The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 10, October 2025
PP 167–177

A Comprehensive Analysis of Stream Ciphers Based on Structure, Applications, and Security Challenges

Ghassan Salloom [1], Layth Hassnawi [1], Karam J.Mohammed [1]
[1] Scientific Research Commission, Iraq
Corresponding Author *: ghassan.kh.salloom@src.edu.iq

Abstract

Stream ciphers are symmetric cryptographic algorithms designed to encrypt digital plaintext sequentially, one bit or byte at a time. The encryption algorithm of these types of ciphers produces keystreams that are combined with plaintext to produce the output ciphertext. This makes stream ciphers very useful for applications that need high performance and adequate security. There are many types of stream ciphers, including RC4, Grain, Salsa20, Trivium, ChaCha20, and MICKEY, each one of them involving different structural characteristics, performance, and security considerations. This paper proposes a comprehensive analysis of stream ciphers based on designs and highlights their various applications. Moreover, this work analyzes the strengths, limitations, and security challenges of stream ciphers, along with a comparative overview of prominent algorithms. The review concludes with future research directions, ensuring lightweight ciphers and energy-efficient designs.

Keywords:
Cipher, Stream, Cryptography, Rc4, Plaintext

1. Introduction

The rapid growth of digital communication and interconnected systems has built an increasing demand for secure and efficient cryptographic mechanisms. Although block ciphers such as AES, DES, Blowfish, and Twofish dominate modern encryption standards, their computational overhead and memory requirements are not always suitable for hardware-constrained environments or real-time applications. In contrast, stream ciphers have emerged as an attractive alternative, providing high performance, low latency, and minimal hardware complexity, and are used in many real-time applications.

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 10, October 2025
PP 167–177

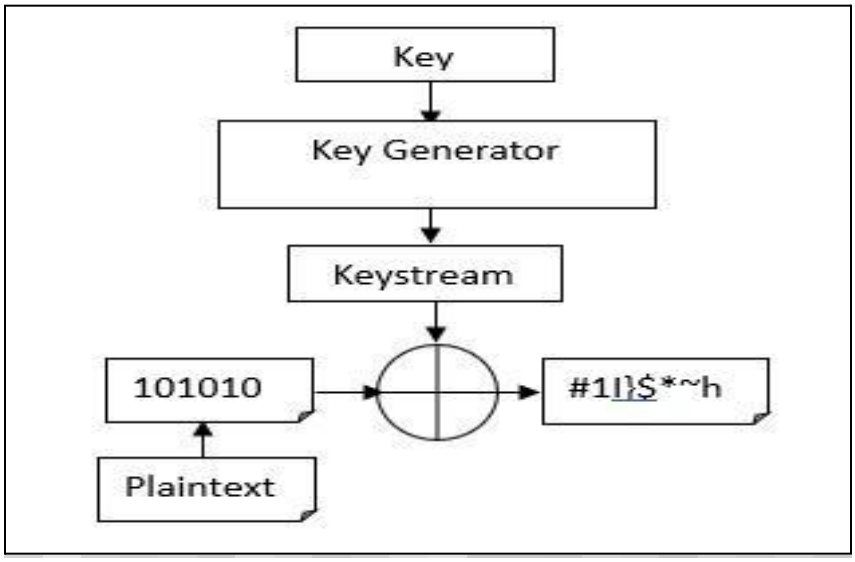
The importance of stream ciphers goes beyond traditional computing systems. They are now integral to emerging technologies, including the Internet of Things (IoT), wireless sensor networks, real-time multimedia applications, mobile communication systems, and satellite links. However, stream ciphers have vulnerabilities to cryptoanalysis such as algebraic, correlation, and state recovery attacks.

