International Journal of Research in Advent Technology, Vol.2, No.2, February 2014

E-ISSN: 2321-9637

Secure RDH for Encrypted Images by Conforming Space before Encryption

Chaithu V Kumar
Computer Science, Anna University
Email: chaithuit16@gmail.com

Abstract — A novel Secure RDH for Encrypted Images by Conforming space before Encryption is presented in this paper. Using RDH, the original image can be recovered without loss after the extraction of embedded data. In existing system, the data is embedded by reversibly vacating room in the encrypted image, which may make some error in data extraction and/or image restoration. In the proposed method, instead of vacating room in the encrypted image, the room is reserved before encryption and the data is reversibly embedded in the encrypted image. Thus, the data extraction and image recovery are achieved without any error.

Index Terms-Reversible data hiding; Image Encryption; Privacy protection; Histogram Shift.

1. INTRODUCTION

By using reversible data hiding (RDH) technique, the original image is recovered after the extraction of embedded data. It is an important property that can be applied to many scenarios, such as medical imagery, military imagery and law forensics, where the original image should be recovered losslessly. For this reason, RDH becomes a hot research topic and is extensively studied over the years.

Existing method tries to vacate room in the encrypted image. Since the entropy of encrypted images has been maximized, this technique can only achieve small payloads or generates poor image quality for large payloads. Also, this technique makes some error rates on data extraction and/or image

In this proposed method, instead of vacating room after encryption, the room is reserved before encryption. LSBs of some pixels are embedded into other pixels with RDH method and then image is encrypted. So the LSBs of the encrypted image are used to accommodate data. The proposed method provides good performance in two different prospects

- Image recovery and data extraction are free of any error.
- The PSNRs of decrypted image containing the embedded data are significantly improved.

This paper is organized in the following manner. Section II discusses about the existing system. The proposed method is elaborated in Section III followed by some implementation issues in Section IV. Experiments with analysis and comparison are given in Section V. The paper is concluded in Section VI.

2. EXISTING METHOD

The existing system was done by "vacating room after encryption (VRAE)" as illustrated in Fig. 1(a).

In this method, the content owner encrypts the original image using standard encryption algorithm with an encryption key. The content owner passes the encrypted image to data hider (e.g., a database manager) and the data hider can embed some confidential data after vacating some room according to the data hiding key. Then a receiver, content owner or authorized third party can extract the data using data hiding key and recover the original image using the encryption key.

In this method, the encrypted image is segmented into number of nonoverlapping blocks sized by $a \times a$, where each block is used to carry one additional bit. To achieve this, the pixels of each block are pseudo-randomly divided into two sets *S1* and *S2* according to a data hiding key. If the additional bit to be embedded is 0, flip the 3 LSBs of each encrypted pixel in *S1*; otherwise flip the 3 encrypted LSBs of pixels in *S2*.In order to achieve both image recovery and data extraction, the receiver flips the three LSBs of *S1* to make a new decrypted block and flips three LSBs of *S2* to make another new block; one of them is decrypted to the original block. Due to spatial correlation in natural images, both the original image and embedded data can be extracted. However, there is a problem in bit extraction and image recovery when divided blocks are very small (e.g., *a=8*) or has much fine detailed texture in the original image.

