ISSN: 0067-2904

Developing the Complexity and Security of the Twofish Algorithm Through a New Key Scheduling Design

Anwar Abbas Hattab*, Ali Hussein Saieed
Department of Computer Science, College of Education, Mustansiriyah University, Baghdad, Iraq

Received: 1/8/2022 Accepted: 29/12/2022 Published: 30/11/2023

Abstract

The Twofish cipher is a very powerful algorithm with a fairly complex structure that permeates most data parsing and switching and can be easily implemented. The keys of the Twofish algorithm are of variable length (128, 192, or 256 bits), and the key schedule is generated once and repeated in encrypting all message blocks, whatever their number, and this reduces the confidentiality of encryption. This article discusses the process of generating cipher keys for each block. This concept is new and unknown in all common block cipher algorithms. It is based on the permanent generation of sub keys for all blocks and the key generation process, each according to its work. The Geffe's Generator is used to generate subkeys to make each explicit block a new key that differs from block to block, gaining protection against attacks. Finally, this algorithm works almost like a One-Time Pad.

Keywords: Geffe's Algorithm, key schedule, security, symmetric block cipher, Twofish algorithm.

تطوير تعقيد وأمن خوارزمية Twofish من خلال تصميم جديد لجدولة المفاتيح

انوار عباس حطاب*، علي حسين سعيد
قسم علوم الحاسوب ، كلية التربية ، الجامعة المستنصرية ، بغداد ، العراق

الخلاصة

يعد تشفير Twofish خوارزمية قوية جدًا ذات بنية معقدة إلى حد ما ، والتي تتخلل معظم عمليات تحليل البيانات وتبديلها ، ويمكن تنفيذها بسهولة. مفاتيح خوارزمية Twofish ذات أطوال متغيرة (128 أو 192 أو 256 بت) ، ويتم إنشاء جدول المفاتيح مرة واحدة ويتكرر في تشفير جميع كتل الرسائل مهما كان عددها ، وهذا يقلل من سرية التشفير. تتناول هذه المقالة عملية إنشاء مفاتيح التشفير لكل كتلة. هذا المفهوم جديد وغير معروف في جميع خوارزميات تشفير الكتلة المعروفة ، ويستند إلى التوليد الدائم للمفاتيح الفرعية لجميع الكتل وعملية إنشاء المفاتيح ، كل حسب عملها. يتم استخدام Geffe's Generator لإنشاء مفاتيح فرعية لجعل كل كتلة صريحة مفتاحًا جديدًا يختلف من كتلة إلى أخرى ، ويكتسب الحماية من الهجمات. أخيرًا ، تعمل هذه الخوارزمية تقريبًا مثل One-Time Pad.

* Email: anwarabbas76@uomustansiriyah.edu.iq

1. Introduction

Information security refers to the protection of data and information systems from illegal access, use, disclosure, disruption, alteration, or degradation. Frequently, the terms "security" and "concept" are used interchangeably. These sectors are interconnected and have the common goal of safeguarding information access, trustworthiness, and privacy [1].

Information saved manually can be protected by hiding it in a safe place locked with a key that only those authorized to use it have access to. Then, what about the information saved on computers and what about the information transmitted through computer networks? It is necessary to protect information and data so that it is not subject to modification, destruction, non-disclosure, and obfuscation, intentionally or unintentionally. Thus, there must be ways to protect the transfer and storage of data and information, such as encrypting the message, hiding its contents, and changing it into a different form before it is sent or stored [2].

Considerable caution should be exercised when picking resource-constrained security algorithms. The choice must take into account implementation costs, power consumption, and the fact that uniform algorithms, especially block ciphers, continue to play a critical role in embedded system security [3]. It is thus necessary to develop a new encryption algorithm to address all DES's weaknesses.

Developed by Bruce Schneier in 1998, Twofish is a symmetric key block cipher based on his Blowfish algorithm and designed to be difficult to exploit [4, 5]. Twofish splits the key into two halves by using a key-dependent key scheduling scheme and a pre-computing S-box to divide the key in half. Approximately one-half of the key is used to produce the actual key, with the other half being used to alter the algorithm [6].
Most block ciphers use S-box ciphers, which use a method called non-linear table-based substitution [6, 7].

2. Encryption

Encryption is the set of transformations performed on the plaintext to obtain the ciphertext. These transformations include the encryption algorithm in addition to the key. Stream encryption and block encryption are the most important types of modern encryptions used today [2].

2.1 Stream Cipher

Stream encryption is one of the most important modern encryption methods and is commonly used in the field of communications. Most of the stream coding systems consist of a group of shift recorders with a Linear Feedback Shift Register. Each shift recorder consists of a group of hoppers (flip-flops), each called a stage. The number of these stages is determined by the length of the shift recorder and its complexity. Here an important concept emerges, which is the equivalent Linear Equivalence.

Equivalent Linear Equivalence is defined as the minimum length of the displacement register used to generate the sequence. Most of these stages can be involved by means of a function called the feedback function. In each pulse, the contents of the register are displaced. Also, linking functions are involved in its formation. Different linear and non-linear combinations are used to connect shift registers to obtain sequences with long cycle, good random properties, and high linear complexity to ensure high confidentiality in ciphers, where the encryption is carried out (bit by bit) with a time-varying function. One of the important algorithms in stream encryption is the Geffe’s Algorithm. This algorithm consists of three

shift registers of different lengths, the GCD (Greatest Common Denominator) between their lengths is one, and the feedback function for each shift register is a linear function [8, 9].

2.2 Block Cipher

Block ciphers are a type of the modern symmetric cipher models that use only one key for encryption and decryption. In this type of cipher, plain text is divided into blocks or blocks of a fixed length, and a single function is used to encrypt each of these segments to produce the ciphertext block. Bruce Schneier introduced in this field the Blowfish and Twofish algorithms as an alternative to the standard DES algorithm in 1994 & 1998. His goal with these algorithms was to achieve standards of application speed, small size, ease of study and analysis, and finally a level of variable security that is proportional to the length of the key used, up to 488 bits [10, 11].

3. Twofish algorithm

Bruce Schneier designed the Twofish algorithm, a symmetric block cipher, in 1998 [12]. The AES architectural standard developed by the "National Institute of Standards and Technology" (NIST) was to the bases for the design. As a sequence of blocks, it uses a 128-bit, 192-bit, or 256-bit key to encrypt the input message [13]. Twofish is a popular key choice due to its robust keys & versatile design. It is quick and efficient with both software and hardware, so it is compatible with a large number of systems [14].

Twofish's technique is built on the blowfish algorithm, which is a type of current block encoding algorithm built from a collection of intricate mathematical processes and key organizational designs that make it difficult for third parties to breach [6]. In 1949, Shannon's ideas for symmetric cipher building were published. Shannon proposed a "mixing transformation" comprised of multiple layers or rounds of confusion and diffusion to build secure, effective product ciphers. Confusion is a technique that utilizes the replacement strategy to confuse the link between both the key as well as the ciphertext to enable the recovery of the plaintext. Diffusion uses the permutation process to determine the more intricate statistical link between encrypted text and plaintext [14].

