





Cryptography by the Echacha20 Algorithm Based on Logistic and Chirikov Chaotic Maps

Ibtesam Jomaa¹ ^{*}, Rana Jassim Mohammed² ², Noor Abdulmuttaleb Jaafar ³ ³,
Hussein Ali Ismael⁴ ⁴

^{1,2,3} Department of Computer Science, Presidency of Diyala University, Diyala, Iraq

⁴ Department of Computer Engineering Techniques, Bilad Alrafidain University College, Diyala, Iraq, 32001

¹ibtesam.jomaa.h@uodiyala.edu.iq, ²ad.ranamohammed@uodiyala.edu.iq
³noor84abd84@uodiyala.edu.iq, ⁴husseinhair89@yahoo.com

Abstract

Nowadays, lightweight cryptography attracts academicians, scientists and researchers to concentrate on its requisite with the increasing usage of low resource devices. In this paper, a new lightweight encryption scheme is proposed using the chaotic map. This encryption scheme is an addition–rotation–XOR block cipher designed for its supremacy, efficacy and speed execution. In this addition–rotation–XOR cipher, the equation for chaotic map is iteratively solved to generate unique random numbers in a speedy manner using the logistic and Chirikov map. Chaotic maps, encryption algorithms, and cryptography are three approaches that are frequently used to safeguard digital data from unauthorized access and use. Chacha20 is a lightweight encryption algorithm, fast and secure and provides a balance between high security and little complexity and execution time the addition, in this work the development of the Chacha20 algorithm is used to provide the required security for data transmission.

Therefore, we created a randomness key to power the algorithm against various attacks Using the chaotic map to generate a random key for the encryption/decryption operations to improve the diffusion of the ChaCha20 cryptography algorithm's stream secret key. Finally, the cipher results are constructed from the input data and evaluated with various statistical as well as randomness tests correlation coefficient, SNR, and UAIC metrics prove that the proposed enhancement of the Chacha20 stream cipher algorithm (EChacha20) with chaotic addition–rotation–XOR stream cipher is efficient in terms of randomness and speed. For the end discussed complete models with security measures in this research.

Keywords: Cryptography, Chacha20, Chaotic Map, Correlation Coefficient, Signal Noise Ratio, Unified Average Change Intensity.

Article history: Received : 28/9/2023 Accepted : 5/1/2024

This article is open-access under the CC BY 4.0 license (<http://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Through internet connectivity, enormous amounts of data are sent daily in many different disciplines. Sometimes the sender sends the data to the receiver using unreliable channels or means. This makes a massive mess and compromises a great deal of sensitive data. Since it is such an important subject, several encryption techniques are being developed and utilized by both public and private enterprises to prevent confidential information from being compromised. Cryptography is an important tool for keeping sensitive data safe from outsiders [1]. Using plain text or code with encryption and decryption, cryptography ensures or protects

information and communications so that only the intended recipient and sender, or those for whom the information is meant, may read and process it. Data security is being strengthened, and data is being protected from unauthorized access. Before granting any user access to any system, cryptography stipulates a code or other requirement. After the condition or code is satisfied, the sender and receiver can access any system [2].

Modern cryptography techniques guarantee the data's confidentiality, integrity, and authentication. Three other categories of cryptography exist: symmetric (private key), asymmetric (public key), and hash function.

* Corresponding Email: ibtesam.jomaa.h@uodiyala.edu.iq

Asymmetric cryptography uses two separate keys, whereas the symmetric cryptography algorithm uses a single key for both encryption and decryption. The cryptographic hash function accepts variable-length inputs and produces results with fixed-length outputs. In general, symmetric key algorithms are chosen because they can be executed electronically considerably more quickly than asymmetric key algorithms. The symmetric key algorithms are divided into two types: block and stream cipher [1]. The term (one-time pad) refers to a random bit stream that can only be used once because the key in stream ciphers cannot be utilized more than once. In block ciphers, the key is always chosen at random and is used for both encryption and decryption. Replay attacks are frequently thwarted using random numbers [3]. In encryption using an asymmetric key, the private and public keys are used, both the transmitter and recipient know the public key, which is then utilized' to scramble the multi-media or information [4].

In Cryptography, the algorithms play a crucial role in offering data security against malicious attacks. One of the most popular encryption algorithms is the chacha20 algorithm. Chacha20 is a high-speed stream cipher designed by Daniel J. B. and based on the Salsa20 cipher to improve the algorithm by increasing the diffusion [5]. One notable advantage of ChaCha20 is its use of the ARX (Addition, Rotation, and XOR) cryptographic approach, which confers expedited operational speed, reduced intricacy, and enhanced resistance against timing-based assaults [6].

Stream cipher will be the main topic of this paper because it is more appropriate for devices and applications. The production of random numbers is a key problem in cryptography. Games, encryption methods, and other applications that produce unpredictable results all employ random numbers. Recently, many devices with limited resources have been widely used. A resource-constrained system, which includes wireless sensors and RFID tags, lacks enough energy sources, computing power, and storage space. Because of their characteristics,

traditional ciphers are challenging to utilize in this industry with limited resources. Lightweight ciphers have consequently drawn a lot of interest. Additionally, nonlinear chaotic systems are fundamentally used in cryptography's pseudorandom number generators (PRNGs) [7].

The main contribution of this work is proposed approach put forth by the author involves the utilization of an Extended Chacha20 (EChacha20) stream cipher, which incorporates a higher number of rounds of the quarter-round function (QR – Fs) to enhance the level of security. It is possible that making the EChacha20 cipher more resistant to well-known cryptographic techniques like differential and linear cryptanalysis could be done by increasing the number of times the QR – Fs operation is done. This has the potential to enhance the protection, security, and authenticity of critical data.

An experiment was undertaken to assess the security of the expanded Chacha20 cipher by subjecting it to a differential cryptanalysis assault. The results were analyzed using correlation coefficient analysis, signal-to-noise ratio (SNR), and average change intensity (UACI), and the observations were documented accordingly. The evaluation of the consistency and variability of assessments

The study has additionally assessed performance metrics such as encryption speed, decryption speed, and memory utilization of EChacha20 while conducting a comparative analysis with the commonly employed Chacha20 cipher. The present evaluation has been employed to conduct a comparative analysis between EChacha20 and Chacha20, with the aim of assessing the impact of security enhancements.

The results illustrate the enhanced security and efficiency of the EChacha20 cipher, positioning it as a viable contender for secure communication applications. The main problem discussed in this article is to improve the performance of the ChaCha20 algorithm based on chaotic maps to ensure a secure transfer of data between the two ends of the connection., we

created a randomness key to power the algorithm against various attacks. We used the chaotic map to generate a random key for the encryption/decryption operations to improve the diffusion of the ChaCha20 cryptography algorithm's stream secret key. The random key that will be generated, is less complicated because it is chaotic; a chaotic map requires fewer complicated calculations and thus generates a less complex random key. We have explained this in detail in the proposed system.

The purpose of this work can be summed up as adding safety to the Chacha20 algorithm, by using a chaotic map (logistic, Chirikov functions). As a result, this new contribution will offer channels for data transmission like text and audio. The remaining parts of this document are arranged as follows: The second part presents a summary of the great efforts that have been made in encryption using the Chacha-20 encryption algorithm, as well as about the chaotic map, which we dealt with in this study with two types of chaotic map functions: logistic and Chirikov functions. Section 3 includes a thorough explanation of each step in the creation of the suggested algorithm as well as the proposed system. Section 4 elucidates the practical application and assessment of the proposed method. The subsequent sections, namely parts 5 and 6, will present the conclusion and associated work, respectively.

