International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 12, No.1 (Sep-2022)

https://dx.doi.org/10.12785/ijcds/120168

Image Encryption using a Binary Search Tree Structure-Based Key

Mohammed Abbas Fadhil Al-Husainy[1], Hamza A. A. Al-Sewadi[2] and Bassam Al-Shargabi[3]

[1]Al-Maaqal University, Basrah, Iraq
[2]Iraq University College, Basrah, Iraq
[3]Middle East University, Amman-Jordan

Received 21 Jun. 2021, Revised 29 Mar. 2022, Accepted 20 May 2022, Published 30 Sep. 2022

Abstract: Due to the ever-increasing cybercrime and hazards on digital information stored or in transit over computer clouds and networks, so many encryption algorithms were developed and practically implemented. On the other hand, hackers and intruders keep on developing methods to break those algorithms. Hence, new methods are always sought and developed by researchers. A binary search tree (BST) is implemented in this paper to produce a new algorithm for image encryption. The BST is utilized to generate an encryption key that consists of two parts; local and global with flexible length capabilities that provide better security. Sharing all image contents to encrypt any byte of the source image helped to achieve Shannon's concept of diffusion and confusion. The experimental application of this algorithm has manifested a satisfactory security performance as compared with the widely used cryptographic systems such as Advanced Encryption Standard (AES) and Data Encryption Standard (DES). These Comparisons included measurement of encryption time complexity, Peak Signal to Noise Ratio (PSNR), Entropy, encryption key space. Besides, the new method offers encryption key length flexibility and involvement of all image contents in its generation.

Keywords: Image-Encryption, Data-security, Cryptography, Avalanche-Effect, Binary Search Tree.

1. Introduction

With the vast growth of computer systems involved in all walks of life for storage, processing and communication, the security, integrity, and authenticity became of prime concern. The solution to this problem can be achieved by either cryptography or data hiding algorithms. Efficient cryptographic algorithms such as advanced encryption standard (AES), data encryption standard (DES), triple DES, and many others [1]–[3] were developed and practically implemented for encrypting all sorts of multimedia (texts, images, audio, and video files). However, with the increased computer efficiency and capabilities, the worrying growth of data security breaching measures by hackers and criminals, demanded continuous efforts to develop new security algorithms. Therefore, great efforts are spent by industrial and academic research teams in order to protect commercial and governmental data that is either transmitted over various digital communication channels or is stored in computer systems [4].

Prior to 1976, only symmetric cryptographic algorithms were known, where only one secret key is needed to be used for both encryption and decryption (referred to as secret-key algorithms). These algorithms require a secure channel to distribute the secret key to intended users. However, a new technique, developed by Diffie and Hillman [5] suggested an asymmetric algorithm which solved the problem of key distribution. It uses a pair of different, but related keys; one is made public and the other is private. One is used for the encryption process while the other for decryption process (referred to as a public-key algorithm). Symmetric algorithms are comparatively fast and more difficult to break, but has the serious drawback of key distribution, while asymmetric algorithms are slower and less secure, but do not need key distribution, besides, it is suitable for key distribution [6][7].

E-mail: dralhusainy@gmail.com, alsewadi@hotmail.com, bshargabi@meu.edu.jo

Image security is of interest for many domestic and commercial applications, hence to achieve strong image security, strong encryption keys for symmetric cryptography algorithms are sought. This work presents a novel technique for image encryption adopting symmetric system, where the encryption keys are generated using a binary search tree (BST) method.

After this brief definition in section 1, section 2 summerizes a related work review. Then, section 3 outlines the methodology of the proposed image cryptography algorithm. Section 4 describes the algorithm implementation, lists the experimental results and includes the discussion. Finally, section 5 concludes the paper.

2. Related Works

Recently, so many techniques have been developed for image encryption, however, only cryptographic algorithms involving binary tree will be mentioned here. For example, in 2007, Lim and Mun [8] applied binary search on prefix length to the area-based quad-tree using a packet classification algorithm. Their method adopts the technique of constructing multiple disjoint trees following relative hierarchical level rule, therefore avoiding the needed pre-computation when conducting a binary search on length. Two new optimization techniques based on rule priorities were suggested by the researchers, implementing few thousands of different rules and recording the memory consumption and the number of bytes per rule in their test. The observed performance is noticed to be steady and independent of the table characteristics.

Nieto et. al., 2012 [9] introduced a forward security (FS) scheme that reduces damage due to compromised keys. This is produced by the powerful setting of hierarchical predicate encryption (HPE) in order to improve the secrecy of decryption keys. The FS-HPE combined setup guarantees forward security for plain messages and for attributes that are hidden in HPE ciphered message. The construction resembles to a technique specific to existing HPE schemes which were built in a form similar to the binary tree encryption by FS-PKE.

Saraswathi and Venkatesulu, 2012 [10] proposed an algorithm that uses blocks of bits instead of only bytes or pixels. Any multimedia file such as audio, image, video, and text data, whether its content is compressed or not, can be encrypted by the binary tree traversal (BTT) random substitution method through performing bits shifting processes for its rows and columns, achieving considerable and comparable security in comparison with well established algorithms, like DES and Blowfish.

Aathithan et. al., 2013 [11] proposed a new symmetric block cipher that provides confidentiality to the transmitted multimedia via the network. They built a complete binary tree to perform the substitution and transposition operations using a two-dimensional array and a pseudo-random permutation key P. The performance of their algorithm showed that it is suitable for real-time multimedia communication applications, however, it suffers from few seconds delay on startup.

Datta and Gupta, 2013 [12] presented a hybrid system combining both encryption and watermarking processes applied to audio files by implementing discrete wavelet transform (DWT) algorithm on audio files up to the third level. This led to produce a binary tree-like scheme. The analysis of signal-to-noise ration (SNR) values during the experiments showed that the proposed approaches succeeded to achieve an encrypted audio file with considerable quality degradation, which makes unauthorized users to only have low quality copies of the audio files.

Damrudi and Ithnin, 2013 [13] utilized parallel processing on the well-known RSA public-key cryptography algorithm using binary tree architecture. They proved that one can speed up the encryption time of the RSA algorithm with reasonably more security.

Agrawal et. al., 2014 [14] discussed the existing transposition techniques of encryption and also presented a new block cipher based on the transposition operation using a binary tree and other data structure approaches.

In order to resist the attacks, a new strategy uses a binary tree encryption scheme has been proposed by Yi and Tan, 2016 [15]. Their strategy aimed to increase the security of encrypting multiple-images to provide high-security management of the authority among the users sharing a cipher image. The simulation tests showed that their scheme was highly dependent on the asymmetric double random encoding in the gyrator domain.

An interesting new encryption algorithm that is based on four block ciphers encryption technique is developed by Priya et. al., 2017 [16]. Binary tree traversal is implemented for executing nonlinear diffusion parallel processing of substitution and two-dimensional array. Memory size requirement, encryption time complexity, and CPU utilization were all enhanced. Moreover, security strength is improved as compared with existing encryption schemes like AES, RC5, and RC6.

Based on a binary tree also, Alderman, et. al, 2017 [17] designed a space-efficient key assignment scheme (KAS), that eliminates public information and imposes a logarithmic bound on the required number of derivations. Users in this design may require more cryptographic material. The authors try to migrate that by designing heuristic optimizations of the mapping. The performance test showed that the designed scheme produces good results compared to existing schemes.