As the first structured implementation of Shannon's principles, the Feistel structure included a network structure composed of a collection of small substitutions (termed S-boxes), introduced via connected-by-bit location permutations or transpositions, and a lookup table, also introduced similarly. Substitution-permutation networks (SPNs) are a type of network structure based on Shannon's concepts that are used to create symmetric-key ciphers, such as the Advanced Encryption Standard (AES). The most critical components of SPN are the diffusion matrices, S-boxes, and key schedules. The diffusion matrices are generated using the Maximum Distance Separable (MDS) matrix as the starting point. Additionally, the MDS is used in Feistel Ciphers like Twofish [6].

There are two types of "f-function & g-function" in the Twofish algorithm, each having four boxes of 8x8 bits, a fixed 4x4 (Maximum distance separable) over GF (28), flips of the bits, and a (Pseudo-Hadamard Transform (PHT)) mechanism that occurs inside the algorithm, all of which is powered by a 16-round Feistel network [15]. It can be seen in Figure (1) how the Twofish technique performs to guarantee that the regular text size is 128 bits in the algorithm and split into four sections where a four-key XOR-ED technique is designated by whitening and introducing some complicated processes explained in two parts under the terms "Twofish and components" [13].

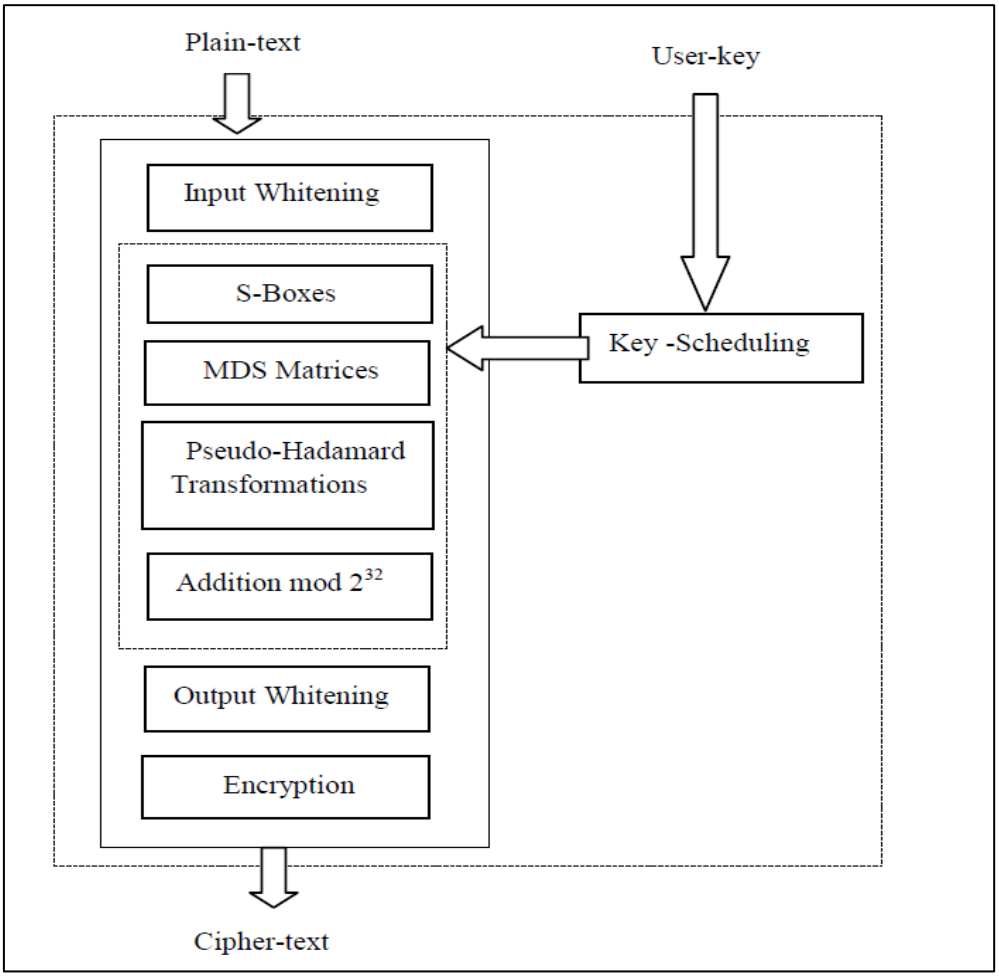

Figure 1: steps of Twofish Algorithm [16].

Twofish has eight sub-keys (Key0-Key7) that are repeatedly numbered and used to XOR the output and input data. This is called input and output whitening. Eight-bit left rotation, a key based on S boxes, MDS, PHT, and two sub keys 232 are all included in the F-function. Figure (2) shows them all. Depending on the key, it has a G-function that has four S-boxes. There is a significant amount of duplication in the encoding process due to the g-function being present twice in the algorithm's structure [13, 16].

