Volume 8, No. 5, May-June 2017
ISSN No. 0976-5697
International Journal of Advanced Research in Computer Science
RESEARCH PAPER
Available Online at www.ijarcs.info

Security in Cloud Based Applications

Nikhil Kumar Singh
Department of Computer Science and Engineering
Babasaheb BhimRao Ambedkar University,
Lucknow, India

Jitendra Kurmi
Department of Computer Science and Engineering
BabaSaheb BhimRao Ambedkar University,
Lucknow, India

Abstract: Now-a-days many unauthorized users try to get the protected information and therefore it is necessary to secure our data. Besides, there are also scenarios that data hiding needs to be done in the encrypted domain or combined with the encryption, especially in the age of big data and cloud computing. In the previous methods first encryption was done and then the room was vacated to hide the data. But there were some errors in data extraction and image recovery. In this paper a novel technique is proposed termed as reversible data hiding (RDH). It is an excellent technique by which we can hide our data. RDH is applied to encrypted images by which we can properly recover our data and the cover image. The hidden data can be in the form of text or image. In the proposed method we first vacate room to hide data and after encrypting image using certain encryption key the data hider reversibly hides the data whether text or image using data hiding key. In previous methods only text was hidden. But the proposed paper is extended to hide the image also in the reserved space. The hidden image is also recovered without any errors. The results given in the paper proves that we can exactly recover the hidden data, hidden image and the cover image as and when needed by the receiver. The traditional RDH algorithm used is histogram shifting. By this method we can properly recover hidden data and cover image without any errors. Large amount of data can be hidden as compared to previous methods.

Keywords: Reversible data hiding, image encryption, secured communication, privacy of data, histogram shift

I. INTRODUCTION

Many data hiding techniques have been proposed previously but reversible data hiding is an excellent technique in which the recovered version is exactly same as the original image [1]. The technique is suitable for military and medical applications. The methods used in RDH are difference expansion and histogram shifting [2]. In this paper histogram shifting technique is used where bins of the histogram are shifted to embed data or image in it. Encryption is done to hide the original content of image. In this paper chaos based encryption technique is used [3]. Previous methods used "VRAE (vacating room after encryption)" in which some errors were seen while recovering original image. But here "RRBE" (reserving room before encryption) is used. In RRBE we first embed LSBs of certain pixels into LSBs of other pixels so that we can obtain space to embed data [4]. The decrypted image is error free as shown in results. The confidentiality of the image is protected from other users.

Not only does the proposed method separate data extraction from image decryption but also achieves excellent performance in various prospects:

a) Image extraction, data extraction and recovered original image are free of any error.
b) The data hider can easily embed data in encrypted image as he knows the position in which he has to place the data.
c) The data hider does not know the original image and just has to hide data in the proper bit position.
d) The quality of decrypted image is improved as compared to previous methods.
e) The hidden information is 10 times more as compared to previous methods.

II. RELATED WORK

In previous methods first the encryption was done using certain encryption key. After this the encrypted image was passed to data hider so that he can hide data or image in it. The data hiders vacates somespace from encrypted image and hide data or image in it as required [5]. Then this image is passed to receiver in which he can recover the hidden text or image using data hiding key. If the receiver has decryption key then he can also recover the original image. But this method has shown some errors while decryption of image. So we have used another method known as "reserving room before encryption" which is error free [6].

III. PROPOSED METHOD

As previous method were prone to some errors we have reversed the order of encryption and vacating room [7]. In proposed method "Reserving room before encryption" is used in which space is reserve to embed data before encrypting the image. The data hiding is reversible as the data hider needs only to hide data in the reserved space without knowing the original image. Thus the contents of original image are protected. Fig. 1 and fig. 2 shows the process of reserving room before encryption [8]. In this the original image is split into two parts and in one of the part the data is hidden by vacating room using histogram shifting technique. The framework "RRBE" primarily consists of four stages:

a) Reserving room
b) Generation of encrypted image
c) Data hiding in encrypted image
d) Decryption of image and extraction of hidden information

A. Reserving Room

It involves reserving room in the image in order to make space to embed data before encrypting the image. This is an

important stage in this algorithm. In this we first empty out room i.e. create space in the image before encryption of image [9].

For the RDH task in encrypted image to be more natural and much easier real reversibility is realized by first lossless compressing the redundant data of image. In this way space is created for embedding data and then we can encrypt the image by different encryption technique.

