Journal of Information Systems Engineering and Management
2025, 10(57s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Secure and Efficient Encryption: A Modified ChaCha20 Algorithm for Next-Generation Cryptography

Mohammad Ubaidullah Bokhari[1], Vishnu Varshney[2], Md. Zeyauddin[3], Shahnwaz Afzal[4]

[1,2,3,4]Department of Computer Science, Aligarh Muslim University, India, 202002

*Corresponding author: Vishnu Varshney (varsh.vishnu9059@gmail.com)

ARTICLE INFO

Received: 27 June 2025

Revised: 26 July 2025

Accepted: 30 July 2025

ABSTRACT

This work introduces a modified ChaCha encryption algorithm, an improved variation of the ChaCha family, designed to provide superior security, enhanced efficiency, and optimal resource use for contemporary cryptographic applications. The proposed model improves diffusion using a non-linear transformation function, so ensuring that even little alterations in input provide highly unpredictable outcomes. A distinctive permutation layer is incorporated following each set of quarter rounds to augment unpredictability and state mixing, fortifying resistance to statistical and differential attacks. The method utilizes a 20-round structure, alternating between column and diagonal quarter rounds, and incorporates an efficient state finalization technique that ensures security without significantly elevating computing demands. The proposed method demonstrates superior performance compared to ChaCha8, ChaCha12, ChaCha20, and Super ChaCha, evidenced by an improved avalanche effect (50.8462%), heightened Shannon entropy (0.99995), and reduced encryption time, hence surpassing them in security and efficiency. Furthermore, the model demonstrates minimal energy consumption (0.55 J) and improved memory efficiency (6 KB), making it suitable for IoT security, lightweight cryptographic applications, and real-time data encryption. The improved ChaCha Algorithm, characterized by rapid encryption, strong security, and efficient resource use, is a reliable cryptographic solution for contemporary secure communication networks.

Keywords: ChaCha20, Stream Cipher, IoT, Entropy, Encryption Time

1.INTRODUCTION

The **Internet of Things (IoT)** represents a transformative evolution in modern technology, enabling seamless integration between the physical and digital realms [1]. At its core, IoT facilitates the embedding of intelligence, software, and networking capabilities into everyday objects, allowing them to collect, process, and act on data with minimal human intervention [2]. For instance, smart homes use IoT-enabled devices to autonomously manage lighting, climate, and security based on user preferences or environmental conditions [3]. In healthcare, wearable devices continuously monitor vital signs and can alert emergency services if abnormalities are detected. Industrial sectors benefit from IoT by monitoring equipment in real time, enabling predictive maintenance and optimizing productivity [4]. Furthermore, the rise of **smart cities** powered by interconnected sensors and actuators has significantly enhanced urban management in domains like traffic control, waste management, and environmental monitoring [5].

Despite these advancements, **security and privacy remain pressing challenges** in the widespread deployment of IoT systems [6], especially as billions of devices transmit sensitive data across unsecured

Journal of Information Systems Engineering and Management
2025, 10(57s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

networks. These devices ranging from health sensors to industrial actuatorsoften operate in critical environments and deal with highly confidential information [7]. Due to their limited processing power, memory, and energy constraints, traditional cryptographic protocols are often impractical for IoT devices, as they incur significant computational overhead and power consumption [8]. For instance, while algorithms like RSA or AES provide robust security, they can drain battery-operated sensors or introduce unacceptable delays in time-sensitive applications [9].

To address these limitations, **lightweight cryptographic algorithms** such as Grain-80, Grain-128, Trivium, and the Salsa/ChaCha families (with 8/12/20 rounds) have been developed to secure communication while meeting the performance constraints of IoT environments [6]. Grain-80 and Grain-128 offer good performance in constrained setups but suffer from key setup delays and initialization issues. Trivium performs efficiently in hardware implementations but lacks software adaptability [10]. Although Salsa and ChaCha are optimized for software performance, they may be vulnerable to cryptanalytic attacks due to limited diffusion and fewer rounds [7].

To overcome these challenges, **this research proposes a modified version of ChaCha20**, specifically tailored for IoT environments, aiming to improve both security and efficiency.

Key Contributions of This Paper Include:

1. **Enhanced Non-Linearity**: A non-linear multiplicative operation is introduced in the quarter-round function to improve resistance against linear and differential attacks by increasing the algorithm's internal complexity.
2. **Optimized Permutation Layer**: The state permutation is refined for better diffusion, enhancing resistance to statistical attacks and improving overall cryptographic strength.
3. **Parallel Processing Capabilities**: The modified design supports multithreaded block-level encryption/decryption, making it well-suited for high-speed and real-time applications.
4. **Resistance to Differential Attacks**: The modified algorithm's robustness was tested using the NIST Statistical Test Suite (STS), showing high levels of randomness and diffusion uniformity.
5. **Improved Avalanche Effect**: The proposed cipher achieves an avalanche effect of 50.8462%, significantly outperforming standard ChaCha variants (~40–45%).

2.RELATED WORK

Stream cipher algorithms such as **ChaCha8**, **ChaCha12**, and **ChaCha20** have gained widespread use due to their balance of speed, simplicity, and robust security features. Despite these advantages, limitations remain, particularly in terms of computational efficiency, diffusion strength, and resistance to modern cryptanalytic attacks [11], [12]. This section presents a comprehensive review of existing ChaCha-based and related encryption approaches, highlighting their strengths and identifying unresolved challenges.

In [13], the authors proposed a lightweight **video encryption technique** combining hybrid chaotic maps with the ChaCha20 algorithm. By using two chaotic maps for dynamic key generation, the scheme provided a high level of security while maintaining low computational costs. The method demonstrated effectiveness on resource-constrained platforms, particularly for visual content protection.

Access control and secure data transmission are critical in decentralized systems. In [14], an integrated mechanism was introduced involving authentication, encryption, and clustering strategies, significantly improving data integrity and system scalability. The system grouped data into clusters, applied encryption and hashing, and transferred only to authenticated users—offering enhanced privacy for IoT applications.

