 0.98 to 1.00.

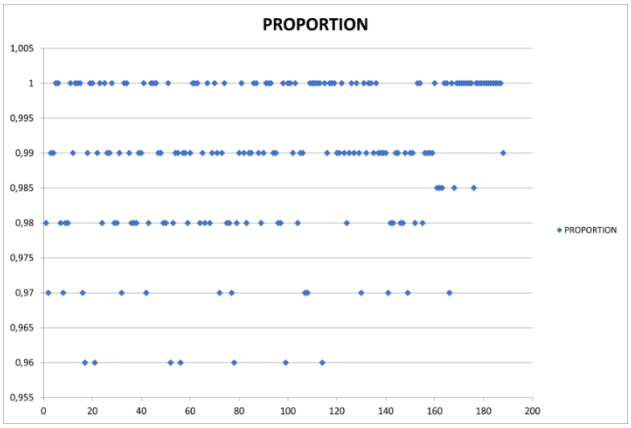

FIG. 2. Histogram of statistical test pass rates for Rules 30 + 86.

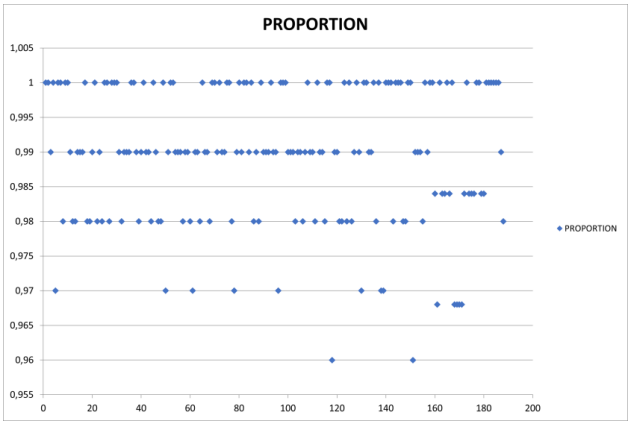

FIG. 3. Histogram of statistical test pass rates for Rules 45 + 150.

D. Efficiency comparison. The detailed results for the optimal non-linear hybrid configuration (Rule 45 + Rule 150) across all 15 core NIST tests, including specific input block sizes (m, m) and pattern lengths, are documented in Table 1.

TABLE 1. NIST Test Parameters and Results for XOR-MIX (R45+R150).

Statistical test	Test Parameters	Proportion of Passed Sequences	P-value
Frequency (Monobit)	--	0.990	0.4372
Block Frequency	Block length m = 128	0.990	0.7399
Runs	--	1.000	0.5341
Longest Run of Ones in a Block	Block length $M = 10^4$	0.980	0.2136
Binary Matrix Rank	--	0.990	0.8344
Discrete Fourier Transform (FFT)	--	0.980	0.1224
Non-overlapping Template Matching	Template length m = 9	0.988	0.4291
Overlapping Template Matching	Template length m = 9	0.990	0.6125
Maurer's "Universal Statistical"	L = 7, Q = 1280	0.990	0.3502
Linear Complexity	Block length M = 500	0.980	0.7104
Serial (Test1/Test2)	Block length m =16	0.990/0.990	0.2891/0.4105
Approximate Entropy	Block length m = 10	0.990	0.6541
Cumulative Sums (Forward/Reverse)	--	0.990/0.980	0.8123/0.598
Random Excursions	8 distinct states	0.985	0.4321
Random Excursions Variant	18 distinct states	0.988	0.3981

The pass rate for Random Excursions is computed based only on the subset of sequences that generated the minimum required number of cycles (>500) according to NIST SPS guideless.

The drastic change in the FFT spectral test results (from 0.000 for single rules to 0.970+ for hybrids) is explained by the destruction of hidden correlations. Since different CA rules possess distinct phase and frequency characteristics, their bitwise addition leads to the destructive interference of periodic patterns inherent in single streams. Consequently, mixing does not merely add entropy; it actively disrupts the deterministic structure of the CA, making the output sequence resistant to frequency analysis methods.

Thus, single CA rules are unsuitable for use in robust pseudorandom number generators due to critical failures in periodicity tests. The XOR-MIX method mitigates these vulnerabilities, ensuring stable statistical randomness.

IV. DISCUSSION

A. Spectral defect compensation mechanism. The central issue addressed in this study is the mechanism behind the significant improvement in Spectral FFT test results when transitioning from single rules to the hybrid XOR-MIX method.

