ZANCO Journal of Pure and Applied Sciences

TYPE Original Research
DOI http://dx.doi.org/10.21271/ZJPAS.36.5.8

OPEN ACCESS

*Corresponding author
Newroz Nooralddin Abdulrazaq
newroz.abudlrazaq@su.edu.krd

RECEIVED :06 /03/ 2024
ACCEPTED :05/08/ 2024
PUBLISHED :31/ 10/ 2024

KEYWORDS:
Cryptography, Encryption, Decryption, Symmetric Key, AES Cryptosystem, Playfair Cipher, S-Box.

Generating of A Dynamic and Secure S-Box for AES Block Cipher System Based on Modified Hexadecimal Playfair Cipher

Newroz Nooralddin Abdulrazaq[1]

[1]Department of Computer Science and Information Technology, College of Science, Salahaddin University-Erbil, Erbil, Kurdistan Region, Iraq.

ABSTRACT

In today's digital world, the extensive use of devices, including smartphones, tablets, IoT devices, and the internet, emphasizes the necessity for strong security measures. These measures are essential to safeguard both user data and sensitive government information. As technology advances, the enhancement of cryptographic methods becomes imperative, ensuring alignment with this rapid progression. This paper introduces a novel approach to encrypting classified data using the modified AES cipher system. It employs a dynamic and secure substitution byte operation with a 4×4 matrix, replacing the static and public 16×16 S-box found in the original version of AES. To construct the presented S-box, the system makes a secret key of 56 bytes (4×14 Rounds), where each round of the AES-256 utilizes 4 bytes to generate a 4×4 Hexadecimal Playfair matrix. By altering the S-boxes in each round and block (where each block utilizes AES key expansion to generate a 56-byte secret key for the Playfair matrix), it becomes feasible to produce distinct encrypted blocks even when the original blocks remain identical. In terms of security, the effectiveness of the provided method has been verified by experimental and analytical studies including the Avalanche effect, Balanced output, Hamming distance, and Time complexity. The result shows that the Avalanche effect of the proposed method exceeds 50% and increase the time of brute force attack. Moreover, the proposed algorithm exhibits impressive execution speed, handling approximately 200 KB in just one second.

1.Introduction

1.1AES Cryptosystem

The Advanced Encryption Standard (AES), initially known as Rijndael, is a method for encrypting and decrypting electronic data that is utilized in both hardware and software systems due to it consider as a simplest and fastest algorithm among the existing techniques. The AES cryptosystem based on a symmetric-key principle, which means the same key is used for both the encrypting\decrypting the classified messages. AES includes a range of different key block sizes: 128, 192, and 256 bits, all with the same block size (128 bits) for encryption/decryption process (Stamp, 2021). AES algorithm functions by operating on data blocks and is grounded in the concept of finite fields. It has gained acceptance by the U.S. government, replacing the Data Encryption Standard (DES) that was introduced in 1977. AES is incorporated in the ISO/IEC 18033-3 standard and was officially recognized as a U.S. federal government standard on May 26, 2002, following endorsement by the U.S. Secretary of Commerce. AES encryption\decryption process is integrated into numerous security systems and is unique in being the sole cipher that is publicly available and sanctioned by the U.S. National Security Agency (NSA) for encipher highly confidential data, given that it is employed within a cryptographic module that has received NSA approval (Stamp, 2021). Advanced Encryption standard technique is an iterative cipher, which means that it carries out a series of interlinked operations on the classified data. The operations include byte substitution, shift rows, mix columns, and add round key respectively, byte substitution is an operation includes single-byte substitution which is rearranged with 16 by 16 matrix in hexadecimal format, the shift rows operation includes a left permutation that is row-wise, the mix columns operation is a mixing process that is column-wise. and finally, the add round key operation includes the addition of the round key. The quantity of processing rounds is contingent upon the length of the key: a 128-bit key necessitates 10 rounds, a 192-bit key requires 12 rounds, and a 256-bit key demands 14 rounds. All rounds of processing are the same, with an exception of the final round which excludes the mix column operation (Stamp, 2021). Although it's faster than a complete brute-force assault, as of 2023, there have been no successful computational attacks on AES. With the biclique attack, the key for AES-128 can be obtained with a computational complexity of $2^{126.1}$. Likewise, the computational complexities for biclique attacks on AES-192 and AES-256 are 2189.7 and $2^{254.4}$, respectively (Tao & Wu, 2015).

1.2Playfair Cipher

The original Playfair cipher employs a 5×5 matrix, using 25 English letters, with ignore the letter "j" and replaced with the letter "i", and a keyword after, removing repeated letters to perform the encryption and decryption process. To encrypt the message, it should be grouped into pairs of letters, avoiding the same letter in one group by inserting an "x" between them. An "x" should also be added at the end of the message, in case there is a missing letter to be grouped. Following that, each group should be compared with the letters in the created matrix to determine the positions of letters and then start to encrypt each group by satisfying the following conditions (Marzan & Sison, 2019):

- If the two values are located in the same row, swap each letter with the one to its right, looping from the end to the start of the row.
- If the two values are located in the same column, swap each letter with the one below it, looping from the bottom to the top of the column.
- If the two values are located in different locations, replace each letter with the one that is in its row and in the column of the other value.

One of the effective attacks on Playfair cipher is the brute force attack, also known as a 'known plaintext attack,' includes decrypting an intercepted message using every possible key and comparing the result to the known plaintext. The known text is typically guessed, but it can be deduced from the fact that communication sessions often start with the same byte sequence. For a successful attack, only a small number of known bytes are needed, simplifying the guessing process. The time required to break un readable message, which is encrypted with the Playfair cipher utilizing a brute-force attack based on the key size and the computational resources available to the attacker. So, the time needed by Brute force attack can be calculated as the following (Marzan & Sison, 2019):

$$Estimmation\ Time = \frac{No.of\ charcters\ set^{keyLength}}{Encryption\ Per\ Seconds}$$

1.3Problem Statement

As previously mentioned, the original variant of Advanced Encryption Standard (AES) is commonly harnessed for encrypting and decrypting classified data in the digital world, which is utilized in both hardware and software systems due to its literary reputation as one of the simplest and fastest encryption algorithms available. Unfortunately, the original variant of the AES technique uses unsecure and fixed substitution bytes (S-box operation) during all AES rounds, which can lead to introducing vulnerabilities and, as a result, authorizes the attacker to gain the classified data. Accordingly, the objective of this work to create a substitution byte with characteristics to keep it hidden and dynamically exchanged during entire AES rounds due to enhancing the complexity of the AES technique using a modified Playfair cipher in the form of Hexadecimal, which substitutes a single byte (where each byte consists of two values of Hexadecimal, each one represents 4 bits).

2.RELATED WORKS

In 2016, Balajee and Gnanasekar provided a method to generate s-box, which is depended on scrambling Pseudo-Random Number Generators (PRNGs) by utilizing two large prime numbers and a shared secret key in order to simplify the complexity of proposed S-box. The suggested method provided better performance in terms of the Hamming Distance, Balanced Output, and Avalanche Effect, making it a viable choice for secure communication systems (Balajee & Gnanasekar, 2016).

In 2020, the authors developed a new method of construction of affine transformations based on chaos using rotational matrices to construct secure key-based S-boxes. It has been stated that under certain design conditions, the chaotic logistic maps' nonlinear trajectories produce rotational matrices, which form key-based S-boxes with cryptographic performance similar to AES S-box characteristics (Mahmood Malik, 2020). Moreover, Assafli & Hashim gave an inventive strategy for progressing the Advanced Encryption Standard (AES) cipher system by implying a time subordinate s-box which is based on the age timestamp and works independently from the encryption key. Taking after that, this strategy gives an upgraded mode of AES-ECB that depends on this unused strategy of creating energetic s-boxes based on Unix time, which offers the next security level by making a one-of-a-kind ciphertext at each execution and diminishing the hazard of encryption key spillage (Assafli & Hashim, 2020).

