Dynamic Lightweight Encryption for Securing Data in Transmission Phase

Haider H. al-Mahmood[1] and Saad N.Alsaad[2]

[1] Informatics Institute for Postgraduate Studies, Iraqi Commission for Computers and Informatics, Baghdad, Iraq

[2]Department of Computer Science, College of Science, University of Mustansiriyah, Baghdad, Iraq

[1)] Corresponding Author: phd202130676@iips.edu.iq

[2)] dr.alsaadcs@uomustansiriyah.edu.iq

Abstract

The exponential expansion of the Internet of Things (IoT) and the extensive utilization of embedded systems, such as health trackers and medical gadgets, pose substantial difficulties in ensuring data security, particularly throughout the process of transmission. Conventional cryptographic systems, albeit being very safe, are not ideal for these devices since they consume a significant amount of power. Lightweight Cryptography (LWC) is a practical option that achieves a balance between security and efficiency for devices with limited resources.

This study examines LWC algorithms, with a specific focus on three stream ciphers that have been authorized by NIST: Grain, Trivium, and MICKEY. The comparison of these ciphers is based on factors such as key size, initialization vector (IV) size, design objectives, core features, and security attributes. The examination emphasizes the appropriateness of each cipher for different applications, especially in resource-constrained contexts. In addition, a thorough literature analysis investigates progress made in lightweight stream ciphers, indicating areas where further research is needed and potential avenues for future study. The study highlights the necessity for effective and reliable encryption solutions specifically designed to meet the limitations of IoT devices.

A new approach for generating dynamic keys is proposed to improve security in data transmission for Internet of Things (IoT) applications. The approach exhibits resilience against diverse attacks and successfully clears NIST randomness tests, guaranteeing elevated levels of security and efficiency. This research highlights the urgent requirement for optimal LWC algorithms to ensure the security of the ever-changing landscape of IoT and embedded devices.

Keywords: Lightweight Cryptography (LWC), Dynamic Key Generation, Stream Cipher, Security and Efficiency, NIST Randomness Tests, IoT devices security

1. Introduction

The rapid growth of the Internet of Things (IoT) in today's lives and the utilization of embedded systems such as health trackers and medical devices have brought a new challenge: securing data from such limited resources.[1], especially in a transition phase. Although conventional cryptosystems provide a higher security level, they are considered unsuitable for such devices due to the limited amount of power embedded systems consume[2]. The best practice that can manage the tradeoff between providing a suitable security level and working well with limited-resource devices is Lightweight Cryptography (LWC)[3].

LWC algorithms are designed to provide an acceptable level of security for encryption while minimizing processing power and memory consumption compared to conventional algorithms. Unsecured devices are vulnerable to various attacks, such as data theft or breaches. LWC's primary goal is to protect data at the point of generation and during transmission [4], particularly for systems with limited capabilities. In the healthcare industry, LWC plays a crucial role in securing data generated by medical devices and ensuring the privacy of patient data.

It is worth mentioning that LWC is not limited to medical devices; it can secure data transmitted through environments like remote computing [5] and basic mobile phones[6] [7].

2. Light Stream Cipher Approaches

Lightweight encryption has been extensively researched to make significant improvements that balance computational cost reduction against security. Objectives are focused on either lowering the calculation time while keeping the security level or increasing the security level with the same calculation cost.

To prioritize essential factors, we conducted a comparative analysis of three stream ciphers: GRAIN, Trivium, and MICKEY, as explained by [8] [9], and [10].

According to [8], the research examines the Grain family of efficient stream ciphers in resource-constrained situations. The analysis covers security and how well various ciphers fight against cryptographic attacks. The authors of [9] studied architecture and differential fault analysis (DFA) of Trivium stream cipher ASIC implementations. The study proves Trivium's fault vulnerability and retrieves the secret key in all tests. [10]assessed performance and security against classical cryptanalysis and side-channel attacks e for the low-power, minimum logic gate cipher family. MICKEY excels in resource-constrained hardware contexts but needs to improve in high-speed ones.

The three significant algorithms are compared to indicate exciting areas with similarities and differences. The comparison factor was determined by considering the key size, the initialization vector (IV) size, the design objectives, the main characteristics, and the common uses. Table (1) provides comprehensive insights into each cipher, including its construction, intended application, comparative advantages, and security goals.

Table 1. Comparison of Lightweight Stream Ciphers (Grain, Trivium, and MICKEY [1,2,3])

Feature	Grain	Trivium	MICKEY
Key Size	80 bits	80 bits	80 bits, 128 bits (MICKEY-128)
IV Size	64 bits	80 bits	Varies, up to 80 bits
Design Goals	High security, low hardware complexity	The simplicity and flexibility of hardware implementation	High security, low complexity of hardware
Algorithm Type	LFSR and NFSR stream ciphers	A stream cipher with three shift registers	A stream cipher with irregular shift registers clocking
Security Features	A resistance to known attacks, including those based on correlation and algebra	As secure as using a one-time pad; resistant to common attacks	Faster than exhaustive key search; no deliberate weaknesses
Primary Features	Flexibility in hardware resources, adjustable speed	Hardware-efficient, low-power consumption	A unique clocking technique to enhance security

Typical Applications	Resource-limited environments such as RFID tags	Environments that require simplicity and low power	Hardware platforms with limited resources, for instance RFID systems
Security Analysis	A robust solution to various cryptanalytic methods	Provides minimal susceptibility to linear correlations	Provides resistance to statistical attacks and complex feedback mechanisms
Hardware Efficiency	A highly efficient system with a high throughput potential, depending on the hardware	Incredibly efficient at generating multiple bits per clock cycle	The design is optimized for minimal hardware use, making it suitable for low-power applications

2. Literature Review

This literature review provides a comprehensive review of the current state of lightweight encryption research. It will review various approaches and methodologies that have been proposed and analyze the strengths and weaknesses of different encryption schemes. This review will analyze existing literature to identify gaps and opportunities for future research. The goal is to contribute to the development of encryption solutions that are both efficient and safe, designed explicitly for resource-constrained applications. The following paragraphs refer to the different studies in the context of lightweight encryption.

