ISSN:0975-9646

Separable Reversible Data Hiding Using Blowfish Algorithm

Rakesh D. Desale[#1], Yogesh S. Patil[*2]

[#] *Research Scholar, Department of Computer Science and Engineering, S.S.G.B. College of Engineering and Technology, Bhusawal, Dist. Jalgaon [M.S.], India*

[*] *Assistant Professor, Department of Computer Science and Engineering, S.S.G.B. College of Engineering and Technology, Bhusawal, Dist. Jalgaon [M.S.], India*

Abstract— **Today's world is the world of internet and social networking. The use of data exchange, multimedia applications, and images that contains data over the internet has been tremendously increased. Hence the need of information (data) security becomes necessary and becomes an important issue. Data security is provided to the images containing data by means of data hiding and encryption and decryption. The proposed work in this paper presents a novel scheme for separable reversible data hiding (SRDH) algorithm that can recover the original image without any kind of distortion from the marked image after extraction of hidden data. It also discusses the problem transferring redundant data over an insecure channel. The proposed work focuses on image encryption; data embedding at sender side and data extraction and/or image recovery at receiver side. The work provides a scheme in which data and image can be extracted in reversible manner without any error. This paper presents an encryption and decryption of images using a well known secret key block cipher algorithm which is called as 64 bit blowfish algorithm that designed to increase the security and to improve the performance. The additional data is embedded (hide) into an image using another well known technique called LSB method. The blowfish algorithm uses a variable length key from 32 bits to 448 bit. It is a Feistel network that simply iterates the encryption function 16 times. By using Blowfish and LSB together, the PSNR of the decrypted image is improved. Due to the strong security provided by blowfish algorithm, the receiver can extract the additional embedded data and original image without any loss.**

Keywords— **Blowfish, Data embedding, Distortion, Image decryption, Image encryption, Image recovery, Payload, PSNR, Reversible (lossless) data hiding, Separable Reversible data hiding.**

I. INTRODUCTION

Day by day, the use of internet has increased all over the world. The huge amount of data and images are available for daily communication over the internet. Even the secret data can be transferred by hiding into images over the internet. So there is a need to provide security by means of authentication to this important data and images. In this regard, the reversible data hiding (RDH), in encrypted images has lot of importance. Reversible data hiding is a technique to embed additional messages into some distortion-unacceptable cover media, such as military or medical images; with a reversible manner so that the original cover content can be perfectly restored after extraction of the hidden message [1]. This technique is also called as lossless, distortion free, or invertible data hiding technique [2].

As an effective and popular means for privacy protection [6], encryption converts the ordinary signal into incomprehensible data, so that the general signal processing typically takes place before encryption or after decryption. However, in some circumstances that a content owner does not trust the service provider, the ability to manipulate the encrypted data when keeping the plain content secret is desired. When the secret data to be transmitted are encrypted, a channel provider without any knowledge of the cryptographic key may compress the encrypted data due to the limited channel resource.

The source is first compressed to its entropy rate using a standard source code [3]. Then, the compressed source is encrypted using one of the many widely available encryption technologies. At the receiver side, decryption is performed first, followed by decompression. Compression of encrypted data has attracted considerable research interest [4].

The traditional way of securely and efficiently transmitting redundant data is to first compress the data to reduce the redundancy, and then to encrypt the compressed data to mask it's meaning [3]. At the receiver side, the decryption and decompression operations are orderly performed to recover the original data. However, in some application scenarios, a sender needs to transmit some data to a receiver and hopes to keep the information confidential to a network operator in provides the channel resource for the transmission. That means the sender should encrypt the original data and the network provider may tend to compress the encrypted data without any knowledge of the cryptographic key and the original data. At receiver side, a decoder integrating decompression and decryption functions will be used to reconstruct the original data.

Many image content encryption algorithms have been proposed. For data security, we must need to encrypt data before it is transmitted. So in this paper we are implementing a strongest and fastest blowfish algorithm for encryption and decryption. Blowfish algorithm is highly secured because of its longer key length. The basic aim behind this algorithm is to get the best security and/or best performance tradeoff over images.

II. NON SEPARABLE REVERSIBLE DATA HIDING

Encryption is an effective means of privacy protection. To share a secret image with other person, a content owner may encrypt the image before transmission. In some cases, a channel administrator needs to add some additional message, such as the origin information, image notation or authentication data, within the encrypted image however he does not know the original image content. It may be also expected that the original content can be recovered without any error after decryption and retrieve of additional message at receiver side [11]. That means a reversible data hiding scheme for encrypted image is desirable [1].

