Indian Journal of Science and Technology, *Vol 10(33), 10.17485/ijst/2017/v10i33/113085, September 2017*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Multiple Image Secret Sharing based on Linear System

Ashwaq Talib Hashim[1] and Zaid Mundher Radeef[2]*

[1]Department of Control and Systems Engineering, University of Technology, Baghdad, Iraq; 60102@uotechnology.edu.iq
[2]Department of Computer Science, University of Technology, Baghdad, Iraq; zaid_m_alani@yahoo.com

Abstract

Background/Objectives: To build a secure multiple secret image sharing system using multiple levels of security. Methods/Analysis: In this paper a multiple secret image sharing based on linear system has been proposed which is performed (*k, n*) threshold scheme. The proposed system has been used a serial of security techniques to enhance security and reduce storage space to the generated shares. Firstly, the inputs images have been compressed to an attempt to minimize data loss without highly affect the image quality. Secondly, design a 64-bit random number generator by employing the RC5 encryption algorithm round functions with some changes in the usage of the key, where two keys have been used and an initial vector to start the sequence generator. Finally, a developing multiple image secret sharing has been used to generate shares. **Finding:** The developed sharing technique depends on a set of linear equations mixed with modular algebra where n shares have been generated from secure input images, and gathering more than k shares can be recover these secret images. The experimental results showed that the generated shares are more secure and unrelated. Also the results showed that the size of shares have been reduced to more than 1/4k of total sizes of all combined images. And the proposed diffuser flattens the histogram of every simple tested by it, and that proves the efficiency of it to work on every possible input. The overall execution time of the system is inacceptable range. **Application/Improvements:** The secret sharing scheme is an advanced cryptography branch that plays crucial role in defense passively, and it would be used to protect valuable or classified information and documents against dangers like robbery and illegal accesses. The improvement was the adding of multiple secret images instead of using one secret at a time.

Keywords: Block Cipher, Compression, Diffuser, Multiple Image Secret Sharing, Secret Sharing, Pseudo Random Generator

1. Introduction

Data protection and information security is the process of protecting and securing any kind of digital data in various ways that fits its application, the simplest form of security is to encrypt a message to be sent with a cryptographic technique and a key where the cryptography is the practice of writing in secret where the plaintext is encrypted into ciphertext which will requires the other party to decrypt the cipher to reveal the plain text[1].

Another form of data protection and information security is the hashing and message digest, which is a protective techniques to indicate if the received information are correct and haven't been altered, this kind of techniques works by calculating a special unique value for each information called digest and when checking the received information a digest calculating takes progress and then the values compared together to detect any modification changing a single bit can lead to a significant change in digest values which will indicates a modification attack[2].

Data security and protection for visual information leads to the invention of more suitable fields for it, such as the image steganography, visual cryptography, watermarking, image hiding and more.

*Author for correspondence

While steganography, watermarking and image hiding requires a carrier (an object to held the hidden or watermarked info in) visual cryptography and its special secret sharing field creates a data carrier by itself called shares. Visual cryptography is a technique to encrypt visual information such as text, image, and etc.[3]

Developed in 1994, the introduction of visual cryptography best-known techniques has been credited to[3] where the image was broken up into *n* shares and only n shares can reconstruct the secret, the shares were printed on a transparent surface, and to decrypt the image shares were overlaid on each other.

2. Research Methodology

In[4] introduced a secret sharing scheme, which is considered the first secret sharing scheme ever; Blakely scheme was based on n-dimensional space, in which the secret specified as a point on this space, shares in this scheme are the hyperplanes that intersect with secret point. To decrypt the shares, *n* hyperplanes set together will reveal the secret point, while ant less than *n* hyperplanes will make a degree of freedom, leaving this point unspecified In[5] introduced the concepts and theory of secret sharing but in different way, Shamir's secret sharing scheme in contrast to Blakely's scheme was based on a *n*-degree polynomial, and shares were the points on that polynomial[6] proposed an extension to Shamir's scheme in which the secret image is shared by n shares, and any k shares (k<n) can be used to reconstruct the secret, this scheme is started with a permutation technique to shuffle the image's pixels and de-correlate it, then shares were made by processing the image pixels or patterns in the spatial domain, each participant receive his own share as a shadow image look like a random noise image holding partial information of the secret. Share size is just 1/*k* of the secret image[7] proposed a secret sharing approach where the secret image is first transformed to a frequency domain using discrete cosine transformation, DCT coefficients were all discarded except the first 10 coefficients ,second to the tenth coefficients are composed in such a way that they need the first coefficient to be recovered, first coefficient will be embedded in the shares, quality of the reconstructed image is controlled by the number of coefficients to be kept from the DCT, more preserved coefficients mean more quality and[8] proposed a secret image sharing technique in which image difference and Huffman coding were applied on the secret image employing[9] proposed a secret color image sharing scheme based on compression, the compression tend to compact the image leading to better image quality in image reconstruction, secret shares generator is based on a polynomial interpolation and combined with Shamir's scheme to generate smaller shares[10] proposed a simple color image encryption scheme. The action of this scheme is to permute RGB color pixels and transform it to YCbCr color bands, to encrypt these bands in shares three equations are constructed, one for each band, and the secret shares are generated after several iterations. To decrypt the secret, an inverse iterations applied to reconstruct the three equation using Lagrange's interpolation[11] proposed using wavelet technique to perform visual cryptography, he used the wavelet to convert the colored image to a gray image followed by an error-diffusion filter; the halftone generated image is used then to produce shares using his proposed scheme[12] proposed a (k, n) secret image sharing scheme, the secret image is encoded in a noisy shadow image to satisfy a *k* out on *n* scheme, making any *k*-1 shares reveal nothing of the image Wu used a prime number (257), using this prime number eliminates the need to truncate to pixels' value to less than 251 (largest prime less than 256)[13] introduce a security secret color image sharing based on transform coding, using wavelet or cosine transformation to produce secure secret shares by first compress the image using one of the transform coding techniques mentioned earlier. The compressed stream is then subject to a data diffuser followed by a random generator to shuffle the image into shares; a secret shares generator system is then applied on these shares to produce the secret shares.

3. Proposed System

Regular secret sharing algorithms takes one image as input and generates multiple secret shares for that input. Some of the algorithms divide the image into shares without securing it, making it almost visually recognizable using one or less than the required number of shares to restore the image, this type of algorithm produces shares equal in size to the original image. Another type of algorithm adds some sense of security but without regarding the size of the shares since it may take 2X the size of the input image.

The proposed system takes multiple images to produce totally unrelated shares less in size to more than half of the overall size of the images.

Figure 1 shows the overall structure of the proposed system, while algorithm (1) describes the system in steps.

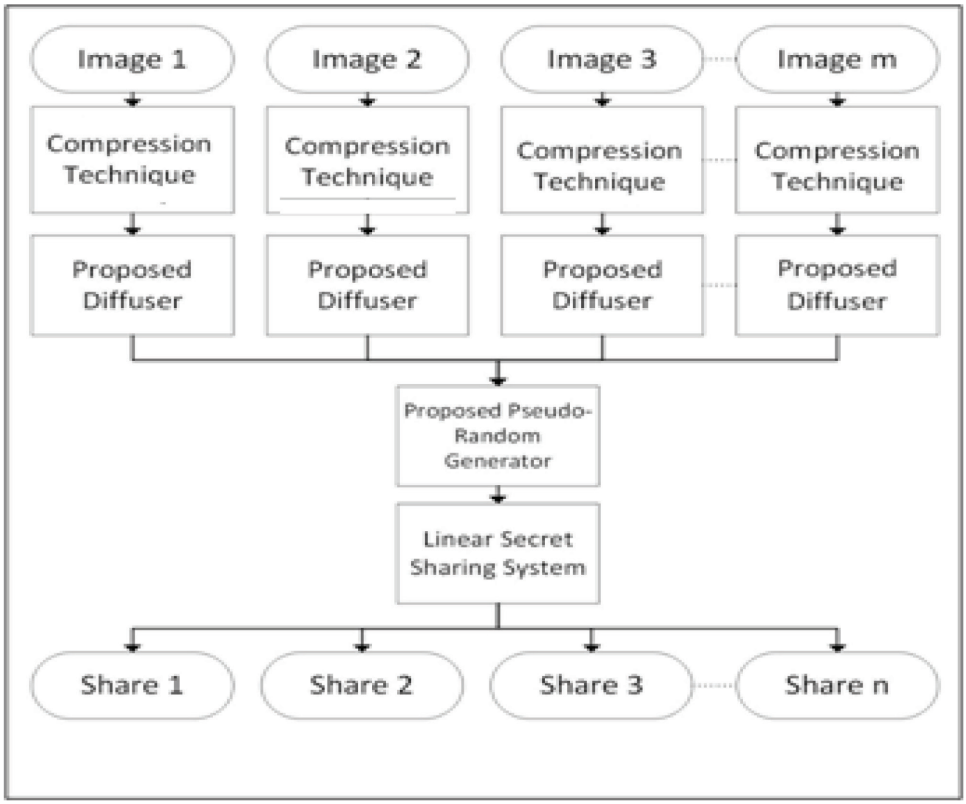


Figure 1. Overall Proposed System.

