Enhanced Network Security Using Text to Image Encoding

Rahulkrishna P K [1], Eshwari R [2], Shree Harsha N J [3], Divyashree E [4], C Gururaj [5]
[1, 2, 3, 4] Students, [5] Assistant Professor, Department of Telecommunication Engineering.
BMS College of Engineering, Bengaluru,
Karnataka, India.
[1] madhurrkpk162@gmail.com, [2] eshwari471994@gmail.com, [3] shreeharsha009@gmail.com, [4]divya.e.yadav@gmail.com,
[5]gururaj.tce@bmsce.ac.in

Abstract*— In this paper, text to image encoding as a means to enhance a network's security using MATLAB tool is discussed. As the network grows bigger and bigger security is the key issue coming in to the picture. Current encoding techniques require lots of complicated algorithms a long with dedicated hardware as well as software support. With the help of this approach a ciphered image is created with comparatively less hardware and software support. Images are generally supported as well as compatible to any network. With help of image creating tool a text is converted its number equivalent value then that number represents the greyscale value of one pixel in the image. The overall paper emphasizes on a compressed way of data communication without compromising on security of the information in the network.

Keywords*— Image Encoding, ASCII, Merging, Compression, Histogram, Bit Depth

I. INTRODUCTION

There is a huge demand for security of a piece of information, in present day situation, across all transmitting media. The information can be of various forms like audio, video, image, text, etc. The most widely used form of data communication is text. Some kinds of text are non-compatible with either devices or networks. To make the text secure, the technique called cryptography [1] is usually utilized.

Audio and video are the other forms of information sharing. But, both of these forms require more storage, more bandwidth to transmit and dedicated applications for their processing.

An image is a two dimensional function, f(x, y) where x and y are the spatial coordinates and amplitude at any pair of coordinates (x, y) is called as gray level or intensity [2]. On comparison with audio and video forms, image has some advantages like less storage, less bandwidth for transmission and compatible across devices and networks. Adding to that image processing is done for various applications in different platforms [3].

Steganography [1] is a process of hiding a piece of information for transmission which is a part of cryptography , where images can be used [4] [5]. The information needs to be made secure using complex algorithms [6] which in turn takes more time to encode and also needs a dedicated hardwares and softwares.

So, steganography is the best approach to make a normal piece of information more secure [7]. But, presently, encoding by steganography involves masking [8] of information by an image. Instead, here an image is created according to the text [9] , where one pixel represents one character of the information. By implementing this technique, even text files with large size can also be compressed and encoded.

For creating and processing this image, Matrix Laboratory (MATLAB©) [10] tool is used. MATLAB© is widely used tool for processing, plotting and analyzing signals, which provides a platform for virtual simulation, for every engineering domain. It is also used in hiding information applications[11].

II. METHODLOGY

The main concern of this project is to make text transmission secure. It is achieved using the concepts of digital image processing over the MATLAB© platform. Text transmission is done by converting text to an image [9]. Since, transmission of an image over any channel is feasible and consumes less data as compared to direct text transmission, this approach is used.

MATLAB© has various inbuilt functions and tool boxes (image processing tool box in particular), to support and read any kind of text formats (file). Also, it can take instantaneous text input from the user and convert it into an equivalent image[10].

Merging [12] is also done by ex-or [13] or ex-nor operations on that equivalent image. MATLAB© makes this easier because it can be achieved by writing single line of code for multiple operations. This is advantageous as the computation time decreases as compared to other possible platforms. Using image processing tool box, one can perform all kinds of image processing techniques and transformations can be developed by the user.

Images can be either coloured or gray, but, colour image processing requires vector approach [2] and the size of the image may also increase. Here, the various formats of image used are .jpg, .tif, .png, .gif, and .bmp.

A character in a piece of text is converted to an equivalent number (which represents the intensity value of the pixel), which is obtained using either American Standard Code for Information Interchange (ASCII) or Unicode [14]. Here, ASCII is chosen over Unicode for its simplicity. Data compression is automatically achieved by implementing this algorithm for text transmission.

