DOI: 10.24996/ijs.2025.66.11.38

ISSN: 0067-2904

RC6 Key Generation Method using Permutation Last Layer Algorithm

Rana. M. zaki[1]*, Hala Bahjat Abdul Wahab[1] , Zaed. S. Mahdi[2]
[1] *Department of Computer Science, University of Technology, Baghdad, Iraq*
[2]*Center of Information Technology, University of Technology, Baghdad, 00964, Iraq*

Received: 14/5/2024 Accepted: 16/10/2024 Published: 30/11/2025

Abstract
Long Term Evolution-LTE is a mobile communication system used in wireless communication networks, while the development of communication methods used today and the high demand for mobile devices of all kinds have necessitated many attacks. This paper suggests a new way to model and build the RC6 algorithm, which creates dynamic keys using last-layer switching (PLL) algorithms, so that it can be used for more than just text and images in a 4G network. Images of different lengths and sizes are selected in the encryption and decryption process. Comparative results show that the proposed security algorithms outperform the standard algorithms. A lot of tests were done on both algorithms, and the proposed algorithm did better than the standard ones in many areas. These included execution time analysis, production encryption and decryption analysis, block-based analysis, potential security vulnerabilities in analysis and key search, statistical tests, entropy, correlation coefficient (CC), and capture-to-signal noise ratio (PSNR). All these criteria prove satisfactory security and good randomness in the proposed algorithm.

Keywords: Encryption and decryption, PLL, LTE, RC6, Security ,4G.

طريقة توليد مفتاح *RC6* باستعمال خوارزمية تبديل الطبقة الأخيرة

رنا محمد حسن زكي[1]* , هالة بهجت عبدالوهاب[1], زيد سمير مهدي[2]
[1] قسم علوم الحاسوب، الجامعة التكنلوجية، بغداد، العراق
[2] قسم مركز تكنلوجيا المعلومات، الجامعة التكنلوجية، بغداد، العراق

الخلاصة
التطور طويل الأمد- LTE هو نظام اتصالات محمول يستعمل في شبكات الاتصالات اللاسلكية، في حين أن تطور طرق الاتصال المستعملة اليوم وارتفاع الطلب على الأجهزة المحمولة من جميع الأنواع استلزم العديد من الهجمات. في هذه الورقة، اقترحت نمذجة وهندسة معمارية جديدة لخوارزمية RC6 لتوليد مفاتيح ديناميكية بناءً على خوارزميات الطبقة الأخيرة للتبديل (PLL) لتوفير بيئة عمل جديدة أخرى بجانب النصوص والصور في شبكة الجيل الرابع.

تم اختيار صور بأطوال وأحجام مختلفة في عملية التشفير وفك التشفير. تظهر النتائج المقارنة أن خوارزميات الأمان المقترحة تتفوق على الخوارزميات القياسية. تم إنجاز العديد من الاختبارات، بما في ذلك تحليل وقت التنفيذ، وتحليل التشفير وفك التشفير للإنتاج، والتحليل القائم على الكتل، وتحليل البحث عن المفتاح،

*Email: Rana.M.Zaki@uotechnology.edu.iq

والاختبارات الإحصائية، والإنتروبيا، ومعامل الارتباط(CC) ، ونسبة الضوضاء من الالتقاط إلى الإشارة (PSNR).كل هذه المعايير تثبت الأمان المرضي والعشوائية الجيدة للخوارزمية المقترحة.

1. Introduction

Nowadays, information technology is permeated into virtually all areas. Therefore, the amount of sensitive and critical information carried via the internet in different dimensions has incredibly increased. As a result, attempts to gain fraudulent or illegal access to private and confidential information have become increasingly attractive. Consequently, in the digital world, data security is still a major issue [1]. Thus, cryptosystems are used to achieve the protection of sensitive data. In this regard, there are two types of cryptosystems: asymmetric and symmetric. Asymmetric systems employ two distinct keys for both encryption and decryption processes, whereas symmetric systems utilize a single private key for both encryption and decryption [2]. On the other hand, several cryptographic primitives, such as stream ciphers, block ciphers, message authentication codes, and hash functions, have been developed in the recent decade, and they outperform the standards of the traditional cryptographic, as shown in Figure 1.

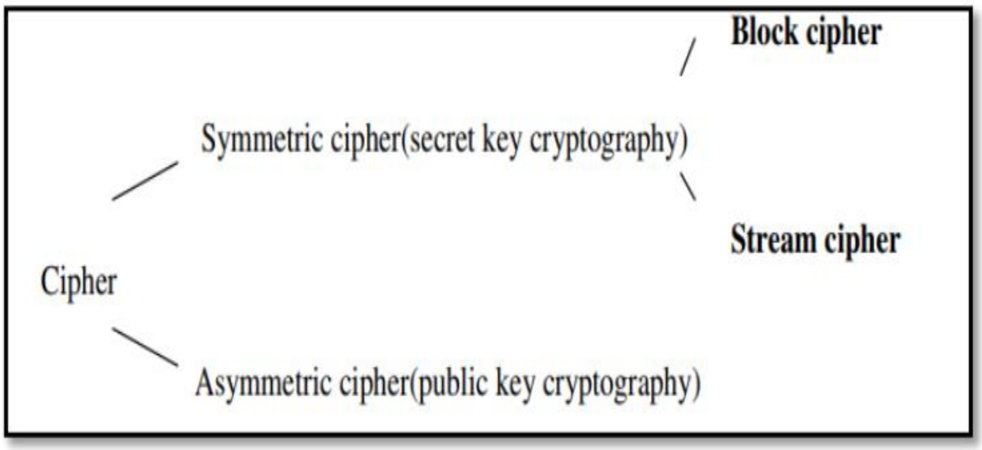
Block cipher
Symmetric cipher(secret key cryptography)
Stream cipher
Cipher
Asymmetric cipher(public key cryptography)

Figure 1: Block cipher and stream cipher [2]

The integrity and confidentiality algorithms in the LTE security algorithm can be defined as two powerful flow algorithms that can withstand a variety of attacks and can be optimized to achieve higher performance and productivity than previous work. Furthermore, the stable algorithm still has some flaws that need to be addressed. More research is needed in this area. LTE, such as its ancestry, is threatened by several types of attacks like eavesdropping, fraudsters, viruses, and different attackers [3, 4]. The LTE and LTE-A algorithms consist of three sets: the first, 128-EEA1/128-EIA1, is derived from the SNOW 3G algorithm; the second, 128-EEA2/128-EIA2, is based on the AES algorithm; and the third, 128-EEA3/128-EIA3, is based on ZUC [5-8]. This paper proposes the use of the Rivest Code 6 (RC6) algorithm, one of the strongest encryption algorithms in block ciphers, to encrypt data between sender and receiver in a 4G network and generate dynamic keys at random. The proposed RC6 algorithm outperforms the traditional RC6 algorithm in terms of data transfer rate for both encryption and decryption.

2. Related Work

The authors in [9] proposed the With RC6-Lite, just five encryption rounds are employed, and the encryption and decryption operations work with 8-bit blocks rather than 32-bit ones. This seeks to deliver security with reduced computing overhead and faster encryption.

According to the paper's results, the average avalanche effect after changing the key was 52.38%, a value that falls within the desirable range of 45%–60% for a good cryptographic technique. This demonstrates that RC6-Lite is effective at thwarting key-based cryptanalysis since it functions effectively when exposed to key alterations. Avalanche Impact of Plaintext Modifications: When changing the plaintext, the avalanche impact was 26.49% on average. This indicates that 26.49% of the ciphertext bits changed when a single character in the plaintext was changed. This value surpasses the previous traditional values.
In [10], the authors suggested a system that utilizes the RC6 encryption algorithm and incorporates various modes of operation, including output feedback, cipher block chaining, electronic codebook, and cipher feedback. The system's main use is the encryption of color images with low levels of detail, an area where conventional encryption methods frequently fail, especially when it comes to masking patterns and minute pixel changes. The system's efficacy was assessed by the authors using a variety of metrics, including NPCR, UACI, correlation coefficient, entropy, and irregular deviation. The suggested cryptosystem, which encrypts color images with little information, is quite successful at utilizing RC6 in many operating modes, especially CBC, CFB, and OFB. According to the study, OFB mode, particularly in loud conditions, provides the optimum compromise between security and performance.

The authors in [11] enhanced data encryption performance by parallelizing the RC6 algorithm. The primary goal of this paper is to analyze the speed and efficiency differences between the standard RC6 implementation and its parallel counterpart. processor and 6 GB RAM. It was discovered that the parallel implementation of RC6 performed noticeably quicker than the sequential version, particularly as the amount of input data increased. The authors tested the sequential and parallel implementations of RC6 using execution times that were measured for various string lengths ranging from 100 to 80 million characters, as well as a speedup ratio between the execution times of the sequential and parallel implementations. They conducted their experiments on a machine featuring an Intel Core i5 processor and 6 GB of RAM. It was discovered that the parallel implementation of RC6 performed noticeably quicker than the sequential version, particularly as the amount of input data increased. With input sizes greater than 10, the speedup was more pronounced, improving by an increase to 4.1 times for strings of 10.

