IJCSMS (International Journal of Computer Science & Management Studies) Vol. 22, Issue 01
Publishing Month: January 2016
An Indexed and Referred Journal with ISSN (Online): 2231–5268
www.ijcsms.com

Efficient and Advanced Encryption using Hill Cipher and RGB Image Steganography

Savita[1] and Sunita Rani[2]

[1]M. Tech. Network Security, CSE and IT Department
Bhagat Phool Singh Women University, Khanpur Kalan, Sonepat, Haryana, India
savitagrewal12892@gmail.com

[2]Assistant Professor, Department of CSE & IT
Bhagat Phool Singh Women University, Khanpur Kalan, Sonepat, Haryana, India
sunita.bpsmv@gmail.com

Publishing Date: January 17, 2016

Abstract

This work enhances the message capacity and increases the security for the communication of message between two ends. In the existing work the data hided in to the image is in the raw form, if any unauthorized entity gets to know about the text hiding then text can be extracted. This problem is removed by encrypted the text before hiding inside the image, so that if user extract the data even though he is not able to understand it. The proposed method uses the hill cipher technique to secure i.e. encrypt the text. While in the decryption the text is decrypted after the extraction from the image. Moreover, the capacity of the image to hide the data effectively is limited. This problem is solved by compressing the text so that the same amount of text takes less number of bits to be stored. Overall the firstly the secret text is compressed by using the LZW compression algorithm. Then this compressed text is encrypted using hill cipher then this compressed and encrypted text is hided inside the image using the existing technique.

Keywords: *Hill cipher, Cryptography, MSE, PSNR, Secure, compression.*

I. Introduction

Cryptography enables the data communication between two persons or between groups to provide secure communication over the network. This kind of communication not only prevents the unauthorized access over the network but also maintains the data integrity. The authentication is provided by digital signature or digital certificates. At the basic level cryptography approaches are divided in two main categorized based on numbers of keys involved in communication. These approaches are called private key cryptography and public key cryptography [1][2]. Private key cryptography approach is also called private key cryptography. As the name suggested in this cryptography approach only single key is involved to enable the encoding and decoding process. Public-Key Cryptography approach is also called asymmetric cryptography approach. According to his approach, encryption and decryption is performed by two different keys. As the encryption process begins, instead of generating single key, two keys are generated called Public key and Private Key. The generator A keeps the private key with him and distributes the public key to all users that can send information to it. Now, as some user B want to send information to the user A. In such case, user B will use the public key of User A to perform the encoding process. Here the cryptography will be performed using public key of receiver. Now after the encoded process, the cipher information is transferred to the receiver A. As receiver receives this information, the decoding process is performed using private key of User A. This decoding process is able to get the information back in its original form [9][10][11].

II. LZW Compression

LZW compression is named after its developers, *A. Lempel and J. Ziv*, with later modifications by *Terry A. Welch*. It is the foremost technique for general purpose data compression due to its simplicity and versatility. Typically, it can be expected to compress text, executable code, and similar data files to about one-half their original size. LZW also performs well when presented with extremely redundant data files, such as tabulated numbers, computer source code, and acquired signals. Compression ratios of 5:1 are common for these cases. LZW is the basis of several personal computer utilities that claim to “double the capacity of our hard drive.”

LZW compression is always used in GIF image files, and offered as an option in TIFF and PostScript. A common choice is to provide 4096 entries in the table. In this case, the LZW encoded data consists entirely of 12 bit codes, each referring to one of the entries in the code

table. Decompression is achieved by taking each code from the compressed file, and translating it through the code table to find what character or characters it represents. Codes 0-255 in the code table are always assigned to represent single bytes from the input file. For example, if only these first 256 codes were used, each byte in the original file would be converted into 12 bits in the LZW encoded file, resulting in a 50% larger file size. During decompression, each 12 bit code would be translated via the code table back into the single bytes.
The LZW method achieves compression by using codes 256 through 4095 to represent sequences of bytes. For example, code 523 may represent the sequence of three bytes: 231 124 234. Each time the compression algorithm encounters this sequence in the input file, code 523 is placed in the encoded file. During decompression, code 523 is translated via the code table to recreate the true 3 byte sequence. The longer the sequence assigned to a single code, and the more often the sequence is repeated, the higher the compression achieved.