In 2021, a method proposed to ensure the security of classified plain data during the encryption process and its successful decryption back to its original form, which is achieved by constructing a dynamic substitution box (s-box). They provided a new idea for generating variable S-boxes based on three operations: a square polynomial transformation, a new affine transformation, and a unique permutation operation, which helps the generating of a large number of effective S-boxes by making small adjustments to the transformation and permutation parameters (Zahid, et al., 2021). within the same year, (Aisha Ejaz, 2021) introduced a method for designing key-dependent dynamic S-boxes with dynamic permutations to improve the security of symmetric block ciphers. The suggested S-box was evaluated using criteria such as bit independence, non-linearity, Hamming distance, balanced output, strict avalanche criteria, and differential and linear approximation probabilities, and it passed several NIST standard statistical tests for randomness. The method is highly sensitive to secret key changes, with a single bit change generating a completely new S-box. Future adaptations could extend this method for different key sizes (192–256 bits) or more.

In 2022, Taher et al. analyzed different 4×4, 8×8, and 16×16 substitution box evaluated based on million-bit encryption and focused on Avalanche effect in order to pick the best substitution box among them. The work suggested that the S-box of size 16x16 demonstrates superiority with a 52% Avalanche effect ratio (Taher, et al., 2022). In the same year, Ibrahim et al. provided a new method for securely store and transmit color images using chaotic system with a 16 round evolutionary DNA encoding, transposing, and substitution operation. Moreover, the encryption process of the proposed methods includes several steps, starting with the creation of round keys using a logistic function, followed by the implementation of rounds using a 16x16 nonlinear DNA Playfair matrix, transformation, and finally, substitution operation, which makes the encryption process strong, efficient, and secure. Additionally, experimental assessments demonstrate the effectiveness of the proposed method in thwarting statistical and differential attacks, with correlation coefficients (less than 0.01), NPCR (greater than 0.99), and UACI (greater than 0.33). Nonetheless, this thesis also contains the NIST analysis for testing the randomness of the encryption method (Ibrahim, et al., 2022). Moreover, Al-Dweik et al. presented an algorithm to generate key-dependent dynamic clone s-boxes that retain the same algebraic properties (bijection, nonlinearity, SAC, and BIC) as the initial s-box. The method uses group actions on the columns and rows of Boolean functions of the s-box. The invariance of these properties in the clone copies is proven. Examples are provided for initial s-boxes of sizes 4 × 4 and 8 × 8, showing significant potential for generating numerous clone copies with preserved properties. This method extends previous works that involved group actions only on columns of Boolean functions (Al-Dweik, et al., 2022).

In 2023, James and Priya offered a new dynamic and nonlinear substitution box generation method, which adapts per round based on the round key, and significantly enhances communication security among Internet of Things (IoT) devices. The provided method indicates powerful nonlinearity, effective differential and linear approximation probability, and a strict avalanche effect, causing it resistant to linear and differential attacks (James & Priya, 2023). In same year, Ali et al. provided a new substitution box that combines the direct product map of cyclic groups with the inversion map of a Galois field comprising 256 elements due to encrypting digital images with the CBC mode of the AES cipher system. A new substitution boxes are then compared with existing substitution box that yield better results, which are verified by analyzing bijectivity, SAC, NL, BIC, LP, and DAP tests. Furthermore, the effectiveness of the provided digital image encryption system has been verified using the best types of standards, for instance, correlation, homogeneity, entropy, energy, NPCR, and UACI tests (Ali, et al., 2023).

In 2024, Razaq et al. introduced a new approach to satisfy the security demands of e-Healthcare systems by generating substitution boxes (S-boxes) from membership values of a fuzzy subset of integers ranging from 1 to 256. The effectiveness of the provided substitution box is guaranteed via examinations indicating the robustness of the system against diverse attack. Using the recommended substitution box a robust encryption method for images was created and tested using standard analyzations, for image encryption (Razaq, et al., 2024). Later In the year 2024, researchers employed LFT (linear fractional transformation) and developed a new technique to generate S-boxes by using the multilayer perceptron architecture. Algorithm suggested herein employs a three-layered perceptron network (input layer, hidden neurons, output neurons) with weights proportional to the S-box structure's dimension. The resulting S-boxes hence have visible near-optimal average nonlinearity values that better previous known ones; one in particular has a nonlinearity of 114. 50. This can be illustrated with an image encryption application which is enhanced through the existence of the S-box (Adil Waheed, 2024).

3.METHODOLGY OF THE PROPOSED TECHNIQUE

3.1Playfair Encryption and Decryption process

The modified Playfair cipher deals with a 4×4 matrix of hexadecimal values, as opposed to the traditional 5×5 matrix that works with the 26 English alphabets. During the encryption process, plaintext of length 16 bytes is converted into hexadecimal values utilizing ASCII code table, whereas each byte yields two hexadecimal values, with each hexadecimal value representing four bits. The secret key used in the Playfair cipher is also converted into hexadecimal form. Any duplicate hexadecimal values from the key are then removed. A 4x4 matrix is formed, filled with the remaining unique hexadecimal values from the key. Any unfilled cells are populated with the remaining hexadecimal values, which are not present in the secret key from 0 to F in order.

Once a 4×4 matrix is created, the encryption process begins. Each byte is selected for encryption. As mentioned earlier, each byte contains two hexadecimal values. If the hexadecimal values are duplicates, the encryption value remains the same as the original hexadecimal values. However, if the values are different, then to perform the substitution, apply the following three rules (as depicted in **Figure 1**):

1) If the two values are located in the same row, swap each hexadecimal with the one to its right, looping from the end to the start of the row.
2) If the two values are located in the same column, swap each hexadecimal with the one below it, looping from the bottom to the top of the column.
3) If the two values are located in different locations, replace each hexadecimal with the one that is in its row and in the column of the other value.

In addition, for decryption process, the same technique will be proceeded with difference in the first two rules:

1) If the two values are located in the same row, swap each hexadecimal with the one to its left.
2) If the two values are located in the same column, swap each hexadecimal with the one up it.