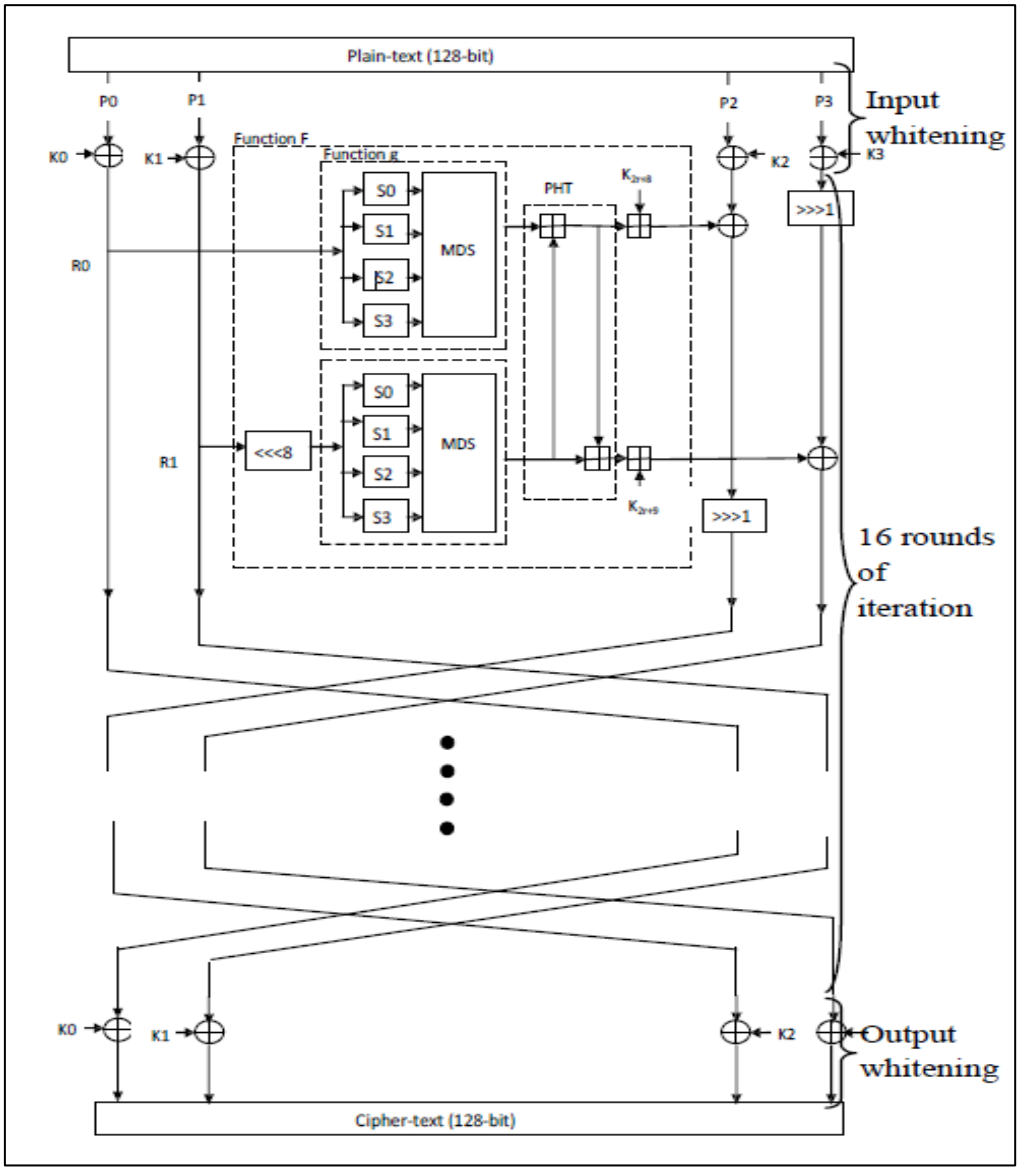

Figure 2: framework of the main Twofish algorithm [16].

Whitening is a process that requires (X-OR) key materials before the (first round) and after the (last round) of the algorithm [17]. In order to make key search attempts against the residue of the cipher more difficult, the first and last round of f-function inputs are hidden from an adversary via whitening. 128 bits of subkeys are XORed by Twofish. Another 128 bits after the final Feistel round and then you get to the first Feistel round [14].

3.1Functions used in Twofish

1. F-Function

The f function's permutation of 64-bit numbers is key-dependent. The rounding factor r is used to choose the necessary subkeys from the two input words R0 and R1. R0 is sent to the g function, which returns the value T 0. To produce T1, first R1 must be rotated by eight bits left and then the g function is used on the result. A PHT combines the results of the T0 and T1 trials and adds two extra keywords. The next section contains a set of equations elaborating on the F function [16]:-

$$T_0 = g(R_0) \quad (1)$$

$$T_1 = g(ROL(R_1;8)) \quad (2)$$

$$F_0 = (T_0+T_1+K\_(2r+8)) \bmod 2^{32} \quad (3)$$

$$F_1= (T_0+2T_1+K\_(2r+9)) \bmod 2^{32} \quad (4)$$

Where:
T_0, T_1: result of first g function, second g function
g: g function
R_0, R_1: two-word result of input whitening
Rol: denotes the leftmost rotation of the first argument by the value of the second argument

2. g-Function

Each word has 4-bytes. all byte is stored in its own (S-box), whose contents are dictated by the keystrokes used to deliver it. Each S-box features an eight-bit input and output. Multiplying each of the four outcomes by the 4x4 MDS matrix results in the same-length four-by-GF (2^8) vector. The output vector is transformed to a 32-bit word using the g-function [14].

3. h-Function

Given 2-inputs, a 32(bit) word (X) and an array of 32-bit words (Lk-1)), this function returns a single word whose length is equal to the length of the list L = (L0... Lk-1). As both names imply, this function operates on a k-stage basis. Each of the four bytes is transmitted via an XORed S-box and replaced from the list with a byte. Following this, the data is sent through another XORed S-box and the 4--bytes are multiplied by the Maximum Distance Separable (MDS), just as they were in (g). Alternatively, the words were divided into bytes [12, 18].

4. Related work

This part presents related work to the use of keys and key scheduling design in the Twofish Encryption Algorithm.

In 2009, Avakian et.al.[17] proposed improving the (Sosemanuk-stream-cipher-algorithm) by using the efficient-characteristics of the Twofish-block-cipher. They also suggested using its key-schedule, key dependent S-box to increase security and randomness and try to avoid the guess and determine attack of the Sosemanuk algorithm. They also compared it to Sosemanuk algorithms by using the tests of r. In comparison to the Sosemanuk algorithm, the findings revealed that the two suggested algorithms have good outcomes in boosting security and unpredictability.

In 2016, Abikoye& Ademarati [19] introduced a method called Twofish used to encode messages to be securely sent over the Internet. The algorithm has a (16 bit) key-length and was implemented using the (Java-programming-language) by the developers. Because a new ciphertext is generated for each round of encryption, this method is quick. The method generates a unique ciphertext each time it is used with a new key.

Hoomod and Hussein [13] in 2019, using the SALSA20 algorithm as a base, showed how to improve the Twofish method, making it more secure and memory efficient. The suggested combining the S-boxes' techniques with those of the SALSA20 algorithm and replacing the s-boxes' key-schedule with the equation in lu system, which will make it harder to decipher the cipher generated by the (S boxes). According to the findings, the updated method has a lower latency than the original algorithm.

Based on a novel operation called Cyclic Group Expanded (CGE #), Suhad Muhajer Kareem, et.al. [20], in 2020, , proposed a new method to improve Twofish-technique by boosting unpredictability. Using 30 cyclic group tables and multiplication in the Galois Field (GF), this is a brand-new 8-bit operation (28)! The (CGE #) procedure is used rather than (X-OR) within every Feistel round of Twofish. One key is chosen from 30 tables; the other is utilized in the encryption and decryption process. The goal of this method is to make the suggested algorithm more resistant to a wide range of assaults.

In 2020, Kareem and Rahma [14] proposed utilizing three keys instead of just one to control the varying bit widths of blocks (1, 2, 4, and 8 bits) that form the status table values, which the researchers recommended as an update mechanism for the Twofish algorithm. As a result, the complexity of the suggested method may be increased by including a Galois field (GF (2n)) into the tables. The findings of the NIST tests were positive: they have good resistance against differential attacks. The results were also acceptable in image encryption, increased security level, and high complexity. Randomness and security of the method were also improved by using different block bit sizes and numerous keys in each round.

5. Proposed method

In this section, we will explain the development of the Twofish algorithm and show the difference between the new advanced algorithm and the original.

It is known that the generation of subkeys in the Twofish algorithm occurs only once and is repeated in all plaintext blocks. The generation of these subkeys depends on the main key and several operations within the algorithm. However, in this research, we made the generation of these subkeys variable so that each block of the plaintext blocks a set of new and different subkeys. These keys are generated by Geffe's algorithm, and each block to be encrypted (128 bits) has a set of subkeys different from the other block. There is a table of those keys generated by Geffe's Algorithm and the master secret key. Corresponding to it from plaintext and with a length of 128 bits.