In non separable reversible data hiding scheme, A content owner encrypts the original image using an encryption key, and a data-hider can embed additional data into the encrypted image using a data-hiding key though the receiver does not know the original content. With an encrypted image containing additional data, a receiver may first decrypt it according to the encryption key, and then extract the embedded data and recover the original image according to the data hiding key. Fig. 1 shows the arrangement of non separable reversible data hiding.

In the scheme, the data extraction is not separable from the content decryption. In other words, the additional data must be extracted from the decrypted image, so that the principal content of original image is revealed before data extraction, and, if someone has the data-hiding key but not the encryption key, receiver cannot extract any information from the encrypted image containing additional data.

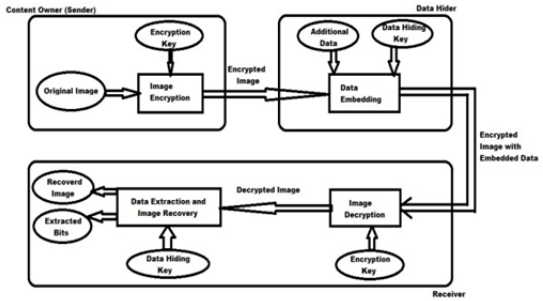


Fig. 1: Non Separable Reversible Data Hiding

III. SEPARABLE REVERSIBLE DATA HIDING

The name itself indicates that it is the reversible data hiding technique with the difference that it is separable. The term separable is used because this scheme separates two things, first is extraction of data which was embedded (additional hidden message) and second is extraction of original cover image which is used to hide the data. This separation exists according to different keys. This is shown in fig. 2.

The separable reversible data hiding scheme consists of three different phases. First is image encryption, second is data embedding and the third is data extraction and/or image recovery. In this scheme, the content owner encrypts the original uncompressed image using an encryption key to produce an encrypted image. Then, the data-hider compresses the least significant bits (LSB) of the encrypted image using a data-hiding key to create a sparse space to accommodate the additional data [6]. At the receiver side, the data embedded in the created space can be easily retrieved from the encrypted image containing additional data according to the data-hiding key. Since the data embedding only affects the LSB, a decryption with the encryption key can result in an image similar to the original version [12]. When using both of the encryption and data-hiding keys, the embedded additional data can be successfully extracted and the original image can be perfectly recovered by exploiting the spatial correlation in natural image.

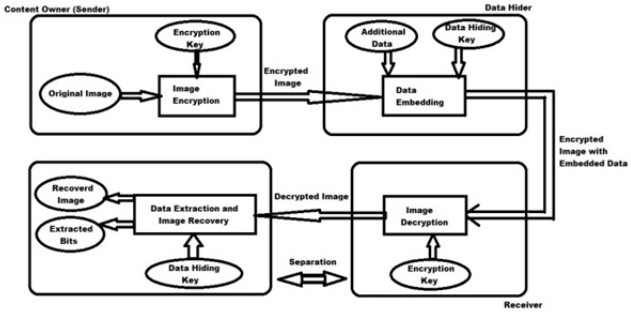


Fig. 2: Separable Reversible Data Hiding

Here, in this scheme, at the receiver side, the three different cases are encountered. First, if the receiver has only the data hiding key and not an encryption key, then he able to extract the additional data only even if he doesn't know the image content. Second, if the receiver has only an encryption key but not a data hiding key, then he can decrypt the received data to get an image which is very similar to the original image, but he cannot extract the embedded additional data. Third and the last case is, if the receiver has both the data hiding key and an encryption key, then he can extract an embedded additional data as well as recover the original image content without any error. These three cases are diagrammatically shown in fig. 3.