978-1-4673-9328-7/15/$31.00 ©2015 IEEE

III. ALGORITHM

A. Encryption

Fig. 1. depicts the flow of text to image encoding and its process flow. The encryption algorithm undergoes the following procedure:

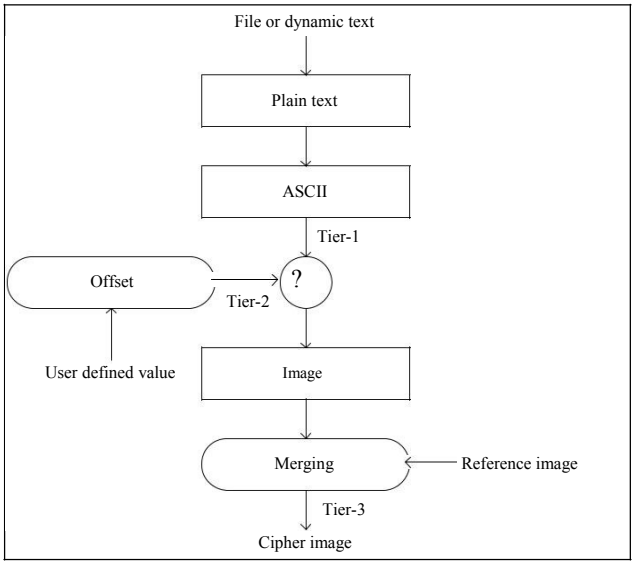


Fig. 1. Diagram of Text to Image Encoding.

1) Feeding the Text

The input text is fed either by typing it when the program is running or by giving it in a file. The text that has to be transmitted should preferably be a confidential data, that demands high security, without being prone to any attacks during transmission.

The input can be of any standard format. But, this file should not be previously encoded by Unicode or Unicode Transformation Format (UTF) [14], preferably it should be American National Standards Institute (ANSI) [15] encoded. The input text file may contain characters of any font style or font size. Hyperlinked special effect characters are automatically converted to normal plain text in this algorithm. If the text is dynamically given in MATLAB© command window, it should be inside single quotes. Here, tab and newline characters cannot be introduced in the text. All the outputs are bound to MATLAB© standards [10].

2) Conversion to ASCII

In this stage, conversion of the text obtained in the first stage to ASCII code [9], which represents the number equivalent of that text is done. The ASCII values are assigned sequentially. The text which has the assigned numbers (where, the assigned numbers indicate the gray value) contribute to the image that is created in the next stage. Ninety four different characters can be input using the keyboard, whose ASCII value ranges from 32 to 126 [14] .

The main reason for choosing ASCII is that it is defined for all the characters that are generally used , and also that it represents eight bit data so that, eight bit depth gray image can be easily created.

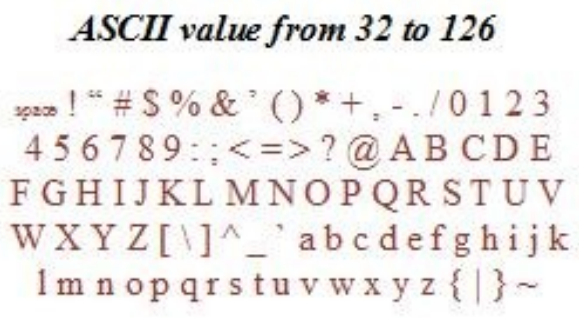


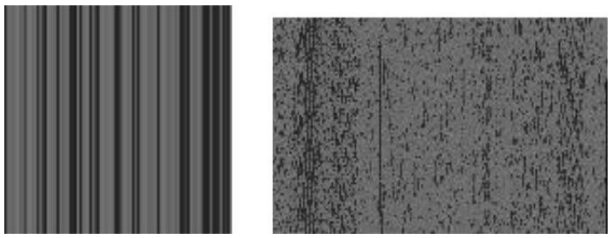
Fig. 2. ASCII values from 32 to 126.

3) Creation of an Equivalent Image

It is the most important part of the process. Here, gray scale image is created using the ASCII values previously computed. If the given text is small, then it is possible to create an image that resembles a bar code. The width of the image is the length of the text and its height left to the user, which resembles the initial row values.