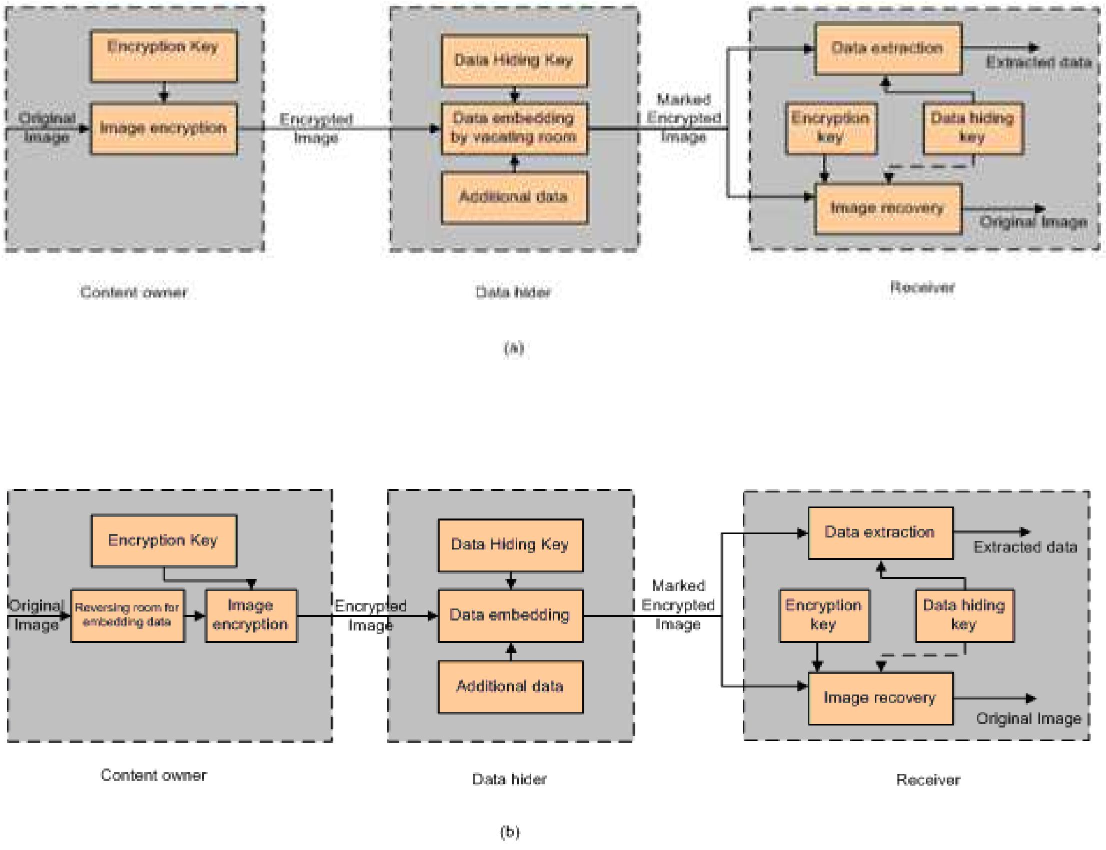

Fig. 1. (a) vacating room after encryption (VRAE) (b) eserving room before encryption (RRBE)

3. PROPOSED METHOD

Vacating room in the encrypted image is somewhat difficult and sometimes inefficient. So, in this proposed method, better result is achieved by changing the order of encryption and reserving room, i.e., reserving room before encryption at content owner side. Hence, more natural and much easier RDH task is achieved in encrypted image. The framework proposed in this paper is known as "reserving room before encryption (RRBE)"

As shown in Fig. 1(b), the content owner reserves enough space to accommodate data and then encrypts the image using encryption key. Now, the data hider embeds the data in the encrypted version of the image. Finally, the receiver extracts the data as well as image losslessly.

Further sections deal with the practical method based on the framework "RRBE", which consists of four stages: generation of encrypted image, data hiding in encrypted image, data extraction and image recovery.

3.1 Generation of Encrypted image

In order to generate the encrypted version of the image, this stage can be divided into three steps: image partition, self reversible embedding and image encryption. In image partition step, the original image is divided into two parts A and B. Then, the LSBs of A are reversibly embedded into B with a standard RDH algorithm. So, the LSBs of A are used to embed confidential data. Finally, this rearranged image is encrypted to get the encrypted version of the image.