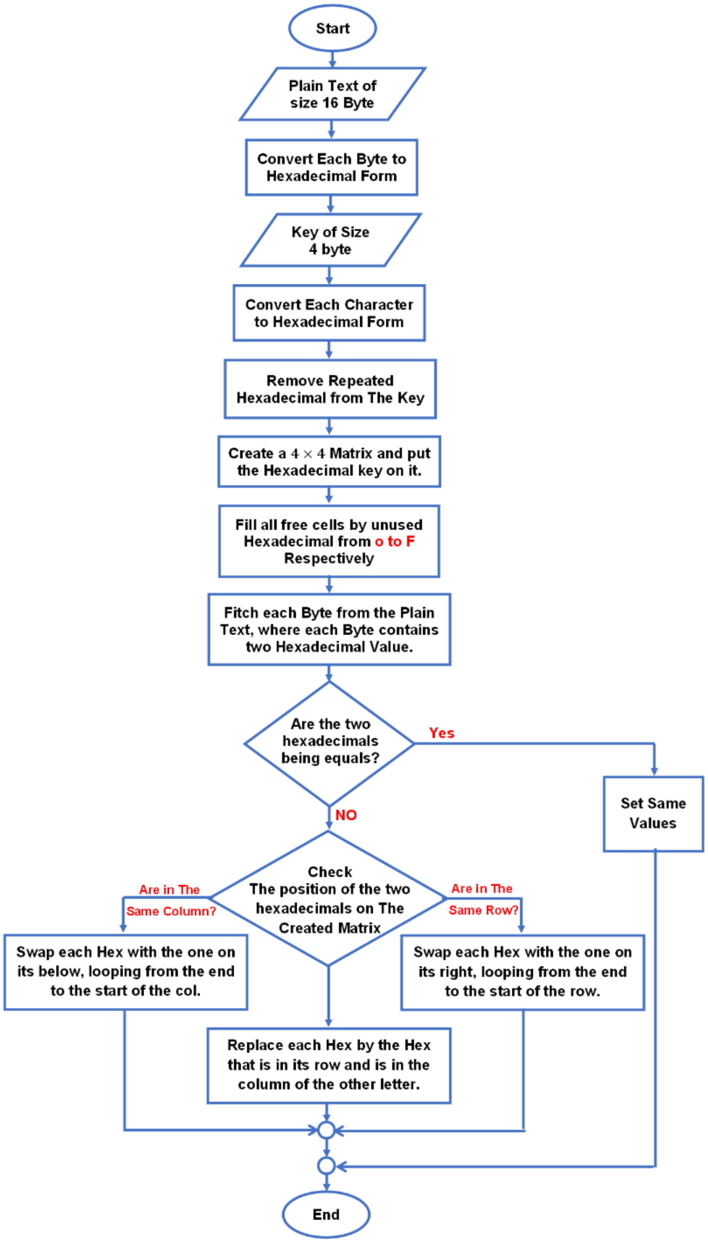

Start
Plain Text of size 16 Byte
Convert Each Byte to Hexadecimal Form
Key of Size 4 byte
Convert Each Character to Hexadecimal Form
Remove Repeated Hexadecimal from The Key
Create a 4 × 4 Matrix and put the Hexadecimal key on it.
Fill all free cells by unused Hexadecimal from o to F Respectively
Fitch each Byte from the Plain Text, where each Byte contains two Hexadecimal Value.
Are the two hexadecimals being equals?
Yes
Set Same Values
NO
Check The position of the two hexadecimals on The Created Matrix
Are in The Same Column?
Are in The Same Row?
Swap each Hex with the one on its below, looping from the end to the start of the col.
Swap each Hex with the one on its right, looping from the end to the start of the row.
Replace each Hex by the Hex that is in its row and is in the column of the other letter.
End

Figure 1: Modified Playfair Cipher Encryption Process

3.2 Key Generation of The Proposed Technique

In the suggested method, two different secret keys are required. The first key, which should be 56 bytes (characters including spaces) long, is for the improved Playfair cipher. The second secret key is for the improved AES cryptosystem. The size of this key based on the AES types in use: 128 bits (16 bytes\characters) for AES-128, 192 bits (24 bytes\characters) for AES-192, and 256 bits (32 bytes\characters) for AES-256. Furthermore, the number of rounds generated is calculated by the length of the key. Specifically, AES-128, AES-192, and AES-256 generate 10, 12, and 14 rounds respectively. **Figure 2** shows the process of AES key generation. In each round, the key is partitioned into four words, and the last word undergoes three operations rot word, sub word and RCON operation.

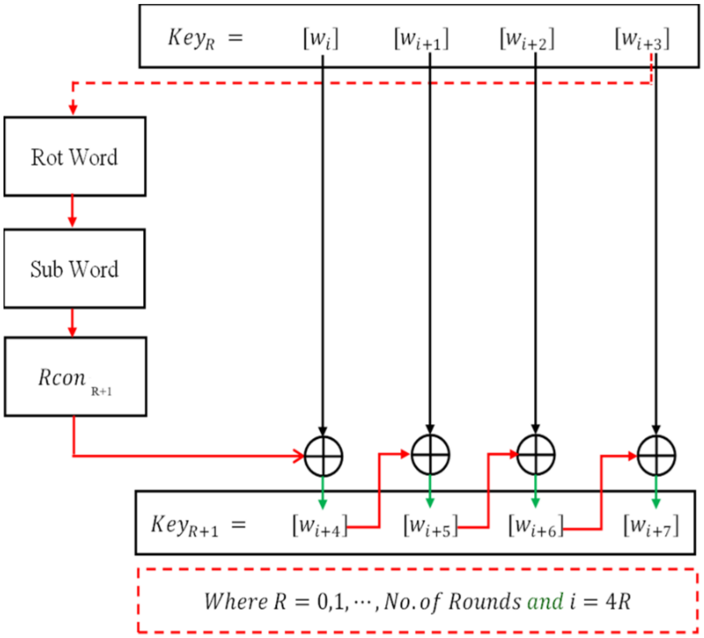

Key_R = $[w_i]$ $[w_{i+1}]$ $[w_{i+2}]$ $[w_{i+3}]$
Rot Word
Sub Word
$Rcon_{R+1}$
Key_{R+1} = $[w_{i+4}]$ $[w_{i+5}]$ $[w_{i+6}]$ $[w_{i+7}]$
Where $R = 0,1,\cdots, No. of\ Rounds$ and $i = 4R$

Figure 2: AES Key Generation

3.3Encryption Process of Proposed Technique

The encryption process in the proposed method begins by converting the plaintext into byte format and then into hexadecimal form. The plaintext is then partitioned into group of blocks, each including128 bits (16 bytes). Each cluster of 16 bytes is arranged into a 4×4 matrix, where each column consists of four bytes in hexadecimal form. Before initiating the first round of encryption, the initial state of the matrix is XORed with the zeroth round of the secret key. Following this, the process moves through the stages or rounds of AES encryption. Each round includes executing four different operations: substituting bytes using the improved Playfair cipher, Shift Row, Mix Column, and Add Round Key. These operations are performed in each round, except for the final round. The final round of the AES encryption process is unique compared to the previous rounds, as it omits the Mix Column operation (as shown in **Figure 3**). Finally, the encryption procedure for the improved method is completed by calculating all rounds (14 rounds for AES-256) for each segment of the partitioned plaintext.

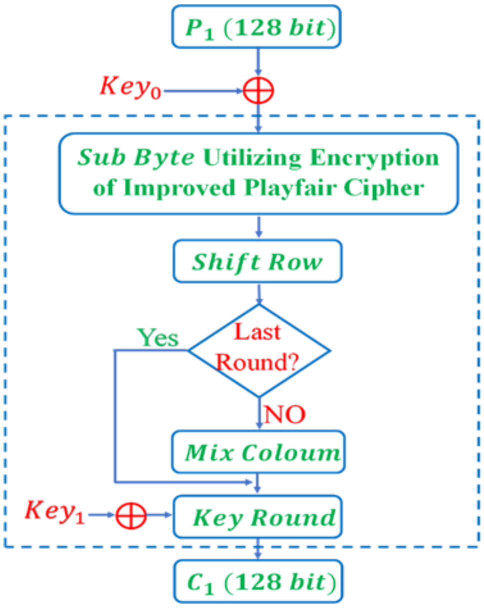


Figure 3: A Part of Encryption for Proposed Technique

3.4Decryption Process of the Proposed Technique

When the receiver obtains the encrypted message, the decryption process begins. The received ciphertext is then partitioned into separate cipher blocks, each block includes 16 bytes bits long (128 bits). Each block is then rearranged into a matrix of size 4x4, which consists of 16 bytes. This crucial arrangement prepares the encrypted data for the next steps. Subsequently, the first matrix is rearranged, it then proceeds through four different steps: Add Round Key, Inverse Mix Column (excluding from first round), Inverse Shift Row, and Inverse Substitution byte using the decryption procedure of the proposed method, these steps are performed sequentially for each round, except the last one. Just like in the proposed encryption process, but reversed, the first step of the AES decryption process has a unique feature. It omits the 'Inverse Mix Column' operation, setting it apart from the previous operations. This is deemed crucial throughout the entire decryption process as it ensures the accurate retrieval of the original message from the encrypted one (as depicted in **Figure 4**). After all 14 rounds are calculated in the AES-256 cipher system, a key step in decrypting the unreadable message. The decryption process ends by XORing the last key round with the final matrix, which is calculated from round 14. In the final step of the decoding process, all decoded parts are collected to retrieve the original message. This is done after removing the extra zeros bytes.