Algorithm (1): The Developed Multiple Secret Image Sharing Based on Linear System

Input: $I_1, I_2, \ldots, I_m$ *// m color images of equal size.*
n // The number of generated shares
k // Threshold value
Key1 *// 64-bit master key number 1*
Key2 *// 64-bit master key number 2*
IV *// 64-bit initial random generator vector*

Output: S_i *// i=1,..,n.*

Step1: Compress the secret color images $I_1, I_2, \ldots, I_m$ using compression technique adapted from 14.

Step2: Diffuse the resulted *m* compressed streams using the proposed diffuser

Step3: Combine the compressed streams in one total image *T*.

Step4: Separate the total image *T* into *k* sub blocks randomly using proposed Pseudo random generator with Key1, Key2 as seed keys, and IV

Step5: Generate *n* shadows $S_1, S_2, \ldots, S_n$ using Linear System.

3.1 Proposed Diffuser

Substitution and Transposition are considered the core functions in most of the popular encryption systems due to its simplicity but the huge impact on security and complexity. For that reason, a diffuser has been proposed to shuffle and change the multiple compressed images data to increase the security and the complexity using the minimum time and processing operations.

The proposed diffuser design is similar to quadrate design. It is 512-bit block cipher, where quadrate looking composition of F-function is used instead of round. The proposed diffuser uses round-function as F-function in a Feistal construction. The input to F-function is two 128-bit input blocks. It is word-oriented, in that all the internal operations are performed on 32-bit words for very efficient implementation. The proposed diffuser is a reversible function applied on the compressed coefficients to provide the necessary diffusion.

Proposed diffuser was introduced to prune the existing bits significance in DCT coefficients, and consequently, to avoid the localization problem. The basic ingredients of modern fast software encryption schemes are the primitive bitwise computer instructions like ROTATE, ADD, XOR etc. Different subsets of such operations will yield an interesting variety of different permutation groups. Figure 2 shows the block diagram of proposed diffuser. The *P1* to *P4* are 128-bit blocks of 512-bit plain text. The *C1* to *C4* are 128-bit blocks, which in assemble give 512-bit cipher text. Algorithm (2) shows the proposed diffuser steps.

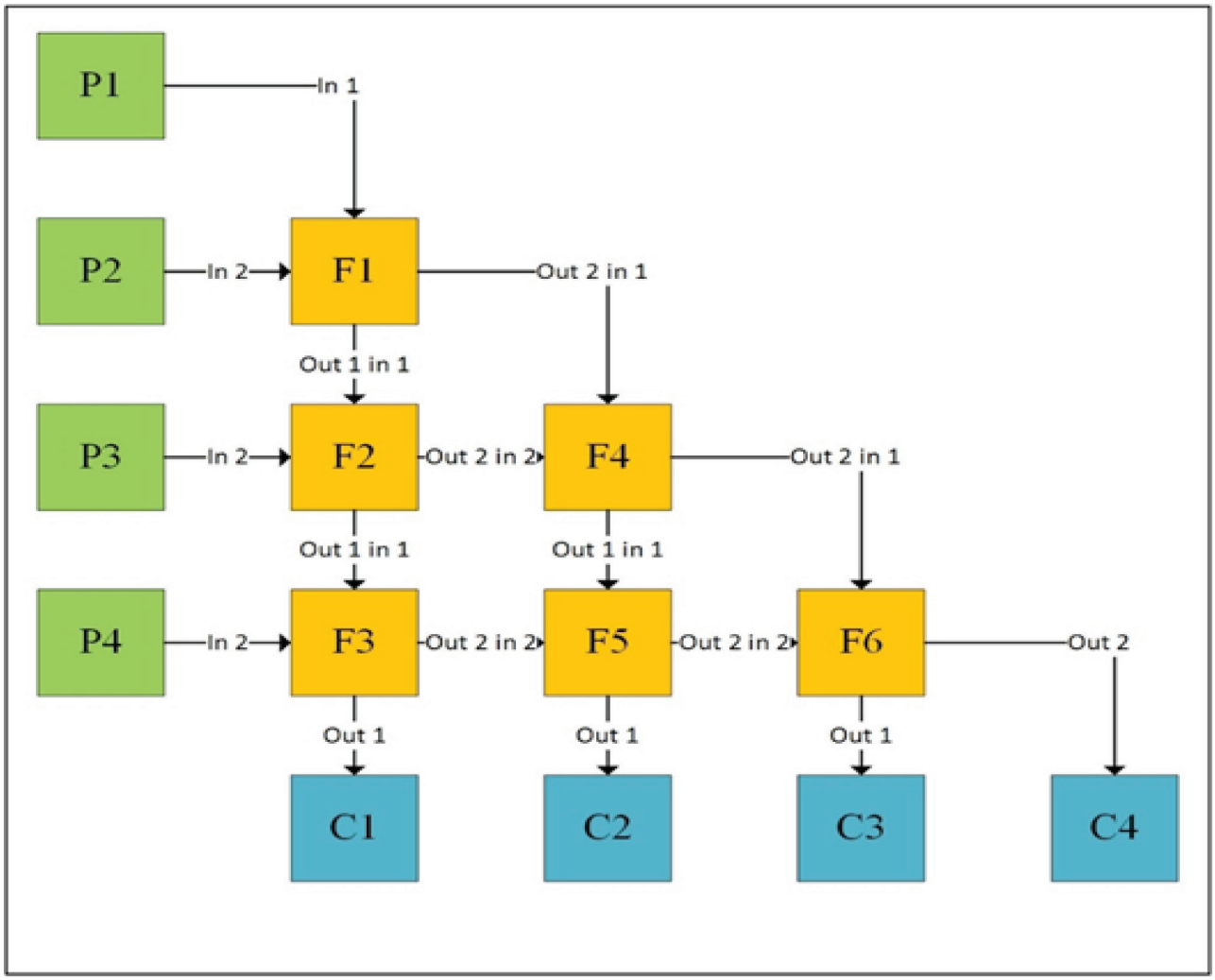


Figure 2. Proposed Diffuser.