3.1.1 Image partition: Assume the original image C, which is an 8 bits gray-scale image with size M × N and pixel $C_{i,j} \in [0,255], 1 \leq i \leq M, 1 \leq j \leq N$. Content owner extracts several overlapping blocks from the original image along the rows. The block size *l* is determined by the size of the to-be-embedded messages. Each block contains *m*

= *[l/M]*, and the number of blocks can be calculated by *n=M-m+1*. Here, each block is overlapped with nearby block along the rows. For each block, first order smoothness is calculated by formula (1)

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

Higher *f* value gives more complex texture. So, the content owner selects the block with highest *f* as *A*, and lowest *f* as *B*. Keeps the *A* at front of the image and B at rest part as shown in Fig. 2. Normally only single LSB-plane of A is embedded into B. But, it is possible to embed two or more LSB-plane of A into B. However, the performance decreases in terms of PSNR

3.1.2 Self-Reversible Embedding: The main aim of this stage is reversibly embedding LSBs of A into B using standard RDH algorithm. Pixels in the image B are categorized into two sets. As shown in Fig. 2, white pixels with its indices i and j satisfying *(i +j) mod 2 = 0* and black pixel whose indices meet *(i + j) mod 2 = 1*. For each white pixel, interpolation is calculated by four surrounding black pixel 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 the weight w_i, $1 \leq i \leq 4$. The estimating error is calculated by $e_{i,j} = B_{i,j} - B'_{i,j}$, and some data can be embedded into the estimating error sequence with histogram shift. Similarly, calculate the estimating error for black pixel with the help of four surrounding white pixels and can be used to accommodate data.

3.1.3 Image Encryption: Image *X* is generated after the self-embedding stage. Encrypted image *E* is obtained by encrypting the image *X* using stream cipher. For example, a gray scale pixel 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

$$X_{i,j}(k) = \left\lfloor \frac{X_{i,j}}{2^k} \right\rfloor \bmod 2, \qquad k = 0,1,\ldots,7.$$

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

$$E_{i,j}(k) = X_{i,j}(k) \oplus r_{i,j}(k). \qquad (4)$$

Where $r_{i,j}(k)$ is generated via stream cipher determined by the encryption key. 10 bits information such as the number of rows and number of bit-planes are embedded into the LSBs of first 10 pixels of encrypted version of A. So, the data hider can embed the data based on the 10 bit information. Here, data hider or third party cannot access the original image without the encryption key.

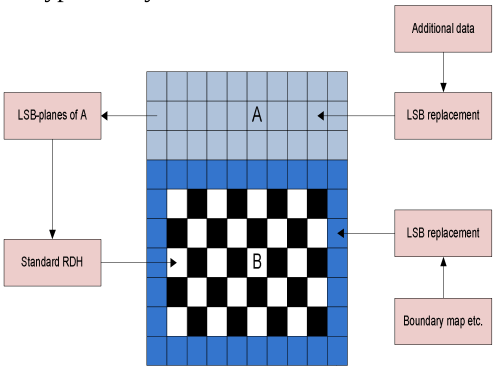

Fig. 2. Illustration of image partition and embedding process.

3.2 Data Hiding in Encrypted Image:

Data hider embeds the data into the encrypted image E. Embedding process begins with identifying the encrypted version of A, denoted by A_E. Since A_E is arranged to the top of E, data hider reads the 10 bits information from the first 10 LSBs of encrypted pixels.

After knowing the number of rows and bit-planes, data hider can embed data m. Finally, data hider sets a label followed by m to denote the end of embedding process. At last the data m is encrypted using data hiding key to form a marked encrypted image denoted by E.

3.3 Data Extraction and Image Recovery:

There are two practical applications since the data extraction is completely independent from image decryption.

Case 1: Extracting Data From Encrypted Images: Normally, personal information of the image is encrypted in the image. To manage and update such information, database manager may access it using data hiding key.

If the database manager gets the data hiding key, he can extract data *m* by decrypting the LSB-planes of A_E. If he wants to update the data *m*, he can update it through LSB replacement and encrypt the updated data with data hiding key. Since all these processes are doing in the encryption version of image, the original image is hidden from database manager.

***Case 2:** Extracting Data From Decrypted Images:* In Case 1, both the data embedding and data extraction are done in encrypted version of image. In some situation, the user wants to decrypt the image first and then extract the data from the decrypted image. The following example is an application for such scenario.