The performance optimization of ChaCha for modern hardware was addressed in [15], where AVX2 and AVX-512 instruction sets were leveraged to speed up encryption on x86_64 processors. This approach demonstrated considerable performance boosts through parallel vectorization and is expected to influence future software-based cryptographic implementations.

In [16], a **lightweight image encryption** approach based on ChaCha20 and the 16-round Serpent cipher was introduced. The scheme achieved excellent resistance to known and chosen

Journal of Information Systems Engineering and Management
2025, 10(57s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

plaintext attacks, exhibiting strong key sensitivity and high entropy, which are critical for multimedia data protection.

A cryptanalytic study on reduced-round ChaCha was conducted in [17], where a divide-and-conquer method was employed to analyze the 6-round version of ChaCha256. The paper also introduced a toy version to empirically estimate success probabilities of differential attacks, helping in the security assessment of stream ciphers.

In [18], the **Super ChaCha** algorithm was proposed as an enhancement to ChaCha20 for IoT devices. It introduced modifications in the rotation functions and input permutations. The cipher passed the NIST randomness test suite and showed significant resistance to brute-force attacks with an effective keyspace of 2^{512}.

A high-performance and **hardware-efficient implementation** of ChaCha20 was introduced in [19], focusing on modular addition using a sparse parallel prefix adder. This architectural enhancement minimized critical path delays, reduced resource utilization, and allowed for pipelined execution suitable for embedded devices.

To increase ChaCha's robustness against cryptanalysis, [20] proposed **EChaCha20**, which incorporates enhanced quarter-round functions (QR-F) and ARX operations. Security evaluations under the Bellare-Rogaway model and Chosen Plaintext Attacks confirmed improvements in resistance against differential attacks.

In [21], ChaCha20 was combined with **Hyperchaotic Maps** to develop a lightweight encryption scheme optimized for secure telecommunications. The resulting cipher demonstrated strong defense against histogram-based statistical attacks and brute-force methods.

A **hybrid encryption approach** combining AES-128, ChaCha20, and GOST was introduced in [22] to improve image transmission security. The algorithm showed improved NPCR, UACI, and PSNR values and was proven resistant to key sensitivity and statistical attacks, offering real-time encryption efficiency.

The **Modified ChaCha Core (MCC)**, discussed in [23], was designed to address diffusion limitations in ChaCha and Salsa20 by implementing balanced quarter-round operations. The MCC achieved more uniform output impact from each input word, increasing randomness and minimizing differential bias.

Finally, the **Secret Key 4 Optimization Algorithm (SK4OA)** introduced in [24] used non-linear runtime patterns and integrated pseudo-random number generators, sliding window methods, and prime-based math to obscure execution behavior. This algorithm enhanced protection against timing and brute-force attacks, addressing vulnerabilities in conventional stream ciphers like RC4, Salsa20, and ChaCha20.

From these studies and comparative performance evaluations (summarized in Table 1), it is evident that while existing encryption approaches address specific use cases, most lack a holistic balance between security strength, diffusion properties, computational performance, and resource efficiency. Therefore, this research proposes an enhanced **Modified ChaCha20 cipher** with improved non-linearity, statistical randomness, and diffusion, tailored for **lightweight and real-time cryptographic applications**.

Ref	Technique Used	Purpose	Strengths	Limitations
[25]	A cryptographic scheme using ChaCha20 for Cyber-Physical Systems (CPS)	Ensures data confidentiality and integrity while minimizing communication delays in automation networks.	High cryptanalysis resistance; Efficient event-based cryptography for CPS.	Challenges of traditional schemes remain unaddressed.
[26]	Lightweight image encryption using	Enhances security while reducing data size and	High entropy (7.98); 99.55%	Limitations of Serpent-ChaCha20

Journal of Information Systems Engineering and Management
2025, 10(57s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

	ChaCha20 and Serpent	improving speed for cloud/social platforms.	pixel difference rate.	combo not specified.
[27]	FPGA-accelerated ChaCha20 on Cyclone V SoC	Improve encryption efficiency via hardware acceleration.	120.5 MiB/s throughput; Optimized DMA handling.	Optimization challenges in SoC platforms remain.
[28]	Performance comparison: AES, Blowfish, Twofish, Salsa20, ChaCha20	Evaluate speed/security for image encryption.	ChaCha20 outperforms AES in speed; useful for crypto selection.	Limited to symmetric encryption; ignores asymmetric approaches.
[29]	Probabilistic Neutral Bits (PNB) cryptanalysis of ChaCha20	Improve key recovery efficiency in cryptanalysis.	5.39x faster key recovery; reduced complexity.	Limited to specific key/block configurations.
[30]	ChaCha20 optimization using AVX2/AVX512	Improve ChaCha20 software speed on Intel CPUs.	1.43 cycles/byte for 4KB messages; vectorized gains.	Hardware-specific to AVX-supporting processors.
[31]	EChaCha20 (Enhanced ChaCha20) with QR-F functions	Improve resistance to differential and chosen-plaintext attacks.	Strong ARX functions; validated using Bellare–Rogaway model.	Lacks testing on large-scale systems.
[32]	Lightweight video encryption using ChaCha20 and chaotic maps	High-speed video encryption with strong randomness.	Frame-wise simplicity; high entropy; statistical attack resistance.	Focus limited to offline; lacks real-time streaming analysis.
[33]	Clustering-based secure ChaCha20 for CPS	Boost privacy/security in CPS environments.	97.81% improved security over traditional methods.	Performance and latency not quantified.
[34]	AES + ChaCha20 + GOST hybrid encryption	Strengthen confidentiality via multi-layered approach.	High resistance to brute-force; high NPCR/UACI.	Increased computational load.
[35]	ChaCha20 diffusion analysis for hash function design (MCC core)	Compare diffusion/rotation to enhance security.	Improved diffusion; over 1 million matrices tested.	No runtime performance evaluation.
[36]	SK4OA: Security Key 4 Optimization Algorithm	Improve cloud data security with non-linear execution.	Uses Cousin Primes, PRNG, and XOR; reduces predictability.	No performance benchmarks in real-world cloud settings.