Honglin et. al., 2020 [18] presented an encryption system based on optical algorithm for hyperspectral images implementing an improved BTS and phase-truncated discrete multiple-parameter fractional Fourier transform (PTdmpFrFT). This improved BTS branch node represents PTdmpFrFT scheme, where the hyperspectral bands of images were considered as leaf nodes. The contents of each pair of these bands are encrypted using the improved IBTS, followed by an asymmetric encryption using PTdmpFrFT scheme. Hence, this method generates different pairs of bands having different secret keys and encryption/decryption paths. Their findings showed fair security strength, reduced computation steps, and moderate storage requirements.

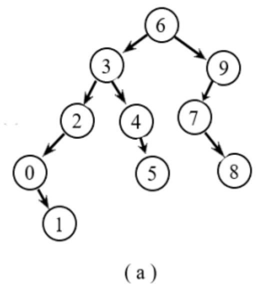
Nematzadeh et. al. 2020 [19] reported an image encryption method based on Deoxyribonucleic Acid (DNA) sequence and BST. The plain image with a chosen candidate BST is converted to a DNA sequence creating DNA-BST, which is then superimposed over the DNA image. Finally, using XOR function, the DNA image is converted to the cipher image. Their experimental results approve the robustness of the proposed method against well-known attacks, but on the price of increased processing time due to the addition of more randomness in the encryption process.

Alabdullah et. al. 2021 [20] reported a novel encryption method based on the reflection of BST. It implements the reflection property of a balanced BST structure to minimize the overhead and dynamic offset to achieve a high security level. Compared with AES and DES algorithms, their method achieved the lowest running time, consumed comparable memory usage, and satisfied the avalanche effect criterion of 50.1%.

Al-Husainy and Al-Sewadi, 2019 [21] reported the idea for generating the encryption key for image encryption algorithm based on the use of BST structure. It has free user controlled, flexible length, resulting in high security level. Preliminary experimentation on digital images were satisfactory and comparable with algorithms like DES and AES algorithms which encouraged the authors to expand the encryption scheme design and parameter test to produce this work.

To clarify the use of BST structure first, a brief description of its implementation for key selection is presented here. The structure of any BST mainly depends upon the order of the values placed in the nodes of the tree. Figure 1 shows an example of two BSTs that are created from the same values (0...9). These values are inserted in two different orders, resulting in two different binary trees, namely; (6, 9, 7, 3, 2, 0, 1, 8, 4, 5) and (4, 0, 1, 9, 8, 6, 5, 3, 7, 2).

When trying to extract the values of the nodes in the path starting from each node down the path to the root of the tree, except the node itself, we obtain different sequences of values in each of the two different trees. The extracted sequences for each node (N_0… N_9) in the two trees are listed in Table I.



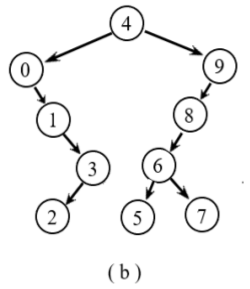


Figure 1. Different sequences of values result in completely two different binary trees. (a): (6, 9, 7, 3, 2, 0, 1, 8, 4,5) and (b): (4, 0, 1, 9, 8, 6,5,3, 7,2).

TABLE I. THE SEQUENCE OF VALUES IN PATHS STARTING FROM EACH NODE TO THE ROOT (EXCEPT FOR THE NODE ITSELF).

Node	Tree (a)	Tree (b)
N_0	2,3,6	4
N_1	0,2,3,6	0,4
N_2	3,6	3,1,0,4
N_3	6	1,0,4
N_4	3,6	-
N_5	4,3,6	6,8,9,4
N_6	-	8,9,4
N_7	9,6	6,8,9,4
N_8	7,9,6	9,4
N_9	6	4

3. MATERIAL AND METHOD

In the proposed encryption method, many different binary trees are created from the primary input key, then these trees are used as sub-keys in the encryption and decryption operations.

The input to the encryption phase is the secret key *K* and the source image *S*, this phase produces the encrypted image *E* as output. Then at the decryption phase, the input is the key *K* and the encrypted image *E* while the output is re-production of the original source image *S*. Both *S* and *E* are bitmap images and the secret key can be any digital file of any type.

The proposed algorithm treats *K*, *S*, and *E* as files that contain a series of bytes. In the encryption phase, S_{Length} and K_{Length} represent the length in bytes of *S* and *K*, respectively. The required preprocessing operations are performed on the input data in both phases. Figure 2 shows a general block diagram summarizing the operations that take place in the encryption phase. A detailed explanation of the operations performed in the encryption phase of the proposed algorithm is given in the following.

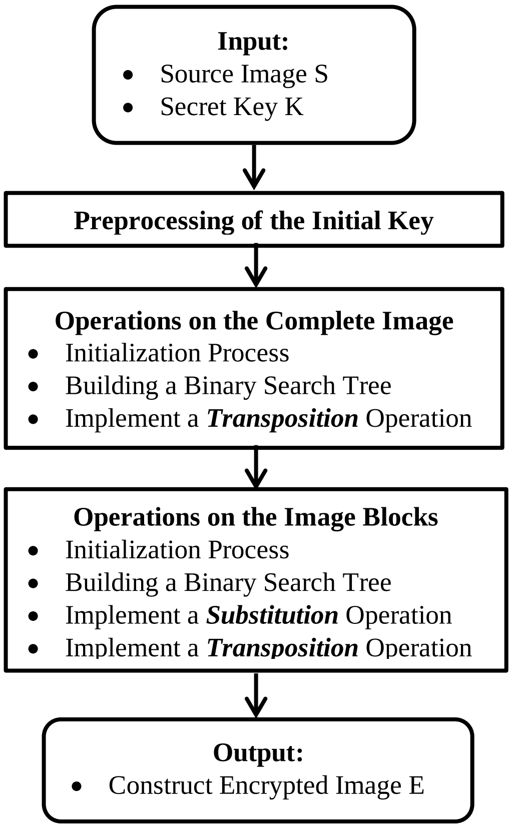

Input:
Source Image S
Secret Key K
Preprocessing of the Initial Key
Operations on the Complete Image
Initialization Process
Building a Binary Search Tree
Implement a Transposition Operation
Operations on the Image Blocks
Initialization Process
Building a Binary Search Tree
Implement a Substitution Operation
Implement a Transposition Operation
Output:
Construct Encrypted Image E

Figure 2. General block diagram of the proposed method

3.1 Preprocessing of the Initial Key

Initially, the bytes stored in *K* are read and used to produce a new value for each byte by implementing equation (1). This operation enhances the difficulties for an attacker to know the length of the secret key and all the secret key bytes in order to breach the security of the encrypted image. The newly created key will be used in the proposed algorithm.

$$K_n = K_n \oplus K_m \;\; for\; n, m: 0\; to\; K_{Length}\; and\; n \neq m \qquad (1)$$

Where *n* and *m* are the nth and mth bytes in the secret key *K*, respectively, and the sign ⊕ represents the XOR operation.

In sections 3.2 and 3.3, the encryption processes are performed at two different levels; on the complete source image as a single block and on image blocks after splitting the source image into a group of small blocks. In both levels, the proposed encryption method generates the required binary search tree(s) in order to perform the necessary operations at each level.

3.2 Operations on the Complete Image

In order to involve all the bytes of the source image in a single operation, a kind of relationship among all source image bytes is created. This will cause a widespread impact on the whole source image. It will force the attacker to find all bytes of the source image in order to recover any single byte. The following operations are implemented by processing all bytes of the source image as a single block.

1. *Initialization Process*

A set of steps is performed to prepare the desired key and the source image in order to implement the encryption operation defined at this level.