The new and developed Twofish algorithm is implemented in the same steps as the original Twofish algorithm, shown in the Figure (2), but here the subkeys are variable from one block to another, so the encryption process and the key generation process is variable in each block. The developed encryption process is done by taking the plaintext to be encrypted, which is 128 bits long, and split into 4-blocks of 32-bits each.

First step, input whitening: the input is processed with four secret key blocks (K3, K2, K1, K0) with the (XORed) process. The next stage is followed by the stage of 16 rounds, which starts from the results of the keys K0, K1, K2, K3 in the left and right side to be the result as inputs to the function F, one of these two inputs is shifted to the left with a length of 8 bits (<<<8). The function g consists of four bytes (wide key) that depends on the s-boxes and the MDS (Maximum Distance Separable) matrix. Through this matrix, a linear mixing of the outputs of the boxes is carried out. Then the two results from the function g are entered into the PHT (Pseudo-Hadamard Transform) and the result is treated with the key words K2r+9 and K2r+8. Then the two results are treated with the operation (XORed) to the resulting part of the input whitening operations, one of which was processed with the key K2 and then rotate or shift the result to the right by 1 bit (>>>1). On the other hand, the results of the K3 key are rotated with the block part to be encrypted to the left by 1 bit (<<<1) and then the result of the key is processed with the output results from the function F, the process of adding XORed. With this, the first round of the sixteen rounds has ended. Then the next step

in which the process of switching the right part to the left is done, and it is an input for the next round.

After executing all the rounds up to round No.16, the left part is changed to the right and the right part to the left (switching) and the results have reached the output stage after performing an XORed operation between the four results and the keywords K4, K5, K6 and K7 to produce the ciphertext (the encrypted block) with a length of 128 bits.

5.1 Generating variable subkeys by Geffe's Algorithm

Stream cipher was adopted in developing the Twofish algorithm, using Geffe's Algorithm (with optional length registers) in permanent and variable generation of subkeys included in.

The encryption process in the Twofish algorithm and Geffe's Algorithm consists of three shift registers of different lengths, X1=7, X2=13, X3=12, and their greatest common denominator is 1 [gcd (X1, X2, X3) = 1]. The feedback function for each shift register is a linear function that gives the greatest length $(2^7\text{-}1) * (2^{13}\text{-}1) * (2^{12}\text{-}1) = 4259852415$. The shift registers are connected with the join functions AND, XOR, and the negation process NOT. As shown in the Figure (3), X1, X2, and X3 are the outputs of the shift registers, and from this connection the nonlinear function y comes out as shown in the figure. These operations expressed by the following equation [21]: -

$$y= (X1 \wedge X2) \oplus (\neg(X2) \wedge X3) \quad (5)$$

Geffe's Algorithm generates new subkeys and places them in a table that defines the set of keys and their corresponding blocks.

These keys are used in the encryption process and the process of generating a variable ciphertext for all plaintext blocks that are used in the Twofish algorithm.

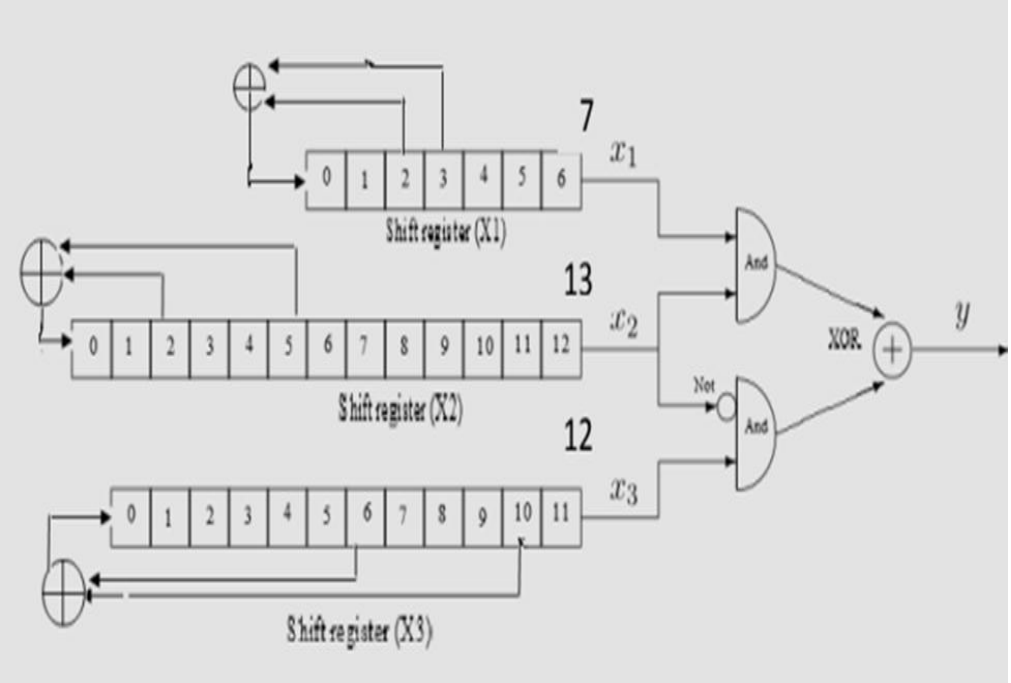

Figure 3: Geffe's generator algorithm [22].

5.2 Variable key scheduling

Key scheduling is the operations on the keys of shifting, rotating, and changing the key bits used in the encryption process to build the Twofish block cipher algorithm to be more robust and highly confidential. The Twofish algorithm needs its keys to be complex and not easily broken by cryptanalysts. The key table generates and prepares 40 key words K0,

K1....K39 and depends on the S-Boxes and g function, and the g function depends on its work on the two matrices RS (Reed Solomon) and MDS. In this study, us dealt with the new Twofish algorithm design with a new concept, which is the permanent change in the encryption process and the process of generating encryption keys for each block, which depends on the generation of new subkeys for all blocks of the plain text encryption and decryption so that each explicit block has a new set of subkeys different from one block to the next. Subkeys in the Twofish algorithm are divided into two types: S-Keys and K-Keys.

As shown in Figure (4), the process of generating subkeys starts with the main key, which is a secret shared and agreed upon by the sender and recipient. The length of this key is variable; it may be 128 bits or 192 bits up to 256 bits. For example, if the key is 128 bits long, so that the key is divided into a number of bytes, each byte is 8 bits long, and then 32 bits are selected, represented by four bytes of the key. Thus, in the case of 128 bits, the byte number (1, 5, 9, 13) is the selection to form those 32 bits, and then they are divided into (7 bits, 13 bits, and 12 bits), which represents the length and initial state of the input to the third registers (LFSR) of the Geffe generator system algorithm. Then the random bits are generated by the Geffe algorithm and those bits are used to generate the subkey set so that they are the number of explicit text blocks each. A subkey enters the table of keys to generate a new set of keys for that block.

