A Framework of Multilayer Encryption Net with MBBS Generator For Images Encryption

Ghada S.mohammed* Osama basil gazi Wala'a A.Mahdi
Madenat Alelem University college
ghaa2090@mauc.edu.iq Osama87@mauc.edu.iq Walaa.a.mahdi@mauc.edu.iq

Abstract :
Ensuring the security of multimedia information, such as images, is essential due to the rapid development of information technology, so various image encryption algorithms have been proposed to ensure privacy and prevent improper behaviour. This paper presents a proposed model called Multi-Layer Encryption Net (MLE-Net) to encrypt the image based on using a multi-layer net to code the image and modify the Blum-Blum-Shub pseudo-random number generator to generate the keys used in the encryption process. The proposed method is organized into two stages: the first stage is plain-image data pre-processing, and the second stage divides the used image into blocks to be input for a multi-layer encryption network. The proposed method uses secure keys generated by the Modified Blum-Blum-Shub (MBBS) random number generator to improve its accuracy, reduce the complexity, and modify its control parameters to maintain data privacy efficiently and accurately. The outcomes show that the suggested approach successfully provides secure information delivery.
Keywords: Multi-Layer Net, Image Encryption, Random Key Generator, Blum-Blum-Shub(BBS), Information security, Pseudo-random number generator.

اطار عمل لشبكة متعددة الطبقات مع مولد**MBBS لتشفير الصور**

غادة سالم محمد اسامة باسل غازي ولاء عبد المجيد مهدي
كلية مدينة العلم الجامعة

الخلاصة :
يعتبر ضمان أمن معلومات الوسائط المتعددة مثل الصور جانبًا مهمًا جدًا نظرًا للتطور السريع لتكنولوجيا المعلومات، لذلك تم اقتراح خوارزميات مختلفة لتشفير الصور لضمان الخصوصية ومنع السلوك غير اللائق. يقدم هذا البحث نموذجا مقترحا يسمى شبكة التشفير متعددة الطبقات (MLE-Net) لتشفير الصورة اعتمادا على استخدام شبكة متعددة الطبقات لتشفير الصورة وتعديل مولد الأرقام العشوائية الزائفة Blum-Blum-Shub لتوليد المفاتيح التي تستخدم في عملية التشفير. تم تنظيم الطريقة المقترحة إلى مرحلتين، المرحلة الأولى: هي المعالجة المسبقة لبيانات الصورة العادية، والمرحلة الثانية تقسم الصورة المستخدمة إلى كتل ليتم إدخالها لشبكة تشفير متعددة الطبقات؛ تستخدم الطريقة المقترحة مفاتيح آمنة تم إنشاؤها بواسطة مولد الأرقام العشوائية المعدل Blum-Blum-Shub(MBBS) لتحسين دقة المولد وتقليل تعقيد الارتباط وتعديل معلومات التحكم الخاصة به للحفاظ على خصوصية البيانات بطريقة دقيقة وفعالة.
الكلمات المفتاحية :شبكة متعددة الطبقات ، تشفير الصورة، مولد المفاتيح العشوائية ،Blum-Blum-Shub(BBS) ، امنية المعلومات ،مولد الارقام العشوائية .

* Corresponding author : Ghada S.mohammed

1. Introductions

The widespread usage of digital information to create contact over the Internet and network is fast rising. As a result, technological developments in pictures, audio, and video have been tremendous. Information security (especially for digital images) has emerged as a critical concern in all forms of communication [1]. Therefore, many methods are used to ensure the highest security of images; one of them is encryption [2]. The purpose of image encryption is to transmit it securely over the network so that no unauthorized or unknown user can decrypt the image [3]. Cryptography can be used to solve a variety of challenges, including confidentiality, authentication, data integrity, and non-repudiation. [4] that is considered a necessary security requirement [5]. Research on digital images and their applications has also been widely applied [6]. Digital photos are comparatively less sensitive than data because any single change in the pixels of the entire image does not change it. In other words, a small amount of digital photos is acceptable compared to data, but they are more susceptible to attackers [7].

Various data encryption algorithms have been proposed and widely used, and every algorithm has advantages and disadvantages based on the techniques applied to images. Many encryption schemes for images have been developed in recent years by many researchers who proposed an encryption model for the image based on using some of the numbers generated by the 3D chaotic map as the running key for a specific interval, in which the image encryption algorithm was designed in three phases, including shuffling the image pixels and then scrambling the shuffled pixels based on mixing and masking rules [8]. Many studies depend on using cryptography techniques with artificial intelligence techniques (AI), where AI is a broad field of study that refers to the capacity of machines to replicate human intellect; the most prevalent branches are ML and DL approaches. ML algorithms undergo training on a wide range of data, and more data can enhance algorithm accuracy [9]. Some of the research related to image encryption is illustrated below:

[10] introduces a novel encryption technique for encrypting grayscale and colour images. The introduction of the image-splitting method depended on image blocks. The image's blocks were then jumbled with a zigzag pattern, rotation, and arbitrary permutation. The jumbled image is then diffused using a chaotic logistic map. The suggested approach for encrypting images is assessed in terms of security and time complexity. The results reveal acceptable security by successfully encrypting grey and colour images.

[11] introduced least-squares generative adversarial networks in the process of random number generation. A unique learning algorithm that generates random numbers is built using an excellent learning capability. Six chaotic structures with varying topologies and dimensions are used as training sets to develop speedy and efficient random numbers. The experiment results show that the encryption key produced via this approach can pass all of the National Institute of Standards and Technology's (NIST) randomness tests. The results are effectively applied to an image encryption technique based on selective scrambling and overlay diffusion, yielding satisfactory results.

