Data Hiding using security id, index variable and 14 squares Substitution Cipher

[1]Milind S. Deshhkar, [2]P.S.Kulkarni

[1]Computer Science & Engineering department, [2]Information Technology Department
[1,2]RCERT, Chandrapur, Maharashtra, India
Email: [1]pmilind_deshkar@yahoo.com, [2]kulkarnips1811@gmail.com

Abstract — In this paper the technique of cryptography and steganography. is proposed to hide useful data. Firstly, the data is encrypted using the proposed substitution cipher algorithm, and then the cipher text is embedded in the carrier image. We will be using only 2-bit combination at an instant. An index variable is found and then depending on its value, the selection of the bit position of the carrier image is decided.. Here the 6th bit means the LSB minus two locations, the 7th bit means the LSB minus one location and 8th bit means the least significant bit (LSB) location. The index value will change from 0 to 1 or 1 to 2 or 2 to 0 after each embedding. Before sending the embedded image the data is locked with a security id. Due to the security lock and the image bytes which depend on the size of the cipher text, this method is a stronger approach.

I. INTRODUCTION

The secrecy of the information is the main issue in the era of information technology and communication. There are mainly two techniques to achieve secrecy. Those are:

1. Cryptography .

2. Steganography.

Cryptography is a technique in which the secret information is encrypted in an new form, so as to prevent it from an intruder. It is of 2 basic types :

1. Public key cryptography

2. Secret key cryptography.

Steganography is a technique of information hiding in which tah data is usually hidden behind an image, audio or video file in such a way that the intruder cannot judge that the secret communication is going on.

II. .EXISTING IMAGE STEGANOGRAPHY METHODS :

In this paper the authors Gandharba Swain and Saroj Kumar Lenka proposed a scheme for secret communication between Alice and Bob by using both cryptography and steganography. They have used 3-bit variations in it. Firstly, they encrypted the secret message using the 12-square cipher algorithm and implemented using Java programming language. A method for steganography using twelve square substitution ciphers which includes both alphabet and digit but the alphabet ‘q’ and some special character ‘space’ is missing. [1]

Discrete Wavelengths Transform (DWT). The frequency domain transform is Haar-DWT. It consists of 2 operations one is horizontal operation and the other is vertical operation. At first, scan the pixels from left to right in horizontal direction then perform the addition and subtraction operations on neighboring pixels and then store the sum on the top and the difference on the bottom. Benefit of this is security is maintained as well as no message can be extracted without the “key matrix” and decoding rules. This technique can be used to reduce the extra data in the stego-image compress the size of key matrix as far as possible..[2]

A new steganography approach for data hiding is proposed in which the Least Significant Bits (LSB) insertion to hide data within encrypted image data was used. The binary representation of the hidden data is used to overwrite the LSB of each byte within the encrypted image randomly. The experimental results showed that the correlation and entropy values of the encrypted image before the insertion are similar to the values of correlation and entropy after the insertion. Since the correlation and entropy have not changed, the method offered a good concealment for data in the encrypted image, and reduces the chance of the encrypted image being detected..[3]

By representing the ASCII code decimal value of each character, in the secret message, as a set of separated

single decimal-digit, also represent each decimal pixel value in the stego-image as a set of separated single decimal-digit. The technique creates a matching list between the decimal-digits of the characters in the secret message with the decimal-digits of the pixels in the stego-image. The technique compresses the created matching list to be as small as possible to embed it in the unused file space at the end of the stego-image file. The results show that the technique provides more security against visual attack because it does not make any changes in the pixel of the stego-image.[4]

This paper proposed an approach in such way that, data is encrypted using Extended Substitution Algorithm and then this cipher text is concealed at two or three LSB positions of the carrier image. This algorithm covers almost all type of symbols and alphabets. The encrypted text is concealed variably into the LSBs. Therefore, it is a stronger approach. The visible characteristics of the carrier image before and after concealment remained almost the same. [5]