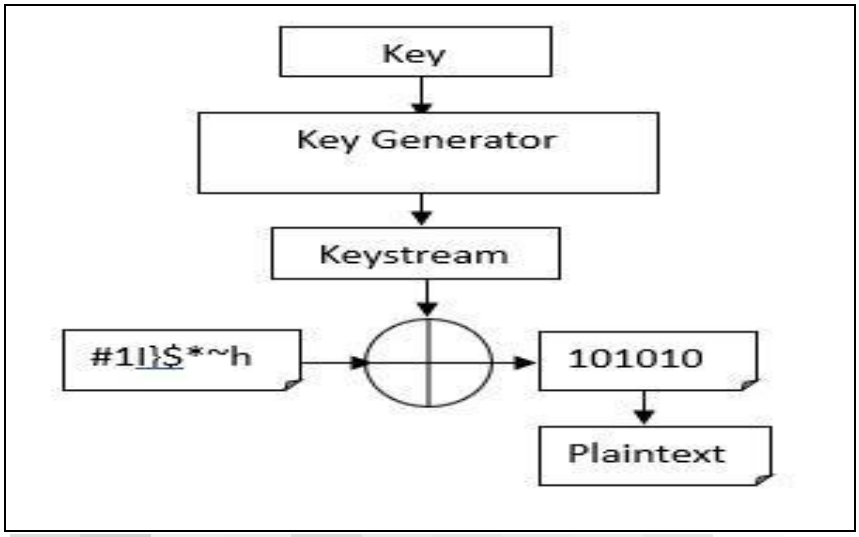
The encryption processing includes a keystream generator that utilizes a master key and generates a pseudorandom keystream based on linear feedback shift registers (LFSR) and nonlinear functions. This keystream is used to combine with the plaintext, typically using a bitwise exclusive-OR (XOR) operation, to produce the ciphertext as shown in Figure 1. While the decryption algorithm reverses the process of the encryption algorithm, using the same secret key to produce the keystream and mixing it with the ciphertext using XORing to recover the original plaintext, as shown in Figure 2.

On the other hand, typically stream ciphers are classified into two main types: synchronous and asynchronous stream ciphers. In synchronous mode, the keystream depends only on the key and initialization vector (IV). It is essential for encryption and decryption processes to remain synchronized. For instance, RC4, Salsa20, ChaCha20, and Trivium. The advantage of this type is providing high performance and simple design. However, any loss of synchronization (e.g., dropped packets) can cause decryption errors. For asynchronous stream ciphers, the keystream is dependent on the key and a static number of previous ciphertext bits or bytes. Thus, it's useful for noisy channels since the system can resynchronize automatically; for instance, cipher feedback (CFB) mode of block ciphers demonstrates this fundamental. The advantage of this type is that it is more robust to transmission errors. However, it is slower than synchronous ciphers.

This study presents a comprehensive review of stream ciphers, explores the structural basics, discusses security issues, and highlights their role across various applications.

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 10, October 2025
PP 167–177





2. Classical Stream Ciphers

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 10, October 2025
PP 167−177

Classical stream ciphers have played a significant role in SSL/TLS protocols, Wi-Fi security, GSM communication, and Bluetooth. This section highlights the most popular classical stream ciphers, such as RC4, A5/1, and E0, based on their structure, security features, and applications.

2.1 **RC4 stream cipher**

Rivest cipher 4 (RC4) was introduced by R. Rivest in 1987 [1,2]. RC4 is a type of synchronous stream cipher with a variable-size key from 40 to 2048 bits. RC4 has been used for a long time in SSL/TLS protocols and WEP/WPA wireless security standards.

The encryption process consists of two main algorithms: the key schedule algorithm (KSA) and the pseudo-random generation algorithm (PRGA). In the KSA, the algorithm initializes the 256-byte vector (S-box), then permutes this vector using the symmetric key to produce a unique initial state. The PRGA is responsible for generating a keystream by updating the initial state array through index manipulation and swapping operations, producing a keystream that is combined with plaintext using XOR to produce the final ciphertext output.

In terms of security, RC4 has been deprecated due to exposed vulnerabilities. The key scheduling algorithm (KSA) presents biases in the initial state of the permutation array (e.g., keystream bytes are not uniformly random**)** [3]. Thus, RC4 was deprecated and replaced by the ChaCha20 cipher.