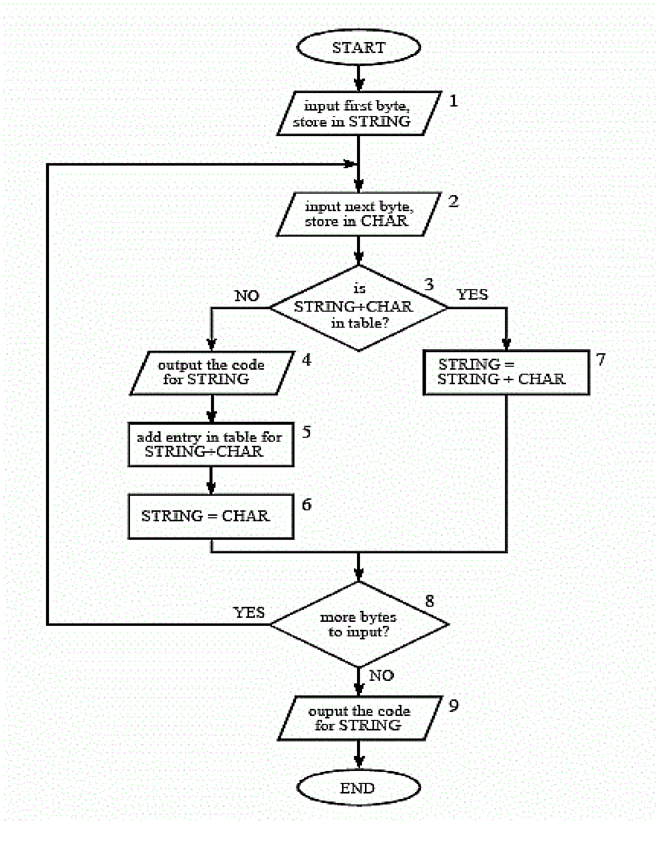

START
input first byte, store in STRING
1
input next byte, store in CHAR
2
is STRING+CHAR in table?
3
NO
YES
output the code for STRING
4
add entry in table for STRING+CHAR
5
STRING = CHAR
6
STRING = STRING + CHAR
7
more bytes to input?
8
YES
NO
ouput the code for STRING
9
END

Figure 1: Flow chart for LZW Compression algorithm

III. Hill Cipher

The core of Hill-cipher is matrix manipulations. It is a multi-letter cipher, developed by the mathematician Lester Hill. For encryption, algorithm takes m successive plaintext letters and instead of that substitutes m cipher letters. In Hill cipher each character is assigned a numerical value like: a=0, b=1… ….. z=25. The substitution of cipher text letters in place of plaintext leads to m linear equations. For m=3, the system can be described as follows:
C1=(K11P1+K12P2+K13P3)MOD26---------(1)
C1=(K21P1+K22P2+K23P3)MOD26---------(2)
C1=(K31P1+K32P2+K33P3)MOD26---------(3)
This can be expressed in terms of column vectors and matrices: C=KP Where C and P are column vectors of length 3, representing the plaintext and the cipher text and K is a 3*3 matrix, which is the encryption key. All operations are performed mod 26 here. Decryption requires the inverse of matrix K. The inverse K-1 of a matrix K is defined by the equation. K K-1= I where I is the Identity matrix. The inverse of a matrix doesn't always exist, but when it does it satisfies the proceeding equation. K-1 is applied to the cipher text, and then the plain text is recovered.
In general terms we can write as follows:
For encryption: C=Ek (P) =KP
For decryption: P=Dk(C) =K-1 C= K-1Kp=P

IV. Existing Work

In this algorithm, sender need to input secret message, two conversion table and a cover image. The sender needs to convert the message into corresponding bit form. After this step each four bits are converted to an alphabet and hill cipher is applied on it. The encrypted message is then gets embedded to the red, green and blue images of the cover image depending on the length of the secret message then these images are combined to form the stego image. For extraction of the secret message, user needs to input the stego image, gets its red green and blue images collects the bits according to the length of the message and combines the bit to get the secret message. Then bits are converted into alphabets and hill cipher is applied to get the message and finally converts the alphabets to 4 bit format from where 8 bit data are formed. Then if the message was in the form of image they are converted into pixel arrays and image forms else each 8 bit are converted to corresponding Unicode characters.