[12] provides an enhanced pseudorandom sequence generator. Because the produced random sequence is used in various cryptographic applications, the mathematical approach, in this case, is intended to handle the problem of the sequences' non-uniform distribution. Furthermore, PRSG has passed the standard statistical and empirical tests, demonstrating that the proposed generator has solid statistical properties. Finally, image encryption was accomplished using the sort-index approach and diffusion processing. After a thorough study of encryption performance, there was no direct correlation between the original and encrypted images.

[13] suggests an encryption strategy based on an XOR operation between the plain picture and another image that will be used as a key agreed upon by both the sender and the recipient. To protect the security of the plain pictures, pixel permutation processes are dynamically executed on the key image and plain image using random integers in each encryption phase, making the encryption technique powerful and brute-force resistant. The results of simulations on

many standard pictures revealed a random distribution of pixel values and a greater pass rate connected with entropy and optimum values for analysis parameters.

[14] provides an encryption approach that combines logistic and sine maps to make the logistic sine map, as well as the fuzzy idea and the Hénon map, all of which are utilized to generate safe secret keys. The picture pixels are scrambled, neighbouring row values are added, and the result is XORed with numbers generated randomly from the chaotic maps. This system also has a high key sensitivity, which implies that changing the secret keys might cause considerable changes in the encrypted picture.

2. The Blum-Blum-Shub Generator (BBS)

The Blum-Blum-Shub Generator (BBS) is one of the most efficient (powerful cryptographic) pseudo-random number generators. It was created in 1986 by Lenore Blum, Manuel Blum, and Michael Shub. The BBS generator is distinguished by its reliance on number theoretic features, making it appropriate for cryptographic applications. Also, it is provably secure under the assumption that large factoring composites are intractable (integer factoring) [15]. A Blum-Blum-Shub Generator is a cryptographically secure, efficient, and robust random number generator, and it has many exciting applications, especially in the field of cryptography [17]. BBS works as follows:

- Select two sufficiently large secret different primes random numbers P and Q.
- Let N={integers N|N = P*Q, such that P,Q are equal length (|P|=|Q|)
- Distinct primes 3 mod 4} be the set of parameter values.
- Select random integer number x (seed)from the interval[1, N-1] such that gcd(x, N)=1.
- let
  $X_N =$
  $\{x^2 \ mod \ N \mid x^2 \in Z_N^*\}$ (1)
- be the quadratic residues mod N.
- Let X = disjoint $\bigcup_{N \in N} X_N$ be the seed domain.
- For I from 1 to Length perform:
- $x_i \leftarrow {x_{i-1}}^2 \ mod \ N$ (2)
- $z_i \leftarrow$
  $Least \ Signifient \ Bit(x_i)$ (3)

The asymptotic security of the (BBS) pseudo-random generator has been studied by Alexi et al. and Vazirani[15], who proved independently that O(log N) bits can be extracted on each iteration, where N is the modulus (a Blum integer) where the primary notion underlying this conclusion is that the BBS generator's least significant bit (LSB) has good unpredictability; The fundamental concept underlying these proofs is to investigate the distribution of the generator's LSB and then demonstrate that additional bits may be recovered with a success probability of O(1/log N) or better. Therefore, in practice, multiple random bits may be obtained at every iteration of the BBS generator [15]. The security of the BBS generator depends on the difficulty of factoring N. An attractive characteristic of this generator is that you can directly calculate any of the x values. Thus, applications where many keys are generated in the following equation (4):

$$y_0{}^{(2)^{i \ mod \ (p-1)(q-1)}} mod \ n \quad (4)$$

It is optional to save them all. Each key can be effectively indexed and recovered from that small index and the initial x and N [16]. So, the proposed scheme used the seed of the BBS generator as a secret key and another secret key to increase the security and efficiency of the proposed system. The BBS generator is unpredictable to the left and unexpected to the right; thus, the cryptanalysts cannot predict the next bit or the previous bit of the sequence generated by the BBS generator. Therefore, no security is based on a complicated generator that nobody understands but on the mathematics underlying the factorization of N.

The BBS generator is well-known for its high security because it is thought to be computationally impossible to anticipate the next bit without knowing the factorization of n into its prime parts p and q. That is because the generator's update function is quadratic. So, for those who want pseudo-random sequences of bits, cryptographically safe is the best choice [17, 7, 18]. Blum-Blum-Shub (BBS) is a pseudo-random number generator (PRNG) that requires a very large modulus and a squaring operation for the generation of each bit [16, 19, 20]. So, I modified this work. From each key, some of the bits were used depending on the number of specifications to avoid using

considerable numbers in the encryption of images and also for more security in generating the keys and ciphering operations.

3. The Proposed Method

The proposed method of encrypting the image information (see Figure 1) is based on a multi-layer. Thus, it is called a Multi-Layer Encryption Net (MLE Net). It comprises multiple layers to achieve its goal of encrypting the image information with high security and capacity. The first layer shifts and permutes the pixel value, thus increasing the randomness of the information, dividing the input image into blocks with prespecified sizes to input each block to the next layer that performs image information coding. The third layer generates the keys used in the encryption algorithm based on MBBS and finally performs the encryption process to produce the secure encrypted image. The action steps of the MLE-Net could be clarified as follows:

Algorithm: MLE-Net

Input: Plain image, keys.
Output: The ciphered image.

