Implementation of Exclusive-OR Algorithm

Author

Shashank[1], Vinay Yadav[2]

[1]*(Research Scholar (M. Tech)/Department of CSE/UPTU, Lucknow)*
[2]*(Asst. Professor/Department of CSE/SR Group of Institutions, Lucknow/India)*

Abstract : *Image encryption is a technique which provides security to images by converting original image to another image which is difficult to understand. In the ever-increasing growth of multimedia applications, security is an important issue in communication and storage of images. Encryption is one the ways to ensure security. Nobody could get to know the content without a key for decryption. All those processes generate a different form of that data. The unencrypted data is referred to as the plaintext and the encrypted data as the cipher text, which is representation of the original data in a difference form. Key-based algorithms use an Encryption key to encrypt the message. One simple and good way to encrypt data is through rotation of bits or sometimes called bit shifting. But, rotation of bits is more advanced than simple bit shifting. In rotation of bits operation, the bits are moved, or shifted, to the left or to the right. The different kinds of shifts typically differ in what they do with the bits.*

Keywords : *Encryption, Decryption, Exclusive-OR (XOR), Gray scale image, Cipher image, Scrambled bit plane*

1. Introduction: The exclusive or operation - a logical function applied to binary bits, like AND, OR, and NOT - is a fundamental encryption technique. It is often used in stream ciphers, which are widely used in web browsers when connecting to secure web servers. When used properly, this technique provides strong protection. In fact, it is the basis for the one-time pad, the only provably uncrackable encryption. However, this protection is easily eroded if the cipher is not used correctly. XOR is a trivial operation for computer logic to perform show the table 1. The operation often appears as a built-in machine instruction so that software can perform it in a single machine operation.

2. Exclusive-OR (XOR) Operation

The following table shows how the XOR operation transforms individual bits. Let A be a bit from the plain text message, and B be a bit from the key. The ⊕ column shows the resulting bit.

A	B	⊕
0	0	0
0	1	1
1	0	1
1	1	0

Table.1 shows how the XOR operation transforms individual bits.

If A wants to send a secret message to B, it takes the sequence of bits in the message (the plain text) and a sequence of bits known only by it and B - the key. To encrypt, she combines the plain text and the key, bit by bit, using XOR. In a one-time pad, A and B must use a different set of secret, randomly generated bits for every message they exchange.

In a stream cipher, A and B share a much smaller number of secret bits and use them to generate a long, hard-to-guess sequence of bits. The stream cipher relies on a cryptographic algorithm to generate that long sequence from a small, shared secret. This generated sequence is then combined with the message using XOR.

For Example: Below we have the image message "Rose image" embedded in a 128 by128-bit image. For a key, we have collected a 128 by 128 matrix of random bits. We will combine the two matrices using XOR.

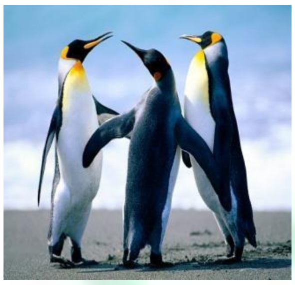
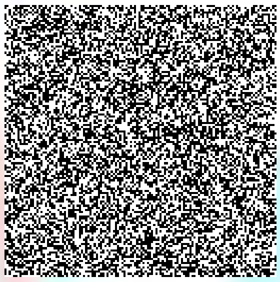
	$\oplus$	
Image Message "bird Image"	***XOR***	***"Encryption Key"***

Figure 3 - Send bird Image Message Encrypted Key by using XOR Operation.

When we apply XOR bit-by-bit to the two matrices, we get the following 128 by 128 matrix of encrypted bits.

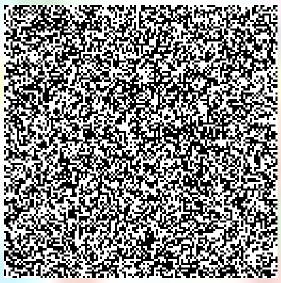

Encrypted " bird Image "

Figure 3 - Encrypted Rose Image.

To decrypt the message, we simply take the encrypted message and compute XOR with the encryption key, bit-by-bit. This yields the original image "Rose image" message.