- Stream cipher technique was specifically developed for IoT devices with limited resources as stated by [11]. The method employs a dynamic key-dependent strategy to attain strong security while minimizing overhead through basic procedures. The suggested encryption guarantees little transmission of errors, cheap additional costs, and a more straightforward execution, making it appropriate for small-scale devices. Authors of [12] asserted a new efficient stream cipher that combines a chaotic system with two Nonlinear Feedback Shift Registers (NFSRs). The cipher exhibits strong cryptographic properties as evidenced by entropy analysis and NIST statistical tests.
- In the analysis by [13] the study presented a keystream generator that generates secret keys using YouTube thumbnails as an entropy source. The primary discovery is that the suggested technique offers a secure and efficient way for generating keys in near-field communication (NFC) devices, utilizing the inherent unpredictability of images for entropy. The paper published by [14] highlighted a strategy for protecting social data by employing lightweight and selective EBCOT (Embedded Block Coding with Optimized Truncation) coding. The method aims to ensure security while keeping computing requirements minimal, making it well-suited for contexts with limited resources, such as the Internet of Things (IoT). It employs a selective encryption technique for essential data blocks, balancing security and efficiency.
- [15] demonstrated exceptional key sensitivity and unpredictability, as verified by rigorous statistical studies such as entropy analysis, PDF analysis, and correlation testing. Validation experiments were conducted to confirm the system's resistance to well-known attacks such as statistical, differential, chosen/known plain-text, and brute-force attacks. Utilizing a dynamic key strategy incorporating dynamic substitution, permutation, and diffusion layers has dramatically augmented the cipher's security. In the article [16], research introduced a low-weight stream cipher that utilizes a Linear Feedback Shift Register (LFSR) and a Feedback with Carry Shift Register (FCSR). The design guarantees robust security and protection against various attacks, such as meet-in-the-middle,

algebraic, exhaustive, differential, and correlation attacks. [17] presented a comprehensive experimental demonstration of successfully breaking Trivium ciphers built using ASIC technology. The study illustrates the susceptibility of Trivium to differential fault analysis, successfully retrieving the secret key in all examined situations with minimal assumptions and under real-world circumstances. According to [18] , the research examined the hardware implementation of the Enocoro128v2 stream cipher, focusing on its small and efficient design for limited embedded systems. Authors [19] provides evidence for novel joint encryption-modulation (JEM) techniques for phase encryption in IoT sensor transceivers. JEM demonstrates minimal complexity and high performance while maintaining a satisfactory packet error rate (PER) and bit error rate (BER) performance. This makes it well-suited for several modulation types, including high-order modulations such as 64-256 QAM.

- As investigated by [20], the suggested system includes a structure similar to Grain and an extra key filter to safeguard against typical cryptanalytic techniques. The design's security and efficiency have been demonstrated through thorough hardware implementations and cryptanalysis. The research [21] demonstrated the notion of "perfect trees" to develop energy-efficient symmetric encryption methods. The primary innovation is a framework designed to optimize encryption algorithms regarding energy usage, rendering them well-suited for energy-limited contexts. Authors [22] discussed the necessity of effective data transmission in the Internet of Things (IoT) through lightweight cryptography. The suggested approach proposes a technique to decrease the amount of data by utilizing compression and substituting SSL/TLS with a more lightweight cryptographic method based on the Vernam cipher principle.
- As founded by [23], cryptographic algorithms that utilize chaos theory has the ability to attain a high level of security while keeping hardware complexity minimal, making them well-suited for Internet of Things (IoT) applications. And when it is implemented on a Field-Programmable Gate Array (FPGA). In the study of [24] the DRACO stream cipher is introduced as an energy-efficient solution with a compact state size and verifiable security against time-memory-data tradeoff (TMDTO) attacks. The primary results highlight DRACO's efficacy and robustness, rendering it well-suited for low-power cryptographic applications. As per [25], authors suggested a new self-shrinking (SSG) generator called the Self-Shrinking Conflation Generator (SSCG). The primary outcome is that SSCG strengthens security by merging discarded and kept bits via XOR operations, hence enhancing resilience against different cryptanalytic assaults. As mentioned by [26] ,the research improved the Salsa20 stream cipher by using random chaotic maps to promote diffusion. The enhanced cipher exhibits enhanced performance and diffusion properties, ensuring security while boosting encryption speed. As per [27] the research explored enhancements in lightweight designs for the SNOW-V cipher, aiming to achieve superior performance in limited contexts. Concentrate on optimizing the execution to improve efficiency and minimize resource use.
- [28] emphasized improving the software implementation of SNOW-V for 32-bit platforms with restricted resources. SNOW-V is specifically engineered as a pseudorandom number generator, primarily focusing on its application in 5G communications.
- [29] argued the Rabbit Algorithm and Aizawa Attractor-based image encryption method. In this algorithm, color images are blocked and encrypted with chaotic Aizawa Attractor keys in this hybrid technique, boosting security and minimizing computer load. PSNR, MSE, SSIM, and NIST demonstrate the method's efficacy and attack resilience. Real-time applications like IoT benefit from this lightweight, effective solution. Fast and secure image encryption is ensured.
- Author [30] introduced a streamlined authentication encryption system integrating stream ciphers with chaotic maps within a sponge structure. This system's primary advantage is its appropriateness

for Internet of Things (IoT) applications, as it offers robust security while requiring minimal computational resources.