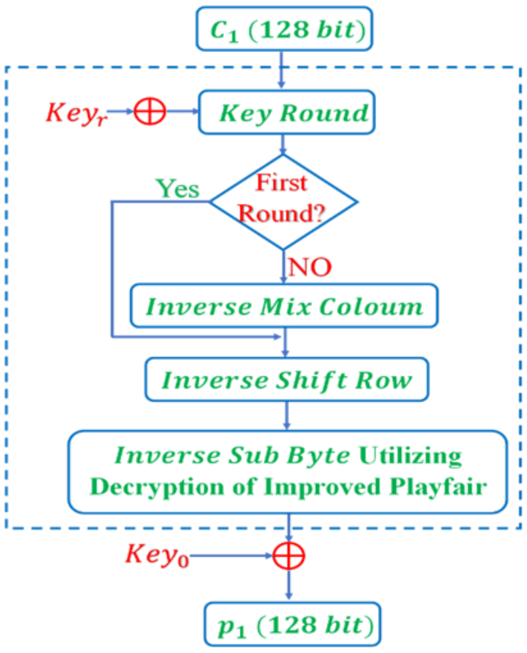


Figure 4: A Part of Decryption Process in The Proposed Technique

4.DISCUSSION OF THE PROPOSED TECHNIQUE

4.1Encryption Process of the Modified AES Cryptosystem

To provide a better understanding of the proposed method, this work discusses the first round of the encryption process. It encrypts the message "Citadel of Erbil" using the Playfair secret key "Love Your Parent", and the first two keys (round0 and round1) for the AES cryptosystem:

$$Key_0 = \begin{bmatrix} 54 & 73 & 20 & 67 \\ 68 & 20 & 4B & 20 \\ 61 & 6D & 75 & 46 \\ 74 & 79 & 6E & 75 \end{bmatrix} \text{ AND } \quad (4\text{-}1)$$

$$Key_1 = \begin{bmatrix} E0 & 91 & B1 & D6 \\ 32 & 12 & 59 & 79 \\ FC & 91 & E4 & A2 \\ F1 & 85 & E6 & 93 \end{bmatrix}$$

First of all, the plaintext: Citadel of Erbil and the Playfair secret key: Love (First 4 bytes for first round) must be covert into bytes of hexadecimal form:

C	i	t	a	d	e	L	
43	69	74	61	64	65	6C	20

o	f		e	r	b	I	l
6f	66	20	45	72	62	69	6c

L	o	v	e
4C	6F	76	65

Form a 4×4 matrix from plaintext of hexadecimal values, which each 4 bytes represent a column and the XORed with Key_0:

$$New\ state = \begin{bmatrix} 43 & 64 & 6F & 72 \\ 69 & 65 & 66 & 62 \\ 74 & 6C & 20 & 69 \\ 61 & 20 & 45 & 6C \end{bmatrix} \oplus \begin{bmatrix} 54 & 73 & 20 & 67 \\ 68 & 20 & 4B & 20 \\ 61 & 6D & 75 & 46 \\ 74 & 79 & 6E & 75 \end{bmatrix} = \begin{bmatrix} 17 & 17 & 4F & 15 \\ 01 & 45 & 2D & 42 \\ 15 & 01 & 55 & 2F \\ 15 & 59 & 2B & 19 \end{bmatrix} \quad (4\text{-}2)$$

The first round commences by carrying out the four operations of the enhanced method. The initial operation involves computing the modified Playfair cipher as a substitute for the substitution box. The first step is to eliminate any repeated hexadecimal values (in this case, the result will be as follows: 4C6F75) and then arrange them in a 4×4 matrix. Any unfilled cells in the matrix should be populated with hexadecimal values ranging from 0 to F. All hexadecimal values should place in the matrix:

$$Playfair\ Matrix\ = \begin{bmatrix} 4 & C & 6 & F \\ 7 & 5 & 0 & 1 \\ 2 & 3 & 8 & 9 \\ A & B & D & E \end{bmatrix} \qquad (4\text{-}3)$$

Now, after constructing the Playfair matrix, the results in the equation (4-2) should be substituted using the aforementioned rules in the improved Playfair cipher:

$$New\ state = \begin{bmatrix} 75 & 75 & C4 & 70 \\ 17 & C7 & 8A & 7A \\ 70 & 17 & 55 & 94 \\ 70 & 13 & 3A & 9E \end{bmatrix} \qquad (4\text{-}4)$$

For instance, to substitute cell (2,2) from equation (4-2), which is (45) utilizing Playfair matrix in equation (4-3). The hexadecimal 4 and 5 are located in different location, so by applying rule3 in the encryption process, the result become C7 and so on.

The second operation is the Shift Row, which is applied to the equation (4-4). For the encryption process, a left shift is applied as follows: no shift for the first row, a 1-row shift for the second row, a 2-row shift for the third row, and finally, a 3-row shift for the last row. The result is as follows:

$$New\ state = \begin{bmatrix} 75 & 75 & C4 & 70 \\ C7 & 8A & 7A & 17 \\ 55 & 94 & 70 & 17 \\ 9E & 70 & 13 & 3A \end{bmatrix} \qquad (4\text{-}5)$$

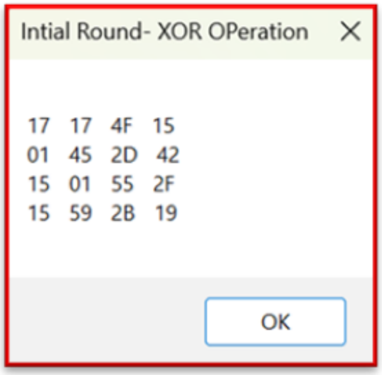
The Mixed column operation, which is the AES encryption procedure, is calculated by multiplying the fixed matrix with matrix in the equation (4-5):

$$New\ state = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \times \begin{bmatrix} 75 & 75 & C4 & 70 \\ C7 & 8A & 7A & 17 \\ 55 & 94 & 70 & 17 \\ 9E & 70 & 13 & 3A \end{bmatrix} \qquad (4\text{-}6)$$
$$= \begin{bmatrix} 73 & 8B & 7E & F4 \\ 81 & AD & B3 & 5D \\ A1 & 5C & 6B & 07 \\ 2A & 61 & 7B & E4 \end{bmatrix}$$

Finally, the outcome in the equation (4-6) is XORed with the key of round1 in the equation (4-1), which is known as adding key operation as mentioned in equation (4-7).

The outcome in equation (4-7) represents the final result of the first round in the encryption process. To obtain the final result of the encryption, it's necessary to carry out the remaining 13 rounds, following the AES-256 key block size as depicted in **Figure 5** and **Figure 6**.