Assume Alice stores her image into cloud server and the image is encrypted to protect its content. Then, some information about the image such as identity of the image, owner, identity of the cloud server, time stamp etc., are embedded into the image and encrypted this data using data hiding key.

Now, an authorized user, Bob who has been shared with encryption key and data hiding key decrypts the image. First, Bob gets the marked decrypted image i.e., decrypted image still having some information about the image.

3.4 Generating the Marked Decrypted Image: To get marked image X'' which is made up of A'' and B'', the content owner decrypts the image except the LSB-planes.

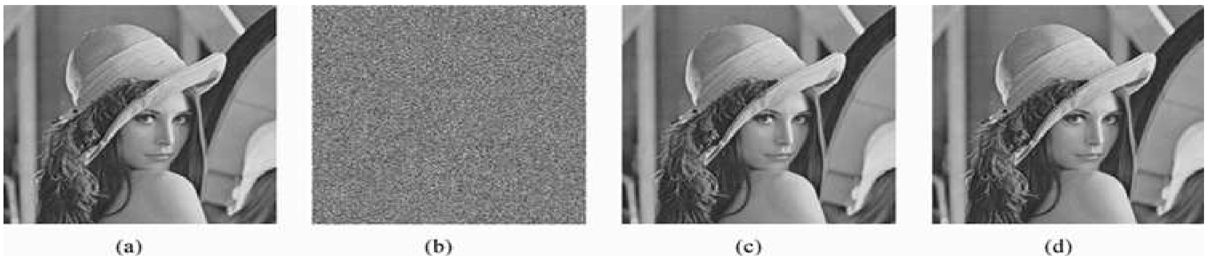

Fig. 3. (a) Original image, (b) encrypted image, (c) decrypted image containing messages, (d) recovery version.

b) Data Extraction and Image Restoration: After generating the marked decrypted image, the content owner can further extract the data and recover original image.

4. IMPLEMENTATION ISSUES

Following are the different implementation issues

4.1 Choice of LSB-Plane Number

The original image C is divide into A and B. The size of A is determined not only to-be-embedded data but also the number of LSB-planes. When multiple LSB-plane is used, the image can be embedded more data. But, the noise rate is increased when the number of LSB-plane is getting increased. So, single LSB-plane gives better result than multiple LSB-planes.

4.2 Choice of Embedding Strategy

There are two possible ways to embed the message 1) embedding data into peak points by making use of part error sequence; and 2) searching for proper points in the histogram of all estimating errors. In this paper, the first solution is used when peak points of estimating error sequence of cover image account for more than 20% of the whole errors; otherwise switch to the second

4.3 Discussion on Boundary Map

Boundary map is used for distinguishing between natural and pseudo boundary pixels and its size is critical to practical applicability of proposed approach. In most cases, no boundary map is needed.

5. EXPERIMENTS AND COMPARISONS

As shown in Fig. 3(a), standard image Lena is used to demonstrate the feasibility of proposed method. Fig. 3(b) is the encrypted image containing embedded messages and the decrypted version with messages is illustrated in Fig. 3(c). Fig. 3(d) depicts the recovery version which is identical to original image.

The existing method may make some errors on data extraction and/or image restoration. But the proposed method is free from any kind of errors.

Quality of the image is compared in terms of PSNR. Error correction codes are used in this paper to eliminate errors. More amounts of data can be embedded than the existing system.

6. PROPOSED METHOD WITH HIGH SECURITY

For the better secure communication uses the AES and SHA1 algorithm. So that the Data Encryption module is also added.