1: Call shifting and permutation algorithm.
2: Check the size of the input image [high(m), width (n)]
3: **IF** (m/3) or (n/3) not equal to zero
4: padding the input image with zeros to be m/3=0 and n/3=0
5: **End IF**
6: Divide the input image into (3X3)blocks
7: **For** each block I no. of block
8: Apply the modify median filter (3×3)
9: Perform colour splitting (each pixel value into three components (pr, pg ,pb))
10: Call coding algorithm
11: Call MBBS algorithm for each input block.
12: Cipher each input block by using the generated array (3×3) of keys.
13: **End For**
14: **End MLE Net**

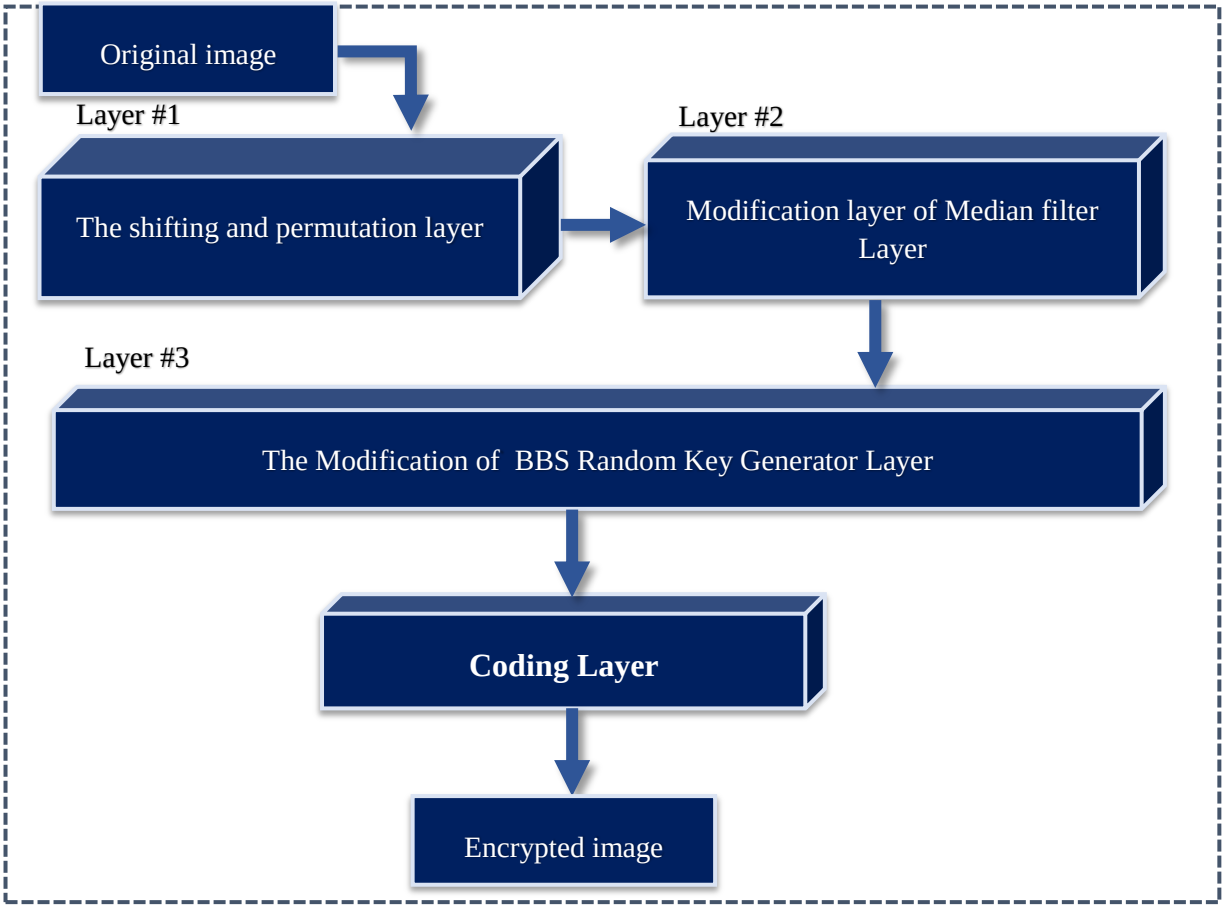

Figure 1: Encryption phase (MLE-Net)

a. The shifting and permutation layer

The "shifting and permutation layer" in cryptography often refers to a stage in a symmetric-key block cypher in which the locations of bits in a block of data are rearranged or permuted precisely. This stage is critical for creating uncertainty and diffusion in the encryption process, which makes it more resistant to cryptographic assaults.

The goal of this layer is to give cryptographic strength by providing confusion (complicating the link between plaintext and ciphertext) and diffusion (spreading the impact of each plaintext bit across several ciphertext bits). These processes make it computationally impossible for an attacker without the encryption key to reverse the encryption process. The benefits of this layer (Confusion and Diffusion, Avalanche Effect, Resistance to Known-Plaintext and Chosen-Plaintext Attacks, Data Integrity, Enhanced Security, and Versatility) add to encryption algorithms' durability and cryptographic strength, making them excellent instruments for securing sensitive information.

