UDC 621.391.253

Olena B. Akhiiezer
Valerii I. Churilov
Maksym I. Glavchev

STATISTICAL QUALITY ANALYSIS OF PSEUDO-RANDOM SEQUENCE GENERATORS IN STREAM CIPHERS: PERFORMANCE BENCHMARKING AND PRACTICAL RECOMMENDATIONS

National Technical University "Kharkiv Polytechnic Institute", Kharkiv

Анотація. *У роботі проведено комплексний порівняльний аналіз потокових шифрів AES-CTR, ChaCha20, Trivium та RC4 за критеріями статистичної якості, продуктивності та криптографічної стійкості. Для оцінки статистичних властивостей псевдовипадкових послідовностей застосовано стандартизований набір тестів NIST SP 800-22 (15 тестів). Проведено багатокритеріальну оцінку алгоритмів та сформульовано практичні рекомендації щодо їх вибору для різних сценаріїв застосування.*

Ключові слова: потоковий шифр, псевдовипадкова послідовність, NIST SP 800-22, AES-CTR, ChaCha20, Trivium, RC4, криптографічна стійкість, продуктивність.

Abstract. *This paper presents a comprehensive comparative analysis of AES-CTR, ChaCha20, Trivium, and RC4 stream ciphers evaluated across statistical quality, performance, and cryptographic security criteria. The NIST SP 800-22 statistical test suite (15 tests applied to 100 sequences of 10^6 bits each) was employed to assess the randomness properties of the generated pseudo-random sequences. A multi-criteria weighted evaluation framework was developed, and practical recommendations are formulated for algorithm selection across different application scenarios.*

Keywords: stream cipher, pseudo-random sequence, NIST SP 800-22, AES-CTR, ChaCha20, Trivium, RC4, cryptographic security, performance benchmarking.

Introduction

Ensuring the confidentiality and integrity of data transmitted over open communication channels is a fundamental challenge of modern information security. Stream ciphers, owing to their high throughput and implementation simplicity, are extensively deployed in real-time protocols such as TLS 1.3, Wi-Fi (WPA/WPA2), and mobile communication standards. However, the cryptographic landscape is rapidly evolving: algorithms once considered secure may be compromised by advances in cryptanalysis or computational power.

Four representative ciphers spanning different generations and design philosophies are examined: AES-256-CTR — a NIST-standardized block cipher operating in counter mode (FIPS 197, 2001); ChaCha20 — a modern dedicated stream cipher adopted as a TLS 1.3 standard (RFC 8439, 2018); Trivium — a lightweight hardware-oriented cipher from the eSTREAM portfolio (ISO/IEC 29192-3:2012); and RC4 — a legacy cipher (1987) officially deprecated due to serious statistical and cryptographic weaknesses.

The absence of objective, reproducible comparative data on these algorithms leads to suboptimal engineering decisions — such as continued reliance on RC4 in legacy systems, or over-engineering solutions for resource-constrained IoT devices. The goal of this work is to conduct a rigorous multi-criteria analysis of these ciphers and to provide evidence-based practical recommendations.

Methodology

The evaluation framework combines five criteria: (1) throughput — pseudo-random sequence generation speed (MB/s); (2) statistical quality — NIST SP 800-22 test suite results; (3) cryptographic security — known attacks and key size; (4) standardization — presence in international standards (NIST FIPS, RFC, ISO/IEC); and (5) hardware efficiency — implementation complexity in gate equivalents (GE).

Throughput measurements were performed on an Intel Core i7-10700 platform (8 cores, 3.8 GHz) with AES-NI hardware acceleration. Data blocks of 1 KB to 10 MB were tested with 1000 iterations averaged per measurement. Software implementations used PyCryptodome 3.19 (AES, RC4), the cryptography library

v41.0 (ChaCha20), and a custom Python implementation for Trivium. All measurements were conducted single-threaded to isolate algorithmic characteristics.

Statistical testing applied all 15 NIST SP 800-22 tests to 100 bit sequences of length 10^6 each, with significance level $\alpha = 0.01$. The pass criterion required p-value ≥ 0.01 for individual tests and a pass rate within the 99% confidence interval [0.96, 1.00] for the proportion of passing sequences [1].