2. Related work

The paper titled "Enhancement of the ChaCha20 Encryption Algorithm Based on Chaotic Maps" proposes a hash function strategy that integrates the ChaCha20 data encryption technique with chaotic maps to produce cryptographic keys. This approach is discussed in reference [8]. The utilization of chaos-based random number generation has resulted in an enhancement of the robustness of the current ChaCha20 encryption technology. The ChaCha20 would perform cryptographic operations and generate keys with high unpredictability based on these random integers. The analysis's findings indicate that adding chaotic maps to the ChaCha20 algorithm

produced keys with high levels of randomness and lightweight hash function efficiency, which boosted security.

OpenSSH and OpenSSL were shown to have serious security flaws in ref. [9], which also included finding cryptographic remnants from the ChaCha20 cipher. Data that has been tunneled can be decrypted using single-target memory extraction. Law enforcement or malevolent actors could use the vulnerability to obtain a set of duplicate encryption keys for each tunneling connection. If the method of memory extraction is active, the virtual machine user will not be prompted to capture the encryption key. Making ciphers harder to recognize and restricting memory access are examples of mitigation techniques.

First, review the current attacks on salsa and ChaCha ciphers in [10]. An accurate calculation of the complexities of the attack has been made for the current technology. This improves the complexity by some margin. Using probability neutral bits against ChaCha and Salsa differential attacks includes two biases in probability: a backward bias (ϵ_a) and a forward bias (ϵ_d). To reduce the complexity of the attack, a method has been proposed to increase the inverse probability bias. Finally, focus on ChaCha's design principle.

Understanding the MAV Link protocol's security flaws led to the [11] proposal of MAV Sec, an integrated security system that uses encryption methods to safeguard the MAV Link messages that are sent between UAVs and GCS. According to experimental findings, ChaCha20 outperforms other encryption algorithms. ChaCha20 integration with MAV Link can guarantee communication confidentiality without compromising performance while using less CPU and taking up less memory, saving battery life and memory for UAVs with limited resources.

In Ref. [12], the Arduino UNO is prototyped as a proposed solution for securing captive nodes connections via MQTT/MQTT-SN. ChaCha20 and Poly1305. The outcomes demonstrated how little memory and processing time are needed for

this method. The suggested design is practical in terms of memory and CPU use and does not call for an IP connection to the MQTT-SN client. This proposal, when the number of nodes increases, will require better management of secret keys.

3. Methodology

3.1 Lightweight cryptographic algorithms

The domain of lightweight cryptography is now seeing tremendous advancements in research and development. The primary objective of this initiative is to ensure the security of devices that possess limited resources. These devices with low resources employ dependable encryption algorithms that consume minimal power and computational resources. The design of the lightweight cipher should prioritize achieving optimal performance while minimizing resource utilization, including memory and battery consumption. Hence, the implementation of lightweight cryptographic algorithms is necessary to ensure data security and provide secure transmission of data between devices and their respective controllers. Symmetric algorithms have superior computational efficiency as compared to asymmetric algorithms. The block cipher exhibits slower performance when compared to the stream cipher. Hardware enhances operational efficiency and increases throughput in comparison to software. In this part, we will examine two types of ciphers: block ciphers and stream ciphers:

- **Block ciphers**

The primary premise of a block cipher is to process a single block of components, typically consisting of n bits, as input. Through the application of cryptographic algorithms, the block cipher generates an output of equal size to the input. The block size of different algorithms can vary, such as in the case of AES, where a 128-bit block is utilized.

- **Stream cipher**

A stream cipher is a cryptographic technique that operates on a digital data stream by

encrypting it on a per-bit or per-byte basis [13]. In this work, we use the Lightweight Chacha20 Stream cipher algorithm.

3.1.1 Chacha 20 Algorithm

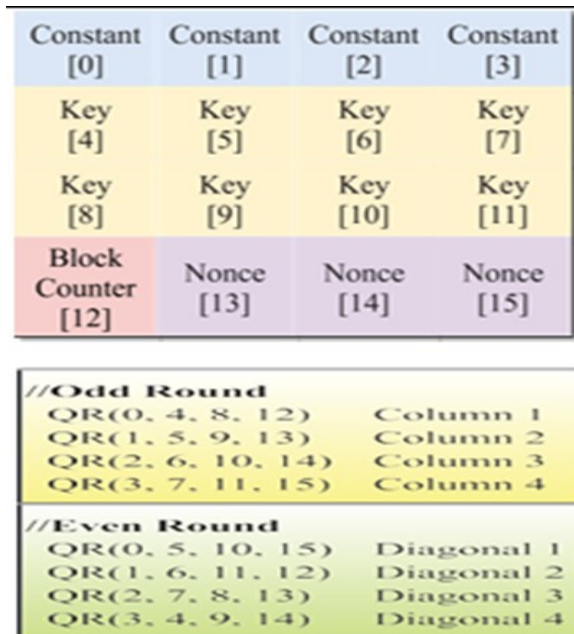
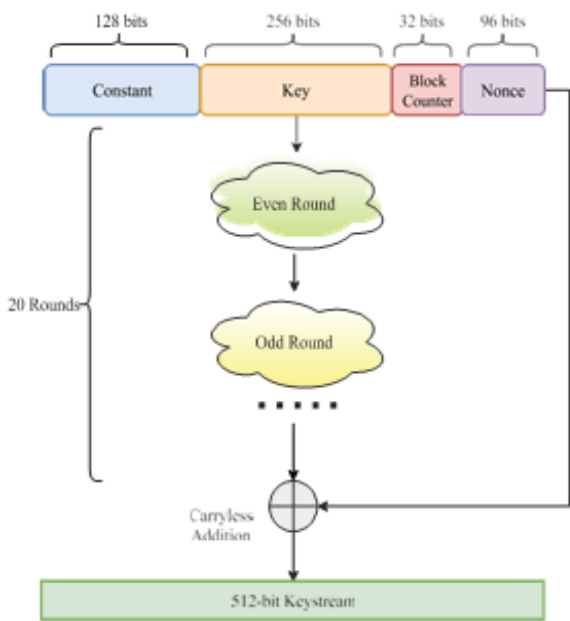
The security and protection of communication networks primarily depend on the utilization of cryptographic primitives. Symmetric-based ciphers have been widely utilized for an extended period due to their ability to provide data integrity, secrecy, and enhanced system security. Stream ciphers play a crucial role in symmetric cryptosystems due to their ability to provide speedier encryption and decryption processes, as well as their ease of implementation in both hardware and software environments.

Primarily, these basic elements have been favored due to the functionalities they employ in the process of cipher encryption. A stream cipher is a cryptographic algorithm that operates on a given key of n bits, which is then expanded to provide a longer keystream. Chacha20 is a stream cipher that has gained significant popularity due to its utilization of permutation functions, which serve to bolster its resilience against cryptanalysis. While the current body of literature acknowledges the positive aspects, it is necessary to delve deeper into the potential vulnerability of the subject to differential attacks [14].

Keystream is a term for the pseudorandom data bits produced by the stream cipher ChaCha20. To create the cipher text, the created keystream is XORed with the input plain text data. By performing an XOR operation on the ciphertext and the keystream, it is also possible to decrypt the ciphertext. The initial state for the ChaCha20 algorithm consists of 16 words with 32 bits each. It consists of a 256-bit private key, a 256-bit constant, a 48-bit block number that starts at zero, and a 96-bit nonce that is specific to a keystream. The initial state matrix is subjected to a sequence of 20 rounds, alternating between even and odd rounds, creating a 512-bit keystream. Fig. 1 depicts the ChaCha20 algorithm's flow. Four quarter-round (QR) functions make up each round. The beginning

matrix and the inputs of the even and odd rounds are different, as seen in Fig. 1 (a, b). Four words, each with 32 bits, are required for each QR. As

depicted in the flow diagram in Fig. 3, it updates the words via additions, XORs, and rotations [15].



a) State Matrix

b) Rounds

Fig. 1 Initial State Matrix and Even/ odd Rounds (a, b) [13].