Step 1: The bytes of *S* are indexed from 0 to $S_{Length-1}$. Therefore, equation (2) is used to find the minimum number of bits *N*, which are necessary for the indexing each byte in *S*.

$$2^N \geq S_{Length} \qquad (2)$$

Step 2: Create a new list *KB* from the secret key *K* that contains the representation (in bits) of all bytes in *K*. Equation (3) is used to calculate the length of the created list *KB*.

$$KB_{Length} = K_{Length} \times 8 \qquad (3)$$

Step 3: From the *KB*, sequentially read *N* bits and represent them as a decimal value. The obtained decimal values are stored in a list to generate a new key *K'*. The remaining bits that are less than *N* at the end of the *KB* are ignored. The values in the key *K'* are limited in the range 0 to $S_{Length-1}$.

2. *Building a Binary Search Tree*

At this stage, the necessary BST will be built in order to be used to extract the desired lists of values that are used in the next 'transposition' operation. This is achieved by the following steps.

Step 1: From the last generated key *K'*, read the distinct values sequentially, without repetition, and use these values to generate the corresponding BST. The total number of nodes in the generated BST is S_{Length}. The tree must contain one node for each value in the range 0 to $S_{Length-1}$. After completing the reading operation of the *K'* values, the missing values that were not found in the *K'* must be inserted into the BST.

Step 2: For each node from N_0 to $N_{SLength-1}$ in the BST generated in Step 1, extract the lists L_0 to $L_{SLength-1}$ of nodes' values in the path starting from each node up to the root of the tree except the value of the node itself.

3. *Implement a Transposition Operation*

The goal of performing the transposition operation in any encryption method is to make a change in the location of each element in the source data. This process achieves wide diffusion effects in the contents of the source image *S*, adding more difficulties to the attacker's efforts. In the proposed method, for each byte S_n at index *n* in *S* (where

n: 0… S_{Length}), perform a successive exchange of the location of S_n with the locations (stored in the list L_n) of bytes in S. Carrying out this transposition operation helps to set a new random location to each byte in S.

3.3 Operations on the Image Blocks

To cause a more specific and profound impact on bytes in the source image S, the source image will be split into a set of blocks of equal size, where the same set of operations is performed on each block. These operations are outlined in the following operation.

1. *Initialization Process*

A set of steps is executed to prepare the secret key K and the source image S to perform the encryption operations at the block level.

Step 1: Divide S into a group of blocks of 256 bytes length and store every 256 bytes in a list. This is done by sequentially reading every 256 bytes of the source image S. The number of generated lists is calculated using equation (4). Where *Div* means integer division and S_i is the ith list of the source image S. Figure 3 illustrates an example of S_i list.

$$NoOfLists = S_{Length} / 256 \qquad (4)$$

S_i	Index	0	1	2	…	254	255
	Byte value	123	19	0		207	87

K_i	Index	0	1	2	…	254	255
	Byte value	150	8	231		69	37

Figure 3. Examples of S_i and K_i lists.

Step 2: Similarly, the secret key K is divided into a group of blocks of 256 bytes length and each block is stored in a list. The number of lists that are created from the secret key K must be equal to the lists in Step 1. Where K_i is the i^{th} list of the secret key K. If the number of bytes in the secret key file K is less than the number of bytes in the source image S, then the bytes in the secret key file are repeated to match the number of bytes in S. Figure 3 illustrates an example of K_i list.

2. *Building a Binary Search Tree*

At this stage, a BST will be built from each previously created secret key list (K_0… $K_{NoOfLists-1}$). This tree is used to extract the desired lists of values to be used in the next steps for the substitution and transposition operations.

Step 1: For each list K_i, read the distinct values sequentially, without repetition, and use these values to generate the corresponding BST. The total number of nodes in the generated BST is 256. The tree must contain one node for each value in the range 0 to 255. After completing the reading operation of the K_i values, the missing values that are not found in the K_i must be inserted into the BST.

Step 2: For each node from N_0 to N_{255} in the BST generated in Step 1, extract the lists L_0 to L_{255} of nodes' values in the path starting from each node up to the root of the tree except the value of the node itself.

3. *Implement a Substitution Operation*

For each byte value B_m at the index m in the list S_i, where m: 0...255, perform the XOR logical operation for B_m with all bytes values in the L_m using equation (5).

$$B_m = B_m \oplus L_{mk} \qquad (5)$$

Where k refers to the values in the list L_m.

The implementation of the substitution operation will change each byte value in S to a new random value based on the generated BST, from the key list K_i, which is used for each list S_i of the source image S.

Changes in bytes' values help to achieve the desired confusion effects in the source image S, i.e. contributing to the protection of encrypted data against attackers.

4. *Implement a Transposition Operation*

In addition to the confusion effect that occurred during the substitution operation, a diffusion effect will be made during the transposition operation by changing the location of each byte within each list S_i. Hence, for each byte B_n at index n in the list S_i, where n: 0...255, perform a successive exchange of the location of B_n with the locations (stored in the list L_n) of bytes in S_i. Hence, a new random location is assigned to every byte in S_i.

This local diffusion affects each block of the source image S and will add to the global diffusion effect that was achieved in the previous section (3.2, equation (3)) to enhance the security of the encrypted data.

3.4 Construct Encrypted Image *E*

In order to build the encrypted image E, the resulted lists of the source image L_0...$L_{NoOfLists-1}$ that are produced from the last transposition operation are combined and stored as an encrypted output image E.

In the decryption phase, the encrypted image E and the same secret key K are entered as input and the source image S will be produced as the output.

To decrypt the encrypted image E, the same operations described in the encryption phase are performed but in reverse order. This will reproduce the original source image S.

4. Experiments and Results

The proposed encryption method is tested and evaluated by conducting experiments on hundreds of images. Various images of different sizes and contents were used to verify the performance of the suggested method and determine its strengths and weakness as compared with existing methods. Of all these images used for the test, six completely different images are selected to be included here as examples. They are Baboon (128×128), Lena (256×256), Petra (424×283), Photographer (128×128), Minions (182×182), and Sunset (461×260). These images are shown in Fig. 4.

A number of standard tests have been used in the experiments to evaluate the suggested method such as visual and statistical tests, avalanche effect, Information Entropy, encryption execution time, key space, and correlation analysis. Moreover, the suggested method is compared with other widely used cryptosystems like DES and AES in terms of some important criteria, such as encryption execution time, key space (Brute force attack), normalized mean absolute error (NMAE), and peak signal-to-noise ratio (PSNR). These results are presented and discussed below.

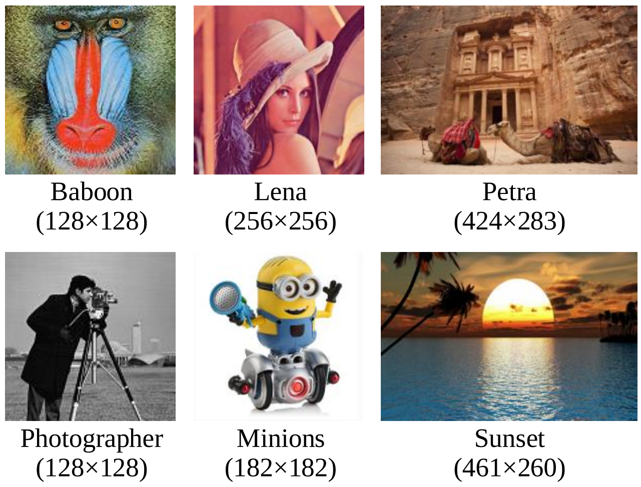


Figure 4. Examples of selected source images used in the experiments.

