IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 2, Ver. VII (Mar-Apr. 2014), PP 26-31
www.iosrjournals.org

A Secure Data Transmission by Embedding Marked Encrypted Image on Cloak Image

Fameela. K. A, Reshma. V.K
Dept. of Computer Science and Engineering Nehru College of engineering and research centre Pampady, Thrissur
Dept. of Computer Science and Engineering Nehru College of engineering and research centre Pampady, Thrissur

Abstract: *Data hiding is the method of embedding confidential information into a set of host data such as photograph, television signal or any other identity card Reversible data hiding in images is a technique, by which the original cover can be losslessly recovered after the embedded message is extracted. To ensure security, we propose a novel method in which the marked encrypted image is embedded on a cloak image.*
Keywords: *Reversible data hiding; Image encryption; cloak image*

I. Introduction

Steganography is the art of concealing a message, image, or file within another message, image or file. Digital steganography and watermarking are the two kinds of data hiding technology to provide hidden communication and authentication. The goal of steganography is to hide a secret message inside harmless medium in such a way that it is not possible even to detect that there is a secret message. The medium for data hiding is also called as cover, host and carrier. Reversible steganography or watermarking can restore the original carrier without any distortion or with ignorable distortion after the extraction of hidden data. So reversible data hiding is now getting popular.

Reversible data hiding (RDH) in images is a technique, by which the original cover can be losslessly recovered after the embedded message is extracted. This important technique is widely used in medical imagery, military imagery and law forensics, where no distortion of the original cover is allowed. The block diagram of RDH is shown in Fig.1.a

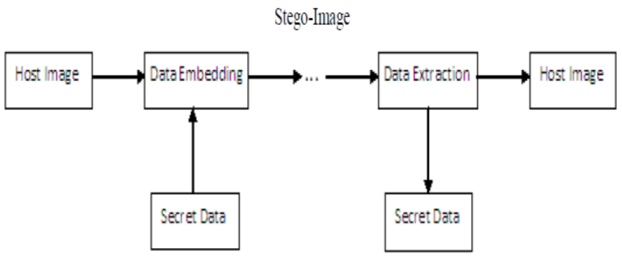


Figure 1.a

With regard to providing confidentiality for images, encryption is an effective and popular means as it converts the original and meaningful content to incomprehensible one. Some promising applications can be generated if RDH can be applied to encrypted images. Suppose a medical image database stored in a data center, notations can be embedded into the encrypted version of a medical image through a RDH technique by a server in the data center. The server can manage the image or verify its integrity by using the notations without having the knowledge of the original content. This will protect the patient's privacy. At the same time, a doctor can decrypt and restore the image for further diagnosing by using the cryptographic key.

It is not possible even to detect that there is a secret message is hidden inside the cover or cloak image. In the present paper we propose a novel method for ensuring security for the confidential information by double embedding. In the proposed paper, we first embed the secret information on to the encrypted image and then the marked encrypted image will be again embedded into a cloak image.

II. Previous Arts

The methods proposed in [ ]-[ ] can be summarized into two frameworks

- Vacating room after encryption

- Reserve room before encryption

In the first framework, vacate room after encryption (VRAE), a content owner first encrypts the original image using a standard cipher with an encryption key. After producing the encrypted image, the content owner hands over it to a data hider (e.g., a database manager) and the data hider can embed some auxiliary data into the encrypted image by losslessly vacating some room according to a data hiding key. Then a receiver, maybe the content owner himself or an authorized third party can extract the embedded data with the data hiding