Results

Performance comparison results are presented in Table 1.

Table 1 – Performance and implementation characteristics of stream ciphers

Algorithm	Throughput (MB/s)	Key size (bits)	Hardware (GE)	Standard
AES-256-CTR	4850 *	256	~22 000	FIPS 197
ChaCha20	2340	256	~18 000	RFC 8439
Trivium	156	80	~3 500 *	ISO/IEC 29192-3
RC4	312	128	~8 000	Deprecated

* AES-256-CTR throughput measured with AES-NI hardware acceleration; Trivium software throughput was 156 MB/s, while hardware implementations achieve up to ~10 Gbps. GE = Gate Equivalent.

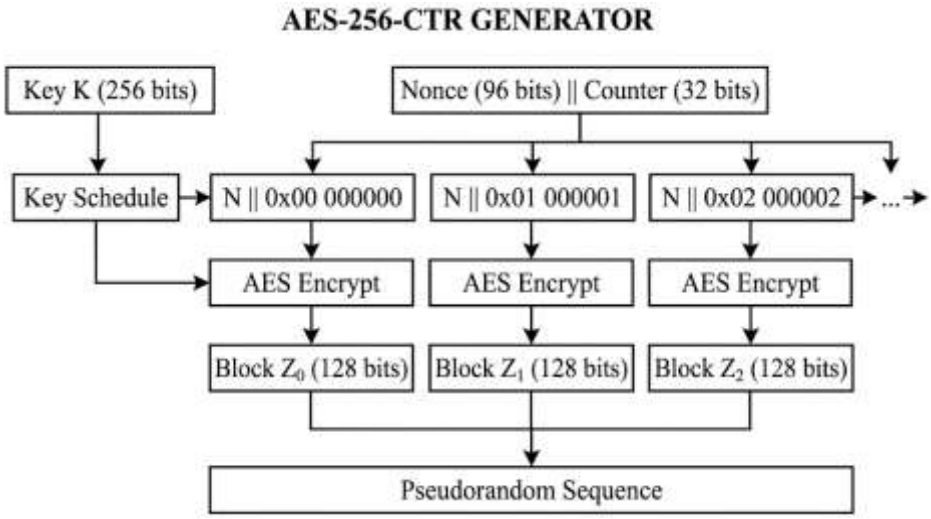


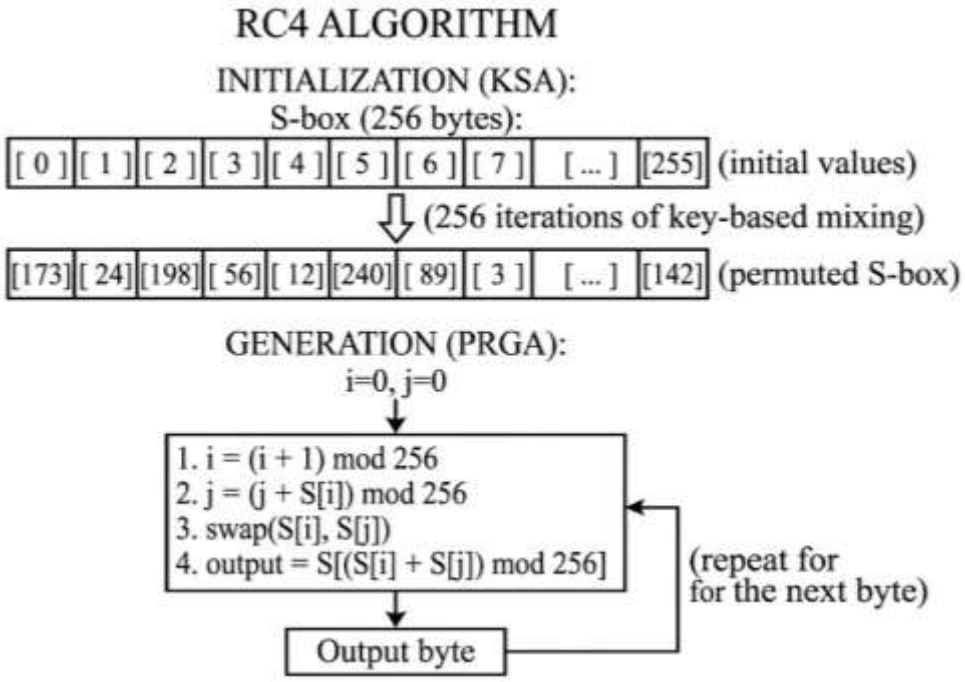
Fig. 1. Throughput comparison on logarithmic scale (bar chart: AES-256-CTR, ChaCha20, RC4, Trivium)