4.1 Visual and Statistical Tests

For any proposed method for image encryption, the level of achievement of diffusion and confusion effects is noticed through the distortion ratio that occurs in the encrypted image. This is one of the major factors in ensuring the efficiency of the image encryption method. A high rate of distortion in the encrypted image prevents observers from knowing the nature of the source image, hence contributes to the protection of the source image. Using the proposed method for encrypting images has resulted in very high distortion in the encrypted images, as illustrated in Fig. 5 for the chosen examples, namely the images of Fig. 4.

The ability to produce a histogram for the encrypted image with a high level of flatness in the distribution of the color intensity of the image contents (or byte values) compared to the source image histogram, is another important factor of the performance efficiency of the proposed image encryption method. Figure 6 illustrates the computed contents histograms of the byte intensity distribution of the source and encrypted images under consideration. Comparison of the obtained histograms for encrypted images with those for source images indicates that using the proposed method for encryption resulted in a high level of flatness in the byte value distribution compared with those of corresponding source images. The flatness factor plays a major role in protecting encrypted images against statistical analysis attacks.

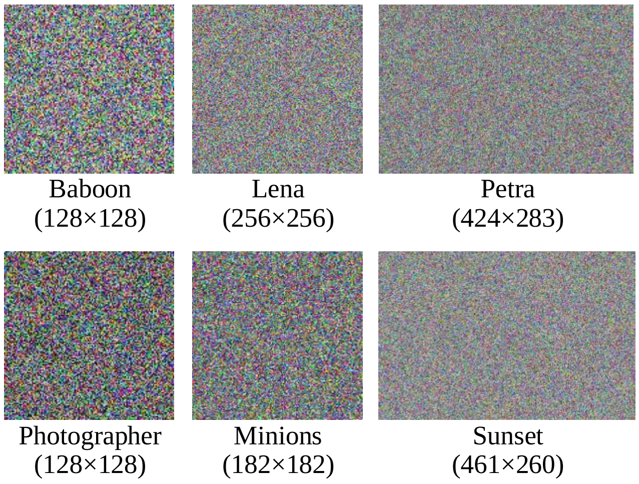
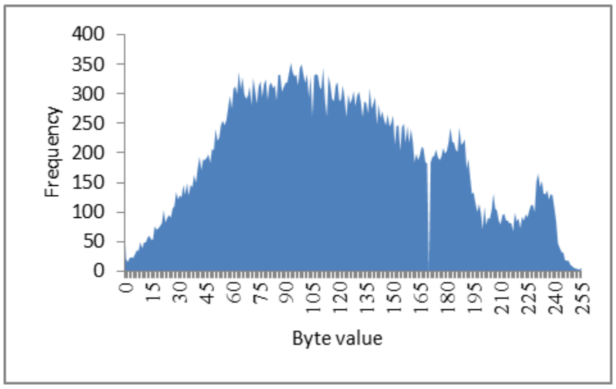
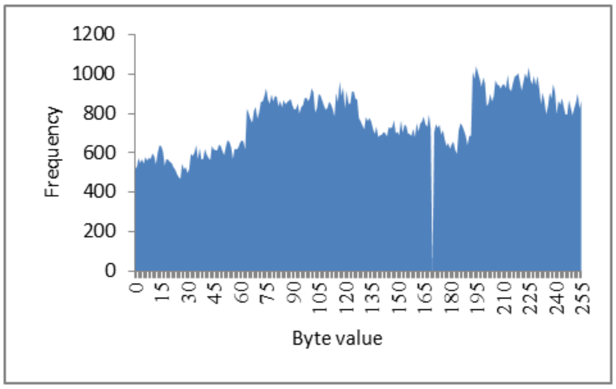


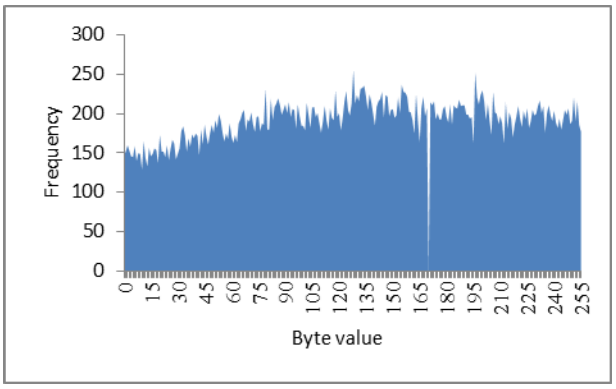
Figure 5. The encrypted image corresponding to the source images of Fig 4.