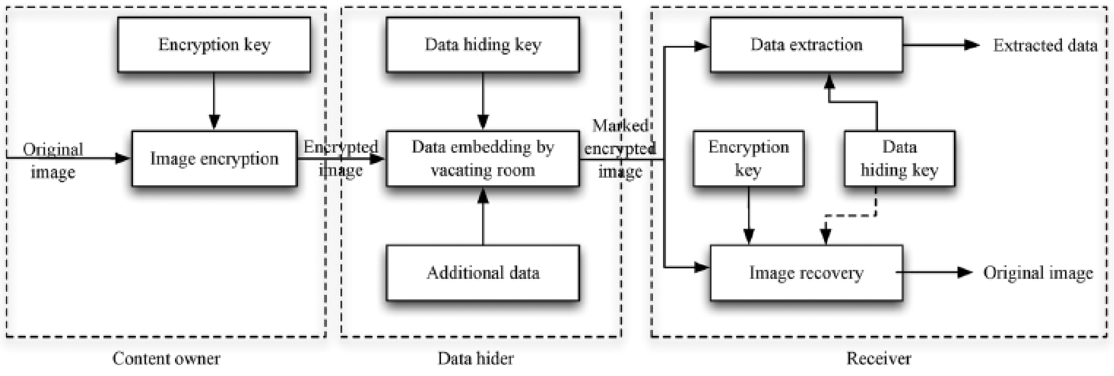


Figure 2.a

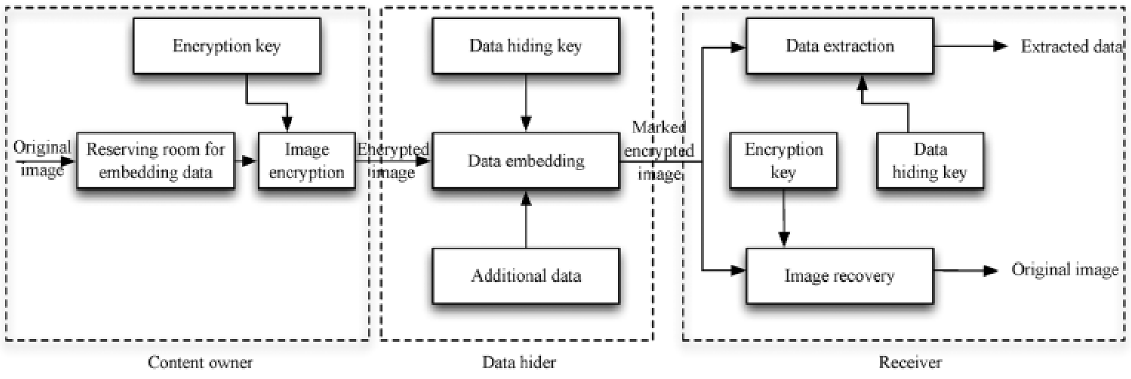


Figure 2.b

key and further recover the original image from the encrypted version according to the encryption key. This framework is illustrated in figure 2.a.

In the second framework, reserve room before encryption (RRBE), the content owner first reserve enough space on original image and then converts the image into its encrypted version with the encryption key. Now, the data embed ding process in encrypted images is inherently reversible for the data hider only needs to accommodate data into the spare space previous emptied out. The data extraction and image recovery are identical to that of Framework VRAE. Obviously, standard RDH algorithms are the ideal operator for reserving room before encryption and can be easily applied to Framework RRBE to achieve better performance compared with techniques from Framework VRAE. This is because in this new framework, a customary idea is followed in which the redundant image content is losslessly compressed and then encrypts it with respect to protecting privacy. This framework is illustrated in figure 2.b [1]

III. Proposed Method

The proposed method uses the RRBE framework for RDH. In this method we first empty out room by embedding LSB's of some pixels into other pixels with a traditional RDH method and then encrypt the image, so the positions of these LSB's in the encrypted image can be used to embed data. After embedding data on encrypted image, the marked encrypted image will be again embedded on a cloak image for more security.
The proposed method consists of five stages

- Generation of encrypted image
- Data hiding in encrypted image
- Marked encrypted image hiding in a cloak image
- Data extraction
- Image recovery

This method is illustrated in figure 3.a. Note that the reserving operation we adopt in the proposed method is a traditional RDH approach.

A. Generation of Encrypted Image

The construction of encrypted image can be done in three steps

- Image partition
- Self-reversible embedding
- Image encryption

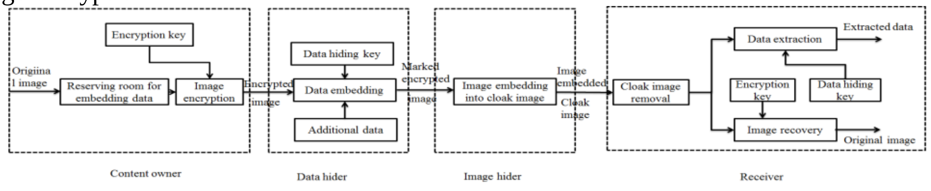


Figure 3.a

The first step is to divide the image into two parts A and B; by using a standard RDH algorithm, LSB's of A were reversibly embedded into B. Now LSB's of A can be used for accommodating messages. Finally, the rearranged image will be encrypted to its final version.