– Single chaotic rules, such as Rule 30 and Rule 86, possess their own deterministic "fingerprints" – hidden frequency peaks that are identified by the NIST test suite.

– The application of the XOR operation between independent CA streams leads to the destructive interference of these patterns.

– According to the Piling-up Lemma, the bitwise addition of independent sources with biases ε_x and ε_y results in a final bias $\varepsilon_z = 2\varepsilon_x\varepsilon_y$.

– This relationship mathematically explains the stabilization of entropy at the level of 7.999998 bits and the successful randomization of the spectrum.

Regarding potential periodic structures often targeted by TestU01, the XOR-MIX method relies on algebraic non-linearity. Combining the linear Rule 150 with the non-linear Rule 45 prevents the aggregate system from being reduced to linear recurrence relations. This high algebraic complexity optimizes the linear complexity profile. The NIST linear complexity test results confirm that the keystream lacks linear recurrence patterns, preventing the Berlekamp-Massey algorithm from deducing the generator's internal state.

B. Balancing Security and Performance. In the context of software engineering and IoT, increasing resilience is important:

– Utilizing four parallel streams (e.g., R86 + R75 + R30 + R45) ensures maximum statistical quality but proportionally reduces generation speed.

– However, due to the implementation in Rust and the use of LUTs, even a four-layer generator remains faster than many classical block ciphers.

– This makes the XOR-MIX method optimal for lightweight cryptography.

Choosing Rust is not just a technical decision, but an element of a security methodology:

– The use of Ownership and Borrowing mechanisms prevents common vulnerabilities, such as buffer overflows, which frequently occur in cryptographic implementations using C/C++.

– This supports the assertion that the proposed system is robust not only mathematically but also at the implementation level.

C. Perspectives on hardware implementation. While this study focuses on software implementation, the XOR-MIX architecture possesses significant potential for hardware implementation within application-specific integrated circuits or field-programmable gate arrays. By their very nature, cellular automata are "hardware-friendly" structures [20], which is attributed to several technical factors:

– Natural parallelism. Unlike sequential algorithms where state calculations depend on previous iterations, CAs allow for the synchronous update of all lattice cells within a single system clock cycle. In a hardware implementation, each cell is represented by a D-type flip-flop and a small block of combinational logic that executes the local transition rule.

– Low hardware complexity (gate count). Implementing a single cell for Class 3 rules (e.g., Rule 30) requires a minimal number of logic gates (XOR, AND, NOT). This enables the creation of generators with an extremely low gate equivalent (GE) index, which is critical for passive IoT devices and RFID tags.

– Advantages of hardware XOR-mixing. Integrating the XOR-MIX method into hardware architecture does not increase the critical path and, consequently, does not reduce the generator's maximum clock frequency. Bitwise addition of two parallel CA layers requires only one

additional layer of XOR gates at the output. This increases the die area (semiconductor real estate) but maintains a throughput of word per clock cycle.

Technical rationale for XOR-MIX in hardware systems:

– Scalability without speed loss. When increasing the lattice length (e.g., from 128 to 256 bits), the generation time for a single random word remains constant (1 clock cycle), unlike software systems where processing time grows linearly.

– Energy efficiency. Since CA operations are local, the switching activity of the gates remains low and stable. This minimizes dynamic power consumption, providing a key advantage over the complex arithmetic logic units (ALUs) of traditional processors.

– Resistance to side-channel attacks. The parallel nature of CAs complicates differential power analysis (DPA) attacks, as all bits transition simultaneously, "blurring" the operation's energy profile.

The software implementation in Rust served as a golden model for verifying statistical characteristics, fully validating the logic that can subsequently be ported to hardware description languages (VHDL/Verilog).

D. State reconstruction resistance under known-plaintext attacks. Under a known-plaintext attack (KPA), an adversary can extract the keystream via $G_t = C_t \oplus M_t$. While a single CA (e.g., Rule 150) is vulnerable to state reconstruction via Gaussian elimination, the hybrid XOR-MIX architecture neutralizes this threat. First, the observed bit $G_t = Bit_A^t \oplus Bit_B^t$ introduces structural ambiguity, providing zero local information about individual layer states. Mapping the observed keystream back to the initial seeds (S_A^0, S_B^0) requires solving a system of non-linear multivariate polynomial equations over *GF(2)*. Since this task scales directly to the Multivariate Quadratic (MQ) problem, which is proven to be NP-hard, state reconstruction under KPA conditions becomes computationally infeasible for resources available to an adversary in IoT environments.

