Information Security through Data Encryption and Data Hiding

Samir Kumar Bandyopadhyay[1] and Somaditya Roy[2]
[1]Department of Computer Science & Engineering
[2]A.K.Choudhuri School of Information Technology
University of Calcutta, India

ABSTRACT
This paper introduces a model for information security and message hiding based on image steganography as well as it describes about various steps of that proposed model in brief.

Keywords-Image steganography, Data Hiding

I. INTRODUCTION

Information Technology is the most essential aspect in today's world. Based on this fact computer application is still developing to handle securely the financial as well as the personal data more effectively. These data are extremely important from every aspect and we need to secure this from unauthorized access. Security is the process of preventing and detecting unauthorized use of data or computer or network. Prevention measures help us to stop unauthorized users from accessing any part of computer system. Detection helps to determine whether or not someone attempted to break into the system, if they were successful, and what they may have done. To achieve that security we may use various cryptography techniques. However, today data encryption is not everything or we cannot achieve strong security through this, we also need to secure the presence of date. Here comes the necessity of steganography. But we need to remember that neither cryptography nor steganography gives the total security to a data or information uniquely, thus we need to apply both techniques to achieve essential security. There are number of ways this can be done, but here we will concern with methods of altering the information in such a way that the recipient can undo the alteration and discover the original text.

In this paper, we concentrate on developing an innovative image steganographic model for information security through data encryption as well as data hiding.

2. MODEL

This is an image steganographic technique for message hiding as well as message encryption and decryption. In our model we use finger impression or finger image for stego

image, but we can also implement it by iris impression or face impression or same.

The basic idea of our model is a sender encrypts a message by his/her fingerprint to create a stegoimage and send it to the intended receiver. Receiver uses sender's public key to decrypt it to get the original message. Here, the finger impressions of users are act like private keys. So, in this model we need to pair of keys or two pair of images as public and private key. Initially we take 8X8 pixel JEPG (gray scale) finger impressions for private key. Now, we need to calculate the ASCII values of each pixel of finger impressions and put it in an 8X8 matrix.

Then, we consider the prime numbers between 0-255 which are 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233, 239, 241 and 251. After that we calculate the difference between the ASCII value and the nearest prime number of it's and form another 8X8 matrix and put difference values into it in binary form. I.e. if we consider the ASCII 100 than its nearest prime number which is 101 then the difference will be +1, again if we consider the ASCII 243 than its nearest prime number which is 241 then the difference will be -2 and so on. There may the case where the number will lie exactly in the middle of two prime numbers. Let us consider the ASCII 60. Now, in this case the nearest prime numbers are 59 as well as 61 both. In such case we take it as +1. After constructing the matrix we convert it to its corresponding 8X8 JEPG gray scale image and it will be our public key or share key.

Now, for encrypt a message we can follow either of two ways. Either we can encrypt the original message by using our private key and send it to the intended receiver or we can encrypt the message by using receiver public key and send it to him/her. Let us take the second possibility. In this case we need to insert the message in LSB of each pixels of our public image. After inserting the whole message we have to look farther to determine the changing pixels positions. For data hiding and data sending we need a cover image. We have no restriction of choosing cover image; we just need to reflect that changing information into it. But to insert that particular information we have to consider the following cases:

Case 1: (Number of change in pixel >

Number of unchanged pixel)

Then,

Change the Cover Image in the row wise as the public image.

If (Number of change in pixel > Number of unchanged pixel)

Change the LSB of unchanged position

If (Number of change in pixel <=Number of unchanged pixel)

Change the 2nd last LSB of changed position

Case 2: (Number of change in pixel <=

Number of unchanged pixel)

Then,

Change Cover Image in Column wise as the public image.

If (Number of change in pixel > Number of unchanged pixel)

Change the LSB of unchanged position

If (Number of change in pixel <= Number of unchanged pixel)

Change the 2nd last LSB of changed position

As the whole information is been reflected in the cover image we are ready to send it to the intended receiver.