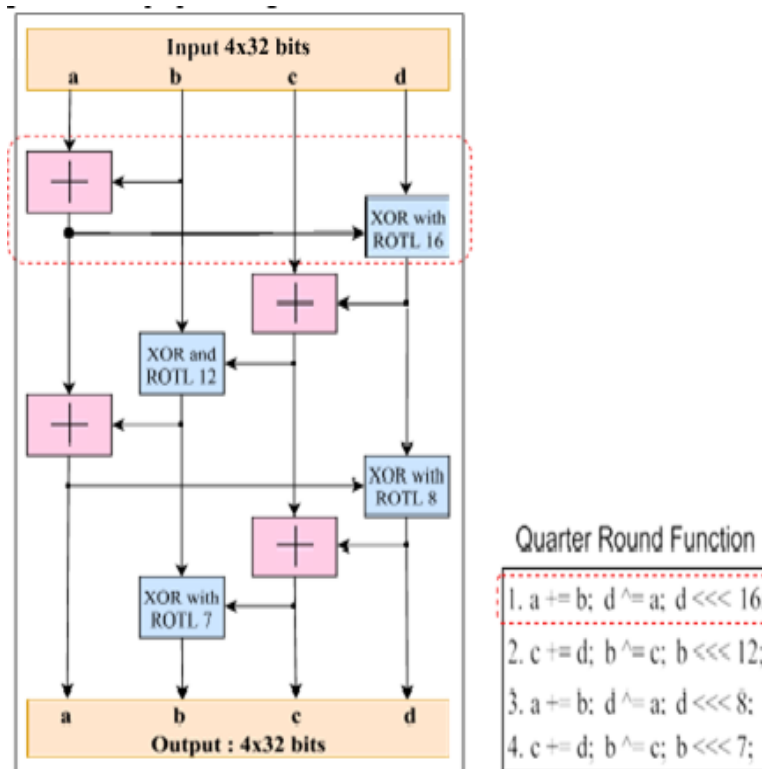


Fig. 2 ChaCha20:Quarter-Round [13]

Furthermore, Chacha20 employs distinct operations within its column transformations, enabling it to do two updates per round, hence enhancing its computational efficiency.

3.2. Chaotic Map

The theory of chaos explains a phenomenon that exhibits irregular appearances because of inevitable delay rules. One of the toughest nonlinear problems might be this one. Engineering has been impacted by the origin of chaos, which began in physics and math. That theory has been labeled (random) in mathematics, which means it arises from simple systems' inevitable influence on the system's initial conditions. In diverse disciplines of research like cryptography, physics, engineering, neurophysiology, and many more, there is currently a great deal of interest in understanding and using chaotic systems. Several crucial characteristics of chaos include its sensitivity, irregularity, ability to make long-term predictions, determinism, and nonlinearity. Studies conducted in the 20th century concentrated on the use of chaos in cryptography to obtain features for a system security design based on the chaos phenomenon. The logistic map and Chebyshev chaotic map that is used for

the suggested system are briefly introduced in the following subsection [8]. A Chaotic map can be defined as the core of a chaotic cipher system. Moreover, its types: are one-dimensional (1D) maps and high-dimensional (HD) [9].

3.2.1 Chirikov standard map (CSM)

One of the most significant maps in conservative chaos theory is the CSM, which is used, among other things, to predict when chaos will start. [16]. CSM is derived from a simple Hamiltonian system with an expelled rotor model. The chaotic behavior and some important characteristics led to the interest in these maps [10]. The two-dimensional CSM equation is :

$$\begin{aligned} X_{n+1} &= X_n + K \sin(2\pi Y_n) \pmod{1} \\ Y_{n+1} &= Y_n + X_{n+1} \pmod{1} \end{aligned} \quad (1)$$

The dynamics are roughly integral for the K parameter of small values. It is chaotic for a k parameter $K \gg 1$, and because the saddle points can have entanglements of the stable and unstable manifolds [17]. The strength of the kick K determines how nonintegrable and chaotic the dynamics produced by the map seen in Fig. 3 . Describe the phase portrait samples from the standard map for different K values [18].

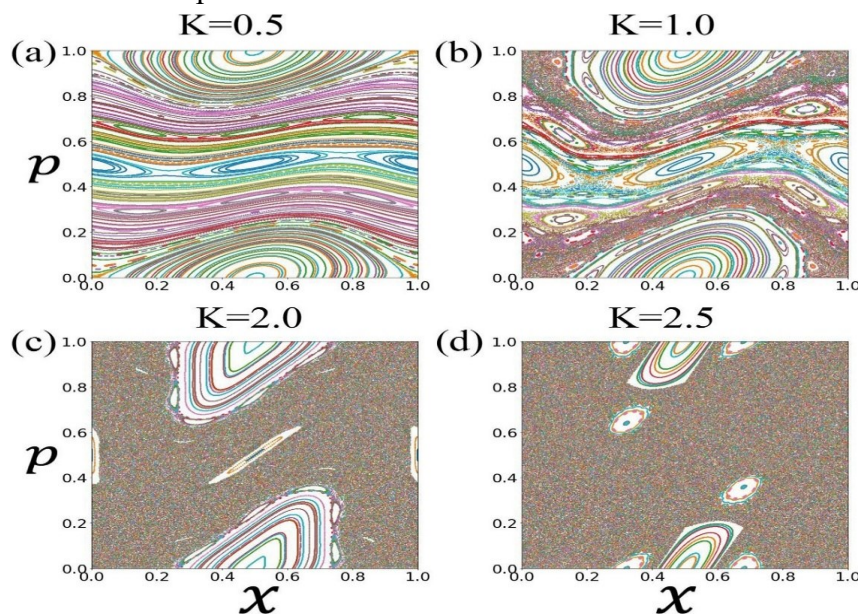


Fig. 3 give an examples of phase portraits (Poincare sections) of the standard map. (a) K = 0.5, (b) K = 1.0, (c) K = 2.0, (d) K = 2.5 [18].

3.2.2. 1D Logistic Map

The 1D logistic map is the most popular discrete chaotic system [16]. One-dimensional chaotic coordination has a straightforward structure, quick sequence creation, low levels of randomness, and a constrained range of chaotic parameters [19]. 1D logistic map defined in equation 2. [20].

$$X_{n+1} = r X_n (1 - X_n) \quad (2)$$

X_n represents the ratio of the current number of the maximum possible population and the values of x_n range between 0 and 1. As for the coefficient r , its value ranges from 0 to 4 [10]. The chaotic behavior of the function indicated in Equation 2 was determined by its bifurcation diagram, which is shown in Fig. 4 a. Where the state variable is defined by the parameter r , r is [1,4] and the control parameter is considered, this behavior is evident in the logistic map's quasi-uniform, invariant natural density, as depicted in Fig. 5b [21].

