American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)
ISSN (Print) 2313-4410, ISSN (Online) 2313-4402

http://asrjetsjournal.org/

An Information Security System Based on Optimized Pixel Mapping Method

Su Mon Thu[a]*, Khin Myo Thant[b]

[a,b]*Mandalay Technological University, Mandalay and 100/107, Myanmar*
[a]*Email: susu052@gmail.com*
[b]*Email: khinmyothant97@gmail.com*

Abstract

In the field of communication system and internet, the information security is playing a crucial role. At the present time, communication through Internet is becoming popular and it should be accurate and secure. To make it reality, cryptography is very useful tool to protect the content of confidential data in many research areas. Another one, steganography is also commonly used to hide the existence of data into cover media known as text, image, audio and video. In this work, it is considered that the two most popular techniques (Cryptography and Steganography) should be combined to develop the strong and robust security system. At first, confidentiality and message authentication requirements are fulfilled with the help of Byte-Rotation Encryption Algorithm (BREA) and Secure Hash Algorithm-512(SHA-512). On the other hand, a new data hiding approach based on the optimization of Pixel Mapping Method (OPMM) also provides the system the higher embedding capacity with minimum degradation of stego image quality. To analyze the performance of the proposed OPMM, the comparison is also presented in terms of embedding capacity and Peak Signal to Noise Ratio (PSNR) values.

Keywords: Cryptography; Steganography; SHA-512; BREA; OPMM; Embedding Capacity; PSNR.

---

* Corresponding author.

1. Introduction

In the today's world, Internet and Network applications are growing very fast. As more and more information has been transmitted over the Internet, security is the most technological challenge in the Internet and network applications. Moreover, a lot of end users can apply tools to synthesize and edit the information and it has been security problems to expand technology which protects the integrity of data content and safe the conscientious property rights of owners. Cryptography and steganography are essential tools to solve such kinds of security problems [1, 2].

As likely, a coin has two sides, steganography conceals the existence of a message but does not protect the content of a message. Cryptography is a tool to supplement it. Steganogrpahy can be classified into image, text, audio and video steganography depending on the cover media used to embed the secret data. Similarly, cryptography can also be categorized into text, image, audio and video cryptography depending on the content of data to scramble the content [3]. In steganography, image steganogrpahy is more famous than other because of embedding secret message within an image as noise to it, which is nearly impossible to be perceptible by human eyes [4-6].

In Cryptography, the advantages of symmetric key algorithms are that they can be designed to have the high rates of data throughput. Some hardware implementations achieve encryption rates of thousands of megabytes per second while software implementations may attain throughput rates in the megabytes per seconds range and they are much faster than asymmetric key encryption. Therefore, the symmetric key encryption algorithm (BREA) is used in this system. Moreover, the one-way hash function (SHA-512) is also applied in this work because the message authentication plays a vital role to boost the security.

The proposed system design (sender and receiver portions) is presented in Section II. Section III explains background theories which consist of SHA-512 hash algorithm, BREA encryption algorithm and PMM embedding algorithm. Results and discussion are described in section IV. Security consideration is presented in section V. Finally, conclusion and future work is illustrated in section VI.

2. Design of the Proposed System

To enhance the security, the system is proposed by using SHA-512 hash algorithm, BREA symmetric key algorithm and OPMM embedding algorithm in this paper. The architecture of the proposed system is composed of two portions: sender site and receiver site that are illustrated in figure 1 and figure 2, respectively.

In the sender site, the original message (M) and secret value (S) are firstly concatenated to generate a hash code H (M | | S) through SHA-512 hash algorithm. And then, the plaintext (M) and the hash code (H) are also concatenated. Secondly, the resulting message is encrypted using BREA symmetric key algorithm with the secret key to generate the ciphertext. Finally, the ciphertext is embedded in a cover image by using OPMM embedding algorithm to form the stego image.