AES-256-CTR achieves the highest throughput (4850 MB/s) due to AES-NI hardware support available in modern Intel/AMD processors. ChaCha20 reaches 2340 MB/s — approximately 2.1× slower in this configuration, yet it is the fastest purely software implementation and requires no special CPU features, making it the preferred choice on platforms lacking AES-NI. RC4 and Trivium show significantly lower software throughput (312 and 156 MB/s, respectively), primarily due to the Python implementation overhead; hardware implementations of Trivium reach approximately 10 Gbps.

The results of NIST SP 800-22 statistical testing and the multi-criteria weighted evaluation are summarized in Table 2.

Table 2 – Multi-criteria weighted evaluation of stream ciphers

Criterion	AES-CTR	ChaCha20	Trivium	RC4	Weight
Throughput	10	8	4	5	30%
Cryptographic security	10	10	7	2	40%
Standardization	10	10	6	0	20%
NIST SP 800-22 pass rate	10	10	10	6	10%
Weighted Total	10.0	9.4	6.8	2.6	—



[Fig. 2. Radar chart: multi-criteria comparison of AES-CTR, ChaCha20, Trivium, RC4 (axes: Throughput, Security, Standardization, NIST Pass Rate)]

AES-256-CTR and ChaCha20 passed all 15 NIST SP 800-22 tests without exception. Trivium also demonstrated a full pass rate, confirming the adequacy of its keystream randomness despite its lightweight architecture. RC4 exhibited statistically significant deviations in several tests — most notably in frequency-based and serial tests — experimentally confirming the known predictability of its keystream, consistent with the Fluhrer-Mantin-Shamir [11] and RC4 NOMORE attacks.

Practical recommendations

Based on the experimental results and multi-criteria evaluation, the following recommendations are formulated (Table 3).

Table 3 – Algorithm selection recommendations by application scenario

Scenario	Recommended	Rationale
High-throughput servers (TLS, VPN, disk encryption)	AES-256-CTR	Peak throughput with AES-NI; FIPS 197 / NIST compliance required
Cross-platform apps / mobile without AES-NI	ChaCha20	Highest software throughput; resistant to timing attacks; TLS 1.3 standard (RFC 8439)
IoT / RFID / embedded hardware	Trivium	Minimal hardware footprint (~3 500 GE); hardware throughput up to 10 Gbps (ISO/IEC 29192-3)
Any new system (migration from legacy)	NOT RC4	Statistically broken; FMS [11], BEAST, RC4 NOMORE attacks; officially deprecated in RFC 7465

Conclusions

1. A rigorous multi-criteria comparative analysis of four stream cipher algorithms was conducted using five criteria with defined weights. AES-256-CTR achieved the highest weighted score (10.0), followed by ChaCha20 (9.4), Trivium (6.8), and RC4 (2.6).

2. AES-256-CTR demonstrated peak throughput (4850 MB/s) owing to AES-NI hardware acceleration, outperforming ChaCha20 by a factor of 2.1. On platforms without AES-NI, ChaCha20 is the optimal choice.

3. Trivium is recommended for embedded and IoT applications due to its exceptionally low hardware footprint (~3 500 GE vs. ~22 000 GE for AES), while its software throughput is limited.

4. RC4 must not be used in any new system design. The NIST SP 800-22 testing experimentally confirmed the presence of statistical anomalies in its keystream, consistent with known cryptanalytic attacks (FMS attack [11], RC4 NOMORE [12, 13]).