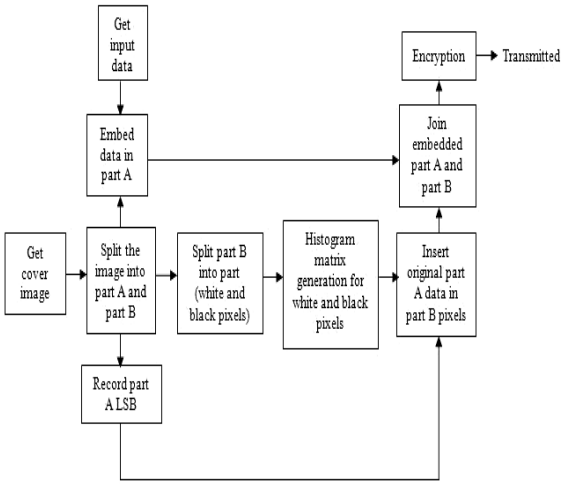

Fig. 1: Framework RRBE

1) Image partition: In this step we divide the original image into two parts A and B. For generating these two parts a particular procedure has to be followed which is explained below. Firstly the original image is divided into several overlapping blocks [10]. Genetic algorithm is used for selecting the blocks which contain relatively more complex textures. The block which has highest features is selected as part A of image. The rest of the part is with fewer textured areas which is considered as part B of the image. These two parts are concatenated and then given for embedding process [11].

For reserving room before encryption standard RDH technique is used. The part B of image is smoother area on which standard RDH algorithm can be applied. To do that, without loss of generality, assume the original image is an 8 bit gray-scale image with its size M×N and pixels

$$\mathbf{C_{i,j} \in [0,255],\ 1 \le i \le M,\ 1 \le j \le N} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

The original image is divided into several overlapping blocks. The number of blocks depends on the size of the information to be stored in the image. The size of embedded message is denoted by l. 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 by

$$\boldsymbol{f} = \sum_{u=2}^{m} \sum_{v=2}^{N-1} | C_{u,v} - (C_{u-1,v} + C_{u+1,v} + C_{u,v-1} + C_{u,v+1}/4)| . (2)$$

The block which has higher f is selected as part A of image. This block is then concatenated by rest of the part of the image denoted by B.

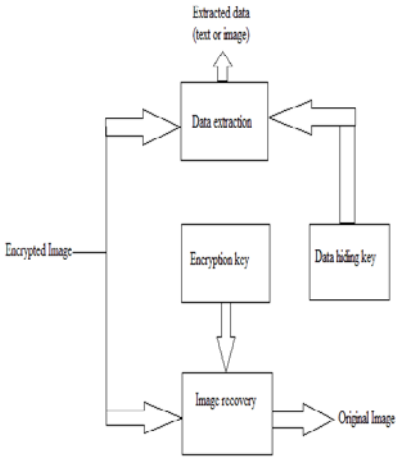

Fig. 2: Recovered image and data

2) Self-reversible embedding:[18]In this step the LSBs of 1st part are reversibly embedded into 2nd part with a standard RDH algorithm so that LSBs of 1st part can be used for accommodating messages [12]. Pixels in the rest of image B are first categorized into two sets: white pixels with its indices i and j satisfying i+j(mod 2)=0 and black pixels whose indices meet i+j(mod 2)=1.Then, each white pixel $B_{i,j}$ is estimated by the interpolation value obtained with the four black pixels surrounding it as follows,

$$\mathbf{B'_{i,j} = w_1 B_{i-1,j} + w_2 B_{i+1,j} + w_3 B_{i,j-1} + w_4 B_{i,j+1}} \ldots\ldots\ldots\ldots\ldots\ldots (3)$$

Where w_i is the weight. The estimating error is calculated via

$$\mathbf{e_{i,j} = B_{i,j} - B'_{i,j}} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4)$$

and then some data can be embedded into the estimating error sequence with histogram shift. If space is not sufficient to embed the entire hidden data then estimating errors of black pixels is also calculated by the same procedure as explained for white pixels. Furthermore, we can also implement multilayer embedding scheme by considering the modified B as "original" one when needed [13].

a) Histogram Shifting Technique: 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 [19], denoted by LM and RM, respectively. For typical images LM= -1, and RM= 0. The error obtained is compared with LM and RM which are the peak bin pixels. Furthermore, search for the zero point in each part, denoted by LN and RN. If the error is equal to LM then the first pixel of the payload message is subtracted from it, i.e. the error is shifted towards the left. If the error is equal to RM then the first pixel of the payload message'd' is added to it, i.e. the error is shifted towards the right. If the error is between LN and LM or RN and RM, then the error is subtracted by 1(shifted to the left by one step) or moved one step forward to the right respectively. If none of these conditions gets satisfied then the same error persists. In this way the new error is calculated [14]. These new errors are added to the predicted value to obtain the new value.

