ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 2, February 2016**

# Implementation of Security through hiding the Data within an Image with the help of Least Significant Bit replacement

Aritra Dutta[1], Souvik Neogi[1], Prof. Dr Pranam Paul[2]

MCA Final Year Student, Narula Insitute of Technology, Agarpara, West Bengal, India[1]

HOD, Dept. of Computer Application, Narula Institute of Technology, Agarpara, West Bengal, India[2]

**ABSTRACT**: Steganography is a branch of information hiding. It allows the people to communicate secretly. As increasingly more material becomes available electronically, the influence of steganography on our lives will continue to grow. Much confidential information was leaked to a rival firm using steganographic tools that hide the information in music and picture files. The application of steganography is an important motivation for feature selection. To ensure the security against the steganalysis attack, a new steganographic algorithm for 8bit (grayscale) or 24bit (color image) is presented in this paper, based on Logical operation. Algorithm converts the secret message to a binary sequence and hides the Most Significant Bits (MSB) of embedded message into Least Significant Bits (LSB) of cover image. In this n LSB of cover image, from a byte is replaced by n MSB of secret message.[1,3]

**KEYWORDS:** Steganography , encryption, steganalysis , Data hiding, Embedding Data, Information Security, Least Significant Bit(LSB) , BMP image file .

## I . INTRODUCTION

Steganography conveys the information secretly by concealing the very existence of information in some other media files such as image, audio, video, or text files over non-secure communication channels. The information to be concealed is called the secret message or simply the secret; the content used to embed information is called the cover media, and the cover along with the secret is called the stego media [3]. Image steganography has come quite far in recent years with the development of fast, powerful graphical computers. This paper presents a Steganography method based on the spatial domain for encoding extra information in an image by making small modifications to its pixels. An image in a computer is an array of numbers that represent light intensities at various points (pixels).
Digital images are stored in either 24-bit (true color images) or 8-bit per pixel files. A common image size is 640 × 480 pixels and 256 colors (or 8 bits per pixel). Such an image could contain about 300 Kb of data. Such large size images should be avoided since the attention when sending over a network or the Internet. Here, each pixel is represented as a single byte, and the pixel's value is between 0 and 255. Grey-scale images are preferred because the shades are changed very gradually between palette entries. This increases the image's ability to hide information. The most well known techniques to data hiding in images are least significant bit (LSB) substitution, and masking & filtering techniques. LSB is a simple approach to embedding information in an image. [4].

## II. ALGORITHMS

### ii.1. ALGORITHM FOR HIDING THE SECRETE DATA IN A COVER IMAGE

**Least Significant Bit (LSB-1) Replacement Method**
The following steps illustrate how this method is used to hide the secret data in cover image which is a Bitmap Image file .
**STEP 1**: Convert the message text(which will be hidden as secrete data) into its binary form and print it into a text file (i.e. “b.txt”) .
**STEP 2:** Segment the binary file into binary streams of n number of bits .

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 2, February 2016**

**STEP 3:** Print the size of file which is containing the secrete message .
**STEP 4:** Take a Bitmap Image file ( i.e. cover image) in which the secrete message will be hidden .
**STEP 5:** Read the image file (i.e. o_img). If the file is empty then print "the file can not open ". Otherwise go to STEP 6.
**STEP 6:** Now, we have to copy the whole image file(i.e. o_img) into another Bitmap image file(i.e.h_img) . So,

i) First, copy the whole 14 bytes bmp header file part of the file "o_img" into bmp image file "h_img".
   - This 14bytes includes 2bytes for ASCII characters of bmp,4bytes for size of bmp file ,4bytes for reserve bits(for future use) and 4bytes for Offbits(i.e specifies the number of file to the starting of pixel data type)
   - Convert the first 10 bytes of these 14bytes into binary streams and store it .

ii) The copy the 40 bytes of Bitmap header information of file "o_img" into file "h_img" .
iii) Copy the pixel data part of the image .

