Data Hiding and Retrieval in Encrypted Images

[1]K.Shankar, [2]Dr.C.Yaashuwanth

[1]PG Scholar, [2]Associate Professor
Information Technology, SRM University, India
[1]sasishankark53@gmail.com, [2]yaashuwanth@gmail.com

***Abstract*—In recent days more attention is paid towards data hiding, this paper aims at providing confidentiality and integrity towards both the data as well as image. Reversible data hiding (RDH) in encrypted images provides excellent property that the original cover can be losslessly recovered. All the previous methods of embedding data by reversibly vacating room after encryption may subject to some errors on the data as well as image extraction. In this paper I propose a novel method by reserving room before encryption with a traditional RDH algorithm, and thus it is easy for the data hider to reversibly embed data in the encrypted images. The proposed method can achieve real reversibility, that is data extraction and image recovery are free from any error. Experiments show that this novel method can embed more than 10 times lager payloads (i.e) efficiency for the same image quality as the previous methods.**

***Index Terms*—Reversible data hiding, image encryption, privacy protection, histogram shift**

I. INTRODUCTION

Reversible Data Hiding (RDH) in images is a technique, by which the original cover can be losselessly recovered after the embedded message is extracted. This important technique is widely used in medical imagery, military images and law forensics, where no distortion of the original cover is allowed.

In theoretical aspect, Kalker and Willems [1] established a rate-distortion model for RDH, however Zhang *et al*. [2], [3] improved the recursive code construction for binary covers and proved that this construction can achieve the rate-distortion bound.

In practical aspect, may RDH techniques have emerged in recent years? Fridrich *et al*. [4] constructed a general framework for RDH. A more popular method is difference expansion (DE) [5], in which difference of each pixel group is expanded, and thus the least significant bits (LSBs) is used for embedding messages. Another strategy is histogram shift (HS) [6], in which the space is saved for data embedding by shifting the bits of histogram gray values. With regard to provide confidentiality for images, encryption [12] is an effective and popular means as it converts the original and meaningful content to incomprehensible one.

Some attempts on RDH in encrypted images have been made. In [16], Zhang divided the encrypted image into several blocks. By flipping 3 LSBs of the half of pixels in each block, room can be vacated for embedded bit. The data extraction and image recovery proceed by finding which part has been flipped in one block. This process can be realized with the help of spatial correlation in decrypted image.

All the three methods try to vacate room from the encrypted images directly. However, since the entropy of the encrypted images has been maximized, these techniques can only achieve small payloads or generate marked image with poor quality for large payload all of them are subject to some error rates on data extraction.

In the present paper, we propose a novel method for RDH in encrypted images, for which we do not **"vacate room after encryption"** as done but **"reserve room after encryption"**. This method achieves excellent performance in two different prospects:

- Real reversibility is realized, that is, data extraction and image recovery are free of any error.
- For given embedding arte, the PSNRs of decrypted image containing the embedded data are significantly improved; and for the acceptable PSNR, the range of embedding rates is greatly enlarged.

II. PREVIOUS ARTS

The methods proposed in this framework, "vacating room after encryption (VRAE)", as illustrated in Fig. 1(a).

In this framework a content owner 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 and the data hider can embed some auxiliary data into the encrypted image by loselessly 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 key and further recover the original image from the encrypted version according to the encryption key.

In all methods of [16]-[18], the encrypted 8-bit gray-scale images are generated by encrypting every bit-planes with a stream cipher. To do this, pixels in 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 pixels in S1, otherwise flip the 3 encrypted LSBs of pixels in S2. For data extraction and image recovery, the receiver flips all the three LSBs of pixels in S1 to form a new decrypted block, and flips all the three LSBs of pixels in S2 to form another new block; one of them will be decrypted to the original block.