B. Generation of encrypted image

After rearranged self-embedded image, denoted by X, is generated, we can encrypt X to construct the encrypted image, denoted by E. There are two ways of encrypting an image i.e. shuffling the pixels of the image or substituting the pixel of the image. For this purpose we use confusion and diffusion, in confusion the pixels of the image are replaced by another value and in the process of diffusion the pixel values are shuffled within the image. These two processes are used in the algorithm for key generation. Selective encryption is carried out with the help of chaos map [15].

The encryption is carried out in two stages:

1. Chaos based key Generation
2. Selective Encryption

Functions:

HENON MAP

$X_{n+1}=Y_{n+1}-aX_n^2$……(5)

$Y_{n+1}=bX_n^2$……(6)

$Z_{n+1}=1-cY_{n+1}+X_n$……(7)

KEY GENERATION

$K_1=(X_n*Y_n)/256$……..(8)

$K_2=(Y_n*Z_n)/256$……..(9)

$K_3=(Z_n*X_n)/256$…….(10)

$K_4=mod(X_n,255)$……....(11)

CONFUSION PROCESS

$R(1,1)= R(1,1)XOR\ K_1$

$G(1,1)= G(1,1)XOR\ K_2$

$B(1, 1)= B(1,1)XOR\ K_3$

After confusion, diffusion process is used. In diffusion process the output bits should depend on the input bits in a very complex way. To achieve this, in this algorithm diffusion process is carried out in two steps i.e. horizontal diffusion and vertical diffusion.

TABLE I
PSNR Comparison for Various Embedding Rates

PSNR results in Db					
Embedding rate(bpp)	0.1	0.2	0.3	0.4	0.5
Lena	52.93	49.56	45.97	43.19	39.87
Airplane	55.23	51.94	50.56	46.23	43.28
Barbara	51.17	47.93	44.59	41.67	37.76

*1) Horizontal Diffusion:*This is one of the techniques used in this algorithm. In horizontal diffusion each pixel is bit XORed with the next pixel row wise. Starting from the second pixel of red channel, the first pixel of the red channel is XORed with the second pixel and in the same way all the three channels are horizontally diffused.

R(i,j+1)= bitxor (R(i,j+1,1),R(i,j,1))

G(i,j+1,2)= bitxor (G(i,j+1,2),G(i,j,2))

B(i,j+1,3)= bitxor (B(i,j+1,3),B(i,j,3))

C. 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. 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 the hidden data topoint out the end position of embedding process and further encrypts the part A according to the data hiding key to formulate marked encrypted image. Image can also be hiddenin the same way as we hide the data. Anyone who does not possess the data hiding key could not extract the additional data or hidden image [16].

D. Decryption of image and extraction of hidden information

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

1) Extracting Data from Encrypted Images: In this the receiver has only the data hiding key to extract the text or image hidden in the cover image. He does not have access to decryption key and so he cannot decrypt the cover image. So the original cover image is protected [17].

2) Extracting Data from Decrypted Images: In above situation the user does not have access to decryption key. So he cannot decrypt the original image. Next, we describe how to generate a marked decrypted image [18].

TABLE II
Length of Boundary Map under Different Embedding Rates

Boundary map size (bits)					
Embedding rate(bpp)	0.1	0.2	0.3	0.4	0.5
Lena	2	5	7	8	3
Airplane	4.5	5.87	6.03	8.23	9.67
Barbara	7	8.01	11.38	15.43	20.42

3) Generating the Marked Decrypted Image: The marked decrypted image means decrypted image in which the LSB planes of A are not decrypted. After generating the marked decrypted image, the content owner can further extract the data and recover original image.

The following outlines the specific steps:

Step 1: Decrypt the LSB-planes of A according to the data hiding key and extract the data until the end label is reached.

