Research Article Volume 8 Issue No.12

Hybird Module for Exchaning Multimedia in Secercy

Hossam Taha[1] and Dr. Emad Othman[2]
[1]Higher Institute of Engineering,
Department of Computer Engineering
[2]Senior Member IEEE - Region 8
AL-ShoroukAcademy[1, 2], Cairo–Egypt

Abstract:

The importance of multimedia security is becoming more and more important with the continuous increment in digital communication on the internet. The increasing use of audio and video in a wide range of application security and privacy issues to serious attention. With the advancement of both computer and internet technology, multimedia data, such as images, audio, videos, are being used more and more widely. In order to maintain privacy or security, sensitive data needs to be protected before transmission or distribution. The main purpose of the presented paper is to establish a secure way by implementing a cryptographic module to deal with the exchanged multimedia files via insecure communication channels using a symmetric key which is generated by a sophisticated mathematical way. Both the sender and receiver share a single key. The sender uses this key to encrypt plaintext and send the ciphertext to the receiver. On the other side the receiver applies the same key to decrypt the multimedia and recover the plaintext. As a first stage the plaintext is converted to spoiled plaintext then encrypting it then transmit it to the receiver who will decrypt the spoiled plaintext too its original by using reconstruction table. The information security uses cryptography on several levels. The information cannot be read without a key to decrypt it. The information maintains its integrity during transmit and while being stored. Cryptography also aids in nonrepudiation. This means that the sender and the delivery of a message can be verified. This module was tested using many image and text files and it was proved to be strong using different cryptanalysis technique.

Keywords: Multimedia, cryptigraphy, plaintext, ciphertext, key and communication.

Introduction

For RGB each color use 8-bits which have values from (0) to (255) , when concatenate this values produce pixel value which eqwevelent to the image , every pixel has value changed based on the three maigor colors which every color is mix between them(red,green,and blue) when change value of any color , the pixel value will change according to this change, and we can get the pixel value in the 2D array that representing the image[2].

Table 1: Each color use 8-bits value

Color	RED	GREEN	BLUE
Index of bits	23 16	15 8	7 0

Table 2: RGB values for every colors

Color	HTML / CSS Name	Hex Code #RRGGBB	Decimal Code (R,G,B)
⬛	Black	#000000	(0,0,0)
⬜	White	#FFFFFF	(255,255,255)
🟥	Red	#FF0000	(255,0,0)
🟩	Lime	#00FF00	(0,255,0)
🟦	Blue	#0000FF	(0,0,255)
🟨	Yellow	#FFFF00	(255,255,0)
🟦	Cyan / Aqua	#00FFFF	(0,255,255)
🟪	Magenta / Fuchsia	#FF00FF	(255,0,255)
◻	Silver	#C0C0C0	(192,192,192)
◼	Gray	#808080	(128,128,128)
🟫	Maroon	#800000	(128,0,0)

Color	HTML / CSS Name	Hex Code #RRGGBB	Decimal Code (R,G,B)
🟫	Olive	#808000	(128,128,0)
🟩	Green	#008000	(0,128,0)
🟪	Purple	#800080	(128,0,128)
🟦	Teal	#008080	(0,128,128)
🟦	Navy	#000080	(0,0,128)

By substitution in the equation we can compute how many possible colors can get from it
RGB=(R*256^2) + (G*256^1) + (B*256^0) this make 256*256*256=16777216 posible colors[3].

Nomenclature

R	red color
G	green color
B	blue color
g	golden number
2D	2 dimentional

This table give an example if we need certain color like white and green colors(table 3)

Table(3): RGB for white and green color

An example color value	RGB
White	255*65536+255*256+255
Green	0*65536+255*256+0

2.Image encryption
The pixel is the main element in image that contain many pixel,and when we need encrypt the image we should be change the value of pixel.
By substitution in the equation of RGB we can talk there is number can't get it from the equation so we can called this number golden number(g), and we can say this number in our favor by using this number after reading the image and convert it to 2 dimensional array and compute its width and height by using image. Get width, and image. Get height (figure 1).