**STEP 7:** Now ,every row of pixel data part of image is in a form of single dimension array [i.e. rw_pxl[ ] ) . Calculate the padding portion each row pixel and store them into an integer variable(i.e. pd).
**STEP 8:** Calculate the new width of the image after padding of each pixel row . Store the new width into another integer variable .(i.e. new_w) .
**STEP 9:** Print the header , width ,height , padding and new width of image after padding .
**STEP 10:** Convert and segment the pixels of each row into 8bit binary bit streams . Each pixel is segmented into 3parts(i.e. for Red,Green and Blue color respectively(RGB color pattern) . Each part is consists of 8bits( 8bits for Red, 8bits for Green and 8bits for Blue color respectively) .
**STEP 11:** Now read the converted binary stream of secrete message from text file(i.e. b.txt) .
**STEP 12:** Replace the Least Significant Bits of each segment of each pixel in the row of pixel of the original image with binary message bits one by one b.txt .Store them in pixel data part of another bmp image file(i.e. "h_img") . The bits of binary message will be taken one by one from MSB of binary streams. Repeat STEP 12 untill all message bits are taken .
**STEP 13:** After hiding all message bits image file "h_img" is containing the image with hidden data.
**STEP 14:** Close both files (i.e "o_img" and "h_img") .
**STEP 15 :** End of the process of hiding the secrete data in image .
**STEP 16:** After hiding the secrete message into image file again convert the binary format of the bmp image file into an image file into an image format and store it into file .

### II.2. ALGORITHM FOR RETRIEVING THE HIDDEN MESSAGE FROM IMAGE

**STEP 1:** Take the image file in which hidden data bits are included .
**STEP 2:** Check whether the image file is empty or not . If empty , then print "file can not open" . Otherwise go to STEP 3 .
**STEP 3:** Read 14bytes bmp file header of the image file containing secrete message .
**STEP 4 :** Segment the file header into binary bit stream of 8bit and print them into a text file (i.e. "temp.txt").
**STEP 5:** Close the text file (temp.txt) .
**STEP 6 :** Again open the text file (i.e. temp.txt) in read mode.
**STEP 7:** Calculate the bmp image file size and print the file size
**STEP 8:** Read the 40 byte header information of the bmp image file ( in which file the data is hidden) .
**STEP 9:** Read the pixel part of the image file .
**STEP 10 :** Calculate the padding of each row of pixels after replacing the message bits .
**STEP 11:** Calculate the new width of the image after hiding data .
**STEP 12:** Print the new width of image after hiding data .
**STEP 13:** Create a text file (i.e. c.txt) .
**STEP 14 :** Compare the new image file size (i.e. the file included hidden data) with size of original image file . Chose an assignment variable (i.e. flag). When the file size will be same then assign flag= 1 . And break the process . Go to STEP 15.
**STEP 15:** Convert and segment each pixel of the image into binary stream of 8bits and store them into an single dimension array(i.e. re_pxl[ ] ) .

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 2, February 2016**

**STEP 16:** Each pixel in pixel row( rw_pxl [ ] ), is divided into 3 segments(for RGB color method). Pick LSB bit of each segment and store them into a text file(c.txt). Repeat this step till the difference of pixel data of the image(i.e. consisting hidden message) with the pixel data of original image , is caught .
**STEP 17:** After retrieving all hidden message bits from the image , now the text file(i.e. c.txt) is containing the secrete message in form of binary bit stream of 8bits .
**STEP 18:** Convert the binary stream of text file(c.txt) character stream and store them into another text file(txt_fl) . Now the new text file is containing the original secrete message . Message is extracted from image.

## III .EXAMPLE

We have elaborate our project work through an illustrative example .We hide the secrete message "ABCD" in a cover image name "Rose.bmp" using LSB method of Steganography

### III.1. HIDING SECRETE DATA IN COVER IMAGE USING LSB METHOD

**Step 1:** Fist each character of the text message( which will be hidden in cover image) is converted into its corresponding ASCII value.

**A→ 65** **B→ 66** **C→ 67** **D→ 68**

**Step 2:** Each ASCII value its converted into its binary form of 8 bit . And we get a binary stream for the text as below,

**65→ 01000001** **66→ 01000010** **67→ 01000011** **68→ 01000100**

Segment the binary bit streams **.**

**01000001 01000010 01000011 01000100**

**Step 3:** Cover image "Rose.bmp" is to be read

***Figure III.1(a)***
**Rose.bmp**

***Table III.1(a)***
***Equivalent ASCII of the Figure III.1(a)***

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 141 | 142 | 146 | 152 | 147 | 151 | 157 | 156 | … |
| 160 | 155 | 159 | 162 | 133 | 123 | 133 | 145 | … |
| 144 | 141 | 141 | 138 | 61 | 55 | 65 | 79 | … |
| 144 | 167 | 131 | 166 | 50 | 59 | 151 | 74 | … |
| 120 | 125 | 167 | 141 | 56 | 56 | 133 | 56 | … |
| 170 | 133 | 131 | 132 | 60 | 61 | 74 | 56 | … |
| 124 | 158 | 139 | 154 | 58 | 120 | 59 | 133 | … |
| 136 | 153 | 154 | 126 | 120 | 88 | … | … | … |