Embedding Algorithm:-

I. Select a cover image and take its RED GREEN and BLUE images.

II. Convert each three image into its 8bit representation and store them into three arrays (for eg. Red [], green [], blue []).

III. Calculate the length of image.

IV. Take a secret message which may be an image or containing any alphabet or number or any Unicode characters.

V. Take 8 bit representation of the secret message. Divide each 8bit into two 4bits and store them into an array, message []

VI. Now make a table which maps each 16 numbers from 0000 to 1111 to a random 16 alphabets

VII. Map each element of message [] i.e. four bit of message with the corresponding alphabet of the table and we will get a new secret message of alphabets only.

IX. Convert the cipher text to 8 bit representation and find the number of bits.

X. If the number of bits is less or equal to the image length then hide the message in any one of the Red[],green[],blue[]) LSBs.

XI. Else if number of bits is less or equal to 2 *image length then make two array of the secret message taking alternate elements hide the message in the LSBs of Red[],green[] or green[],blue[] or blue[], Red[].

XII. Else make three arrays of the message and hide them in the Red [], green [], blue [] respectively.

XIII. Combine the Red [], green [], blue [] images and get a stego image send it to the receiver,

XIV. Send the message length and mapping table through a secure channel or encrypt them by a technique known to both and send to the receiver.

Extraction Algorithm:

I. Receiver the secret message containing message length and mapping table.

II. Find the message length and length of the covered image.

III. Check weather message is equal to or less than image length or less than 2*image length or greater than 2* image length.

IV. Get the stegano image and find its red [], blue [] and green [] images and finally array of its individual LSBs.

V. If the length of the message is less the image length then take all LSBs of red [], green [] and blue [].

VI. Now if length of message =2*image length. Take LSBs of Red [], green [] or green [], blue [] or blue [], Red [].

VII. Else if the length of the message is less than the image length than it takes any one LSB array of Red [], green [] or blue [] depending of the sender.

VIII. Then Convert each 4 bit to an alphabet according to the second mapping table and apply hill cipher and after getting the corresponding input alphabet apply the first table to get the 4 bit representation.

IX. Combining all bits of array get each 8bit representation. And convert them into image or Unicode character using first mapping table.

X. This is the original message.

V. Proposed Algorithm

In the existing work the data hided in to the image is in the raw form, if any unauthorized entity gets to know about the text hiding then text can be extracted. This problem is removed by encrypted the text before hiding inside the image, so that if user extract the data even though he is not able to understand it. The proposed method uses the hill cipher technique to secure i.e. encrypt the text. While in the decryption the text is decrypted after the extraction from the image. Moreover, the capacity of the image to hide the data effectively is limited. This problem is solved by compressing the text so that the same amount of text takes less number of bits to be stored. Overall the firstly the secret text is compressed by using the LZW compression algorithm. Then this compressed text is encrypted using hill cipher then this compressed and encrypted text is hided inside the image using the existing technique. The process of decryption follows the reverse process. The process can also be easily understood by following algorithm:

Hiding Algorithm

I. Select a cover image and take its RED GREEN and BLUE images.

II. Convert each three image into its 8bit representation and store them into three arrays (for e.g. Red [], green [], blue []).

III. Calculate the length of image.

IV. Take a secret message which may be an image or containing any alphabet or number or any Unicode characters.

V. Compress the secret message by using LZW compression.

VI. Take 8 bit representation of the compressed secret message. Divide each 8bit into two 4bits and store them into an array, message []

VII. Now make a table which maps each 16 numbers from 0000 to 1111 to a random 16 alphabets

VIII. Map each element of message [] i.e. four bit of message with the corresponding alphabet of the table and we will get a new secret message of alphabets only.