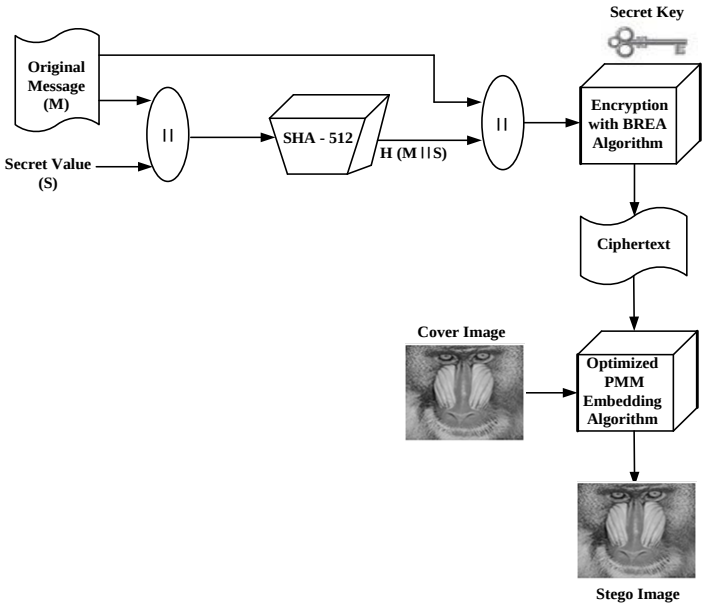


Figure 1: proposed system design for sender site

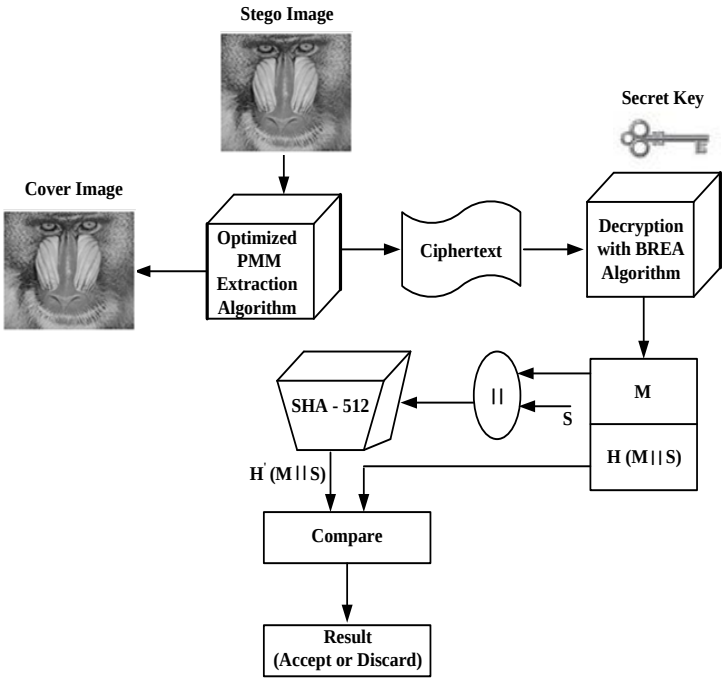


Figure 2: proposed system design for receiver site

In the receiver site, the receiver extracts the ciphertext message using the OPMM extracting algorithm. And then, the ciphertext message is decrypted using BREA decryption algorithm with the secret key to get the concatenated value that includes the original message and hash code. To check the message integrity, the message (M) and secret value (S) are concatenated to calculate the hash code H' (M | | S) with the help of SHA-512 hash algorithm. Finally, the resulting two hash messages H (M | | S) and H' (M | | S) are compared whether it is identical or not. If it is exactly the same, it ensures the data integrity.

3. Background Theory

In this paper, SHA-512 hash algorithm, BREA symmetric key algorithm and optimized PMM (OPMM) embedding algorithm are integrated to develop the proposed system.

3.1 Secure hash algorithm-512 (SHA-512)

The algorithm takes as input a message with a maximum length of less than 2^{128} bits and produces as output a 512-bit message digest. The input is processed in 1024-bit blocks. Figure 3 depicts the overall processing of a message to produce a digest [7].

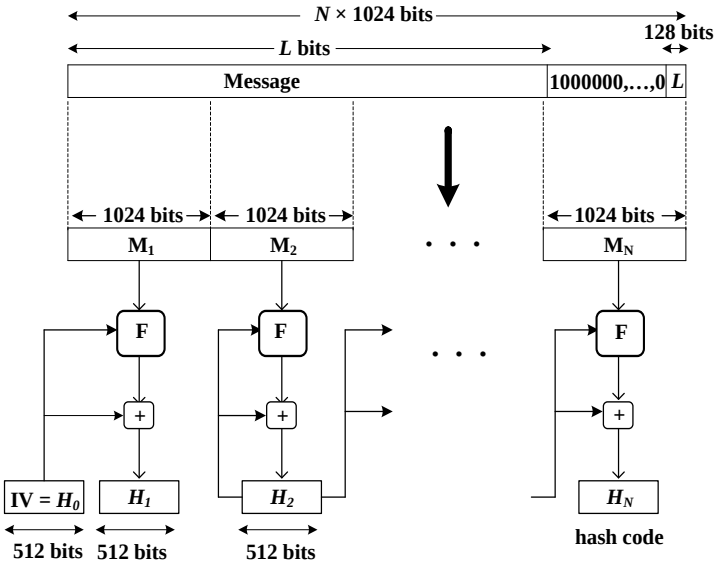


Figure 3: message digest generation using SHA-512 [7]

The processing consists of the following steps:

- **Step 1: Append padding bits.** The message is padded so that its length is congruent to 896 modulo 1024 [length = 896 (mod 1024)]. Although the message is already the desired length, padding is always added. Thus, the amount of padding bits is ranging from 1 to 1024. The padding consists of a single1-bit chased by the essential number of 0-bits.
- **Step 2: Append length.** A block of 128 bits is added to the message. This block is treated as an unsigned 128-bit integer (most significant byte first) and includes the length of the original message (before the padding). The result of the first two steps yields a message that is an integer multiple of 1024 bits in length. In Figure 3, the expanded message is represented as the sequence of 1024-bit blocks M_1, M_2,..., M_N, so that the total length of the expanded message is *N* x 1024 bits [7].
- **Step 3: Initialize hash buffer.** A 512-bit buffer is used to hold intermediate and final results of the hash function. The buffer can be represented as eight 64-bit registers (a, b, c, d, e, f, g, h).
- **Step 4: Process message in 1024-bit (128-word) blocks.** The heart of the algorithm is a module, F which has 80 rounds as shown in figure 3.

- **Step 5: Output.** After processing all *N* 1024-bit blocks, the output from the N^{th} stage is the 512-bit message digest [7].

3.2 Byte – rotation encryption algorithm (BREA)

BREA enhances the security as well as speed of the encryption scheme. The BREA is applied on different blocks of plaintext and executes in parallel manner through multithreading concept of single processor system. The BREA possesses the following features:

1. It is a Symmetric Key Block Cipher Algorithm.

2. Each block size is of 16 bytes.

3. Size of Key matrix is 16 bytes.

4. Values of Key matrix are randomly selected and ranging from 1 to 26.

5. Mono alphabetic substitution concept is followed.

6. Byte-Rotation technique is used [8].

The operation of BREA algorithm performs the following steps:

1. The letters of alphabet are assigned numerical values from 1 to 26 in sequence i.e. A, B, C, ..., X, Y, Z assigned numerical values 1, 2, 3, ..., 24, 25, 26 respectively, the digits from 1 to 9 assigned numerical values from 27 to 35 respectively and the zero (0) remains as it is.

2. The plaintext is partitioned into fixed-length blocks of size 16 bytes (or 128 bits) each. These blocks are represented by a matrix M_p.

3. The values of Key matrix (K) are randomly selected from the range 1 to 26. The size of Key matrix is equivalent to the block size of plaintext i.e. 16 bytes.

K = [k, k2, ..., k16]

K = Random (1, 26, 16)

4. Calculate the Transpose matrix of plaintext block matrix (M_p), which is denoted by M_P^T.
5. Calculate encrypted Key matrix K_e using the following formula:

$K_e = K \bmod 2$

6. Add both the matrices M_P^T and K_e and the resultant matrix is denoted by C_{pk}.

$C_{pk} = M_p^T + K_e$

7. Rotate first three rows horizontally of C_{pk} matrix such that rotate one byte from first row, rotate two bytes from second row, rotate three bytes from third row and fourth row remains untouched. The resultant matrix is denoted by C_{hr}.

8. Rotate first three columns vertically of C_{hr} matrix such that rotate one byte from first column, rotate two bytes from second column, rotate three bytes from third column and fourth column remains untouched. The resultant matrix is denoted by C_{vr}.

9. Replace numeric values of C_{vr} matrix by their corresponding letters and if 36 exist in C_{vr} matrix, it is replaced by the special character #. The resultant matrix is denoted by C_e .

After sending the cipher text block C_e and key matrix K to the recipient, the reverse process is executed using BREA decryption algorithm to decrypt the cipher text into plaintext [8].

3.3 Pixel mapping method (PMM)

In spatial domain, PMM is an information hiding technique, which is used to embed a secret message into a gray scale image. In embedding process, the input message is firstly converted into digital form and the pixel of cover image is selected as a seed. The embedding pixels are secondly selected based on pixel selection algorithm, in which the seed pixel's intensity value and its 8 neighbours are selected in counter clockwise direction as shown in figure 4 [9]. It can be seen that the starting pixel of counter is at the right of seed pixel in the PMM method. It is needed to check whether the selected embedding pixels or its neighbours lie at the border of the image or not before embedding. If the seed pixel is at the border, the pixel is not considered as a seed. If these pixels does not lie at the border of image, each four bits of the secret message is mapped into each of the neighbor pixels. This is illustrated in table 1.

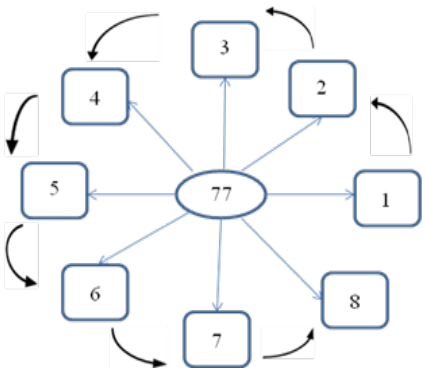


Figure 4: sequence of data embedding [9]

In the following algorithm, the next seed pixel is generated based on the coordinates of previous seed pixel and its intensity. It includes a decision factor (dp) which is based on intensity with a fixed way of calculating the next pixel.