Algorithm (2): Proposed Diffuser

Input: *Data* //512-bit block from data

Output: *Diffused Data* //512-bit diffused data block

Step1: Main data divided into blocks 512-bit each

Step2: Each block is divided into four 128-bit sub blocks
P1= first 128-bit from data block
P2= second 128-bit from data block
P3= third 128-bit from data block
P4= fourth 128-bit from data block

Step3: Function applied on sub-blocks such as follow:
Each function takes two sub-blocks as input and produce two sub-blocks of same size
(*F11, F12*) = F1(*P1*, *P2*)
(*F21, F22*) = F2(F11, *P3*)
(C1 , F32) = F3(F21, *P4*)
(*F41, F42*) = F4(*F12, F22*)
(*C2* , *F52*) = F5(*F41, F32*)
(*C3* , *C4*) = F6(*F42, F52*)

Step4: The combined sub-blocks *C1,C2,C3,C4* in *Diffused Data* represents the output of the proposed diffuser

Figure 3 shows the function *F* of proposed diffuser which is a 512-bit block Feistel network and algorithm (3) lists the F-function steps:

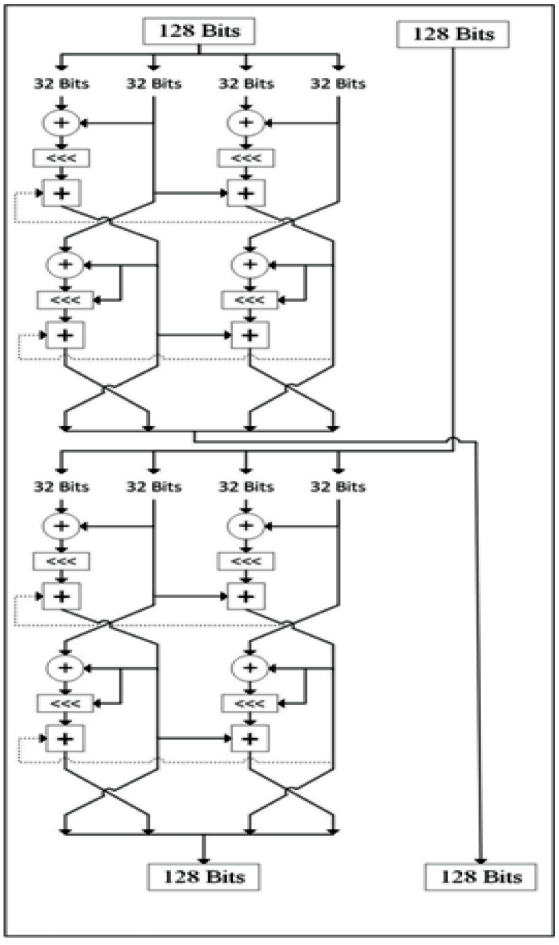


Figure 3. Proposed Diffuser Function.

Algorithm (3): Proposed F-function

Input: *X, Y* // Two 128-bit block

Output: *X', Y'* //Two 128-bit diffused data block

Step1: Divide *X* into four 32-bit sub-blocks x_1,x_2,x_3, and x_4

Step2: Bitwise operators are applied on sub-blocks such as following:

$A_1 = x_1 \oplus x_2$
$B_1 = x_3 \oplus x_4$
$A_1 = A_1 <<< 3$
$B_1 = B_1 <<< 4$
$A_1 = A_1 + x_4$
$B_1 = B_1 + x_2$
$C_1 = x_2 \oplus A_1$
$D_1 = x_4 \oplus B_1$
$C_1 = C_1 <<< A_1$
$D_1 = D_1 <<< B_1$
$C_1 = C_1 + B_1$
$D_1 = D_1 + A_1$
$X' (A_1, C_1, B_1, D_1)$

Step3: Repeat step 2 on the second 128-bit input block providing $Y'(A_2, C_2, B_2, D_2)$ as a result.

Step4: The final results are two 128-bit *X'* and *Y'*.

3.2 Proposed Pseudo Random Generator Based Block Cipher (BCPRG)

A Block Cipher based Pseudo Random number Generator (BCPRG) has been proposed. This step is to de-correlate the output result from compressor and diffuser by distributing it into *N* blocks using a proposed BCPRG based on a secret key, it will generates a random sequence of numbers each has a length equal to the length of combined secret data for all images after compression and their values are ranged to be 1..*r*, then the secret information will be permuted randomly into *N* block according to a certain shuffling mechanism.

This step is considered a preparation to the secret shares creation step by shuffling the combined compressed secret images data in the prototype shares. The shuffling mechanism works by distributing the data to the shares by investigating the random sequence positions and substitute it with a secret data. The algorithm (4) illustrates the detail steps of the proposed BCPRG.

Figure 4 depicts the general structure of the BCPRG where the Key Stream Generator (KSG) is a pseudo random number generator based on block cipher and the keys (i.e., K_1, K'_1) is the seed of the BCPRG and IV is the initial vector. The proposed pseudo random generator is based on using the rounds of the highly random block cipher algorithm RC5.

The KSG is a serial combination of two instances of RC5 block cipher placed into the cipher block chaining encryption mode. The input of the first RC5 is initialized to a public IV, and each block cipher is initialized with its own master key, denoted k_i and k'_i respectively, these keys playing the role of seed for the pseudorandom generator. Figure 5 shows the block diagram of KSG.

As noticed in Figure (5), the xi is the input to the first RC5, and the m_i is an intermediate. Then the output of the KSG is y_i.

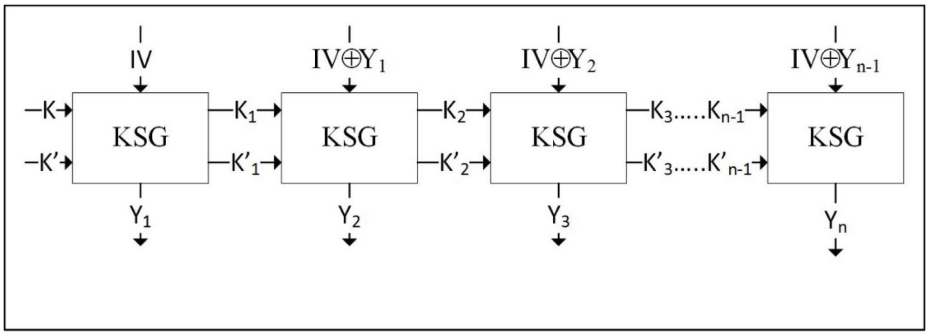


Figure 4. The Proposed block Cipher based Pseudorandom Generator (BCPRG).

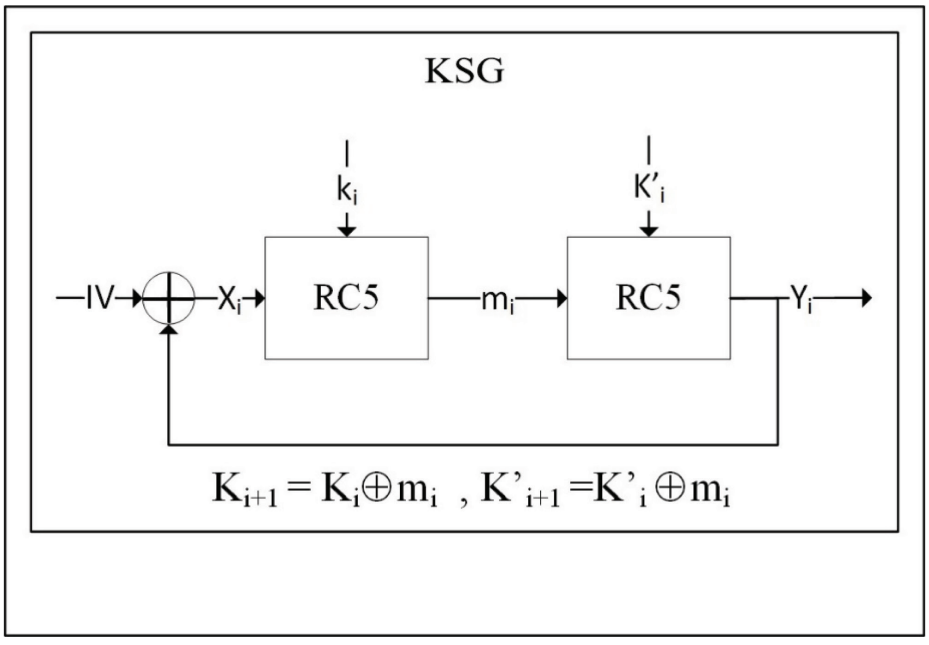


Figure 5. The Proposed key Stream Generator (KSG)