IX. Apply HILL cipher to the message.

X. Convert the cipher text to 8 bit representation and find the number of bits.

XI. If the number of bits is less or equal to the image length then hide the message in any one of the Red[],green[],blue[]) LSBs.

XII. Else if number of bits is less or equal to 2 *image length then make two array of the secret message taking alternate elements hide the message in the LSBs of Red[],green[] or green[],blue[] or blue[], Red[].

XIII. Else make three arrays of the message and hide them in the Red [], green [], blue [] respectively.

XIV. Combine the Red [], green [], blue [] images and get a stego image send it to the receiver,

XV. Send the message length and mapping table through a secure channel or encrypt them by a technique known to both and send to the receiver.

Extraction Algorithm:

I. Decrypt the secret message containing message length and mapping table.

II. Find the message length and length of the covered image.

III. Check weather message is equal to or less than image length or less than 2*image length or greater than 2* image length.

IV. Get the stegano image and find its red [], blue [] and green [] images and finally array of its individual LSBs.

V. If the length of the message is less the image length then take all LSBs of red[], green[] and blue[] .

VI. Now if length of message =2*image length. Take LSBs of Red [], green [] or green [], blue [] or blue [], Red [].

VII. Else if the length of the message is less than the image length than it takes any one LSB array of Red [], green [] or blue [] depending of the sender.

VIII. Then Convert each 4 bit to an alphabet according to the second mapping table and apply hill cipher and after getting the corresponding input alphabet apply the first table to get the 4 bit representation.

IX. Combining all bits of array get each 8bit representation. And convert them into image or Unicode character using first mapping table.

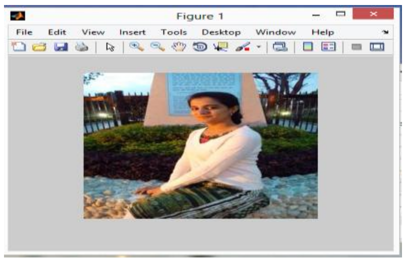
X. This is the original compressed message.

XI. Decompress the text by using the LZW decompression algorithm.

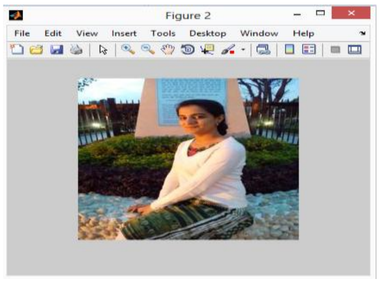
The implementation of the above algorithm using the MATLAB is discussed in the next chapter.

VI. Results

The presented work is implemented in mat lab environment. The work is applied on real time images. The analysis of work is defined under different parameters. The parameters included in this work are MSE, PSNR. The input image on which the cryptography is applied is shown



2 (a): Input Image



2(b): Result Image

Figure 2: Input and Result Image

The work is tested on multiple input images. The results obtained from the work under different parameters is shown in table 1.

Table 1: Analysis Parameters

Image	PSNR(in dB)		MSE	
	Existing	Proposed	Existing	Proposed
Image1	+56.03	+59.36	+0.16	+0.08
Image2	+55.16	+57.66	+0.20	+0.11
Image3	+59.02	+96.75	+0.08	+0.01
Image4	+64.55	+65.47	+0.02	+0.02
Image5	+55.34	+57.78	+0.19	+0.11

The graphical represents of results is shown in figure 3&4.