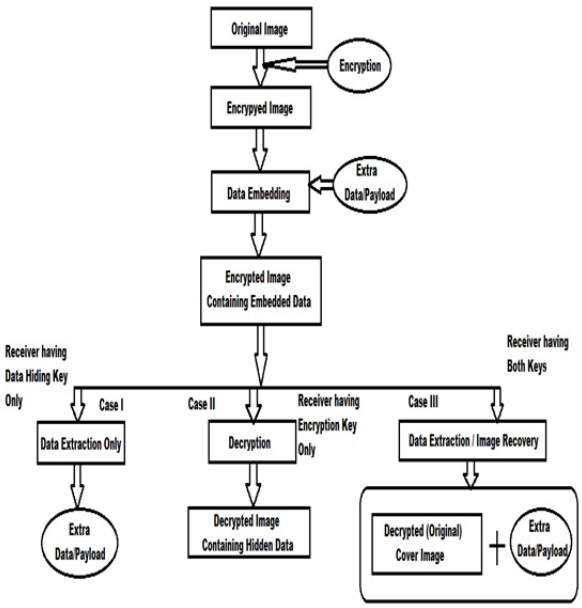


Fig. 3: Three Cases at Receiver Side of Separable Reversible Data Hiding

IV. PROPOSED SCHEME

A new separable reversible data hiding technique is proposed based on the concept of LSB and Blowfish algorithm. The proposed scheme is also a separable reversible data hiding scheme and is also made up of three phases as we see above. These phases are image encryption, data embedding and data extraction / image recovery. Firstly the content owner encrypts the original uncompressed image using an encryption key to produce an encrypted image. In the proposed scheme we use the blowfish algorithm for encryption and decryption of images, because it provides a stronger security as compared to other existing encryption algorithm. Then in second phase, the data-hider compresses the least significant bits (LSB) of the encrypted image using a data-hiding key to create a sparse space to accommodate the additional data [6]. For this we use LSB method for data embedding. At the receiver side, the data embedded in the created space can be easily retrieved from the encrypted image containing additional data according to the data-hiding key. Since the data embedding only affects the LSB, a decryption with the encryption key can result in an image similar to the original version. When using both of the encryption and data-hiding keys, the embedded additional data can be successfully extracted and the original image can be perfectly recovered.

In this new proposed scheme we can use both kind of images i.e. grayscale as well as color images as the cover image which was not in the previous methods. The second new thing in this scheme is that we can hide the data i.e. text as well as an image as a data inside the cover image.

Firstly the sender takes an original uncompressed image as a cover medium in which the secret data is to be transmitted over the network to the receiver. After this, the sender can encrypt the original image using a standard blowfish encryption algorithm. Then sender enters the data to hide into the encrypted image using a LSB method that makes some room inside the image for this secured (additional) data by calculating and compressing the least significant bits of the encrypted pixel. This process is known as data embedding process.

There are three possibilities in the data hiding/embedding process of proposed scheme. The first one is we can embed only an image as a data in encrypted image and send it to the receiver. Second possibility is we can embed a data i.e. only text in an encrypted image and then send it to the receiver. The third possibility is we can embed both data i.e. text as well as image inside encrypted an image. At the receiver side the receiver can extract embedded additional data i.e. text or image or both according to the data hiding key and recover original image according to an encryption key. Now with all this knowledge the complete work of the proposed system is as follows.

A sender who wants to transmit the image called as original uncompressed image first encrypts the image using an encryption key. Then inside this encrypted image, message image or additional data or both can be embedded using a data hiding key to produce a stego image. This stego image consists of three parts, first is the original image, second is the encrypted image and the third is embedded data that may be an image, an additional data or both. After embedding data, at the receiver side, the receiver can perform the reverse operation i. e. decryption and decompression using data hiding and an encryption key to obtain the image similar to the original image without any loss. If the receiver has only the data hiding key then he is able to extract the additional data though he don't know the image content. If the receiver has only the encryption key then he can decrypt the received data to obtain an image similar to the original one, but could not extract the additional data. If the receiver has both the keys i.e. data hiding key and an encryption key then he can extract the embedded additional data as well as can recover the original image without any error.

As stated earlier, the proposed work is consists of three phases i.e. Image Encryption, Data Embedding and Data Extraction and/or Image Recovery. The detail working is as follows:

A. Image Encryption

The first step in the proposed scheme is the image encryption. For that, take one image to which we can call it as an original image. Now this original image is used to hide the message (Text/Image). For this operation, we need to perform an encryption operation on original image. This encryption operation is performed using blowfish algorithm.

1) Blowfish Algorithm: Blowfish algorithm is a 64 bit block cipher that contains a variable length key from 32 bit to 448 bits. This algorithm is used in an application where key does not changed often such as an automatic file encryption. It is observed that when this algorithm is implemented on 32 bit microprocessors having large data caches, the performance is faster than other existing encryption algorithm.

The blowfish algorithm contains two different parts: one is the key expansion part and the other is data encryption part. It is noted that the key expansion converts a key of at most 448 bits into several subkey arrays around 4168 bytes. The data encryption occurs through 16 round Feistel network and each round consists of a key dependent permutation and a key, and a data dependent substitution. All the operations are performed by XORs and additions on 32 bit words.

Blowfish uses large number of subkeys and can be precomputed before any data encryption or decryption. The P-array consists of 18 32-bit subkeys from P1 to P18. There are 4 32-bit S-boxes having 256 entries each as follows.