- According to the research of [31], this study investigates several arrangements of the Espresso stream cipher, assessing their efficacy and robustness for optimal utilization in resource-limited settings. As observed by [32], a rigorous examination of stream ciphers' security against general assaults provided verifiable security assurances. The primary outcome is the determination of limits on adversaries' advantage, which guarantees the resilience of the examined stream ciphers.

3. Problem Statement

The processing and transmission of massive data volumes often raise several security issues. Since many of these devices use limited power and other resources, traditional cryptographic systems are inappropriate due to their high energy and resource consumption. This means a critical need for a workable solution that can also preserve the limited resources of these devices. The resource investigated did not perform dynamic key generation, so if a ciphertext is broken, the encryption key will no longer protect the transmitted data. To deal with the problem of encryption key breaches, the paper explores a new algorithm for increasing security through "dynamic" key generation.

4. Aims

Create and optimize a Dynamic Lightweight Cryptography (LCW) algorithm that provides an acceptable level of encryption security without consuming excessive processor power or memory, especially compared to conventional cryptographic algorithms. The goal could improve the broad application range for LCW application, such as medical devices, remote computing, and securing data transmission on mobile phones.

5. Methodology

Initially, a one-time pad with five million random byte values between 0 and 255 is created for the investigation. This configuration must satisfy two essential requirements: firstly, it must consistently produce the identical sequence of values whenever executed with the same initial parameters, ensuring that both the sender and recipient possess the same one-time pad. Furthermore, the system needs to have the ability to produce a distinct sequence of values when the initial pseudo-random number (seed) used to construct the sequence is altered. Thus, if the encrypting side, A, and party B (the decrypting side) are given the same seed, they will produce identical sequences. Figure (1) represents the block diagram that abstracts the cryptanalysis system.

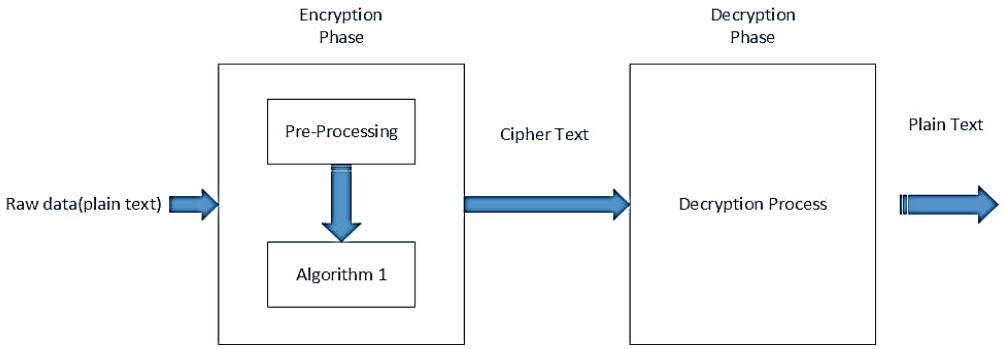

Fig. (1) The block diagram for the Encryption-decryption phases

Algorithm 1: general system

- Input: data file
- Output: array of ciphered record and its related hash digits

Steps:

1. Read input data file as byte
2. Divide the input data file into records with equal length
3. Calculate the Hash Digits for each record
4. Apply encryption algorithm for each record
5. Attach and send the hash digit with its related ciphered record
6. At the receiver side, calculate the hash digit for the deciphered record
7. Compare between calculated Hash Digit at the receiver side and the received hash digit from the sender
8. If the hashes are identical, then received records intact; otherwise, resend the modified record and its related hash digit

End

Figure (2) represents the Activity diagram for general algorithm 1. While figure (3) represents the activity diagram for the encryption process.