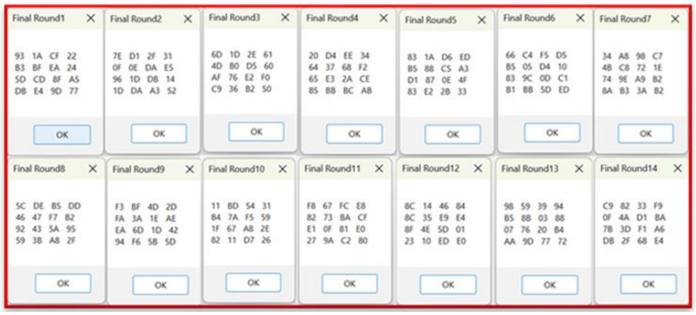
$$New\ state = \begin{bmatrix} 73 & 8B & 7E & F4 \\ 81 & AD & B3 & 5D \\ A1 & 5C & 6B & 07 \\ 2A & 61 & 7B & E4 \end{bmatrix} \oplus \begin{bmatrix} E0 & 91 & B1 & D6 \\ 32 & 12 & 59 & 79 \\ FC & 91 & E4 & A2 \\ F1 & 85 & E6 & 93 \end{bmatrix} \qquad (4\text{-}7)$$
$$= \begin{bmatrix} 93 & 1A & CF & 22 \\ B3 & BF & EA & 24 \\ 5D & CD & 8F & A5 \\ DB & E4 & 9D & 77 \end{bmatrix}$$



Intial Round- XOR OPeration

```
17 17 4F 15
01 45 2D 42
15 01 55 2F
15 59 2B 19
```

Figure 5: Result of Initial Round for Modified AES Encryption Process



Final Round1

```
93 1A CF 22
B3 BF EA 24
5D CD 8F A5
DB E4 9D 77
```

Final Round2

```
7E D1 2F 31
0F 0E DA E5
96 1D DB 14
1D DA A3 52
```

Final Round3

```
6D 1D 2E 61
4D B0 D5 60
AF 76 E2 F0
C9 36 B2 50
```

Final Round4

```
20 D4 EE 34
64 37 68 F2
65 E3 2A CE
85 B8 BC AB
```

Final Round5

```
83 1A D6 ED
B5 88 C5 A3
D1 87 0E 4F
83 E2 2B 33
```

Final Round6

```
66 C4 F5 D5
B5 05 D4 10
83 9C 0D C1
B1 B8 5D ED
```

Final Round7

```
34 A8 98 C7
4B C8 72 1E
74 9E A9 B2
8A B3 3A B2
```

Final Round8

```
5C DE B5 DD
46 47 F7 B2
92 43 5A 95
59 3B A8 2F
```

Final Round9

```
F3 BF 4D 2D
FA 3A 1E AE
EA 6D 1D 42
94 F6 5B 5D
```

Final Round10

```
11 BD 54 31
B4 7A F5 59
1F 67 A8 2E
82 11 D7 26
```

Final Round11

```
F8 67 FC E8
82 73 BA CF
E1 0F 81 E0
27 9A C2 80
```

Final Round12

```
8C 14 46 84
8C 35 E9 E4
8F 4E 5D 01
23 10 ED E0
```

Final Round13

```
98 59 39 94
B5 8B 03 88
07 76 20 B4
AA 9D 77 72
```

Final Round14

```
C9 82 33 F9
0F 4A D1 BA
7B 3D F1 A6
DB 2F 68 E4
```

Figure 6: Results of Round 1-14 Modified AES Encryption Process

4.2Decryption Process of the Modified AES Cryptosystem

In the decryption process of the modified AES cryptosystem, the operations are performed in reverse order compared to the encryption process. This means that the first operation in the decryption process would be the addition of the round key. Therefore, the new state is obtained by XORing the state matrix (as given in equation (4-7)) with the first key (as mentioned in equation (4-1)).

$$New\ state = \begin{bmatrix} 93 & 1A & CF & 22 \\ B3 & BF & EA & 24 \\ 5D & CD & 8F & A5 \\ DB & E4 & 9D & 77 \end{bmatrix} \oplus \begin{bmatrix} E0 & 91 & B1 & D6 \\ 32 & 12 & 59 & 79 \\ FC & 91 & E4 & A2 \\ F1 & 85 & E6 & 93 \end{bmatrix} \quad (4\text{-}8)$$
$$= \begin{bmatrix} 73 & 8B & 7E & F4 \\ 81 & AD & B3 & 5D \\ A1 & 5C & 6B & 07 \\ 2A & 61 & 7B & E4 \end{bmatrix}$$

Subsequently, the state in Equation (4-8) performs the inverse mix column operation. In contrast to the encryption process, the inverse mix column operation should be excluded in the first round of the decryption process.

$$New\ state = \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \times \begin{bmatrix} 73 & 8B & 7E & F4 \\ 81 & AD & B3 & 5D \\ A1 & 5C & 6B & 07 \\ 2A & 61 & 7B & E4 \end{bmatrix} \quad (4\text{-}9)$$
$$= \begin{bmatrix} 75 & 75 & C4 & 70 \\ C7 & 8A & 7A & 17 \\ 55 & 94 & 70 & 17 \\ 9E & 70 & 13 & 3A \end{bmatrix}$$

The inverse shift row operation, which performs a right shift on the last obtained state in Equation (4-9), is applied as follows for the decryption process: no shift for the first row, a 1-row shift to the right for the second row, a 2-row shift to the right for the third row, and finally, a 3-row shift to the right for the last row.

$$New\ State = \begin{bmatrix} 75 & 75 & C4 & 70 \\ 17 & C7 & 8A & 7A \\ 70 & 17 & 55 & 94 \\ 70 & 13 & 3A & 94 \end{bmatrix} \quad (4\text{-}10)$$

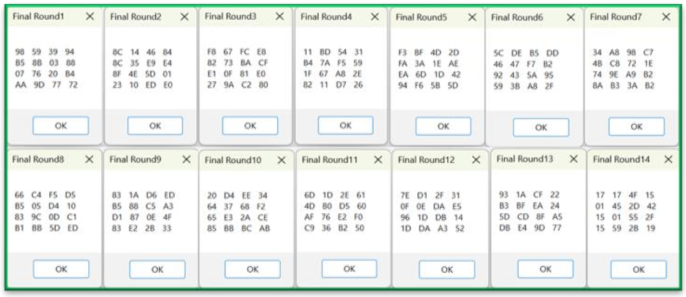
The last operation performs a byte substitution on Equation (4-10) using the decryption process of the modified Playfair cipher (as depicted in Equation (4-3)). For instance, the substitution cell (3,2), which is 17, becomes 01. This is because the hexadecimal values 1 and 7 are positioned in the same row (see Equation (4-3)), so Rule 1 should be applied to it as previously mentioned. So, the new state becomes:

$$New\ State = \begin{bmatrix} 17 & 17 & 4F & 15 \\ 01 & 45 & 2D & 42 \\ 15 & 01 & 55 & 2F \\ 15 & 59 & 2B & 19 \end{bmatrix} \quad (4\text{-}11)$$

After completing all the rounds, the result from Equation (4-11) should be XORed with the initial key, denoted as key0, as given in Equation (4-1):

$$Final\ state = \begin{bmatrix} 17 & 17 & 4F & 15 \\ 01 & 45 & 2D & 42 \\ 15 & 01 & 55 & 2F \\ 15 & 59 & 2B & 19 \end{bmatrix} \oplus \begin{bmatrix} 54 & 73 & 20 & 67 \\ 68 & 20 & 4B & 20 \\ 61 & 6D & 75 & 46 \\ 74 & 79 & 6E & 75 \end{bmatrix} \quad (4\text{-}12)$$
$$= \begin{bmatrix} 43 & 64 & 6F & 72 \\ 69 & 65 & 66 & 62 \\ 74 & 6C & 20 & 69 \\ 61 & 20 & 45 & 6C \end{bmatrix}$$

Finally, each byte from the final state in Equation (4- 12) should be converted into an ASCII character, column by column, to retrieve the original plaintext: “Citadel of Erbil”. This step is part of the AES decryption procedure, where the byte output from the previous operations is converted back into readable form. Each byte represents an ASCII character, and by fetching these bytes column by column from the final state of the matrix, the original plaintext message can be constructed as shown in **Figure 7** and **Figure 8**.



```
Final Round1
98 59 39 94
B5 88 03 88
07 76 20 B4
AA 9D 77 72
```

```
Final Round2
8C 14 46 84
8C 35 E9 E4
8F 4E 5D 01
23 10 ED E0
```

```
Final Round3
F8 67 FC E8
82 73 BA CF
E1 0F 81 E0
27 9A C2 80
```

```
Final Round4
11 BD 54 31
B4 7A F5 59
1F 67 A8 2E
82 11 D7 26
```

```
Final Round5
F3 BF 4D 2D
FA 3A 1E AE
EA 6D 1D 42
94 F6 5B 5D
```

```
Final Round6
5C DE B5 DD
46 47 F7 B2
92 43 5A 95
59 3B A8 2F
```

```
Final Round7
34 A8 98 C7
4B C8 72 1E
74 9E A9 B2
8A B3 3A B2
```

```
Final Round8
66 C4 F5 D5
B5 05 D4 10
83 9C 0D C1
B1 BB 5D ED
```

```
Final Round9
83 1A D6 ED
B5 88 C5 A3
D1 87 0E 4F
83 E2 2B 33
```

```
Final Round10
20 D4 EE 34
64 37 68 F2
65 E3 2A CE
85 BB BC AB
```