In [12], the authors proposed an enhanced RC6 encryption technique specifically for wireless sensor networks (WSNs). The encryption performance was evaluated using varying key lengths (16 bytes, 24 bytes, and 32 bytes) and different encryption rounds. In conclusion, the modified RC6 algorithm offers a practical solution for improving security in wireless sensor networks, addressing the challenges of power efficiency, computational limitations, and communication stability.

The authors in [13] proposed a double encryption system that combines the RC6 algorithm with chaotic synchronization by utilizing the chaotic nature of signals; this integration improves encryption and strengthens the system's defense against attacks. The study's findings demonstrated the effective collaboration between the RC6 algorithm and chaotic synchronization in the system. The use of chaotic synchronization enhances the system's complexity and unpredictability, thereby strengthening its resistance to attacks, including those from quantum computers. The chaotic system provides an additional layer of security to the encrypted data. Quicker Synchronization Furthermore, the study discovered that IGA produced superior feedback gains in comparison to conventional techniques, leading to synchronization that was quicker and more dependable. Finally, the study's fuzzy controller successfully maintained the system's stability and safety, maintaining synchronization even in the face of external disruptions and ensuring security under a variety of conditions.

3. RC6 Algorithm

Confidentiality and data Integrity is the two most important features of the cryptography that can be achieved by the use of symmetric ciphers. RC6 is a symmetric block cipher based on RC5 and developed by Rivest, Sydney, and Yin for RSA security. Like RC5, RC6 is a parameterized algorithm where the block size, the key size, and the number of rounds are variables. RC6 was developed to meet the requirements of AES. RC6 Proper has a block size of 128 bits and supports key sizes of 128, 192, and 256 bits, like RC5. Moreover, it incorporates an additional multiplication operation that is absent in RC5. This multiplication operation is used to make the rotation operation dependent on every bit in a word. RC6 can be more accurately specified as RC6-w/r/b, where the word size is w bits, encryption consists of a nonnegative number of rounds r, and b represents the length of the encryption key in bytes. The standard value of w = 32, r = 20, and b = 16, RC6 is used as the shorthand to refer to these versions. When some other value of w, b, or r is used, the parameter values will be specified as RC6-w/r/b. The following Figure 2 represents the RC6 encryption algorithm [14–17]. For all variants, RC6-w/r/b operates on units of four w-bit words using the following basic operations:

A + B Addition of two’s complement words
A – B Subtraction of two’s complement words
$A \oplus B$ Exclusive-OR with bit-wise words
A <<< B Word A left cyclic rotation by B bits
A >>> B Word A right cyclic rotation by B bits
$A * B$ The integer multiplication modulo 2^{w}
(A, B, C, D) = (B, C, D, A) parallel assignment $F(x) = x(2x+1) \bmod 2^{w}$

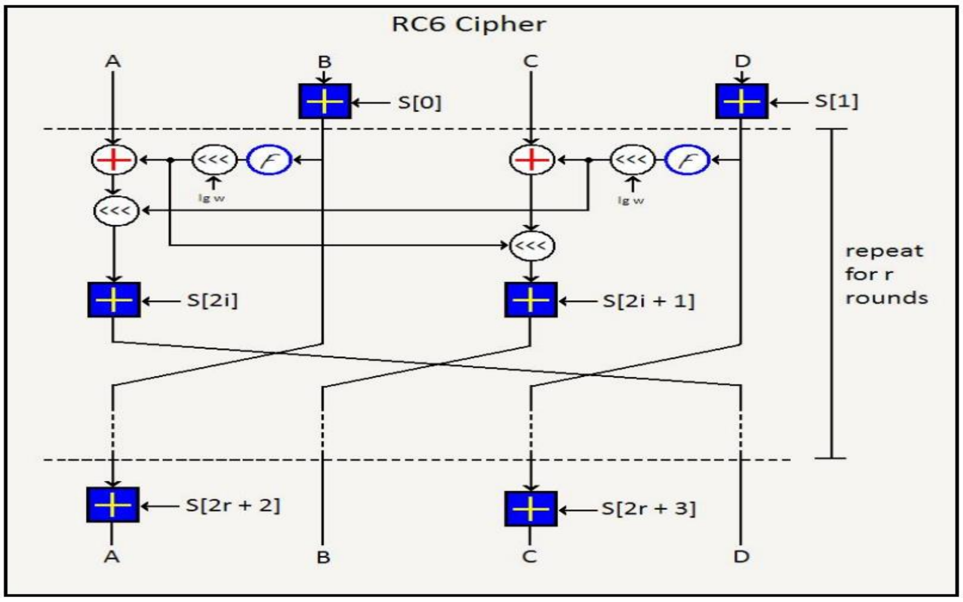

Figure 2: RC6 encryption algorithm [16]

The RC6 algorithm includes three components, a key expansion algorithm, an encryption algorithm, v and a decryption algorithm.

3.1 key expansion algorithm

The key expansion algorithm is used to expand the user-supplied key to fill an array S. The user must supply a key of b bytes, from which (2r+4) words are derived and stored in a round key array S. The key bytes are loaded into an array L [0,..,c-1] of c=(b/u) where u=w/8 in little-endian order. Any unfilled byte positions in L are zeroed. The (2r+4) derived words are stored in array S [0,...,2r+3] for later encryption or decryption processes. The key schedule also uses a magic constant for Pw and Qw, as shown in Table 1. The algorithm 1 is utilized for key expansion [17].

Table 1: Magic Constant Values P_w and Q_w

W	**16**	**32**	**64**
p_w	B7E1	B7E15163	B7E151628AED2A6A
Q_w	9E37	9E3779B9	9E3779B97F4A7C15
Odd(x) function will produce the odd integer closest to x			

Algorithm (1): Key Expansion-Standard RC6 algorithm [9]

```
Input: b=User supplied key loaded in array L[],r=Number of rounds
   $P_w$=odd ((e-2)$2^w$)= B7E15163 // e=2.718281828459…..
   $Q_w$=odd ((t-2) $2^w$)= 9E3779B9 // t= 1.618033988749 ….
Output: w-bit round keys s[0,…,2r+3]
Begin
S[0]= $P_w$
For i=1 to 2r+3 do
S[i]=S[i-1]+ $Q_w$
 X, Y , i ,j= 0
For v = 1 to (3 x (2r+4)) do
 {
  X=S[i]=(S[i]+X+Y)<<<3
  Y=L[j]=(L[j]+X+Y)<<<(X+Y)
   i=(i+1)mod (2r+4)
   j=(j+1) mod c
  }
```

3.2 Encryption Data Procedure

Four w-bit registers A, B, C, and D contain the initial input plaintext as well as the output ciphertext at the end of encryption. The first byte of plaintext is placed in the least significant byte of A; the last byte of plaintext is placed into the most significant byte of D. As shown in algorithm 2.

Algorithm (2): Encryption Algorithm-Standard RC6 algorithm [9]

Input: Plaintext stored in four w-bit input registers A,B,C,D , w-bit round keys S[0,…2r+3],Number of rounds.
Output: Ciphertext stored in A,B,C,D

```
Begin
   B= B+S[0]
    D= D+S[1]
    For i=1 to r do
        {
          t= (B*(2B+1))<<< log w
          u= (D*(2D+1))<<< log w
          A=((A⊕ t)<<< u)+ S[2i]
          C= (C ⊕ u)<<< t) + S[2i+1]
           (A,B,C,D)=(B,C,D,A)
          }
          A=A+S[2r+2]
          C=C+S[2r+3]
End
```

3.3 Decryption Data Procedure

For decryption of cipher text, load this cipher text into registers A, B, C, and D. The algorithm uses integer subtraction modulo 2w and right rotation on registers for getting plain text [17], as shown in algorithm (3).

Algorithm (3): Decryption Algorithm-Standard RC6 algorithm [9]

Input: Ciphertext stored in four w-bit input registers A,B,C,D ,w-bit round keys S[0,…2r+3], Number of rounds.
Output: Plaintext stored in A,B,C,D

```
Begin
   C= C-S[2r+3]
   A= A-S[2r+2]
    For i= r down to 1 do
    {
     (A,B,C,D)=(D,A,B,C)
     u= (D*(2D+1))<<< log w
      t= (B*(2B+1))<<< log w
     C=((C - S[2i+1])>>> t) u
     A =((A – S[2i])>>> u) t
      }
       D=D-S[1]
       B=B-S[0]
End
```

4. PLL Algorithms (Permutation Last Layer)

PLL is an enhanced form of MC's technology, which, as Jessica Fridrich's progress entails memorizing a large number of algorithms with a logical connection between them. Which are used by the vast majority of speed cubes these days and called the Fridrich Method. PLL, this part is to change the position (permutation) of the pieces of the last layer without rotating them. Then the cube would be solved [18]. The sophisticated method employed by Jessica F. layers the puzzle, and so one should aid in the layering of each successive layer of the cube with the use of algorithms at each one of the steps without harming the actual parts, which can be seen in Figure 3.