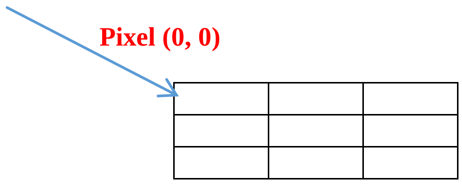


Figure (1): image contain many pixels

After this action we search include this array if there repeated number or no, if there we remove all repeated numbers except first one and place golden number in the index of repeated numbers, then we obtain new 2 dimensional array with same width and height for the original 2 dimensional array.

After this we can obtain third 2 dimensional arrays by replace all numbers in second array by zeros except golden number (g), this array is key of encryption.

when need to decrypt the encrypted image we can append the key array and array which included golden number to obtain the original array, and write this array to obtain the original image (figure2).

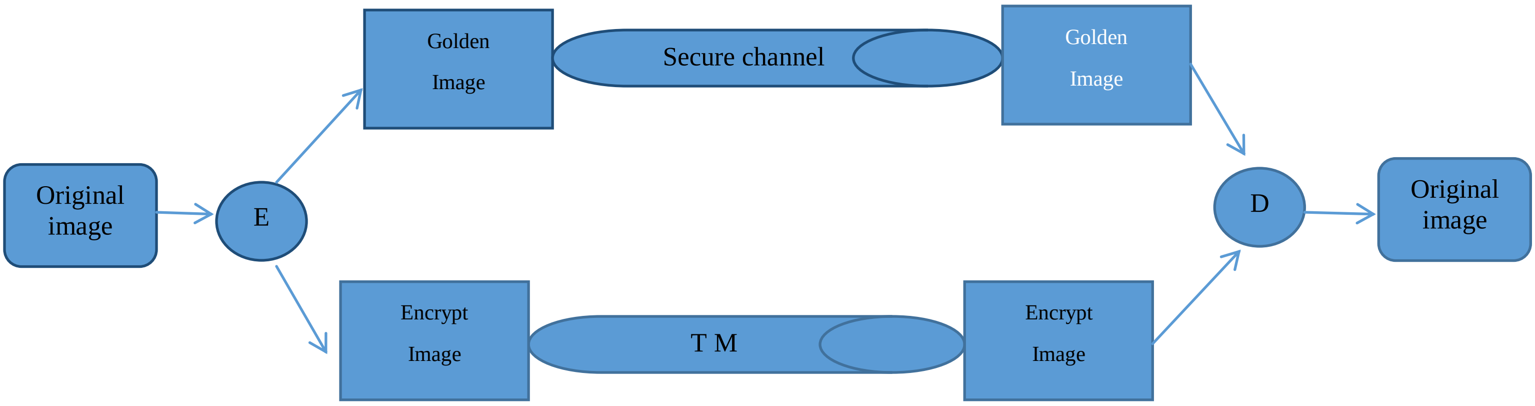


Figure (2): Schematic diagram for encryption and decryption system

For example:

1 2 3 5 3 1 7 1 9   1 2 3 5 g g 7 g 9   0 0 0 0 3 1 0 1 0

1 2 3 5 3 1 7 1 9

Image (plaintext) → first step of encrypted image → Key → original values

Figure (3): original arrayconvert to encrypted array

To make the encryption process little more difficulty we should make segmentationover encrypted image by segment the encrypted image in group of rows and columns entered by user (figure 4).