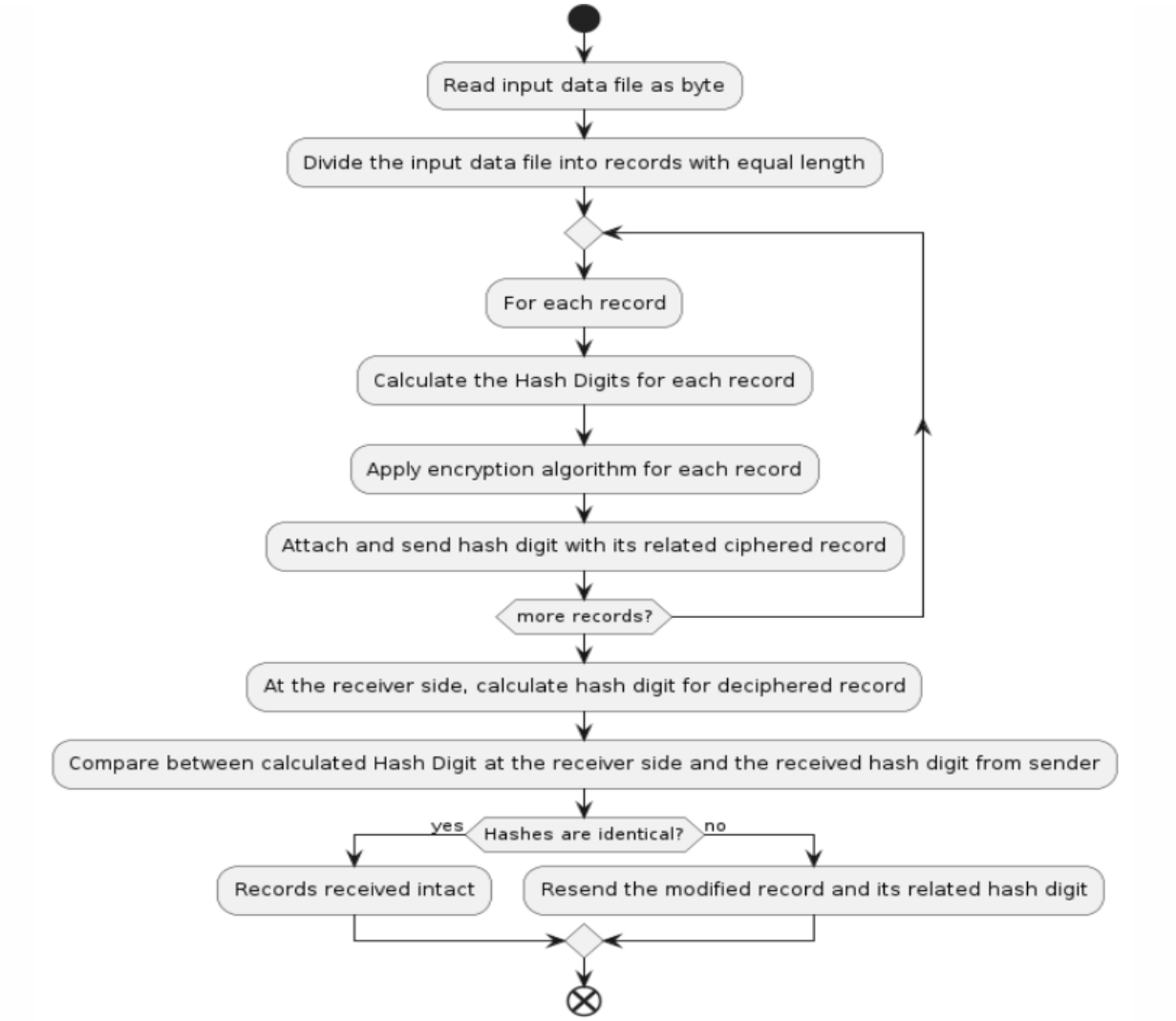

Fig. (2) The Suggested System Activity Diagram

Algorithm 2 (Encryption process)

1. Initialization Variables:
2. Input: Array of records
3. Output: Ciphered text

Steps

1. Convert record characters into integers between 0-255
2. Create a one-time pad with byte value (0-255)
3. For each character in the record
   a. Compute index value utilizing symbolic hashing (ensure maximum index equal or less than one-time pad size)
   b. Choose a one-time pad element according to the computed index
   c. XOR the plain text value with One-time pad vale to produce cipher text
   d. Update Index value (make sure the new index value is not equal to the previous value and does not exceed one-time pad maximum size)