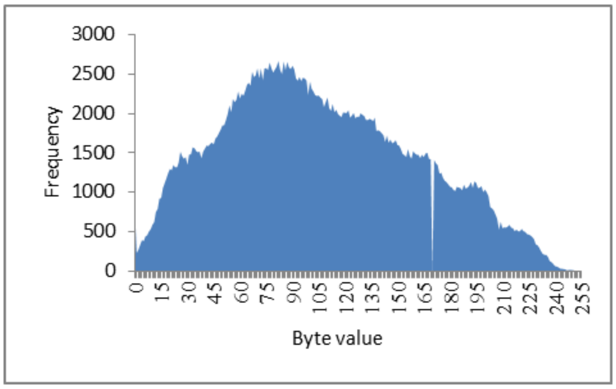
Source



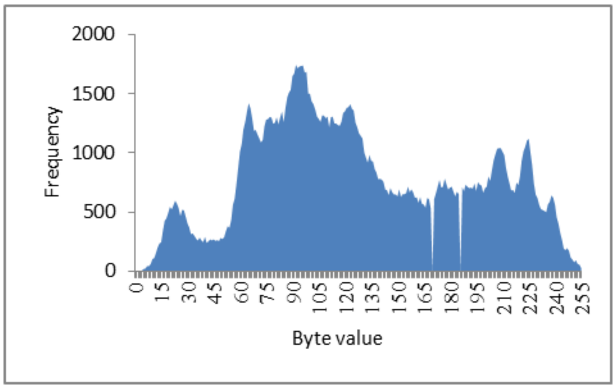
Encrypted (b) Lena



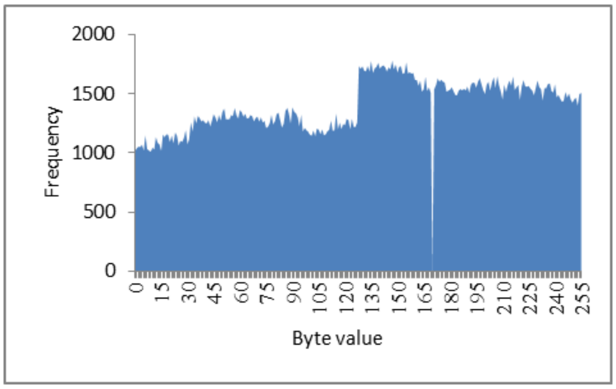
Encrypted (a) Baboon



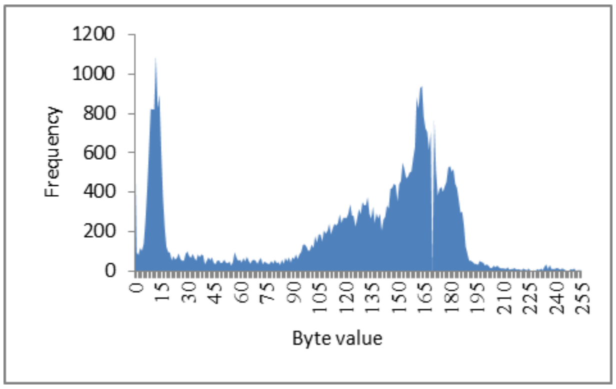
Source



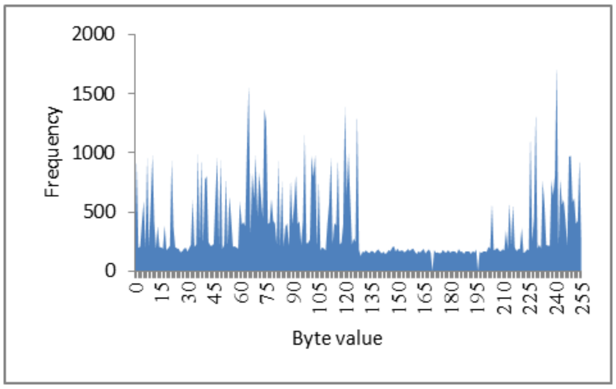
Source



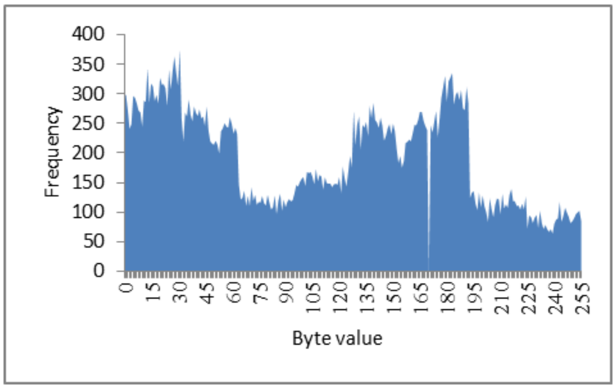
Encrypted (c) Petra



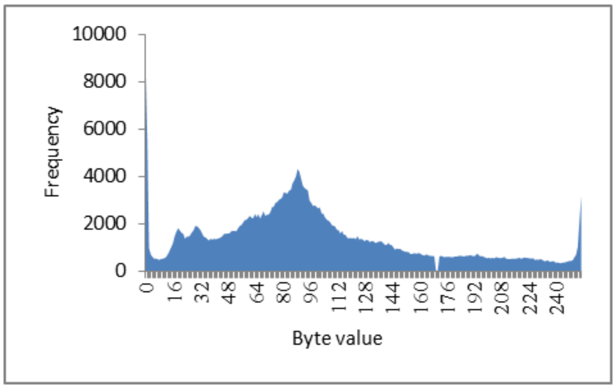
Source



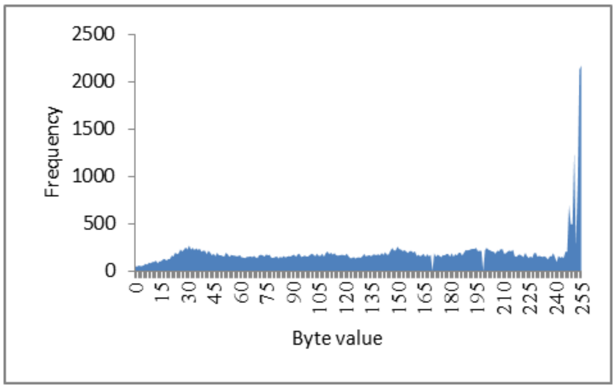
Encrypted (e) Minions



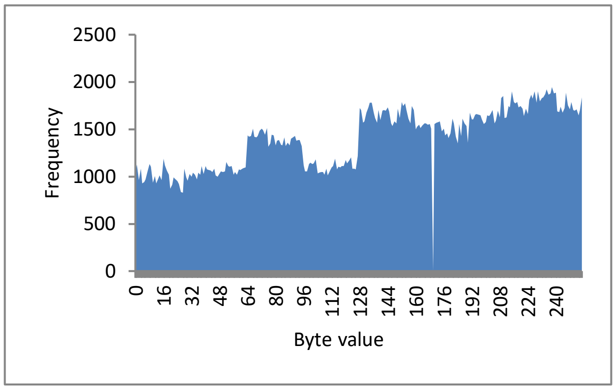
Encrypted (d) Photographer



Source



Source



Encrypted (f) Sunset

Figure 6. Histograms of the source and encrypted images, (a) Baboon (b) Lena (c) Petra, (d) Photographer (e) Minions and (f) Sunset.

4.2 Avalanche Effect Test

One of its performance efficiencies of the encryption method is the sensitivity to any small changes in its parameters. This point should be taken into consideration as one of the main design objectives for any encryption method. Avalanche effect is a numeric and visual test that is commonly used to evaluate the sensitivity of any encryption method.

Avalanche effect for any cryptosystem is good when a slight change either in the source data or the key causes a considerable change in the ciphered data. In the next two subsections, a different number of bits were changed in the encryption key, and the effects of these changes in the generated encrypted image and the recovered source image are calculated.

4.2.1 Avalanche effect on the encrypted image

In order to examine the avalanche effect on the suggested image encryption method, equation (6) [20] is used during the experiments to calculate the avalanche effect or the percentage of the number of bits in the encrypted image that will change when few bits are changed in the used encryption key.

$$Aval.Effect = \frac{Number\ of\ Changed\ bits\ in\ key\ used}{Total\ number\ of\ bits\ in\ encrypted\ image} \times 100 \qquad (6)$$

This test is performed for hundreds of images during the experiments, and the calculated avalanche effect is listed here for the encryption of the six example images; Baboon, Lena, Petra, Photographer, Minions, and Sunset. The test was conducted by changing a different number of bits in the encryption key in the range from 1 bit to 50 bits. Table II lists the recorded results in these experiments.

TABLE II. AVALANCHE TEST RECORDED RESULTS.

Number of changes in the encryption key (bits)	**Avalanche Effect**					
	Baboon	***Lena***	***Petra***	***Photographer***	***Minions***	***Sunset***
1	50.1	49.9	50.1	50.1	49.8	49.9
7	50.6	50.5	50.2	50.4	50.5	50.6
20	51.8	52.6	52.6	51.9	51.9	51.8
35	54.3	53.8	54.3	54.8	55.8	53.9
50	60.2	59.9	58.9	59.1	60.1	59.6

Table II indicates that only one changed a bit in the encryption key produced about 50% avalanche effect for all considered images. Obviously, it increases with an increase in the number of changed bits in the encryption key.

Moreover, to assess the proposed method efficiency using the avalanche effect with changed bits in the encryption key, it is compared with commonly used encryption methods; AES and DES cryptosystems.

The average of avalanche effect values with changing bits in the encryption key of Table II is calculated for the proposed method as well as for DES and AES cryptosystems using the same example images and listed in Table III. Comparison of the recorded values in this table indicates an improvement in the sensitivity of the proposed method to changes in the secret key compared with the other methods.