Example for segment the image in 4 columns and 4 rows the figure show the dimensional of image
Figure (4): segmentation sectors

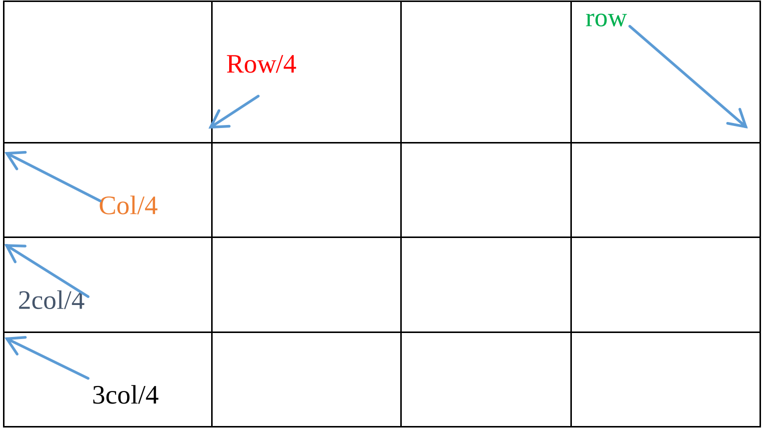
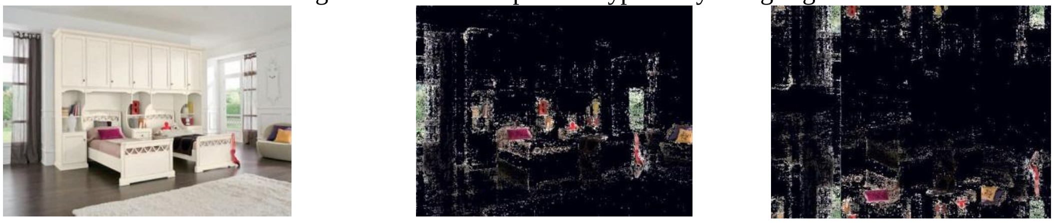


figure 4-a show the original image , figure 4-b show the image after replacing all repeated numbers with golden number, figure 4-c show the image after second step of encryption by using segmentation.



a)Original image   b)First step encrypted image   c) Second step (segmented

Figure(5):Encrypt and segment image

3. Message encryption

Use this method in encrypt a message by ask the user to enter a message, for example "hello my dear doctor emad" To be encrypted.

To make the encryption process little more difficulty we must make the following:

a. Spoil the original message by storing only a letter and the positions of the character and other like letters in the plaintext message (table 1 show this positions).

b. Shifting each letter on the spoiled message by a given number of places determined by in the user (user enter number of shifting which he needed)[5].

In this method we ask the user to enter the message that we need encrypt it then we compute the size of message (length of message) to create a reconstruct array his size equal to the original message, after that we create ragged array which contains the position of each letter (contain the total index), and create a temporary array to store index of every letter once [9].

We can decrypt the original message by using reconstruction array which have positions of every letter in the original message (table4).

For example:

Plaintext message:" hello my dear doctor emad"

Spoiled message will be string collchar"helo mydarct"

The positions of a given letter in the original message are shown in (table 2) as follows:

Table (4): reconstruction table

letter	First index position	Second index position	Third index position	Fourth index position
H	0			
e	1	10	21	
l	2	3		
0	4	15	18	
space	5	8	13	20
m	6	22		

y	7			
D	9	14	24	
A	11	23		
R	12	19		
C	16			
t	17			

Create a secret key for each letter of the spoiled message according to the following equation:

Letter special secret key= (int) (letter)*∑ (positions of the letter and the other like letters in the original message)