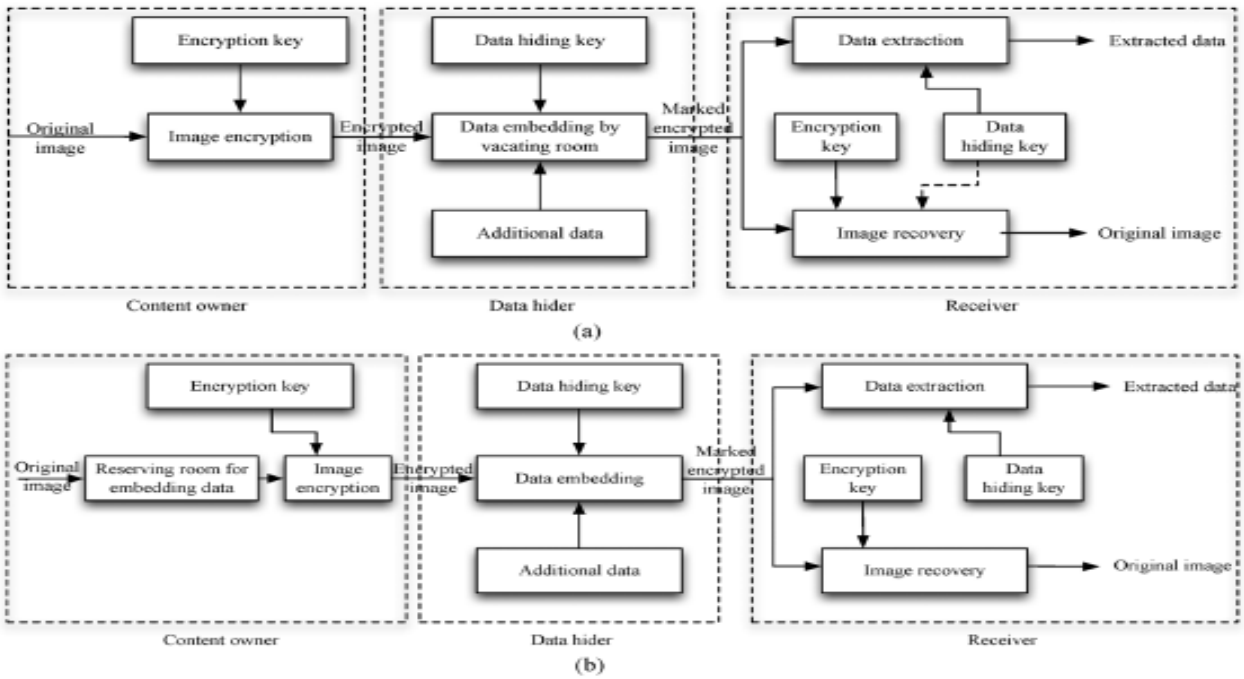

Fig. 1. Framework: "vacating room after encryption (VRAE)" versus framework: "reserving room before encryption (RRBE)." (Dashed line in (a) states that the need of data hiding key in image recovery varies in different practical methods). (a) Framework VRAE. (b) Framework RRBE.

III. PROPOSED METHOD

Since lossless vacating room from the encrypted images is relatively difficult and sometimes inefficient, why are we still so obsessed to find novel RDH techniques working directly for encrypted images? If we reverse the order of encryption and vacating room the RDH tasks in encrypted images would be more natural and much easier which leads us to the novel framework (RRBE), this framework achieve better performance compared with the techniques from framework (VRAE). This is because (RRBE) framework first losslessly compresses the redundant image content and then encrypts it with respect to protecting privacy.

Practical methods of RRBE framework consists of four stages: generation of the encryption images, data hiding in encrypted image, data extraction and image recovery.

A. GENERATION OF ENCRYPTED IMAGE

Actually the first stage can be divided into three steps: image partition, self-reversible embedding followed by image encryption. At the beginning, image partition step divides original image in two parts A and B then, the LSBs of A are reversibly embedded into B with a standard RDH algorithm so that LSBs of A can be used for accommodation of messages

1) ***Image partition:*** The goal of the image partition is to construct a smoother area B, on which standard RDH algorithms can achieve better performance. To do that assume the original image C is an 8 bit gray-scale image with its size M×N and pixels Ci,j €[0,255]. First the content owner extracts 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. For each block, define a function to measure its first-order smoothness

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

The above discussion implicitly relies on the fact that only single LSB-plane of A is recorded. It is straightforward that the content owner can embed two or more LSB-planes of A into B. however the performance of A in terms of PSNR, after embedding in the second stage decreases significantly with growing bit-planes exploited.

Therefore we investigate situation that at most three LSB-planes of A are employed and determine the number of bit-planes with regard to different payloads experimentally.

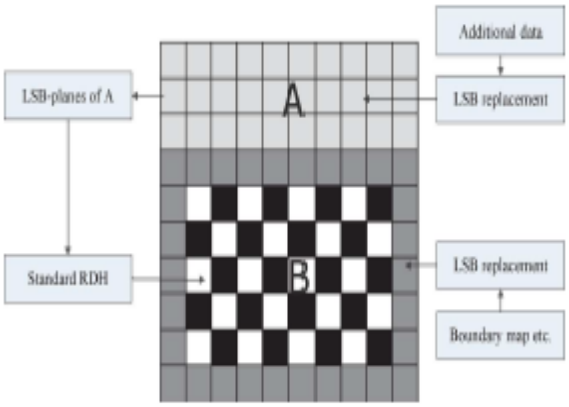

Fig. 2. Illustration of image partition and embedding process.