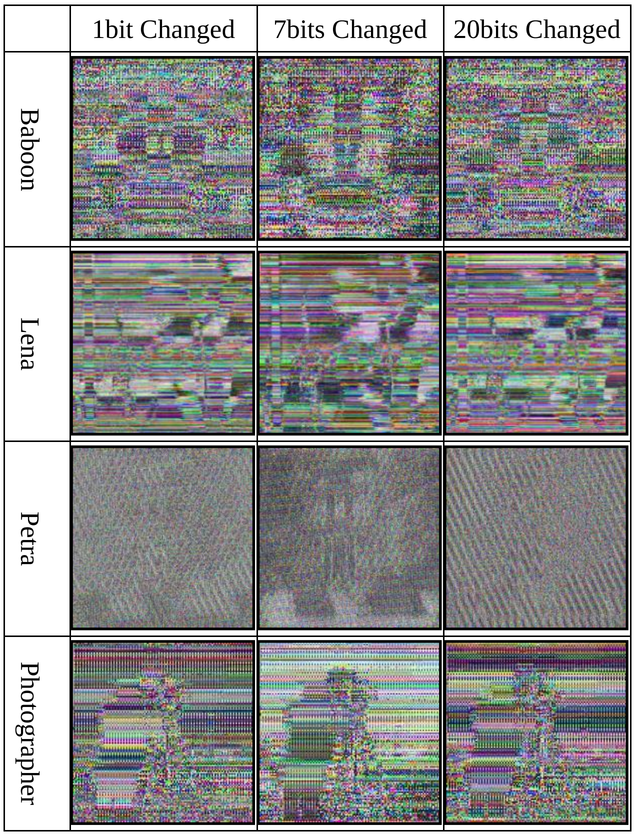
Analysis of the results in Tables II and III indicates that the average avalanche effect calculated for the proposed method is acceptable and with slight improvement over those of AES and DES cryptosystems.

4.2.2 Avalanche effect on the recovered source image

The avalanche effect can also be used in another way to test the effect on the content of the recovered images with any changes in the encryption key during the decryption phase. Therefore, some bits in the encryption key were changed, then the key is used for decrypting the encrypted image in order to recover the source image. Figure 7 depicts the effect of changing a number of bits in encryption key on the recovered images for the same examples of Fig. 4.

TABLE III. AVERAGE OF THE AVALANCHE EFFECT FOR THE PROPOSED, AES, AND DES CRYPTOSYSTEMS.

Cryptosystem	**Average of the Avalanche Effect**					
	Baboon	***Lena***	***Petra***	***Photographer***	***Minions***	***Sunset***
Proposed	53.4	53.3	53.2	53.3	53.6	53.2
DES	52.7	52.3	51.1	52.1	50.1	50.2
AES	52.6	52.1	50.9	51.8	51.1	50.1



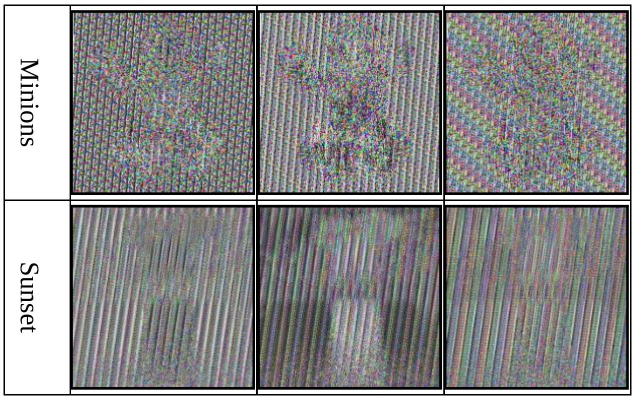


Figure 7. The recovered source images after changing bits in the key.

It is clear from Fig. 7, that when some bits in the key were changed in the decreption phase, the encrypted image does not produce the original source image. This confirms that the proposed method uses an encryption key that is highly sensitive to the avalanche effect.

4.3 NMAE and PSNR Comparison

Three metrics will be used to evaluate the efficiency of the suggested cryptosystem. These metrics are NMAE, PSNR, and Encryption Time (ET).

NMAE and PSNR were calculated using equations (7) and (8) [22], respectively, while ET is the measured elapse time for performing the encryption process. In this work, the experiments were repeated many times and the average encryption time is calculated in order to reduce the time complexity error of the used machine.

$$NMAE = \frac{\sum_{k=0}^{SLength-1} \left| S(k) - E(k) \right|}{SLength} \times 100 \tag{7}$$

$$PSNR_{db} = 10.\log_{10}\left(\frac{Max_S^2}{NMAE} \right) \tag{8}$$

Where: S and E represent the source and encrypted images, respectively and Max_S is the maximum possible pixel value of S.

The NMAE and PSNR are calculated for the proposed method as well as for DES and AES cryptosystems for the six selected images (Baboon, Lena, Petra, Photographer, Minions, and Sunset) and listed in Table IV and Table V, respectively. In these two tables, the comparisons of the recorded results for NMAE and PSNR measurements, whether on single image valuses or when the averages of all the six images are considered, confirm an acceptable performance of the proposed encryption method as compared with the AES and DES algorithms. These comparisons are also shown graphically in Fig. 8 and Fig. 9, respectively.

TABLE IV. COMPARISON OF NMAE VALUES FOR THE PROPOSED METHOD WITH DES AND AES

Image	NMAE (%)		
	Proposed	***DES***	***AES***
Baboon	81.10	79.82	80.98
Lena	86.74	85.26	86.01
Petra	77.34	76.15	76.28
Photographer	58.92	57.88	58.66
Minions	56.92	54.85	54.91
Sunset	99.01	97.29	97.31
Average NMAE value	76.67	75.21	75.69

TABLE V. COMPARISON OF PSNR VALUES (DB) FOR THE PROPOSED METHOD WITH DES AND AES

Image	PSNR (dB)		
	Proposed	***DES***	***AES***
Baboon	8.50	8.71	8.98
Lena	6.98	7.10	6.99
Petra	7.33	7.40	7.38
Photographer	6.40	6.74	6.94
Minions	5.70	6.04	6.04
Sunset	5.78	6.01	6.01
Average PSNR value	6.78	7.00	7.06

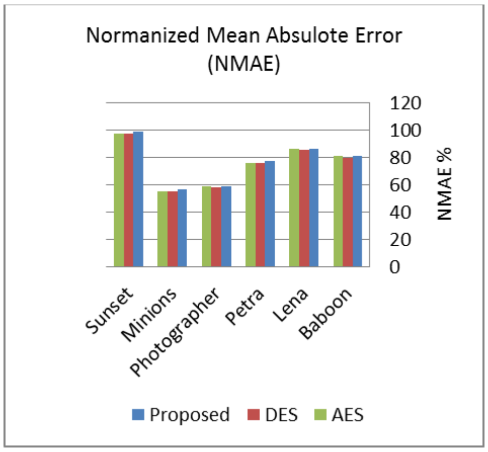


Figure 8. The graphical chart of NAME values of Table IV.

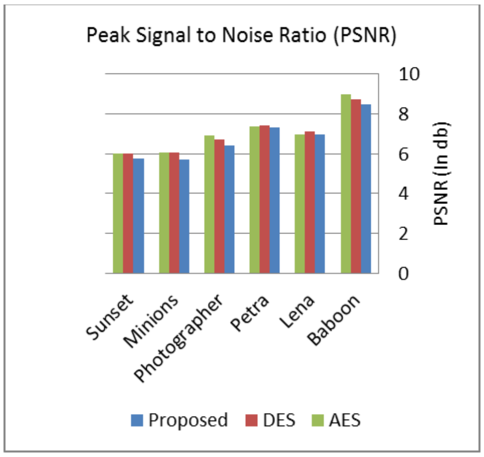


Figure 9. The graphical chart of PSNR values of Table V.