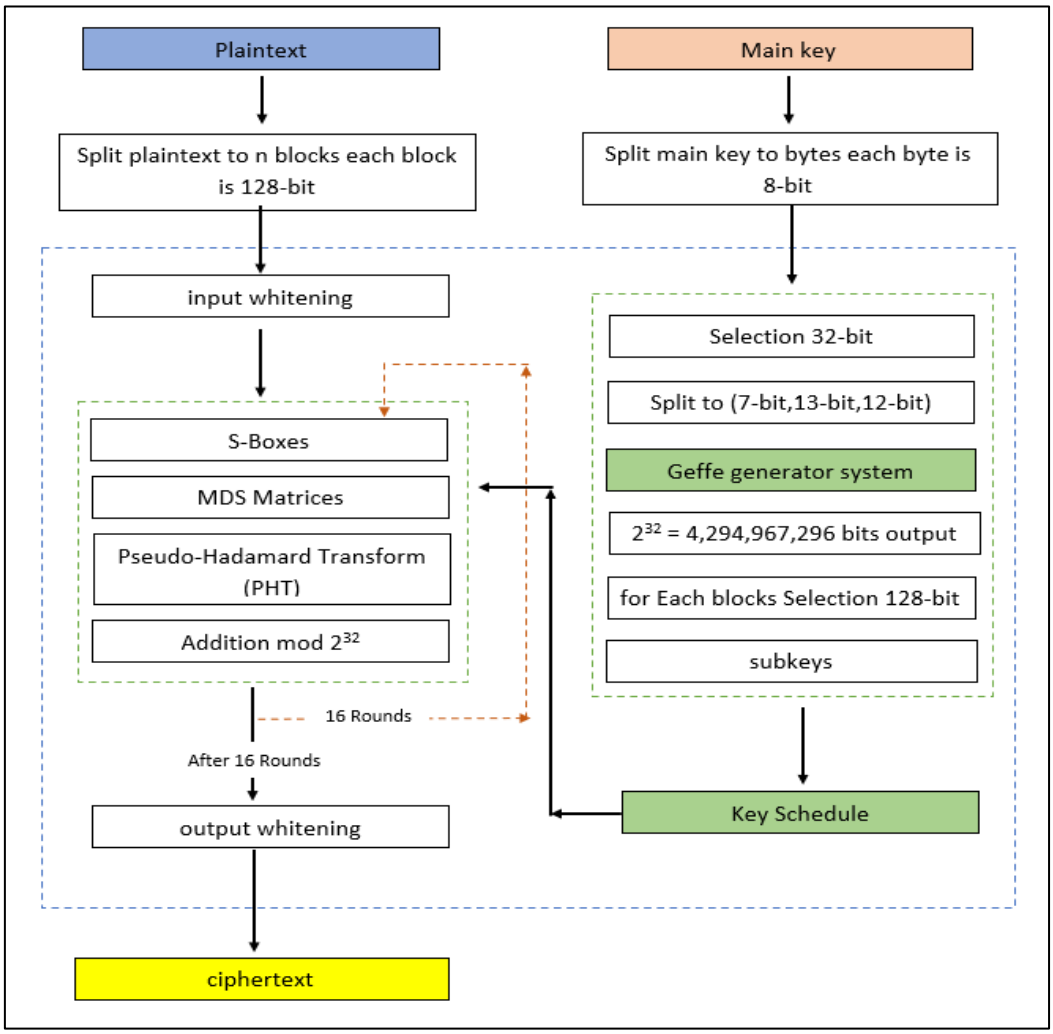

Figure 4: Proposed method.

Algorithm 1: Generating variable subkeys by Geffe's Algorithm

Input: Main key 128,192,256 bits, number of blocks for plaintext
Output: Set of subkeys
Begin:
Step 1: Split main key to number of bytes, where each byte = 8-bit
Step 2: Chosen 32 bit as input to Geffe generator system from main key
128-bit = combine byte number (1,5.9,13)
192-bit = combine byte number (1,9,13,21)
256-bit = combine byte number (1,13,17,29)
Step 3: Split the 32- bit into (7-bit,13-bit,12-bit)
that represent the Initial value for the registers X1, X2, X3;
step 4: Implement the Geffe generator system
x1 =LFSR (X1);
x2 =LFSR (X2);
x3 =LFSR (X3);
y= (x1 ∧ x2) ⊕ (¬ (x2) ∧ x3).
Step 5: Generate the matrix to store the Stream of random bit generated from Geffe generator
Mat = y
Step 6: for I = 1 to number of blocks
Spilt Mat to number equal to number of blocks
Subkey [i] = Mat[i]
Step 7: End for
Step 8: set the subkeys for all blocks
End

Algorithm 2: Encryption Algorithm

Input: Plaintext split into four 32 bits words (R_0, R_1, R_2, R_3), main key 128,192,256 bits
Output: ciphertext
Begin:
Step 1: for i = 1 to number of blocks
Step 2: subkey[i]=Geffe generator
Step 3: key scheduling (subkey[i]), generate the set of (s, k) subkeys
Step 4: $X_0=R_0 \oplus K_0$, $X_1=R_1 \oplus K_1$, $X_2=R_2 \oplus K_2$, $X_3=R_3 \oplus K_3$.
Step 5: For j=1 to 16
Step 6: $T_0=g(R_0)$, $T_1=g(<<<R_1)$. Where g represents a function using S-box.
Step 7: $F_0=T_0+T_1+K_{2r+8}$ Mod 2^{32}.
Step 8: $F_1=T_0+2T_1+K_{2r+9}$ Mod 2^{32}.
Step 9: $R_2=>>>1(F_0 \oplus R_2)$; $R_3=F_1 \oplus (<<<R_3)$.
Step 10: $X_0 \leftrightarrow R_2$; $R_1 \leftrightarrow R_3$
Step 11: end for
Step 11: $C_0=R_2 \oplus K_4$, $C_1=R_3 \oplus K_5$, $C_2=R_0 \oplus K_6$, $C_3=R_1 \oplus K_7$.
Step 12: ciphertext = C_0, C_1, C_2, C_3
Step 13: end for
End

Algorithm 3: Decryption Algorithm

Decryption is similar to Encryption with the same structure but the subkeys are applied in reverse order. Also, the replacements by the following: $R_2=F_0 \oplus (<<<1R_2)$, $R_3=>>>1(F_1 \oplus R_3)$

6. Results and discussion

This paper proposes a new improvement to the Twofish Algorithm by generating cipher keys for each block. This concept is new and unknown in all established block cipher algorithms; it is based on continuous creation of sub-keys for all blocks and the key generation mechanism for each block is dependent on its purpose. The proposed technique was assessed using the evaluation methods (the Complexity, Time of Encryption, Throughput, NIST-tests, Correlation coefficient, and Entropy) detailed below.