2.2 **A5/1 stream cipher**

In 1987 [4,5] introduced the stream cipher A5/1. It's efficient in real-time encryption, suitable for hardware implementation in mobile devices. A5/1 was used to protect voice, messaging, and data traffic in mobile 2G and 3G communication systems.
The structure of A5/1 consists of three main linear feedback shift registers (LFSRs) of different lengths that are irregularly clocked using a rule mechanism to produce the keystream. This processing improves the nonlinearity to resist against correlation attacks. Furthermore, the output ciphertext is produced by combining the pseudorandom keystream generated with plaintext using the XOR operator.
From the security point of view, [6] showed practical attacks on A5/1, providing time–memory trade-off, correlation, and brute-force methods accelerated by precomputed tables. Thus, it's considered cryptographically weak and replaced by more secure ciphers like A5/3 and ZUC in next generations of mobile communication.

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 10, October 2025
PP 167–177

2.3 E0 Stream cipher

In the late 1990s, E0 was designed by the Bluetooth Special Interest Group (SIG) [7, 8], providing confidentiality in low-power, short-range communications. The E0 algorithm is used in the Bluetooth wireless standard for encrypting data transmitted between paired devices.

The structure of E0 consists of four LFSRs with different lengths, 25, 31, 33, and 39 bits, initialized using a 128-bit session key and a 48-bit. These LFSRs are combined by using a nonlinear function to produce the keystream as shown in Figure 3. Finally, ciphertext is generated by performing XORing the keystream with the plaintext.

E0 is not considered secure due to several attacks that can recover the encryption key with significantly less effort than a brute-force attack, such as known-plaintext attacks and algebraic attacks [9].

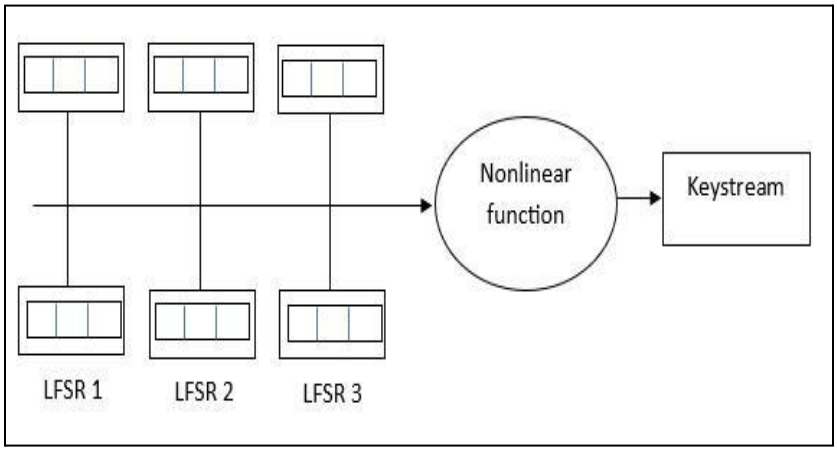


Fig . SEQ Figure * ARABIC 3. Structure of E0 Cipher

3. Modern Stream Ciphers

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 10, October 2025
PP 167–177

Grain [10], Trivium [11], and MICKEY [12] were developed based on eSTREAM project as lightweight alternatives optimized for hardware-constrained devices such as RFID and IoT with improved randomness and side-channel resistance. Additionally, Salsa20, ChaCha20, and ZUC were introduced by [13, 14, 15], respectively. These modern generations of stream ciphers provide stronger security, better statistical properties, and more efficient implementations compared to traditional ciphers such as RC4, A5/1, and E0.

3.1 Grain

The structure of the Grain family (Grain-128 and Grain-128a) cipher consists of three main phases: initializing, keystream generation, and final ciphertext output.

In the initialization phase, the secret key and IV are loaded into the dual registers, linear feedback shift register (LFSR) and nonlinear feedback shift register (NLFSR), where both registers are clocked (160 cycles) before generating the keystream output. The keystream output is fed back into the registers to mix key and IV. During the keystream phase, the Grain algorithm generates 1 keystream bit after the previous phase, then combines selected taps from the LFSR and NLFSR using XOR and AND operations to generate final keystream output. Thus, the ciphertext is generated by mixing the keystream and plaintext by using the XOR operation.