**Step 4 :** Convert the Cover Image from decimal to binary and getting result is being shown in table III.1(b)

***Table III.1(b)***
***Equivalent ASCII of the Figure III.1(a)***

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 10010000 | 10001110 | 10010010 | 10011000 | 10010011 | 10010111 | 10011101 | 10011100 | … |
| 10100000 | 10011011 | 10011111 | 10100010 | 10000101 | 01111011 | 10000101 | 10010001 | … |
| 10010000 | 10001101 | 10001101 | 10001010 | 00111101 | 00110111 | 01000001 | 01001111 | … |
| 10010000 | 10100111 | 10000011 | 10000010 | 00110010 | 00111011 | 10010111 | 01001010 | … |
| 01111000 | 01111101 | 10100111 | 10001101 | 00111000 | 00111000 | 10011010 | 00111000 | … |
| 10101010 | 10000101 | 10000011 | 10000100 | 00111100 | 00111101 | 01001010 | 00111000 | … |
| 01111100 | 10011110 | 10001011 | 10011010 | 00111010 | 01111101 | 00111001 | 10000101 | … |
| 10001000 | 10011001 | 10011010 | 01111110 | 01111000 | 01011000 | … | … | … |

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 2, February 2016**

**Step 5** : Break the secrete message bytes to be hidden into bits .

**01000001 01000010 01000011 01000100**

**0 1 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 0 1 0 0 0 1 0 0**

**Step 6:** Take first row of pixels from binary table form of cover image

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 10010000 | 10001110 | 10010010 | 10011000 | 10010011 | 10010111 | 10011101 | 10011100 | … |

Suppose we chose 5th byte from first row.

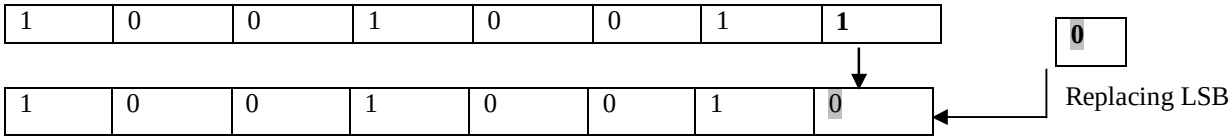
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |

**Step 7:** Suppose we chose 5th byte of first row for hiding MSB5 of the message bit
---- 5th byte of of original cover image is

----- 5th bit of message bits is 0

**0 1 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 0 1 0 0 0 1 0 0**

**Step 8:** Replace the Least Significant Bit(LSB) of pixel bit with the MSB of the secrete message .




------ Now repeat this step for all bytes of cover image until all message bits are hidden. After all replacement it is being show in Table III.1(c).

***Table III.1(c)***
***Pixel bits of cover image after hiding message***

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 10010000 | 10001111 | 10010010 | 10011000 | 10010010 | 10010110 | 10011100 | 10011101 | … |
| 10100000 | 10011011 | 10011110 | 10100010 | 10000100 | 01111010 | 10000101 | 10010000 | … |
| 10010000 | 10001101 | 10001100 | 10001010 | 00111100 | 00110110 | 01000001 | 01001111 | … |
| 10010000 | 10100111 | 10000010 | 10000010 | 00110010 | 00111011 | 10010110 | 01001010 | … |
| 01111000 | 01111101 | 10100111 | 10001101 | 00111000 | 00111000 | 10011010 | 00111000 | … |
| 10101010 | 10000101 | 10000011 | 10000100 | 00111100 | 00111101 | 01001010 | 00111000 | … |
| 01111100 | 10011110 | 10001011 | 10011010 | 00111010 | 01111101 | 00111001 | 10000101 | … |
| 10001000 | 10011001 | 10011010 | 01111110 | 01111000 | 01011000 | … | … | … |

Highlighted areas are indicating the bit replacement .
[N.B.the replaced bits are highlighted only for shake of explanation]