Considering a file which contains many lines, image doesn't resemble a bar code but, it appears grainy. Either height or width of the image can be set by the user. If height is fixed the characters (ASCII values) are filled column wise. On the other hand if the width is fixed the characters are filled row wise. The process of filling continuous until the last character is obtained. If a matrix of proper dimension is not obtained (if a row or column is incompletely filled), then, it is completed by filling out the remaining pixels as black. The resultant image provides tier one security.

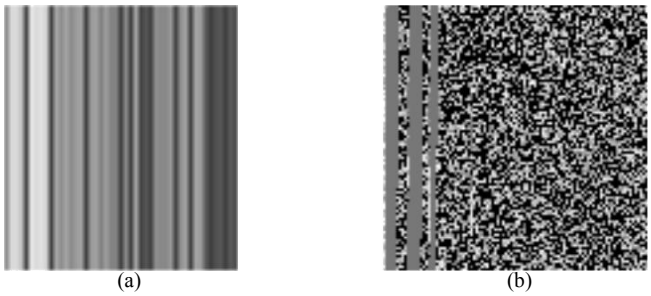
Fig 3. Shows a form of text encrypted as an image. If a third person gets this image while transmission, using histogram analysis of the image and the knowledge of ASCII values, it might be possible to retrieve the text back.



(a) (b)

Fig. 3. (a) Simple bar code resembling image for 91 character text. (b) Tier 1 encoded image for 25kB text file.

To avoid any unauthorized decryption, offsetting the equivalent image is done. Offsetting is the process of scaling up or down the pixel values based on conditions. Different conditions can be given to different pixel value ranges without any overlapping, which results in a fairly separated histogram (similar to histogram equalization), making it difficult to decrypt..



(a) (b)

Fig. 4. Offset value of 120 added up to the image for (a) simple bar code resembling image (b) file encoded image

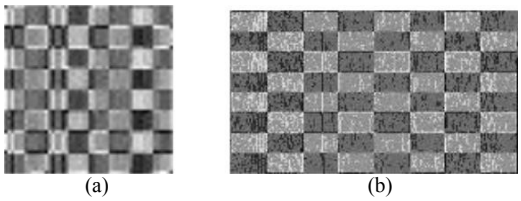
By offsetting the image, a second tier of security is provided to the text. Merging the entire image gives it a third level of security.

4) Merging of Images

This is the final step of encryption of text. Here, the obtained image is merged with the reference image. Bitwise Ex-OR or Ex-NOR operation is performed on individual pixel values [13]. Bitwise logical operation is preferred in this context since, it avoids complications and the probability of error is minimal. The reason for choosing Ex-OR or Ex-NOR operation over other logical operations is that it is reversible (decryption is possible with the same operation). In case of logical NOT operation, it uses a single image and it is easy to decrypt by unauthorized person.

The condition to be satisfied to perform the above mentioned operation is that, the size of the two images must be the same. The advantage of this technique is that any number of such images can be merged. If many images are merged, more secure will be the message. Also, bit depth [2] of the image does not vary upon merging, which does not pose further complications. It would be more suitable if the images are of the same format.

The final ciphered [8] image forms the third tier security, which is used for transmission of the message over any medium (device or protocol). Even, an entire file can be encoded using the above mentioned algorithm as shown in Fig. 5.



(a) (b)

Fig. 5. Resultant image after merging (a) Barcode resembling image and (b) entire file of 54kB memory encoded image with 4kB chess board image.

B. Decryption

On the receiver's end, decryption of the encoded image to the required message text is performed. At first, the receiver should know whether the received image is merged or not. If the image is already merged by the sender, then, the receiver can de-merge the image using the same logical operations, with the knowledge of the referral images used for encoding the text by the sender.

Then, the obtained demerged image should be checked for offsetting. If so, by using a suitable operation (either by subtraction or by addition of the offset value for the suitable range), offset is nullified. Further, the obtained image is checked for its features and a decision is taken, verifying whether it is a file image or a barcode type image.