A new algorithm to hide data inside mage using steganography technique is proposed. Here a bitmap (bmp) image will be used to hide the data. Data will be embedded inside the image using the pixels. Then the pixels of stego-image can be accessed back in order to retrieve back the hidden data inside the image. Two stages are involved. The first stage is to come up with a new steganography algorithm in order to hide the data inside the image and the second stage is to come up with a decryption algorithm using data retrieving method in order to retrieve the hidden data that is hided within the stego -image. Advantage of this technique is SIS maintains privacy, confidentiality and accuracy of the data. [6]

Here a highly efficient steganography protocol is proposed. It is based on hamming codes, the embedding and
the retrieval algorithm which have the same computational cost. The main idea behind this technique is to use a product code of two hamming codes with goal of improving the embedding rate .[7]

This technique gives two different schemes are investigated. The first one is derived from a blind watermarking scheme. The second scheme is designed for steganography such that perfect security is achieved, that means the relative entropy between cover data and stego data tends to zero. In this technique, information embedding has been investigated in particular in the context of digital watermarking. For digital watermarking, information embedding techniques have to be designed such that subsequent processing does not destroy the embedded information. This property makes digital watermarking technology also attractive for steganography when information embedding is followed by lossy compression. Advantage of this technique the performance of the schemes is compared with respect to security, embedding distortion and embedding rate. [8]

III. THE PROPOSED METHODOLOGY

In the paper, we are proposing a new scheme to hide the data effectively behind a carrier image. The communication between Alice and Bob is taken to a new dimension, by applying the method of both Cryptography and Steganography.

The mixed techniques of cryptography and steganography provide a stronger approach of communication. The index variable and cryptography allows secure communication and it requires a key to read the information. An attacker is not able to remove the encryption but it is easy to modify the file and making it unreadable for the intended recipient.

Steganography allows secure communication It cannot be removed and it requires significantly altering the data in which it is embedded. The embedded data will be confidential until an attacker is able to find a way to detect it.

The overall work is done in 3 modules

1. First module Data Mixing(Sender side)

   i. The secret information and an image to be sent is selected. .

   ii. The cipher text to be embedded.

   iii. The resultant image after embedding

2. Second module, Embed the cipher text bit into image depending upon index variable and security id.

3. Third module Data Extracting (Receiver side)

   i. After the embedded image and the security id is received, the information retrieval process is performed.

A. Proposed Algorithm

- Step1 - Transform the carrier image to binary
- Step2- Apply the Substitution Cipher to get the cipher text of the secret message.
- Step3- Convert the cipher text to binary.
- Step4- The length of carrier image is sufficient enough to conceal the cipher text.

- Step5- Embed the cipher into the cover image using proper embedding process.
- Step6- Attach the security id along with the image.
- Step7- Now send the resultant image to receiver.
- Step8- The receiver applies the reverse process to decipher the information.

In the paper[1], the author had used 12 squares substitution ciphers, which includes alphabets, numerals and special symbols. The problem with this method was that it missed some of the alphabets, like 'q' and some of the special characters. To accommodate all characters and alphabets, we are here proposing a method in which, we are using 14 squares and substitution cipher methods to get the better end- results. Squares 1 to 8 are taken for upper and lower case alphabets and squares 9 to 14, are used for special symbols. In our algorithm, there are 9X6 square matrix of alphabets and same size is for special characters and numerals. Each of the 9 by 6 matrices contains the letters of the alphabet (upper case and lower case) and another six 6 by 7 matrices arranged in squares for digits and special characters, as shown in table-2. All the special characters and digits from your desktop/laptop keyboard are included in this table. So the following describes that how the table 1 is prepared:

In square-1, we have taken fifty two alphabets and two special characters @ and ?, out of which twenty six are capital letters and twenty six are small letters In each row we have arranged nine alphabets and each column contains six alphabets. Square-2 is made from square-1 by taking the first row of square-1 to sixth row place and other rows one position up. Similarly square-3 is created from square-2 by taking the first row of square-2 to sixth row place and other rows one position up. The same thing about square-4 which is created from square-3 by taking the first row of square-3 to sixth row place and other rows one position up. In square-5, We have converted rows into column and inter changed first and last alphabets. The same steps follows in square-6 to square-8 by taking first row of previous square and to sixth row place and other rows one position up. The plain text is read from left to right. If the character is an alphabet it refers to table-1, otherwise if it is a number or a special character it refers to table-2. The conversion from plain text to cipher text is performed as follows:

The plain text- cipher text combination for first alphabet is taken from the first row of square-1 and square-5.