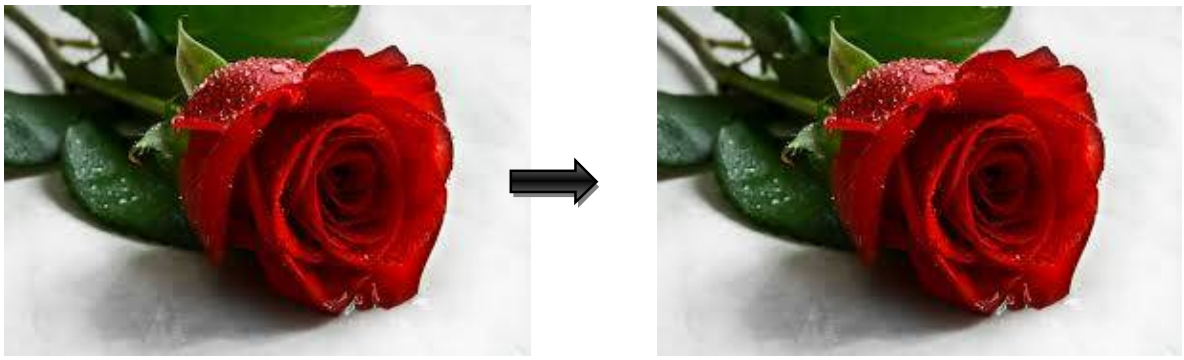
Finally the cover image before and after steganography is shown in figure 3.1(b)

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 2, February 2016



**cover image before steganography** **cover image after steganography**

***Figure III.1(b)***

***Cover image before and after steganography***

Step 9 : Number of message bit to be hidden in the reserved section(4bytes) of image file header starting from 6th byte to 9th byte .

### III.2. RETRIEVING THE HIDDEN MESSAGE FROM COVER IMAGE

**Step 1 :** Read the new cover image in which the message is hidden .

***Figure III.2(a)***

***Stego image of Rose.bmp***

After conversion of the figure III.2(a), Table III.2(a) is being gotten.

**Table III.2(a)**
**Pixel values after steganography**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 141 | 143 | 146 | 152 | 146 | 150 | 156 | 157 | …… |
| 160 | 155 | 158 | 162 | 132 | 122 | 133 | 144 | …… |
| 144 | 141 | 140 | 138 | 60 | 54 | 65 | 79 | ……. |
| 144 | 167 | 130 | 130 | 50 | 59 | 150 | 74 | ……. |
| 121 | 125 | 166 | 141 | 56 | 56 | 154 | 57 | …….. |
| 170 | 132 | 131 | 132 | 60 | 61 | 74 | 56 | …….. |
| 124 | 158 | 139 | 154 | 58 | 120 | 59 | 133 | …….. |
| 136 | 153 | 154 | 126 | 120 | 88 | …….. | …… | …… |

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

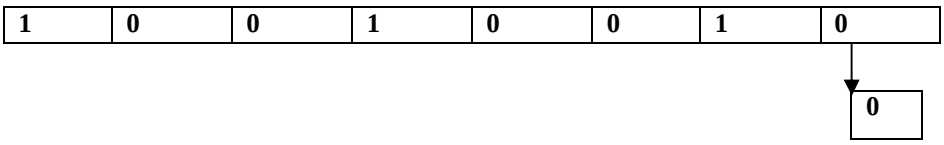
**Vol. 4, Issue 2, February 2016**

Step 2 : Convert the pixel values into 8 bits of binary stream .

***Table III.2(b)***
***Binary stream of Cover image after hiding the message***

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 10010000 | 10001111 | 10010010 | 10011000 | 10010010 | 10010110 | 10011100 | 10011101 | … |
| 10100000 | 10011011 | 10011110 | 10100010 | 10000100 | 01111010 | 10000101 | 10010000 | … |
| 10010000 | 10001101 | 10001100 | 10001010 | 00111100 | 00110110 | 01000001 | 01001111 | … |
| 10010000 | 10100111 | 10000010 | 10000010 | 00110010 | 00111011 | 10010110 | 01001010 | … |
| 01111000 | 01111101 | 10100111 | 10001101 | 00111000 | 00111000 | 10011010 | 00111000 | … |
| 10101010 | 10000101 | 10000011 | 10000100 | 00111100 | 00111101 | 01001010 | 00111000 | … |
| 01111100 | 10011110 | 10001011 | 10011010 | 00111010 | 01111101 | 00111001 | 10000101 | … |
| 10001000 | 10011001 | 10011010 | 01111110 | 01111000 | 01011000 | … | … | … |

Step 3 : Extract the LSB s of each byte . Start from first row of the table . Suppose we take the 5th byte from first row .




-------Repeat this step for each pixel until all message bits are retrieved. The 5th bit of the message is 0 which is correctly retrieved .