2) ***Self-Reversible embedding:*** the goal of self-reversible embedding is to embed the LSB-planes of A into B by employing traditional RDH algorithms.

Pixels in the rest of image B are categorized into two sets: white pixels with its indices I and j satisfying (I + j) mod 2 = 0 and the black pixels whose indices meet (I + j) mod 2 = 1, as shown in fig.2. Then each white pixel is indicated with the values obtained with the four black pixels.

$$B'_{i,j} = w_1 B_{i-1,j} + w_2 B_{i+1,j} + w_3 B_{i,j-1} + w_4 B_{i,j+1},$$

After that we can embed some data with the help of histogram shift, after that we can calculate the black pixel with the help of surrounding white pixels that may be modified.

By bidirectional histogram shift, some messages can be embedded on each error sequence. That is first divide histogram into two parts, the left part and the right part, and search for the highest point in each part, denoted by LM and RM respectively. Search for the zero point in each part, denoted by LN and RN. To embed some message 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 part 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.

The problem is that overflow/underflow problem occurs when the natural boundary pixels change from 255 to 256 or from 0 to -1. To avoid it we embed data from 1 to 0 or from 254 to 255 during the embedding process. These boundaries are defined as pseudo-boundary pixels.

Hence a boundary map is introduced to tell whether boundary pixels in the marked image are natural or pseudo in the extracting process. It is a binary sequence with bit "0" for natural boundary pixel, bit "1" for pseudo –boundary pixel.

These parameters play an important role in the data extraction and image recovery

3) ***Image Encryption:*** After rearranging self-embedded image, we can encrypt the image. With a stream cipher, the encryption version can be easily obtained. For example, a gray value Xi,j ranging from 0 to 255 can be represented by 8 bits, Xi,j(0), Xi,j(1) . . .Xi,j(7), such that

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

The encrypted bits Ei,j(k) can be calculated through exclusive-or operation

$$\mathbf{E}_{i,j}(k) = \mathbf{X}_{i,j}(k) \oplus r_{i,j}(k),$$

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 encryption version of A to tell data hider the number of rows and the number of bit-planes. 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 is being protected.

B. ***Data Hiding in Encryption Image***

Once the data hider acquires the encrypted image E, he can embed some data into it, although he does get access to the original image. The embedding process starts with locating the encryption version of A, denoted by AE. 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 the addition data.

C) Data extraction and image recovery

Since data extraction and image extraction are two different independent applications.

1) Case1: Extracting Data from Encrypted Image:

When the database manager gets the data hiding key, he can decrypt the LSB-planes of AE and extract the additional data *m* by directly reading the decrypted version. The data base manager then updates information through LSB replacement and encrypts updates information according to the data hiding key all over again. It avoids the leakage of the original content.

2) Case2: Extracting Data from Decrypted Images:

In case 1, both embedding and extraction of data are manipulated in encrypted domain. On the other hand, there is different situation that the user wants to decrypt the image first and extracts the data from the decrypted image when it is needed. However the data to be embedded can be a third party , he is not authorized to view the encrypted image leaving the confidentiality of the image , thus only the person who gets the image key is authorized to view the data . The decrypted images 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