In case of a file image, the individual pixel values are reverted back to their character forms. It can be either displayed on the command window of MATLAB© or can written back to a file. If it is a barcode type image, through iterations, each row pixel values are reverted until a meaningful message is obtained. This is how, deciphering of the ciphered image to text takes place.

In summary, using the MATLAB© function 'double()' [10], conversion of text characters to their respective ASCII values is achieved. Similarly, using the function 'char()' [10], ASCII value to their corresponding character conversion is done.

The simplicity of this algorithm can be expressed by the following expressions:

1) Merging:

Let 'A' represent the ASCII equivalent matrix of the message text. 'B' be the pixel matrix [2] of the referral image to be merged. If 'X' is the output merged pixel matrix, $\oplus$ being Ex-OR operation and $\odot$ be the Ex-NOR operation, then:

$$A \oplus B = X \text{ else } A \odot B = X. \qquad (1)$$

2) De-merging:

The final matrix 'A' is obtained as follows:

$$X \oplus B = (A \oplus B) \oplus B = A. \qquad (2)$$

IV. RESULT ANALYSIS

This algorithm was tested for various cases of input, like a single line text and a text file. Many parameters (like histograms [2], compression ratios, and image sizes) were analyzed and tabulated for various image formats.

Any image is analyzed by its histogram. Fig. 6. shows the histogram plot of a text equivalent image without merging and offsetting. It clearly depicts that the pixel values [2] range from 32 to 126 which corresponds to the ASCII values of the commonly used characters.:

Fig. 7. Represents the merged image of the text file. It can be compared to histogram equalization technique. This is the true cipher image that can be used to transmit confidential messages or secure text file with less chance of being tampered by a third unknown party [1].

The number of pixels corresponding to an intensity value in the histogram for a simple barcode type image is given by the following equation (3).

$$\text{Number of pixels} = (\text{Frequency of a character appearing}) * (\text{Height of image}) \qquad (3)$$

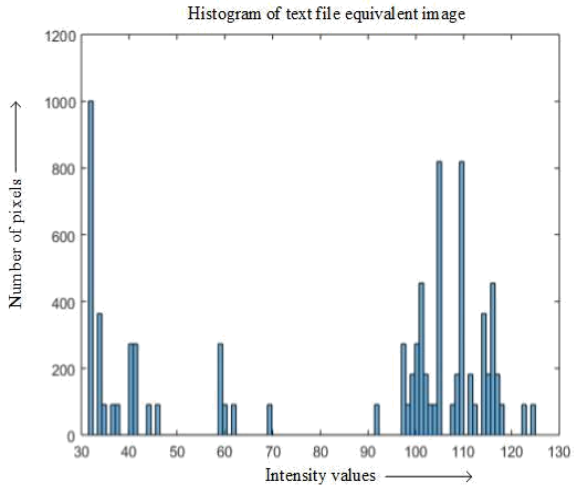


Fig. 6. Histogram of text file equivalent image.

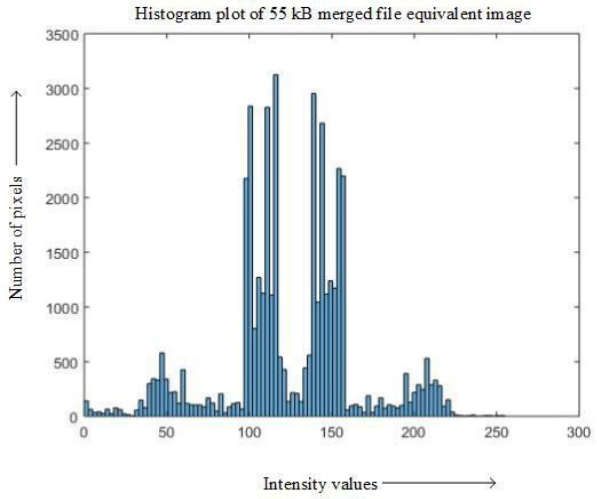


Fig. 7. Histogram plot of text file equivalent image after merging

TABLE I. STORAGE SIZE OFDIFFERENT IMAGE FOR DIFFERENT NUMBER OF CHARACTERS BEFORE MERGING.