1) *Image partition*

Since we are using a traditional RDH method for reserving room before encryption, image partition is used to find a smoother area B on which the standard RDH algorithms can achieve better performance. Assume the original image is an 8 bits grey scale image with size MxN and pixels $C_{i,j}$. First, the content owner extracts from the original image, along the rows, several overlapping blocks whose number is determined by the size of to-be-embedded messages, denoted by l. In detail, every block consists of m rows, where m=l/n, and the number of blocks can be computed through n=M-m+1. An important point here is that each block is overlapped by pervious and/or sub sequential blocks along the rows. For each block, define a function to measure its first order smoothness

$$f = \sum_{u=2}^{m} \sum_{v=2}^{N-1} \left| c_{u,v} - \frac{c_{u-1,v} + c_{u+1,v} + c_{u,v-1} + c_{u,v+1}}{4} \right|$$

Higher f relates to blocks which contain relatively more complex textures. The content owner, therefore, selects the particular block with the highest f to be A , and puts it to the front of the image concatenated by the rest part with fewer textured areas, as shown in figure 3.b. .

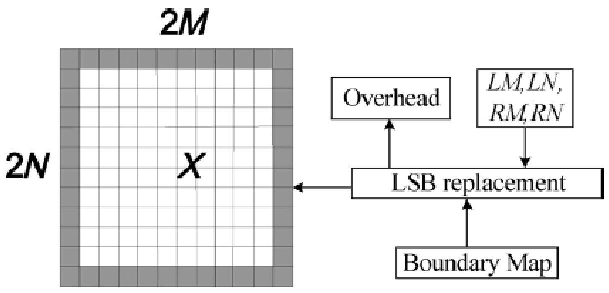


Figure 3.b

2) *Self-reversible embedding*

Self-reversible embedding is used to embed the LSB planes of A into B by using traditional RDH algorithms. Pixels in the rest of image B are first categorized into two sets: white pixels with its indices i and j satisfying (i+ j) mod2=0 and black pixels whose indices meet (i+ j) mod2=1, as shown in Figure. Then, each white pixel,$B_{i,j}$, is estimated by the interpolation value obtained with the four black pixels surrounding it as follows

$$B'_{i,j} = w_1 B_{i-1,j} + w_2 B_{i+1,j} + w_3 B_{i,j-1} + W_4 B_{i,j+1}$$

Where w_i is the weight. The estimating error is calculated via $e_{i,j} = B_{i,j} - B'_{i,j}$ and then some data can be embedded into the estimating error sequence with histogram shift.
By bidirectional histogram shift, some messages can be embedded on each error sequence. That is, first divide the histogram of estimating errors into two parts, i.e., the left part and the right part, and search for the highest point in each part, denoted by LM and RM, respectively. For typical images LM=-1, and RM=0. Furthermore, search for the zero point in each part, denoted by LN and RN. To embed messages into positions with an estimating error that is equal to RM, shift all error values between RM+1 and RN-1 with one step toward right, and then, we can represent the bit 0 with RM and the bit 1 with RM+1. The embedding process in the left part is similar except that the shifting direction is left, and the shift is realized by subtracting 1 from the corresponding pixel values.
After secret messages are embedded, some overhead information is needed to extract the covert information and restore the original image. Generally, the overhead information contains the following:

- The information to identify those pixels containing embedded bits;
- The information to solve the overflow/underflow problem.

In our proposed scheme, LM, LN, RM, and RN, is used to identify the pixels containing embedded bits, and exploit a boundary map, to record information on solving the overflow/underflow problem. Since overflow/underflow happens when pixels are changed from 255 to 256 or from 0 to -1 (boundary pixels), we apply additive interpolation- error expansion only to pixels valued from 1 to 254. However, ambiguities arise when non boundary pixels are changed from 1 to 0 or from 254 to 255 (pseudo boundary pixels) during the embedding process. When a boundary pixel is encountered during the extracting process, it is originally either a boundary pixel or a pseudo boundary pixel. Therefore, to find the original boundary pixels, we only need to tell whether boundary pixels in the watermarked image are genuine or pseudo. A boundary map is the right judge to distinguish between genuine and pseudo. It is a binary array with its every element corresponding to a boundary pixel in the watermarked image, 0 for genuine and 1 for pseudo. The detailed description of the embedding process is given as follows
1) Record some original LSB bits of the marginal area as overhead and add “0” to the beginning of boundary map as a label. Then, assemble overhead and watermark information to form payload W.
2) Calculate interpolation-errors e of the non-sample pixels
3) Work out the frequency of every interpolation-error and find out LM, LN, RM, and RN. Next, scan the cover-image from the beginning and start to undertake the embedding operation.
4) If $x \in \{0,255\}$, put a “0” into the boundary map and move to the next one. Else, expand e through additive expansion and work out the watermarked pixel x''. If $x'' \in \{0,255\}$, put a “1” into the boundary map.
5) For convenience, let C_1 denote the condition when W is not completely embedded, and C_2 denote the condition when the current pixel is not the end of non-sample pixels. If C_1 and C_2 are both satisfied, go to Step 4). If C_1 is satisfied but C_2 is not satisfied, record the length of the boundary map (denoted by L) and replace the header of boundary map with “1”, Then, calculate the interpolation-errors of the sample pixels and go to Step 3).
6) Embed boundary map, L, LM’s, LN’s, RM’s, and RN’s into marginal area of the cover-image using LSB replacement.