The plain text- cipher text combination for second alphabet is taken from the same row of square-2 and square-6.

The plain text- cipher text combination for third alphabet is taken from the same row of square-3 and square-7.

The plain text- cipher text combination for fourth alphabet is taken from the same row of square-4 and square-8 and so on.

Table 1: Plain Text and Cipher Text from square 1 to square 8 (Alphabets)

square1
{ "s", "t", "u", "v", "w", "x", "y", "z", "A" } { "B", "C", "D", "E", "F", "G", "H", "I", "J" } { "K", "L", "M", "N", "O", "P", "Q", "R", "S" } { "T", "U", "V", "W", "X", "Y", "Z", "@", "?" } { "a", "b", "c", "d", "e", "f", "g", "h", "i" } { "j", "k", "l", "m", "n", "o", "p", "q", "r" }

square2
{ "B", "C", "D", "E", "F", "G", "H", "I", "J" } { "K", "L", "M", "N", "O", "P", "Q", "R", "S" } { "T", "U", "V", "W", "X", "Y", "Z", "@", "?" } { "a", "b", "c", "d", "e", "f", "g", "h", "i" } { "j", "k", "l", "m", "n", "o", "p", "q", "r" } { "s", "t", "u", "v", "w", "x", "y", "z", "A" }

square3
{ "K", "L", "M", "N", "O", "P", "Q", "R", "S" }
{ "T", "U", "V", "W", "X", "Y", "Z", "@", "?" }
{ "a", "b", "c", "d", "e", "f", "g", "h", "i" }
{"j", "k", "l", "m", "n", "o", "p", "q", "r" } { "s", "t", "u", "v", "w", "x", "y", "z", "A" }
{ "B", "C", "D", "E", "F", "G", "H", "I", "J" }

square4
{ "T", "U", "V", "W", "X", "Y", "Z", "@", "?" } { "a", "b", "c", "d", "e", "f", "g", "h", "i" } { "j", "k", "l", "m", "n", "o", "p", "q", "r" } { "s", "t", "u", "v", "w", "x", "y", "z", "A" }
{ "B", "C", "D", "E", "F", "G", "H", "I", "J" } { "K", "L", "M", "N", "O", "P", "Q", "R", "S" }

square5
{ "r", "y", "E", "K", "Q", "W", "a", "g", "m" } { "t", "z", "F", "L", "R", "X", "b", "h", "n" } { "u", "A", "G", "M", "S", "Y", "c", "i", "o" } { "v", "B", "H", "N", "T", "Z", "d", "j", "p" } { "w", "C", "I", "O", "U", "@", "e", "k", "q" }
{ "x", "D", "J", "P", "V", "?", "f", "l", "s" }

square6
{ "t", "z", "F", "L", "R", "X", "b", "h", "n" } { "u", "A", "G", "M", "S", "Y", "c", "i", "o" } { "v", "B", "H", "N", "T", "Z", "d", "j", "p" } { "w", "C", "I", "O", "U", "@", "e", "k", "q" } { "x", "D", "J", "P", "V", "?", "f", "l", "s" } { "r", "y", "E", "K", "Q", "W", "a", "g", "m" }

square7
{ "u", "A", "G", "M", "S", "Y", "c", "i", "o" } { "v", "B", "H", "N", "T", "Z", "d", "j", "p" } { "w", "C", "I", "O", "U", "@", "e", "k", "q" } { "x", "D", "J", "P", "V", "?", "f", "l", "s" } { "r", "y", "E", "K", "Q", "W", "a", "g", "m" } { "t", "z", "F", "L", "R", "X", "b", "h", "n" }

square8

{ "v", "B", "H", "N", "T", "Z", "d", "j", "p" } { "w", "C", "I", "O", "U", "@", "e", "k", "q" } { "x", "D", "J", "P", "V", "?", "f", "l", "s" } { "r", "y", "E", "K", "Q", "W", "a", "g", "m" } { "t", "z", "F", "L", "R", "X", "b", "h", "n" } { "u", "A", "G", "M", "S", "Y", "c", "i", "o" }