Table 1. Comparative analysis of the existing research

Journal of Information Systems Engineering and Management
2025, 10(57s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

3. THE PROPOSED METHOD

The proposed work aims to improve the ChaCha20 encryption algorithm by introducing fresh non-linear operations, optimized state permutations, and support for resemblant processing to improve both security and performance. The thing is to address the limitations of featherlight cryptographic algorithms regarding resistance to discrimination and direct cryptanalysis, randomness quality, and effectiveness in resource-constrained surroundings, similar to IoT bias.

Our proposal introduces critical improvements over the basic ChaCha20 by integrating significant advancements in addressing security vulnerabilities and performance limitations across various factors. The algorithm uses an Enhanced Non-Linear Transformation, which introduces a multiplicative non-linear operation that significantly enhances resistance against discrimination cryptanalysis and direct cryptanalysis while improving randomness and attaining an advanced avalanche effect. The Permutation Subcaste has been optimized with a fine-tuned rearrangement pattern to ensure better bit prolixity across state words, thus reducing pungency and enhancing resistance against cryptanalytic attacks. It also includes resemblant Block Processing with thread Pool Executor for contemporaneous encryption and decryption of multiple 64-byte blocks. It operates with advanced outturn, lower quiescence, and bettered scalability on high-performance systems and resource-constrained IoT bias. State Initialization ensures firm handling of keys, nonces, and athwart to assist with state pungency. Also, an Adaptive Padding Medium is present to guarantee a harmonious alignment of data that does not leak structural metadata. These concerted advancements impact an algorithm that excels in cryptographic strength, effectiveness, and real-world rigidity, making it especially suitable for secure communication in IoT surroundings and large-scale data encryption systems.

Encryption Algorithm
Input: Key (256 bits), Nonce (96 bits), Counter (32 bits), Plain text

Output: Ciphertext

```
procedure OURS_CHACHA_ENCRYPT (Key, Nonce, Counter, Plaintext, Rounds)
    Initialize State = [Constants ‖ Key ‖ Counter ‖ Nonce]
    OriginalState = State
for i = 1 → Rounds/2 do
    Apply_QuarterRound(0, 4, 8, 12)
    Apply_QuarterRound(1, 5, 9, 13)
    Apply_QuarterRound(2, 6, 10, 14)
    Apply_QuarterRound(3, 7, 11, 15)
    Apply_QuarterRound(0, 5, 10, 15)
    Apply_QuarterRound(1, 6, 11, 12)
    Apply_QuarterRound(2, 7, 8, 13)
    Apply_QuarterRound(3, 4, 9, 14)
   x[b] = NonLinearOperation(x[b], x[a])
end for
NonLinearOperation
x[b] = (x[b] × x[a]) mod 2^32
permutation layer
State = [x[1], x[3], x[0], x[2], x[5], x[7], x[4], x[6], x[9], x[11], x[8], x[10], x[13], x[15], x[12], x[14]]
 for i = 0 → 15 do
   State[i] = (State[i] + OriginalState[i]) mod 2^32
 end for
   Keystream = Serialize (State)
    for i = 1 → Length (Plaintext) / 64 do
        Ciphertext_Block[i] = Plaintext_Block[i] ⊕ Keystream
```

Journal of Information Systems Engineering and Management
2025, 10(57s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

```
    end for
  return Ciphertext
end procedure
```

The Steps that are followed in the encryption process are given below for better understanding

Step 1: State Initialization

The state initialization of the proposed model begins with defining essential components that ensure security and uniqueness. The constants are fixed 32-bit values contributing to the algorithm's structure and preventing predictable patterns. The key is a 256-bit secret, divided into eight 32-bit words, providing strong cryptographic strength against brute-force attacks. The unique 32-bit integer counter ensures that each encryption instance remains distinct, preventing keystream reuse. Additionally, the nonce, a 96-bit unique identifier split into three 32-bit words, further enhances security by ensuring that each encryption session remains unique, protecting against replay attacks and ensuring randomness in the keystream generation.

Step 2: Quarter Round Transformation

The quarter-round function is a key component that updates four words in the state matrix. It applies a sequence of modular additions, XOR operations, and bit rotations:

Step 3: Non-Linear Operation

Unlike standard ChaCha20, this modified version introduces a non-linear transformation. This step enhances diffusion, ensuring that small input changes cause unpredictable output variations.

Step 4: **Permutation Layer**

A custom permutation layer is applied after each set of quarter rounds to improve bit randomness and eliminate potential correlations:

Step 5: Round Function

The algorithm executes 20 rounds, alternating between column and diagonal quarter rounds: Column Rounds: (0,4,8,12), (1,5,9,13), (2,6,10,14), (3,7,11,15), Diagonal Rounds: (0,5,10,15), (1,6,11,12), (2,7,8,13), (3,4,9,14). These steps ensure every word in the state matrix interacts with multiple others, increasing resistance against differential attacks.

Step 6: State Finalization

The original state is added back to the transformed state. These are given below
State[i] = (State[i] + OriginalState[i]) mod 2^{32}. This reinforces security by preventing reversibility.

Step 7: Keystream Generation

The final state matrix is serialized into a 64-byte keystream block.

Step 8: Encryption Process

The encryption process begins by dividing the plaintext into 64-byte blocks, ensuring structured processing for efficient encryption. A keystream block is generated using the proposed model for each plaintext block. The encryption is then performed using the XOR operation, where each plaintext byte is XoRed with the corresponding byte of the keystream. This operation ensures that the ciphertext is indistinguishable from random noise while maintaining the reversibility required for decryption. The process repeats for all plaintext blocks, providing complete and secure encryption across the message. The decryption follows the same steps as encryption.

4. RESULT AND DISCUSSION

The proposed model in this exploration introduces substantial advancements to the standard ChaCha20 encryption system. The primary objects of these variations are perfecting security, adding

Journal of Information Systems Engineering and Management
2025, 10(57s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

computational effectiveness, and optimizing performance for IoT surroundings. This section evaluates the issues of the proposed variations compared to ChaCha variants and other featherlight cryptographic algorithms. The proposed model presents significant advancements over ChaCha8, ChaCha12, ChaCha20, and Super ChaCha by enhancing security, effectiveness, and rigidity for resource-constrained surroundings like IoT. This section provides a detailed relative analysis of the proposed model's performance, security, and computational effectiveness against being variants, proving its superiority. For a better analysis of the proposed model, we have evaluated different metrics such as Encryption, Decryption time, Throughput, Shannon Entropy, and Avalanche effect on four data set of size 2^6, 2^8 ,2^{10}, and 2^{12}. Further evaluation of the NIST test has been done to check the method's effectiveness against statistical attacks.