6.1 The complexity analysis

All mathematically generated encryption algorithms are theoretically breakable. But what sets them apart is the time required for the outage process. This time is related to the computational power available to the code analyzer and to the importance of the information over time. The attacker could use the random method to crack the encryption, or in this case, trial and error. The number of possibilities to try is 2^{128} attempts in the case where the key length used is 128 bits. This equals approximately $3.4 * 10^{38}$ attempts. So, if an attacker can try a key every 1 microsecond, the time required to try all keys would be about 10^{25} years.
However, in the case of using parallel processing by means of several processors that all address the problem simultaneously, a single processor that can process one key every one nanoseconds (if that is possible) can try approximately 10^{14} keys per day. Meaning that it takes approximately 10^{24} days to try all keys. This means that we need 10^{24} processors working in parallel at this rate to complete the detection within one day. Of course, achieving such a machine is impossible at the present time. The complications mentioned were for one key used to encrypt one block of the message. As for the Twofish algorithm, what is new here is that the key changes permanently with each block's encryption, so to decode a message of length N blocks, it will need $N*2^{128}$ tries and approximately $N*10^{25}$ years to try all the keys and break the message. It will require considerable time and effort. It is not easy to know all the subkeys since they are generated by Geffe's Algorithm, which is strong and secret and is not easy to crack. It should also be noted that it is generated with different data for each block to be encrypted. This is not easy, if not impossible to break.

6.2 Time of Encryption

Another measure of algorithm performance is cipher time, which is calculated using the time required to convert plaintext to ciphertext. Results were obtained using a laptop running Windows 11 and an Intel(R) Core (TM) i5-1135G7 @ 2.40 GHz processor.

6.3 Throughput

The measure of throughput, as applied in this context, is calculated using the equation [14]:

$$T = \text{plaintext size} / \text{time of encryption} \quad (6)$$

Table (1) and Figure (5) show the experimental results from the calculation of coding time and throughput for the original and modified Twofish algorithms. As can be seen from the results, the encoding time of the proposed Twofish algorithm is approximately the same as that of the original algorithm. This means good results for our method, as we get closer to the original time with more complexity compared to the original algorithm.

Table 1: Time of encryption and throughout 16 rounds

Input Size (KB)	Time of Encryption (in seconds)	
	Original	**Proposed**
1	0.4101	0.2908
2	0.8294	0.612
3	1.1764	0.9447
10	3.8166	3.9438
15	6.3244	5.5543
25	10.214	10.248
Average time	3.795	3.766
Throughput	20.386	22.189

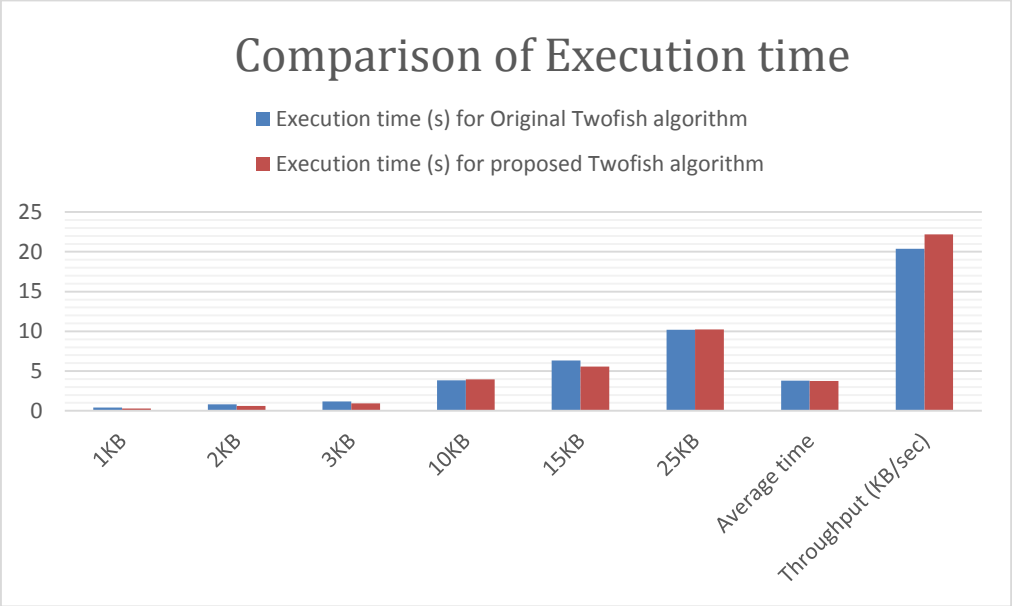

Figure 5: comparison of Execution time and Throughput

6.4 Analysis of NIST Tests

The encoding method's output should be unexpected and random. To calculate randomness, several methods were employed, including the NIST (National Institute of Standards and Technology), Hardcore tests, and TestU01. 15 NIST statistical tests were used to investigate the randomness of binary sequences generated by the known and proposed Twofish. To establish whether the output is random, the p - values (the probability value) is set to 0.01.

Table 2 displays the computed average scores. If the test results show a p-value that is asymptotically near to one, the output should look fully random. The p-value shows that the outcome is not chance. The pass status implies that the p-value for these tests is more than 0.001 and the result is satisfactory. The p-values of the majority of the suggested Twofish tests are larger than the p-values of the original Twofish, as shown in Table (2) and Figure (6). As a result, in the majority of tests, the suggested Twofish outperforms the original Twofish.

Table 2: Results of NIST test analysis.

Test no.	Statistical Test Name	Original Twofish algorithm		Proposed Twofish algorithm	
		P-value	Status	P-value	Status
1	Frequency Test (Monobit)	0.844	Pass	0.563	Pass
2	Frequency Test within a Block	0.837	Pass	0.423	Pass
3	Run Test	0.339	Pass	0.722	Pass
4	Longest Run of Ones in a Block	0.373	Pass	0.785	Pass
5	Binary Matrix Rank Test	0.133	Pass	0.267	Pass
6	Discrete Fourier Transform Test	0.090	Pass	0.061	Pass
7	Non-Overlapping Template Matching Test	0.369	Pass	0.605	Pass
8	Overlapping Template Matching Test	0.401	Pass	0.499	Pass
9	Maurer's Universal Statistical test	0.670	Pass	0.751	Pass
10	Linear Complexity Test	0.141	Pass	0.445	Pass
11	Serial test	0.233	Pass	0.493	Pass
12	Approximate Entropy Test	0.187	Pass	0.351	Pass
13	Cumulative Sums Test	0.601	Pass	0.239	Pass
14	Random Excursions Test	0.580	Pass	0.490	Pass
15	Random Excursions Variant Test	0.251	Pass	0.632	Pass