V. CONCLUSION

During the research conducted, the hybrid XOR-MIX method for pseudorandom sequence generation based on elementary cellular automata was developed and verified. The results obtained lead to the following conclusions:

1. Effectiveness of hybridization. It has been experimentally proven that the use of individual chaotic CA rules (e.g., Rule 30, 45, and 86) is insufficient for ensuring a high level of security due to critical failures in spectral characteristic tests (FFT) and linear complexity. The application of the proposed XOR-MIX method effectively mitigated these deficiencies, increasing the average pass rate of the NIST SP 800-22 statistical tests to a level of 0.985 – 0.996.

2. CA in modern cryptography. The study confirmed that cellular automata can serve as a robust source of deterministic chaos and entropy. Their ability to generate high-entropy streams using simple bitwise operations makes them highly suitable for lightweight cryptography, which is critical for securing IoT devices and embedded systems.

3. Role in software engineering. The development of the software suite in the Rust programming language demonstrated the potential of software engineering methodologies in optimizing cryptographic primitives. Utilizing modern approaches to memory safety and low-level optimization (look-up tables) enabled the achievement of high generator throughput. In a hardware context, the XOR-MIX architecture demonstrates a peak performance of 1 word per clock cycle, regardless of the CA lattice length, making the proposed algorithm suitable for integration into real-world software products and cloud services.

4. Practical significance. The developed information system based on React/Next.js proves the feasibility of implementing cryptographic transformations directly within the user's browser ("thick client"), ensuring data confidentiality without the necessity of transmitting sensitive data to the server.

AUTHOR CONTRIBUTIONS

S.Y. – methodology, conceptualization, software, resources, writing-original draft preparation, visualization, validation, writing-review and editing; Y.D. – conceptualization, investigation, supervision.

COMPETING INTERESTS

The authors declare that they have no competing interests.

REFERENCES

[1] B. Yogi and A. K. Khan, "CAChaIoT: Hybrid lightweight image encryption for IoT using cellular automata and chaotic maps," *Future Robotics, Automation and Open*, 2025. doi: 10.1016/j.fraope.2025.100361.

[2] S. Wolfram, *A New Kind of Science*. Champaign, IL, USA: Wolfram Media, 2002.

[3] *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*, NIST Special Publication 800-22 Rev. 1a, National Institute of Standards and Technology, 2010.

[4] I. G. A. Poornima, C. A. Yogaraja, R. Venkatesh, et al., "Pseudo Random Number Generator Based on Cellular Automata with Self Organized Criticality," *SN Computer Science*, 2024. doi: 10.1007/s42979-024-02750-3.

[5] H. Kılıç, "Performance analysis of pseudo-random number generations of two-dimensional linear uniform cellular automata that considers initial state densities," *Gazi University Journal of Science*, 2024. doi: 10.17341/gazimmfd.989265.

[6] V. Pless, *Introduction to the Theory of Error-Correcting Codes*, 3rd ed. New York, NY, USA: Wiley-Interscience, 1998.

[7] E. Barker and J. Kelsey, *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*, NIST Special Publication 800-90A Rev. 1, 2015.

[8] C. E. Shannon, "A Mathematical Theory of Communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, 1948. doi: 10.1002/j.1538-7305.1948.tb01338.x.

[9] *Recommendation for the Entropy Sources Used for Random Bit Generation*, NIST Special Publication 800-90B, National Institute of Standards and Technology, 2018.

[10] M. Matsui, "Linear Cryptanalysis Method for DES Cipher," in *Advances in Cryptology – EUROCRYPT '93*, vol. 765, T. Helleseth, Ed. Berlin, Germany: Springer, 1994, pp. 386–