In the receiver end we need the private key or the private image of the corresponding sender to get the actual message. One can only get the information of the change or the unchanged positions in pixels through steganalysis on cover image but not ever able to decrypt it into the actual message without having the private image or private key. After steganalysis we get the information of the position of changed or unchanged pixels. In our case, we encrypt the message using public image so we need to decrypt it using private image. Now, the interesting facts of our particular model is that we insert the message in LSB of our private image and then we look for the change and unchanged positions of pixels, if we perform the some in public image we can observe that it is perfectly opposite of private image. I.e. all the change positions in private image are totally unchanged in public image and all the unchanged positions of private image are totally changed in public image. So, at the receiver end after getting the changed and unchanged position's information we needed to update LSB of the public image in opposite fashion. The changed position remains unchanged and the unchanged position will need to change. After this updating is done we simply consider the LSB in serial manner to get the original message. Now, public image can be same for different users as we take only the ASCII and prime difference but none can decrypt it properly without having the actual private image. But Private images cannot be same in any case as we take the finger impression.

3. ANALYSIS

Let our Original Message: Good Boy
ASCII Value : 71, 111, 111, 100, 32, 66, 111, 121
Binary Representation: 01000111, 01101111, 01101111, 01100100, 00100000, 01000010, 01101111, 01111001.

We take public image for message encryption.

Public Image + Original Message (Update LSB of public image either by 1 or by 0 depending on message)
= Initial Stego Image

Private Image

8X8 JEPG grey scale Finger impression or private image.

A = Private key or private image

255	255	244	168	173	245	255	254
255	251	145	101	84	142	255	255
254	216	93	110	95	98	240	255
255	197	93	97	95	84	227	254
255	187	86	81	75	76	225	254
254	201	61	68	63	86	240	255
255	245	100	60	64	131	255	253
255	255	232	170	178	243	255	255

Figure 1. **ASCII value of each pixels of private image.**

B = Public key or public image

-4	-4	-3	-1	0	-4	-4	-3
-4	0	+4	0	-1	-3	-4	-4
-3	-5	+4	-1	+2	-1	+1	-4
-4	0	+4	0	+2	-1	0	-3
-4	+4	+3	+2	-2	+3	2	-3
-3	-2	0	-1	-2	+3	1	-4
-4	-4	+1	+1	+3	0	-4	-2
-4	-4	+1	+3	+1	-2	-4	-4

Figure 2. Values of difference between ASCII to nearest prime.

5	5	12	34	74	96	105	120
27	38	64	96	140	183	202	192
53	28	16	7	42	32	56	127
60	56	33	0	55	41	98	80
120	124	70	35	25	55	191	183
150	159	128	102	106	152	177	161
158	145	144	153	162	152	155	185
178	162	163	165	178	193	213	221

1100	1101	1100	1110	0000	1100	1101	1101
1100	0001	0101	0000	1111	1101	1101	1101
1100	1011	0101	1110	0011	1111	0001	1101
1100	0001	0101	0000	0010	1111	0000	1100
1100	0100	0011	0010	1110	0010	0010	1100
1100	1111	0000	1110	1110	0010	0001	1100
1100	1101	0001	0000	0011	0001	1101	1111
1100	1101	0001	0011	0001	1110	1100	1101

Figure 3. Binary representation of above.

In our case, we take public image for message encryption. Public image + Original Message (Update LSB of public image either by 1 or by 0 depending on message) = Initial Stego Image

Figure 4. After inserting the whole message to the above table.

Determine the positions of changed pixel in public image:

[0, 1] [0, 2] [0, 3] [0, 5] [0, 6] = change in 5 pixels
[1, 1] [1, 2] [1, 6] [1, 7] = change in 4 pixels
[2, 0] [2, 2] [2, 3] [2, 4] [2, 7] = change in 5 pixels
[3, 1] [3, 2] [3, 7] = change in 3 pixels
[4, 5] [4, 7] = change in 2 pixels
[5, 0] [5, 1] [5, 3] [5, 5] = change in 4 pixels
[6, 1] [6, 3] [6, 5] [6, 6] [6, 7] = change in 5 pixels
[7, 1] [7, 7] = change in 2 pixels

Total Change = 30 pixels

Here, number of changed in pixel positions = 30
Therefore, number of unchanged positions = (64-30) = 34
That means, Number of change in pixel positions> Number of unchanged pixel positions So, we change the cover image in column wise.

Cover Image

1100	1100	1101	1111	0000	1100	1100	1101
1100	0000	0100	0000	1111	1101	1100	1100
1101	1011	0100	1111	0010	1111	0001	1100
1100	0000	0100	0000	0010	1111	0000	1101
1100	0100	0011	0010	1110	0011	0010	1101
1101	1110	0000	1111	1110	0011	0001	1100
1100	1100	0001	0001	0011	0000	1100	1110
1100	1100	0001	0011	0001	1110	1100	1100

8X8 JEPG grey scale Cover Image.