Number of dynamic characters	Image size (in Bytes)				
	.bmp	***.gif***	***.jpg***	***.png***	***.tif***
20	1,478	937	412	114	618
50	3,678	1,366	869	163	2,748
100	9,450	2,274	2,041	243	8,586

TABLE II. STORAGE SIZE OF DIFFERENT IMAGE FORMATS FOR DIFFERENT NUMBER OF CHARACTERS AFTER MERGING.

Number of dynamic characters	Image size (in Bytes)				
	.bmp	***.gif***	***.jpg***	***.png***	***.tif***
20	1,478	1,189	617	462	618
50	3,678	2,203	1,616	960	2,748
100	9,450	4,026	3,491	1,542	8,586

TABLE III. STORAGE SIZE OF DIFFERENT IMAGE FORMATS ALONG WITH .RAR BEFORE MERGING.

File size	Image size (in Bytes)					.rar
	.bmp	***.gif***	***.jpg***	***.png***	***.tif***	
8,370	6,278	5,068	2,320	4,438	5,502	2,012
24,706	18,678	15,667	6,258	14,425	18,014	6,732
55,038	43,478	35,697	12,941	33,344	42,830	16,632
117,632	57,878	44,112	15,191	42,592	57,306	42,560

TABLE IV. STORAGE SIZE OF DIFFERENT IMAGE FORMATS AFTER MERGING.

File size	Image size (in Bytes)				
	.bmp	***.gif***	***.jpg***	***.png***	***.tif***
8,370	6,278	6,581	2,879	4,720	5,502
24,706	18,678	18,858	7,315	14,813	18,012
55,038	43,478	41,902	15,059	34,126	42,816
117,632	57,878	52,051	18,117	43,463	57,304

$$\text{Compression ratio} = \frac{\text{Uncompressed data size}}{\text{Compressed data size}} \quad (4)$$

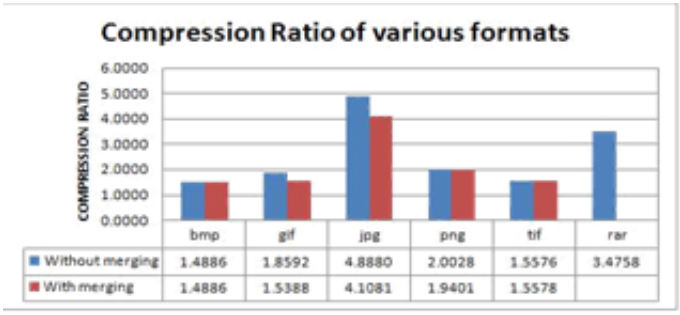


Fig. 8. Graphical representation of compression ratios of various image formats with compressed .rar fie

By the above analysis, it is clear that .jpg has much lesser storage data and .bmp file size does not change after merging. .tif has approximately the same storage size, even after merging. It shows that, irrespective of the pixel values in these two formats, its size depends only on the maximum pixel value and its dimensions.

It is evident from the graph that, the images with .jpg extension has higher compression ratio than that of normal .rar (zip) file. Thus, along with secure communication [3], size compression is also achieved. This feature can be advantageous while secure transmission.

V. CONCLUSION

A. *This algorithm has some of the following applications:*

1) Secure Communication

- Compared to messages that are coded using other algorithms, which can be decoded after several try. It is not the case in this algorithm as it provides tier three security in which many images can be used merged, which should also be present at the receiver end.
- In comparison with symmetrical cipher model [3], here plain text is the input text or text file, offset value and the merging image is the secret key. Encryption algorithm is encoding of text to image and decryption is the reverse procedure of that.

2) Large Text Transfer

This algorithm also results in compression of data. So large text can be safely transmitted along with compression. For example, a 22kB file can be compressed to a 6kB .jpg file.

3) Storage

Huge amount of text can be saved in a limited. amount of space by storing the text in the form of an image using this algorithm.

B. *This algorithm can be further improved mainly in the following mentioned ways:*

- Using colour image processing, more security and compression (may be) can be achieved.
- Using larger bit depth or pixel depth, more than one characters can be encoded in a pixel.
- Processing speed and computational time can be greatly enhanced, if this algorithm is implemented on a suitable hardware or software platform.
- Secure communication that is present today, can be integrated (like a coded message can be further coded using this algorithm) to enhance the security of the message even more.

