

## Separable Reversible Data Hiding For Enhanced Image Security

**Peddinti Vamsi**

M.Tech (CST),  
GITAM School of Technology,  
Hyderabad.

**Dr.S.Phani Kumar, M.Tech, Ph.D**

Professor & HOD of CSE,  
GITAM School of Technology,  
Hyderabad.

### ABSTRACT:

Advancement in the technology leading to conflicts while transmitting the data. Security must be there for sending the data. Separable Reversible Data Hiding is one the advanced technique to send the data using an image secretly by embedding data into an image and encryption it and recovering the image losslessly after the extraction of data. In this paper a new way to enhance the security of an image has been done.

### 1.INTRODUCTION:

Data and information security have always been a vibrant area of research. In the area of data security various traditional approaches like Cryptography, Steganography, and Data Hiding can be used. Cryptography refers to the study of mathematical techniques and related aspects of Information Security like data confidentiality, data integrity, and of data authentication. In cryptography a plain message is encrypted into cipher text and that might look like a meaningless jumble of character whereas in case of steganography, the plain message is hidden inside a medium that looks quite normal and does not provide any reason for suspecting the existence of a hidden message. Such an image is called as stego-image. Data hiding conceals the existence of secret information while cryptography protects the content of messages. More and more attention is paid to reversible data hiding in encrypted images. The hidden data in the cover image may be any text related to the image such as authentication data or author information [2]. Reversible data hiding [6] represents a technique where the data is embedded in the host media and at the receiving end the secret data and also the host media will be recovered loss less.

### 2. EXISTING SYSTEM:

Assume we are considering a grayscale image of size  $m \times n$ . First scan the image into a matrix sized  $m \times n$ . For a grayscale image, the pixel values fall in the range [0, 255].

We use the chaotic based permutation method [1] in encrypt the image. A sequence of pseudorandom numbers are generated by using the logistic function

$$X_{p+1} = \mu * X_p * (1 - X_p) \quad (1)$$

Where the value of  $\mu$  lies in the range [0, 4]. Usually initial seed of the logistic function lies between [0,1] but instead of directly specifying here algorithm uses 120 bit encryption key and calculate the initial seed from the key using (2)&(3).

$$\text{Key} = \{K_1, K_2, K_3 \dots K_{15}\} \quad (2)$$

$$X_0 = (K_1 \text{ XOR } K_2 \text{ XOR } \dots \text{ XOR } K_8 + \text{Sum}(K_1, K_2, \dots, K_{15})) / 2^{12} \quad (3)$$

Where  $X_0$  is the initial seed to the logistic function. Value of  $\mu$  is chosen as the optimum one from the literatures. Least complexity, sensitiveness to the initial seed values and desirable chaotic behavior in the range [3.57, 4] of  $\mu$  value makes logistic function as the best choice for permutation. Generate  $N = mn$  pseudorandom numbers using (1) and store them in a vector  $C$  (Chaotic). Sort the vector  $C$  in ascending order. Note the position changes during sorting to generate a location map  $LM$  (Location Map). Location map is such that,  $LM(1)$  is the original index of smallest value,  $LM(2)$  that of second smallest and so on. Arrange pixels of the image into a vector  $P$  (Picture). Rearrange  $P$  according to the location map  $LM$ . Reshape vector  $P$  into matrix sized  $m \times n$  to get the encrypted image.

### Data Hiding:

As permutation based encryption do not alter the histogram of original image, histogram modification based method can be used to hide data into the encrypted image. A clear idea about histogram and image organization is the prerequisite for the better understanding of the concept. The method follows:

- Pseudo randomly permute the encrypted image pixels using data hiding key. Adopt the same mechanism described for generating chaotic sequence from encryption key.

- First H (Header) pixels of permuted image are used to hide parameters of data hiding. The parameters considered are MAX-MIN pixel values & number of MAX-MIN pairs. Let there are S MAX-MIN pairs. Then, the number of pixels needed to hide the parameters are given by

$$H = 2 * S + 1 * 8(4)$$

- Generate histogram of the remaining N-H pixels and find the maximum (MAX) and minimum (MIN) points. A MAX point is the grayscale value having the maximum number of pixels in the image. A MIN point is the grayscale value having minimum number of pixels in the image. Data hiding with a single pair only explained here. From the observations in most of the cases, MAX<MIN. Data are embedded into pixels with grayscale value equal to MAX. The pixel positions with grayscale value equal to MIN are stored as overhead information and will be hidden into the image along with pure data.