C=

Figure 5. *ASCII value of each pixels of cover image*.

0000 0101	0000 0101	0000 1100	0010 0010	0100 1010	0110 0000	0110 1001	0111 1000
0001 1011	0010 0110	0100 0000	0110 0000	1000 1100	1011 0111	1100 1010	1100 0000
0011 0101	0001 1100	0001 0000	0000 0111	0010 1010	0010 0000	0011 1000	0111 1111
0011 1100	0011 1000	0010 0001	0000 0000	0011 0111	0010 1001	0110 0010	0101 0000
0111 1000	0111 1100	0100 0110	0010 0011	0001 1001	0011 0111	1011 1111	1011 0111
1001 0110	1001 1111	1000 0000	0110 0110	0110 1010	1001 1000	1011 0001	1010 0001
1001 1110	1001 0001	1001 0000	1001 1001	1010 0010	1001 1000	1001 1011	1011 1001
1011 0010	1010 0010	1010 0011	1010 0101	1011 0010	1100 0001	1101 0101	1101 1101

Figure 6. Binary representation of the above.

Change row wise:

0000 0100	0000 0101	0000 1100	0010 0010	0100 1011	0110 0000	0110 1001	0111 1001
0001 1011	0010 0100	0100 0010	0110 0000	1000 1100	1011 0111	1100 1000	1100 0010
0011 0101	0001 1101	0001 0000	0000 0111	0010 1010	0010 0001	0011 1001	0111 1111
0011 1100	0011 1010	0010 0011	0000 0000	0011 0111	0010 1001	0110 0010	0101 0010
0111 1000	0111 1100	0100 0110	0010 0011	0001 1001	0011 0101	1011 1111	1011 0101
1001 0100	1001 1101	1000 0000	0110 0100	0110 1010	1001 1010	1011 0001	1010 0001
1001 1111	1001 0001	1001 0001	1001 1001	1010 0000	1001 1000	1001 1011	1011 1001
1011 0010	1010 0000	1010 0011	1010 0101	1011 0010	1100 0001	1101 0101	1101 1111

Figure 7. Binary representation of the above after inserting the information of change in pixels positions in rowwise.

Change column wise:

0000 0101	0000 0101	0000 1110	0010 0000	0100 1010	0110 0010	0110 1011	0111 1001
0001 1011	0010 0110	0100 0010	0110 0000	1000 1100	1011 0111	1100 1000	1100 0000
0011 0111	0001 1101	0001 0010	0000 0101	0010 1000	0010 0000	0011 1000	0111 1111
0011 1100	0011 1000	0010 0011	0000 0000	0011 0111	0010 1001	0110 0010	0101 0000
0111 1000	0111 1101	0100 0110	0010 0011	0001 1001	0011 0101	1011 1111	1011 0111
1001 0100	1001 1111	1000 0000	0110 0100	0110 1010	1001 1010	1011 0001	1010 0000
1001 1110	1001 0001	1001 0000	1001 1011	1010 0010	1001 1010	1001 1001	1011 1001
1011 0010	1010 0010	1010 0011	1010 0101	1011 0010	1100 0001	1101 0101	1101 1101

Figure 8. Binary representation of the above after inserting the information of change in pixels positions in columnwise.

Unchanged pixel position: Green
Changed pixel positions: Blue
Here, we change pixels of the cover image in column wise.

C1=

5	5	14	36	74	98	107	121
27	38	66	96	140	183	204	192
55	29	18	9	44	32	56	127
60	56	35	0	55	41	98	80
120	125	70	35	25	57	191	183
148	159	128	104	106	154	177	162
158	145	144	155	162	154	157	185
178	162	163	165	178	193	213	221

Figure 9. After inserting the information of change in pixels positions in columnwise.

C1 is the expected final stego image which is ready to send to the intended receiver.