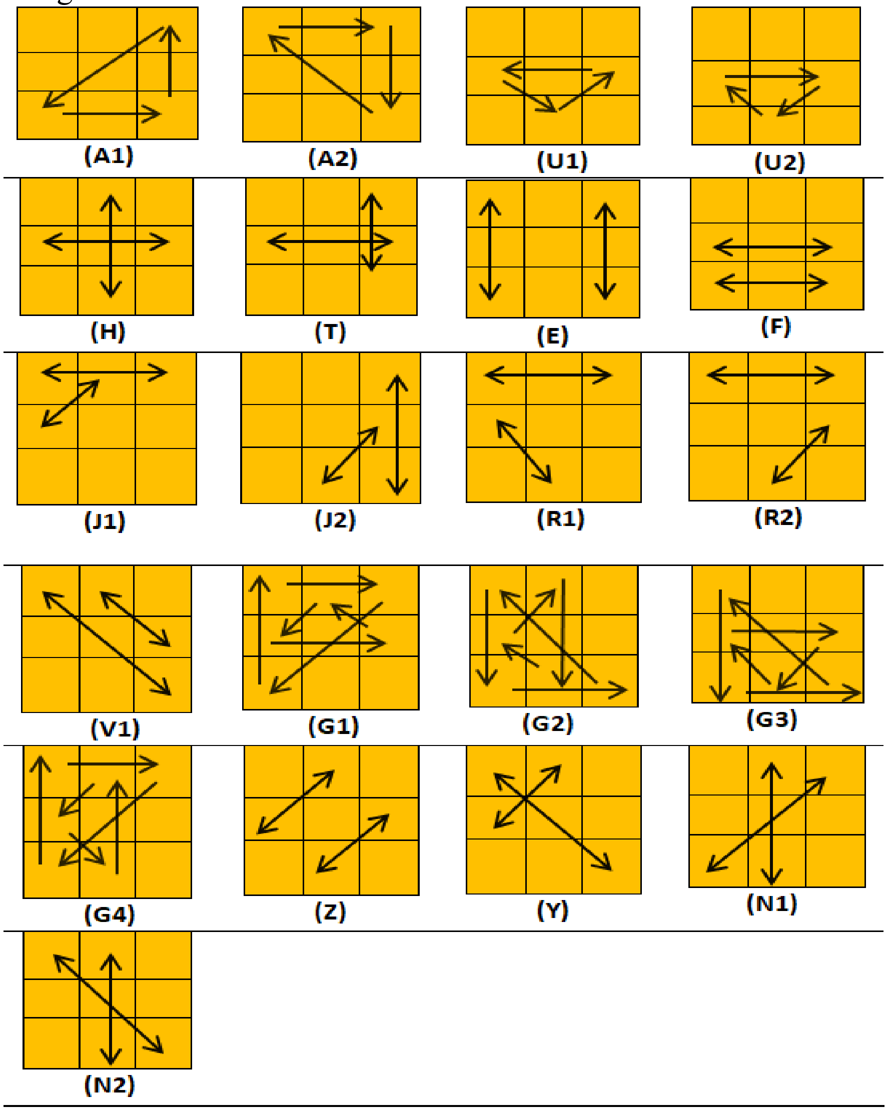

Figure :3 Permutation of last-layer Algorithms [18]

5. Evaluation Metrics

To measure the evaluation of encryption systems, a number of metrics are used to ensure the strength of encryption and decryption [15, 19].

6. Evaluation of Entropy

The amount of randomness in the information contained is measured by the information entropy. The entropy of an image determines whether it is a random image with random pixel values. The term entropy refers to the amount of information that can be stored, as shown in the equation.

$$E = -\sum_{g=0}^{n} P(g)\log_2(P(g)) \qquad (1) \qquad [20]$$

The g is present the probability of pixel value $P(g)$, and n the number of pixel values (0-256) for the gray level image 2^8. Since there is a correlation in the original plain image and pixel values are rarely random, the value of the entropy is usually less than 8. When all pixel values are distributed randomly, entropy reaches its maximal ideal value, which is 8. According to the findings, the encrypted images' entropy is very close to the ideal value of 8 [20].

5.2 Correlation Coefficient (CC)

The CC is a statistical measure that describes the relationships between two variables. In most images with visible content, each pixel has a strong relationship with its neighbors in all, horizontal, vertical, and diagonal. The result of a good encryption model should be encrypted images without such correlations in the adjacent pixels. According to equation 2, the correlation coefficient of adjacent pixels is calculated [20].

$$e(x) = {}^{1}\!/_{n} \sum_{i=1}^{n} xi \qquad (2)\ [21]$$

$$d(x) = {}^{1}\!/_{n} \sum_{i=1}^{n} (xi - e(x))^2 \qquad (3)[21]$$

$$\text{Covariance}(x,y)\ {}^{1}\!/_{n} \sum_{i=1}^{n} (xi - e(x))(yi - e(y)) \qquad (4)[21]$$

$$CC_{xy} = \frac{\text{Covariance}(x,y)}{\sqrt{d(x) * d(y)}} \qquad (5)[21]$$

While are neighboring pixels of premier or encrypted images, e(x) represents mean value, d(x) represents deviation to the mean value, covariance (x, y) represents covariance between neighbor pixels, and is the correlation coefficient. The correlation coefficients of pairs of neighboring pixels have been chosen randomly in horizontal, vertical, and diagonal locations to test the correlation in the original and encrypted images, and their coefficients of correlation are determined by calculating using equation 5 [21].

3.5 Peak Signal to Noise Ratio (PSNR) Analyses

It can also be utilized in order to conduct objective evaluations of encryption techniques. Calculated by considering a plain image as the signal and an encrypted image as the noise. The equation below may be used to compute the value of the PSNR.

$$\text{PSNR} = 10 \log_{10}\left(\frac{R}{MSE}\right) \qquad (6)[20]$$

R represents 255 for eight bits in an image that has the highest intensity, and the mean square error (MSE) is calculated according to the equation.

$$MSE = \frac{1}{mn} \sum_{i=1}^{m} \cdot \sum_{j=1}^{n} ((I(i,j) - I'(i,j))^2 \qquad (7)[20]$$

Where I (i,j) represents the pixel value of the original plain image and (i,j) represents the pixel value of the encrypted image at the given location (i,j). To measure the encryption/decryption algorithm performance, follow the instructions in Table 2 below. If these results are obtained, the encryption is powerful, indicating that attackers will have difficulty retrieving the original plain image from the encrypted image [20, 22].

Table 2: Encryption and Decryption algorithm performance [23].

No.	Image A - Image B	MSE	PSNR
Case1	Original - Encrypted	high	low
Case2	Original - Decrypted	zero	infinity

7. The Proposed Algorithm

This paper presents a development of the RC6 algorithm that utilizes PLL algorithms to generate multidimensional keys for encryption and decryption, thereby enhancing security. It also employs varying input key sizes for each image or text sent within the 4G wireless network. The development of the RC6 algorithm also includes adding additional registers to the encryption and decryption processes and increasing the word size from 32 bit to 64 bit. The development of the RC6 algorithm dealt with three parts: a new architecture for the RC6 algorithm, generating keys securely, and an encryption/decryption process.

7.1 Stage1: A new architecture for the RC6 algorithm

The proposed development of the RC6 algorithm is a block cipher with 384 bits divided into six registers (A, B, C, D, E, and F), where each register has 2W (64 bits), round 20, and a 384-bit key size instead of 128. This provides a higher level of security compared to RC6 with a larger key size. This adds another layer of protection to the cipher message or image. This makes it more difficult for attackers to use brute force to break the encryption. In certain situations, the algorithm can work more efficiently thanks to the increased data throughput resulting from the use of the additional six registers. This is especially true if the system or hardware allows parallel processing. However, we note that increasing the number of registers results in a greater increase in memory, which is necessary for storing and processing data during encryption and decryption. Using more memory can overcome this minor problem. Finally, it can be said that increasing the key size and increasing the number of registers enhances security with a slight or imperceptible impact on performance and speed, while slightly increasing memory consumption. Figure 4 represents the development of the RC6 algorithm.