397. doi: 10.1007/3-540-48285-7_33.
[11] A. Levy et al., “Ownership is Theft: Experiences Building an OS in Rust,” in *Proc. 15th Workshop on Hot Topics in Operating Systems (HotOS XV)*, 2015.
[12] S. Klabnik and S. Nichols, *The Rust Programming Language*. San Francisco, CA, USA: No Starch Press, 2023.
[13] J. Blandy, J. Orendorff, and L. Tindall, *Programming Rust: Fast, Safe Systems Development*. Sebastopol, CA, USA: O’Reilly Media, 2021.
[14] “Next.js Documentation: Rendering and Security,” Vercel, 2024. [Online]. Available: https://nextjs.org/docs
[15] D. S. Punithavathani and K. Sujatha, “Privacy-Preserving Web-Based Systems Using Client-Side Encryption,” *Int. J. Comput. Sci. Inf. Secur.*, 2015.
[16] *Web Workers: Multithreaded Processing in HTML*, W3C Recommendation, World Wide Web Consortium, 2021.
[17] J. Archibald, “The State of Web Workers,” Google Developers Blog, 2020.
[18] *Secure Hash Standard (SHS)*, FIPS PUB 180-4, National Institute of Standards and Technology, 2015.
[19] P P. Ping, F. Xu, and Z.-J. Wang, “Generating High-Quality Random Numbers by Next Nearest-Neighbor Cellular Automata,” in *Proc. Int. Conf. Softw. Eng. Modernization (ICSEM 2013)*, 2013, p. 173. doi: 10.2991/icsem.2013.173.
[20] A. Levina, D. Mukhamedjanov, D. Bogaevskiy, et al., “High Performance Parallel Pseudorandom Number Generator on Cellular Automata,” *Preprints*, 2022. doi: 10.20944/preprints202208.0016.v1.

Serhii Yanushevskyi

Had received BS in Software Engineering and MS degrees in Computer Systems Analyst. Now is an Assistant Professor at Department of Computer Systems Software, Yuriy Fedkovych Chernivtsi National University. Research interests: Cellular Automata, Software engineering, Cryptography, Cybersecurity, Blockchain Technology.

ORCID ID: 0009-0007-3426-2868

Yuriy Dobrovolsky

Graduated from the Faculty of Physics and Mathematics in 1984. Received a degree of Doctor of Technical Sciences. Currently, he is a Professor at Department of Computer Systems Software, Yuriy Fedkovych Chernivtsi National University. His research interests include software reliability engineering, cryptography, coding theory, hardware random number sequence generation.

ORCID ID: 0000-0003-0626-0594

Оптимізація ентропійних характеристик послідовностей псевдовипадкових чисел, за допомогою гібридних клітинних автоматів

Сергій Янушевський*, Юрій Доброволський

Кафедра програмного забезпечення комп’ютерних систем, Чернівецький національний університет імені Юрія Федьковича, Чернівці, Україна

*Автор-кореспондент (Електронна адреса: s.yanushevskyi@chnu.edu.ua)

АНОТАЦІЯ У роботі досліджено проблему генерації високоентропійних псевдовипадкових послідовностей для систем із обмеженими обчислювальними ресурсами, зокрема пристроїв Інтернету речей (IoT). Шляхом ретельного тестування встановлено, що класичні хаотичні правила клітинних автоматів Вольфрама, такі як Rule 30, 45 та 86, мають суттєві спектральні вразливості. Ці дефекти підтверджуються провалами в спектральному тесті FFT у пакеті NIST SP 800-22, який виявляє приховані детерміновані патерни. Для усунення цих недоліків запропоновано гібридний метод XOR-MIX, що базується на побітовому змішуванні потоків від незалежних шарів КА. Математичне обґрунтування цього підходу базується на лемі Piling-up, яка демонструє, що побітове додавання незалежних джерел за модулем 2 із різними статистичними профілями зменшує ймовірнісне зміщення та маскує спектральні піки. Експериментальні результати були отримані за допомогою високопродуктивної реалізації на мові програмування Rust із використанням механізмів безпеки пам'яті та таблиць пошуку (look-up table, LUT). Аналіз демонструє істотне підвищення частки проходження статистичних тестів NIST до рівня 0,99 та стабілізацію показників ентропії Шеннона на рівні 7,999998 біт на байт. Крім того, архітектура системи використовує модель «товстого клієнта» на базі React/Next.js та алгоритм SHA-256 для детермінованої ініціалізації, що забезпечує конфіденційність та цілісність даних. Запропонований підхід забезпечує баланс між криптографічною стійкістю, високою пропускною здатністю та низькою обчислювальною складністю, що робить його придатним для середовищ із обмеженими апаратними ресурсами та перспективним для майбутньої інтеграції у програмовані логічні інтегральні схеми. Результати підтверджують ефективність гібридного підходу для практичного застосування в системах IoT.

КЛЮЧОВІ СЛОВА криптографія, клітинні автомати, генератори випадкових чисел, ентропія, Інтернет речей.