In this layer, the value of the pixel in the input image breaks and removes the correlation between the image pixels, so after the deciphering process, the original image must be constructed by inverse shifting and permutation, as illustrated in the following algorithm:

Algorithm: Shifting and Permutation Algorithm

Input: The original image.
Output: The shifted and permutated image

```
For each row in the length of image (m)
    If the number of rows is odd
        For each pixel in this row
            If the index of it is odd .
                Add 1 to the value of pixel
            Else
                Add 2 to the value of pixel
            End IF
        End For
    End IF
End For
IF the number of row is even then :
        For each pixel in this row
            IF the index of it is odd .
                subtract 1 from the value of pixel
            Else
                subtract 2 from the value of pixel
            End IF
        End For
End IF
For each i in the length of image (m)
    For each j in the depth of image (n)
        Replace the pixel value in the location (i<j) with the value in (i>j)
    End For
End For
End Shifting and Permutation
```

b. Modification layer of Median filter.

This layer is based on implementing the convolution process on the image, where the size of the image (rows, columns) is checked to be divided into 3x3 blocks, so if the remainder of the division, the number of rows, is not equal to zero, the image will be padded with rows of zeros to solve this issue. In the same case with the number of columns, this padding process will make splitting the image into equal blocks efficiently and simply.

All resulting blocks deal with it as follows:

- Arrange the values of pixels in the input block into ascending order if the index of the block is odd, and take the pixel value in the location specified by using equation (5) and put it in the centre of the block:

$$\frac{number\ of\ pixels\ in\ the\ block}{2} + 1 \qquad (5)$$

- Arrange the value of each block in descending order if the index of the block is even, and take the pixel value in the location specified by using the equation (6) and put it in the centre of the block:

$$\frac{number\ of\ pixels\ in\ the\ block}{2} - 1 \qquad (6)$$

c. The Modification of BBS Random Key Generator Layer

In this layer, the BBS random key generator is modified to generate a 2D [3×3] array of keys, where the two large prime numbers are required for the first block only. For the other block, the keys are generated based on the previous key, and so on to the last block in the input image.

Algorithm: Shifting and Permutation Algorithm

Input: Two Random prime numbers.
Output: The 2D[3×3] array of keys.

The recipient chooses two distinct primes p and q that are 3 modulo 4.
calculate n = p .q (The number n will be public key, p and q (n factorization)are private key).
The sender chooses a random number $1 < xx_0 < N$ (xx0:it used as a seed for BBS random key generator which is a quadratic residue modulo n)
check gcd(f(n), xx0)= 1. //gcd: greatest common denominator)
check $2 < xx0 < f(n)$ // $(n) = (p-1\,.\,q-1)$
Calculate the sequence k0, k1, k2, ..., kn, kn+1 for each i $\in$ [0, n]

$$k_{i+1} = k_i^2 \bmod n$$

For each generated key in the array [3x3]
  IF index of generated key odd
    Summation of the value in the array of keys with its index
  Else
    Summation of the value in the array of keys with its index+2
  End IF
End For
End Shifting and Permutation

d. Coding Layer.

In this layer, every block resulting from the previous layer was coded, and every pixel was split into three components red, green, and blue (pr, pg, pb). Each component (8-bits) was then input into the following coding steps:

Algorithm: Coding Algorithm

Input The input (3×3)block
Output The coded block
1: **For** each block in No.blocks
2: Generate array[3x3] of key(K1) using MBBS random key generator algorithm
3: Perform the inner permutation for each block $NL = IB \bmod 9$ // NL: new location of the pixel in the input block // IB: index of the current pixel in the input image
4: Convert KI to binary
5: S1=the first least 8-bits of KI
6: **IF** Length(S1) < 8
7: **IF** KI is even

padding it with 0's bits until the S1 length reaches 8
Else
padding with 1's bits until the S1 length reaches 8
End IF
calculate : $Npr = NOT(pr) XOR KI$
$Npg = pg \quad XOR \quad KI\left(NOT\left(\begin{array}{c}\text{only for the bits}\\ \text{in the even}\\ \text{index value}\end{array}\right)\right)$
$Npb = pb \quad XOR \quad KI\left(NOT\left(\begin{array}{c}\text{only for the bits}\\ \text{in the odd}\\ \text{index value}\end{array}\right)\right)$
New pixel(Npr ,Npg , Npb)
End IF
End For
End Coding

4. Inverse MLE-Net Method

Decryption is the process of converting encrypted data or ciphertext back into its original, readable form. It is a crucial component of modern cryptography and protects sensitive information from unauthorized access during transmission or storage. The decryption process typically involves using an encryption algorithm and a decryption key, which must match the ones used for encryption. Here are the general steps involved in the decryption process: obtaining the ciphertext, authentication, decryption, providing the decryption key, executing decryption, obtaining the plaintext, and post-processing (optional). It's important to note that the security of encryption relies heavily on the confidentiality of the encryption key. If an unauthorized party gains access to the encryption key, they can decrypt the ciphertext and access the sensitive information. Therefore, protecting the encryption key is crucial for maintaining the security of encrypted data. The action steps of the proposed decryption algorithm (see Figure 2) are clarified as follows:

Algorithm: Inverse MLE-Net Algorithm

Input Cipher image, keys
Output The plain image.