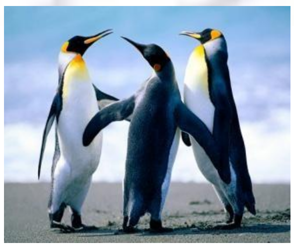

Decrypted " bird Image "

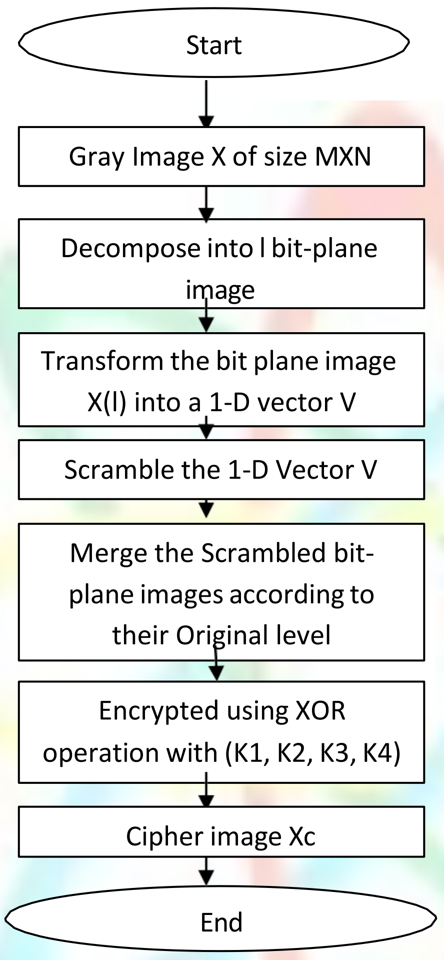
Figure 4 - Decrypted Original bird Image.

3. Types of XOR Algorithm

There are two step of XOR algorithm:

3.1) XOR Encryption algorithm at sender side
***3.2)** XOR Decryption algorithm at receiver side*

3.1) Flow Chart of XOR encryption algorithm


Start
Gray Image X of size MXN
Decompose into l bit-plane image
Transform the bit plane image X(l) into a 1-D vector V
Scramble the 1-D Vector V
Merge the Scrambled bit-plane images according to their Original level
Encrypted using XOR operation with (K1, K2, K3, K4)
Cipher image Xc
End

Algorithm 1: Encryption Algorithm at Sender Side

Image encryption process starts with selecting a gray scale image X of $M \times N$ pixel size with L bit per pixel .which is to be converted into encrypted form before transmitting to the other end.

Input: A Gray scale image X.
Output: Cipher image X_C.

1. Input a gray scale image X of M × N size with L bits per pixel.
2. Then we decomposed a gray image into *l* bit-plane images.

$$X^{(l)} = B^{(l)}(X) \ldots\ldots (1)$$

If *X (m, n)* is a pixel located at *(m, n)*, then the *l*th bit of *X (m,n)* is:

$$\hat{}^{(l)}(m, n) = B^{(l)} = \left\{ \left(x(m,n)/2^{(l)}\right) \bmod 2 \right\}$$

3. We transform the bit-plane image $X^{(l)}$ into a 1-D vector $V(l)$

4. Then it uses a random natural number generator and Chooses a couple of different seeds to produce two random sequences R_S and R_D with the same length as V. and scrambles the 1-D vector V.

$$V(R_S(i)) \leftrightarrow V(R_D(i)) i = 0,1,\ldots,(M \times N - 1) \ldots\ldots\ldots (3)$$

We merged the scrambled bit-plane images according to their original levels on bit-planes and gained a Transformed image X_T.