Grain is designed to resist many types of attacks, such as correlation, algebraic attacks, and distinguishing attacks. On the other hand, Grain is tailored for hardware-constrained and high-performance environments. It's widely used in IoT, mobile communication, Internet security protocols, and VPNs.

3.2 Trivium

Another modern stream cipher is Trivium, which was introduced in 2005 by Canniere and Preneel (eSTREAM finalist). It's considered an industry-standard lightweight cipher that was designed for efficiency in hardware implementations; it uses 80 bits to initialize the vector (IV) and key.

The structure of this algorithm consists of three nonlinear shift registers of total length up to 288 bits, making it robust against attacks. In initializing steps, IV and key are loaded into an internal 288-bit state composed of three feedback shift registers of lengths 93, 84, and 111 bits. During this step, the internal state is clocked 1152 times to shuffle key and IV bits before

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 10, October 2025
PP 167–177

generating the keystream. Then, the algorithm produces one keystream bit per cycle through a combination of linear and nonlinear feedback shift registers that integrate XOR and AND operations from selected register taps. Finally, the keystream is then XORed with plaintext to produce the output ciphertext.

Furthermore, key features of Trivium algorithms include a very compact hardware design (approximately less than 2000 (GE) and strong resistance against many types of attacks. It's vital for real-time services such as secure satellite links due to encrypted continuous telemetry or command data with minimal delay.

3.3 MICKEY

This cipher was introduced in 2005 by Steve Babbage and Matthew Dodd, developed as a part of eSTREM project (European ECRYPT network). This cipher is well-suited for constrained and low-power devices.

In terms of structure, MICKEY is a synchronous stream cipher with an 80-bit key and IV. It consists of a dual register: a 100-bit linear shift register (LFSR) provides a long period and linear complexity with regular updates, and feedback depends partly on control from the nonlinear register, and a 100-bit nonlinear shift register (NLFSR) introduces nonlinearity via Boolean functions. In the encryption mode, there are three stages. Key and IV setup: the key and IV are loaded into the dual registers in the initialization process. Keystream generation: for each clock cycle, the algorithm computes feedback bits from both registers, control bits from LFSR, decides whether certain feedback operations in NLFSR are applied, shifts both registers, inserting new bits, and then generates the output keystream bit by nonlinear combination of taps from both registers. The final output ciphertext is generated after combining the keystream and plaintext using the XOR operation.

In the applications part, MICKEY is perfect for RFID tags, wireless sensor networks (WSN), and IoT devices.

3.4 Salsa20

Salsa20 is one of the most important stream ciphers from the eSTREAM project and was designed in 2005 by Daniel J. Bernstein (2005). It's tailored for high speed in software, achieves fast diffusion, and has resistance against known attacks.

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 10, October 2025
PP 167−177

The structure of Salsa20 is built from a 4×4 matrix in which each element comprises 32-bit words. The elements of this matrix include a fixed 32-bit constant, a 256-bit key (or 128-bit), a 64-bit initial vector (IV), and 64-bit block counter. In the core operation, this matrix is updated by quarter-round ARX function (Addition, Rotation, XOR) operations. Where, for 20 rounds, each round consists of two parts: the column round, which applies the quarter-round function to each of the four columns of the 4×4 matrix, and the row round, which employs the quarter-round function to each of the four rows. After 20 rounds, the keystream block (512-bit) is generated. Thus, the ciphertext is produced by mixing the resulting keystream with the plaintext.

In applications and usage, Salsa20 is widely used in cryptographic libraries, backup tools, VPNs, and embedded systems, TLS 1.3, QUIC (Google), and OpenSSH.

In the security part, it's resistant against known attacks and an alternative to AES in environments where AES hardware acceleration is unavailable.

3.5 ChaCha20

Introduced by Daniel J. Bernstein (2008), it's derived from Salsa20 with some modifications to improve the security.

The structure of ChaCha20 depends on ARX designs of Salsa20 with some modifications. The key size is 256 bits, the IV is 96 bits, the block counter is 32 bits, and the internal state is 512 bits (like Salsa20).