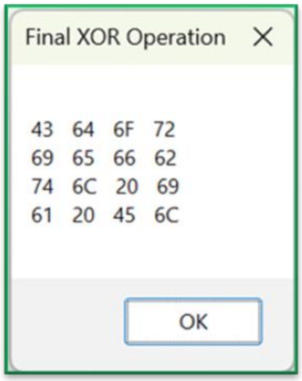
```
Final Round11
6D 1D 2E 61
4D B0 D5 60
AF 76 E2 F0
C9 36 B2 50
```

```
Final Round12
7E D1 2F 31
0F 0E DA E5
96 1D DB 14
1D DA A3 52
```

```
Final Round13
93 1A CF 22
B3 BF EA 24
5D CD 8F A5
DB E4 9D 77
```

```
Final Round14
17 17 4F 15
01 45 2D 42
15 01 55 2F
15 59 2B 19
```

Figure 7: Results of Round1-14 for Modified AES Decryption Process



```
43  64  6F  72
69  65  66  62
74  6C  20  69
61  20  45  6C
```

Figure 8: Result of Final Round for Modified AES Decryption Process

5.Results

In this section, the work analyzes the strength of the proposed method based on several analysis tests such as Hamming distance, balanced output, Avalanche effect, and time complexity.

5.1Hamming Distance

The Hamming distance is a measure used to determine how different two strings of equal length are. It does this by counting the number of positions where the corresponding symbols are not the same. In the context of the proposed work, the Hamming distance is computed for the proposed dynamic substitution box (Hexadecimal Playfair cipher) and is then compared with related works, which is mentioned in (Balajee & Gnanasekar, 2016). The Results show that the proposed method is more stable than the other suggested works. However, the proposed work has less Hamming code in only some related proposed works as depicted in **Table 1** and **Figure 9**. These illustrate the number of differing positions between the plaintext and ciphertext utilizing random plaintexts.

Table 1:Hamming Distance For S-Box

Plaintext	Pi	LCG	Balajee Alg.1	Balajee Alg.1	Proposed Work
RJ45JACK	26	35	32	37	33
FUNCTION	28	29	37	34	29
FRAGMENT	25	33	34	34	32
ZXQRASPK	28	31	37	27	33
INVERTER	28	32	37	36	32
ELECTRON	31	31	38	40	32
Average %	43%	50%	56%	56%	50%

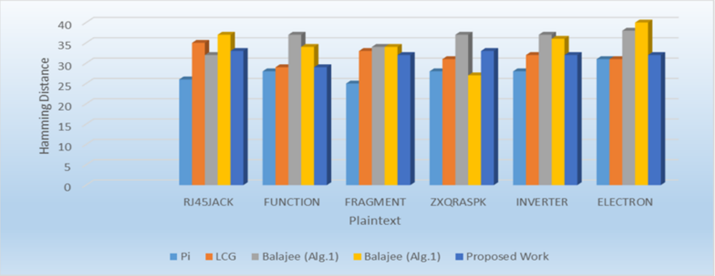


Figure 9: Hamming Distance for Different S-Boxes

5.2Balanced Outcomes

A balanced output refers to an output that contains approximately the same number of ones and zeros. In the context of the proposed work, the balanced outcome is computed for the proposed dynamic substitution box (Hexadecimal Playfair cipher) and then is compared with related works, which is mentioned in (Balajee Maram K, J M Gnanasekar, 2016). The Results show that the proposed method is more balanced than the other proposed works as depicted in **Table 2** and **Figure 10**. These visualizations depict the number of zeros in the ciphertext for the given plaintexts.

Table 2:Balanced Output for S-Box

Plaintext	Pi	LCG	Balajee Alg.1	Balajee Alg.1	Proposed Work	Pi	LCG	Balajee Alg.1	Balajee Alg.1	Proposed Work
	Number of Zero's					Number of One's				
RJ45JACK	37	30	32	27	33	27	34	32	37	31
FUNCTION	29	32	27	30	31	39	32	37	34	33
FRAGMENT	31	31	30	30	34	33	33	34	34	30
ZXQRASPK	29	30	27	37	34	35	34	37	27	30
INVERTER	32	26	27	28	33	32	38	37	36	31
ELECTRON	36	34	26	24	33	28	30	38	40	31

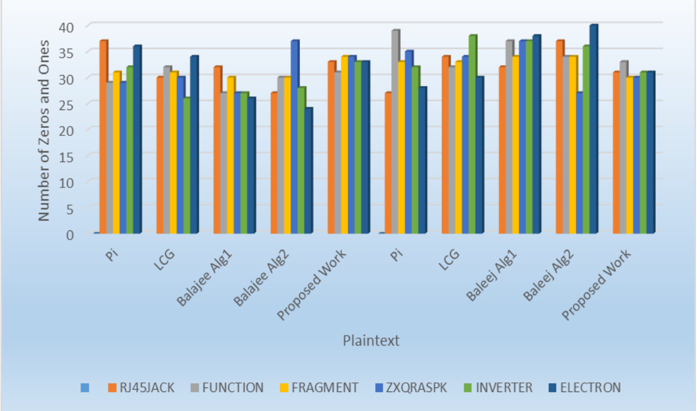


Figure 10: Balanced Output for S-Sox

5.3 Avalanche Effect

The Strict Avalanche Criterion (SAC) posits that a minuscule alteration, specifically a single bit modification in the plaintext, should exert influence on more than 50% of the bits constituting the ciphertext. Similarly, an alter in one bit of the secret key should also influence more than 50% of the bits in the ciphertext. The proposed work is analyzed to calculate the Avalanche effect on the proposed S-box only (Modified Hexadecimal Playfair cipher of Size 4 by 4) by implementing 1000 iterations of randomly generated bytes and then compared with existing works, as mentioned in Table 4 (Taher, et al., 2022). The Avalanche effect has also been calculated on the entire process of the modified AES with a block size of 128. The outcome of the calculated Avalanche effect on the proposal depicts that the proposed S-Box is approximately as effective as the other suggested works (as depicted in **Table 3** and **Figure 11**).

Table 3: Percentage of Avalanche Effect On 4 ×4 S-Box

S-Box	Percentage of Avalanche Effect
Bogdanov et al.	47%
Li et al.	38%
Shawkat et al.	30%
Suzaki *et al.*	51%
Banik et al.	42%
Proposed Work	43%