Algorithm (4): Proposed Pseudo Random Generator Based on Block Cipher(BCPRG)	
Input:	S // Combined compressed streams $I_1, I_2, \ldots, I_n$ N // Length of S k // Threshold value K, K' // Two 64 bit master keys IV //64-bit initial random generator vector
Output:	Blocks // Generated *k* subblocks
Step1:	The new input for the first *BK* function is: $x_i = IV$ (1)
Step2:	An intermediate value m_i is computed as: $RC5_{Ki}(x_i)$ (2)
Step3:	Then the output of the KSG is computed as: $y_i = RC5_{K'i}(m_i)$ (3)
Step4:	The new input for the next *BK* function is: $x_{i+1} = IV_{\oplus} y_i$. (4)
Step5:	In the KSG design, the internal state at each step has been used to update running keys such as following $K_{i+1} = K_{i\oplus} m_i$ (5) and $K'_{i+1} = K'_{i\oplus} m_i$ (6) The *K* and *K'* referred to master keys and to K_i and K'_i as the running keys.
Step6:	Repeat step2 to step5 until generate y_i where *i=1..3N*
Step7:	Initialize *b* as a sequence of length *N*: *For I= 1 N* *b[I]= I mod r* *End loop I*
Step8: **Step8-1:** **Step8-2:**	Let $C_1 = y_i$, $C_2 = y_{i+1}$, $C_3 = y_{i+2}$ where *i=1..3N*, check the y_i components (i.e., C_1, C_2, and C_3), if they are not prime then search to find they closest prime numbers: C_1*To_Next_Prime* (C_1) C_2*To_Next_Prime* (C_2) C_3*To_Next_Prime* (C_3) Let $J = C_1$ *For I =N-1→1* *J(× j +) mod I* *Swap b[I], b[J]* *End loop I*
Step9:	*For I=1 → N* *Xb[I]* *IF(X <> 0)* *XX× N / k* *ENDIF* *nocount[X]+1* *wX+ no* *Blocks[w]=S[I]* *End loop I*

3.3 Developed Multiple Secret Image Sharing System

The input $I_1, I_2, \ldots, I_m$ images have been compressed, diffused and then shuffled randomly into k blocks. Figure 6 shows the scrambling stream into k blocks. In the developed multiple image secret sharing, the input content is composed by a set of equal sized blocks (i.e., $Block_1$, $Block_2, \ldots, Block_n$). Each of the blocks can be any type of content.

The developed method consist of two phases: sharing phase and the reveal phase. Details about each phase of the proposed algorithm are described below.

Modula algebra has been used as an assistant with linear equation to reduce the size of the shares.

3.3.1 Sharing Phase

Set of linear equations has been used in the proposed multiple secret image sharing system adapted from[13]; where each i^{th} share has a secret set of *k* integer numbers, a_{ij} (where, $i \in [1,n]$ and $j \in [1,k]$, where *n* is the number of shares and *k* is the minimum required number of shares to rebuild the images).

In other words, for a block of data $\{Qj | j=1..k\}$ (i.e., each Q_j is the j^{th} block), the i^{th} share is computed using the following linear equations:

$$\begin{aligned} S_1 &= c_{11}Q_1 + c_{12}Q_2 + \ldots + c_{1k}Q_k \bmod 255, \\ S_2 &= c_{21}Q_1 + c_{22}Q_2 + \ldots + c_{2k}Q_k \bmod 255, \\ S_n &= c_{n1}Q_1 + c_{n2}Q_2 + \ldots + c_{nk}Q_k \bmod 255, \end{aligned} \quad (7)$$

Where, S_i is the i^{th} generated share for the block *Q()*, c_{ij} is the j^{th} coefficient in the linear equation representing the ith share.

3.3.2 Reveal Phase

The reveal phase is the inverse of coding phase. *k* different encrypted shares, taken from the total *n* shares, are collected for decoding. These *k* shares are used to construct *k* linear equations set (i.e., one for each share), and thereby the secret bytes $\{Qj | j=1,2,\ldots k\}$ can be obtained by solving these linear equations set. If less than *k* of simultaneous linear equations are collected, the linear equations cannot be solved to retrieve the secret bytes {*Q()*}.

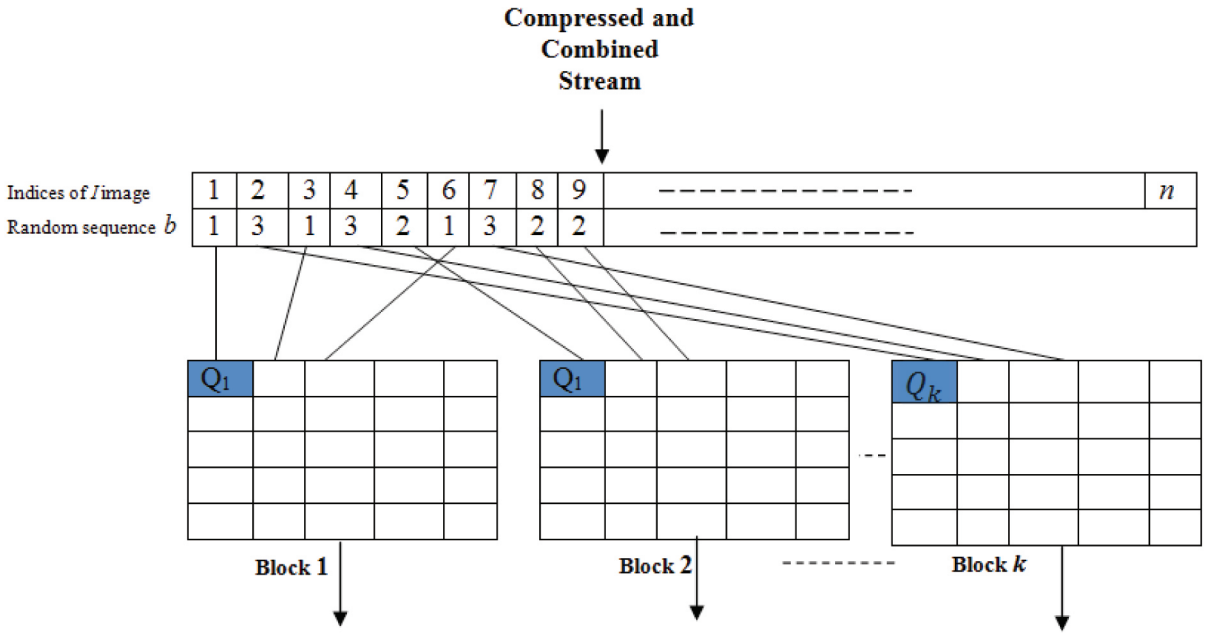


Figure 6. Scramble Stream into k blocks.

4. Experimental Results and Discussion

The experimental results that have been obtained in this chapter showing that the proposed scheme is highly secured and altering of noisy shares will not reveal any partial information about secret images.

To prove the efficiency of each proposed technique tests and implementation has been done on each technique as a separate portion.

4.1 Proposed Diffuser Evaluation

Tests were made on 20 samples with different sizes and scenes, the first test was the visual perception, the result of the 20 samples was totally different than the originals that's an advantage in terms of security, also the results images were almost like each other giving no chance to guess the original image and that's a huge advantage in term of security also.

One of the challengeable terms to achieve in security is the pure frequency, where each component should have a frequency of 1/number of components which is called a uniform distribution, implementing the histogram as a frequency test shows a histogram near to uniformity to the uniform distribution histogram, an a totally different histogram compared to the original image histogram. Table 1 shows the original image, diffused image and the histogram of each image.