3. Send the ciphered record to the receiver entity
4. End

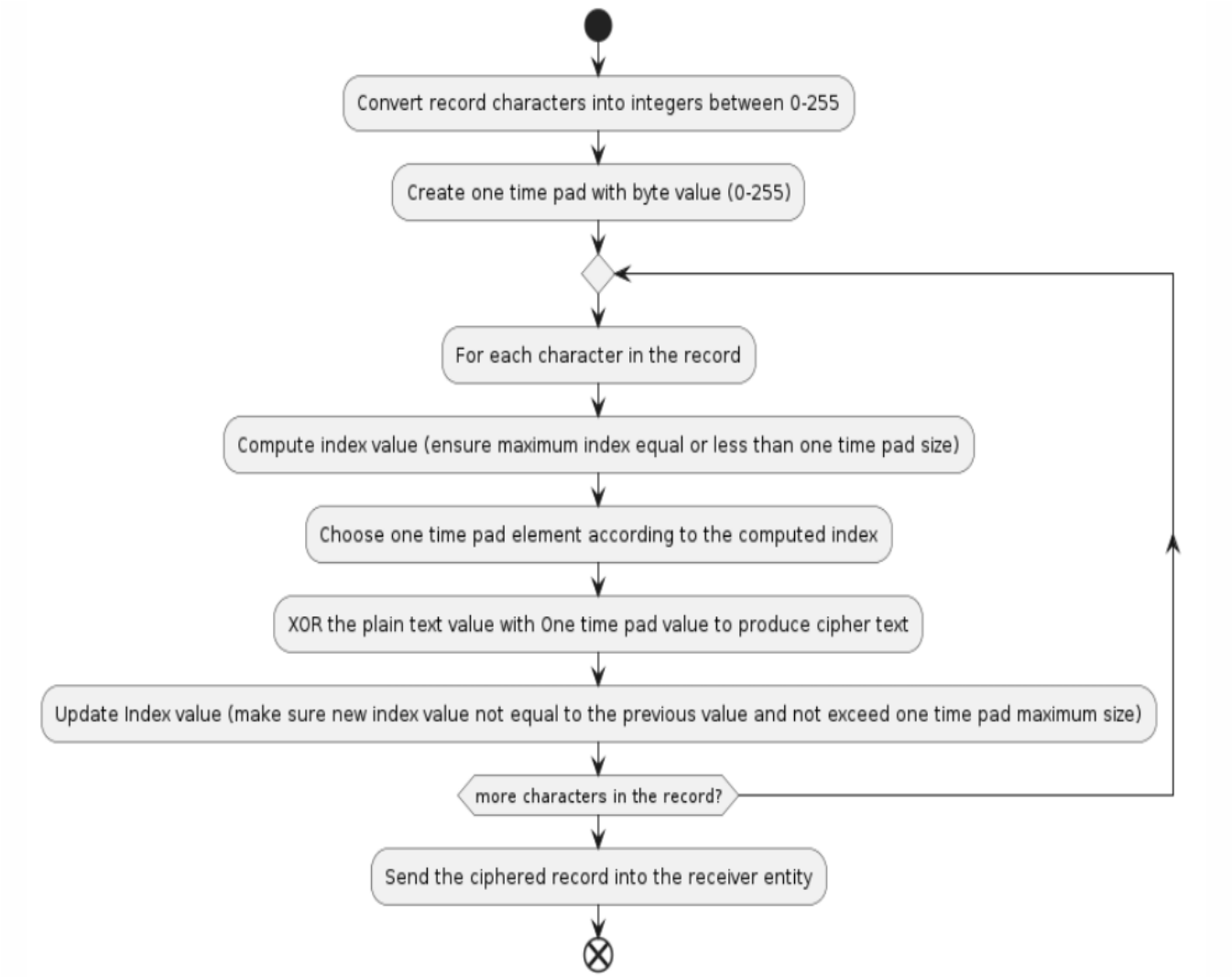

Fig. (3) Encryption process

The suggested encryption system can be considered as a hashing process; hashing final equation is designed to have multiple input values and arithmetic operations such as adding, multiplying, adding, ANDing, and Xor operations to produce index value within the maximum number not exceeding the one-time pad. For example,

r_1= (record number & long number1) * prime number1
r_2= (record number & long number2) * prime number1
..
..
r_6= (record number & long number6) * prime number1

j_1 = 2*[(prime number 2 * r1 + prime number 3 * r2) & (length of one-time pad_x)] % (length of one-time pad)

…
…

j_6 = 2*[(prime number 5 * r3) + (prime number 6 * 12) & (length of one-time pad_x)] % (length of one-time pad)

where, int x < length of the one-time pad, the system will choose the six values from the one-time pad stored in positions j_1, $j_2 \ldots j_6$. Then, these 6 values input to XoR operation between them produce only one value, which represents the key.
If we consider the one tie pad array named rg, then

key[i]=$rg_1[j_1]$ XoR $rg_2[j_2]$ XoR $rg_3[j_3]$ XoR $rg_4[j_4]$ XoR $rg_5[j_5]$ XoR $rg_6[j_6]$

where $0 \leq i \leq number\ of\ characters\ in\ plain\ text.$

The variable "record number" can be set to any distinct value, meaning that if an attacker compromises the input values, they can be changed instantly. The research utilized a data file containing 5000 records with 62 characters arranged in different lines or continuously. Characters encompass both alphabetical letters and unique symbols. The key is divided into two parts: the first part is a seed number used to generate the one-time pad, which should be 16 bits long. The remaining half is used in the encryption computation.

To validate the proposed method's effectiveness in generating secure data, the research conducted various randomness tests, such as the NIST randomness tests [33] and entropy calculation [34]. These tests play a crucial role in ensuring the security of the encrypted data. The encrypted data is transformed into a binary representation before a randomness test.

Cryptography resilience is crucially assessed by the correlation between the ciphertext and the plaintext and between several ciphertexts produced from the same plaintext using slightly different inputs. Determining those minor modifications in the inputs of the encryption system result in uncorrelated ciphertexts is of utmost importance. The Pearson correlation is used to evaluate this characteristic. This statistical metric provides a means to assess the level of correlation, therefore assuring that the encryption system generates outputs that exhibit a high level of resistance to patterns or predictability, even when minor modifications are introduced to the input variables. This guarantees the security of the encryption procedure, posing challenges for unauthorized organizations to derive significant conclusions from the ciphertext [35].

6. Results

Multiple sub-keys of varying lengths and numbers are employed to generate the master key, which is then utilized for encryption and decryption.

Table (2) presents the results achieved in each instance, where varying the number of sub-keys and the length of each base key led to different outcomes. The maximum number of sub-keys required to ensure compliance with cipher text security standards is also stated. The results showed the recommended length of the encryption key is between (52 -92) bits.

Table 2. Result of tests: The data file size is 300000 characters, and the Length of each record=62 characters

Key Length	Encryption time (ms)	NIST test	No of (0, zeros)	No. of (1, ones)	Ratio of (0)	Ratio of (1)	Entropy for 2,400,000 bits
98	75	Pass	1198784	1201216	0.49949	0.50050	0.99999925
76	89	Pass	1198088	1201912	0.49920	0.50079	0.99999816
64	118	Pass	1201987	1198013	0.50082	0.49917	0.99999807
52	99	Pass	1207354	1192646	0.50306	0.49693	0.99997290
40	89	Fail	1204975	1195025	0.50207	0.49792	0.99998760
28	83	Fail	1208011	1191923	0.50333	0.49666	0.99999678

The results obtained for implementing the suggested algorithm are listed in Table (3).

Table (3) NIST test results (file size =2400000 bits), key length= 98 bits

Test	p-value	state
Frequency	0.911413	pass
Block Frequency	0.739918	pass
Cumulative Sums	0.911413	pass
Run	0.534146	pass
Longest Run	0.213309	pass
Rank	0.004301	pass
FFT	0.350485	pass
Non-Overlapping Template	0.000001	pass
Overlapping Template	0.213309	pass
Universal	0.000000	pass
Approximate Entropy	0.122325	Pass at block 1500
Serial	0.534146	pass
Linear Complexity	0.122325	Pass

These results were obtained when The NIST test application was set up at the below initials:

- Significance level (α): 0.01;
- Size of the block in the frequency test within a block: 128;
- Size of the length in bits of each template in the non-overlapping template matching test: 9;
- Length in bits of the template in the overlapping template matching test: 9;
- The length of each block in the approximate entropy test: 10;
- The length in bits of each block in the serial test: 16;
- The length in bits of a block in the linear complexity test: 500;

Furthermore, plain text values range between 10–122, containing 50 distinct values, while cipher text has a full range from (0 to 255). The histogram clearly illustrates the distribution of contrasting original

and ciphered values. The distribution of the frequent values in the original data, as depicted in Figure (4), is entirely dissimilar to that observed in the ciphered data file, as seen in Figure (5).