The parameters specified in step 6 plays an important role in data extraction and image recovery process.

3) Image encryption

After rearranged self-embedded image, denoted by X, is generated, we can encrypt X to construct the encrypted image, denoted by E. With a stream cipher, the encryption version of X is easily obtained. For example, a grey value $X_{i,j}$ ranging from 0 to 255 can be represented by 8 bits, $X_{i,j}(0)$, $X_{i,j}(1)$, ….., $X_{i,j}(7)$, such that.

$$X_{i,j}(k) = \left\lfloor \frac{X_{i,j}}{2^k} \right\rfloor mod2, \quad k = 0,1, \dots .7$$

The encrypted bits $E_{i,j}(k)$ can be calculated through exclusive or operation

$$E_{i,j}(k) = X_{i,j}(k) \oplus r_{i,j}(k)$$

Where $r_{i,j}(k)$ is generated via a standard stream cipher determined by the encryption key. Finally, we embed 10 bits information into LSBs of first 10 pixels in encrypted version of A to tell data hider the number of rows and the number of bit-planes he can embed information into. Note that after image encryption, the data hider or a third party cannot access the content of original image without the encryption key, thus privacy of the content owner being protected.

B. Data Hiding in Encrypted Image

Once the data hider acquires the encrypted image, he can embed some data into it, although he does not get access to the original image. The embedding process starts with locating the encrypted version of A, denoted by A_E. Since has been rearranged to the top of E, it is effortless for the data hider to read 10 bits information in LSBs of first 10 encrypted pixels. After knowing how many bit-planes and rows of pixels he can modify, the data hider simply adopts LSB replacement to substitute the available bit-planes with additional data . Finally, the data hider sets a label following to point out the end position of embedding process and further encrypts according to the data hiding key to formulate marked encrypted image denoted by E'. Anyone who does not possess the data hiding key could not extract the additional data.

C. Marked Encrypted Image Hiding on a Cloak Image

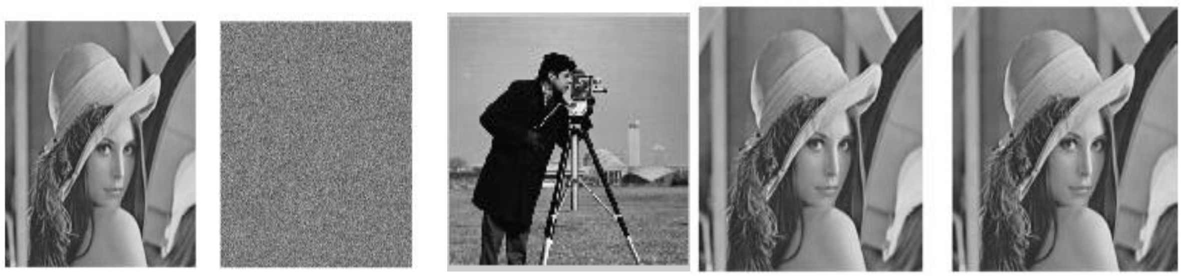
When the image hider(may be the data owner or image owner itself) gets the marked encrypted image can start the embedding process for more security. Here we select a plain image of enough size as a cover image and discards the LSB bits of some pixels by using LSB replacement. The positions of the discarded pixels can be used for embedding marked encrypted image. We use this method to hide marked encrypted image inside harmless medium in such a way that it is not possible even to detect that there is a secret message.

D. Data Extraction and Image Recovery

Since data extraction is completely independent from image decryption, the order of them implies two different practical applications.