In terms of security, there are no known practical attacks against ChaCha20/20. Furthermore, design makes it resistant to timing and cache attacks.

In the application domain, ChaCha20 is more efficient than AES when constrained by computational and memory resources, easy to implement in constant time, and safe for cryptographic protocols.

3.6 ZUC

The ZUC stream cipher was designed in 2008 by the Chinese Academy of Sciences. ZUC is a type of synchronous stream cipher supported by a 128-bit key and 128-bit initialization vector (IV).

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 10, October 2025
PP 167–177

The structure of ZUC consists of three main stages: linear feedback shift registers (LFSRs), nonlinear functions, and finite field arithmetic. The encryption process starts by combining the key and IV and then loading them into LFSR to produce a 32-bit keystream. By mixing the generated keystream with the plaintext using the XOR operation, the final ciphertext is generated.

The security of the ZUC cipher is designed for resistance against algebraic attacks, correlation attacks, and distinguishing attacks.

ZUC is used in many applications, including LTE (4G) networks and 5G mobile communication.

4. Comparative Analysis of Stream Ciphers

In this section, we conduct a comparative analysis of different stream ciphers that have developed from early designs such as RC4 to more robust modern ciphers like ChaCha20. A comparative analysis is essential to understand their trade-offs in terms of key size, initialization requirements, throughput, memory consumption, and resistance to cryptanalytic attacks, as shown in Tables 1 and 2.

Table 1. Comparative analysis based on structure

Cipher	Key Size	IV Size	Internal State
RC4	40–2048 bits	Variable (commonly 64 bit)	256 bytes (permutation array)
Grain v1/v2	80 / 128 bits	64 / 96 bits	160 / 256 bits
Trivium	80 bits	80 bits	288 bits
MICKEY 2.0	80 bits	80 bits	2000 bits
Salsa20	128 / 256 bits	64 bits	512 bits (state matrix)
ChaCha20	256 bits	96 bits	512 bits (state matrix)

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 10, October 2025
PP 167–177

Table 2. Throughput, Applications and Security

Cipher	Throughput	Application	Security Status
RC4	~1 byte/cycle	SSL/TLS, WEP, WPA (legacy)	Broken (bias, key recovery)
Grain v1/v2	High (hardware)	IoT, RFID, WSN, WSN	Secure (no practical break)
Trivium	High (hardware-efficient)	Lightweight crypto, eSTREAM	Secure (reduced-round attacks only)
MICKEY 2.0	High	IoT, constrained devices	Secure (no practical break)
Salsa20	4–14 cycles/byte	VPNs, TLS, file encryption	Secure (no practical break)
ChaCha20	~3 cycles/byte	TLS 1.3, QUIC, WireGuard VPN	Secure (widely standardized)

Tables 1 and 2 present a comparative summary of both traditional and modern stream ciphers reviewed in this study. The comparison includes their key and IV sizes, internal structure, state complexity, typical application domains, and current security status. Traditional designs such as RC4, A5/1, and E0 show significant weaknesses, including statistical biases and correlation vulnerabilities, which have led to their deprecation. In contrast, modern ciphers such as Grain, Trivium, MICKEY, Salsa20, ChaCha20, and ZUC exhibit improved nonlinearity, diffusion, and resistance against known attacks, making them suitable for lightweight and high-speed cryptographic applications.

5. Conclusion

In this work, we introduced a comprehensive review of stream ciphers with particular emphasis on their structural designs, strengths, limitations, security challenges, and a wide range of applications.

Furthermore, stream ciphers such as Grain, Trivium, and MICKEY provide cost-effective solutions for constrained hardware resources, whereas stream ciphers like Salsa20 and ChaCha20 are utilized in the Internet protocols and VPNs due to their robustness and implementation efficiency. However, challenges remain in ensuring resistance to algebraic and side-channel attacks, as well as addressing long-term threats from quantum computing.