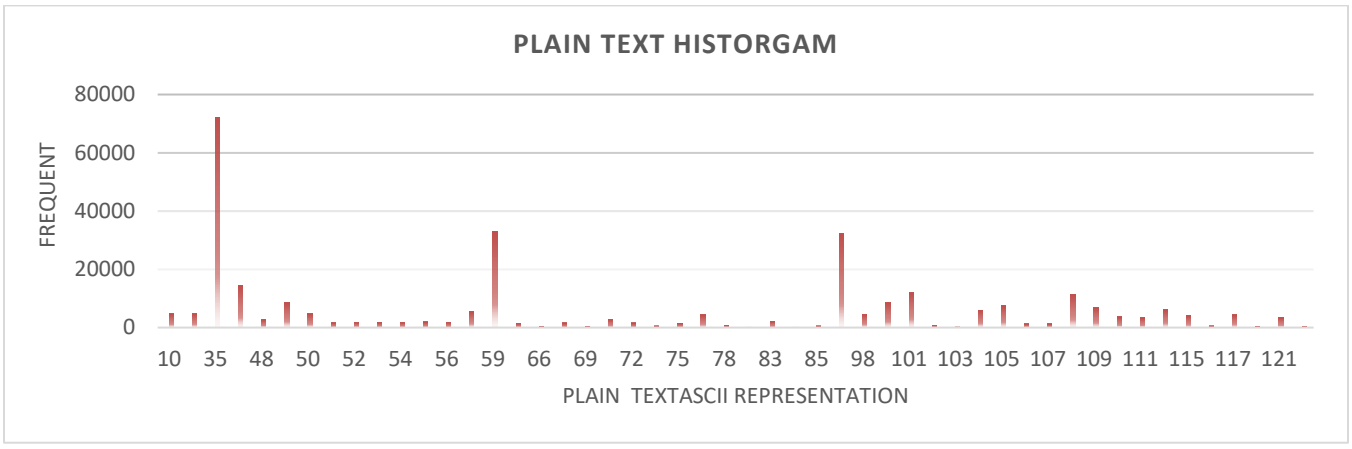

Fig (4) Histogram for original data representation

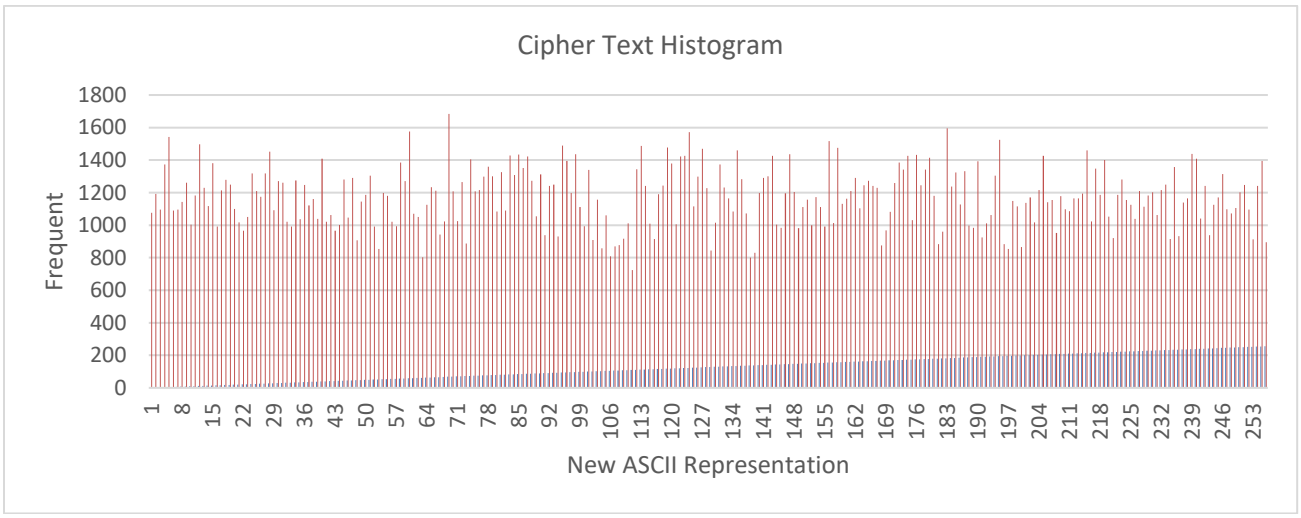

Fig. (5.) Histogram for ciphered data representation

Pearson correlation test was performed to evaluate the resilience of the encryption process by analyzing the correlation between the ciphered values following minor modifications to one or more input variables. Table 4 displays the Pearson correlation coefficients that correlate to the fluctuations in the different input variables.

Table 4: Pearson Correlation Values Based on Changes in Input Variables

Variable	Old value	New Value	Pearson
prime number1	201	711	-0.017736696
prime number1	711	1009	-0.007301424
long number2	xr2 = "0xaca"	xr2= "0xffc"	0.030874208
long number3	xr3 = "0xffb"	xr3 = "0xaca"	0.073759102
seed	254521	125430	-0.020805635

Below is a quantitative examination of the outcomes based on the Pearson values obtained while modifying the inputs:

1. Prime number 1 (Values: 201 → 711, Pearson correlation coefficient: -0.017736696) Based on the Pearson correlation coefficient of -0.0177, it may be inferred that the change in the input from 201 to 711 led to a negligible linear connection between the original and the ciphered data. The subtle negative sign indicates a very feeble inverse correlation. Thus, the encryption process seems resilient against this perturbation, as the correlation is virtually non-existent.
2. Prime number 1 (Values: 711→1009, Pearson correlation coefficient: -0.007001424) The Pearson correlation coefficient of -0.0073 is nearly zero, mirroring the earlier finding. This observation suggests a much less pronounced negative correlation (almost insignificant) between the original and ciphered numbers when the input changes from 711 to 1009. The encryption process retains robustness since the ciphered data do not closely correlate with the input change.
3. Long number2 (Values: xr2="0xaca" *Æ* "0xffc", Pearson correlation coefficient: 0.030874208) An input change from "0xaca" to "0xffc" results in a very weak positive correlation between the original and ciphered values, as seen by the Pearson correlation coefficient of 0.0309. Although there is little inclination for the ciphered values to rise with higher input values, the correlation is so feeble that it is nearly statistically negligible. These findings indicate that the encryption mechanism remains mostly unaltered by this particular modification in the input.
4. Long integer 3 (Values: xr3="0xffb" "0xaca", Pearson correlation coefficient: 0.073759102) The obtained Pearson correlation coefficient of 0.0738 indicates a modest positive association, somewhat more pronounced than in earlier instances but yet negligible. These findings indicate a limited inclination for the ciphered values to rise when the input value transitions from "0xffb" to "0xaca". Although a modest positive correlation exists, it lacks sufficient strength to suggest a substantial weakness in the encryption procedure.
5. Seed (Values: 254521 - 125430, Pearson correlation coefficient: -0.020805635)

An analysis of the Pearson correlation coefficient of -0.0208 indicates a minimal negative correlation between the original and ciphered values as the seed value varies from 254521 to 125430. The correlation

is nearly negligible, suggesting that changes in the seed have minimal or no effect on the correlation between the input and the ciphered output. This indicates that the encryption process is highly resilient to variations in the seed.

The Pearson correlation coefficients for the various input modifications are near zero, with only a few correlations detected. The negative Pearson coefficients indicate modest inverse correlations, whereas the positive coefficients indicate relatively weak direct correlations. Nevertheless, the correlations are negligible in every instance, suggesting that the encryption process is resilient and not significantly affected by minor variations in the input data. The low Pearson values indicate the intended attribute of an encryption method, in which the output (ciphered data) should not exhibit a substantial linear correlation with the input, preserving both security and unpredictability.

7. Conclusion

Conclusively, this work emphasizes the crucial need for Lightweight Cryptography (LWC) in safeguarding data in situations with limited resources, such as IoT devices and data transmitted via mobile device applications. This work aims to analyze and contrast current stream cipher algorithms and propose a dynamic, lightweight cryptography algorithm to meet the urgent requirement for secure and efficient encryption techniques appropriate for low-power applications. The capacity of the proposed approach to produce dynamic keys dramatically reduces the likelihood of encryption key breaches, improving overall security. The empirical findings from various tests, encompassing randomness and correlation evaluations, validate the algorithm's efficacy in upholding security while preserving performance. The present study contributes to continuously advancing secure and efficient encryption methods designed explicitly for resource-constrained settings. This discovery creates opportunities for broader use in healthcare, remote computing, and mobile communications.

References

[1] K. Shaukat, T. M. Alam, I. A. Hameed, W. A. Khan, N. Abbas, and S. Luo, "A Review on Security Challenges in Internet of Things (IoT)," in *2021 26th International Conference on Automation and Computing (ICAC)*, 2-4 Sept. 2021 2021, pp. 1-6, doi: 10.23919/ICAC50006.2021.9594183.

[2] M. Rana, Q. Mamun, and R. Islam, "Lightweight cryptography in IoT networks: A survey," *Future Generation Computer Systems*, vol. 129, pp. 77-89, 2022.

[3] A. Alahdal and N. K. Deshmukh, "A systematic technical survey of lightweight cryptography on IoT environment," *International Journal of Scientific & Technology Research,* vol. 9, no. 3, 2020.

[4] M. A. F. Al-Husainy, B. Al-Shargabi, and S. Aljawarneh, "Lightweight cryptography system for IoT devices using DNA," *Computers and Electrical Engineering,* vol. 95, p. 107418, 2021.

[5] Y. Guo, W. Liu, W. Chen, Q. Yan, and Y. Lu, "ECLBC: A Lightweight Block Cipher With Error Detection and Correction Mechanisms," *IEEE Internet of Things Journal,* 2024.

[6] M. A. Dar, A. Askar, D. Alyahya, and S. A. Bhat, "Lightweight and Secure Elliptical Curve Cryptography (ECC) Key Exchange for Mobile Phones," *International Journal of Interactive Mobile Technologies,* vol. 15, no. 23, 2021.

[7] S. Singh, P. K. Sharma, S. Y. Moon, and J. H. Park, "Advanced lightweight encryption algorithms for IoT devices: survey, challenges and solutions," *Journal of Ambient Intelligence and Humanized Computing,* pp. 1-18, 2024.