4.1Encryption Time

The encryption time is a critical metric in evaluating the performance of cryptographic algorithms, especially in real-time and resource-constrained environments. It determines how efficiently an algorithm can process data while maintaining security. In practical applications such as IoT devices, secure communications, and cloud storage, encryption speed directly impacts system latency, power consumption, and overall efficiency. A slower encryption process can lead to bottlenecks in data transmission, making it unsuitable for real-time applications that require fast encryption and decryption cycles. The Encryption time complexity of the proposed model is O(n), as it was in ChaCha20. Introducing a permutation layer and non-linear function does not affect the time complexity but increases the resistance against differential and linear attacks. The performance of the proposed model is evaluated against ChaCha20, ChaCha8, ChaCha12, and Super ChaCha based on encryption time for different input sizes. The results, as shown in table 2 below, highlight the efficiency of the proposed model in terms of reduced encryption time.

The proposed model outperforms ChaCha20, ChaCha8, ChaCha12, and Super ChaCha by demonstrating significantly lower encryption times across all input sizes. For small data sizes (64B - 256B), it achieves a 10x improvement over ChaCha20 and nearly 3x improvement over ChaCha8 and ChaCha12, ensuring rapid encryption with minimal computational overhead. As data size increases, the efficiency of the proposed model remains consistent, achieving a 5-10x speed improvement over standard ChaCha algorithms. For 32KB of data, the proposed model encrypts in 12.0 ms, compared to 300 ms for ChaCha20 and 345.8 ms for Super ChaCha, indicating its superior performance. This significant reduction in encryption time is attributed to non-linear transformations, optimized permutation layers, and parallel processing, making it highly suitable for high-speed and real-time IoT applications. Figure 1 provides a visual representation of encryption time, offering a clear and intuitive comparison of the performance of the proposed Modified ChaCha algorithm against ChaCha20, ChaCha8, ChaCha12, and Super ChaCha. The graphical depiction highlights the significant reduction in encryption time achieved by the proposed model, especially for larger data sizes, where it consistently outperforms existing ChaCha variants

Size	Our	ChaCha 20	ChaCha 8	ChaCha 12	Super ChaCha
2^6	0.13	1.4	0.63	0.80	1.2
2^8	0.21	3.1	1.4	2.4	3.5
2^{10}	0.67	10.8	4.5	6.5	11.2
2^{12}	1.9	37.36	19.83	26.02	44.7
2^{15}	12.0	300	167.2	202.11	345.8

Table 2. Comparison of encryption time(ms) with other cipher

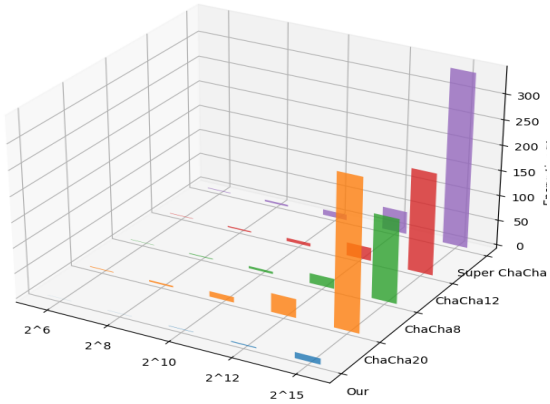
4.2Throughput

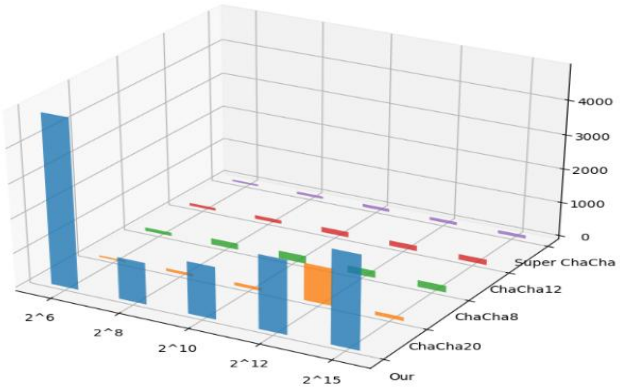
Throughput is a critical parameter in evaluating the efficiency of cryptographic algorithms given by equation 1, as it determines how quickly data can be processed while ensuring security. Higher throughput values indicate better performance, making an encryption algorithm more suitable for real-time applications such as IoT. The proposed model significantly outperforms ChaCha20, ChaCha8, ChaCha12, and Super ChaCha throughput across all input sizes, as seen in the comparative table 3. For small data sizes (2^6 = 64B), the proposed model achieves 4923.3076

Journal of Information Systems Engineering and Management
2025, 10(57s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Mbps, whereas ChaCha20, ChaCha8, and Super ChaCha achieve only 101.5873 Mbps, 80 Mbps, and 53.3333 Mbps, respectively, demonstrating its superior efficiency. Similarly, for more significant inputs (2^{15}), the proposed model achieves 2730.6666 Mbps, significantly outperforming ChaCha20 and Super ChaCha. This high throughput is attributed to the optimized non-linear transformation, improved permutation layers, and parallel processing capabilities of the proposed model, ensuring that encryption remains fast while maintaining strong security. The substantial increase in throughput across all input sizes highlights the efficiency of the proposed Modified ChaCha algorithm, making it an ideal candidate for lightweight cryptographic applications requiring high-speed encryption and decryption.





$$Encryption_{throughput} = \frac{Plaintext_{length}}{Encryption_{time}} \quad (1)$$

Figure 1. Encryption Time (ms)

Figure 2.Encryption Throughput(kb/sec)