1: Split the ciphered image into (3×3) blocks
2: Prepare the keys and generate the array (3×3) of keys using the MBBS algorithm for each input block.
3: Decipher each input block by using the generated array(3×3) of keys
4: Call Decoding algorithm
5: Split each pixel value in the colour image into three components (pr, pg,pb).
6: Apply the inverse of the modified median filter on each block
7: Merge the decipher blocks to construct the decipher (plain) image.
8: Perform the inverse shifting algorithm.
9: Save the resulting plain image
10: **End Inverse MLE-Net**

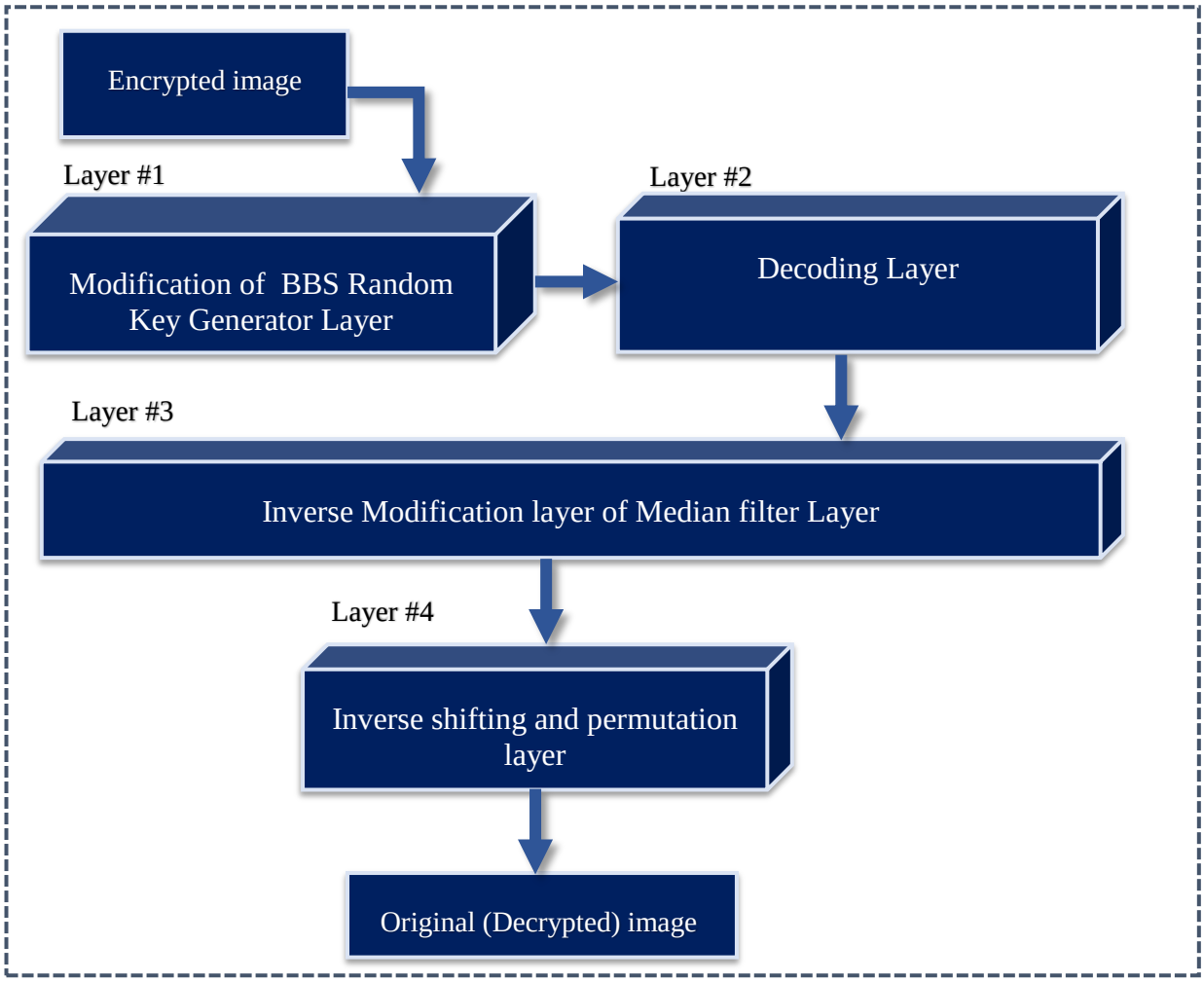

Figure 2: Decryption phase (Inverse MLE-Net)

a. The Decoding algorithm

This decoding algorithm is performed on each pixel in the color-ciphered image after splitting it into three components (pr, pg, and pb), each with 8 bits. The steps of this algorithm are:

Algorithm: Decoding Algorithm

Input Cipher image, keys
Output The decoded block

1: Generate an array [3x3] of keys using the MBBS random key generator algorithm for each block in the input image.
2: **For** each block in the input image
3: **For** I from 1 to 9
4: Perform the inner permutation for each block $OL = IB \bmod 9$ // OL: location of the pixel in the input block
5: Convert KI to binary
6: S1= the first least 8-bits of KI
7: **IF** Length(S1) < 8
8: **IF** KI is even
9: padding it with 0's bits until the S1 length reaches 8
10: **Else**
11: padding with 1's bits until the S1 length reaches 8
12: **End IF**

Calculate : pr = NOT(Npr XOR KI)

pg = Npg XOR KI(NOT(only for the bits in the even index value))

pb = Npb XOR KI(NOT(only for the bits in the odd index value))

New pixel(pr , pg , pb)

End IF

End For

End For

End Inverse MLE-Net

a. The inverse of the Modification layer for the Median filter.

The size of the used block in the proposed algorithm is (3×3).

- If the index of the block is odd, take the pixel value in the centre of the block and return it to the location (number of pixels in the block/2) -1. Then, arrange the value of the block pixels into descending order.
- If the index of the block is even, take the pixel value in the centre of the block and return it to the location (number of pixels in the block/2) + 1. Then, arrange the value of the block pixels into ascending order.

b. The inverse shifting and permutation algorithm.