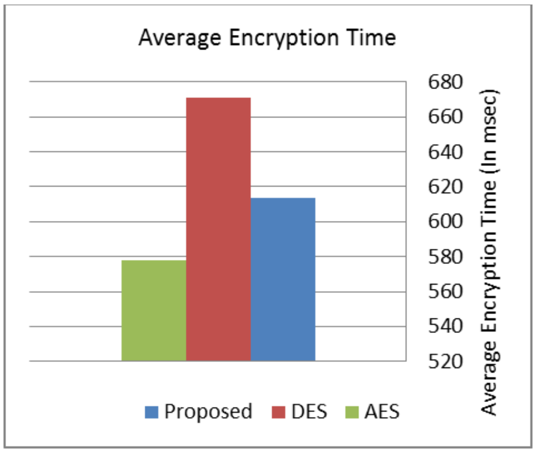


Figure 10. The graphical chart of the average encryption time of Table VI.

In order to examine the speed of the encryption process, the same images have been encrypted using the proposed method as well as DES and AES algorithms, and the observed encryption execution times are recorded. To make a realistic comparison of execution time, the three methods have been implemented using the same computing environment, besides each encryption process is repeated many times, then the average ET is calculated to reduce the expected time complexity error of the used machine. Table VI lists the comparison results for the three methods, while Fig. 10 illustrates the graphical chart of these recorded encryption time values.

It has been found that the measured execution time for the suggested method is shorter than that for DES but longer than that for AES. Hence, one can say that the proposed method can be efficiently used to encrypt images.

TABLE VI. COMPARISON OF AVERAGE ENCRYPTION TIME (ET)

Encryption Method	ET (msec)
Proposed	613.60
DES	671.08
AES	578.22

4.4 Key Size and its Complexity

In the development of any encryption method, the size of the key and the way of using this key in the encryption process play a major factor in the immunity of the method and the level of difficulty encountered by attackers. The brute force attack is primarily affected by the key space and encryption time, the larger the key space, the longer it takes to breach the security. In the proposed method the input key is used at two levels of implementation. First, when processing the whole image as one block and second when segmenting the image into blocks of 256 pixels and then separately processing each block. The number of bits in the key (key size) at of the whole image (K_{Whole}) and in each image block (K_{Block}) levels is calculated using equations (9) and (10).

$$K_{Whole} = K_{Length} \times 8 \quad (9)$$

$$K_{Block} = 256 \times 8 \quad (10)$$

In the proposed encryption method, the user can select a file of any size to be used as an input key, hence any attacker who tries to decrypt one block of the encrypted image needs to know the entire number of bits in the two keys (i.e. $K_{Block} + K_{Whole}$). Certainly, choosing a key size large enough makes the attacker's task extremely difficult. Table VII lists the available key size used in the proposed method together with other encryption methods [23]–[27].

TABLE VII. COMPARISON OF ENCRYPTION KEY SIZE [7]

Algorithm	Largest Key Size Used (in bit)
3DES	168
AES	256
Twofish	256
Blowfish	448
RC4	2048
Proposed Method	> 2048

Furthermore, using a Binary search tree structure in the proposed method to generate the keys used in the two levels to encrypt the image data adds more difficulties to the attackers where each tree has a different structure based on the values used to generate that tree.

4.5 Information Entropy

Information entropy is a fundamental property of the randomness of an input image. Entropy is the average (predicted) amount of information from the data. For a digital image, it is hard to predict the content if its information entropy is high. The information entropy is calculated using equation (11) [28][29].

$$\text{Entropy} = -\sum_{i=1}^{n} P_i . \log_2(P_i) \tag{11}$$

Where n is the number of different values of data and P_i is the occurrence probability of the data value.

The information entropy of the source and encrypted images are listed in Table VIII. These entropy values show that the proposed encryption method achieved a competitive level of randomness in the encrypted image compared with the other considered encryption methods; AES and DES.

TABLE VIII. ENTROPY VALUES OF THE SOURCE AND ENCRYPTED IMAGES

Image	Entropy			
	Source	Proposed	DES	AES
Baboon	4.1827	7.9997	7.9996	7.9998
Lena	4.1740	7.9966	7.9980	7.9975
Petra	4.1802	7.9971	7.9961	7.9973
Photographer	4.2201	7.9974	7.9976	7.9977
Minions	4.1791	7.9923	7.9935	7.9936
Sunset	4.2382	7.9989	7.9977	7.9991
Average Entropy value	4.1957	7.9970	7.9971	7.9975

From Table VIII, the average entropy value for the three encryption methods is found to be almost 8, which indicates that the proposed method is comparable with the other methods and shows that it is complicated to carry out a successful attack.

4.6 Correlation Analysis

The correlation value depicts the relationship between the values of the adjacent pixels in the encrypted image. The encryption technique that produces a small correlation value achieves high randomness between adjacent pixels in the encrypted image. Table VII shows the correlation values of the proposed method together with those of the other commonly used encryption methods, namely AES and DES.

TABLE IX. COMPARISON OF CORRELATION VALUES FOR THE PROPOSED METHOD WITH DES AND AES S

Image	Correlation Values		
	Proposed	*DES*	*AES*
Baboon	0.089	0.090	0.090
Lena	0.110	0.101	0.102
Petra	0.083	0.084	0.083
Photographer	0.082	0.087	0.087
Minions	0.114	0.113	0.122
Sunset	0.090	0.085	0.084
Average correlation value	**0.094**	**0.092**	**0.095**

The average correlation value of the six samples shown in Table IX indicates that the proposed method comparable with other methods

5. CONCLUSION

The binary search tree technique has proven to be useful to produce a strong encryption key for digital images. It lends itself to producing very long, user-controlled keys, resulting in huge key-space that is much larger than the currently available for other encryption algorithms, like AES, triple DES, or RC4. Observed images contents histograms of encrypted images were promising due to the flatness in frequency distribution for byte values in the range from 0 to 256, as well as image visual cognition. Besides, experimental results showed a clear improvement in the encryption execution time, avalanche effect test, correlation analysis test, and good agreement with AES and DES as for the calculated NAME and PSNR values. Hence the proposed algorithm is expected to prove useful as an alternative for practical image encryption sensitive application.

ACKNOWLEDGMENT

Mohammed A. Fadhil Al-Husainy like to thank the Al-Maaqal University, Basrah (Iraq) for financially supporting publication expenses.

Hamza A. Al-Sewadi like to thank the Iraq University College, Basrah (Iraq) for the encouragement and financially supporting publication expenses.

Bassam Al-Shargabi would also like to thank the Middle East University, Amman (Jordan) for financially supporting publication expenses.

REFERENCES

[1] M. Khan and T. Shah, “A literature review on image encryption techniques,” 3D Research, vol. 5, no. 4, p. 29, 2014.

[2] H. Suo, J. Wan, C. Zou, and J. Liu, “Security in the internet of things: a review,” in 2012 International Conference on Computer Science and Electronics Engineering, vol. 3, pp. 648–651, 2012.

[3] S. William, Computer security: Principles and practice. Pearson Education India, 2008.

[4] M. A. F. Al-Husainy, “A novel image encryption algorithm based on the extracted map of overlapping paths from the secret key,” RAIRO-Theoretical Informatics and Applications, vol. 50, no. 3, pp. 241–249, 2016.

[5] W. Diffie and M. Hellman, "New directions in cryptography," IEEE Transactions on Information Theory, vol. 22, no. 6, pp. 644–654, 1976.

[6] S. Chandra, S. Paira, S. S. Alam, and G. Sanyal, "A comparative survey of symmetric and asymmetric key cryptography," in 2014 International Conference on Electronics, Communication and Computational Engineering (ICECCE), pp. 83–93, 2014.