S1,0,	S1,1,...,	S1,255;
S2,0,	S2,1,..,	S2,255;
S3,0,	S3,1,...,	S3,255;
S4,0,	S4,1,..,	S4,255.

The blowfish has 16 round. Let the input is a 64 bit data element, x. Now the actual algorithm is as follows:

Divide X into two 32-bit halves XL and XR
For i=1 to 16:
XL = XL XOR Pi
XR = F (XL) XOR XR
Swap XL and XR
End for
Swap XL and XR
XR = XR XOR P17
XL = XL XOR P18

Finally, Recombine XL and XR to get the cipher text.
Output X (64-bit data block: cipher text)

B. Data Embedding

After image encryption phase, the next phase is data embedding phase in which the secured data (Text/Image) is embedded into an encrypted image by implementing a well known LSB method. In this proposed work, both grayscale and color images are used. If we take grayscale image to embed then the LSB algorithm is working as mentioned in [6]. If we take color image then the working is slightly different. Each pixel in color images will have three planes i.e. Red(R), Green(G) and Blue(B). The pixel values of these color components will be in the range of [0 255]. The message bits are embedded in all these three planes and can be recombined to form the original color image. Here the message bits are embedded in every Red component in the RGB plane [14]. The rest of the work is same as in [6]. All the calculation regarding embedded rate are same as in [3], [6] and [13]. If we take text to embed in an image then working of LSB is same as in [13].

C. Data Extraction/Image Recovery

In this last phase, we have to consider the three cases as we discuss earlier i.e. the receiver has only the data hiding key, only an encryption key, and both data hiding as well as encryption keys.

If we consider case one i.e. only data hiding key, then the receiver can extract additional data though he don't know the image content. For data extraction the LSB is used. If the text is to be extracted then the logic is same as in [13]. If an image is to be extracted then the logic is same as in [3], [6].

If we consider case two i.e. only an encryption key, then the receiver can decrypt the received data to obtain an image similar to the original image, but cannot extract the additional embedded information. For decryption of image again we use the blowfish algorithm. Here decryption is same as an encryption with only difference is using the P-array. The P1 to P18 are used in reverse order.

If we consider case three, then the receiver can extract the additional data and recover the original image without any error. To extract data the method is same as in case one above and to recover the image the method is same as in case two.

V. EXPERIMENTAL RESULT

In our experiment, we use various images of standard size and different format (.jpg, .bmp, .png) for testing as cover image and hidden image. Here in the experiment, we are interesting in calculating the PSNR (Peak Signal to Noise Ration) of images. The PSNR, in simple language, is nothing but the difference between the two images for example, the difference between Original Image and the decrypted image after extracting the hidden data. It gives the ratio of corrupting noise produced in the original image after extraction of hidden data. From the study of various existing algorithm, it is found that if the difference is very close to zero or near about 1% then the PSNR will be around 38-39 dB. In our experiment, if the images are grayscale images then obtained resulting PSNR will be nearly same as in [3].

If the images are color images then the resulting PSNR that obtained by our experiment is near around 43 dB. That means PSNR value is improved in our system. Our proposed system is very close to the zero difference according to the PSNR value. The PSNR value is calculated by using formula:

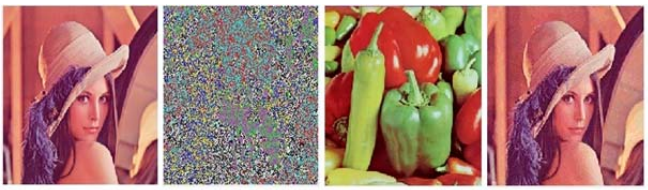
$$\text{PSNR (dB)} = 10 * \log10 (256^2 / \text{MSE}) \quad (1)$$

Where MSE is Mean Square Error and is calculated by,

$$\text{MSE} = \sum_{i=1}^{x} \sum_{j=1}^{y} \frac{\left|A_{ij} - B_{ij}\right|\text{^}2}{x * y} \quad (2)$$

Where x: width of image
y: height of image
By multiplying x & y, we get the number of pixels.

For testing the result, we take a standard lena image of size 512X512 as original cover image shown in fig. 4(a). After that, we encrypt the original cover image, as fig. 4(b). Then we take another image to hide into the encrypted cover image in fig. 4(c). Then embed this image into encrypted cover image to produce an image to which we call it as a stego image, as in fig. 4(d). Now, this stego image contains an original cover image, encrypted image and an embedded message (image). We can also embed the text as well into the stego image along with another hidden image as shown in fig. 5.