$$X_T = \sum_{l=0}^{L-1} B^{-1(l)}(X^{(l)})$$

For a pixel at position (m,n) ,we also have

$$X_T(m,n) = \sum_{l=0}^{L-1} 2^{(l)} \times X^{(l)}(m,n) \ldots\ldots (4)$$

5. The transformed image then divided into 2 pixels × 2 Pixels blocks.

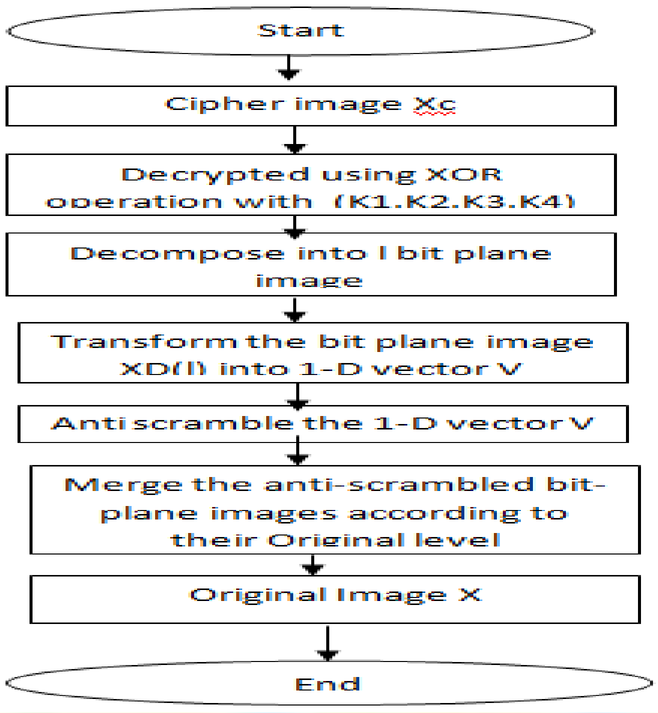
6. Each block $B_{i,j}$ of X_T is encrypted using XOR operation by four 8-bit keys (K_1,K_2,K_3,K_4).

$$P'_{1.1}=P_{1.1} \oplus K_1$$
$$P'_{1.2}=P_{1.2} \oplus K2 \ldots\ldots\ldots\ldots (5)$$
$$P'_{2.1}=P_{2.1} \oplus K_3$$
$$P'_{2.2}=P_{2.2} \oplus K_4$$

Where $P_{i,j}$ is the pixel value at ith and jth location in block resulted image called by cipher image X_C is ready to be sent to receiver site.

7. End.

3.2) Flow Chart of XOR Decryption algorithm:


Start
Cipher image Xc
Decrypted using XOR operation with (K1.K2.K3.K4)
Decompose into l bit plane image
Transform the bit plane image XD(l) into 1-D vector V
Anti scramble the 1-D vector V
Merge the anti-scrambled bit-plane images according to their Original level
Original Image X
End

Algorithm 2: Decryption Algorithm at Receiver Side:
The input is a gray scale encrypted image XC of $M \times N$ pixel size with L bit per pixel.
Input: Cipher image XC.
Output: A Gray scale image X.

1. For Decryption, the cipher image XC is first divided into 2 pixels × 2 pixels blocks.

2. Each pixel of every block is decrypted using XOR

Operation with keys (K1, K2, K3, K4)

Decrypt P'1.1 as P1.1=P'1.1⊕ K1
Decrypt P'1.2 as P1.2=P'1.2 ⊕K2 ....**(6)**
Decrypt P'2.1 as P2.1=P'2.1 ⊕ K3
Decrypt P'2.2 as P2.2=P'2.2 ⊕ K4

3. The decrypted image XD is then decomposed again into bit-plane images by using. The formula used in eq. (2).

4. We then transform the bit-plane image XD(*l*) into a 1-D vector V(*l*) .Then we use again random natural number generator and use the same a couple of seeds used at encryption time to produce same random sequences R_S and R_D with the same length as V. and ant scrambles the 1-D vector V .

$$V^{(l)}\left(R_S(i)\right) \leftrightarrow V^{(l)}\left(R_D(i)\right) i=0,1,\ldots.\left(M \times N-1\right);\ l=0,1,\ldots.L-1$$

5. Lastly, we merged the ant scrambled bit-plane images according to their original levels on bit-planes and gained an Original image

6. End

4. Proposed encryption method and histogram for City image

In the experiment, we do scrambling by using Affine's transform with the help of Affine key and then apply X-OR operation on scrambled image for the gray image City as shown in Figure 3(a), which is of size 256×256. The results are shown as in Figure 3(b). Figure 3(c) is the histogram of original image of City. Figure 3(d) is the histogram of the result image scrambled by the proposed method.