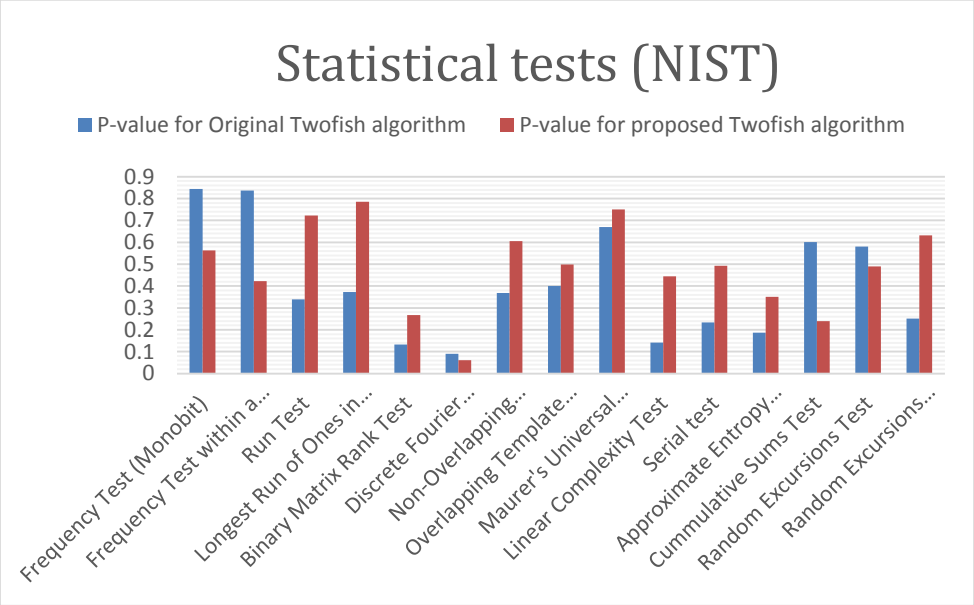

Figure 6: The statistical tests for algorithms.

6.5 Correlation coefficient

This metric is used to find out the strength of the correlation between the original content and the encrypted content generated by the encryption algorithm. Where the output of this scale indicates the extent of the ability of this algorithm to defend against statistical attacks. Standard values of r are in the range ($-1 \leq r \leq 1$). If the value of r is close to zero, the correlation between the original file and the encrypted file is completely uncorrelated. If the value of r is negative, this indicates that the encrypted file is inverted compared to the original file [20].

$$\text{Corr Coef}(x, y) = \frac{\sum_{i=1}^{n}(x_i - \mu(x))(y_i - \mu(y))}{\sigma(x)\sigma(y)} \quad (7)$$

Table (3) and Figure (7) show the results calculating correlation coefficients for the original and modified Twofish algorithms.

6.6 Analysis of Entropy

Entropy is one of the most prominent features for measuring the randomness of an encryption algorithm. The information entropy H(s) of the message source can be calculated as [23]:

$$entropy = H(X) = \sum_{i=0}^{M} p_i log_2(\frac{1}{p_i}) \quad (8)$$

From Table (4) and Figure (8), it is shown that the entropy value is closer to 8. The probability of accidental information leakage is extremely low if it is close to 8.

Table 3: Results of correlation coefficients

File size (KB)	Correlation coefficients	
	Original Twofish algorithm	Proposed Twofish algorithm
1	-0.0121	0.0243
2	0.0146	0.0321
3	0.0239	0.0247
10	-0.0062	0.0097
15	0.0049	0.0073
25	0.0024	0.0054
Average	0.0046	0.01725

Table 4: Results of entropy analysis

File size (KB)	Entropy analysis	
	Original Twofish algorithm	Proposed Twofish algorithm
1	7.8064	7.7711
2	7.9118	7.8962
3	7.9031	7.9385
10	7.968	7.983
15	7.98	7.988
25	7.965	7.984
Average	7.9224	7.926

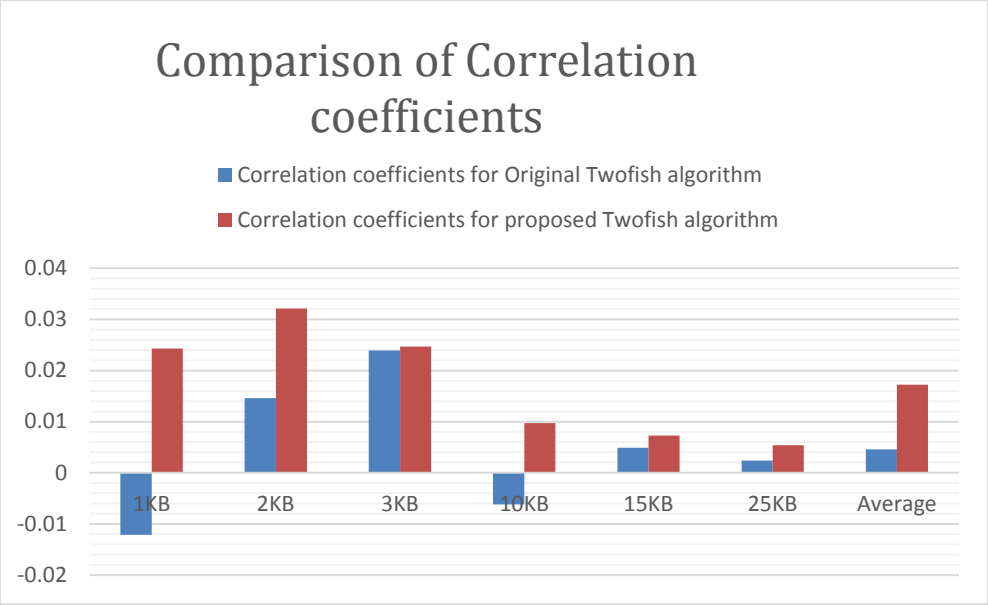

Figure 7: Comparison of correlation coefficients.

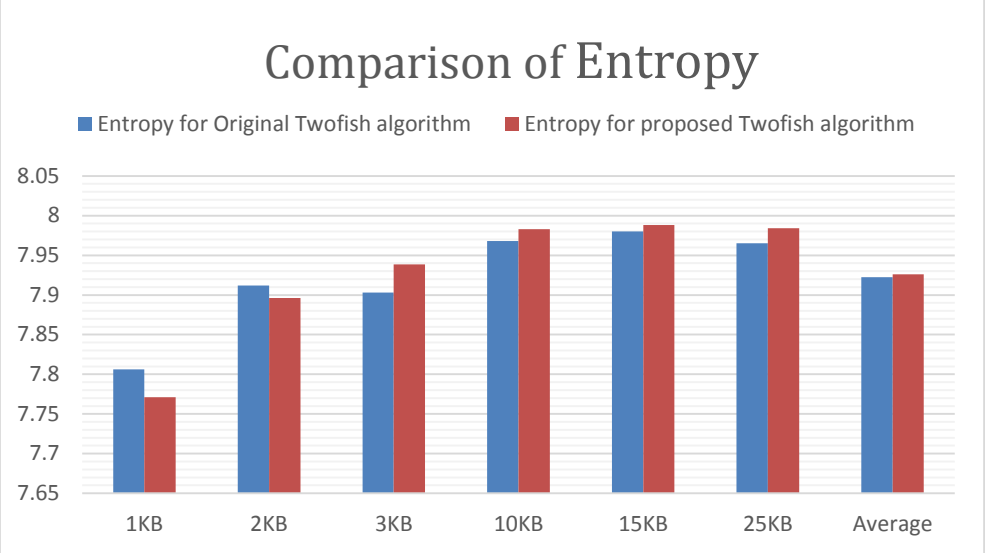

Figure 8: comparison of Entropy.