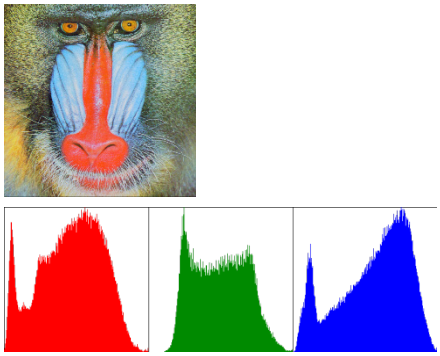
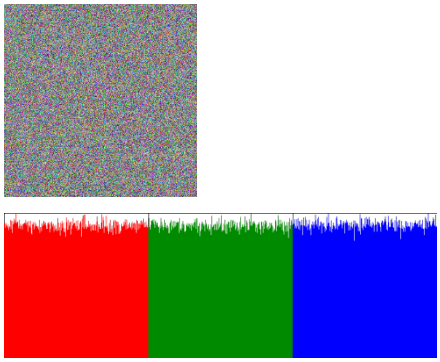
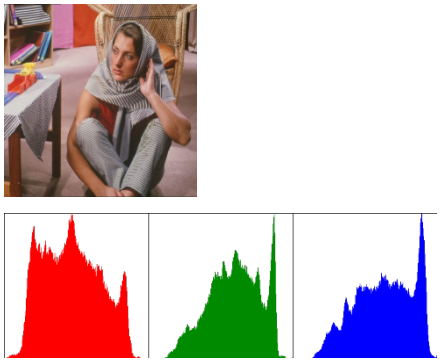
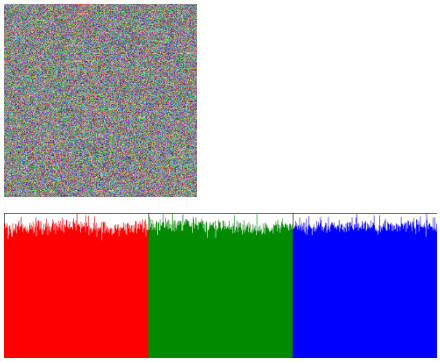
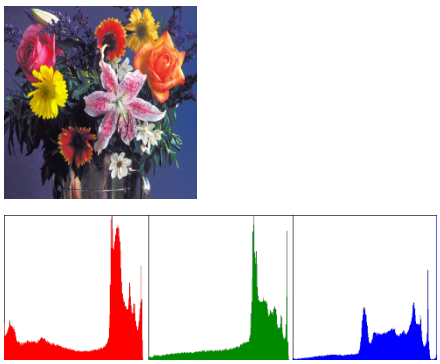
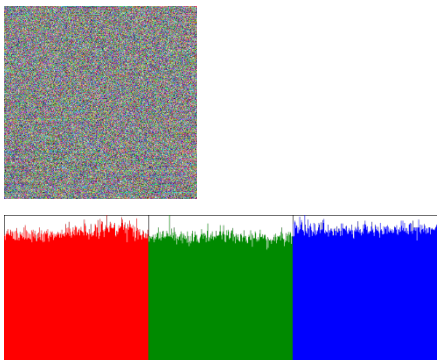
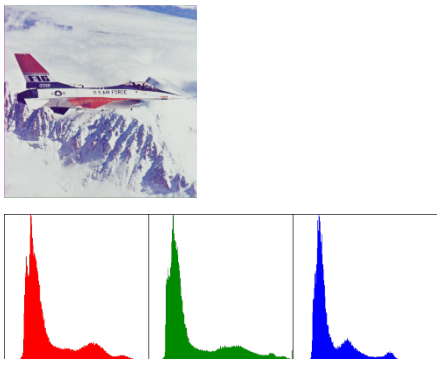
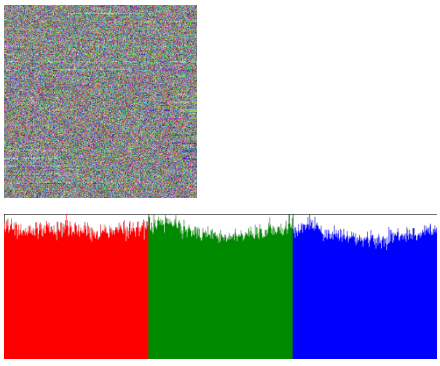
To design a good diffuser scheme, the entropy of diffused image should be as close as possible to the highest value (i.e., for random occurrence of all color or intensity levels are equally probable). The information entropy test which will confirm the previous test results about frequency; for a color image the best information entropy for an 8-bit color band is eight which represent a uniform distribution. Table 2 illustrates the results of each image before and after diffuser.

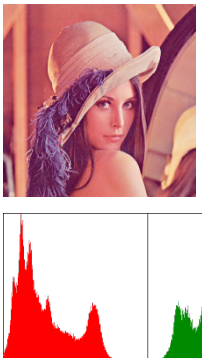
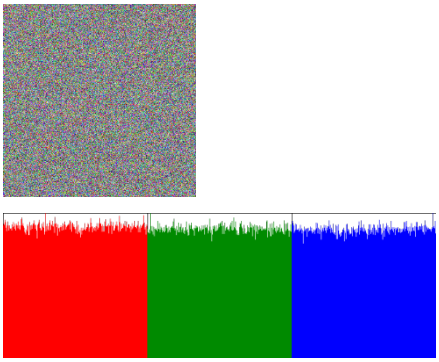
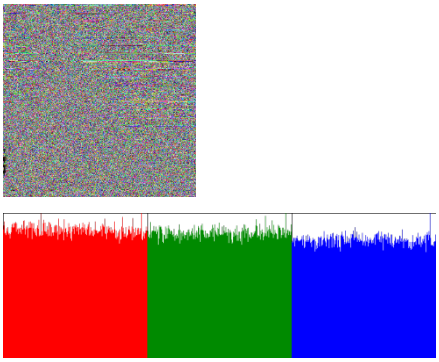
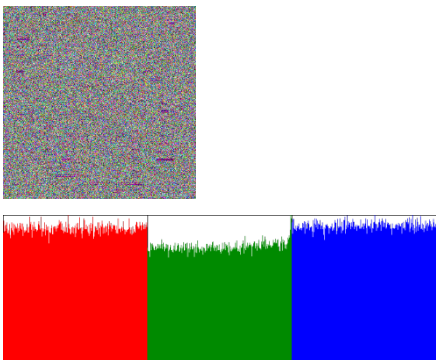
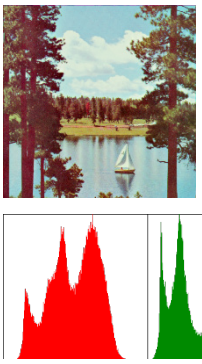
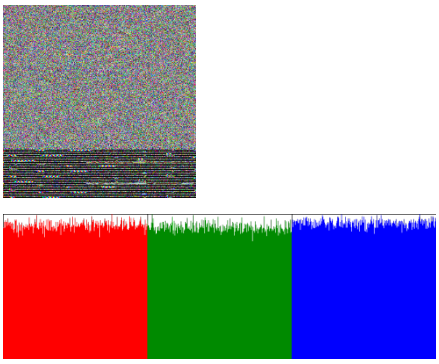
Samples of the test results are shown in Table 3 It is notice that the correlation coefficients are very small ($C \approx 0$), which shows that the plain images and their corresponding diffused images are completely uncorrelated with each other.