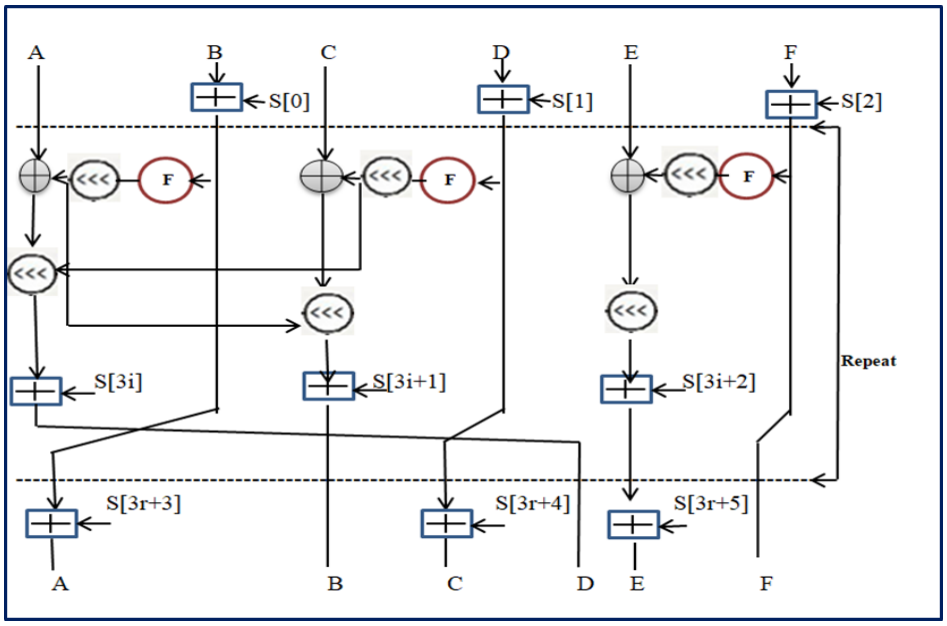

Figure 4: Proposed Architecture for RC6 Algorithm

8. Stage2: Key Generation Process (Key Expansion)

One of the improvements to the proposed RC6 algorithm is the ability to generate static and dynamic keys for two types of data sent (either text or image). The Permutation Last Layer algorithm used in Rubik's cube was used because of its speed and randomness in key generation. This feature prevents many attacks during real-time data transmission. This method was used to generate keys with the work of two versions with 64/20/348 and the output keys generated (3r+6), starting from [0, 1, to 3r+5].

- **The first version** includes one block of keys with a length of 48 bytes and the generation of 66 keys of the size (64-bit) of each from PLL, and it remains static to encrypt and decrypt all the messages despite the high randomness in the generation of the key, and the exposure to attacks is low.
- **The second version** generates the number of keys according to the length of the message, as the process of generating the first block of keys from the first PLL includes 66 keys and then makes permutations of the Rubik's cube to generate the second block of keys, i.e., 66 different keys, and so on. So, the key will be non-fixed (dynamic), and this type in Compared to the first version, the keys are more difficult to attack and guess. Because of the way games are thought of, the way they are programmed, and the fact that there are many different types of encryption in game programming, including the PLL algorithm, this paper relies on the PLL algorithm to show an algorithm for making keys, which is shown in algorithms 4 and 5. Table 3 shows a comparison between the traditional RC6 algorithm and the proposed RC6 algorithm for the two versions in 4G, with the difference in key generation caused by the PLL algorithm.

```
Algorithm4: Generate keys with Permutation Last Layer Algorithms
Input: Number of rounds=20, Word=64, Message, Length of massage, number of keys.
Output: Generate number from keys according to length of massage.
Begin
Step1: M = input message
Step2: L = length of massage
Step3: SS1[] = one dimension array for hexadecimal  to generate number of keys with using
PLL algorithms based    on message
Step4: P [] = permutation for length of SS1[]
Step5:  S []= Convert  P[] from hexadecimal to binary
Step6:  generate key(nb): if the static encryption for one block key(nb=1) then generate 66
key
    jj=0
   W= [0] *66 * nb
   for z in range(nb):
      s1=cubic loop () // to generate first cubic
      s2=cubic loop () // to generate second cubic
      st="" // create array empty
      for i = 0 to s1 and s2
         for j = 0 to 16
               a="". join (s1[i,j])
               b="". join (s2[i,j])
               c=a+b // concatenation to generate 66 keys
               st=st+c
      Key bit = bytes To Binary(st)
Step7:  for i= 0 to length (Key) // Loop through blocks of 6 words A,B,C,D,E,F
         W[jj] = [i:i + 64]
         jj=jj+1
 Step6: To generate more than one block go to Step3
End
```

Table 3: Compared between traditional RC6 algorithm and proposed RC6 algorithm.

Parameters	traditional RC6 algorithm	proposed RC6 algorithm version (1)	proposed RC6 algorithm version (2)
The Block Size in bits	128	384	384
The Number of Block Size	4w	6W	6W
bit size of both a word(W)	32	64	64
Length of the Key	128	384	384
The number of rounds(r)	20	20	20
The number of rounds in the Key	2r + 4	3r + 6	3r + 6

Function has been used	t = (B × (2B +1))
	u = (D × (2D+1))
		t = (B × (2B + 1))
		u = (D × (2D +1))
		p= (F × (2F + 1))
			t = (B × (2B + 1))
			u = (D × (2D + 1))
			p= (F × (2F + 1))

Operation has been used	*,+ , - , <<< , >>> ,⊕	*,+ , - , <<< , >>> ,⊕	*,+ , - , <<< , >>>⊕
Generate Keys	Key Static for all plain text	Generate Static Key for all plain text by using PLL algorithm	Generate Dynamic Key for all plain text by using PLL algorithm

8.1 Stage 3: Encryption/Decryption process

The encryption and decryption processes will be done, as mentioned earlier, for both the texts and the images, with no image processing operation to maintain the transmission time within the 4G network environment.

8.1.1 Encryption process

The initial input plaintext (message) or image is stored in six w-bit registers A, B, C, D, E, and F, which are used in RC6 encryption. Plaintext's first byte is stored in A's least significant byte. The most significant byte of F is filled with the last byte of plaintext. The arrangement of (A,B,C,D,E,F) = (B,C,D,A,E,F) is similar to the parallel assignment of values (bytes) on the right to the registers on the left 2. The following is an encryption algorithm 5.

```
 Algorithm 5: The proposed RC6 Encryption Algorithm for text (message)
Input:  Plain text or Image stored in six input registers A,B,C,D,E,F
Number of rounds =20, w-bit round keys S[0, 1, . . . , 3r + 5],W=64
Output: Encryption Text or Image stored in six output register B,C,D,A,E,F
Begin
w=64
 Step1: if use the static key for encryption all massage then
 Step2: Cube generate key (1)   // key generation by cube 1 for all massage
Step3: for i= 0 to length Plain text or length of image // Loop through blocks of 6 words   A, B, C, D, E, F
```

```
 A = [i + w]
 B = [i + w:i + 2 * w]
 C = [i + 2 * w:i + 3 * w]
 D = [i + 3 * w:i + 4 * w]
 E = [i + 4 * w:i + 5 * w]
 F = [i + 5 * w:i + 6 * w]

 B = (B + W [0])
 D = (D + W [1])
 F = (F + W [2])
Step4: for i = 1 to r do
 t = (B × (2B + 1)) <<<1g w //Rotating a 64-bit word left by 3-bit positions
 u = (D × (2D + 1)) <<<1g w
 p= (F × (2F + 1)) <<<1g w

A = ((A ⊕t) <<< u) + W [3i]
C = ((C ⊕u) <<< t) + W [3i + 1]

  A, B, C, D, F, E = B, C, D, A, F, E
Step5: Return to Step4
 A = (A + W [3 * r + 3])
 C = (C + W [3 * r + 4])
 E = (E + W [3 * r + 5])
Step6: Else go to Step 2
Step7: Encryption text or image
End
```