**Step 4:** After repeating Step 3 all the message bits are retrieved .

**0 1 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 0 1 0 0 0 1 0 0**

**Step 5 :** Segment the message bits into 8 bits of binary stream .

**01000001 01000010 01000011 01000100**

**Step 6 :** Convert each segment into their integer value respectively .

**01000001 → 65 01000010 → 66 01000011 → 67 01000100 → 68**

**Step 7 :** Convert the integers into their corresponding character value .

**65→ A 66→ B 67→ C 68→ D**

--------The original message **"ABCD"** is retrieved .

## IV. RESULT AND ANALYSIS

The pixels of the cover image must fulfill the minimum requirement for the process of data hiding. The minimum image pixel for width is at least 150 while the minimum image pixel for height is at least 112. [2] Both cover and stego images are alike with the images with the images those are shown in our previous example with near-zero distortion noticeable by naked eyes. Therefore, the LSB steganography algorithm is a strong yet robust algorithm to produce a stego image which will not be doubted by outsider that the image contains any secret message. If the file size exceed 1MB then the data hiding process fails .This failure occurs cause we are using an array in our coding portion which supports up to array size 32667bits. So,it can be failed in case of files whose size is more than 1MB and more than 1MB.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 2, February 2016**

**Table IV.1**
**Result of Experiment for Hiding the Data through this Algorithm**

The image file format used in proposed algorithm is focused on bitmap (BMP) format. The BMP file format handles graphics files within the Microsoft Windows OS. Typically, BMP files are uncompressed, hence they are large. The advantage of using BMP files is the simplicity and wide acceptance of BMP files in Windows programs. Thus, this type of image is chosen to be used in our proposed algorithm. Since BMP image has a relatively larger size, the pixels in image are relatively larger as well. Thus, it provides more space for binary codes to be encoded within it. To increase as much as characters that can be hidden, zip technique can be used to reduce to total size of file and to enhance the security of the file. Above Table shows the comparison of different sizes in BMP image by using the proposed steganography algorithm.

## V. CONCLUSION

Our conclusion towards this project work is that we have tested the implementation of our proposed algorithm and this algorithm worked correctly for the above set of image files. From this we can assume that algorithm can correctly be implemented for various type and size of file. It will be secured.

| Image File | Image File size (in KB) | Text file name | Text file size (in KB) | Stego image name | Stego image size | Hide message | Retrieve message |
|---|---|---|---|---|---|---|---|
| Bird.bmp | 438 | En.txt | 4.01 | Hid1.bmp | 438 KB | √ | √ |
| miley.bmp | 1 23 | X.txt | 4 | Hid2.bmp | 123KB | √ | √ |
| Dog.bmp | 1 | Message.txt | 12.1 | Hidsmile.bmp | FAILED | | |
| Rose.bmp | 3.14 | n.txt | 2.0 | Hrose.bmp | 3.14 KB | √ | √ |
| Build.bmp | 3.14 | File1.txt | 5.1 | Buildh.bmp | FAILED | | |
| mage.bmp | 6.74 | File2.txt | 4.1 | Imageh.bmp | 6.74 KB | √ | √ |
| Sky.bmp | 9.9 | M.txt | 3.34 | Usky.bmp | 9.9 KB | √ | √ |
| Forest.bmp | 3.15 | z.txt | 10.5 | For.bmp | FAILED | | |

We have fulfill our expectation only for BMP image file . And it is working correctly.

## REFERENCES

**[1]** Vijay kumar sharma ,Vishal shrivastava ; **"A Steganography Algorithm For Hiding Image In Image By Improved LSB Substitution By Minimize Detection** " , Journal of Theoretical and Applied Information Technology 15th February 2012. Vol. 36 No.1.. ISSN: **1992-8645 ;** E-ISSN: **1817-3195 ;**

**[2]** Rosziati Ibrahim and Teoh Suk Kuan ;, " **Steganography Algorithm To Hide Secret Message Inside An Image ",** Published: February 25, 2011 **;**

**[3]** Hussein Al-Bahadili ; **"A Secure Block Permutation Image Steganography Algorithm" ; International Journal on Cryptography and Information Security (IJCIS), Vol.3, No. 3,** September 2013 **;**

**[4]** A. E.Mustafa , A.M.F.ElGamal , M.E.ElAlmi , Ahmed.BD; "A Proposed Algorithm **For Steganography In Digital Image Based on Least Significant Bit**"; Research Journal Specific Education ; April. 2011 ;