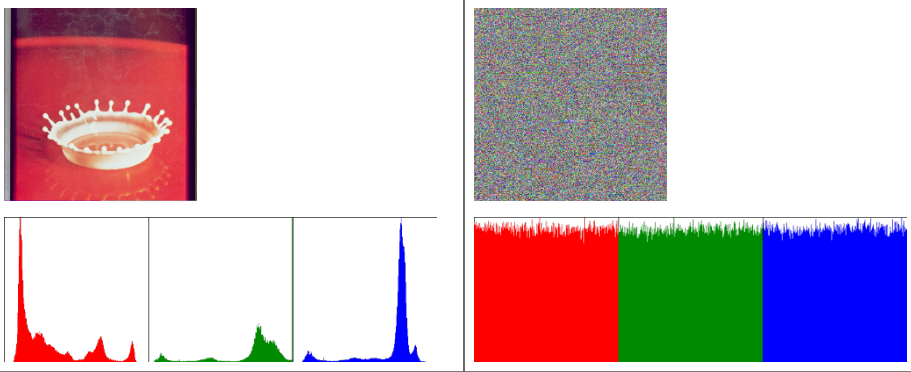
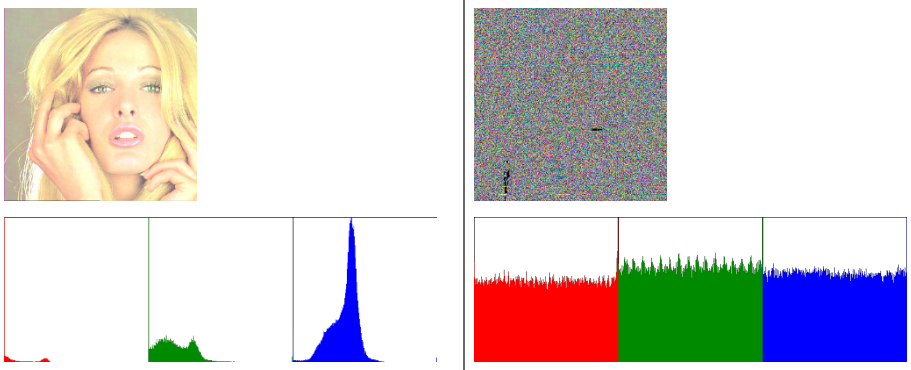
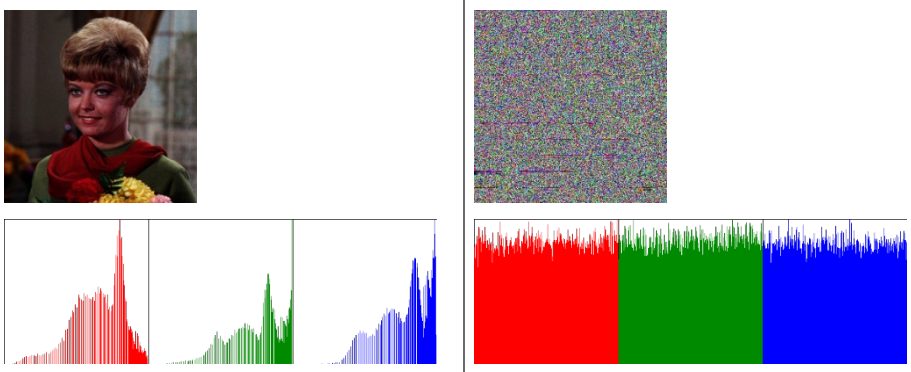
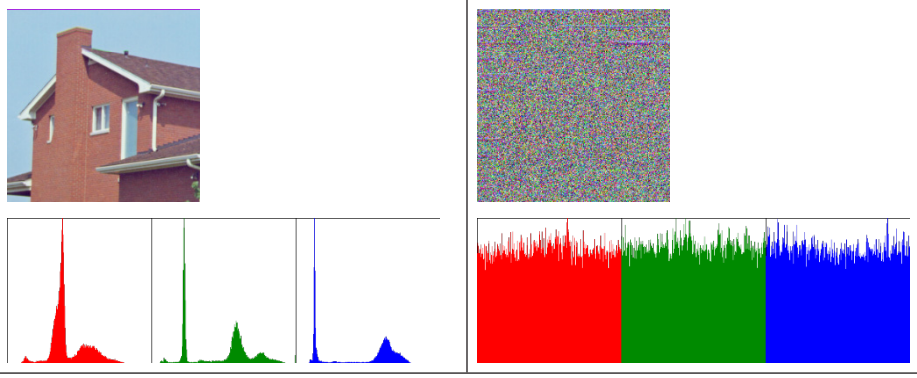
4.2 Proposed Multiple Secret Image Sharing Evaluation

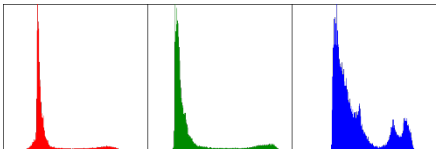
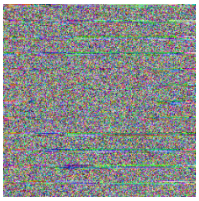
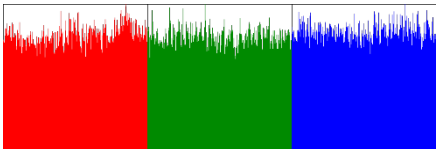
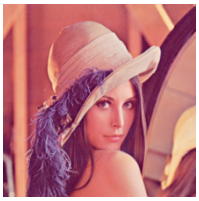
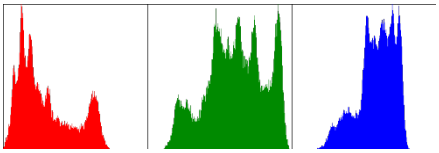
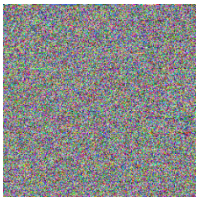
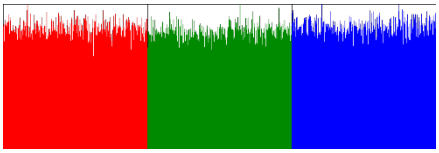
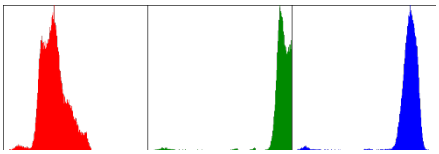
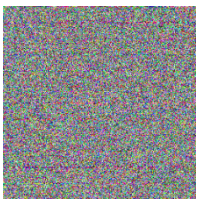
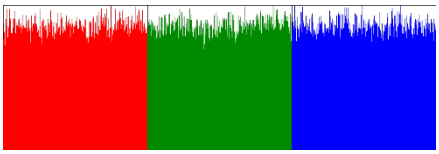
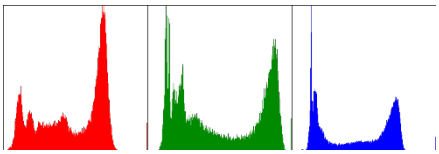
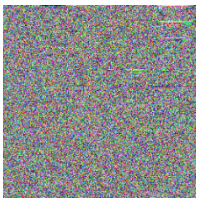
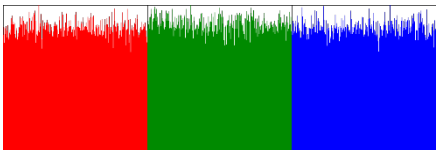
The multi-image sharing is tested. Suppose the Baboon, Barbara and jet plan images are shared together with a (3,5)-threshold model. The proposed multiple secret image sharing coding and revealing are shown in Figure 7 and 8 respectively.

Table 1. Images and its Histogram before and after Diffusing.

Name / size	Image and Histogram before diffusing	Image and Histogram after diffusing
Baboon 512		
Barbara 512		
Flowers 512		
Jet plane 512		

Name / size	Image and Histogram before diffusing	Image and Histogram after diffusing
Lena 512		
Lichtenstein 512		
Peppers 512		
Sailboat 512		

Name / size	Image and Histogram before diffusing	Image and Histogram after diffusing
Splash 512		
Tiffany 512		
Girl 256		
House 256		

Name / size	Image and Histogram before diffusing	Image and Histogram after diffusing
Jelly beans 256	 	 
Lena256	 	 
Nike 256	 	 
Tree 256	 	 