8.1.2 Decryption Process

Firstly, it takes the first encrypted block of data and divides it into six registries using the same key that was agreed upon in the encryption, and so on sequentially to obtain the plain data. The algorithm 6 illustrates the decryption process.

```
Algorithm6: The Proposed RC6 Decryption algorithm for text or image
Input: Cipher (text or image) stored in six input registers B,C,D,A,E,F Number of
rounds, r =20, w-bit round keys S[0, 1, . . . , 3r + 5],w=64
Output: Plain text or original image) stored in six output registers A, B, C, D, E,
and F
Begin
Step1: if use the static key for decryption all massage then
Step2: Cube generate key (1)   // key generation by cube 1 for all massage
Step3: for i= 0 to length Plain text //Loop through blocks of 6 words A, B, C, D,
E, F
    A = [i + w]
    B = [i + w:i + 2 * w]
    C = [i + 2 * w:i + 3 * w]
    D = [i + 3 * w:i + 4 * w]
    E = [i + 4 * w:i + 5 * w]
    F = [i + 5 * w:i + 6 * w]
   A = (A – W [3 * r + 3])
   C = (C – W [3 * r + 4])
   E = (E – W [3 * r + 5])
```

```
Step4:    For i= r down to 1
  A, B, C, D, E, F = D, A, B, C, E, F
t = (B × (2B + 1)) <<<1g w // Rotating a 64-bit word left by 3 bit positions
u = (D × (2D + 1)) <<<1g w
p= (F × (2F + 1)) <<<1g w
 A = XOR (A − W [3 * i]) >>> u), t) // Rotating a 64-bit word right by 3-bit
positions
C = XOR (((C – W [3 * i + 1]) >>> t), u)
 E = XOR (((E – W [3 * i+2]) >>> P), p)
B = (B – W [0])
D = (D – W [1])
 F = (F – W [2])
Step5: Return to Step4
 B = (B – W [0])
 D = (D – W [1])
  F = (F – W [2])
 Step6: Else go to Step 2
Step7: Plain text or original image
End
```

Analysis and Discussion of the Experimental Results

In this section, the RC6 algorithm is tested using several texts and images with different content. The results of the traditional RC6 algorithm are compared with the proposed RC6 algorithm for four records, block length 384, number of keys 66, and the same key generation method as the traditional RC6. Then, it is compared with the first proposed version of the RC6 algorithm by generating static keys using the PLL algorithm, and finally, it is compared with the second proposed version of the RC6 algorithm to generate dynamic keys also using PLL, algorithm.

8.2 Execution Time Analysis

Both versions of the proposed RC6 algorithm demonstrate improvements by using 6 registers instead of the standard 4, allowing for the encryption of both text and images. The proposed RC6 algorithm version 1 encrypts text and images by generating 66 fixed keys using the PLL algorithm for image encryption. The proposed RC6 algorithm version (2), on the other hand, generates a variable number of keys with different lengths based on the length of the text and the size of the image, also using the PLL algorithm.

A comparison was made between the traditional RC6 algorithm and the two proposed versions in terms of encryption and decryption times (in milliseconds) for both text and image files of various types, as shown in Tables 4 and 5. The results indicate that the proposed RC6 algorithm version (2) outperforms the others in terms of execution time for encryption and decryption, particularly with the text files and images used in this study. This improvement is attributed to the increase in word size from 32-bit to 64-bit for each register and the use of 6 registers, compared to the 4 registers in the standard RC6 algorithm.

Table 4: Execution Time in encryption.

Total Execution Time in milliseconds					
File Name	**File Size (KB)**	**Traditional RC6 algorithm**	**Proposed RC6 algorithm**	**proposed RC6 algorithm version (1)**	**proposed RC6 algorithm version (2)**
Iraq.txt	48	9.16	5.27	0.64	0.43
R.txt	72.46	11.43	6.21	7.76	7.80
Iraq.bmp	16.6	6.4	5.49	4.39	4.38
date.jpg	218	84.86	44.47	41	46.61
ali.jpg	93.2	58.61	44	43	32.78
lana.jpg	172	55.70	52	40.87	41.42
canser.png	41.8	17.57	14.35	14	14.24
soner.jpg	24.4	6.95	5.19	5	3.90
Baby.jpg	70.7	19	17	16	17.17
Total	757.16	269.68	193.98	185.02	168.73
Throughput Encryption		2.8076	3.9032	4.0923	4.4874

Table 5: Execution Time in decryption.

Total Execution Time in milliseconds					
File Name	**File Size (KB)**	**Traditional RC6 algorithm**	**Proposed RC6 algorithm**	**proposed RC6 algorithm version (1)**	**proposed RC6 algorithm version (2)**
Iraq.txt	48	7.76	7.10	1.14	0.38
R.txt	72.46	5.51	4.65	7.68	7.27
Iraq.bmp	16.6	6.30	4.78	4.72	4.24
date.jpg	218	85.69	64	61.68	49.67
ali.jpg	93.2	64.91	45.30	42.47	32.92
lana.jpg	172	81.42	56.32	41.91	42.71
canser.png	41.8	20.88	14.55	14.55	15.54
soner.jpg	24.4	7.82	5.58	5.42	4.11
Baby.jpg	70.7	25.71	17.36	17.22	17.75
Total	757.16	310.44	219.64	207.73	174.59
Throughput decryption		2.4389	3.4472	3.6449	4.3367

8.3 Throughput encryption and decryption analysis

The throughput was computed by dividing the total execution time by the total number of files that were used.

The throughput of the traditional RC6 and the proposed RC6, proposed RC6 version 1, and proposed RC6 version 2 algorithms is shown in Figures 5 and 6, respectively, for encryption and decryption. This demonstrates that the proposed RC6 outperforms the standard RC6 in terms of throughput. This would be due to the use of six registers and a 64-bit word size per register. This tends to mean that the proposed RC6 has 384 bits per iteration, compared to 128 bits per iteration in the traditional.

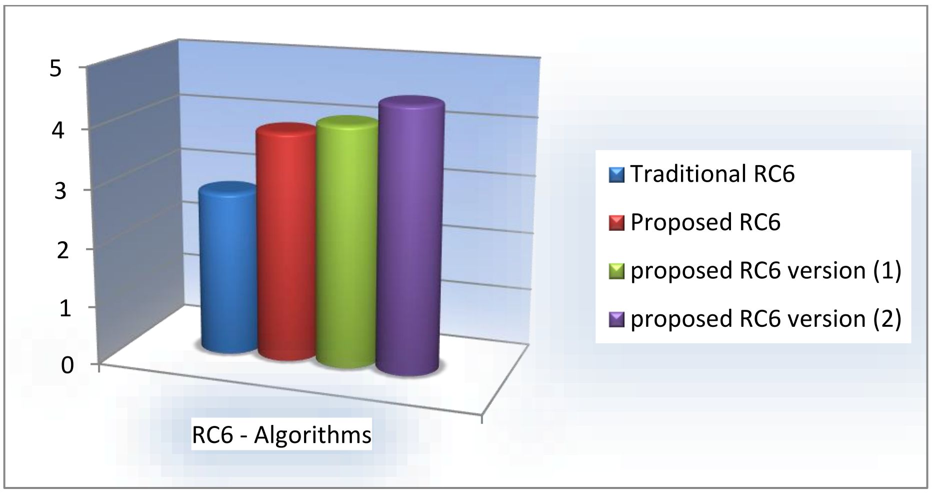

Figure 5: Throughput on Execution Time for encryption of the algorithms

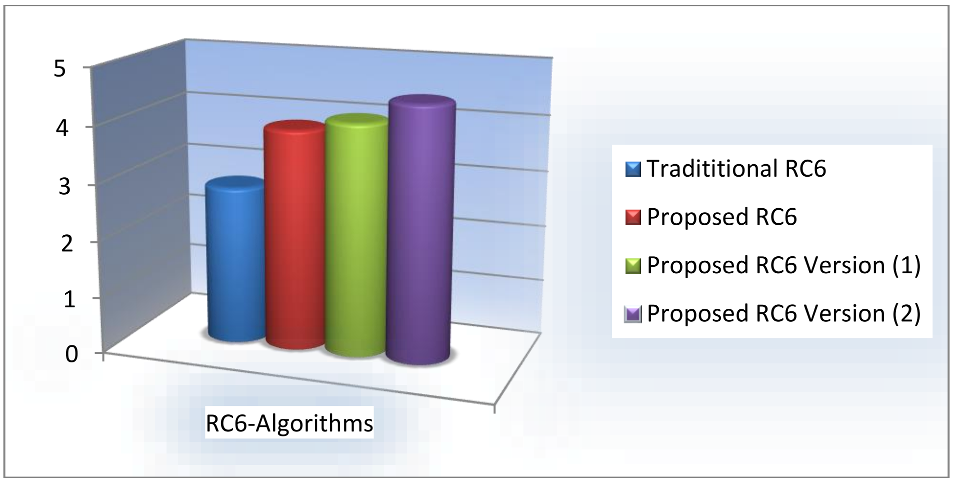

Figure 6: Throughput on Execution Time for decryption of the algorithms

8.4 Blocks-based analysis

Table 6 illustrates the number of blocks needed by each algorithm to process a given file size. This is achieved by converting kilobytes to bytes, then to bits, and then dividing the result by the block length to determine the required number of blocks. The proposed RC6 algorithm contains 384 bits, while the standard RC6 algorithm contains only 128 bits. Therefore, we conclude that the proposed algorithm requires fewer blocks to finish the given file because it uses six registers, and the word size is 64 bits per register.