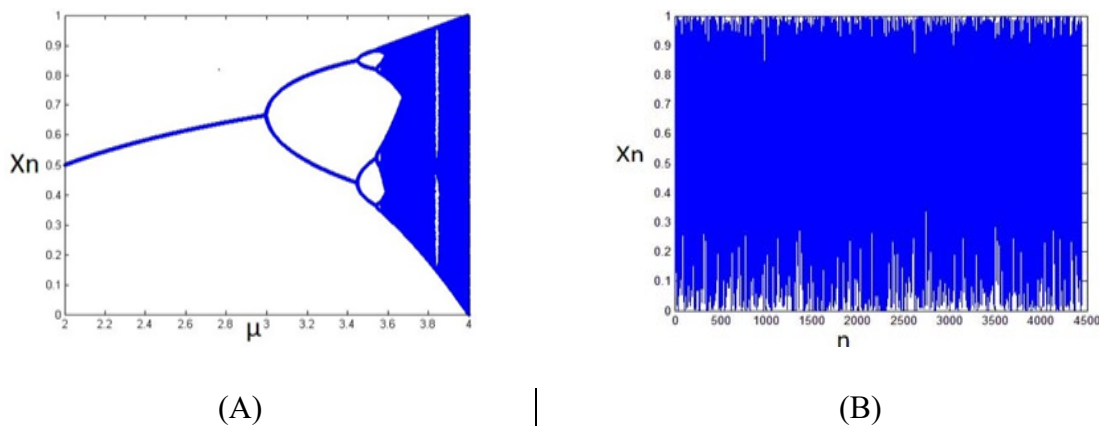


Fig. 4 (A) The bifurcation diagram (B) Invariant natural density [21].

4. Proposed Method

As shown in Fig. 5, the chaotic map stage, the Chacha20 stage, and the encoding/decoding

stages are the three primary stages of the proposed algorithm's general block diagrams.

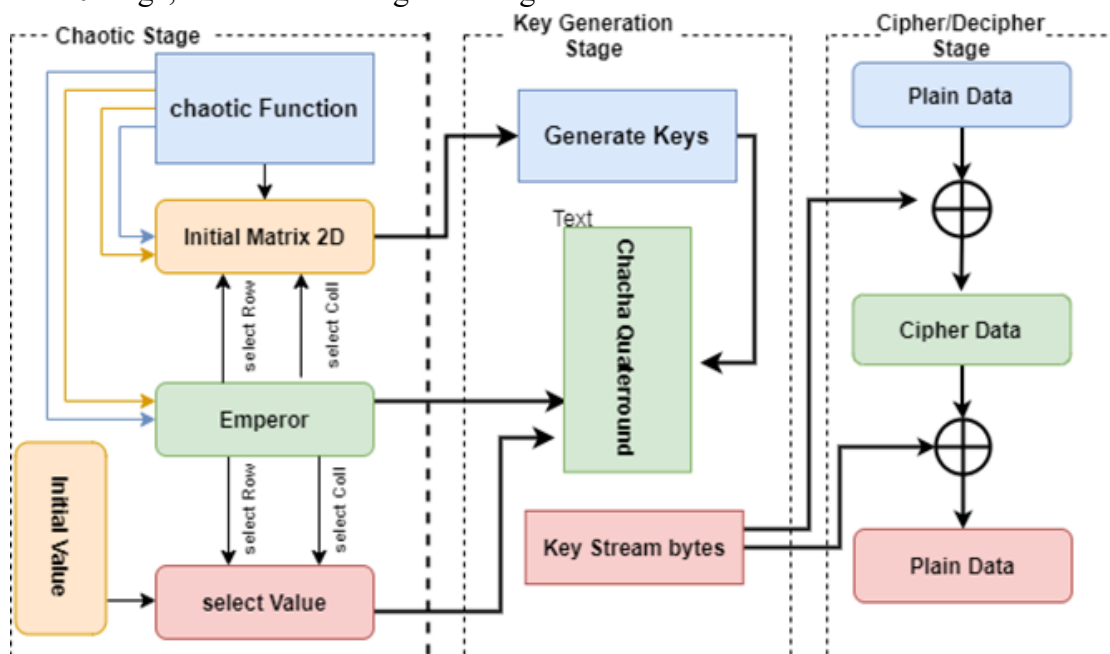


Fig. 5 General Block Diagrams of the proposed (EChacha20)

5. The Chaotic stage

This stage signifies improving the security level to the Original Chacha 20 algorithm, where the primary matrix has been created in step one than in second step selecting element from the matrix. The primary matrix has been created in first step with size $[24 \times 24]$ and Using 1d logistic, the values of this matrix are unsigned, random 48-bit integers. and Chirikov chaotic functions. Both employ the same method to create random

unsigned 48-bit integer numbers, which then used to fill the initial metric.

In second step, improve diffusion and complexity in the suggested method, resulting in increased security. This step employs a proposed technique for selecting one element at a time from an initial matrix formed in the previous step in a random manner. Fig. 6 elucidates block diagrams of the system.

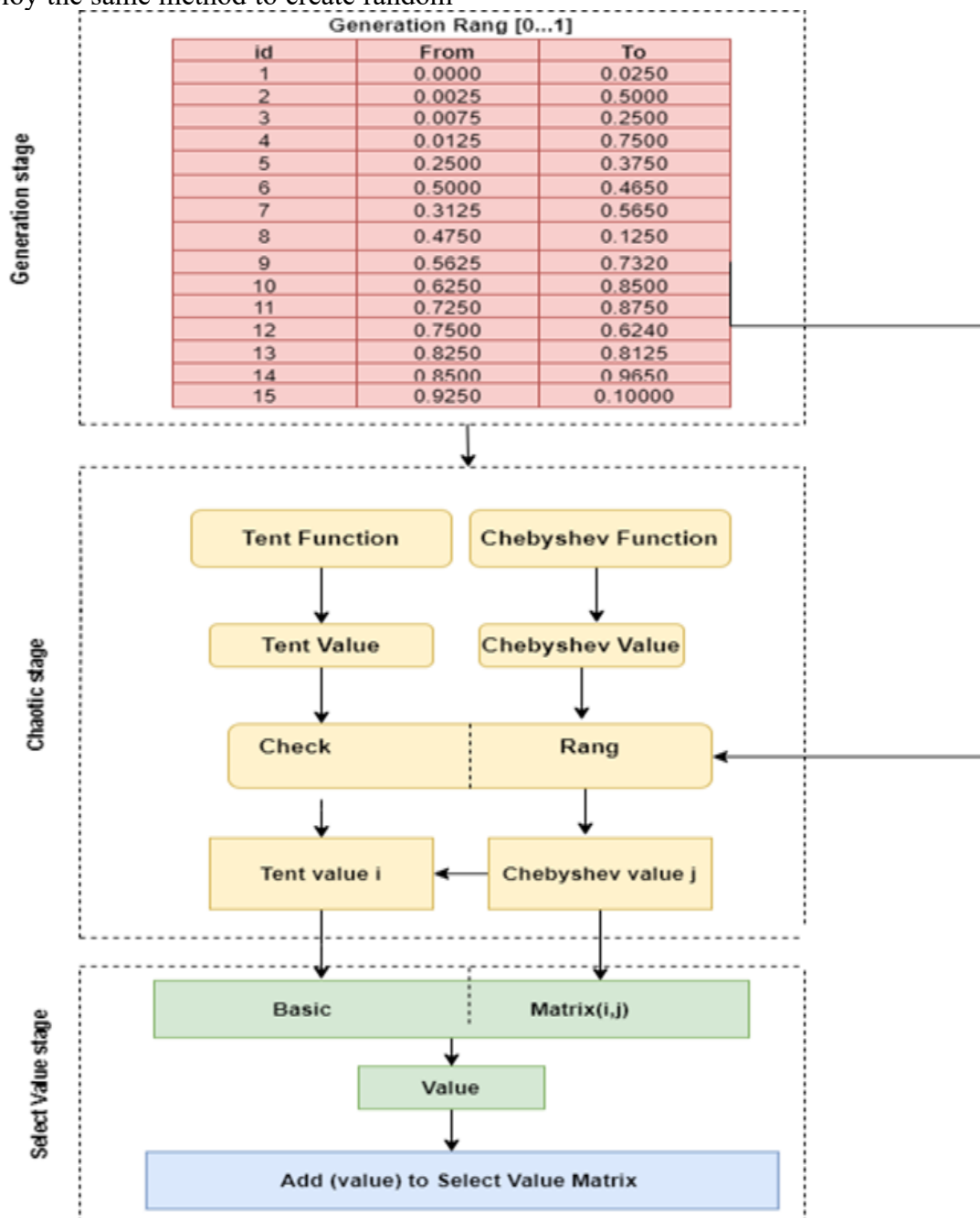


Fig. 6 block diagrams of the system

Proposed techniques ordered into three parts, as shown in Fig.7, to enable random selection from a primary matrix.