Input: C, previous pixel position (x, y), pixel intensity value (v).

Consider dp (Decision Factor) = 1 if (intensity ≤ 80), dp = 2 if (80 < intensity ≤ 160), dp = 3 if (160 < intensity ≤ 255).

t = x + 2 + dp

if **(** t ≥ N **)** m = 2, n = y + 2 + dp

else m = x + 2 + dp, n = y

Return m and n.

End

For embedding 4 bits data, the mapping rules are illustrated in Table 1, where the 0^{th} and 1^{st} pairs of data bits are masked into the 5^{th} and 6^{th} position of pixel. And then, the parity and intensity of pixel is changed on the basis of 2^{nd} and 3^{rd} pair of message bit [10].

Table 1: Mapping rules for hiding 4 bits per pixel [10]

Message bits (2^{nd}/3^{rd} bits and 0^{th}/ 1^{st} bits pair)		Pixel intensity	Parity
00	00	Even	Even
	01	Even	Even
	10	Even	Even
	11	Even	Even
01	00	Even	Odd
	01	Even	Odd
	10	Even	Odd
	11	Even	Odd
10	00	Odd	Even
	01	Odd	Even
	10	Odd	Even
	11	Odd	Even
11	00	Odd	Odd
	01	Odd	Odd
	10	Odd	Odd
	11	Odd	Odd

In other word, the embedding process is described as follows:

- Step 1: Mask the 5^{th} bit from left with the 0^{th} message bit in the binary number of pixel intensity value
- Step 2: Mask the 6^{th} bit from left with the 1^{st} message bit in the binary number of pixel intensity value
- Step 3: Use the mapping rules shown in table 1to embed the 2^{nd} and 3^{rd} bits into the last 2 bits of the binary number of pixel intensity value
- Step 4: Select the next neighbouring pixel $P_{r'c'}$ based on the previous pixel $P_{r'c'}$ to embed the next message bits and repeat

- Step 5: Generate Stego image

Like embedding process, the neighbouring pixel $P_{r'c'}$ of the first seed pixel P_{rc} is selected to extract the first four bits as the following steps:

- Step 1: Find the first seed pixel and its neighbours
- Step 2: Extract the 0^{th} message bit from 3^{rd} bit of the binary number of pixel intensity from right
- Step 3: Extract the 1^{st} message bit from 2^{nd} bit of the binary number of pixel intensity from right

- Step 4: Extract the 2^{nd} message bit from the zeroth bit of the binary number of pixel intensity from right
- Step 5: Extract the 3^{rd} message bit depending on the number of ones of $P_{r'c'}$ of intensity (cnt) i.e. if cnt = 0, the 3^{rd} bit is 0 else the 3^{rd} bit is 1
- Step 6: Select the next neighbouring pixel $P_{r'c'}$ based on the previous pixel $P_{r'c'}$ to extract the next message bits and repeat
- Step 7: Convert the binary message to the original message

3.4 Optimized pixel mapping method (OPMM)

In this paper, PMM is optimized to be higher embedding capacity with minimum distortion of image quality and this method is known as optimized pixel mapping method (OPMM). In the OPMM, the seed pixel and its 8 neighbors are also selected in counter clockwise direction, but the pixel selection process depends on the pixel intensity value of the seed pixel. It is divided into same size blocks which consist of seed pixels and their eight neighbours. The continuous nine blocks are horizontally selected at a time whereas the seed pixels and their neighbours are randomly selected in PMM. Therefore, the embedding capacity of OPMM is higher than that of PMM.

In OPMM, the first block is taken as a key block. The continuous remaining eight blocks are planned for embedding the message bits. Before embedding, the embedding block is chosen that has the least difference between the binary bit of secret message and the pixel intensity value and the embedding positions are marked in the key block. To counter in clockwise direction of the embedding block and key blocks, the first pixel is selected based on the number of ones in the binary values of seed pixel. The seed pixel and its neighbours are selected by using the following algorithm.

Input: Bitmap Cover Image (img), Pixel Position (x,y), Pixel Intensity Value (V)