7. Conclusion

The new Twofish algorithm, using a key length of 128 bits, achieved results at a level of confidentiality and randomness much better than the original algorithm after using Geffe's Algorithm to generate a set of subkeys. It achieved the greatest success and best results by passing most of the statistical tests. It passed the frequency test, the serial test, and the run test. In addition, it passed the Cumulative Sums Test with success and results that exceeded the results of the original Twofish algorithm with high and clear percentages (the most random is the one that passes the most random tests and is of acceptable randomness if it passes at least three tests). Furthermore, the encryption time was comparable to the previous technique despite the additional complexity. As mentioned earlier, if the key length is 128 bits and the message length to be encrypted is 300 blocks, and if the assumed the time to try one key is 1 microsecond, it will need 2^{128} attempts to break. Thus, to break this encryption, it takes 10^{25} years to get the message in the original Twofish encryption. When encrypted using the new method, it will require $2^{128} * 300$ attempts and a time of $10^{25} * 300$ years to break this

encryption. This indicates that the developed Twofish algorithm is better, stronger, and more secret and difficult to break by attackers.

References

[1] H. Najm, H. K. Hoomod, and R. Hassan, "A proposed hybrid cryptography algorithm based on GOST and salsa (20)," *Periodicals of Engineering and Natural Sciences*, vol. 8, no. 3, 2020, doi: 10.21533/pen. v8i3.1619.

[2] E. K. . Gbashi, "Text Compression & Encryption Method Based on RNA and MTF ", *eijs*, vol. 58, no. 2C, pp. 1149–1158, Jan. 2022.

[3] M. Iavich, S. Gnatyuk, E. Jintcharadze, Y. Polishchuk, A. Fesenko and A. Abisheva, "Comparison and Hybrid Implementation of Blowfish, Twofish and RSA Cryptosystems," 2019 *IEEE 2nd Ukraine Conference on Electrical and Computer Engineering* (UKRCON), 2019, pp. 970-974, doi: 10.1109/UKRCON.2019.8880005.

[4] M. A. , D. H. , H. S. , & M. S. Mushtaque, "Evaluation of DES, TDES, AES, blowfish and two fish encryption algorithm: based on space complexity," *International Journal of Engineering Research & Technology* (IJERT) 3.4, 2014.

[5] E. Jeevalatha and S. Senthil Murugan, "Evolution of AES, blowfish and twofish encryption algorithm," *Int. J. Sci. Eng. Res*, vol. 9, no. 4, pp. 115–118, 2018.

[6] K. Aparna, J. Solomon, M. Harini, and V. Indhumathi, "A study of Twofish algorithm," *International Journal for Engineering Research and Application* (IJERA), vol.4, no. 2, pp. 148–150, 2016.

[7] Ms.S.Selvakumari, "Comparative Study on Blowfish & Twofish Algorithms in Iot Applications," *Journal of Engineering Research and Application*, vol. 10, Issue 02 (Series -III), 2020.

[8] I. Gorbenko, A. Kuznetsov, V. Tymchenko, Y. Gorbenko, and O. Kachko, "Experimental studies of the modern symmetric stream ciphers," in 2018 *International Scientific-Practical Conference Problems of Infocommunications. Science and Technology* (PIC S&T), 2018, pp. 125–128.

[9] A. A. Ghazi and F. H. Ali, "Robust and Efficient Dynamic Stream Cipher Cryptosystem", *eijs*, vol. 59, no. 2C, pp. 1105–1114, Jun. 2018.

[10] M. A. Kareem, N. H. M. . Ali, and S. A. Abead, "Modified Blowfish Algorithm for Image Encryption using Multi Keys based on five Sboxes", *eijs*, vol. 57, no. 4C, pp. 2968–2978, Jan. 2022.

[11] H. B. A. Wahab and S. A. Sarab, "Modify Symmetric Block Cipher Algorithm Using Generated Digital 3D Fractal Image," *eijs*, vol. 54, no. 4, pp. 955–964, 2013.

[12] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, and N. Ferguson, "Twofish: A 128-bit block cipher," *NIST AES Proposal*, vol. 15, no. 1, pp. 23–91, 1998.

[13] H. K. Hoomod and A. M. Hussein, "New Modified Twofish for Data Protection Using Salsa20 and Lü system," in 2019 *International Conference on Intelligent Computing and Control Systems* (ICCS), 2019, pp. 1189–1195.

[14] S. M. Kareem and A. M. S. Rahma, "A novel approach for the development of the Twofish algorithm based on multi-level key space," *Journal of Information Security and Applications*, vol. 50, p. 102410, 2020.

[15] O. de S. M. Gomes and R. L. Moreno, "A compact 128-bits symmetric cryptography hardware module," in 2016 8th *International Conference on Information Technology and Electrical Engineering* (ICITEE), 2016, pp. 1–5.

[16] P. Gehlot, S. R. Biradar, and B. P. Singh, "Implementation of Modified Twofish Algorithm using 128 and 192-bit keys on VHDL," *Int J Comput Appl*, vol. 70, no. 13, 2013.

[17] S. K. Avakian and R. S. Mohammed, "Improving Keystream Generation for Sosemanuk Stream Cipher Using Twofish Block Cipher," *Engineering and Technology Journal*, vol. 27, no. 10, 2009.

[18] S. Murphy and M. J. B. Robshaw, "Key-dependent S-boxes and differential cryptanalysis," *Des Codes Cryptogr*, vol. 27, no. 3, pp. 229–255, 2002.

[19] Abikoye, Oluwakemi & Ademarati, Oluwatoyin. "Information Security Model Using Twofish Encryption and Decryption Algorithm". *International Journal of Information Processing and Communication*, vol. 4, no. 1 & 2, Oct. 2016.

[20] S. Kareem and A. M. Rahma, "A Modified On Twofish Algorithm Based On Cyclic Group And Irreducible Polynomial In Gf (28)," *Al-Qadisiyah Journal Of Pure Science*, vol. 25, pp. 1–9, Oct. 2020, doi: 10.29350/jops.2020.25.1.997.

[21] F. Maqsood, "Modified Geffe Generator Circuit for Stream Cipher," *International Journal of Computer Science and Mobile Computing*, vol. 7, no. 11, pp. 105–111, 2018.

[22] F. Handayani and N. P. R. Adiati, "Analysis of Geffe Generator LFSR properties on the application of algebraic attack," in *AIP Conference Proceedings*, 2019, vol. 2168, no. 1, p. 20029.

[23] T. U. Haq, T. Shah, G. F. Siddiqui, M. Z. Iqbal, I. A. Hameed, and H. Jamil, "Improved Twofish Algorithm: A Digital Image Enciphering Application," *IEEE Access*, vol. 9, pp. 76518–76530, 2021.