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

(e) (f) (g)

Fig. 4: (a) Original Lena Image (b) Its Encrypted version (c) Message Image to be hide (d) Stego Image containing Original, Encrypted and Message Image (e) Extracted Encrypted Image from Stego Image (f) Extracted Cover Image (g) Extracted Message Image

From the stego image, we then decrypt the encrypted image and an original image, according to the key, shown in fig. 4(e) & fig. 4(f), respectively. Then at last we extract an additional embedded data (text and hidden image) from the decrypted image, as shown in fig. 4(g) and fig. 5.

Fig. 5: Data Embedding and Extraction

From the experimental result, it is clear that, if we had a data hiding key, then we could extract the additional data from an encrypted image containing embedded data. If we had an encryption key then we can directly decrypt an encrypted image containing embedded data. If we had both keys, then we could successfully extract an embedded data and perfectly restored the original image from the encrypted image containing embedded data.

From the tested color image, the PSNR value of the decrypted image that we obtain is 43.6 dB, which shows that the improved PSNR value, and improved result of the algorithm. The image recovered using blowfish algorithm is same as the original image.

VI. CONCLUSION

In this paper, a separable reversible data hiding using blowfish algorithm is proposed, which consists of image encryption, data embedding and data extraction/image recovery phases. The receiver can extract an additional data from encrypted image having data hiding key only. The receiver can extract an image similar to the original one having an encryption key only. If the receiver has both keys then he can extract additional data and perfectly recover an original image from an encrypted image containing data. It is also observed from the result that, the PSNR value is improved using blowfish algorithm. The blowfish algorithm is fastest in nature in data processing and provides a strong security compared to other algorithms. The blowfish algorithm is highly secured because of its longer key. It is also found that blowfish algorithm has no security weak point yet, so that it can be considered as a standard encryption algorithm. We can embed only an image, only a text or both text and image in an encrypted image. By using blowfish algorithm, the performance is highly improved and more security is provided to the important data. In the experiment, it is also found that, both color and grayscale images can be used. However, it is not possible to embed grayscale image into color image or vice versa. This deserves further investigation for future work. Also this blowfish algorithm can be used for encryption and decryption of important data in video and audio as well in future.

REFERENCES

[1] X. Zhang, “Reversible data hiding in encrypted image," IEEE Signal Process, vol. 18, no. 4, pp. 255-258, Apr 2011.

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

[3] V. Suresh, C. Saraswathy, “Separable Reversible Data Hiding Using Rc4 Algorithm”, proceedings of the 2013 International Conference on Pattern Recognition, Informatics and Mobile Engineering (PRIME), pp. 164-168, Feb 2013.

[4] W. Liu, W. Zeng, L. Dong, and Q. Yao, “Efficient compression of encrypted grayscale images”, IEEE Trans. Image Process., vol. 19, no. 4, pp. 1097–1102, Apr. 2010.

[5] Mohammad Awrangjeb, “An Overview of Reversible Data hiding”, ICCIT, Jahangirnagar University, Bangladesh, pp. 75–79, Dec. 2003.

[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] Shilpy Mukherjee, A. R. Mahajan, “Review on Algorithms and Techniques of Reversible Data Hiding”, International Journal of Research in Computer and Communication Technology, vol. 3, issue 3, pp. 291–295, Mar 2014.

[8] X. Zhang, “Reversible data hiding with optimal value transfer,” *IEEE Trans. Multimedia*, vol. 15, no. 2, pp. 316–325, 2013.

[9] 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.

[10] Zhenxing Qian, Xinpeng Zhang, and Shuozhong Wang, “Reversible Data Hiding in Encrypted JPEG Bitstream”, *IEEE Transactions on Multimedia*, VOL. 16, NO. 5, pp. 1486–1491, August 2014.

[11] Vinit Agham, Tareek Pattewar, “A Survey on Separable Reversible Data Hiding Technique", IMACST, Vol. 4, No. 1, pp. 9–13, May 2013.

[12] J. Tian, “Reversible data embedding using a difference expansion", *IEEE Trans. Circuits Syst. Video Technology*, Vol. 13, No. 8, pp. 890-896, Aug 2003.

[13] Vinit Agham, Tareek Pattewar, “A Novel Approach Towards Separable Reversible Data Hiding Technique”, *IEEE International Conference on Issues and Challenges in Intelligent Computing Techniques*, pp. 776-780, 2014.

[14] Shilpa Sreekumar, Vincy Salam, “Advanced Reversible Data Hiding with Encrypted Data”, *International Journal of Engineering Trends and Technology (IJETT)*, Vol. 13, No. 7, pp. 310-313, July 2014.