1. Declare integer variables blockPixels, numPixels, numBlocks, keyStartPixel, msgStartPixel, currBlock, msgBit, msgBlock, difference
2. Set difference to 0
3. Set blockPixels to 9
4. width = img.getWidth
5. height = img.Height
6. numPixels = width * height
7. numBlocks = numPixels / blockPixeks
8. Set x and y to 1
9. for (i = 0; i < numBlocks; i++)

currBlock = i % blockPixels

if (currBlock = = 0) // Key Block

V = img.getPixel at position x and y

keyStartPixel = no: of ones of binary number of V

Endif

if (currBlock $\neq$ 0) // Message Blocks

Set x and y to new values

V = img.getPixel at position x and y

msgStartPixel = no: of ones of binary number of V

Endif

x + = 3

if (x $\geq$ width)

x $\leftarrow$ 1

y + = 3

Endif

```
Endfor
Begin
   msgBit = GetMessageBits
  Begin
     Find the first message block
     lsbUsed = Get the last 4 bits of 8 neighbors
     for (i = 0; i < lsbUsed; i++)
        if (lsbBits [i] = msgBits [i])
           difference = 0;
        else
           difference += 1;
    Endfor
    Return difference
    Get the next message block and repeat
  End
Compare differences between 8 message blocks
Mark  message block with minimum difference into first position of  key block
Get next message bits to find the next minimum difference between the remaining message blocks
End
```

In the embedding process, the secret message is firstly converted into binary format and the intensity value of pixel is also converted into binary form. The last four bits of pixel are masked with the four bits of secret message and the next four bits of secret message are also embedded as the same process. Thefollowing algorithm shows the embedding process of OPMM.

Input : Cover Image(C), Message (MSG).

Bincvr = Binary number of pixel intensity value (V)

Find the first seed pixel P_{rc}.

count = 1.

while (count ≤ n)

begin (for embedding message in message surrounding a seed pixel).

m_k = Get first msg bit.

count = count + 1.

Mask the 6^{TH} bit from left with the m_k in 'Bincvr'

m_{k+1} = Get next msg bit.

count = count + 1.

Mask the 5^{TH} bit from left with the m_{k+1} in 'Bincvr'

m_{k+2} = Get next msg bit.

count = count + 1.

Mask the 8^{TH} bit from left with the m_{k+2} in 'Bincvr'

m_{k+3} = Get next msg bit.

count = count + 1.

Mask the 7^{TH} bit from left with the m_{k+3} in 'Bincvr'

End

Get the next neighbouring pixel $P_{r' c'}$ for embedding based on previous $P_{r' c'}$ and repeat.

End

Return the stego image (S).

The extracting process of OPMM is proposed in the following algorithm, in which the seed pixels with their neighbours are selected. The secret message is extracted from the selected pixel in reverse process of the

OPMM embedding algorithm.

Input : Stego image (S) , count.

Bincvr = Binary number of pixel intensity value (V)

count = count ÷ 2.

BinMsg = ” ”.

Find the first seed pixel P_{rc}.

I = 0.

while (count ≤ N)

begin (for extract message in message around a seed pixel).

Get the (First/Next) neighbor pixel $P_{r' c'}$.

Bincvr = Binary of V.

Binmsg(i) = 1^{st} Bit of Bincvr from Right.

i = i+1.

Binmsg(i) = ZerothBit of Bincvr.

i = i+1.

Binmsg(i) =3^{rd} Bit of Bincvr from Right.

i = i + 1.

Binmsg(i) = 2^{nd} Bit of Bincvr from Right.

i = i + 1.

count = count + 1.

End.

Get the next neighbouring pixel $P_{r' c'}$ for extracting based on previous $P_{r' c'}$ and repeat.

End loop.

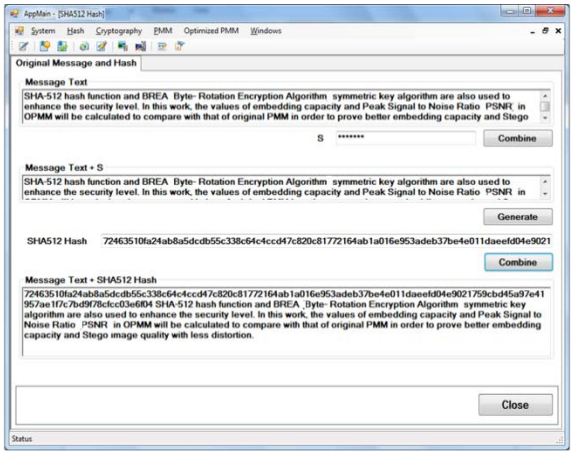
Binmsg is converted back to Original message.

Return Original Message.

End

4. Results and Discussion

The implementation results of the proposed system are represented as a series of interface. It is implemented by using C# programming language.



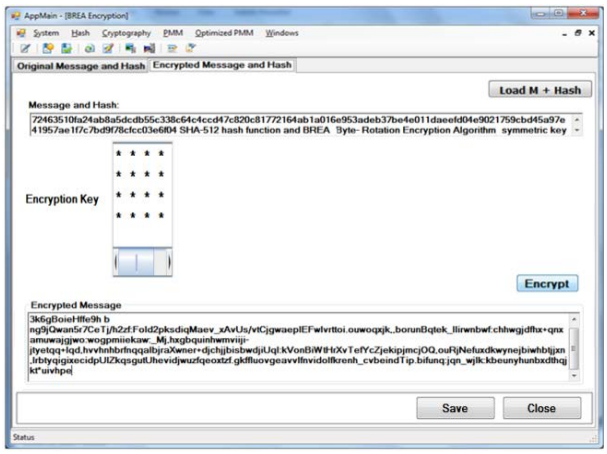


Figure 5: combination of message and hash code **Figure 6:** encryption Process

The secret value (s) is used by sharing between the sender and receiver. In figure 5, firstly, the user must type seven characters in the "s" text box. Then, the user has to click "Combine" button to concatenate the original message (plaintext) and the secret value (s). Secondly, the user has to click "Generate" button to generate hash code through SHA-512 hash algorithm. Finally, the user has to click "Combine" button to combine the original message and hash code.