Table 2: Plain Text and Cipher Text from square 9 to square 14 (Digit and Special Characters)

square9

{ "0", "1", "2", "3", "4", "5", "6" }
{ "7", "8", "9", "'", "~", "!", "@" }
{ "#", "$", "%", "^", "&", "*", "(" }
{ "0", "_", "-", "+", "=", "{", "[" }
{ "}", "]", ";", ":", "", "'", "" }
{ "|", "<", ",", ">", ".", "?", "/" }

square10

{ "7", "8", "9", "'", "~", "!", "@" }
{ "#", "$", "%", "^", "&", "*", "(" }
{ "0", "_", "-", "+", "=", "{", "[" }
{ "}", "]", ";", ":", "", "'", "" }
{ "|", "<", ",", ">", ".", "?", "/" }
{ "0", "1", "2", "3", "4", "5", "6" }

square11

{ "#", "$", "%", "^", "&", "*", "(" }
{ "0", "_", "-", "+", "=", "{", "[" }
{ "}", "]", ";", ":", "", "'", "" }
{ "|", "<", ",", ">", ".", "?", "/" }
{ "0", "1", "2", "3", "4", "5", "6" }
{ "7", "8", "9", "'", "~", "!", "@" }

square12

{ "0", "6", "!", "&", "+", ";", "<" }
{ "1", "7", "@", "*", "=", ":", "," }
{ "2", "8", "#", "(", "{", "", ">" }
{ "3", "9", "$", ")", "[", "'", "." }
{ "4", "'", "%", "_", "}", "", "?" }
{ "5", "~", "^", "-", "]", "|", "/" }

square13

{ "1", "7", "@", "*", "=", ":", "," }
{ "2", "8", "#", "(", "{", "", ">" }
{ "0", "6", "!", "&", "+", ";", "<" }
{ "3", "9", "$", ")", "[", "'", "." }
{ "4", "'", "%", "_", "}", "", "?" }
{ "5", "~", "^", "-", "]", "|", "/" }

square14

{ "1", "7", "@", "*", "=", ":", "," }
{ "2", "8", "#", "(", "{", "", ">" }
{ "3", "9", "$", ")", "[", "'", "." }
{ "4", "'", "%", "_", "}", "", "?" }
{ "5", "~", "^", "-", "]", "|", "/" }
{ "0", "6", "!", "&", "+", ";", "<" }

For example if the plain text is: My name is msd9

Its cipher text would be: Ga VwPU qr PrO05!

IV. THE EMBEDDING PROCESS

The carrier image is transformed into binary form, where each pixel becomes of 1 byte. The cipher text of the secret message is also converted into bytes and then we calculate the number of bytes, suppose it is n. Divide it by 2 and find the remainder. This is known as an index variable. The value index=0, corresponds to 6th and 7th bit locations, index=1 corresponds to 7th and 8th bit locations, of any pixel (byte) of the digital image. If present value of index=0 hide the two bits of cipher text in 6th and 7th bit locations of the present pixel (byte), and next value of index is 1 for the next pixel. If present value of index=1 hide the two bits of cipher text in 7th and 8th bit locations of the present pixel (byte), and next value of index is 0 for the next pixel.

Table 3: BYTE Selection using Index Variable

Carrier file Byte	Operation	Location	Index
Byte A	Embed(11)	6^{th} and 7^{th}	1
Byte B	Embed(00)	7^{th} and 8^{th}	0
Byte C	Embed(10)	6^{th} and 7^{th}	1
Byte D	Embed(11)	7^{th} and 8^{th}	0
Byte E	Embed(01)	6^{th} and 7^{th}	1
Byte F	Embed(11)	7^{th} and 8^{th}	0
Byte G	Embed(10)	6^{th} and 7^{th}	1
Byte H	Embed(10)	7^{th} and 8^{th}	0
Byte I	Embed(10)	6^{th} and 7^{th}	1
Byte J	Embed(10)	7^{th} and 8^{th}	0
Byte K	Embed(10)	6^{th} and 7^{th}	1
Byte L	Embed(10)	7^{th} and 8^{th}	0
And so on…			

After embedding the text to the carrier image, the security id is entered. This security id is sent through communication to the receiver. This security id is needed in the reverse process to extract the secret message. Consider the cipher text to be sent is:

10111001 01001011 1100 0010

The number of bytes of the carrier image is 3,so n=3. The index variable is calculated, by performing n/2 and taking its remainder. So the value of index variable is =1. Consider the bytes of the digital image as A,B,C.