References

1. Rukhin A., Soto J., Nechvatal J. et al. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. NIST SP 800-22 Rev. 1a. Gaithersburg: NIST, 2010. 131 p.
2. NIST FIPS 197. Advanced Encryption Standard (AES). Gaithersburg: NIST, 2001. 51 p.
3. Dworkin M. Recommendation for Block Cipher Modes of Operation. NIST SP 800-38A. Gaithersburg: NIST, 2001. 66 p.
4. Nir Y., Langley A. ChaCha20 and Poly1305 for IETF Protocols. RFC 8439. IRTF, 2018. 46 p. URL: https://datatracker.ietf.org/doc/html/rfc8439
5. ISO/IEC 29192-3:2012. Information technology – Lightweight cryptography – Part 3: Stream ciphers. Geneva: ISO, 2012. 24 p.
6. Bernstein D. J. ChaCha, a variant of Salsa20. Workshop Record of SASC 2008. 2008. P. 273–278.
7. De Cannière C., Preneel B. Trivium. New Stream Cipher Designs: The eSTREAM Finalists. LNCS 4986. Berlin: Springer, 2008. P. 244–266. DOI: 10.1007/978-3-540-68351-3_18
8. De Cannière C. Trivium: A Stream Cipher Construction Inspired by Block Cipher Design Principles. Lecture Notes in Computer Science, vol. 4176. Berlin: Springer, 2006. P. 171–186.
9. Daemen J., Rijmen V. The Design of Rijndael: AES – The Advanced Encryption Standard. 2nd ed. Berlin: Springer, 2020. 272 p.
10. Barker E., Kelsey J. Recommendation for Random Number Generation Using Deterministic Random Bit Generators. NIST SP 800-90A Rev. 1. Gaithersburg: NIST, 2015. 110 p.
11. Fluhrer S., Mantin I., Shamir A. Weaknesses in the Key Scheduling Algorithm of RC4. Lecture Notes in Computer Science, vol. 2259. Berlin: Springer, 2001. P. 1–24.
12. Mantin I., Shamir A. A Practical Attack on Broadcast RC4. Lecture Notes in Computer Science, vol. 2355. Berlin: Springer, 2002. P. 152–164.
13. Klein A. Attacks on the RC4 stream cipher. Designs, Codes and Cryptography. 2008. Vol. 48, No. 3. P. 269–286.
14. Schneier B. Applied Cryptography: Protocols, Algorithms and Source Code in C. 2nd ed. New York: Wiley, 1996. 758 p.
15. Stallings W. Cryptography and Network Security: Principles and Practice. 7th ed. Boston: Pearson, 2017. 752 p.

Ахієзер Олена Борисівна - кандидат технічних наук, доцент, завідувач кафедри «Комп'ютерна математика і аналіз даних», Національний технічний університет «Харківський політехнічний інститут», м.Харків, e-mail:. Olena.Akhiiezer@khpi.edu.ua

Чурілов Валерій Ігорович - магістрант, група КН-Н124, факультет «Комп'ютерні науки та інформаційні технології», Національний технічний університет «Харківський політехнічний інститут», м.Харків, e-mail:. Valerii.Churilov@cs.khpi.edu.ua

Главчев Максим Ігорович - кандидат економічних наук, доцент, професор кафедри комп'ютерної інженерії та програмування, Національний технічний університет «Харківський політехнічний інститут», м.Харків, e-mail: Maksym.Glavchev@khpi.edu.ua.

Olena B. Akhiiezer - Candidate of Technical Sciences, Associate Professor, Head of the Department of Computer Mathematics and Data Analysis, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, e-mail: Olena.Akhiiezer@khpi.edu.ua

Valerii I. Churilov - Master's Student, Group KN-N124, Faculty of Computer Science and Information Technologies, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, e-mail: Valerii.Churilov@cs.khpi.edu.ua

Maksym I. Glavchev – Candidate of Economic Sciences (Ph.D.), Associate Professor, Professor at the Department of Computer Engineering and Programming, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, e-mail: Maksym.Glavchev@khpi.edu.ua.