- The N-H pixels used for generating the histogram are scanned in the sequential order. The grayscale values of those pixels in the range [MAX+1, MIN-1] are incremented by 1, by leaving the MAX+1 slot empty. This process is called histogram shifting.

- Let there be p pixels corresponding to the grayscale value MAX. These p pixels are used to hide the data. Data to be embedded is converted to binary bit stream. The N- H pixels are scanned in the sequential order. Whenever a pixel with grayscale value MAX is encountered, check the bit to be embedded. If the corresponding bit to be embedded into that pixel is “1”, the pixel value is incremented by 1. If the bit to be embedded is “0”, the pixel value remains unchanged.

- The parameters of data hiding namely, number of MAX-MIN pairs and MAX-MIN pixel values, are hidden into the image by replacing the LSB of first H pixels. The original H LSBs are also hidden as overhead information along with pure data.

- Perform inverse permutation on the image. The above steps complete data hiding process. It is observed that data hiding capacity of the method is (P-O-H) bits, where O is the overhead information due to MIN positions.

### Data Extraction and Image Recovery:

When the receiver has only data hiding key and encrypted image containing hidden data, he must be able to generate an image similar to the original image, but should not be able to read the hidden data. The receiver can decrypt the encrypted image containing hidden data by using the encryption key only. For this, he generates the same chaotic sequence as in encryption phase using and the initial conditions. The logistic map is generated by sorting the sequence in ascending order. Using the logistic map, pixels of the encrypted image are rearranged to their original position to get the decrypted image. The decryption process can bring back all the pixels to their original position. Only distortion in the decrypted image is a difference of 1 in grayscale value for those pixels used for data hiding. The lower bound of Peak Signal to Noise Ratio (PSNR) of this decrypted image can be proved to be larger than 48 dB as follows. It is observed that the grayscale value of pixels between MAX and MIN will either be incremented or decremented by 1 during data hiding. In the worst case, the value of every pixel differs by a value of 1 from their original value. Thus, the Mean Square Error (MSE) of worst case is 1 and the lower bound of PSNR of the decrypted image containing hidden data is given by

$$\begin{aligned} \text{PSNR} &= 10 \log_{10} (255 \times 255) / \text{MSE} \\ &= 10 \log_{10} 255^2 \\ &= 48.13 \text{ dB} \end{aligned}$$

When the receiver has both keys, he first extracts the hidden data using data hiding key as in case (i). This will recover the original value of distorted pixels due to data hiding. Then, the receiver decrypts the image as in case (ii). Thus, the hidden data is extracted exactly and the original image is recovered completely.

### 3. PROPOSED WORK:

#### Image encryption:

The Advanced Encryption Standard (AES) algorithm[4] consists of a set of processing steps repeated for a number of iterations called rounds. The number of rounds depends on the size of the key and the size of the data block. The number of rounds is 9 for example, if both the block and the key are 128 bits long. Given a sequence  $\{X_1, X_2, \dots, X_n\}$  of bit plaintext blocks, each  $X_i$  is encrypted with the same secret key  $k$  producing the cipher text blocks  $\{Y_1, Y_2, \dots, Y_n\}$ , as described in the scheme from

Fig. 1. To encipher a data block  $X_i$  in AES you first perform an AddRoundKey step by XORing a subkey with the block. The incoming data and the key are added together in the first AddRoundKey step. Afterwards, it follows the round operation. Each regular round operation involves four steps. In the SubBytes step, each byte of the block is replaced by its substitute in a substitution box (S-Box). In cryptography, an S-box is a basic component of symmetric key algorithms used to obscure the relationship between the plaintext and the cipher text.

The next one is the ShiftRows step where the rows are cyclically shifted over different offsets. The next step is the MixColumns, where each column is multiplied with a matrix over the Gallois Field, denoted as GF (28). The last step of the round operation is another AddRoundKey. It is a simple XOR with the actual data and the subkey for the current round.

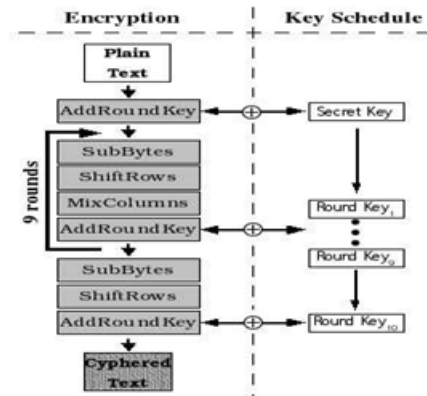
Before producing the final ciphered data  $Y_i$ , the AES performs an extra final routine that is composed of (SubBytes, ShiftRows and AddRoundKey) steps, as shown in Fig. 1. The ECB mode is actually the basic AES algorithm. With the ECB mode, each plaintext block  $X_i$  is encrypted with the same secret key  $k$  producing the cipher text block  $Y_i$

$$Y_i = E_k(X_i) \quad (1)$$

For the proposed method, the ECB mode of AES algorithm has been chosen to encrypt the images. The images are thus encrypted by blocks of 128 bits. The message is divided into blocks, and each block is encrypted separately. It is useful when you are encrypting random blocks of data.