Now, if we insert the message in LSB of private image we will get the following:

1111 **1110**	1111 1111	1111 0100	1010 1000	**1010** **1100**	1111 0101	1111 1111	**1111** **1111**
1111 **1110**	1111 1011	1001 0001	**0110** **0100**	**0101** **0101**	**1000** **1111**	1111 1111	1111 1111
1111 1110	**1101** **1001**	0101 1101	0110 1110	0101 1111	**0110** **0011**	**1111** **0001**	1111 1111
1111 **1110**	1100 0101	0101 1101	**0110** **0000**	**0101** **1110**	**0101** **0101**	**1110** **0010**	1111 1110
1111 **1110**	**1011** **1010**	**0101** **0111**	**0101** **0000**	**0100** **1010**	0100 1100	**1110** **0000**	1111 1110
1111 1110	1100 1001	**0011** **1100**	0100 0100	**0011** **1110**	0101 0110	**1111** **0001**	**1111** **1110**
1111 **1110**	1111 0101	**0110** **0101**	0011 1100	**0100** **0001**	1000 0011	1111 1111	1111 1101
1111 **1110**	1111 1111	**1110** **1001**	**1010** **1011**	**1011** **0011**	**1111** **0010**	**1111** **1110**	1111 1111

Figure 10. After inserting the message in LSB of the private image.

Determine the positions of changed pixel in private image:

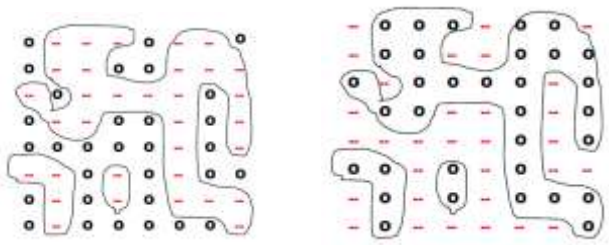
[0, 0] [0, 4] [0, 7] = change in 3 pixels
[1, 0] [1, 3] [1, 4] [1, 5] = change in 4 pixels
[2, 1] [2, 5] [2, 6] = change in 3 pixels
[3, 0] [3, 3] [3, 4] [3, 5] [3, 6] = change in 5 pixels
[4, 0] [4, 1] [4, 2] [4, 3] [4, 4] [4, 6] = change in 6 pixels
[5, 2] [5, 4] [5, 6] [5, 7] = change in 4 pixels
[6, 0] [6, 2] [6, 4] = change in 3 pixels
[7, 0] [7, 2] [7, 3] [7, 4] [7, 5] [7, 6] = change in 6 pixels

Total Change = 34 pixels

We can observe that the positions of changed pixels in public image are perfectly the opposite in case of private image. So, in the receiver end after we get the stego image first we need to perform steganalysis on it to get the information of positions of changed and unchanged pixels. As soon we get that we just required inserting that in LSB of private image but in opposite manner. After inserting the whole information if we consider the LSBs together of the private image we will get the original message.

4. CONCLUSIION

In our model we sincerely consider all the basic principles of information security and try to keep it. As we use two different images one for private and another for public key it gives us to preserve the authentication and helps to overcome non-repudiation issue. In the actual carrier which is cover image in our case, we only send the minimal information of the changes in pixel positions of the key instead of the actual message and none can decrypt or retrieve the original message from it without knowing any knowledge of key. It gives us the privileged to sincerely keep the confidentiality of data or message. This model is very simple to implement yet very complicated to break. Initially we construct this model on image steganography with simple gray scale impression but there is a larger scope to implement it in other steganographic techniques also in future.



(A) (B)

Figure 11. (A) Private Image, (B) Public Image

Unchanged pixel position: **o**
Changed pixel positions: **--**

ACKNOWLEDGMENT *(HEADING 5)*

The preferred spelling of the word "acknowledgment" in America is without an "e" after the "g". Avoid the stilted expression, "One of us (R. B. G.) thanks . . ." Instead, try "R. B. G. thanks". Put sponsor acknowledgments in the unnumbered footnote on the first page.

REFERENCES

[1] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*

[2] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[3] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[4] K. Elissa, "Title of paper if known," unpublished.

[5] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

AUTHORS PROFILE

Authors Profile …

Authors:

Prof. Samir Kumar Bandyopadhyay did his B.E., M.Tech., Ph. D (ComputerScience & Engineering), His research interests include Bio-medical Engg, Mobile Computing, Pattern Recognition, Graph Theory, Software Engg.,etc. He is Senior Member of IEEE.

Mr. Somaditya Roy is presently working as Research Scholar in the Department of Computer Science and Engineering, University of Calcutta, India.