The steps of the shifting and permutation inverse process, are illustrated as follows:

Algorithm: Inverse of Shifting and Permutation Algorithm

Input: The ciphered image.

Output: The shifted and permutated image

Arrange the pixel value in the location (i=j) in descending order

Replace the pixel value in the location (i>j) with the value in (i<j)

For each row in the length of image (m)

If the number of rows is odd

For each pixel in this row

If the index of it is odd .

subtract 1from the value of pixel

Else

subtract 1from the value of pixel

End IF

End For

End IF

End For

IF the number of row is even then :

For each pixel in this row

IF the index of it is odd .

Add to the value of pixel

Else

Add 2 to the value of pixel

End IF

End For

End IF

For each i in the length of image (m)

For each j in the depth of image (n)

Replace the pixel value in the location (i>j) with the value in (<j)

End For

End For

End Inverse of Shifting and Permutation Algorithm

5. Experimental Analysis and Results

To test the robustness of the proposed encryption and decryption algorithm and determine whether it is safe enough to be implemented [6]. The quality of the encrypted image is usually measured by histogram uniformity, correlation of adjacent pixels, and image entropy.

a. Histogram Analysis.

The colour histogram analysis technique is used to view the suitability of the colour distribution between a plain image and a cypher image [6]. Figs.1-4 below show the histogram of the original and cipher images. The histogram value has a significant distribution of diversity in the cypher image. Also, it has substantial differences from the plain image histogram, so the cypher image does not give any clues to perform a statistical attack on the encryption algorithm.

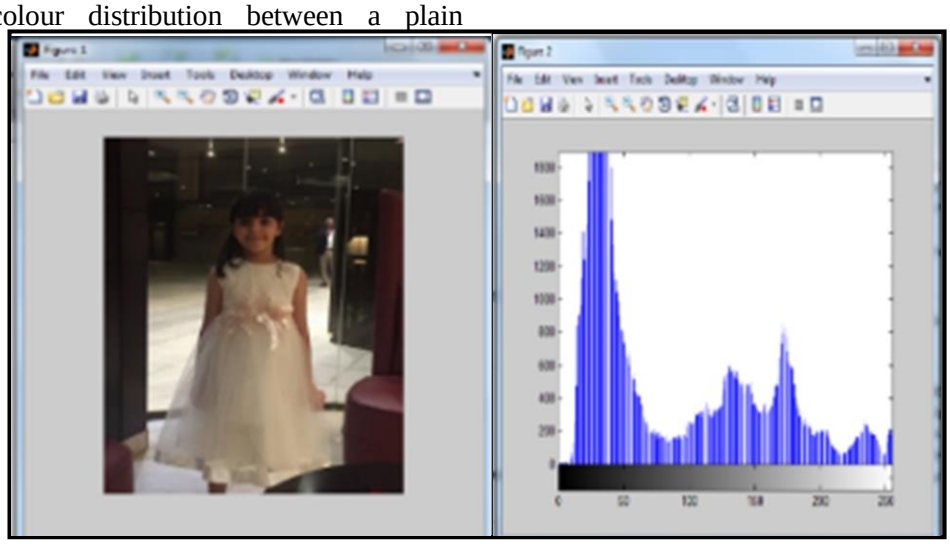

Fig.1 : The First plain image.

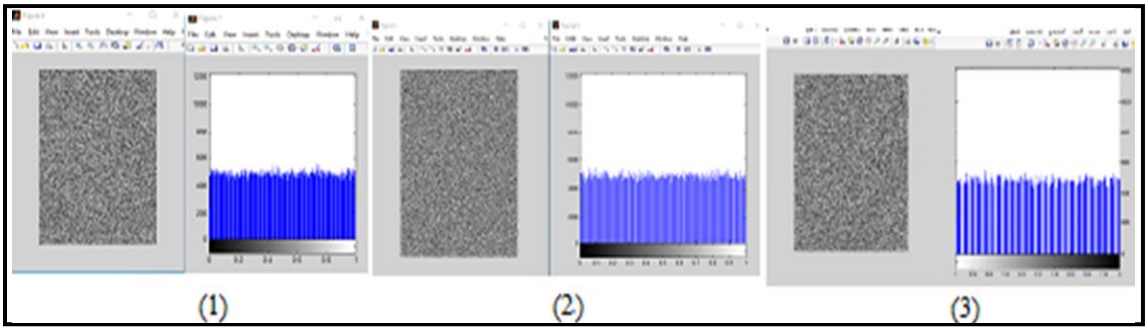

Fig.2: The First encrypted image :(1)red ban component(2)green band component(3) blue band component.

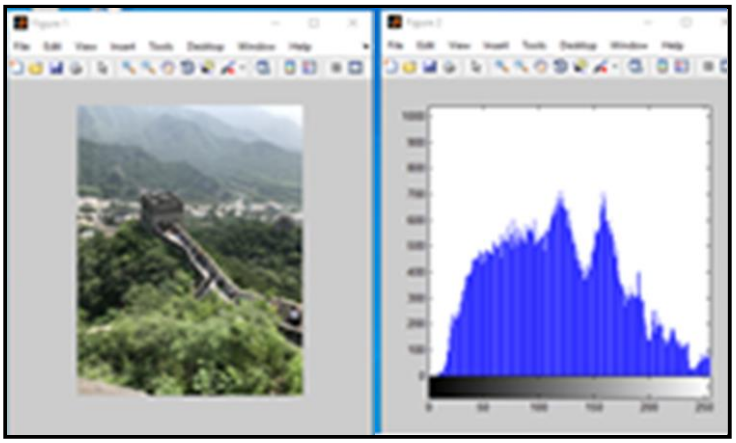

Fig.3 : The second plain image.