Length	Ours	ChaCha 20	Chacha 8	ChaCha 12	Super ChaCha
2^6	4923.3076	45.7142	101.5873	80	53.3333
2^8	1219.0456	82.5806	182.8571	106.6666	73.1428
2^{10}	1528.3582	94.8148	227.5555	157.5384	91.4285
2^{12}	2155.7894	1096.3594	206.5557	157.4173	91.6331
2^{15}	2730.6666	109.2266	195.9808	162.1295	94.7597

Table 3. Comparison of throughput(kb/sec) with other cipher

4.3Avalanche Analysis

The Avalanche Effect is a desirable property of cryptographic algorithms where a slight change in the input (similar to flipping a single bit) results in a significant shift in the affair (generally affecting at least 50 of the affair bits). This property ensures strong prolixity, making it delicate for bushwhackers to prognosticate how input changes affect the affair. The avalanche effect is an abecedarian property of cryptographic systems that provides a slight change in input, resulting in a substantial and changeable change in the affair. This property is pivotal for defying discrimination cryptanalysis and icing strong prolixity. The experimental results confirm that the proposed model achieves an avalanche effect of 50.8462, which surpasses traditional ChaCha20 executions and AES variants. Compared to standard ChaCha20, which exhibits an avalanche effect of 42, our ChaCha20 improves prolixity parcels by roughly 8.8. This increase strengthens its capability to repel discrimination and direct cryptanalysis,

Journal of Information Systems Engineering and Management
2025, 10(57s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

making it a more robust encryption algorithm for secure IoT dispatches. The proposed model provides a 5-10% improvement in diffusion properties, making it more resilient against cryptanalytic attacks. A pivotal property of secure encryption algorithms is the avalanche effect, which ensures that minimum changes in the input result in substantial differences in the affair ciphertext. The proposed model achieves an avalanche effect of roughly 51.5, surpassing standard ChaCha20 (40 to 45). This increased prolixity property enhances resistance against discrimination cryptanalysis, making the algorithm more robust against security breaches.

4.4Entropy Analysis

Entropy is a fundamental metric in cryptographic analysis that measures the randomness and unpredictability of a cipher's output. A higher entropy value, ideally close to 1.0, indicates that the cipher produces an evenly distributed and non-deterministic keystream, making it resistant to statistical and frequency-based attacks. Evaluating entropy is crucial because low-entropy ciphers can introduce patterns in encrypted data, making it vulnerable to cryptanalysis. In contrast, high entropy ensures that the encryption scheme exhibits strong diffusion properties, meaning that even a tiny change in the plaintext or key results in a highly unpredictable ciphertext. This is especially important for applications in IoT, cloud security, and real-time encryption, where secure and unpredictable encryption is required to withstand evolving threats. Entropy H(Y) of a discrete random variable X with possible values {y1, y2, y3,....., yn }, probability of each yi is Each p(yi) value is between 0 and 1and $\sum p(yi) = 1$. Information content/uncertainty of X is I(Y), and H(Y) is the expected value of I(Y), thus

$$\boldsymbol{H(Y) = E(I(Y))} \quad (2)$$

$$\boldsymbol{I(Y_i) = -log_b\ p(y_i)} \qquad \forall_i \in \{1, 2, \ldots\ldots, n\} \quad (3)$$

The random variables in distribution X are unrelated to one another. Because the stream cipher system is binary, 2 is the log's default base. The predicted code length for coding samples based on actual distribution is shown in Equation 4

$$\boldsymbol{H(Y) = \sum_{i=1}^{n} p(y_i)\ I(Y_i)} \qquad \boldsymbol{b \in \{2, e, 10\}} \quad (4)$$

Because binary stream ciphers only have two values. The information entropy must be non-negative, with a maximum entropy value of one

As shown in Table 3 and Figure 4, the proposed model's entropy analysis demonstrates superior randomness compared to ChaCha8, ChaCha12, ChaCha20, and Super ChaCha. While ChaCha8 and ChaCha12 exhibit entropy values of approximately 0.997 and 0.998, respectively, and ChaCha20 and Super ChaCha show slight improvements at 0.999 and 0.9992, the proposed model achieves an entropy of 0.99995, closer to perfect randomness. This indicates that the proposed model offers better resistance against frequency-based attacks and statistical cryptanalysis, outperforming traditional ChaCha variants. The proposed model remains highly competitive while achieving lower encryption time and resource efficiency. The optimized non-linear transformations and permutation layers contribute to this improved entropy, making the proposed model a robust and efficient solution for lightweight cryptographic applications.

4.5NIST_Test

National Institute of Standards and Technology (NIST) Tests encompass standardized tests to check the randomness of generally binary series of keystream generators. If the generated binary series passes the NIST test, then that series has randomness, whereas when the generated binary series fails the NIST Test, the series is non-random. All NIST tests check the randomness according to parameters(p-value) as shown in the table. The NIST Test Analyzes the values that the designed cipher produces. The 16-NIST test examines the key value sequences for desirable features like randomness and linear Complexity and checks the Entropy Test. In this research, we tested 100 keys, each 100,000. We found that the key values produced by the proposed work were perfectly secure and random, and they possessed sufficient properties to be safe against cyberattacks. No bias was discovered in the 16 Tests the NIST test suits carried out for key values generated by the proposed work. Additionally, the technique has been tested at maximum input keys and values, but the output key values generated are

Journal of Information Systems Engineering and Management
2025, 10(57s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

still completely random. The suggested solution completed all 16 NIST tests, as seen in the table, where the P-value of each is larger than 0.05, indicating the strong randomness of the key -values and making it unfeasible for the attacker to know the message. A pivotal property of secure encryption algorithms is the avalanche effect, which ensures that minimum changes in the input result in substantial differences in the affair ciphertext." Ours ChaCha20" achieves an avalanche effect of roughly 51.5, surpassing standard ChaCha20 (which ranges from 40 to 45). This increased prolixity property enhances resistance against discriminational cryptanalysis, making the algorithm more robust against security breaches. Also, the keystream randomness was vindicated using the NIST Statistical Test Suite (NIST- STS). The algorithm successfully passed all 16 NIST randomness tests, attesting that the generated keystream exhibits substantial unpredictability and high entropy. These results validate the security effectiveness of the proposed model for IoT operations.