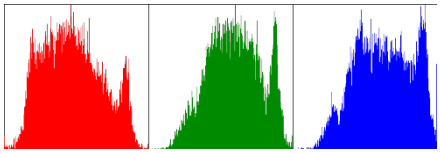
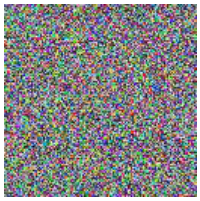
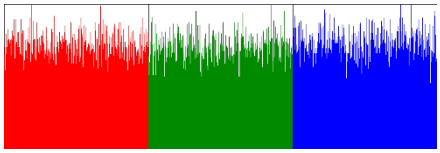
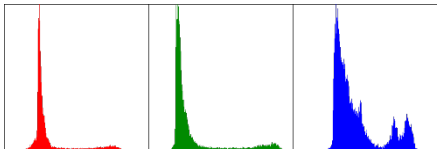
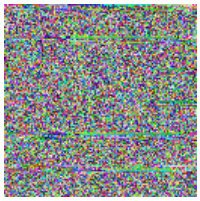
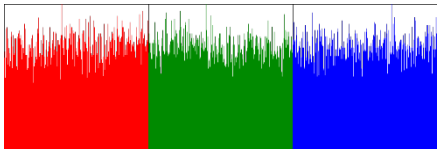
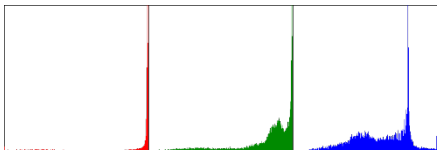
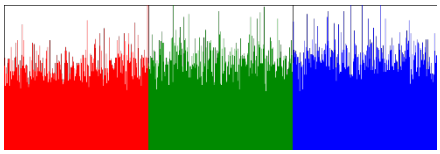
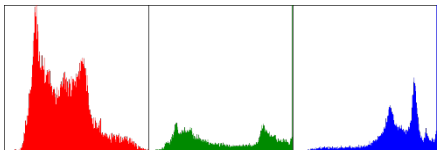
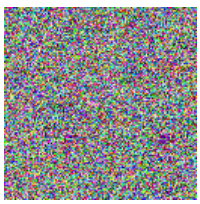
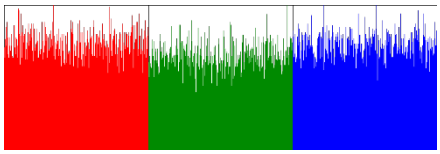
Name / size	Image and Histogram before diffusing	Image and Histogram after diffusing
Barbara 128	 	 
Jelly beans 128	 	 
Jellyfish 128	 	 
Peppers 128	 	 

Table 2. Information entropy before and after diffusing

Image	Before diffusing	After diffusing
Baboon 512	7.380397763	7.747876015
Barbara 512	7.43880727	7.747366178
Flowers 512	7.246651934	7.747126487
jet plane 512	6.740122679	7.759124501
Lena 512	7.498165162	7.745491483
Lichtenstein 512	7.452735799	7.751139183
Peppers 512	7.638737781	7.751850399
Sailboat 512	7.530937146	7.747499069
Splash 512	7.217827243	7.745470093
Tiffany 512	6.626156522	7.754093488
Girl 256	7.055156925	7.745846824
House 256	6.515938622	7.740038296
jelly beans 256	5.72280974	7.743003827
Lena256	7.473549266	7.743610928
Nike 256	5.730159646	7.742996519
Tree 256	7.346574724	7.749822551
Barbara 128	7.511322121	7.726591936
Jelly beans 128	5.751834546	7.740206144
Jellyfish 128	6.427127473	7.720252502
Peppers 128	7.649292995	7.736124326

Table 3. Cross correlation (*CC*) between the original images and their corresponding diffused images for each color band

Image	Red Cross correlation (RCC)	Green Cross correlation (GCC)	Blue Cross correlation (BCC)
Baboon 512	0.001	-0.000212	-0.0016
Barbara 512	-0.002	-0.000431	-0.0011
Flowers 512	0.0072	-0.000346	-0.00046
jet plane 512	-0.0034	0.0067	0.0077
Lena 512	-0.00076	-0.0035	0.004
Lichtenstein 512	-0.00025	0.0048	0.0055
Peppers 512	0.0017	0.0111	-0.000039
Sailboat 512	-0.00058	-0.00093	-0.00000001
Splash 512	-0.0073	0.0053	0.0021
Tiffany 512	-0.0066	-0.00012	-0.0169
Girl 256	-0.0017	0.0164	0.0049
House 256	-0.0031	-0.014	0.0165
jelly beans 256	-0.164	0.004	-0.0052
Lena256	-0.0063	0.0027	0.0066
Nike 256	0.009	-0.00028	0.0053
Tree 256	-0.0031	-0.0021	0.007
Barbara 128	0.017	-0.0049	-0.0051
Jelly beans 128	-0.0059	-0.0059	-0.0035
Jellyfish 128	0.0183	-0.006	0.0049
Peppers 128	0.0026	-0.00059	-0.0158

Table 4 shows the performance evaluation for creating shares from 3 untouched (using only secret sharing scheme) images, while Table 5 shows the time needed for the compressor to compress different images using different settings.

Table 6 shows the overall performance of the system such as the overall time, original total secrets size and the produces shares size

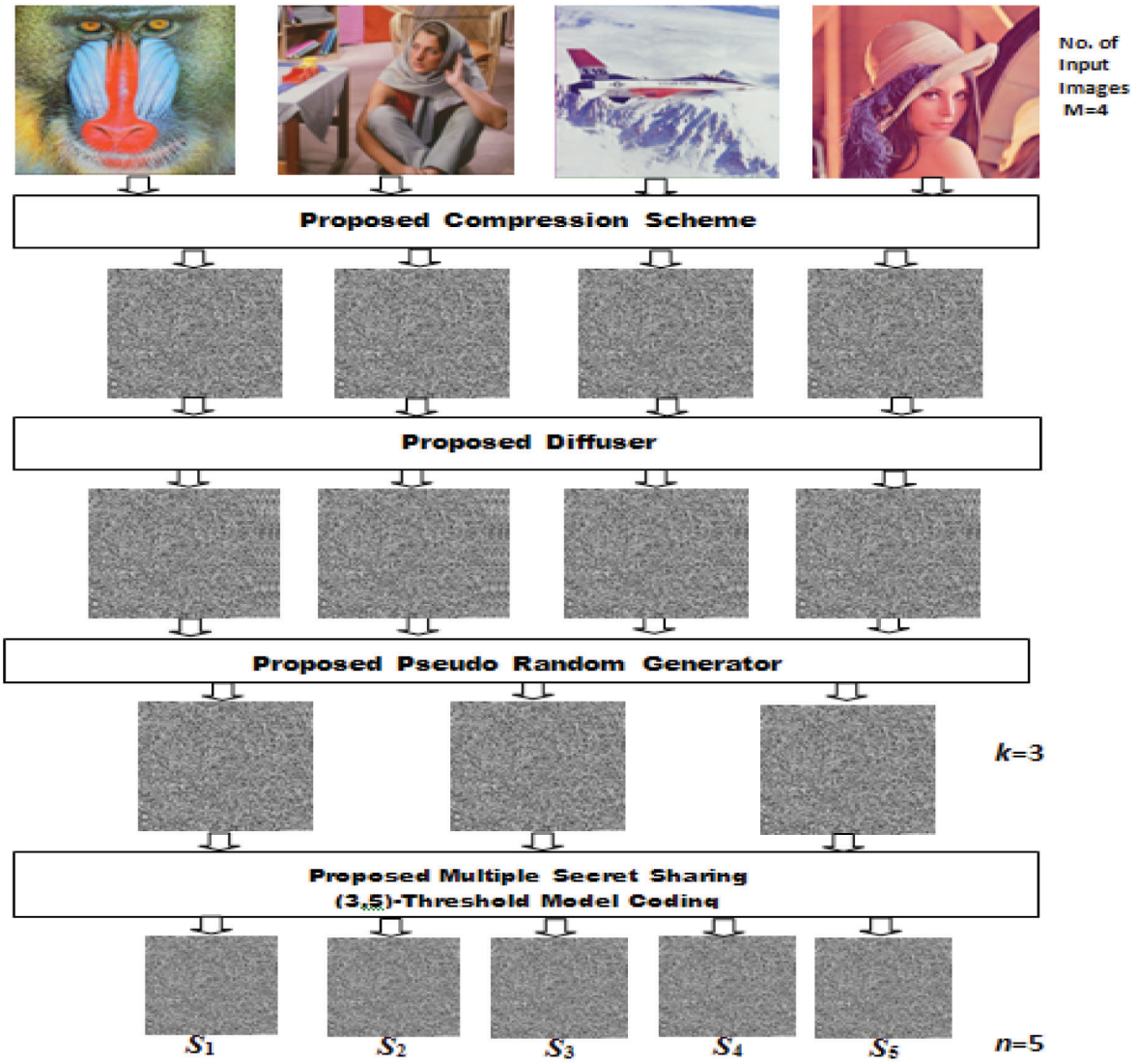


Figure 7. Example of Proposed Multiple Secret Image Sharing cod.