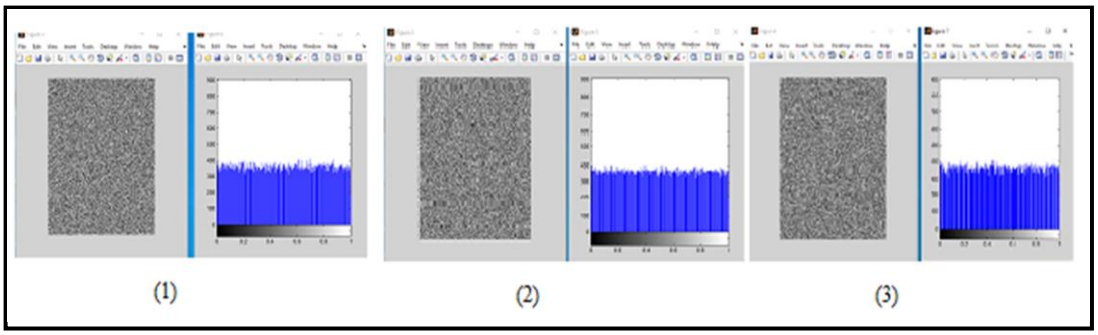

Fig.4: The second encrypted image(1)red ban component(2)green band component(3) blue band component.

a. Correlation

Calculation of correlation and entropy is performed to assess the quality of the encryption image; the lower the correlation between pixels and the higher the entropy value, the more the encryption system can be safe [6]. The correlation coefficients are shown in Table 1, calculated using the following formula:

$$R = \sum \frac{n\sum xy - \sum x \sum y}{\sqrt{[n(\sum x^2 - (\sum x)^2)][n\sum(y^2)\sum y^2]}}$$

where :

r: correlation value, n: the amount of data, x and y: the number of multiplications of x and y, x: the number of data x, y: the number of data y, x^2: the number of x squared, y^2: the number of y squared [6]. In the results in the table, I emphasize that there is no correlation between the pixels of an encrypted image because the encryption algorithm destroyed it.

Table 1: Correlation of plain and encrypted image

Image	Red band	Green band	Blue band
plain image (PI1)	0.9952	0.9949	0.9937
Encrypted image	0.0465	0.0546	0.0482
plain image (PI2)	0.9620	0.9616	0.9774
Encrypted image	0.0437	0.0399	0.0444

b. Speed Test:

In this work, the design and analysis of the proposed encryption and decryption algorithms were simulated using the Matlab (R(2013)a(8.1)) program on a computer (processor: Intel (R)Core

Table 4: Comparison of Entropy information of original and encrypted image and previous works

(TM)i7-4500U CPU @ 1.80GHz 2.40GHz.RAM: 16.00GB). In order to verify the exact operation of the proposed algorithms, many colours were used as a plain image, such as the 512x512x3 colour image. The computational time of the encryption and decryption processes is shown in Table 2. The results illustrate that the proposed system is flexible and high-

Table 2: Processing time

Image	Encryption Process	Decryption Process
plain image PI1)	0.035621	0.031012
plain image (PI2)	0.036843	0.032271

408x306x3 colour image and the speed for applications.

c. Information Entropy Analysis

The degree of uncertainty or unpredictability inside a tiny section of an image is measured by the entropy measure, which is a crucial feature of image encryption. [14] [16]. The information entropies of the encrypted images are less compared to the ideal case (ideal entropy value is 7.99902 (» 8)) [6, 16], which might be suggestive of its resistance to numerous threats. To improve the security of image encryption schemes, it is necessary to consider the local entropy of the image and find ways to increase it. This can be achieved through various techniques, such as using more complex and diverse chaotic maps, incorporating additional randomness in the encryption process, and applying multiple transformations to the image. Table 3 shows the entropy of both original and encrypted photos used in this study. The entropy value can be calculated by the following formula:

$$H = -\sum_{K=0}^{G-1} P(K)\, log_2\big(P(K)\big)$$

Where: H: entropy. , G: The grey value of the image (0-255). ,P(k) : incidence probability of k symbol[4].

Table 3: Entropy information for plain and encrypted image

Image	Red band	Green band	Blue band
plain image PI1)	7.4166	7.2638	7.1491
Encrypted image	7.9995	7.9997	7.9996
plain image (PI2)	7.7009	7.7070	7.7511
Encrypted image	7.9994	7.9996	7.9997

In comparison to the previous study based on the entropy measure, Can Notes outperforms this study, so this means the proposed system is more reliable and secure. See Table 4.