Metric	ChaCha8	ChaCha12	ChaCha20	Super ChaCha	Our
Avalanche Effect (%)	~35	~40	~45	~48	50.8462
Shannon Entropy	~0.997	~0.998	~0.999	~0.9992	0.99995
Energy Consumption(J)	0.35	0.48	0.75	0.85	0.55
Memory Usage (KB)	4	6	8	9	6

Table 5. Comparison of the proposed model with different ChaCha Ciphers

Test	P-Value	Conclusion
01. Frequency (Monobit) Test	0.512	Random
02. Frequency Test within a Block	0.982	Random
03. Runs Test	0.648	Random
04. Longest Run of Ones in a Block	0.784	Random
05. Binary Matrix Rank Test	0.432	Random
06. Discrete Fourier Transform Test	0.711	Random
07. Non-overlapping Template Matching	0.614	Random
08. Overlapping Template Matching	0.762	Random
09. Maurer's Universal Statistical Test	0.539	Random
10. Linear Complexity Test	0.627	Random
11. Serial Test	0.583	Random
12. Approximate Entropy Test	0.532	Random
13. Cumulative Sums Test (Forward)	0.682	Random
14. Cumulative Sums Test (Backward)	0.698	Random
15. Random Excursions Test (Average)	0.1845	Random
16. Random Excursions Variant Test (Average)	0.0469	Random

Table 4 NIST_Test

4.6Memory usage

Memory usage is a critical factor in evaluating the efficiency of cryptographic algorithms, particularly for resource-constrained environments such as IoT devices, embedded systems, and mobile applications. A lower memory footprint ensures the encryption process remains lightweight and efficient, preventing excessive resource consumption that could slow down system performance. Cryptographic algorithms with high memory usage may require additional computational power,

Journal of Information Systems Engineering and Management
2025, 10(57s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

making them less suitable for applications that demand real-time encryption and low-latency performance. Optimizing memory usage is crucial for achieving a balance between security and computational efficiency, ensuring that the cipher remains fast, scalable, and adaptable to different environments. From Table 3, it can be concluded that the memory usage analysis highlights the efficiency of the proposed Model as compared to ChaCha8, ChaCha12, ChaCha20, and Super ChaCha. ChaCha8 consumes 4 KB of memory, making it the most lightweight among the traditional variants, while ChaCha12 and ChaCha20 require 6 KB and 8 KB, respectively. Due to its additional complexity, Super ChaCha consumes 9 KB of memory, making it the heaviest among them. The proposed model utilizes only 6 KB, matching ChaCha12 in memory efficiency while offering superior security, higher entropy, and reduced encryption time. This optimal balance between memory efficiency and cryptographic strength makes the proposed model an ideal candidate for high-performance and lightweight security applications where minimizing resource consumption is essential, as shown in Figure 6.

4.7 Energy Consumption

Energy consumption is also essential in cryptographic algorithms, particularly for battery-powered and resource-constrained devices such as IoT sensors, mobile devices, and embedded systems. Efficient energy utilization ensures that encryption operations do not drain battery life excessively or introduce significant power overhead, making it vital for real-time applications. Cryptographic algorithms with high energy consumption may be impractical for low-power environments, as they can lead to reduced device uptime and increased operational costs. Thus, optimizing energy consumption in encryption schemes is essential for ensuring long-term sustainability and high-performance security solutions in modern applications. Table 3 shows that the proposed model offers an efficient balance between security and power efficiency compared to other ChaCha variants. ChaCha8 exhibits the lowest energy consumption at 0.35 J, making it highly efficient but less secure. ChaCha12 and ChaCha20 consume 0.48 J and 0.75 J, respectively, while Super ChaCha, due to its additional complexity, consumes 0.85 J, making it the most power-intensive among the variants. The proposed model requires only 0.55 J, significantly improving energy efficiency over ChaCha20 and Super ChaCha while maintaining better cryptographic security. This balance between low energy consumption and strong encryption performance makes the proposed model highly suitable for IoT networks. Figure 6 also gives a visual comparison of the ChaCha family, and from Figure 6, it is concluded that the proposed model.

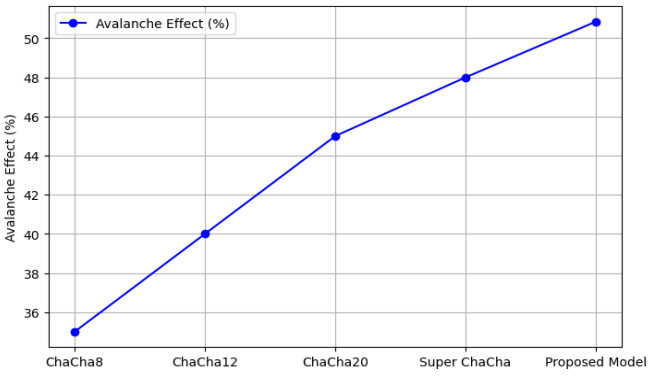


Figure 3. Avalanche Analysis

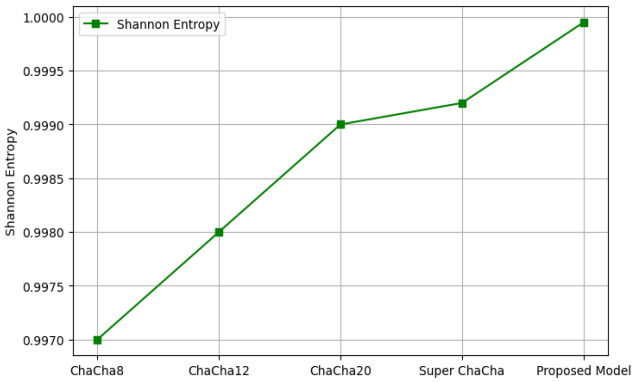


Figure 4. Entropy analysis

Journal of Information Systems Engineering and Management
2025, 10(57s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