Table 6: The number of blocks based on file size.

File Name	File Size (KB)	Traditional RC6 Register size(32-bit) Key Size(128-bit)	Proposed RC6 Register size(64-bit) Key Size(384-bit)
Iraq.txt	48	3072	1024
R.txt	72.46	4637	1545
Iraq.bmp	16.6	1062	354
date.jpg	218	13952	4650
ali.jpg	93.2	5965	1988
lana.jpg	172	11	4
canser.png	41.8	3	1
soner.jpg	24.4	2	1
Baby.jpg	70.7	5	2

8.5 Key Search Analysis

To test the randomness of the key generated by the PLL algorithm of the proposed RC6 algorithm so that it cannot be guessed and accessed by using two tests to display the key space analysis versus brute-force attacks.

8.5.1 Exhaustive Search Analysis

As the key increases, the number of keys stored in the array also increases. In the traditional algorithm, a 128-bit key will store 4 keys by dividing the key length by the register length (32); 1024 will have 32 keys stored; a 3500-bit key will have 110 keys stored; and a 6000-bit key will have 188 keys stored. In the proposed algorithm, a 128-bit key stores 2 keys by dividing the key length by the register length (64), resulting in 16 keys for 1024, 55 keys for a 3500-bit key, and 94 keys for a 6000-bit key, as Table 7 illustrates. The result is determined by testing the length stored in the array and the resulting key. The strength associated with the encryption depends on the key length. Although the encryption methods currently in use are believed to be secure, as technology advances, they will eventually become vulnerable to brute force attacks if given enough time and computing power. Because the strength of an algorithm's encryption depends directly on the size of the key, the choice of key in encryption is of paramount importance.

Table 7: The result of Increasing Key Size

the size of key	Traditional RC6 Algorithm with Key Generator 44	proposed RC6 Algorithm with Key Generator 66
Bits	Length of array	Length of array
128	4	2
1024	32	16
3500	110	55
6000	188	94

Table 8 presents the number of years required to break a specific key size on a computer running the Python programming language on the Windows 10 operating system, equipped with an Intel (R), Core (TM) i7-CPU 2.71 GHz, and 16.0 GB of RAM. A 128-bit key would take 3.59676 years to break the key. The computation was done by adopting the formula for exhaustive key search.

$$\textit{Breaking for Years} = \frac{2^{128}}{(3\times10^{9})\times60\times60\times24\times365} = 3.59676\times10^{21}$$

Table 8: Exhaustive Key Search.

The Size of Key (Bits)	Breaking for Years
128	3.59676×10^{21}
1024	$1.191548749\times10^{291}$
3500	$3.715094286\times10^{1034}$
6000	$4.797808976\times10^{1784}$

8.5.2 Key Sensitivity Analysis

The key to block ciphers is the Avalanche Criteria, which defines how well a minor variation in input bits tends to rise to a large (i.e., avalanche) variance in the output value. This criterion, with an optimum system of 0.50, is a desirable aspect of block cipher systems because it aligns with the mathematical conclusion of dispersion. When designing a block cipher, one must consider the avalanche that occurs when a single change in an odd input bit results in a completely different outcome. In this paper, we will encrypt the same plain text using the first key (rana), then encrypt it again using the different key (lana), thereby enhancing the system's resistance to brute force attacks. We will calculate the avalanche effect of the cipher text for both keys using the equation below.

$$AC = \frac{\text{NO. of Flip. Bits in (output)Cipher Text}}{\text{NO. of every Bits in (output) Cipher Text}} \qquad (8)\ [24]$$

Key1: Rana
Plain Text: hello my name is Ali
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 1 0 1 0 0 0
1 1 1 0 1 0 1 0 1 1 0 0 1 1 1 0 0 1 1 0 1 1 0 0 0 1 0 0 1 0 0 0 1 1 1 0 0 1 1 1 0 0 0 0 1 1 0 0 1 1
1 0 1 1 0 0 0 1 0 0 1 0 1 1 0 0 0 0 1 1 0 1 0 1 1 0 1 1 0 0 1 1 1 0 1 1 0 0 0 1 0 0 1 1 0 1 1 0 1 0
1 0 0 0 1 1 0 0 1 1 1 0 1 0 0 0 1 0 1 1 0 1 0 0 1 0 1 0 0 1 1 0 1 0 0 0 0 1 0 0 1 1 1 0 0 0 0 0 1 1
0 0 0 1 1 0 1 1 1 1 0 0 1 0 1 0 0 0 0 1 0 0 1 1 0 1 0 0 0 0 1 1 0 0 0 1 0 0 1 0 0 0 1 1 1 0 1 1 0 1
0 1 1 0 0 1 1 1 1 0 1
Cipher text1:
0 0 1 1 1 1 0 1 0 1 0 1 1 0 1 0 0 1 1 0 1 1 1 0 1 0 1 0 0 1 0 1 0 1 0 0 1 0 0 1 1 0 0 1 0 1 0 1 0 1
0 1 0 1 1 0 1 0 1 1 1 1 0 0 1 0 1 0 0 1 0 0 1 0 1 1 0 1 1 0 0 1 1 1 1 0 1 1 0 1 1 0 0 0 0 0 1 0 1 0
0 0 1 1 0 1 1 1 0 1 1 1 1 0 1 0 0 0 1 0 1 0 0 0 1 0 1 0 0 1 0 1 1 0 0 0 0 1 0 1 1 1 0 0 1 0 1 0 1 0
0 1 1 0 0 1 1 0 0 0 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 1 1 0 0 1 0 0 0 0 0 0 1 0 1 1 1 1 0 1 0 1 1 0 0 0
1 0 1 0 1 0 0 1 0 0 1 0 1 1 0 1 1 0 0 1 0 1 1 1 0 1 1 0 0 0 1 0 0 0 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0
0 0 1 1 0 1

The avalanche effect percentage is 52.14% due to the bit length being 257 bits and the number of splintered bits being 134 bits.

Key2: Lana
Plain Text: hello my name is Ali
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 1 0 1 0 0
0 1 1 1 0 1 0 1 0 1 1 0 0 1 1 1 0 0 1 1 0 1 1 0 0 0 1 0 0 1 0 0 0 1 1 1 0 0 1 1 1 0 0 0 0 1 1 0 0 1
1 1 0 1 1 0 0 0 1 0 0 1 0 1 1 0 0 0 0 1 1 0 1 0 1 1 0 1 1 0 0 1 1 1 0 1 1 0 0 0 1 0 0 1 1 0 1 1 0 1
0 1 0 0 0 1 1 0 0 1 1 1 0 1 0 0 0 1 0 1 1 0 1 0 0 1 0 1 0 0 1 1 0 1 0 0 0 0 1 0 0 1 1 1 0 0 0 0 0 1
1 0 0 0 1 1 0 1 1 1 1 0 0 1 0 1 0 0 0 0 1 0 0 1 1 0 1 0 0 0 0 1 1 0 0 0 1 0 0 1 0 0 0 1 1 1 0 1 1 0
1 0 1 1 0 0 1 1 1 1 0 1
Cipher text2:
0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 0 0 1 1 1 1 1 1 1 0 0 0 1 0 0 1 0 1 0 0 1 1 1 1 1 1 1 1 1 0 1 1 0 1
0 0 1 0 0 0 1 0 1 1 1 0 1 0 0 1 0 1 0 0 1 1 0 1 1 0 1 1 1 1 1 1 0 0 1 1 1 1 0 0 0 1 1 1 1 0 1 1 0 1
1 1 1 0 0 1 1 0 1 1 1 1 1 0 0 0 0 0 1 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 1 0 1 1 1 0 0 0 0 1 1 0 1 0 0 1
1 0 1 0 0 0 0 1 1 1 0 0 0 1 1 1 1 1 0 1 1 1 1 1 1 0 0 1 0 0 0 0 1 1 1 1 0 0 0 1 0 1 0 1 1 1 1 1 1 0
1 0 0 1 1 0 1 1 1 1 1 0 0 0 0 0 0 1 0 1 1 0 0 1 1 1 0 0 0 0 1 1 0 1 0 1 1 0 0 1 0 1 1 1 1 1 1 1 1 1
1 1 1 1 0 0

The avalanche effect percentage is 47.85% due to the bit length being 257 bits and the number of splintered bits being 123 bits.

8.6 Results with Statistical Tests

The NIST benchmark tests are a set of standardized tests used to ensure that keys used in encryption contain a high level of randomness. Keys are considered random if they pass the NIST benchmark tests for use in security applications. Table 9 shows the randomness of the keys used in text encryption for the proposed algorithm and compares the results with the two versions and the traditional algorithm.