1) Case 1: Extracting Data From Encrypted Images: To manage and update personal information of images which are encrypted for protecting clients' privacy, an inferior database manager may only get access to the data hiding key and have to manipulate data in encrypted domain. The order of data extraction before image decryption guarantees the feasibility of this work in this case. When the database manager gets the data hiding key, he can decrypt the LSB-planes of and extract the additional data by directly reading the decrypted version. When requesting for updating information of encrypted images, the database manager, then, updates information through LSB replacement and encrypts updated information according to the data hiding key all over again. As the whole process is entirely operated on encrypted domain, it avoids the leakage of original content.

2) Case 2: Extracting Data from Decrypted Images: In Case 1, both embedding and extraction of the data are manipulated in encrypted domain. On the other hand, there is a different



(a) (b) (c) (d) (e)

Figue 4. (a) original image, (b) encrypted image, (c) cloak image containing marked encrypted image, (d) decrypted image containing messages, (e) recovery version.

Situation that the user wants to decrypt the image first and extracts the from the decrypted image when it is needed. The following example is an application for such scenario. Assume Alice outsourced her images to a cloud server, and the images are encrypted to protect their contents. Into the encrypted images, the cloud server marks the images by embedding some notation, including the identity of the images' owner, the identity of the cloud server and time stamps, to manage the encrypted images. Note that the cloud server has no right to do any permanent damage to the images. Now an authorized user, Bob who has been shared the encryption key and the data hiding key, downloaded and decrypted the images. Bob hoped to get marked decrypted images, i.e., decrypted images still including the notation, which can be used to trace the source and history of the data. The order of image decryption before/without data extraction is perfectly suitable for this case.

IV. Experiments And Implementation Issues

The proposed approach will be tested on public available standard images, which include “Lena”, “Airplane”, “Barbara”, ”Baboon”, “peppers” and “boat”. The size of all images is 512x512x8.The standard image lena, shown in figure 4.a, is taken to demonstrate the feasibility of proposed method. Figure 4.b is the encrypted image containing embedded messages and the cloak image containing marked encrypted image is shown in figure4.c. The decrypted version with messages is illustrated in figure 4.d. figure 4.e depicts the recovery version which is identical to the original image. Several implement details for the proposed method can be discussed first.

A .Choice of LSB-plane Number

The size of A is determined not only by the length of to-be-embedded messages but also by the number of LSB-planes embedded reversibly in B, when dividing the original image C into A and B.

B. Discussion on Boundary Map

Boundary Map in this paper, is used for distinguishing between natural and pseudo boundary pixels and its size is critical to practical applicability of proposed approach.

V. Conclusion

Reversible data hiding in encrypted images is a new topic drawing attention because of the privacy preserving requirements from cloud data management. Proposed method implement RDH in encrypted images by reserving room before encryption. Thus the data hider can benefit from the extra space emptied out in previous stage to make data hiding process effort- less. Even though to ensure more security the proposed method uses the concept of double embedding. In this method the marked encrypted image was embedded on a cloak image. This method achieves excellent performance without loss of perfect secrecy. Furthermore, this novel method can achieve real reversibility, separate data extraction and greatly improvement on the quality of marked decrypted images.

References

[1] Kede Ma, Weiming Zhang, Xianfeng Zhao, “reversible Data Hiding In Encrypted Images By Reserving Room Before Encryption” Ieee Transactions On Information Forensics And Security, Vol. 8, No. 3, March 2013

[2] Z. Ni, Y. Shi, N. Ansari, and S. Wei, “Reversible data hiding,” IEEE Trans. Circuits Syst. Video Technol., vol. 16, no. 3, pp. 354–362, Mar.2006.

[3] L. Luo et al., “Reversible image watermarking using interpolation,” IEEE Trans. Inf. Forensics Security, vol. 5, no. 1, pp. 187–193, 2010 .

[4] X. Zhang, “Reversible data hiding in encrypted images,” IEEE Signal Process. Lett., vol. 18, no. 4, pp. 255–258, Apr. 2011.

[5] W. Hong, T. Chen, and H.Wu, “An improved reversible data hiding in encrypted images using side match,” IEEE Signal Process. Lett., vol.19, no. 4, pp. 199–202, Apr. 2012.

[6] X. Zhang, “Separable reversible data hiding in encrypted image,” IEEE Trans. Inf. Forensics Security, vol. 7, no. 2, pp. 826–832, Apr. 2012.

[7] D.R.Denslin Brabin and Dr.J.Jebamalar Tamilselvi. “Reversible data hiding: a survey,”International Journal of Innovative Research in Computer and Communication Engineering Vol. 1, Issue 3, May 2013 .