In figure 6, the user must load the concatenated values of message and hash code. To encrypt these values by using BREA encryption algorithm with the help of the secret key, the user has to click "Encrypt" button and then the encrypted message (ciphertext) is generated.

In figure 7, the encrypted messages are hidden into cover image using OPMM embedding algorithm to produce stego image. To see the information of stego image, the user has to click "Information" button.

In figure 8, the receiver opens the stego image by clicking "Load Stego Image" button. To retrieve the embedded messages, he or she has to click "Retrieve Message" button. The embedded messages are the encrypted messages and so it is needed to decrypt. When the receiver clicks "Decrypt" button using the same

key, the hash code and original messages are generated as shown in figure 9.

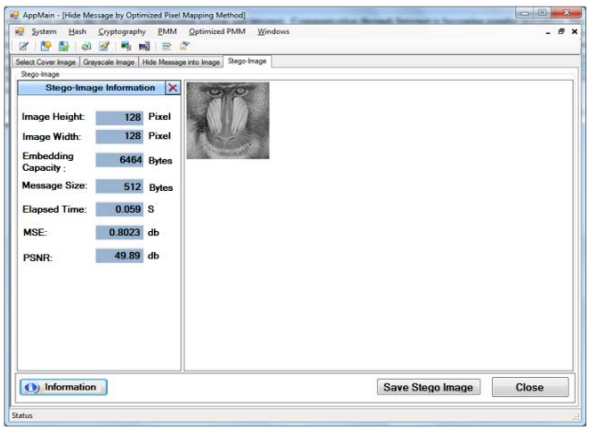


Figure 7: generation of stego image using OPMM

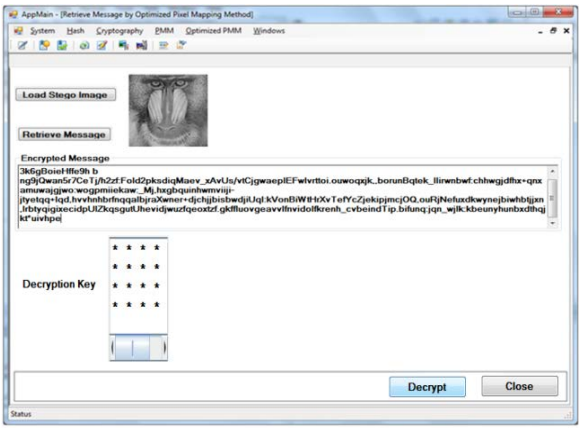


Figure 8: decryption process

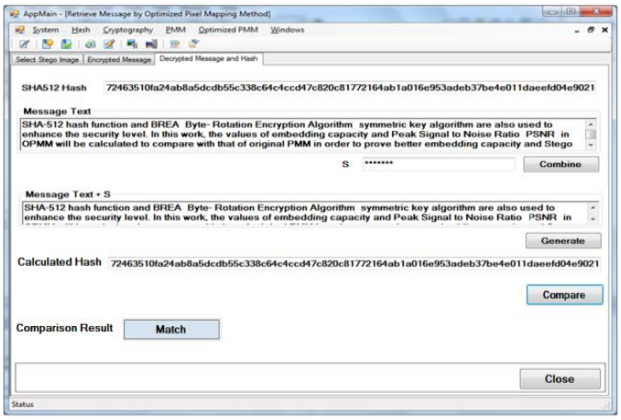


Figure 9: checking the data integrity

And then, the original messages and secret value (s) are also concatenated to calculate the hash codes in the receiver site. To check the data integrity, the hash code and the calculated hash codes are compared by clicking "Compare" button. If it is the same, it is ensured that it meets the data integrity as depicted in figure 9. Therefore, this system provides the secret message confidentiality as well as data integrity.

In this section, there are three points of view for performance consideration: embedding capacity consideration and image quality consideration by calculating the PSNR values and processing time.