[8] M. Hell, T. Johansson, and W. Meier, "Grain: a stream cipher for constrained environments," *International journal of wireless and mobile computing,* vol. 2, no. 1, pp. 86-93, 2007.
[9] M. Robshaw and O. Billet, *New stream cipher designs: the eSTREAM finalists*. Springer, 2008.
[10] S. Babbage and M. Dodd, "The MICKEY stream ciphers," in *New Stream Cipher Designs: The eSTREAM Finalists*: Springer, 2008, pp. 191-209.
[11] H. Noura, R. Couturier, C. Pham, and A. Chehab, "Lightweight stream cipher scheme for resource-constrained IoT devices," in *2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2019: IEEE, pp. 1-8.
[12] L. Ding, C. Liu, Y. Zhang, and Q. Ding, "A new lightweight stream cipher based on chaos," *Symmetry,* vol. 11, no. 7, p. 853, 2019.
[13] P. Varalakshmi, R. Narayanan, K. Radhakrishnan, N. S. Prakash, A. Saranya, and J. Sivaramakrishnan, "YouTube-Based Keystream Generator—Secure and Lightweight Secret Key Generation for Near Field Communication Devices," in *2019 11th International Conference on Advanced Computing (ICoAC)*, 2019: IEEE, pp. 415-419.
[14] H. Qiu, M. Qiu, M. Liu, and Z. Ming, "Lightweight selective encryption for social data protection based on EBCOT coding," *IEEE Transactions on Computational Social Systems,* vol. 7, no. 1, pp. 205-214, 2019.
[15] R. Couturier, H. N. Noura, and A. Chehab, "ESSENCE: GPU-based and dynamic key-dependent efficient stream cipher for multimedia contents," *Multimedia Tools and Applications,* vol. 79, no. 19, pp. 13559-13579, 2020.
[16] N. A. Mohandas, A. Swathi, R. Abhijith, A. Nazar, and G. Sharath, "A4: A lightweight stream cipher," in *2020 5th international conference on communication and electronics systems (ICCES)*, 2020: IEEE, pp. 573-577.
[17] F. E. Potestad-Ordóñez, M. Valencia-Barrero, C. Baena-Oliva, P. Parra-Fernández, and C. J. Jiménez-Fernández, "Breaking trivium stream cipher implemented in asic using experimental attacks and dfa," *Sensors (Switzerland)*, Article vol. 20, no. 23, pp. 1-19, 2020, Art no. 6909, doi: 10.3390/s20236909.
[18] L. Pyrgas and P. Kitsos, "Compact hardware architectures of enocoro-128v2 stream cipher for constrained embedded devices," *Electronics (Switzerland),* Article vol. 9, no. 9, pp. 1-14, 2020, Art no. 1505, doi: 10.3390/electronics9091505.
[19] D. Long Hoang, T. Hong Tran, and Y. Nakashima, "A low complexity joint encryption-modulation method for iot sensor transceivers," *Electronics (Switzerland),* Article vol. 9, no. 4, 2020, Art no. 663, doi: 10.3390/electronics9040663.
[20] S. Banik *et al.*, "Atom: A stream cipher with double key filter," *IACR Transactions on Symmetric Cryptology,* Article vol. 2021, no. 1, pp. 5-36, 2021, doi: 10.46586/tosc.v2021.i1.5-36.
[21] A. Caforio *et al.*, "Perfect trees: designing energy-optimal symmetric encryption primitives," *Cryptology ePrint Archive,* 2021.
[22] I. Sokol, P. Hubinský, and Ľ. Chovanec, "Lightweight cryptography for the encryption of data communication of iot devices," *Electronics,* vol. 10, no. 21, p. 2567, 2021.
[23] Y. Guang *et al.*, "Chaos-Based Lightweight Cryptographic Algorithm Design and FPGA Implementation," *Entropy,* Article vol. 24, no. 11, 2022, Art no. 1610, doi: 10.3390/e24111610.
[24] M. Hamann, A. Moch, M. Krause, and V. Mikhalev, "The DRACO stream cipher: a power-efficient small-state stream cipher with full provable security against TMDTO attacks," *IACR transactions on symmetric cryptology,* vol. 2022, no. 2, pp. 1-42, 2022.

[25] V. Kanth, T. Martinsen, and P. Stanica, "The Self-Shrinking Conflation Generator: A Proposed Improvement to the Self-Shrinking Generator," *European Journal of Pure and Applied Mathematics,* vol. 15, no. 4, pp. 1426-1443, 2022.
[26] I. Alshawi and L. Muhalhal, "Improved Salsa20 stream cipher diffusion based on random chaotic maps," *Informatica,* vol. 46, no. 7, 2022.
[27] A. Caforio, F. Balli, and S. Banik, "Melting SNOW-V: improved lightweight architectures," *Journal of Cryptographic Engineering,* vol. 12, no. 1, pp. 53-73, 2022.
[28] J. M. Gil, Ó. C. Álvarez, Y. G. González, and I. M. Corbella, "Improving the lightweight implementation of SNOW-V," *Wireless Networks,* pp. 1-13, 2023.
[29] M. G. Alwan, E. T. Khudair, and E. F. Naser, "A Hybrid Algorithms Based on the Aizawa Attractor and Rabbit-Lightweight Cipher for Image Encryption," *Iraqi Journal of Science,* pp. 6534-6547, 2023.
[30] R. S. Mohammed, "Design a Lightweight Authentication Encryption Based on Stream Cipher and Chaotic Maps with Sponge Structure for Internet of Things Applications," *International Journal of Intelligent Engineering & Systems,* vol. 16, no. 1, 2023.
[31] Z. Shi *et al.*, "Design space exploration of galois and fibonacci configuration based on espresso stream cipher," *ACM Transactions on Reconfigurable Technology and Systems,* vol. 16, no. 3, pp. 1-24, 2023.
[32] A. Moch, "Provable security against generic attacks on stream ciphers," *Journal of Mathematical Cryptology,* vol. 17, no. 1, p. 20220033, 2023.
[33] E. Almaraz Luengo and J. Román Villaizán, "Cryptographically Secured Pseudo-Random Number Generators: Analysis and Testing with NIST Statistical Test Suite," *Mathematics,* vol. 11, no. 23, p. 4812, 2023.
[34] M. Ribeiro *et al.*, "The entropy universe," *Entropy,* vol. 23, no. 2, p. 222, 2021.
[35] E. Saccenti, M. H. Hendriks, and A. K. Smilde, "Corruption of the Pearson correlation coefficient by measurement error and its estimation, bias, and correction under different error models," *Scientific reports,* vol. 10, no. 1, p. 438, 2020.