5.1 Key Generations Stage

Fig. 7 depicts the features of the proposed algorithm's key stream generation stage.

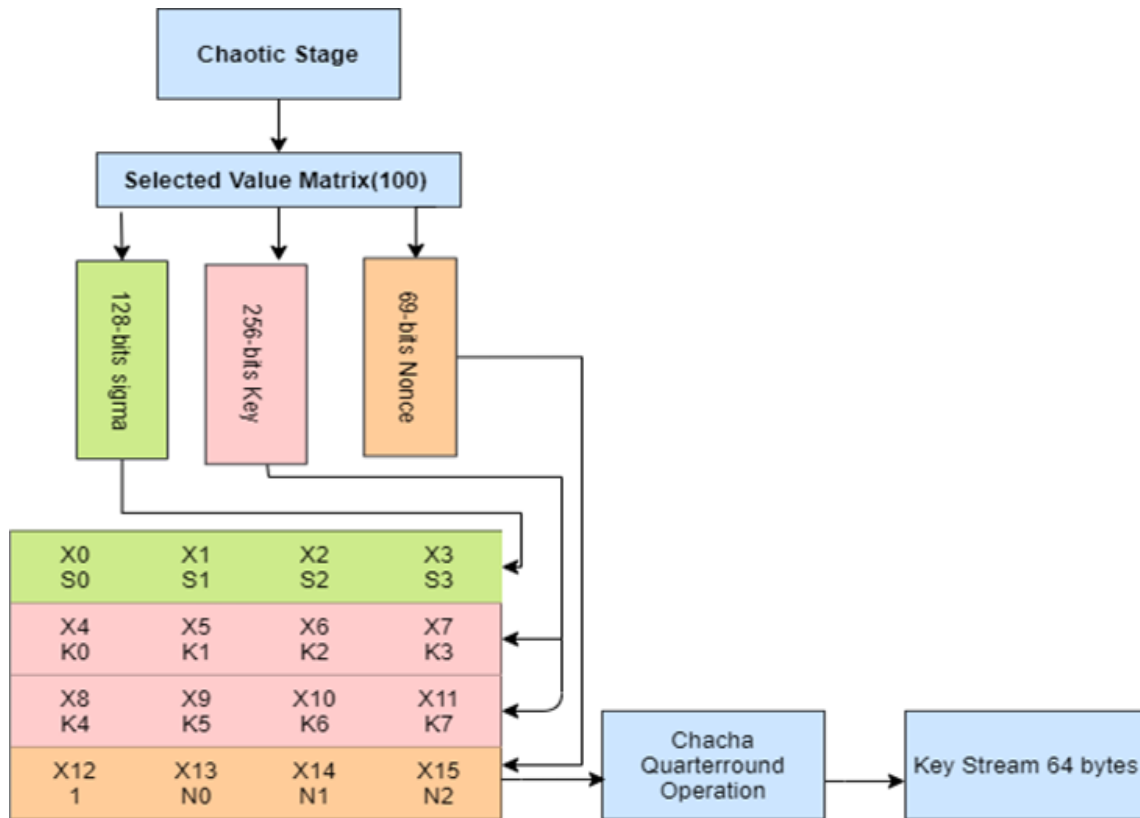


Fig. 7 Block diagram of chacha20 key stream generating.

The algorithm to performs computations (sigma bit length 128 bits, key length 256 bits, nonce length 96 bits) is offered in many steps as follows: (full-value as input, 256-bitkey, 96-bitnonce and 128-bitblocksigma as output) Fig.9

provides a visual representation of the practical implementation of the requested Echacha20 encryption algorithm, as described in the text sample (#1).

a) Split plain text #1																
Hash	Byte #1	Byte #2	Byte #3	Byte #4	Byte #5	Byte #6	Byte #7	Byte #8	Byte #9	Byte #10	Byte #11	Byte #12	Byte #13	Byte #14	Byte #15	Byte #16
0000	88	105	110	105	115	116	114	121	32	111	102	32	72	105	103	104
0016	100	114	32	69	100	117	99	97	116	105	111	110	32	97	110	100
0032	13	10	83	99	105	101	110	116	105	102	105	99	32	82	101	115
0048	101	97	114	99	104	47	68	105	121	97	108	97	32	32	14	85

b) Each character of the plaintext #1 using ASCII code

1274659156	3334001404	3459224807	1503453245
3288287804	2769032972	2097303670	809070498
3344945395	2802150961	833558683	3134718543
1	2328993245	3208036621	2440375332
2304560478	955107783	1169771929	1394803219
4203182483	3039001457	839477794	696523384
3289532234	238946272	1492119273	1192979932
1245592433	95381558	3553174574	1081436885

c) Convert the plaintext #1

Byte #1	Byte #2	Byte #3	Byte #4	Byte #5	Byte #6	Byte #7	Byte #8	Byte #9	Byte #10	Byte #11	Byte #12	Byte #13	Byte #14	Byte #15	Byte #16
192	164	145	88	86	119	112	158	33	77	65	111	165	252	189	53
210	214	198	241	62	119	73	154	160	205	137	212	240	254	6	235
197	113	15	74	222	209	152	74	197	172	13	50	85	4	163	215
172	120	26	247	219	215	37	104	211	104	140	61	57	84	159	139

d) 64-bytes of stream key using chaotic map

Hash	Char #1	Char #2	Char #3	Char #4	Char #5	Char #6	Char #7	Char #8	Char #9	Char #10	Char #11	Char #12	Char #13	Char #14	Char #15	Char #16
0000	141	205	255	49	37	3	2	231	1	34	141	205	255	45	37	3
0016	183	164	230	180	90	2	42	251	212	164	183	164	230	165	90	2
0032	200	123	92	41	183	180	246	62	172	202	200	123	92	41	183	180
0048	201	25	104	148	179	248	97	1	170	9	201	25	104	148	179	248

e) ASCII code for ciphertext #1

Hash	Char 1	Char 2	Char 3	Char 4	Char 5	Char 6	Char 7	Char 8	Char 9	Char 10	Char 11	Char 12	Char 13	Char 14	Char 15	Char 16
0000	•	Í	ÿ	1	%		ç		"	'	o	í	•	Ú]	
0016	·	⊠	Æ	'	Z		*	û	Ô	⊠	Æ	°	Ð	ÿ	h	•
0032	È	{	\)	·	'	ö	>	¬	Ê	D	Q	U	V	Æ	⊠

0048 É H ll ³ ø a a À \ t = Þ

f) Ciphertext#1

Has	Cha	Cha	Cha	Cha	Char#	Char#	Cha	Cha	Cha	Cha	Cha	Cha	Cha	Cha	Cha	Cha
h	r #1	r #2	r #3	r #4	5	6	r #7	r #8	r #9	r #10	r #11	r #12	r #13	r #14	r #15	r #16
000	M	I	N	i	s	t	R	Y		o	F		H	i	g	H
001 6	E	R		E	d	u	c	A	t	i	O	N		a	n	D
003 2		S	C	i	e	n	t	I	f	i	C		R	e	s	E
004 8	A	R	C	h	/	D	i	Y	a	l	A		U	n	i	V

128-bit sigma			96-bit Nonce		
#	Chaotic[x]	Byte	#	Chaotic[x]	Byte
		39			228
		245			61
1	212137255	164	1	382025188	197
		12			22
		244			255
		36			239
2	1741169908	200	2	355905945	34
		103			212
		213			97
		218			245
3	3735345877	164	3	706803041	32
		222			42

6. Results and Discussion

As previously mentioned, this paper introduces a new modification of the Chacha20 Algorithm. This is to enhance the security and speed level by using two type of chaotic map. Hence it will be very secure and faster. In this section, the security analysis of the proposed algorithm will be clarified, and finally, our

methods will be compared with previous work also described in this section.