The embedding capacity is considered according to the following parameters:

- Total number of pixels (P_t)
- Block Size (B_s) = seed pixel + eight neighbours
- Total number of blocks (B_t) = P_t / B_s
- Number of unused message blocks (B_{unused}) = B_t mod B_s
- Number of key blocks (B_k) = P_t / B_s^2

- Eight neighbor pixels surrounding the seed pixels (N)
- Number of hidden bits in each pixel (T)

The total number of seed pixels is computed using Equation (1).

$$Number\ of\ Seed\ Pixels\ (S) = B_t - B_{unused} - B_k \qquad (1)$$

The embedding capacity can be calculated by using Equation (2).

$$Embedding\ Capacity\ (Bytes) = S \times N \times T \qquad (2)$$

To do the comparison results of OPMM and PMM for embedding capacity, the experiment is done into three different image sizes (Image Type – Baboon and Message Size = 224 bytes). According to the experimental results, it can be found that the embedding capacity values of OPMM are higher than that of PMM as shown in figure 10.

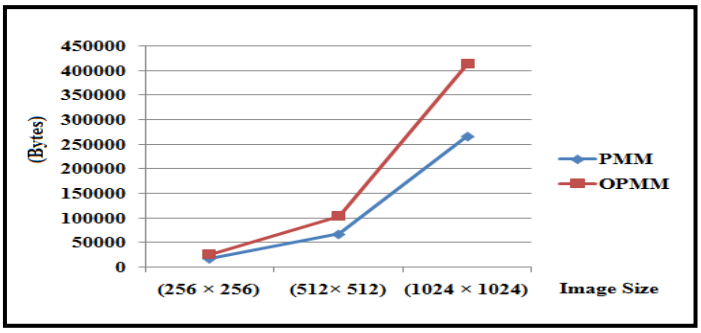


Figure 10: comparison results of embedding capacity

PSNR measures the quality of the image by comparing the original image or cover image with the stego image. After embedding the secret message in the cover, the PSNR is used to calculate the stego image quality. Mean squared error (MSE) of the stego image is calculated as Equation (3).

$$MSE = \frac{1}{[N \times N]^2} \sum_{i=0}^{N} \sum_{j=0}^{N} [C(i, j) - S(i, j)]^2 \qquad (3)$$

Where,

C (i,j) = Cover image that consists of N by N pixels

S (i,j) = Stego image

The PSNR is computed by using Equation (4).

$$PSNR = 10\ log_{10}\ 255^2 / MSE\ dB \qquad (4)$$

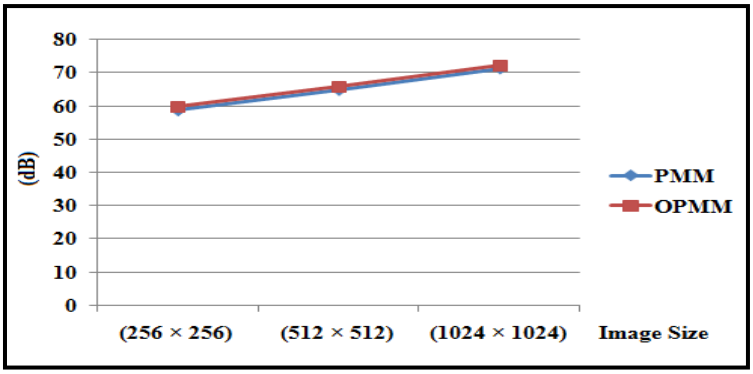


Figure 11: comparison results of PSNR.

The experiments were conducted based on the three image sizes (Image Type – Baboon and Message Size = 224 bytes) to calculate the PSNR values of stego image. The experimental results prove that the PSNR values of OPMM and PMM are almost identical as shown in figure 11. Therefore, there is no major changes for the quality of OPMM' stego image.

Moreover, the embedding capacity and PSNR values are more increased in OPMM than in PMM. The increased percentage of embedding capacity and PSNR values for OPMM is also calculated as depicted in table 2.

Table 2: Increased percentage of embedding capacity and PSNR values for OPMM

		Percentage Increase	
Image	Image size	Embedding capacity	PSNR
Baboon	256 × 256	57.699 %	1.446 %
	512 × 512	56.291 %	1.265 %
	1024 × 1024	55.575 %	0.826 %
Lena	256 × 256	49.988 %	0.185 %
	512 × 512	49.262 %	0.261 %
	1024 × 1024	49.727 %	1.736 %

The processing time (second) of OPMM and PMM can be compared on the three image sizes by implementing in Baboon image with the message size (224 bytes) as shown in Figure 12. After embedding the message within cover image, the processing time of OPMM is slower than that of PMM because of matching the message bits with the last four bits of the neighbour pixels before embedding the message bits. According to the experimental results, the OPMM method is more efficient in capacity and security, but the processing time is slightly slower than the PMM method.