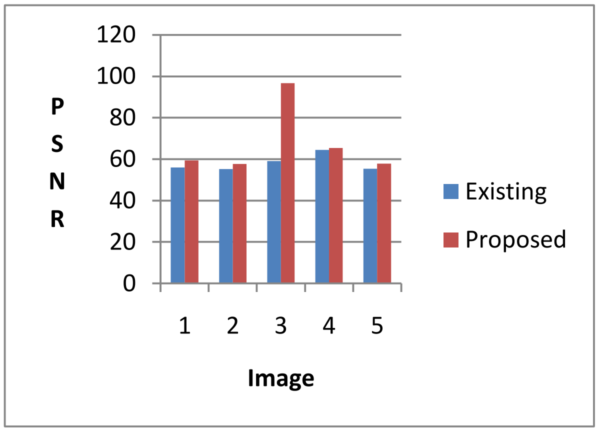


Figure 3: PSNR Comparison

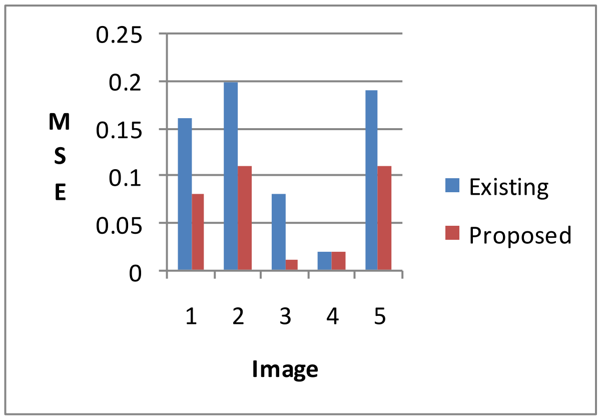


Figure 4 : MSE Analysis Results

The obtained results shows, the presented work is effective in terms of obtained PSNR and MSE values.

VII. Conclusion

The work is implemented using the MATLAB. It is analyzed that the increase in text size leads to decrease in the PSNR and correspondingly increase in the MSE. The PSNR of the proposed technique is around 59 db and MSE is around 0.15. The increase in the PSNR value and decrease in the MSE value shows the effectiveness of the technique. In future following work can be done: The data can be hided by using other available medium like video, DNA etc. The process can be applied for the medical images as well as on satellite images..

References

[1] Ohood S. Althobaiti, an Enhanced Elliptic Curve Cryptography for Biometric, 7th International Conference on Computing and Convergence Technology (ICCCT), pp 1048–1055, 2012

[2] Seny Kamara, Dynamic Searchable Symmetric Encryption, CCS'12, October 16–18, 2012, Raleigh, North Carolina, USA. ACM 978-1-4503-1651-4/12/10 (pp 965-976)

[3] Mamta Juneja and Parvinder Singh Sandhu, (2013) A New Approach for Information security using an Improved Steganography Technique, Journal of Info.Pro.Systems, Vol 9, No: 3, pp.405-424.

[4] Ibrahim, R., & Kuan, T. S. (2011). Steganography algorithm to hide secret message inside an image. ArXiv preprint arXiv: 1112.2809.

[5] Ahmed, H. E. D. H., Kalash, H. M., & Allah, O. S. F. (2006). Encryption quality analysis of the RC5 block cipher algorithm for digital images. Optical Engineering, 45(10), 107003-107003.

[6] Stevens, M., Sotirov, A., Appelbaum, J., Lenstra, A., Molnar, D., Osvik, D. A., & De Weger, B. (2009). Short chosen-prefix collisions for MD5 and the creation of a rogue CA certificate. In Advances in Cryptology-CRYPTO 2009 (pp. 55-69). Springer Berlin Heidelberg.

[7] D.R. Stinson, (2002) "Cryptography Theory and Practice," CRC Press, Inc.

[8] Al-Husainy, M. A. F. (2012). A Novel Encryption Method for Image Security.International Journal of Security and Its Applications, 6(1).

[9] Balaji, N., & Nagaraj, N. (2008). Cryptanalysis of a Chaotic Image Encryption Algorithm. ArXiv preprint arXiv: 0801.0276.

[10] Fridrich, J., Goljan, M., & Du, R. (2001, October). Reliable detection of LSB steganography in color and grayscale images. In Proceedings of the 2001 workshop on Multimedia and security: new challenges (pp. 27-30). ACM.

[11] H. B. Kekre, Archana Athawale, and Pallavi N. Halarnkar, (2008) Increased Capacity of Information Hiding in LSB's Method for Text and Image, World Academy of Science, Engineering and Technology 17.