Future research must therefore focus on integrating lightweight cryptography principles with post-quantum security requirements. In addition to this, exploring hybrid designs that combine

the strengths of stream and block cipher models. By doing so, stream ciphers can maintain their relevance as indispensable tools for securing digital communication in both constrained resources and high-performance environments.

References

1. Rivest, R. L. (1990). Cryptography. In Algorithms and complexity (pp. 717-755). Elsevier.
2. M. M. Msallam and R. Samet, "An Advanced Rivest Cipher 4 Algorithm to Transfer Fast and Secure Data Using Li-Fi Technology," *2023 IEEE 13th International Conference on System Engineering and Technology (ICSET)*, Shah Alam, Malaysia, 2023, pp. 194-199, doi: 10.1109/ICSET59111.2023.10295143.
3. Mantin, I., & Shamir, A. (2001). A practical attack on broadcast RC4. In M. Matsui (Ed.), *Fast software encryption (FSE 2001)* (pp. 152–164). Springer. https://doi.org/10.1007/3-540-45473-X_11.
4. Morii, M., & Todo, Y. (2011). Cryptanalysis for rc4 and breaking wep/wpa-tkip. *IEICE TRANSACTIONS on Information and Systems*, *94*(11), 2087-2094.
5. Lalar, D., & Nahta, R. (2016). STREAM CIPHER. *International Journal of Advanced Research in Computer Science*, *7*(7).
6. Biryukov, A., Shamir, A., & Wagner, D. (2001). Real time cryptanalysis of A5/1 on a PC. In *Fast software encryption (FSE 2000)* (pp. 1–18). Springer. https://doi.org/10.1007/3-540-44706-7_1.
7. Biryukov, A., Shamir, A., & Wagner, D. (2001). Real time cryptanalysis of A5/1 on a PC. In *Fast software encryption (FSE 2000)* (pp. 1–18). Springer. https://doi.org/10.1007/3-540-44706-7_1.
8. Hermelin, M., & Nyberg, K. (1999). Correlation properties of the Bluetooth stream cipher E0. In *Selected areas in cryptography (SAC 1999)* (pp. 17–30). Springer. https://doi.org/10.1007/3-540-46513-8_2
9. Lu, J., & Vaudenay, S. (2004). Cryptanalysis of the Bluetooth stream cipher E0. In *Advances in cryptology—CRYPTO 2004* (pp. 515–530). Springer. https://doi.org/10.1007/978-3-540-28628-8_32
10. Lu, J., & Vaudenay, S. (2004). Cryptanalysis of the Bluetooth stream cipher E0. In M. Franklin (Ed.), *Advances in cryptology – CRYPTO 2004* (pp. 515–530). Springer. https://doi.org/10.1007/978-3-540-28628-8_32.
11. Hell, M., Johansson, T., & Meier, W. (2005). Grain: A stream cipher for constrained environments. In *International workshop on cryptographic hardware and embedded systems (CHES 2005)* (pp. 145–160).
12. De Canniere, C., & Preneel, B. (2005). Trivium specifications. In *eSTREAM, ECRYPT Stream Cipher Project*. Retrieved from https://www.ecrypt.eu.org/stream/trivium.html.
13. Babbage, S., & Canniere, C. D. (2006). MICKEY 2.0: Mutual irregular clocking keystream generator. In *Workshop record of SASC 2006: The state of the art of stream ciphers* (pp. 1–12).
14. Bernstein, D. J. (2008). The Salsa20 family of stream ciphers. In M. Robshaw & O. Billet (Eds.), *New stream cipher designs: The eSTREAM finalists* (pp. 84–97), Springer. https://doi.org/10.1007/978-3-540-68351-3_6.
15. Bernstein, D. J. (2008). *ChaCha, a variant of Salsa20*. In SASC 2008: The State of the Art of Stream Ciphers (pp. 3–5). Retrieved from https://cr.yp.to/chacha.html
16. Wu, H., & Huang, T. (2011). The ZUC stream cipher. In B. Preneel & T. Takagi (Eds.), *Cryptographic hardware and embedded systems – CHES 2011* (pp. 1–13). Springer. https://doi.org/10.1007/978-3-642-23951-9_1