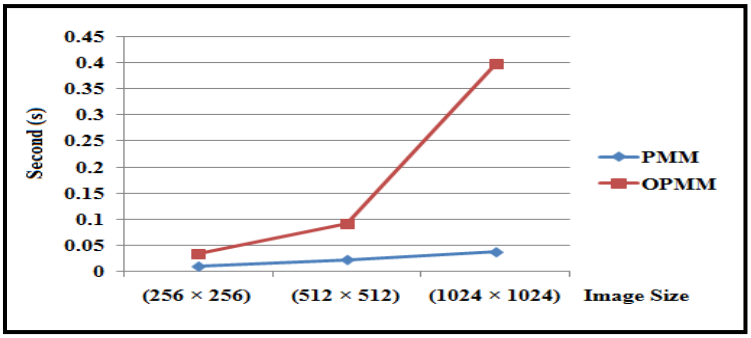


Figure 12: comparison results of processing time

5. Security Consideration

In the proposed system, the proposed method (OPMM) is based on PMM. The heart of PMM depends on the pixel selection algorithm. In PMM selection algorithm, the 8 neighbours of the seed pixels are selected in counter clockwise direction before embedding the message bits into cover image. The starting pixel or point is taken into account on the right of the seed pixel. However, the starting pixel within 8 neighbours is selected by counting the number of ones of the seed pixel and then the next pixels are selected in counter clockwise direction in the proposed method (OPMM). At the security point of view, everyone cannot guess easily how to extract the secret message from the stego image without knowing the starting embedded pixel. Therefore, the proposed method (OPMM) is more secure than PMM.

6. Conclusion & Future Work

Because of combining SHA-512 hash algorithm, BREA encryption algorithm and OPMM embedding algorithm, the proposed scheme is a secure solution to satisfy the security requirements. Moreover, the proposed system provides higher embedding capacity and less distortion of stego-image quality with the help of proposed embedding approach (OPMM). Although OPMM possesses a larger embedding capacity, the processing time is slower than that of PMM. Like PMM, OPMM can also do the processes only gray scale image as a carrier.

As further extension, it can be improved to process in the color images. Various carriers such as text, audio, video files can be added for secure data hiding systems. Moreover, the proposed approach can be enhanced not to take long processing time in order to get more efficient system. This system can be applied in a variety of security awareness applications such as secure data exchange system in economic organizations, secret communication from one military to another and so on.

As system limitation, this system can only encrypt the text files (.txt) which consists of alphanumeric characters. Among the image file formats, bitmap (.bmp) image file type can only be considered as a cover image file in this paper. The size of hidden information relatively depends on the size of the cover image. So, the cover file must be larger than the secret message file. Only single data file can be performed at one time or more than one file cannot be processed at the same time.

Acknowledgements

I would like to give my special thanks to my supervisor, Dr. Khin Myo Thant, Lecturer of our university, MTU, (Mandalay Technological University) for her invaluable recommendations, detailed guidance and patient supervision throughout the preparation of this research. I would like to express my deepest thanks to all my teachers and colleagues who lend a hand directly or indirectly during the arduous process of completing this work successfully. I also show admiration to my parents for their mental supporting.

References

[1] Menezes, Alfred , Paul C van Oorschot, Scott A.Vanstone. Handbook of Applied Cryptography. Canada: Waterloo University, 1996, pp.1-816.

[2] William Stallings. Cryptography and Network Security: Principles and practices. India: Prentice Hall in Pearson education, 2002, pp. 1- 681.

[3] Davida M. Chapman and M. Rennhard. “A practical and effective approach to large-scale automated linguistic steganography,” in Proc . The Information Security Conference, 2001, pp.156–165.

[4] Kran Bailey Kevin Curran. “An Evaluation of Image Based Steganography Methods.” The International Journal of Digital Evidence, vol. 2, pp.1-40, 2003.

[5] Jr. L. M. Marvel, C. G. Boncelet and C. T. Retter. “Spread Spectrum Image Steganography.” The International Journal of IEEE Transactions on Image Processing, vol. 8, pp.1-11, February.1999.

[6] Nasir Memon R. Chandramouli. “Analysis of lsb based image steganography techniques,” In Proc.IEEE ICIP, 2001, pp.1-8.

[7] William Stallings. Cryptography and Network Security: Principles and Practices. India: Prentice Hall in Pearson Education, 2005, pp.1-679.

[8] Sunita Bhati, Anita Bhati and S. K. Sharma. “A new approach towards encryption schemes: byte – rotation encryption algorithm,” In Proc. the World Congress on Engineering and Computer Science, San Francisco, USA, 2012, pp.1-4.

[9] Souvik Bhattacharyya and Gautam Sanyal, “A data hiding model with high security features combining finite state machines and PMM method.” The International Journal of International Scholarly and Scientific Research & Innovation, vol. 4, pp. 324–331, October. 2010.

[10] Parvinder Singh and Prince Kumar Panjabi. “An Enhanced Data Hiding Approach Using Pixel Mapping Method with Optimal Substitution Approach.” The International Journal of Computer Applications, vol. 74, pp. 36–43, July. 2013.