1.1 Results of Key Create Stage

The outcomes of the stream key steps, employing a 256-bit key, a 96-bit nonce, and a 128-bit sigma, are presented in Table 1.

Table 1 Results from the key setup

256-bitkey					
#	Chaotic[x]	Byte	#	Chaotic[x]	Byte
		194			105
1	2706988738	94	5	1306717033	235
		89			226
		161			77
		98			133
2	778892386	244	6	217904261	244
		108			252
		46			12
		127			23
3	1752591231	107	7	3246222359	108
		118			125
		104			193

1.2 Results of the proposed EChacha20 algorithm's encryption of data

This section displays the outcomes of the encryption procedure using the specified EChacha20 algorithm. This subsection applies two distinct forms of data (text and audio) and displays the results of independently encrypting each type of data (i.e., 1 and 2).

1.2.1 Text encryption results using the EChacha20 algorithm

Text samples to use in test encryption of the proposed EChacha 20 algorithms show in table 2. The testing methodology is a random splitting of data into predetermined sizes (8, 16, 32, 64, 128, 256 MB) and sending them to cryptographic module

Table 2 Tests samples of plaintext

No. Text	Text	Text Size
#1	Ministry of Higher Education and Scientific Research/ Diyala University/ College of Science/ Computers Department	114 Char
#2	Convince what God has given you may be the richest people, but you do not know	77 Char

Table 3 shows a comparison between original Chacha20 and proposed Echacha20 algorithm using text samples as shown in table 2. , this comparison is done based on correlation coefficient measurement as indicated in equation (3), (4), and (5).

Table 3 Comparison between original chacha20 and proposed ichacha20

No Text	Original Chacha20	CorrelationCoefficient
#1	I think that some people need to be ignored a little, to know how beautiful our concern for them	0.595143378412081
#2	Ministry of higher education and scientific research, university of technology, faculty of computer and software engineering, department of information technology	0.616140519997412

Fig. 8 shows comparison between original and enhancement chacha20 encryption algorithm, where the optimal values of correlation coefficient metric is when closed to zero that means the proposed enhancement chacha20 encryption algorithm can generating the original algorithm's secret key is completely different.

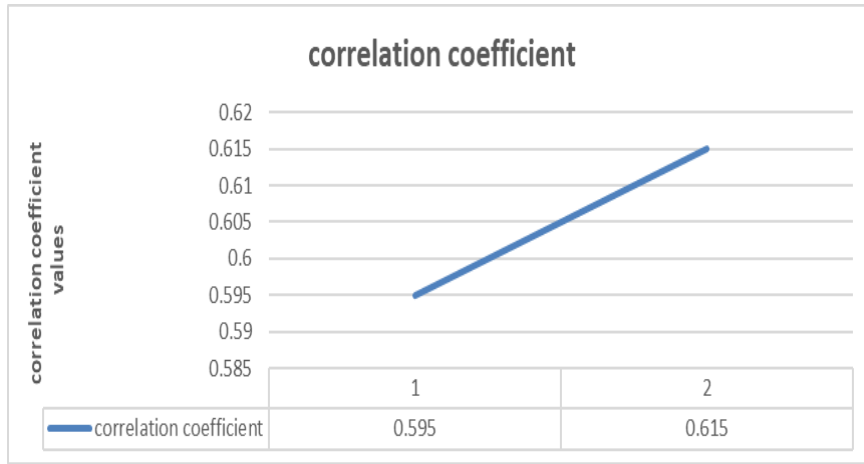


Fig. 8 Comparison between original Chacha20 and Proposed IChacha20 based on correlation coefficient metric.

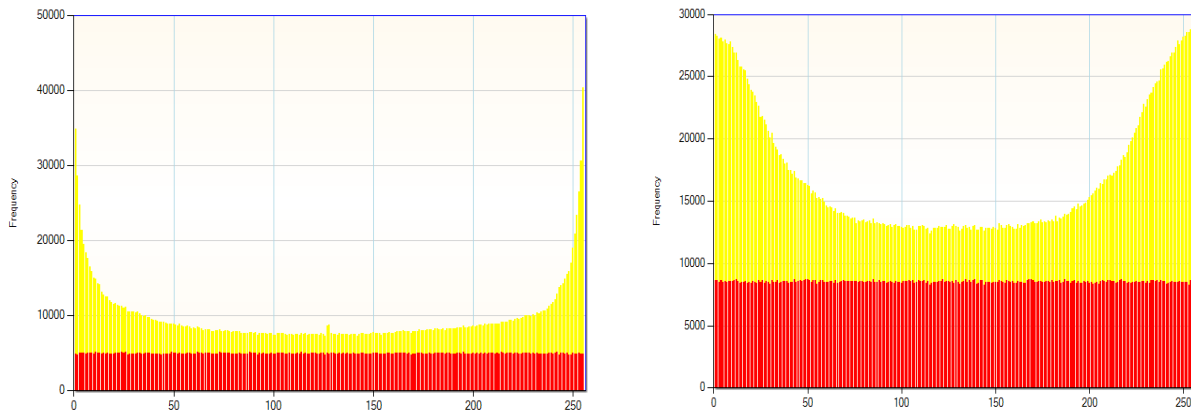
1.2.2 EChacha20 Audio Encryption Results

samples that were used to evaluate the proposed EChacha20 algorithm's performance. This algorithm is based on correlation analysis in equations (3), (4), and (5) and signal-to-noise ratio (SNR) equations (6) and (7), and Unified

Average changing Intensity (UACI) as in equation (8) metrics as shown in table 4. Using the specified EChacha20 algorithms, Fig. 9 illustrates the histogram corresponding to normal audio and cipher audio. When yellow pixels indicate normal audio and red pixels encrypt audio.

Table 4 samples of Audio. (Table 4 displays two voice)

No. Audio	Audio Time	Size	Bit Rate	Format
#1	00.00.11Sec	2.08MB	1536kbps	Wave
#2	00.00.07Sec	1.21MB	1411kbps	Wave



a) Histogram of the original and cipher audio #1 signal#1

b) Histogram of the original and cipher audio signal#2

Fig.9 Both audio signals #1 and #2 have histograms of the original and cipher audio signals.

In the encryption process, encrypted data is displayed as a histogram of integers. The performance of the encryption process is satisfactory if the distributions of these integers are somewhat close, as illustrated in Fig. 9. The

EChacha20-ciphered audio signal's histogram is distinct and uniform. This shows that EChacha20 has strong encryption quality, hence its high security grade.

Table 5. Result of quality metrics for Echacha 20 encryption algorithm

Metrics	No. Audio	Proposed Echacha20
Correlation Analysis	#1	0.000520658335128544
	#2	0.00107000038949594
SNR	#1	1.79684220590056
	#2	1.95699428118793
UACI	#1	0.204325678876544
	#2	1.324400988762221

The resulting analysis is dependent on Signal-To-Noise Ratio (SNR), Unified Average changing Intensity (UACI), and Correlation Analysis.

6.3. The Evaluation Metrics of the Proposed EChacha20 algorithm

Signal-to-Noise Ratio (SNR), Unified Average Changing Intensity (UACI), and Correlation Analysis are the most important characteristics used in cryptographic systems to ensure the quality of two voices.