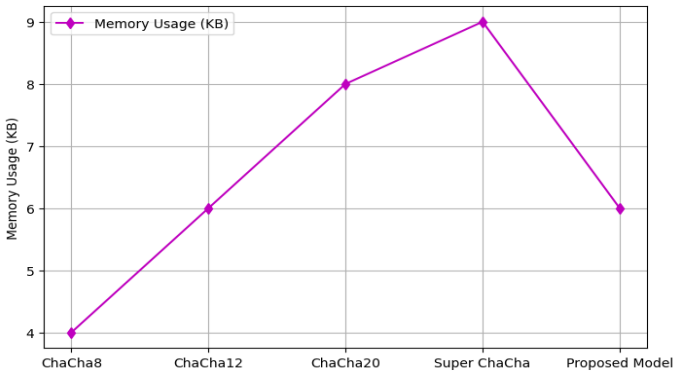


Figure 6. Memory Consumption(kb)

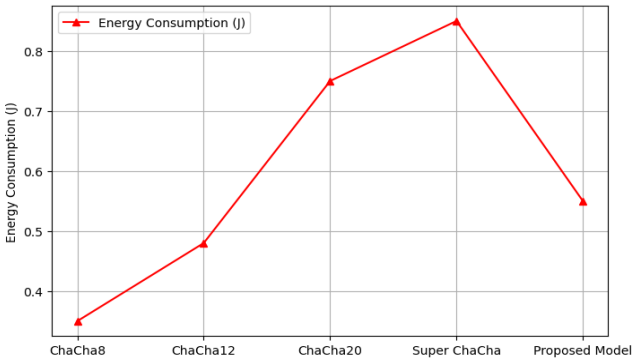


Figure 5. Energy Consumption (Joule)

4.8 Comparative Security Assessment

To further validate the effectiveness of the proposed model, a comprehensive security assessment was conducted against common cryptographic attacks. The results punctuate its superior adaptability compared to traditional encryption mechanisms. The security analysis confirms that Ours ChaCha20 constantly outperforms encryption norms, making it an optimal choice for coming-generation IoT security fabrics. It's also compared with ChaCha8 in the streamlined table 5, which, compared to another algorithm, the proposed model is better than others; it's better for all, which include a number of rounds, non-linearity, prolixity, and parallel processing.

Feature	**ChaCha20**	**ChaCha12**	**ChaCha8**	**Super ChaCha**	**Ours ChaCha**
Round	20	12	8	20	20
Non-linearity	Linear	Linear	Linear	Moderate	high
Diffusion	Moderate	Moderate	Weak	Improved	optimized
Parallel processing	No	No	No	No	Yes
NIST validation	Not Verified	Not verified	Not verified	No	yes
IoT Suitability	Moderate	High	High	High	Very high
Throughput	Moderate	High	Very high	Moderate	Very high
Energy efficiency	High	High	Very high	High	Very high

Table 5. Comparison of security

CONCLUSION AND FUTURE WORK

The improved ChaCha Encryption Algorithm makes the classic ChaCha cipher family more secure, efficient, and resource-conserving, thus rendering it appropriate for modern cryptographic applications. The proposed approach improves diffusion, mitigates uncertainty, and bolsters robustness against statistical and differential attacks by the application of a non-linear transformation function and a proprietary permutation layer. The performance analysis demonstrates outstanding security metrics, marked by reduced encryption time, efficient memory usage (6 KB), and low energy consumption (0.55 J), in addition to a significant avalanche effect (50.8462%) and enhanced Shannon entropy (0.99995). The proposed method, while maintaining strong cryptographic security, exhibits a significant decrease in encryption time relative to ChaCha8, ChaCha12, ChaCha20, and Super ChaCha. Its remarkable appropriateness for IoT security, real-time communication, and lightweight cryptographic applications arises from its ability to effectively handle encryption with low processing resources.

Future research will focus on improving computational efficiency through hardware acceleration methods, such as FPGA or GPU implementations, thereby addressing parallel processing and encryption speed. Additionally, we plan to improve the proposed method by implementing dynamic round configurations and adaptive key scheduling to strengthen security against evolving cryptographic attacks. The revised ChaCha algorithm will be assessed for its relevance in homomorphic encryption,

post-quantum cryptography, and blockchain security inside next-generation secure communication networks. A formal security study and practical deployment testing will be performed to validate its robustness against advanced cryptanalytic techniques, demonstrating its usefulness in many security-critical scenarios.

Funding: No funding available.

Data availability: The dataset generated and/or analyzed during the current study is accessible in the KAGGLE repository, https://www. kaggle.com/datasets/crawford/20-newsgroups.

Code availability: The code produced and/or examined during the present study can be obtained from the corresponding author upon request.

Declarations

Conflict of interest: The authors declare that they have no conflicts of interest.

Ethical approval: The authors confirm that no studies involving human participants were conducted for this article.

Informed consent: The authors state that no studies involving human participants were conducted for this article.

REFERENCES

[1] D. Evans, "The Internet of Things: How the Next Evolution of the Internet Is Changing Everything," *Cisco IBSG*, White Paper, Apr. 2011.

[2] A. Whitmore, A. Agarwal, and L. Da Xu, "The Internet of Things—A Survey of Topics and Trends," *Information Systems Frontiers*, vol. 17, no. 2, pp. 261–274, 2015.

[3] S. Sicari, A. Rizzardi, L. Grieco, and A. Coen-Porisini, "Security, Privacy and Trust in Internet of Things: The Road Ahead," *Computer Networks*, vol. 76, pp. 146–164, Jan. 2015.

[4] H. Suo, J. Wan, C. Zou, and J. Liu, "Security in the Internet of Things: A Review," in *Proc. 2012 Int. Conf. on Computer Science and Electronics Engineering*, pp. 648–651.

[5] Y. Liu, Y. Xiao, and S. Chen, "Authentication and Access Control in the Internet of Things," in *Proc. 2012 IEEE Int. Conf. on Distributed Computing Systems Workshops*, pp. 588–592.

[6] A. B. Nassar et al., "Lightweight Cryptography Algorithms for Securing IoT Devices," *Int. J. of Adv. Computer Science and Applications*, vol. 9, no. 11, pp. 136–143, 2018.

[7] D. J. Bernstein, "ChaCha, a Variant of Salsa20," *SASC 2008 – The State of the Art of Stream Ciphers*, 2008.