	This study						[13]						[14]					
	Original image			Encrypted image			Original image			Encrypted image			Original image			Encrypted image		
Entropy	Red 7.4166	Green 7.2638	Blue 7.1491	Red 7.9995	Green 7.9997	Blue 7.9996	Red 7.2865	Green 7.5592	Blue 7.0527	Red 7.9981	Green 7.998	Blue 7.9976	Red 7.4710	Green 7.4710	Blue 7.4710	Red 7.8982	Green 7.8982	Blue 7.8982

6. Conclusion

In this research, it was proposed to use a multi-layer model to encrypt colour images efficiently with the use of the modified Blum-Blum-Shub pseudo-random number generator: The pseudo-random number generator was used to generate the appropriate keys to encrypt the image due to the many features it has. Image encryption with the proposed model (called Multi-Layer Encryption Net (MLE-Net)) was implemented in the form of layers, each of which has an essential role in encrypting and decrypting the image securely and efficiently. The experimental results showed that the proposed model could achieve excellent effects in the process of encrypting and decrypting images and that it is secure because the colour histogram of the original image and the encrypted image showed a significant difference between them. Also, the information entropy analysis showed that the information leakage in the encrypted image is low and that the entropy value is close to 8. So, the image is robust against an entropy rule attack. Also, the average correlation values between the original and encrypted images are close to 0. The execution time for encryption and decryption of the proposed model is relatively fast. Therefore, the complexity of time and computations will be low, which is considered a good indicator of the success of the method in encryption and maintaining data privacy.

References

[1] Zakaria, S. B., & Navi, K. (2022). Image encryption and decryption using exclusive-OR based on ternary value logic. Computers and Electrical Engineering, 101, 108021. https://doi.org/10.1016/j.compeleceng.2022.108021.

[2] Ephin, M., Joy, J. A., & Vasanthi, N. A. (2013). Survey of Chaos-based Image encryption and decryption techniques. In Amrita International Conference of Women in Computing (AICWIC'13) Proceedings published by International Journal of Computer Applications (IJCA).

[3] Gupta, K. R. I. S. H. A. N. (2013). Different image encryption and decryption techniques and KA image cryptography. Int J Adv Comput Eng Netw, 1(10), 2320-2106.

[4] Mohamed, G. S., & Eng, D. P. (2015). Design and Implementation of A New Hybrid Encryption Algorithm. Journal of Madenat Alelem University College, 7(2), 49-64. DOI: 10.13140/RG.2.2.33143.06565.

[5] Mohammed, G. S. (2017). Text encryption algorithm based on chaotic neural network and random key generator. Ibn AL-Haitham Journal For Pure and Applied Science, 29(3), 222-233.DOI: 10.13140/RG.2.2.19721.29285.

[6] Setyaningsih, E., & Iswahyudi, C. (2012). Image encryption on mobile phone using super encryption algorithm. TELKOMNIKA Indonesian Journal of Electrical Engineering, 10(4), 835-843.

[7] Oad, A., Yadav, H., & Jain, A. (2014). A review: image encryption techniques and its terminologies. International Journal of Engineering and Advanced Technology (IJEAT) ISSN, 2249-8958.

[8] Singh, P., & Singh, K. (2013). Image encryption and decryption using blowfish algorithm in MATLAB. International Journal of Scientific & Engineering Research, 4(7), 150-154.

[9] Mohammed, G. S., Al-Janabi, S., & Haider, T. (2022, December). A comprehensive study and understanding—A neurocomputing prediction techniques in renewable energies. In International Conference on Hybrid Intelligent Systems (pp. 152-165). Cham: Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-27409-1_14

[10] Kamal, S. T., Hosny, K. M., Elgindy, T. M., Darwish, M. M., & Fouda, M. M. (2021). A new image encryption algorithm for grey and color medical images. IEEE Access, 9, 37855-37865.

[11] Man, Z., Li, J., Di, X., Liu, X., Zhou, J., Wang, J., & Zhang, X. (2021). A novel image encryption algorithm based on least squares generative adversarial network random number generator. Multimedia Tools and Applications, 80, 27445-27469.

[12] Sinha, K., Paul, P., & Amritanjali, A. (2022). An improved pseudorandom sequence generator and its application to image encryption. KSII Transactions on Internet and Information Systems (TIIS), 16(4), 1307-1329. https://doi.org/10.3837/tiis.2022.04.012

[13] Jirjees, S. W., Alkalid, F. F., & Shareef, W. F. (2023). Image encryption using dynamic image as a key based on multilayers of chaotic permutation. Symmetry, 15(2), 409. https://doi.org/10.3390/sym15020409

[14] Mfungo, D. E., Fu, X., Xian, Y., & Wang, X. (2023). A Novel Image Encryption Scheme Using Chaotic Maps and Fuzzy Numbers for Secure Transmission of Information. Applied Sciences, 13(12), 7113. https://doi.org/10.3390/app13127113

[15] Boneh, D. (2011). Blum-Blum-Shub Pseudorandom Bit Generator.

[16] Gawande, K., & Mundle, M. (1999). Various implementations of Blum Blum Shub pseudo-random sequence generator.

[17] Sidorenko, A., & Schoenmakers, B. (2005, December). Concrete security of the Blum-Blum-Shub pseudorandom generator. In IMA International Conference on Cryptography and Coding (pp. 355-375). Springer, Berlin, Heidelberg.

[18] Taha, M. S., Mohd Rahim, M. S., Lafta, S. A., Hashim, M. M., & Alzuabidi, H. M. (2019, May). Combination of steganography and cryptography: A short survey. In IOP conference series: materials science and engineering (Vol. 518, No. 5, p. 052003). IOP Publishing. DOI 10.1088/1757-899X/518/5/052003

[19] Bikos, A., Nastou, P. E., Petroudis, G., & Stamatiou, Y. (2023). Random Number Generators: Principles and Applications. doi: 10.20944/preprints202309.0879.v1

[20] Vybornova, Y. D. (2017). Password-based key derivation function as one of Blum-Blum-Shub pseudo-random generator applications. Procedia Engineering, 201, 428-435.