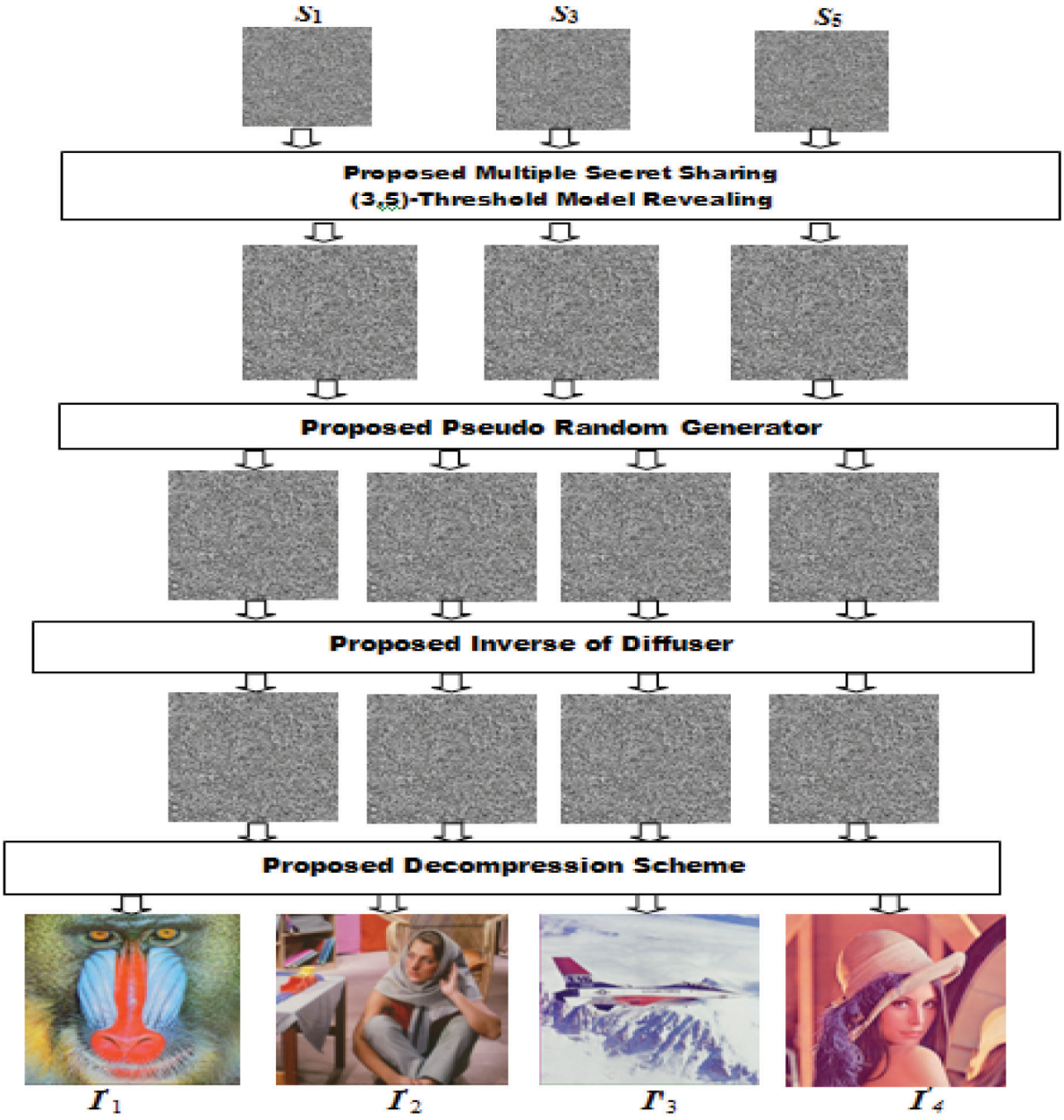


Figure 8. Example of Proposed Multiple secret Image sharing Revealing.

Table 4. Linear secret sharing system time consumption

Image size	Time in ms
512	130
256	30
128	4

Table 6. Overall system performance using m=3 secret images and generating$n$= 5 shares with k=3

Total size of images	Overall system time	Size of each share
144 KB	6575 ms	15.1 KB
576 KB	20504 ms	56 KB
1536 KB	80676 ms	203 KB

Table 5. Proposed compression time consumption using various images with different sizes and compression system configuration

Image size	Quad-tree threshold	Q0 value	α value	Time range consumption
512×512	10	1	1	9032-14791 ms
256 × 256	10	1	1	1010-1504 ms
128 × 128	10	1	1	318-380 ms
512 × 512	20	1	1	3900-7800 ms
256 × 256	20	1	1	1100-1300 ms
128 × 128	20	1	1	210-340 ms
512 × 512	10	5	5	3700-5100 ms
256 × 256	10	5	5	990-1100 ms
128 × 128	10	5	5	240-310 ms

5. Conclusions

In this paper, a (*k, n*)-threshold multiple color images secret sharing has been proposed along with proposed diffusing technique and pseudo random number generator technique.

The system starts by the uses compression technique based on Quad-tree and discrete cosine transformation to greatly reduce the sizes of the multiple secret images, followed by a diffuser to de-correlate the image data to add more security to system and neutralize statistical analysis, pseudo random number generator based on RC6 block-cipher technique was a tricky step to add a huge sense of security because it acts as a one-time pad fashion, since it generates a long random sequence and permute the multiple images data in pre-generated shares. A block cipher-based pseudo random generator is based on a re-keying approach and used the rounds of the RC6 which is a highly random block cipher algorithm. The proposed diffuser flattens the histogram of every simple tested by it, and that proves the efficiency of it to work on every possible input. The overall execution time of the system is inacceptable range. Losing up to *k*-1 shares will not expose any information about the secrets.

6. References

1. Legal Information institute. Date accessed: 17/03/2017: Available from: Crossref.
2. Konheim AG. Wiley: Hashing in Computer Science: Fifty years of slicing and dicing. 2010; p. 386. Crossref.
3. Moni N, Shamir A. Visual cryptography. Advances in cryptology. Eurocrypt 94 Proceeding LNCS. 1995; 950:1-12. PMid:8565049
4. Blakley GR. Safeguarding cryptographic keys. Proceedings of the National Computer Conference 48. 1979; p. 313-17.
5. Shamir A. How to share a secret. Communications of the ACM. 1979; 22(11):612-13. Crossref.
6. Thien CC, Lin JC. Secret image sharing. Computers and graphics. 2002; 26:765-70. Crossref.
7. Lin CC, Tsai WH. Secret image sharing with capability of share data reduction. Optical Engineering. 2004; 42(8):1377-85.
8. Wang RZ, Su CH. Secret image sharing with smaller shadow images. Pattern recognition letter. 2006; 27(6):551-55. Crossref.
9. Lin CC, Zhang WX. Secret sharing scheme with non-expansible shadow size for color images. 8th international conference on intelligent systems design and applications, Kaohsiung. 2008; 3:302-07.
10. Luo H, Yu FX, Li H, Huang ZL. Color image encryption based on secret sharing and iteration. Information technology journal. 2010; 9 (3):446-52.
11. Nerella SK, Varma K, Chaganti GR. Securing images using colour visual cryptography and wavelets. International journal of advanced research in computer science and software engineering. 2012; 2(3):164-68.
12. Wu KS. A secret image sharing scheme for light images. EURASIP journal on advances in signal processing. 2013; 49:1-5. Crossref.
13. Hashim A, George EL. Secret Image Sharing Based on Wavelet Transform. International Conference on Information Technology in Signal and Image Processing. 2013 October; p. 18-19.
14. Hashim AT, Radeef ZM. Correlated Block Quad-Tree Segmented and DCT based Scheme for Color Image Compression. Indian Journal of Science and Technology. 2016 July; 9(26):1-8.