Table 9: NIST for Generated key

No. Test	NIST test	Traditional RC6 algorithm	Proposed RC6 algorithm	proposed RC6 algorithm version (1)	proposed RC6 algorithm version (2)
1	Monobit-test(frequency)	0.032203	0.203022	0.561227	0.911413
2	frequency-within-block-Test	0.314042	0.678686	0.574903	0.678686
3	Runs-Test	0.498093	0.514124	0.015598	0.883171
4	longest-run-ones-in-block-Test	0.487863	0.602458	0.631249	0.961593
5	binary-matrix-rank-test	0.746572	0.955835	0.630178	0.394870
6	DFT –Test	0.113151	0.477568	0.540878	0.678686
7	non-overlapping-template-matching-Test	0.271167	0.345115	0.448190	0.840081
8	overlapping-template-matching-test	0.015570	0.262249	0.413032	0.616305
9	maurer-universal-test	0.547637	0.986515	0.249284	0.994031
10	linear-complexity-Test	0.245072	0.483309	0.706149	0.981767
11	Serial-Test	0.500713	0.753185	0.70614	0.989179
12	approximate-entropy-Test	0.772760	0.739918	0.726503	0.495514
13	cumulative-sums-Test	0.003996	0.198690	0.020548	0.520670
14	random-excursion-Test	0.745791	0.231703	0.772760	0.599390
15	random-Excursion-variant-Test	0.374495	0.904349	0.683283	0.891010

8.7 Evolution of Entropy

The proposed RC algorithm has been applied to several images of different sizes and properties where the information seeped into the proposed encryption algorithm is minimal and secure against entropy-based attacks. Table 10 compares the entropy values for the original plain images and the traditional RC6 algorithm with the entropy values for proposed algorithms.

Table 10: Comparing the Proposed RC6 algorithm with other methods for different images

Images	Entropy for original image	traditional RC6 algorithm	Proposed RC6 algorithm	Proposed RC6 algorithm version (1)	Proposed RC6 algorithm version (2)
Iraq.bmp	3.50625	7.27426	7.64912	7.58678	7.99965
date.jpg	7.76857	7.99949	7.99950	7.99967	7.99968
Ali.jpg	7.74611	7.99878	7.99782	7.99643	7.99968
lana.jpg	7.80539	7.99973	7.99904	7.99981	7.99994
Caner.png	6.95013	7.99472	7.97517	7.97877	7.99829
soner.png	6.82693	7.96732	7.98457	7.97823	7.99843
Baby.png	7.10884	7.90851	7.95947	7.95795	7.99861

8.8 Correlation Coefficient (CC)

The computed correlation coefficients of the original, which have different dimensions and extensions, are (jpg, bmp, png), and the encrypted images show that in the original images, two neighboring pixels have been highly correlated with each other, whereas the correlation coefficients regarding the encrypted images are very near 0, as shown in Figure 7. As a result, the proposed method is immune to statistical attacks. Tables 11 compare correlation results in encryption images for the traditional RC6 algorithm, the proposed RC6 algorithm, v1, and v2 for different images.

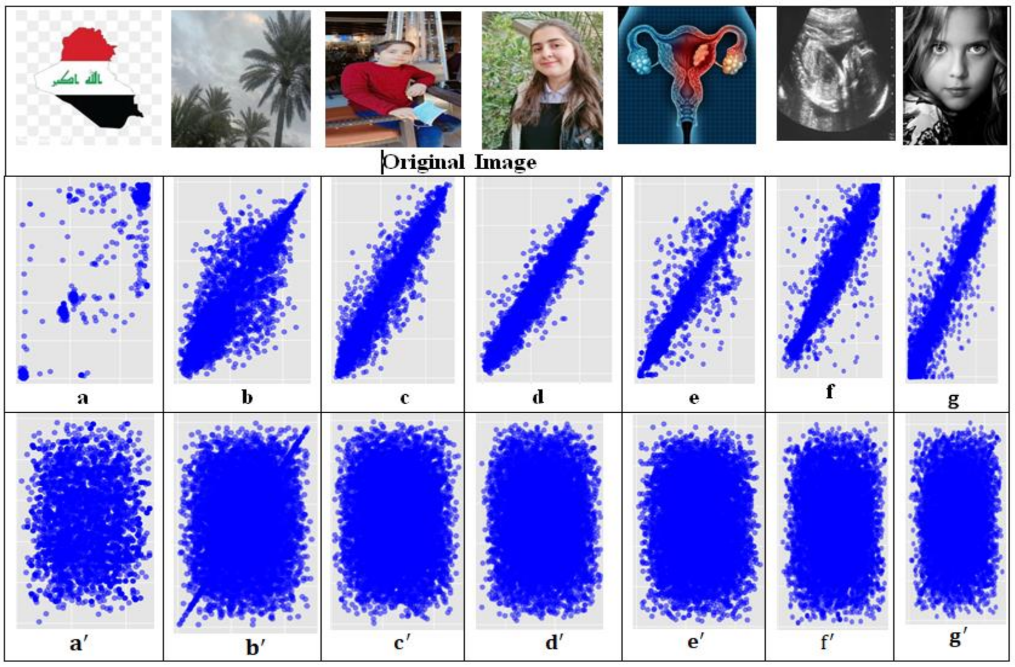

Figure 7: Correlation of original and encrypted images, original image correlation (a) and Image correlation using the proposed RC6 algorithm (a′).

Table 11: Comparing the Correlation Analysis with Proposed RC6 algorithm for different images.

Image	Direction Image-ENC	traditional RC6 algorithm	Proposed RC6	proposed RC6 algorithm version (1)	proposed RC6 algorithm version (2)
Iraq.bmp	Horizontal	-0.0465688	-0.0694198	0.00603166	0.01088491
	vertical	0.0048526	0.0369139	0.04327495	0.05846015
	diagonal	-0.0458235	-0.0341372	-0.03479745	0.00055017
date.jpg	Horizontal	0.2056030	0.2531631	0.19084655	0.00875239
	vertical	0.0191903	-0.0166073	0.02301082	0.05553587
	diagonal	0.0011878	-0.0117086	0.00695352	0.0023189
Ali.jpg	Horizontal	0.0021303	0.0160774	0.03582225	0.00467151
	vertical	0.0048065	0.0042734	0.04417402	0.00773119
	diagonal	-0.0202007	0.0008176	0.01727751	0.00392531
lana.jpg	Horizontal	0.0238797	0.1539607	0.11910758	0.01752741
	vertical	0.0075168	0.0125777	0.02007712	0.01606352
	diagonal	-0.0057829	0.0166172	0.00583789	0.01869405
Caner.png	Horizontal	-0.0040229	-0.0209094	-0.00907428	-0.00335369
	vertical	-0.0076748	0.0054232	-0.01770325	0.04784932
	diagonal	-0.0217218	0.0326412	-0.00059578	0.01935661
Sonar.png	Horizontal	0.013104	-0.0088993	-0.02067631	-0.03276173
	vertical	0.0034850	-0.0052981	0.13388575	0.01695662
	diagonal	0.0729497	0.0152348	-0.00675208	0.00418184
Baby.png	Horizontal	0.1899465	0.1991687	0.26204721	0.01007902
	vertical	0.0345551	-0.0245777	0.01414364	0.06345495
	diagonal	0.0391232	-0.0270387	0.02478969	-0.00537095

8.9 Peak Signal to Noise Ratio (PSNR) Analysis

The PSNR equation is applied after calculating the mean square error according to the equations. Table 12 presents a list of computed values for several test images, demonstrating that the values are sufficiently low to hinder attackers from decrypting the encrypted image and retrieving the original plain image. When applying the RC6 algorithm to encrypt Ali's image and get the results, it was noticed that the MSE between encrypted image and original image of traditional RC6 is (98.95841), while in the proposed RC6 it is (97.43519), and in the proposed RC6 version 1 it is (99.55742), and finally, the highest result in the proposed RC6 version 2 is (105.62179). According to table 4.20, the best PSNR result for the proposed RC6 version 2 is 27.89326, which is also the lowest. In terms of decryption, all the proposed algorithms obtained zero results in MSE, which means that the original image was retrieved after encryption. In addition to the results of the PSNR, an infinite value was obtained. The result of division by zero according to equation (6) in Section 5.3. The proposed RC6 algorithm version (2) is highly effective in both encryption and decryption, ensuring optimal results.

Table 12: The analysis results of MSE and PSNR in the Encryption and decryption algorithm for traditional RC6 algorithm, the proposed RC6 algorithm, v1, and v2 for different images.