Using table-3 we embedded the data bits in the byte A of the carrier file 11 in 6^{th} & 7^{th} bit locations , and next value of index becomes 0.

We then embed the data bits in the byte B of the carrier file 00 in 6^{th} and 7^{th} bit locations, and the value of index variable becomes 1. Now we embed the next two data bits in the byte C of the carrier file 00 in 7^{th} and 8^{th} bit locations, and the value of index variable becomes 1 and so on. Some bytes in every image representing the image features should not be altered. In JPEG images of size more than one Mega Bytes, there will be a

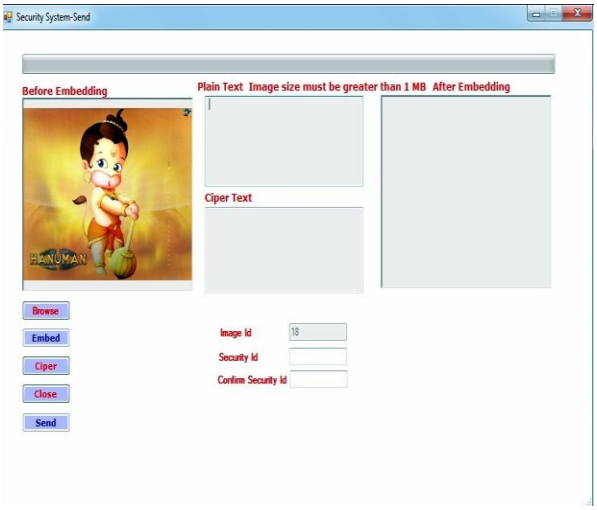
maximum of hundred bytes carrying the image characteristics, if we modify these bytes the image will be disturbed. So these bytes should not be altered. For different image formats like BMP, JPG, TIF it is different.

V. THE DATA EXTRACTION PROCESS

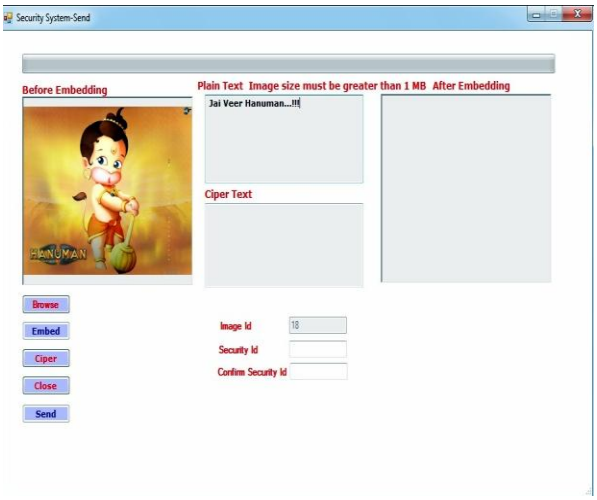
At the receiver side, the message is extracted by applying the security code first and the algorithm and extraction in the reverse to get back the secret code.