Step 2: Extract LN, RN, LM, RM, LP, RP, R_b, x and boundary map from the LSB of marginal area of B. Then, scan B to undertake the following steps.

Step 3: If R_b is equal to 0 it means no black pixels participate in embedding process so go to Step 5.

Step 4: Calculate estimating errors $e'_{i,j}$ of the black pixels. If it is [1, 254], recover the estimating error and original pixel value in a reverse order and extract embedded bits when $e'_{i,j}$ is equal to LN, LM (or LP), RM (or RP) and RN. If it is {0,

255}, refer to the corresponding bit b in boundary map. If b =0, skip this one, else operate like $B_{i,j} \epsilon$ [1, 254]. Repeat this step until the part of payload R_b is extracted. If extracted bits are LSBs of pixels in marginal area, restore them immediately.

Step 5: Calculate estimating errors $e'_{i,j}$ of the white pixels and extract embedded bits and recover white pixels in the same manner with Step 4. If extracted bits are LSBs of pixels in marginal area, restore them immediately.

Step 6: Continue doing Step 2 to Step 5 x-1 rounds on part B and merge all extracted bits to form LSB-planes of A. Thus part B is recovered.

Step7: Replace marked LSB-planes of A with its original bits extracted from part B to get original cover image.

IV. EXPERIMENTS AND RESULTS

We have taken standard images Lena, Airplane, Boat, Barbara for testing purpose. The proposed method is free of any error for all kinds of images. The quality of marked decrypted images is compared in terms of PSNR which is much higher than the previous methods. In addition, another advantage of our approach is the much wider range of embedding rate for acceptable PSNRs. In fact, the proposed method can embed more than 10 times aslarge payloads for the same acceptable PSNR (e.g. PSNR=40 dB) as the previous methods. Table I shows PSNR under various embedding rates. 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. Table II shows the boundary map size of six standard images. In most cases, no boundary map is needed. Even for Peppers image, the largest size is 7550 bits (with a large embedding rate 0.5 bpp by adopting embedding scheme 4 rounds) and the marginal area (512×4×4=8912 bits) is large enough to accommodate it.

Fig. 5 shows comparison of PSNR with the previous and proposed method for hiding text for lena and baboon image.

(a)

(b)

(c)

(d)

Fig. 3: (a) Original image, (b) encrypted image, (c) decrypted image containing messages (embedding rate 0.2 bpp), (d) recovered image

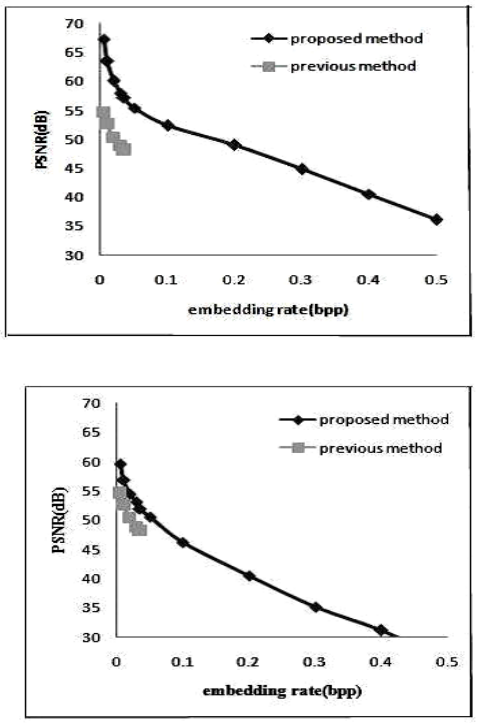

Fig. 4: PSNR comparison for previous method and proposed method for (a) lena (b) barbara.

V. CONCLUSION

The method explained in the paper is tested for six standard images. There is no loss of data or information as the main factor to be considered. By using this approach there is protection of information as data extraction is independent of image restoration. If the user does not have access to decryption key then it's not possible for him to see the cover image. Thus the security of information is more. By using reversible data hiding in encrypted images we can embed huge amount of data or image in the cover image. For single LSB plane also the maximum data hiding capacity is 16384 bytes which increases as number of LSB planes increases. The PSNR is also good for high embedding rates as shown in the table. If we hide image in the reserved space instead of hiding text then also the PSNR remains same i.e. the quality of decrypted image is not affected. Even for single LSB plane the maximum size of image which can be hidden in the cover image is 128×128 which increases with increasing number of planes.