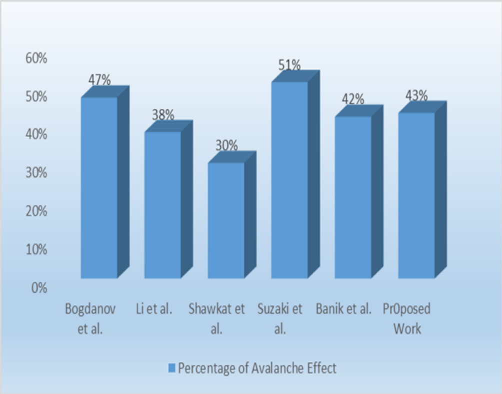
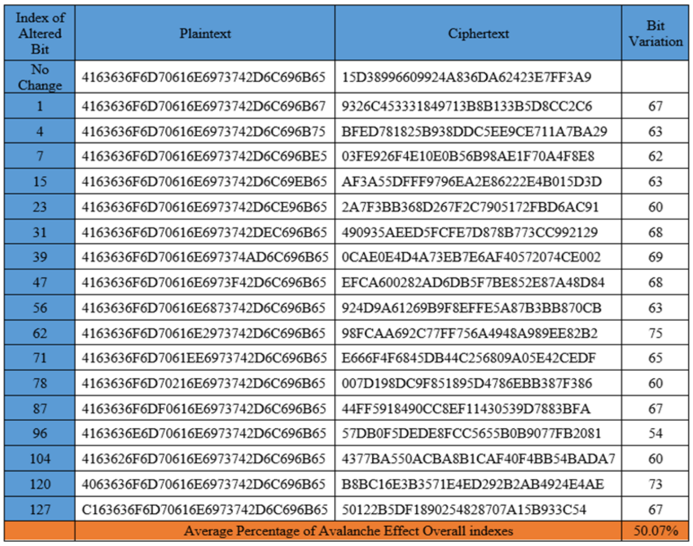


Figure 11: Percentage of Avalanche Effective On S-Box

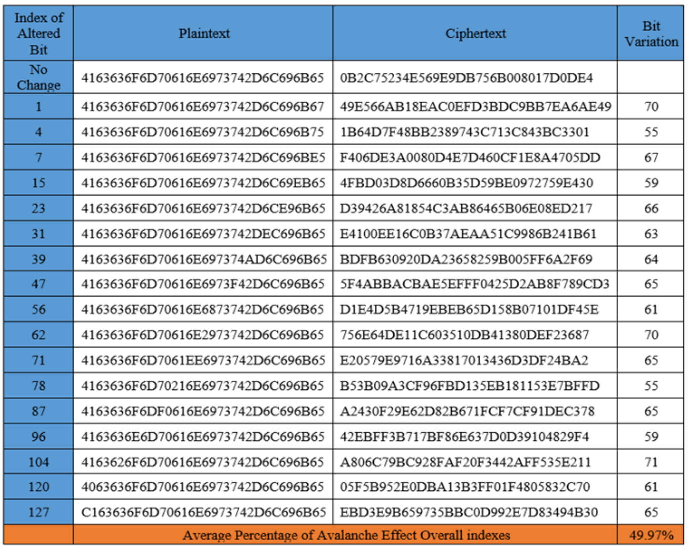
These visualizations demonstrate the Avalanche effect between the ciphertext and plaintext, achieved by iteratively swapping one bit at various positions in the binary plaintext. Whereas the Avalanche effect analyzer on the entire process of the Modified AES encryption process, compared to the original AES, shows that the average percentage of the modified AES method is approximately close to that of the original AES cryptosystem, as depicted in **Table 4** and **Table 5**. However, the outcomes from analyzing several plaintexts of size 128 bit (16 byte) for the proposed work show that the Avalanche effect lies within the interval of 43% and 52%.

Table 4: Sample of Avalanche Effective on Entire Process of Modified AES



Index of Altered Bit	Plaintext	Ciphertext	Bit Variation
No Change	4163636F6D70616E6973742D6C696B65	15D38996609924A836DA62423E7FF3A9	
1	4163636F6D70616E6973742D6C696B67	9326C453331849713B8B133B5D8CC2C6	67
4	4163636F6D70616E6973742D6C696B75	BFED781825B938DDC5EE9CE711A7BA29	63
7	4163636F6D70616E6973742D6C696BE5	03FE926F4E10E0B56B98AE1F70A4F8E8	62
15	4163636F6D70616E6973742D6C69EB65	AF3A55DFFF9796EA2E86222E4B015D3D	63
23	4163636F6D70616E6973742D6CE96B65	2A7F3BB368D267F2C7905172FBD6AC91	60
31	4163636F6D70616E6973742DEC696B65	490935AEED5FCFE7D878B773CC992129	68
39	4163636F6D70616E697374AD6C696B65	0CAE0E4D4A73EB7E6AF40572074CE002	69
47	4163636F6D70616E6973F42D6C696B65	EFCA600282AD6DB5F7BE852E87A48D84	68
56	4163636F6D70616E6873742D6C696B65	924D9A61269B9F8EFFE5A87B3BB870CB	63
62	4163636F6D70616E2973742D6C696B65	98FCAA692C77FF756A4948A989EE82B2	75
71	4163636F6D7061EE6973742D6C696B65	E666F4F6845DB44C256809A05E42CEDF	65
78	4163636F6D70216E6973742D6C696B65	007D198DC9F851895D4786EBB387F386	60
87	4163636F6DF0616E6973742D6C696B65	44FF5918490CC8EF11430539D7883BFA	67
96	4163636E6D70616E6973742D6C696B65	57DB0F5DEDE8FCC5655B0B9077FB2081	54
104	4163626F6D70616E6973742D6C696B65	4377BA550ACBA8B1CAF40F4BB54BADA7	60
120	4063636F6D70616E6973742D6C696B65	B8BC16E3B3571E4ED292B2AB4924E4AE	73
127	C163636F6D70616E6973742D6C696B65	50122B5DF1890254828707A15B933C54	67
	Average Percentage of Avalanche Effect Overall indexes		50.07%

Table 5:Sample of Avalanche Effective on Entire Process of Original AES

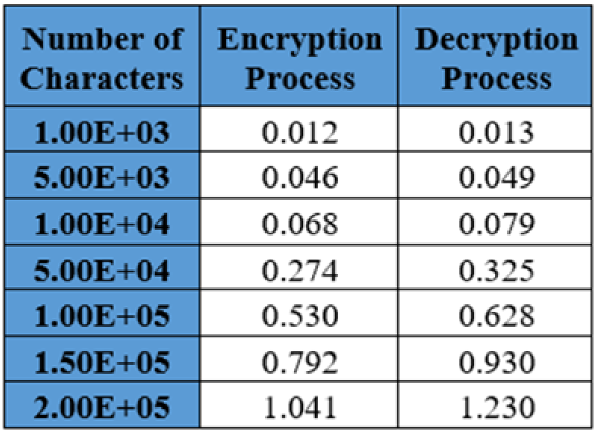


Index of Altered Bit	Plaintext	Ciphertext	Bit Variation
No Change	4163636F6D70616E6973742D6C696B65	0B2C75234E569E9DB756B008017D0DE4	
1	4163636F6D70616E6973742D6C696B67	49E566AB18EAC0EFD3BDC9BB7EA6AE49	70
4	4163636F6D70616E6973742D6C696B75	1B64D7F48BB2389743C713C843BC3301	55
7	4163636F6D70616E6973742D6C696BE5	F406DE3A0080D4E7D460CF1E8A4705DD	67
15	4163636F6D70616E6973742D6C69EB65	4FBD03D8D6660B35D59BE0972759E430	59
23	4163636F6D70616E6973742D6CE96B65	D39426A81854C3AB86465B06E08ED217	66
31	4163636F6D70616E6973742DEC696B65	E4100EE16C0B37AEAA51C9986B241B61	63
39	4163636F6D70616E697374AD6C696B65	BDFB630920DA23658259B005FF6A2F69	64
47	4163636F6D70616E6973F42D6C696B65	5F4ABBACBAE5EFFF0425D2AB8F789CD3	65
56	4163636F6D70616E6873742D6C696B65	D1E4D5B4719EBEB65D158B07101DF45E	61
62	4163636F6D70616E2973742D6C696B65	756E64DE11C603510DB41380DEF23687	70
71	4163636F6D7061EE6973742D6C696B65	E20579E9716A33817013436D3DF24BA2	65
78	4163636F6D70216E6973742D6C696B65	B53B09A3CF96FBD135EB181153E7BFFD	55
87	4163636F6DF0616E6973742D6C696B65	A2430F29E62D82B671FCF7CF91DEC378	65
96	4163636E6D70616E6973742D6C696B65	42EBFF3B717BF86E637D0D39104829F4	59
104	4163626F6D70616E6973742D6C696B65	A806C79BC928FAF20F3442AFF535E211	71
120	4063636F6D70616E6973742D6C696B65	05F5B952E0DBA13B3FF01F4805832C70	61
127	C163636F6D70616E6973742D6C696B65	EBD3E9B659735BBC0D992E7D83494B30	65
	Average Percentage of Avalanche Effect Overall indexes		49.97%