VI. THE EXPERIMENTAL RESULTS

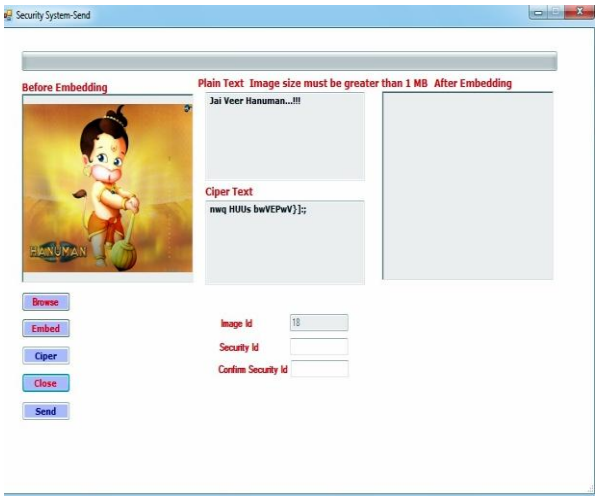
A. Selection of an image



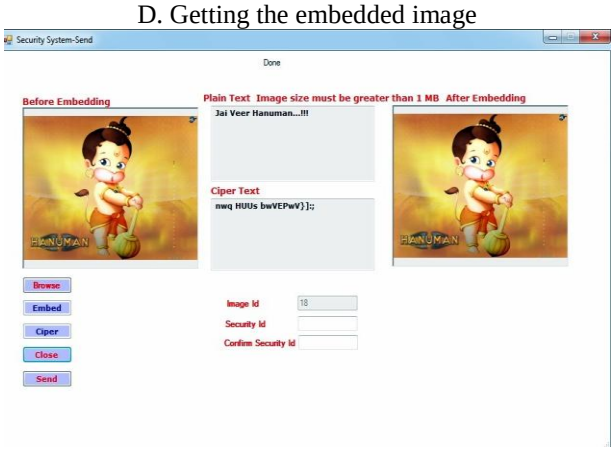
B. Entering plain text



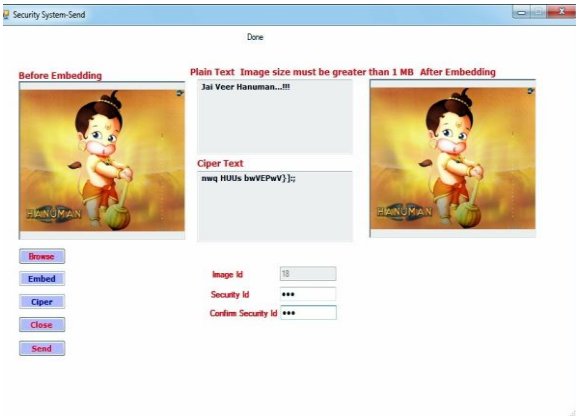
C. Conversion into cipher text



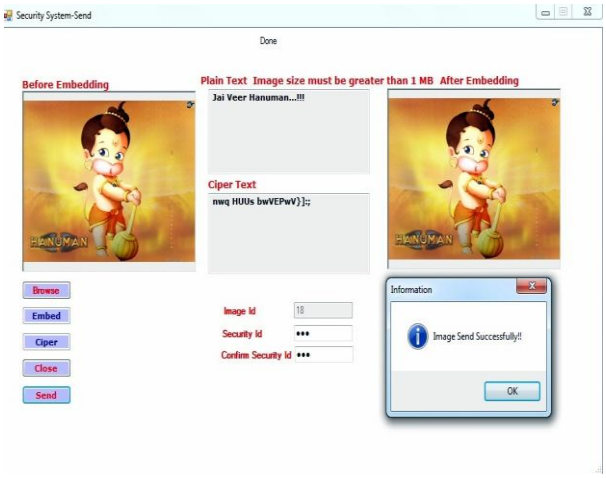
D. Getting the embedded image



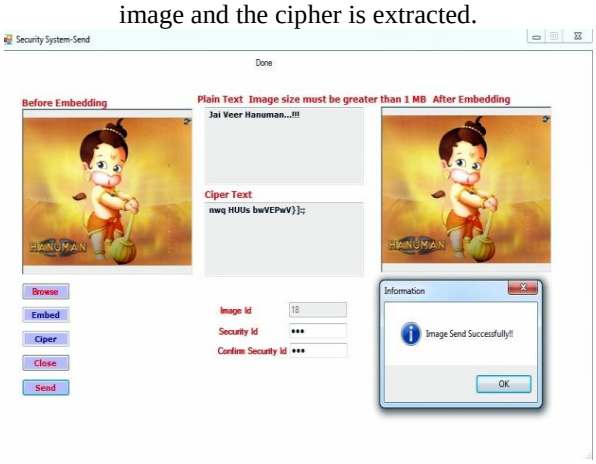
E. Entering security code



F. Sending the image successfully



G. After entering the security id at the receiving end, the image and the cipher is extracted.



H..Receiving the secret message i.e., the plain text

CONCLUSION

It is observed that the algorithm works fine. It provides two levels of security One at the cryptography level and the other at the steganography level. If at all the intruder suspects it is very difficult for him to steal the data. The size of the image before and after embedding process is unchanged. The role of an index variable and the security code , makes this approach a stronger approach.