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] Vikash Yadav, Monika Verma, Vandana Dixit Kaushik, Comparative Analysis of Contrast Enhancement Techniques of Different Images, International ConferenceonComputational Intelligence andCommunication Technology (CICT-2016),pp. 76-81, ISBN: 978-1-5090-0210-8, February 12-13, 2016.

[3] MasoudNosrati,RonakKarimi, Mehdi Hariri, Reversible Data Hiding: Principles, Techniques, and Recent Studies, World Applied Programming, Vol (2), Issue (5), May 2012

[4] Vikash Yadav, Monika Verma, Vandana Dixit Kaushik,An approach for pixel based binarization of Gray Images", International Journal of Research in Engineering & Technology (IJRET)eISSN: 2319-1163,pISSN: 2321-7308, Vol. 5, Special Issue 6, pp. 1-4, May 2016.

[5] Dr. J. Jagadeesan, Mr. Balika J. Chelliah, NikhilaNyapathy, Neha Tiwari, Reversible Data Hiding In Encrypted Images Using AES Data Encryption Technique, International Journal of Emerging Research in Management &Technology (Volume-3, Issue-4), April 2014

[6] Lixin Luo, Zhenyong Chen, Ming Chen, Xiao Zeng, and Zhang Xiong "Reversible Image Watermarking Using Interpolation Technique", IEEE Transactions On Information Forensics And Security, Vol. 5, No. 1, March 2010

[7] Wien Hong, Tung-Shou Chen, and Han-Yan WuAn "Improved Reversible Data Hiding in Encrypted Images Using Side Match", IEEE Signal Processing Letters, Vol. 19, No. 4, April 2012

[8] Xinpeng Zhang "Separable Reversible Data Hiding in Encrypted Image", IEEE Transactions on Information Forensics and Security, Vol. 7, No. 2, April 2012

[9] Xinpeng Zhang "Reversible Data Hiding in Encrypted Image", IEEE Signal Processing Letters,Vol.18, No.4, April 2011.

[10] Ahmad-Reza Sadeghi, Thomas Schneider, and Marcel Winandy, "Token - Based Cloud Computing Secure Outsourcing of Data and Arbitrary Computations with Lower Latency", TRUST 2010, LNCS6101, pp . 417–429, 2010.

[11] Trusted Computing Group, "Solving the Data Security Dilemma with Self-Encrypting Drives", May 2010

[12] Hongwei Li, Yuanshun Dai, Ling Tian and Haomiao Yang, "Identity-Based Authentication for Cloud Computing", CloudCom 2009, LNCS 5931, pp. 157–166, 2009

[13] Sven Bugiel, Stefan Nurnberger, Ahmad-Reza Sadeghi, Thomas Schneider, "Twin Clouds: Secure Cloud Computing with Low Latency", CASED, Germany, 2011

[14] Luis M. Vaquero, Luis Rodero-Merino, Daniel Morán, "Locking the sky: a survey on IaaS cloud security", Computing (2011) 91:93–118

[15] Yang Tang, Patrick P. C. Lee, John C. S. Lui, and Radia Perlman, "FADE: Secure Overlay Cloud Storage with File Assured Deletion", 2010

[16] Thuy D. Nguyen, Mark A. Gondree, David J. Shifflett, Jean Khosalim, Timothy E. Levin, Cynthia E. Irvine, "A Cloud-Oriented Cross-Domain Security Architecture", The 2010 Military Communications Conference, U.S. Govt.

[17] Cong Wang, Qian Wang, and Kui Ren, Wenjing Lou, "Ensuring Data Storage Security in Cloud Computing", US National Science Foundation under grant CNS-0831963, CNS-0626601, CNS-0716306, andCNS-0831628,2009.

[18] C. Vara Lakshmi, B.Geeta Rani, "Optimized Reserving Room Approach For Reversible Data Hiding Algorithm Before Encryption On Encrypted Digital Images", International Journal Of Engineering And Computer Science, ISSN: 2319-7242, Volume 4 Issue September 2015, pp. 14319-14327.

[19] Balika J. Chelliah, J.Jagadeesan, SnehaMathur, Saurav Dutta, "Reversible Data Hiding In Encrypted Images Using Improved Encryption Technique", An International Journal of Advanced Computer Technology, ISSN: 2320-0790, Volume 3, Issue 3, March-2014.