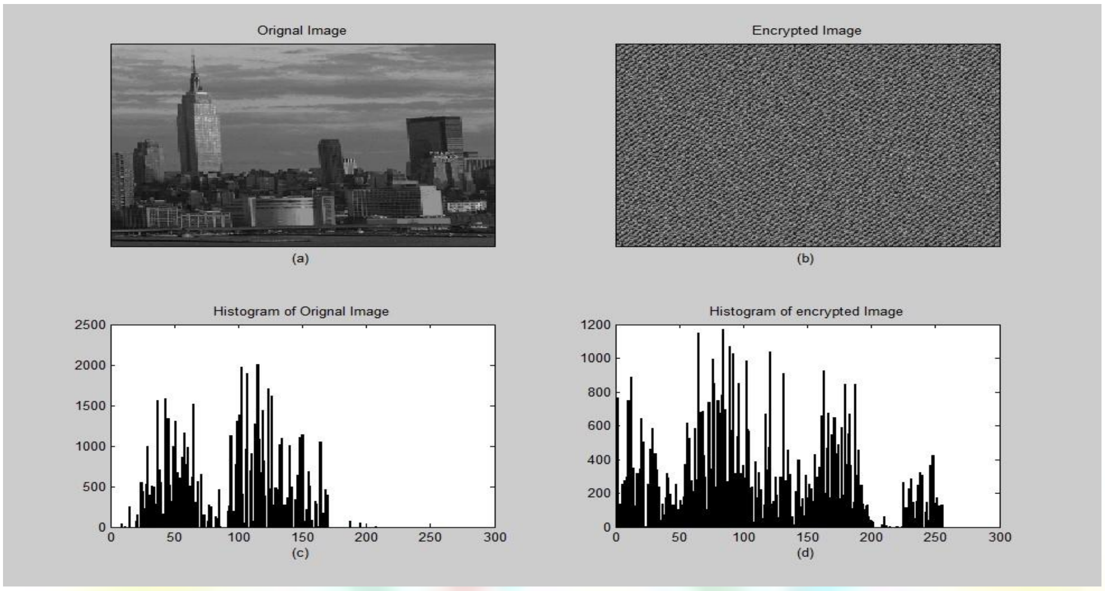


Figure 3 - depicts the image encryption system for City image. Here (a) shows input Original image (b) Encrypted image. (c) Histogram of original image (d) Histogram of encryption image

5. Conclusion:

We proposed a symmetric key image encryption technique that first scramble the locations of the pixels using 4 8-bit sub keys and then encrypt the pixel values by XOR the selected 8-bit key. The scrambling operation is done using Affine transformation cipher techniques that breaks the correlations of the neighboring pixels and make the image unidentifiable. The XOR operation then change the pixel values making the image very meaningless. The encryption and decryption process are simple enough to be carried out on any large sized image or video files, but provides enough security. The proposed encryption method in this study has been tested on different gray images of 256*256 and showed good results.

6. Future Work

The future work can be summarized as follows:

a) Implementing image encryption with a fractal approach.
b) Efficient encryption of large block size of data

References

[1]. Aloka Sinha , Kehar Singh, "A Technique for Image Encryption using Digital Signature", Optics Communications,Vol-218 ,229-234 ,2003.

[2]. Amitava Nag, Jyoti Prakash Singh, Srabani Khan, Saswati Ghosh, Sushanta Biswas, D. Sarkar Partha Pratim Sarkar, ―Image Encryption Using Affine Transform and XOR Operation‖,International Conference on Signal Processing, Communication, Computing and Networking Technologies (ICSCCN 2011).

[3]. W. Xiao, J. Zhang and W. Wu, " A Watermarking Algorithm Based on Chaotic Encryption" , Proceedings of IEEE TENCON, pp. 545-548, 2002.
[4]. S. Li and X. Zheng, "On The Security of An Image Encryption Method", In Proceedings IEEE Int. Conference on Image Processing (ICIP), Vol. 2, pp. 925 928,2002.
[5]. John Justin M, Manimurugan S, "A Survey on Various Encryption Techniques", (IJSCE) ISSN: 2231-2307, Volume-2, Issue-1, March 2012.