**[5]** Ayan Banerjee, Prof. Dr. Pranam Paul ; **"Blockbased Data Encryptionand Decryption usingAlgorithmic Operations";** 2015;

**[6]** Pranam Paul, Saurabh Dutta, A K Bhattacharjee ; "**An Approach to ensure Security through Bit-level Encryption with Possible Lossless Compression",** International Journal of Computer Science and Network Security", Vol. 08, No. 2, pp. 291 – 299, 2008 ;

**[7]** Andreas Westfeld ; **Steganographic Algorithm High Capacity Despite Better Steganalysis;** I. S. Moskowitz (Ed.): IH 2001, LNCS 2137, pp. 289–302, 2001.

**[8]** Moinak chowdhury , Prof. Dr. Pranam Paul ; **Block Based Data Encryption and ecryption Using The Distence Between Prime Numbers" ;** 2015 ;

**[9]** Ramadhan Mstafa, Christian Bach ; **"Information Hiding in Images Using Steganography Techniques" ;** March 14-16, 2013 ;

**[10]** Paulina T. Nguyen; **"Bitmap File Format and Manipulation"**; April 14, 2005 .

**[11]** Zoran Duric, Michael Jacobs, and Sushil Jajodia. **"Information Hiding: Steganography and Steganalysis"** Review Article Handbook of Statistics, Vol. 24, pp. 171-187, 2005.

**[12]** Nagham Hamid, Abid Yahya, R. Badlishah Ahmad, Dheiaa Najim, Lubna Kanaan; "**Steganography in image files: A survey",** Australian Journal of Basic and Applied Sciences, 7(1): 35-55, 2013 ISSN 1991-8178 .

**[13]** Dwayne Phillips**;** first edition of "**Image Processing in C" (Copyright 1994, ISBN 0-13-104548-2) ;** was published by R & D Publications;

**[14]** Jasleen Kour , Deepankar Verma ; **" Steganography Techniques –A Review Paper" ;** May 2014 **;** ISSN: 2278-9359 (Volume-3, Issue-5) ;

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 2, February 2016**

**[15]** Yang, Chunfang., Liu, Fenlin., Luo, Xiangyang., and Zeng, Ying., ***"Pixel Group Trace Model-Based Quantitative Steganalysis for Multiple Least-Significant Bits Steganography"***, IEEE Transactions on Information Forensics and Security, Vol. 8, No. 1, January 2013.
**[16]** Ishwarjot Singh ,J.P Raina***," Advance Scheme for Secret Data Hiding System using Hop field & LSB"*** International Journal of Computer Trends and Technology (IJCTT) – volume 4 Issue 7–July 2013.

**BIOGRAPHY**

**Aritra Dutta**, pursuing MCA(masters of computer application) from Narula Institute of Technology (NIT). Before it she had done her graduation from Seth Anandram Jaipuria College, Kolkata in the year of 2013. She have passed higher secondary examination and madhyamik examination from salkia K.B.R Girl's High School the year of 2010 and 2008 respectively.

**Souvik Neogi**, pursuing MCA(masters of computer application) from Narula Institute of Technology (NIT). Before it he had done his graduation from Vidyasagar Evening College, Kolkata in the year og 2013. He have passed higher secondary examination and madhyamik examination from Belgachia Monohar Academy in the year of 2009 and Kumar Ashutosh Institution (MAIN) Boy's in the year of 2007 respectively.

**Dr Pranam Paul,** ***Assistant Professor and Departmental Head, CA Department, Narula Institute of Technology (NIT), Agarpara*** had completed MCA in 2005. Then his carrier had been started as an academician from MCKV Institute of Technology, Liluah. Parallel, At the same time, he continued his research work. At October, 2006, National Institute of Technology (NIT), Durgapur had agreed to enroll his name as a registered Ph.D. scholar. Then he had joined Bengal College of Engineering and Technology, Durgapur. After that Dr. B. C. Roy Engineering College hired him in the MCA department at 2007. At the age of 30, he had got Ph.D. from National Institute of Technology, Durgapur, and West Bengal. He had submitted his Ph.D. thesis only within 2 Years and 5 Months. After completing the Ph.D., he had joined Narula Institute of Technology in Computer Application Department. Parallel he continues his research work. For that, he has 39 International Journal Publications among 54 accepted papers in different areas. He also reviewer of International Journal of Network Security (IJNS), Taiwan and International Journal of Computer Science Issue (IJCSI); **Republic of Mauritius.**