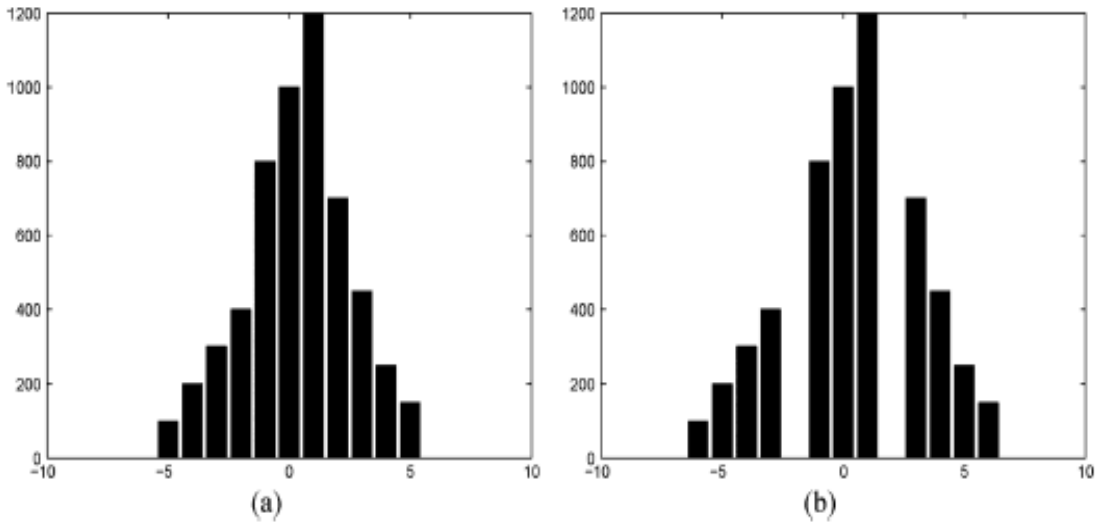

Fig. 3. Selection of proper points. (a) original histogram, (b) shifted histogram. (In this figure, length of messages is 1000 bits, $LP = -2$ and $RP = 2$.)

perfectly suitable for this case. Next, we describe how to generate a marked decrypted image.

IV. IMPLEMENTATION ISSUES

The size of all images is 512×512×8. The objective criteria PSNR is employed to evaluate the quality of marked decrypted image quantitatively. To achieve high PSNRs, several implementation details for the proposed method are discussed first.

A. Choice of LSB-Plane Number

When the image is divided into A and B, the size of A is determined not only the length of the message but also by the number of LSB-planes embedded reversibly in B. Therefore it is more likely that B only need to implement embedding schemes once to accommodate LSB-planes of A, thus leading to distortion happens in B. the embedding rate is measured by bits per pixel (bpp).

B. Choice of Embedding Strategy

- Embedding data into peak points by making use of part of error sequence
- Searching for proper points in the histogram of all estimating errors.

The first solution performs better than the other when the cover image is relatively smooth with little fine-detail regions, therefore resulting in a sharper representation in error histogram. The improvement can have 2 to 4 dB at low embedding rate levels.

C. Discussion of Boundary Map

Boundary map is distinguished between natural and pseudo boundary pixels and its size is critical to the practical applicability of proposed approach

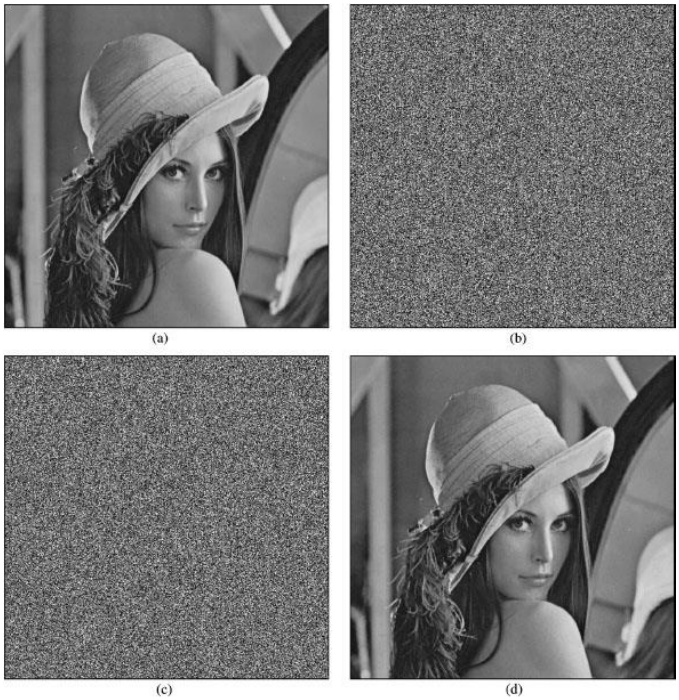

Fig 4 (a) Original Image converted into gray scale (b) Encrypted Image (c) Encrypted Image sent to Data Hider (d) Decrypted Image as well as Data

V. CONCLUSION

Reversible data hiding in encrypted images is anew topic drawing attention because of the privacy of the privacy-preservation requirements from cloud data management. The proposed method data hider can benefit from the extra space emptied out in previous stages to make data hiding process effortless. Thus it achieve excellent performance without loss of perfect secrecy

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, Security And 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 image watermarking 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.

[12] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, FL, USA: CRC, 1996.

[13] K. Hwang and D. Li, "Trusted cloud computing with secure resources and data coloring," *IEEE Internet Comput.*, vol. 14, no. 5, pp. 14–22, Sep./Oct. 2010.

[14] M. Johnson, P. Ishwar, V. M. Prabhakaran, D. Schonberg, and K. Ramchandran, "On compressing encrypted data," *IEEE Trans. Signal*