[7] D. P. Joseph, M. Krishna, and K. Arun, "Cognitive Analytics and Comparison of Symmetric and Asymmetric Cryptography Algorithms," International Journal of Advanced Research for Computer Science, vol. 6, no. 3, 2015.

[8] H. Lim and J. H. Mun, "High-speed packet classification using binary search on length," in Proceedings of the 3rd ACM/IEEE Symposium on Architecture for Networking and Communications Systems, pp. 137–144, 2007.

[9] J. M. G. Nieto, M. Manulis, and D. Sun, "Forward-secure hierarchical predicate encryption," in International Conference on Pairing-Based Cryptography, pp. 83–101, 2012.

[10] P. V. Saraswathi and M. Venkatesulu, "A block cipher algorithm for multimedia content protection with random substitution using binary tree traversal," Journal of Computer Science, vol. 8, no. 9, p. 1541, 2012.

[11] N. R. Aathithan and M. Venkatesulu, "A complete binary tree structure block cipher for real-time multimedia," in 2013 Science and Information Conference, pp. 346–352, 2013.

[12] K. Datta and I. Sen Gupta, "Partial encryption and watermarking scheme for audio files with controlled degradation of quality," Multimedia Tools and Applications, vol. 64, no. 3, pp. 649–669, 2013.

[13] M. Damrudi and N. Ithnin, "Parallel RSA encryption based on tree architecture," Journal of the Chinese Institute of Engineers, vol. 36, no. 5, pp. 658–666, 2013.

[14] N. Agrawal, M. Kumar, and M. A. Rizvi, "Transposition cryptography algorithm using tree data structure," in International Conference on Information Communication and Embedded Systems (ICICES2014), pp. 1–6, 2014.

[15] J. Yi and G. Tan, "Binary-tree encryption strategy for optical multiple-image encryption," Appllied Optics, vol. 55, no. 20, pp. 5280–5291, 2016

[16] A. Priya, K. Sinha, M. P. Darshani, and S. K. Sahana, "A Novel Multimedia Encryption and Decryption Technique Using Binary Tree Traversal," in Proceeding of the Second International Conference on Microelectronics, Computing and Communication Systems (MCCS 2017), pp. 163–178, 2019.

[17] J. Alderman, N. Farley, and J. Crampton, "Tree-based cryptographic access control," in European Symposium on Research in Computer Security, pp. 47–64, 2017.

[18] H. Li, X. Bai, M. Shan, Z. Zhong, L. Liu, and B. Liu, "Optical encryption of hyperspectral images using improved binary tree structure and phase-truncated discrete multiple-parameter fractional Fourier transform," Journal of Optics, vol. 22, no. 5, pp. 55701, 2020.

[19] H. Nematzadeh, R. Enayatifar, M. Yadollahi, M. Lee, and G. Jeong, "Binary search tree image encryption with DNA," Optik, Vol. 202, 2020, ISSN 0030-4026, https://doi.org/10.1016/j.ijleo.2019.163505.

[20] B. Alabdullah, N. Beloff, and M. White, "E-ART: A New Encryption Algorithm Based on the Reflection of Binary Search Tree." Cryptography 2021, Vol. 5, No. 4. https://doi.org/10.3390/cryptography5010004.

[21] M. A. F. Al-Husainy and H. A. A. Al-Sewadi, "Implementing binary search tree concept for image cryptography," International Journal of Advanced Science and Technology Vol.130, pp.21-32, 2019. http://dx.doi.org/10.33832/ijast.2019.130.03

[22] M. A. F. Al-Husainy and H. A. A. Al-Sewadi, "Full Capacity Image Steganography Using Seven-Segment Display Pattern as Secret Key," Journal of *Computer Science*, Vol. 14, No. 6, pp 753-763. https://doi.org/10.3844/jcssp.2018.753.763

[23] M. Tagashira and T. Nakagawa, "Biometric Authentication Based on Auscultated Heart Sounds in Healthcare," IAENG International Journal of Computer Science, vol. 47, no. 3, 2020.

[24] S. Liu, C. Guo, and J. T. Sheridan, "A review of optical image encryption techniques," Optics and Laser Technol., vol. 57, pp. 327–342, 2014.

[25] N. Aleisa, "A Comparison of the 3DES and AES Encryption Standards," International Journal of Security and its Applications, vol. 9, no. 7, pp. 241–246, 2015.

[26] K. YueJuan, L. Yong, and L. Ping, "A Searchable Ciphertext Retrieval Method Based on Counting Bloom Filter over Cloud Encrypted Data," IAENG International Journal of Computer Science, vol. 47, no. 2, 2020.

[27] P. K. Kushwaha, M. P. Singh, and P. Kumar, "A survey on lightweight block ciphers," International Journal of Computer Applications, vol. 96, no. 17, 2014.

[28] H. Zhang, J. E. Fritts, and S. A. Goldman, "Entropy-based objective evaluation method for image segmentation," in Storage and Retrieval Methods and Applications for Multimedia 2004, vol. 5307, pp. 38–49, 2003.

[29] W. Zhang, S. Wang, W. Han, H. Yu, and Z. Zhu, "An Image Encryption Algorithm Based on Random Hamiltonian Path," *Entropy*, vol. 22, no. 1, p. 73, 2020.

Mohammed Abbas Fadhil Al-Husainy received the M.Sc. and Ph.D. degrees in 1996 and 2002, respectively. From 1997 to 2002, he was a lecturer in the Department of Computer Science, Al-Hadba University of Mosul. From 2002 to 2013 he has been an associate professor in the Department of Computer Science and Multimedia Systems, Faculty of Science and Information Technology, Al-Zaytoonah University of Jordan. Between 2014 and 2019, he has been an associate professor in the Department of Computer Science, Faculty of Information Technology, Middle East University, Amman-Jordan. Between 2019 and 2020, he has been a professor in the Department of Computer Science, Faculty of Information Technology, Middle East University, Amman-Jordan.

Prof. Al-Husainy taught many courses such as microprocessors, data structures, algorithms design and analysis, digital design systems, operating systems, cryptography, computer organization, programming languages. His research interests are in the broad field of algorithm design, including multi-media data processing, scheduling algorithms, information, and system security, and cryptography and steganography algorithms.

Hamza A. A. Al_Sewadi is currently a full professor at the Iraq University College, Basrah, Iraq. He got his B.Sc. degree in 1968 from Basrah University, Iraq, then M.Sc. and Ph.D. degrees in 1973 and 1977 respectively, from King's College, University of London, UK. He worked as a professor at various universities, including Basrah University (Iraq), Zarqa University, Isra University, Princess Sumaya University for Technology, and Middle East University (Jordan), and visiting professor at the University of Aizu (Japan). His research interests include Cryptography, Steganography, Information and Computer Network Security, Authentication, Discrete Algorithms, Digital Signature, Artificial Intelligence and Neural Networks.

Bassam Al-Shargabi received his Ph. D and M. Sc in Computer Information Systems from the Arab Academy for Banking & Financial Sciences (Jordan) in 2009 and 2004, respectively. He received his BSc degree in Computer Science from Applied Science University (Jordan) in 2003. He has been an associate professor in Departments: Computer Information Systems, Faculty Information Technology, Isra University Amman-Jordan from 2014-2018. Currently, he is an Associate professor Department of Computer Information System, Faculty of Information Technology, Middle East University, Amman-Jordan. AL-Shargabi is an IEEE member. His current research interests are in Natural language processing, information retrieval, and Service Oriented architecture.