6.3.1 Correlation Coefficient

According to [8] the range of values for the correlation coefficient is assumed to be between -1 and +1. The following equation explains how to use the correlation coefficient functions:

$$E(c) = \frac{1}{s} \sum_{i=1}^s p_i \quad (3)$$

$$D(c) = \frac{1}{s} \sum_{i=1}^s [p_i - E(p)]^2 \quad (4)$$

Finally, the related coefficients Γ_{PC} can be described in this Eq. 5

$$\Gamma_{pc} = \frac{E\{[p - E(c)][c - E(c)]\}}{\sqrt{D(P)} \sqrt{D(P)}} \quad (5)$$

6.3.2 Signal to Noise Ratio (SNR)

To test the effectiveness of the audio encryption using the SNR, often expressed in decibels (dB). More signals than noise is present in the equation if the ratio is larger than one (or 0 dB) (6) [22]:

$$SNR_{MS} = \frac{\sum_{x=0}^{N-1} g(x)^2}{\sum_{x=0}^{N-1} (g(x) - h(x))^2} \quad (6)$$

The Signal-to-Noise Ratio (SNR_{rms}) produced by equation's root is its square value (7).

$$SNR_{rms} = (SNR_{MS})^{0.5} \quad (7)$$

6.3.3 Unified Average Change Intensity (UACI)

The difference in the average image density of plain and encrypted text can be calculated by UACI. The mathematical formulation is:

$$UACI = \frac{1}{M \times N} \left[\sum_{i,j} \frac{|E1(i,j) - E2(i,j)|}{255} \right] \times 100\% \quad (8)$$

Where E1 and E2 encrypted the original and encrypted plaintext respectively [23]. Based on the scale values utilized in the proposed system, Fig. 10 depicts the evaluation production of the EChacha20 with authentic Chacha20 by scale result used in the proposed system. The results point out that the Correlation coefficient value between the plain and cipher audio is large. So, the results SNR and UACI show that the values encoding results using the proposed method are small compared with the original method.

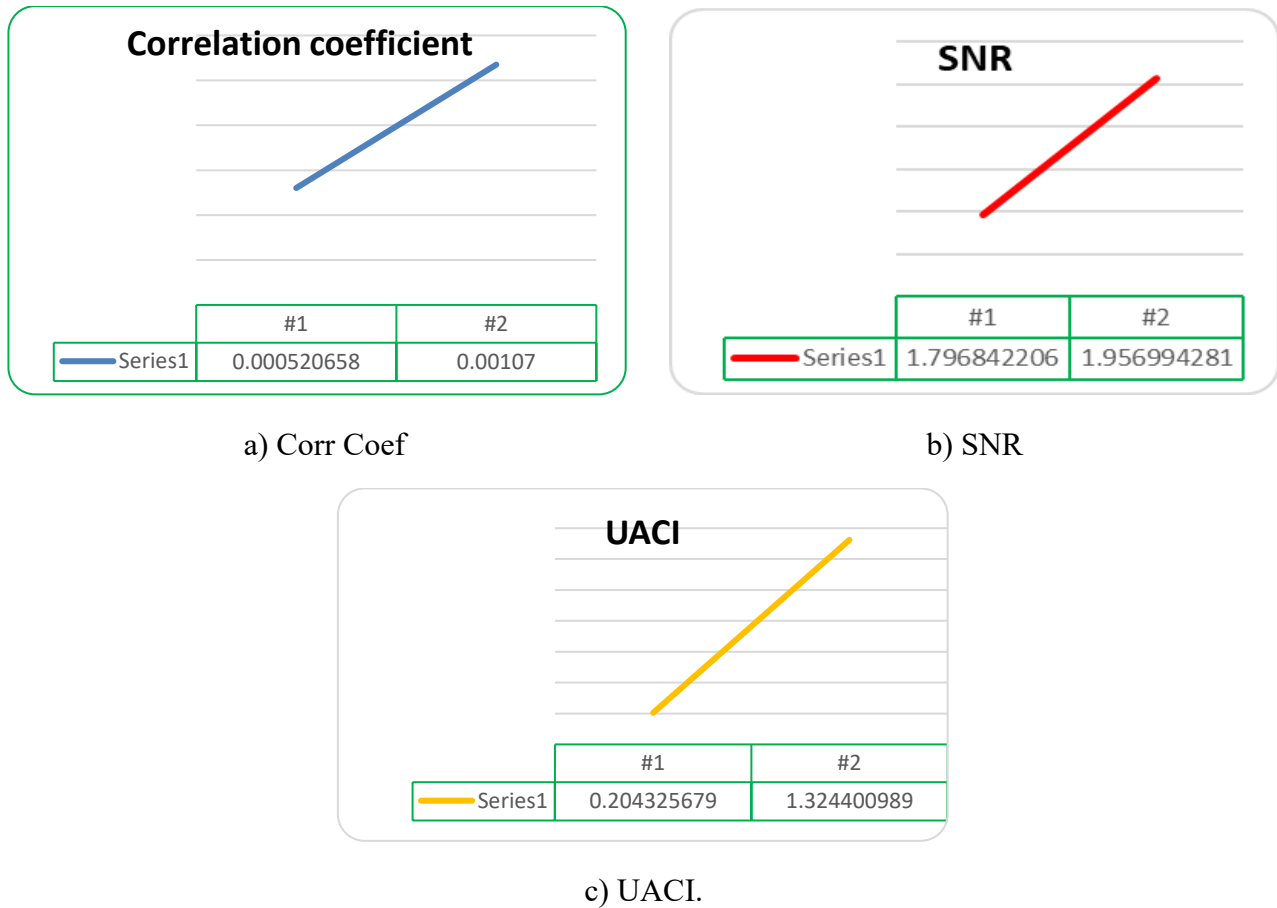


Fig.10 Evaluation production of the EChacha20 with authentic Chacha20 by scale result: a) Corr Coef, b) SNR, c) UACI.

There are many studies and researchers to modify the Chacha20 version, as illustrated in Table6.

Table 6 Comparison between the proposed system and several previous works.

No.	Reference	Methods	Performance
1	Taha, M. H., et al.[22]	Chaotic +chacha20	EChacha20 is more resistant to assaults aimed at extracting information from the encrypted picture.
2	Mahdi, M. S., et al.[23]	Chaotic +chacha20	The lightweight encrypted images are proven robust against brute force attacks.
3	Nidaa F. H., et al.[25]	Digital Speech Files encrypted + chaotic maps	Cryptographic algorithm is proposed for a firm and safe speech encryption. Speech data are scattered according to a cubic model;
4	Our proposed System	Chaotic +chacha20	When compared to the original Chacha20, EChacha20 is more resistant to attacks aimed at extracting information from the encrypted data and is quicker. Moreover, for any change that can occur in the secret key the proposed system is able to generate new keys each time.

In this equation, s stands for the total number of bits; P , and C are measured in series of s ; and $E(c)$ is a mathematical expectation of c . P is the value of the original bits or plaintext, and C is the value of the changed bits or ciphertext. The variance of p is represented by the following equal:

7. Conclusion

To check the security of the system, some tests done as given below: correlation analysis, Unified Average changing Intensity (UACI), Signal to noise ratio (SNR) These tests are performed on ten individual records in the system. The Chacha20 encryption has been the target of several attacks in recent years, including key recovery, differential, collision, and other attacks. Based on this, we provided an algorithm in this work that, thanks to the employment of a chaotic function, can produce keys that are extremely sensitive to any change in the initial values and with increased diffusion.