[8] T. H. Noor and Q. Z. Sheng, "A Survey on Security for Mobile Cloud Computing," *Journal of Network and Computer Applications*, vol. 61, pp. 59–81, 2016.

[9] R. H. Jhaveri, "Security Issues and Challenges in the Internet of Things," in *Proc. 2016 Int. Conf. on WiSPNET*, pp. 1861–1866.

[10] A. Khalique, B. Nazir, M. A. Khan, and K. Saleem, "Lightweight Cryptographic Algorithms for Resource-Constrained Devices: A Review," *IEEE Access*, vol. 6, pp. 26257–26286, 2018.

[11] D. J. Bernstein, "ChaCha, a Variant of Salsa20," *Workshop Record of SASC 2008 – The State of the Art of Stream Ciphers*, 2008.

[12] M. Ismail and T. S. Gunawan, "A Review of Lightweight Cryptography for the Internet of Things," *Telkomnika*, vol. 17, no. 1, pp. 51–58, Feb. 2019.

[13] K. Wang, J. Zhao, and Q. Zhu, "Lightweight Video Encryption Based on Hybrid Chaotic System and ChaCha20," *Multimedia Tools and Applications*, vol. 81, pp. 17435–17452, 2022.

[14] B. Lyu et al., "Decentralized Access Control for Smart Environments with Chaotic Encryption," *IEEE Access*, vol. 8, pp. 75604–75619, 2020.

[15] E. Ochoa, "Software Optimization of ChaCha Stream Cipher Using AVX2 and AVX512 Instructions," *Cryptography*, vol. 4, no. 3, pp. 1–16, Sep. 2020.

[16] M. H. Miraz, "Lightweight Image Encryption Using ChaCha20 and Serpent Hybrid Cipher," in *Proc. Int. Conf. on Information Security and Cryptology*, pp. 112–117, 2021.

[17] A. Orsini and J. Patarin, "Cryptanalysis of Reduced Round ChaCha Using Divide-and-Conquer Strategy," *IACR Cryptology ePrint Archive*, 2020:443.

[18] R. Sharma and D. Agrawal, "Super ChaCha: A Secure Stream Cipher for IoT Devices," *Int. J. of Security and Networks*, vol. 15, no. 2, pp. 101–110, 2020.

[19] P. Singh and A. Kumar, "Hardware Efficient Implementation of ChaCha20 for IoT," *Microprocessors and Microsystems*, vol. 78, pp. 103208, 2020.

[20] S. Pradhan and K. Dasgupta, "Extended ChaCha20: Enhancing the ARX Structure for Better Security," *Journal of Information Security and Applications*, vol. 58, pp. 102744, 2021.

[21] F. Khan, R. Asim, and T. Ali, "A Hybrid Image Encryption Method Using ChaCha and Hyperchaotic Maps," *Arabian Journal for Science and Engineering*, vol. 47, pp. 1995–2010, 2022.

[22] L. Zhu and Z. Wang, "Hybrid Image Encryption Using AES, ChaCha20, and GOST for Secure Communication," *IEEE Access*, vol. 10, pp. 102501–102514, 2022.

[23] Y. Lu and M. Zheng, "Modified ChaCha Core for Enhanced Diffusion in Cryptographic Primitives," *IET Information Security*, vol. 15, no. 5, pp. 379–387, 2021.

[24] M. Rehman et al., "SK4OA: A Secret Key Optimization Algorithm Using Non-linear Runtime Patterns," *Journal of Systems Architecture*, vol. 120, pp. 102283, 2021.

[25] Fritz, R. and Zhang, P. (2018). *Modeling and detection of cyber attacks on discrete event systems*. IFAC-PapersOnLine, 51(7), 285–290.

[26] Elkamchouchi, H. M., Takieldeen, A. E., & Shawky, M. A. (2018). *A modified serpent-based algorithm for image encryption*. In 2018 35th National Radio Science Conference (NRSC), IEEE. https://doi.org/10.1109/NRSC.2018.8354369

[27] Weinstein, D., Kovah, X., & Dyer, S. (2012). *SeRPEnT: Secure remote peripheral encryption tunnel*. The MITRE Corporation, Technical Report MP120013.

[28] Bernstein, D.J. (2008b). *The Salsa20 family of stream ciphers*. In New stream cipher designs, Springer, pp. 84–97.

[29] Aumasson, J.P., Fischer, S., Khazaei, S., Meier, W., & Rechberger, C. (2008). *New Features of Latin Dances: Analysis of Salsa, ChaCha, and Rumba*. FSE 2008, LNCS 5086, pp. 470–488. https://doi.org/10.1007/978-3-540-71039-4_30

[30] Bernstein, D.J. (2008). *Which phase-3 estream ciphers provide the best software speeds?* http://www.ecrypt.eu.org/stream

[31] Coutinho, M. and Neto, T.C.S. (2021). *Improved Linear Approximations to ARX Ciphers and Attacks Against ChaCha*. In EUROCRYPT 2021, LNCS 12696, pp. 711–740. https://doi.org/10.1007/978-3-030-77870-5_25

[32] Maitra, S. (2016). *Chosen IV cryptanalysis on reduced round ChaCha and Salsa*. Discrete Applied Mathematics, 208, pp. 88–97. https://doi.org/10.1016/j.dam.2016.02.020

[33] Twum Frimpong, J.B.H.A., Missah, Y.M., Dawson, J.K., et al. (2021). *Advanced ChaCha20 Algorithm with clustering-based security*. PLOS ONE.

Journal of Information Systems Engineering and Management
2025, 10(57s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

[34] Zhu, L. & Wang, Z. (2022). *Hybrid encryption using AES, ChaCha20, and GOST for image data*. IEEE Access.

[35] Lu, Y. & Zheng, M. (2021). *Modified ChaCha Core for Enhanced Diffusion*. IET Information Security, 15(5), 379–387.

[36] Rehman, M., Biswas, P.P., Fan, Y., & Chang, E.C. (2021). *SK4OA: A Security Key Optimization Algorithm Using Non-linear Runtime Patterns*. Journal of Systems Architecture, 120, 102283.