Image	State	traditional RC6 algorithm	Proposed RC6	proposed RC6 algorithm version (1)	proposed RC6 algorithm version (2)
Iraq.bmp	MSE of Original and ENC	98.95841	97.43519	99.55742	105.62179
	PSNR of Original and ENC	28.17627	28.24364	28.15006	27.89326
	MSE of Original and DEC	0	0	0	0
	PSNR of Original and DEC	infinity	infinity	infinity	infinity
date.jpg	MSE of Original and ENC	104.0345	104.02570	110.68433	104.91955
	PSNR of Original and ENC	27.95887	27.95939	27.68994	27.92223
	MSE of Original and DEC	0	0	0	0
	PSNR of Original and DEC	infinity	infinity	infinity	infinity
Ali.jpg	MSE of Original and ENC	105.501314	104.03848	101.88306	104.53473
	PSNR of Original and ENC	27.89822	27.95886	28.04978	27.93819
	MSE of Original and DEC	0	0	0	0
	PSNR of Original and DEC	infinity	infinity	infinity	infinity
lana.jpg	MSE of Original and ENC	105.59947	110.20262	105.70085	105.88092
	PSNR of Original and ENC	27.89418	27.70888	27.89001	27.44043
	MSE of Original and DEC	0	0	0	0
	PSNR of Original and DEC	infinity	infinity	infinity	infinity

9. Conclusion

This paper aims to increase security by minimizing the likelihood of different attacks occurring during real-time data transmission, as well as reducing the time required for the cryptographic algorithm and maintaining data secrecy. The proposed RC6 outperforms the traditional RC6 in terms of throughput for both encryption and decryption. When using block size 128, as in traditional RC6, compared to the developed RC6 algorithm, block size 384 is bigger, and there will be fewer iterations and faster encryption for data. This is what we aspire to achieve in the 4G network. A measurement of the entropy, correlation coefficient, and PSNR are used to demonstrate the resistance to differential analysis. Therefore, the suggested RC6 algorithm is robust, resistant to cryptographic attacks, and provides effective confusion. The proposed RC6 algorithm was implemented with text and image data, where the images were of different dimensions. Also, the images were taken in different ways, some of which were taken using iPhone mobile phone filters and some of the Huawei mobile. It was concluded that the pictures taken with the filter are faster in the encryption process because they do not contain noise or the lack of noise in the picture, as in Ali's image. This speeds up the transfer process. Lastly, increasing the key length had only a small impact on the algorithm's speed through encryption and decryption. Future work can explore other algorithms like AES, Twofish, and Serpent, analyzing their security, speed, resource efficiency, and their application in 4G networks. 5G networks can also utilize the proposed RC6 algorithm. Future studies could design and implement a mobile application that encrypts user data between sender and receiver, utilizing the proposed algorithms to send not only images and messages but also videos and voice calls.

Conflicts of Interest: The authors declare no conflicts of interest.

References

[1] H. B. A. Wahab, S. M. Kadhem, E. A. R. J. E. Kadhim, and T. Journal, "Proposed approach for key generation based on elliptic curve (EC) algebra and metaheuristic algorithms," *Engineering and Technology Journal*, vol. 32, no. 2,p. 14, 2014.

[2] R. Mishra and P. Bhanodiya, "A review on steganography and cryptography," in *2015 International Conference on Advances in Computer Engineering and Applications*, pp. 119-122: IEEE,2015.

[3] Zaed S Mahdi, Rana M Zaki, Laith Alzubaidi, "A Secure and Adaptive Framework for Enhancing Intrusion Detection in IoT Networks Using Incremental Learning and Blockchain," *Security and Privacy,* Vol 8, No.4, pp: e70071, 15 July 2025.

[4] R. J. I. J. o. S. Ismail, "A Secure Session Management Based on Threat Modeling," vol. 54, no. 4Appendix, pp. 1176-1182, 2013.

[5] G. M. J. I. W. C. Koien, "An introduction to access security in UMTS," *IEEE Wireless Communications*, vol. 11, no. 1, pp. 8-18, 2004, DOI: 10.1109/MWC.2004.1269712 .

[6] M. Madani and C. Tanougast, "Combined and robust SNOW-ZUC algorithm based on chaotic system," International Conference on *Cyber Security and Protection of Digital Services(CyberSecurity),IEEEXplore*,Glasgow,UK,1-12June2018,pp.1-7.DOI: 10.1109/CyberSecPODS.2018.8560677.

[7] R. M. Zaki and H. B. A. J. I. J. o. I. M. T. Wahab, "4G Network Security Algorithms: Overview," *InternationalJournalofInteractiveMobileTechnologies*,vol.15,no.16,p.127,2021. DOI:10.3991/ijim.v15i16.24175.

[8] E. A. Mohammed, N. F. Areed, A. Takieldeen, and M. J. I. J. o. C. A. Abd-elazeem, "Novel cryptographic algorithm for 4G/LTE-A," *International Journal of Computer Applications (0975 – 8887)*, vol. 163, no. 1,p. 8887, April 2017.

[9] A. Subandi, M. S. Lydia, and R. W. Sembiring, "Analysis of RC6-Lite Implementation for Data Encryption," in *Proceedings of the 3rd International Conference of Computer, Environment, Agriculture, Social Science, Health Science, Engineering and Technology*, (ICEST 2018), pp 42-47, 2021, DOI: 10.5220/001003750042004.

[10] O. S. Faragallah *et al.*, "Efficiently encrypting color images with few details based on RC6 and different operation modes for cybersecurity applications," vol. 8, pp. 103200-103218, 2020.

[11] A. Berisha, H. J. J. o. C. S. Kastrati, and T. Studies, "Parallel Implementation of RC6 Algorithm," vol. 3, no. 2, pp. 01-09, 2021.

[12] X. Ji, Y. Chen, W. Yang, and Q. J. R. i. P. Wu, "Security and data encryption effect of high ciphertext based on improved RC6 algorithm for WSN," vol. 53, p. 106959, 2023.

[13] F.-H. Hsiao and S.-W. J. I. A. Chang, "Integrating RC6 Stream Cipher to A Chaotic Synchronization System," 2024.

[14] O. S. Faragallah *et al.*, "Improved RC6 block cipher based on data dependent rotations," vol. 70, no. 1, pp. 1921-1934, 2022.

[15] RM Zaki, IS Naser, "Hybrid classifier for detecting zero-day attacks on IoT networks," *Mesopotamian Journal of Cybersecurity*, vol. 4, no. 3, pp. 59-74, 2024,

[16] E. O. Agu, M. O. Ogar, and A. O. J. I. J. o. A. R. i. C. S. Okwori, "Formation of an improved RC6 (IRC6) cryptographic algorithm," *International Journal of Advanced Research in ComputerScience*,vol.10,no.4,July–August2019,DOI: http://dx.doi.org/10.26483/ijarcs.v10i4.6458.

[17] K. Aggarwal, "Comparison of RC6, modified RC6 & enhancement of RC6," in International Conference on *Advances in Computer Engineering and Applications: IEEE Xplore*, Ghaziabad, India, 19-20 March 2015,pp. 444-449, DOI: 10.1109/ICACEA.2015.7164746.

[18] R. M. Zaki, H. B. A. J. P. o. E. Wahab, and N. Sciences, "A novel of substitution-box design using PLL algorithms in magic cube," *Periodicals of Engineering and Natural Sciences*, vol. 9, no. 4, pp. 674-684, 2021, DOI: http://dx.doi.org/10.21533/pen.v9i4.2402.

[19] Rana M Zaki, Hala Bahjat Abdul Wahab, "A novel of substitution-box design using PLL algorithms in magic cube," *Periodicals of Engineering and Natural Sciences (PEN)*, Vol. 9, No. 4, pp: 674-684, 2021,

[20] G. Hanchinamani and L. J. I. J. o. H. I. T. Kulakarni, "Image encryption based on 2-D Zaslavskii chaotic map and pseudo Hadmard transform," vol. 7, no. 4, pp. 185-200, 2014.

[21] E. Ashraf, N. F. Areed, and A. J. I. J. o. C. A. Takieldeen, "Novel Cryptographic Algorithm for 4G/LTE-A," vol. 975, p. 8887, 2017.

[22] R. M. Zaki, T. W. Khairi, and A. E. Ali, "Secure data sharing based on linear congruetial method in cloud computing," Conference Next Generation of Internet of Things: Proceedings of *ICNG IoT, Singapore*, 15 June 2021, vol. 201,pp. 129-140, https://doi.org/10.1007/978-981-16-0666-3_13.

[23] R. Guesmi, M. A. B. Farah, A. Kachouri, and M. Samet, "A novel design of Chaos based S-Boxes using genetic algorithm techniques," in *2014 IEEE/ACS 11th International Conference on Computer Systems and Applications (AICCSA)*, 2014, pp. 678-684: IEEE.

[24] R. M. Zaki and H. B. A. J. i. Wahab, "A Novel SNOW3G-M Algorithm and Medical," *InternationalJournalofOnline&BiomedicalEngineering*,Vol.18,Issue3,p134,2022m DOI:10.3991/ijoe.v18i03.28013**.**