5.4Time Complexity

The execution time is a significant measurement of the efficiency of any proposed work, particularly in cryptography techniques. In the recent analysis, the time complexity of the modified AES technique performed remarkably well. For instance, when processing a block of 1×10^3 characters, the modified AES encryption process completed its in just under 0.012 seconds. As the size of the block is increasing to 2×10^5 characters, the execution time rose to approximately 1.041 seconds, which shows that with the expansion of data size, the performance of the proposed work remained steady as depicted in **Table 6** And **Figure 12**. Additionally, the time complexity for the decryption process in the modified AES requires more time compared to the encryption process. This is viewed as a favorable procedure, given that the decryption process is expected to consume more time than the encryption process.

Table 6: Time Complexity of Modified AES



Number of Characters	Encryption Process	Decryption Process
1.00E+03	0.012	0.013
5.00E+03	0.046	0.049
1.00E+04	0.068	0.079
5.00E+04	0.274	0.325
1.00E+05	0.530	0.628
1.50E+05	0.792	0.930
2.00E+05	1.041	1.230

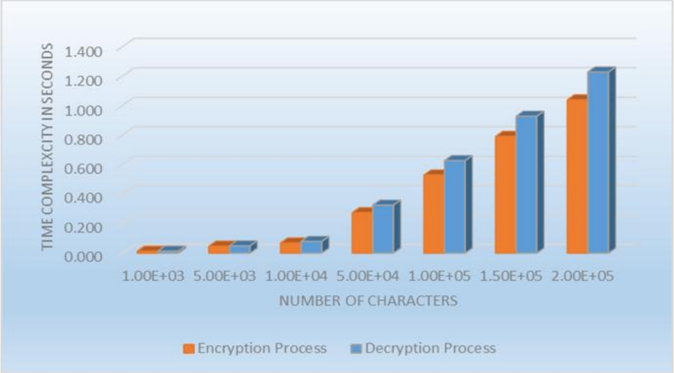


Figure 12: Time Complexity for Modified AES

5.5Brute Force Attack on Modified Playfair Cipher

A successful brute-force attack is achieved by systematically exploring all possible key combinations until the right key is identified, which is then compared to the known plaintext. The aim is to exhaust all potential combinations until a satisfactory match is found. As mentioned earlier, the original Playfair cipher needed 26! possible keys to break the message. In the proposed modified AES system, breaking the modified Playfair cipher requires more than 14× (16!) (Number of rounds and number of Hexadecimal symbols) possible keys. From another perspective, in the original Playfair cipher, the message has meaning which aids the eavesdropper in succeeding in a brute force attack. However, in the proposed method, the message only gains meaning after 14 rounds due to using Hexadecimal form instead of 26 English letters, which requires a several combinations and permutations of attempts in order to succeed in breaking the message.

Conclusions

The AES cipher system is an important method, widely used in the digital world due to its simplicity and fast implementation. It is used in both hardware and software systems, which motivates researchers to enhance and invent improved techniques to make the system more reliable for today's needs. This paper designs a method to increase the security complexity of AES through the use of a similar Playfair cipher which has been adapted. Through this, a secure and dynamic substitution byte is mounted in the solution where a static and known substitution byte operation of the traditional AES cipher system is replaced. Furthermore, in each round of the AES encryption system, the modified Playfair cipher generates a different random 4 × 4 matrix of form Hexadecimal, making the system more complex to break from intruders by using a random secret key of size 56 bytes. Along with that the resistance of the proposed encryption scheme has been examined through the metrics, such as avalanche effect, balanced outcomes, and time complexity tests. The result indicates that the proposed algorithm has an average 43%

to 52% Avalanche effect ratio and it is more balanced since the existing system. As well, the proposed algorithm demonstrates a remarkable execution time, processing 2×10^5 characters (approximately 200 KB) within the span of one second. In the future, the system should be adapted to implement a single secret key instead of two, using key expansion techniques.

Acknowledgment: I express sincere appreciation to the staff of the Department of Computer Science and Information Technology at the College of Science, Salahaddin University-Erbil, for their unwavering support.

Financial support: No financial support.

Potential conflicts of interest. There is no conflict of interest for this paper

References

Adil Waheed, F. S. M. M. S. M. M. A., 2024. Molding Robust S-Box Design Based On Linear Fractional Transformation And Multilayer Perceptron: Applications To Multimedia Security. Egyptian Informatics Journal, Volume 26.

Aisha Ejaz, I. A. S. U. I. A. R. A. K., 2021. A Secure Key Dependent Dynamic Substitution Method For Symmetric Cryptosystems. Peerj Comput. Sci, Volume 7.

Al-Dweik, A. Y., Hussain, I., Saleh, M. & Mustafa, M., 2022. A Novel Method To Generate Key-Dependent S-Boxes With Identical Algebraic Properties. Journal Of Information Security And Applications, Volume 64.

Ali, R. Et Al., 2023. A Robust S Box Design Using Cyclic Groups And Image Encryption. Ieee Access, Volume 11, P. 135880135890.

Assafli, H. T. & Hashim, I. A., 2020. Generation And Evaluation Of A New Time-Dependent Dynamic S-Box Algorithm For Aes Block Cipher Cryptosystems. S.L., S.N.

Balajee, M. & Gnanasekar, J., 2016. Evaluation Of Key Dependent S-Box Based Data Security Algorithm Using Hamming Distance And Balanced Output. Tem Journal, 5(1), Pp. 67-75.

Ibrahim, D., Ahmed, K., Abdallah, M. & Ali, A., 2022. A New Chaotic-Based Rgb Image Encryption Technique Using.

James, D. & Priya, T., 2023. An Innovative Approach For Dynamic Key Dependent S-Box To Enhance Security Of Iot Systems.. Measurement: Sensors, Volume 30.

Mahmood Malik, M. S. A. A. M. A. A. K. M. A. A. E.-U.-H. M. A. S. S. N. M. A. R. M. A. A. W., 2020. Generation Of Highly Nonlinear And Dynamic Aes Substitution-Boxes (S-Boxes) Using Chaos-Based Rotational Matrices. Ieee Access, Volume 8, Pp. 35682-35695.

Marzan, R. M. & Sison, A. M., 2019. An Enhanced Key Security Of Playfair Cipher Algorithm. Malaysia, Proceedings Of The 2019 8th International Conference On Software And Computer Applications, Pp.457-461.

Razaq, A. Et Al., 2024. Fuzzy Logic-Based Substitution-Box For Robust Medical Image Encryption In Telemedicine. Ieee Access, Volume 12, Pp. 7584-7608.

Stamp, M., 2021. Information Security: Principles And Practice. 3rd Ed. S.L.:Wiley. ISBN: 978-1-119-50588-4

Taher, H. M., Al-Rahman, S. Q. A. & Shawkat, S. A., 2022. Best S-Box Amongst Differently Sized S-Boxes Based On The Avalanche Effect In. International Journal Of Electrical And Computer Engineering (Ijece), December.Pp. 6535-6544.

Tao, B. & Wu, H., 2015. Improving The Biclique Cryptanalysis Of Aes. S.L., Springer, Pp. 39-56.

Zahid, Et Al., 2021. Dynamic S-Box Design Using A Novel Square Polynomial Transformation And Permutation. Ieee Access, Volume 9, Pp. 82390--82401. http://doi.org/10.1109/ACCESS.2021.3086717