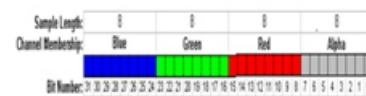
### Embedding messages in digital images:

The system for representing colour is the ARGB system-which stores pixel data in the form of red, green, blue and alpha (transparency). Under ARGB system, first 8 bits (0 to 7) of the pixel belong to Alpha value or the transparency value. The second 8 bits (8 to 15) represent red colour, third 8 bits (16 to 23) represent green colour and the last 8 bits (24 to 31) represent blue colour.



**Fig.1 The scheme of the AES algorithm containing 9 rounds of processing steps.**

Now that the pixel level organization of ARGB system is clear, we should understand that the maximum value for each parameter of ARGB system is 28 i.e., 256. Maximum value is shown when the pixel stores one in all bits. If a change is made to the value at the least significant bit, i.e. the bit location 0 for alpha value, 8 for red value, 16 for green value and 24 for blue value, the impact is likely to be 0.39% .Since the change in original value is very low, we might use the least significant bit of any or all the four ARGB bytes for storing the information we wish to transmit incognito. Before embedding the message, the length of the message should be written into the image. This will exclude the appearance of junk values in the decoded message.



After extracting bit number 0 from the first 32 pixels, the bits should be neatly arranged inside an integer variable to know the length of message embedded into the image. Pixels following the 32nd pixel store the bits needed for reconstructing the byte value needed to create the original string. Hence, an image with 1 million pixels (or 1 Mega Pixel) might be able to store a message containing a maximum of 1,24,996 characters.  $((10,00,000-32)/8 = 1,24,996)$ . Although 1 Mega Pixel image is considered a low resolution image, it could store a lot of characters in the form of a text message. Maximum size of message that could be embedded in an image at the rate of 1 bit from each pixel can be calculated using the relation







$$n = \frac{(P-32)}{8}$$



If we increase the storage locations for message to the least significant bit of all the four components of ARGB system (pixel numbers 0, 8, 16 and 24), the storage capacity increases to

$$n = \frac{(4P - 8)}{8}$$

Here n is the maximum length of the message and P is the number of pixels.

Original image	Data embedded image	PSNR
		54.75
		96.64
		56.15

## CONCLUSION:

In this project, a scheme for separable reversible data hiding in encrypted image is proposed, which consists of image encryption, data embedding and data-extraction/image recovery phases. In the first phase, the content owner encrypts the original uncompressed image using an encryption key and embeds data using embedding key. With an encrypted image containing additional data, the receiver may extract the additional data using only the data-hiding key, or obtain an image similar to the original one using only the encryption key.

When the receiver has both of the keys, he can extract the additional data and recover the original content without any error by exploiting the spatial correlation in natural image if the amount of additional data is not too large. Still more investigation and research should be done for increasing the security and we may implement this procedure via network and we can extend this process of encryption and decryption on different types of data i.e. on audios, videos etc.

## REFERENCES:

- [1] Arun K Mohan, Saranya M R, An Algorithm for Enhanced Image Security with Reversible Data Hiding, IEEE conference, 978-1-4799-6629-5, 2014.
- [2] Rintu Jose, Gincy Abraham, "A Separable Reversible Data Hiding in Encrypted Image with Improved Performance", International Conference on Microelectronics, Communication and Renewable Energy, ICMiCR-2013.
- [3] A. Karnik, "Performance of TCP congestion control with rate feedback: TCP/ABR and rate adaptive TCP/IP," M. Eng. thesis, Indian Institute of Science, Bangalore, India, Jan. 1999.
- [4] William Puech, Marc Chaumont, Olivier Strauss. A Reversible Data Hiding Method for Encrypted Images. IS&T/SPIE Electronic Imaging 2008- Security, Forensics, Steganography, and Watermarking of Multimedia Contents, San Jose, CA, United States. SPIE/IS&T, 6819, pp.N/A, 2008
- [5] Reversible data hiding, Zhicheng Ni, Yun-qing Shi, Nirwan Ansari, Wei Su, IEEE Trans. Circuits Syst. Video Technol 2006.