REFERENCES

[1] Gandharba Swain and Saroj Kumar Lenka, “Steganography Using the Twelve Square Substitution Cipher and an Index Variable”, 3rd IEEE International Conference on Electronics Computer Technology (ICECT),Vol.3, Pg. No. 84-88/2011

[2] Po-Yueh Chen and Hung-Ju Lin, “A DWT Based Approach for Image Steganography”, International Journal of Applied Science and Engineering Vol. 4, No. 3: Pg. no. 275-290 / 2006

[3] Mohammad Ali Bani Younes and Aman Jantan , “A New Steganography Approach for Image Encryption Exchange by Using the Least Significant Bit Insertion”, IJCSNS International Journal of Computer Science and Network Security, Vol.8 No.6/ June 2008

[4] Mohammed Abbas Fadhil Al-Husainy, “A New Image Steganography Based on Decimal-Digits Representation”, Computer and Information Science Journal, Vol. 4, No. 6/ November 2011

[5] R.S. Gutte, Y.D. Chincholkar and P.U. Lahane, “STEGANOGRAPHY FOR TWO AND THREE LSBs USING EXTENDED SUBSTITUTION ALGORITHM”, ICTACT Journal on Communication Technology, Vol.04, Issue. 01/ March 2013

[6] Rosziati Ibrahim and Teoh Suk Kuan, “ Steganography Algorithm to Hide Secret Message inside an Image”, Computer Technology and Application Vol. 2 pp. 102-108

[7] H. Rifa-Pous and J. Rifa, “Product Perfect Codes and Steganography”, Digital Signal Processing, Vol.19, 2009, pp. 764-769.

[8] Joachim J. Eggers, R.Bauml and Bernd Girod,“ A Communications Approach to image steganography”, Proc. of SPIE Volume 4675, San Jose, Ca, 2002, pp. 1-12.

[9] Iwata, M., Miyake, K., and Shiozaki, A. , “Digital Steganography Utilizing Features of JPEG Images”,IEICE Transfusion Fundamentals, E87-A, 4:929-936/ 2004.

[10] A. Sinha, K. Singh, "A technique for image encryption using digital signature", Optics Communications, vol.218, no. 4, 2003, pp.229-234.

[11] EE. Kisik Chang, J. Changho, L. Sangjin, "High Quality Perceptual Steganographic Techniques,", Springer, Vol. 2939, 2004, pp.518-531.

[12] G. C. Kessler, "Steganography: Hiding Data Within Data," Windows & .NET Magazine/ September 2001.

[13] Alvaro Martin, Guillermo Sapiro, & GadielSeroussi,"Is Steganography Natural", IEEE Transactions on Image Processing, 14(12), 2040-2050/ 2005

[14] P.Mohan Kumar, & D.Roopa, "An Image Steganography Framework with Improved Tamper Proofing", Asian Journal of Information Technology, 6(10), 1023-1029. ISSN: 1682-3915/ 2007

[15] Hardik J. Patel and Preeti K. Dave, "Least Significant Bits Based Steganography Technique", International Journal of Electronics Communication and Computer Engineering (IJECCE) pp.44-50/ 2012

[16] Ross J. Anderson and Fabian A.P. Petitcolas, "On The Limits of steganography", IEEE Journal of selected Areas in communication, Vol.16, No.4, pp. 474-481/ 1998.

[17] Y. K. Jain and R. R. Ahirwal, "A Novel Image Steganography Method with Adaptive Number of Least Significant Bits Modification Based on Private Stego Keys", International Journal of Computer Science and Security, vol. 4, no. 1, pp. 40-49/,2010

[18] M. Juneja and P. S. Sandhu, "Designing of Robust Steganography Technique Based on LSB Insertion and Encryption", Proceedings of International Conference on Advances in Recent Technologies in Communication and Computing, pp. 302-305/ , 2009

◈◈◈