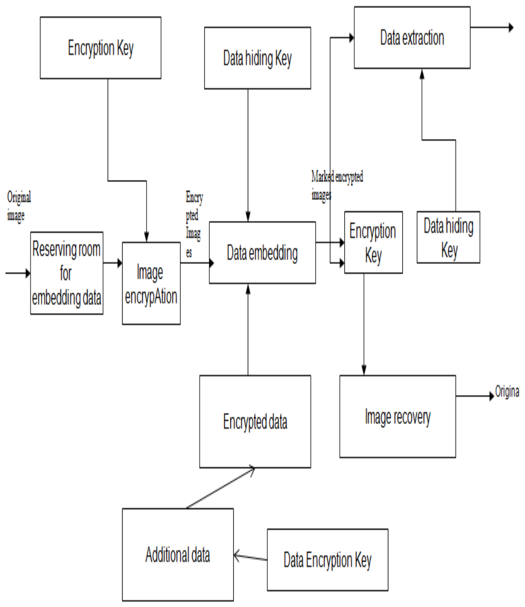

Fig.4. Proposed System Architecture

Here the data is encrypted using AES algorithm and the size of the key which is used in this is 128.And the key is generated by using the SHA1 Algorithm according to the given input. The input which we given is less than 16 characters. Then the given data is applied to SHA1 algorithm to get the key. The data is encrypted be using the generated key by using AES.

7. CONCLUSION

For cloud data management, RDH in encrypted image is being used since it provides privacy. In existing system, the data is stored in encrypted image after vacating enough space. Whereas, in proposed system, the enough space is reserved before encrypting the image. So the data hiding process effort is reduced. The proposed method uses the advantages of traditional RDH techniques and achieves good performance. Furthermore, it provides separate data extraction without image decryption and improves the quality of the marked image.

Acknowledgment

First the Author thanks to god and also wish to thanks the Management of Loyola Institute of Technology and Science for providing the excellent computing facility and encouragement

References

[1] T. Kalker and F.M.Willems, "Capacity bounds and code constructions for reversible data-hiding," in Proc. 14th Int. Conf. Digital Signal Processing (DSP2002), 2002, pp. 71–76.

[2] W. Zhang, B. Chen, and N. Yu, "Capacity-approaching codes for reversible data hiding," in Proc 13th Information Hiding (IH'2011), LNCS 6958, 2011, pp. 255–269, Springer-Verlag.

[3] W. Zhang, B. Chen, and N. Yu, "Improving various reversible data hiding schemes via optimal codes for binary covers," IEEE Trans.Image Process., vol. 21, no. 6, pp. 2991–3003, Jun. 2012.

[4] J. Fridrich and M. Goljan, "Lossless data embedding for all image formats," in Proc. SPIE Proc. Photonics West, Electronic Imaging, Securityand Watermarking of Multimedia Contents, San Jose, CA, USA,Jan. 2002, vol. 4675, pp. 572–583.

[5] J. Tian, "Reversible data embedding using a difference expansion," IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 8, pp. 890–896,Aug. 2003.

[6] 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.

[7] D.M. Thodi and J. J. Rodriguez, "Expansion embedding techniques for reversible watermarking," IEEE Trans. Image Process., vol. 16, no. 3,pp. 721–730, Mar. 2007.

[8] X. L. Li, B. Yang, and T. Y. Zeng, "Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection," IEEE Trans. Image Process., vol. 20, no. 12, pp. 3524–3533, Dec.2011.

[9] P. Tsai, Y. C. Hu, and H. L. Yeh, "Reversible image hiding scheme using predictive coding and histogram shifting," Signal Process., vol.89, pp. 1129–1143, 2009.

[10] L. Luo et al., "Reversible imagewatermarking using interpolation technique," IEEE Trans. Inf. Forensics Security, vol. 5, no. 1, pp. 187–193,Mar. 2010.

[11] V. Sachnev, H. J. Kim, J. Nam, S. Suresh, and Y.-Q. Shi, "Reversible watermarking algorithm using sorting and prediction," IEEE Trans.Circuits Syst. Video Technol.,vol. 19, no. 7, pp. 989–999, Jul. 2009.

Chaithu v Kumar received B.Tech degree in Information Technology from Marthandam College of Engineering and Technology, Marthandam, Tamil Nadu in 2012 and currently doing M.E degree in Loyola Institute of Technology and Science, Nagercoil, Tamil Nadu.