Communication is a very crucial aspect in one's life. It is also important for the communicated message to be secure. This is an attempt to make secure transmission of messages between the sender and the receiver digitally. Implementation of this algorithm is user friendly, and this form of encoding may open up newer horizons of encryption algorithm. Any disadvantages in the algorithm will be rectified by proper up gradations.

ACKNOWLEDGMENT

The work reported in this paper is supported by the college through the Technical Education Quality Improvement Programme [TEQIP-II] of the MHRD, Government of India.

REFERENCES

[1] William Stallings .'Cryptography and Network Security Principles and Practice' 5th Ed, Prentice Hall Publishers, 2011.

[2] Rafael. L Gonzalez, Richard. E Woods, 'Digital Image Processing', 3rd Ed., Prentice Hall Publishers, 2006.

[3] C Gururaj, D Jayadevappa, Satish Tunga, " An Effective Implementation of Exudate Extraction from Fundus Images of the Eye for a Content Based Image Retrieval System Through Hardware Description Language", Third International Conference on Emerging Research in Computing, Information, Communication and Applications (ERCICA – 2015) published in Springer, ISBN: 978-81-322-2552-2, 31st July and 1st August 2015, pp 279 – 290, NMIT, Bengaluru, India, DOI: 10.1007/978-81-322-2553-9.

[4] John Justin M, Manimurugan S,"A Survey on Various Encryption Techniques" in Magnetism, International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-2, Issue-1, March 2012.

[5] Rinki Pakshwar, Vijay Kumar Trivedi, Vineet Richhariya, "A Survey On Different Image Encryption and Decryption Techniques." International Journal of Computer Science and Information Technologies(IJCSIT). Vol. 4 (1) , 113 – 116, 2013, ISSN: 0975-9646

[6] Sandeep kaur, Sukhpreet Singh, Sonia, "A review on Image encryption techniques," International Journal of Emerging Trends & Technology in Computer Science (IJETTCS). Volume 2, Issue 3, May – June 2013. ISSN: 2278-6856

[7] Shikha Vidhu, Kiran Dutt, "Steganography: The Art of Hiding Text in Image using Matlab," International Journal of Advanced Research in Computer Science and Software Engineering. Volume 4, Issue 9, September 2014 . ISSN: 2277 128X

[8] Paul A.J, Lekshmi R. Nair "Matrix based Substitution and Diffusion Procedure for Fast Image Encryption," International Journal of Computer Applications (0975 – 8887) Volume 80 – No3, October 2013.

[9] Ahmad Abusukhon, Mohamad Talib , Issa Ottoum, " Secure Network Communication Based on Text-to-Image Encryption," International Journal

of Cyber-Security and Digital Forensics (IJCSDF) 1(4): 263-271 The Society of Digital Information and Wireless Communications (SDIWC) ISSN: 2305-0012. 2012

[10] MATLAB© website, http://in.mathworks.com/products/matlab/ .

[11] Agniswar Dutta, Sankar Das, Asoke Nath, Abhirup Kumar Sen, Shalabh Agarwal, "New Data Hiding Algorithm in MATLAB using Encrypted secret message", 2011 International Conference on Communication Systems and Network Technologies.

[12] Gunasekaran G. and Bimal Kumar Ray, "Encryting and Decrypting Image using computer visualisation techniques," Asian Research Publishing Network (ARPN) VOL. 9, NO. 5, MAY 2014 ISSN:1819-6608.

[13] Reza Moradi Rad, Abdolrahman Attar, and Reza Ebrahimi Atani, " A New Fast and Simple Image Encryption Algorithm Using Scan Patterns and XOR," International Journal of Signal Processing, Image Processing and Pattern Recognition(IJSIP). Vol.6, No.5 (2013), pp.275-290 ISSN: 2005-4254

[14] Unicode website, www.unicode.org/charts/ .

[15] ANSI website, www.ansi.org/