Echacha20 Algorithm can be success for encryption text using chaotic stream key with the ability to generate a secret key that completely different from the original algorithm based on correlation coefficient measurement, where the best value of correlation coefficient measurement that achieved by Echacha20 = 0.61614052 with text size =77 char The suggested Echacha20 algorithm has a faster execution time than the original one, measured in seconds, it was 01:47:9 in the encryption process while in the original algorithm it was 02:45:3. The proposed Echacha20 algorithm produces a significantly different and uniform histogram for the ciphered audio signal than the original audio. Thus, the suggested Echacha20 achieves a very high level of security through very good encryption.

The results of audio quality metrics of the proposed Echacha20, the proposed algorithm obtains acceptance value for all testing, the best correlation analysis result was 0.000520658335128544, SNR = 1.79684220590056, and UAC was 0.204325678876544.

In the future work, It is possible to integrate more than one lightweight algorithm such as the salsa20 to implement the system, as well as use other types of the chaotic map easily to ensure encryption with a higher security level and a more powerful key against breakage and benefit from this system not only as a security system but it can be employed as part of an integrated system to protect data in Blockchain encryption of database using Echacha20 based on DNA sequence. Additionally, we suggest employing the system for guiding an assault, specifically a key recovery attack, on the suggested chacha20 method to assess the robustness of the cipher.

Conflict of interest

There are no conflicts of interest regarding the publication of this manuscript.

References

- [1] Jindal, P., Kaushik, A., & Kumar, K. "Design and implementation of advanced encryption standard algorithm on 7th series field programmable gate array," 7th international conference on smart structures and systems (ICSSS), pp. 1-3, IEEE 2020.
- [2] Karim, R., Rumi, L. S., Islam, A., Kobita, A. A., Tabassum, T., & Sagar Hossen, M. "Digital signature authentication for a bank using asymmetric key cryptography algorithm and token-based encryption," In Evolutionary Computing and Mobile Sustainable Networks, vol. 53, pp. 853-859, Singapore, 2021.
- [3] P. Nannipieri et al., "True random number generator based on Fibonacci-Galois ring oscillators for FPGA," Appl. Sci., vol. 11, no. 8, pp. 3330, 2021.
- [4] R. L. Rivest, A. Shamir, and L. Adleman. 1978. A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM, vol. 21, no. 2, pp. 120–126, Feb. 1978. <https://doi.org/10.1145/359340.359342>.
- [5] D. A. F. Saraiva, V. R. Q. Leithardt, D. de Paula, A. Sales Mendes, G. V. González, and

- P. Crocker, "Prisec: Comparison of symmetric key algorithms for iot devices," *Sensors*, vol. 19, no. 19, pp. 4312, 2019.
- [6] S. M. S. Reza, A. Ayob, M. M. Arifeen, N. Amin, M. H. M. Saad, and A. Hussain, "A lightweight security scheme for advanced metering infrastructures in smart grid," *Bull. Electr. Eng. Informatics*, vol. 9, no. 2, pp. 777–784, 2020.
- [7] S. A. Jassim and A. K. Farhan, "Combined Chebyshev and logistic maps to generate pseudorandom number generator for internet of things," *Int. J. Electr. Comput. Eng.*, vol. 12, no. 3, pp. 3287, 2022.
- [8] Alyas, H. H., & Abdullah, A. A. , "Enhancement the ChaCha20 Encryption Algorithm Based on Chaotic Maps," In *Next Generation of Internet of Things*, vol. 201, pp. 91-107, Singapore, 2021.
- [9] P. McLaren, W. J. Buchanan, G. Russell, and Z. Tan, "Deriving ChaCha20 key streams from targeted memory analysis," *J. Inf. Secur. Appl.*, vol. 48, pp. 102372, Oct. 2019.
- [10] S. Dey, T. Roy, and S. Sarkar, "Revisiting design principles of Salsa and ChaCha", *Adv. Math. Commun.*, vol. 13, no. 4, pp. 689-704, 2019.
- [11] A. Allouch, O. Cheikhrouhou, A. Koubâa, M. Khalgui, and T. Abbes, "MAVSec: Securing the MAVLink protocol for ardupilot/PX4 unmanned aerial systems", *15th International Wireless Communications & Mobile Computing Conference (IWCMC)*, pp. 621–628, 2019.
- [12] Sadio, O., Ngom, I., & Lishou, C, "Lightweight security scheme for mqtt/mqttn protocol", In *2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS)* , pp. 119-123, IEEE, October 2019.
- [13] Bhagat, V., Kumar, S., Gupta, S. K., & Chaube, M. K. (2023). Lightweight cryptographic algorithms based on different model architectures: A systematic review and futuristic applications. *Concurrency and Computation: Practice and Experience*, 35(1), e7425.
- [14] Kebande, V. R. (2023). Extended-Chacha20 Stream Cipher with Enhanced Quarter Round Function. *IEEE Access*.
- [15] Aamir, M., Sharma, S., & Grover, A. , "ChaCha20-in-Memory for Side-Channel Resistance in IoT Edge-Node Devices", *IEEE Open Journal of Circuits and Systems*, vol. 2, pp. 833-842, 2021. doi: 10.1109/OJCAS.2021.3127273.
- [16] Mašović, D. R. "Kicked rotor with attosecond pulse train," *Journal of Physics A: Mathematical and Theoretical*, vol. 54, No. 9, pp. 095701, 2021.
- [17] S. Dey and S. Sarkar, "Proving the biases of Salsa and ChaCha in differential attack," *Des. Codes Cryptogr.*, vol. 88, no. 9, pp. 1827–1856, 2020.
- [18] Lee, W. S., & Flach, S., "Deep learning of chaos classification," *Machine Learning: Science and Technology*, vol. 1, No. 4, pp. 045019, 2020.
- [19] S. Jing, Y. Guo, and W. Chen, "Meaningful ciphertext encryption algorithm based on bit scrambling, discrete wavelet transform, and improved chaos," *IET Image Process.*, vol. 15, no. 5, pp. 1053–1071, 2021.
- [20] M. S. Fadhil, A. K. Farhan, and M. N. Fadhil, "Designing Substitution Box Based on the 1D Logistic Map Chaotic System," *IOP Conference Series: Materials Science and Engineering*, vol. 1076, no. 1, pp. 12041, 2021.
- [21] Benrhouma, O., Alkhodre, A. B., AlZahrani, A., Namoun, A., & Bhat, W. A., "Using Singular Value Decomposition and Chaotic Maps for Selective Encryption of Video Feeds in Smart Traffic Management," *Applied Sciences*, vol. 12, No. 8, pp. 3917, 2022.

- [22]N. S. Mohammed, Z. T. M. Al-Ta, and S. S. Mohammed, "An Enhancement of LSB Audio Steganography Using Magic Cube," *Diyala J. Pure Sci.*, vol. 15, no. 03, pp. 88-102, 2019.
- [23]A. Qayyum et al., "Chaos-based confusion and diffusion of image pixels using dynamic substitution," *IEEE Access*, vol. 8, pp. 140876–140895, 2020.
- [24]M. H. Taha and J. M. Al-Tuwaijari, "Improvement of Chacha20 Algorithm based on Tent and Chebyshev Chaotic Maps", *Iraqi Journal of Science*, vol. 62, no. 6, pp. 2029–2039, Jul. 2021.
- [25]M. S. Mahdi, R. A. Azeez, and N. F. Hassan, "A proposed lightweight image encryption using ChaCha with hyperchaotic maps," *Period. Eng. Nat. Sci.*, vol. 8, no. 4, pp. 2138–2145, 2020.
- [26]N. F. Hassan, A. Al-Adhami, and M. S. Mahdi, "Digital Speech Files Encryption based on Hénon and Gingerbread Chaotic Maps", *Iraqi Journal of Science*, vol. 63, no 2, pp. 830–842, Feb. 2022