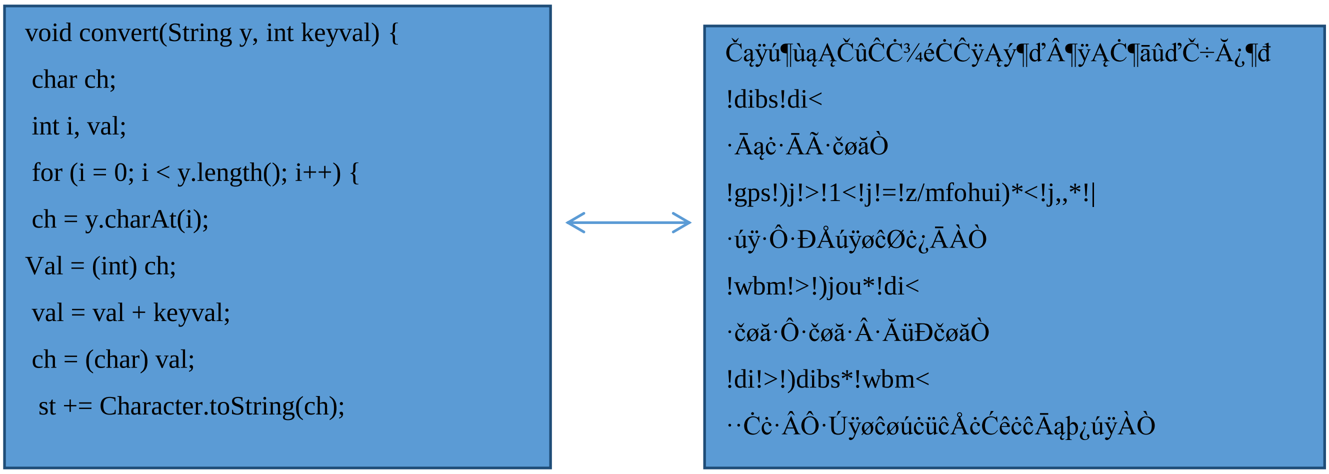
This method encodes the given message string using a Caesar by shifting each letter by a given number of places [11].

4-Text encryption

If we need encrypt a text file put the text which we need encrypt it in test plantext, after that take first word in text and convert every character to its equivalent integer number and make operation with a key which the user entered it , then convert this number to the equivalent character and obtain a first string in the text and so on.

When we obtain the encrypt words we put it in encrypted text and display this text to the user.

In encrypt text we use two methods first one to encrypt every character and the other to decrypt the original character to display it to the user (figure 6).



```
void convert(String y, int keyval) {
 char ch;
 int i, val;
 for (i = 0; i < y.length(); i++) {
 ch = y.charAt(i);
Val = (int) ch;
 val = val + keyval;
 ch = (char) val;
  st += Character.toString(ch);
```

Original text

```
ČąÿúˈùąĄČûĈĊ¾éĊĈÿĄý¶ďÂ¶ÿĄĊ¶āûďČ÷Ă¿¶đ
!dibs!di<
·Āąċ·ĀÃ·čøăÒ
!gps!)j!>!1<!j!=!z/mfohui)*<!j,,*!|
·úÿ·Ô·ĐÅúÿøĉØċ¿ĀÀÒ
!wbm!>!)jou*!di<
·čøă·Ô·čøă·Â·ĂüĐčøăÒ
!di!>!)dibs*!wbm<
··Ċċ·ÂÔ·ÚÿøĉøúċüĉÅċĆêċĉĀąþ¿úÿÀÒ
```

Encrypt text

Figure (6): Encrypt and decrypttext

5-Conclusion

This paper in hands introduces the multimedia files encryption and decryption such as image encryption, and message encryption by using substitution method to encrypt files and using reconstruction table to decrypt the Plaintext.

REFERENCES

[1] Sicari, S., Rizzardi, A., Grieco, L.A., Coen-Porisini, A. (2015). Security, privcy and trust in the Internet of Things: The Road ahead, *Computer Networks*, 76 (15) 146-164.

[2] Mahmoud, R., Yousuf, T., Aloul, F., Zualkernan, I. (2015). Internet of Things (IoT) security: Current status, challenges and prospective measures, *in:* 10th Int. Conf. for Internet Technol. and Secured Transactions, p.336 – 341.

[3] Alvi, Sheeraz A., Afzal, Bilal. Shah, Ghalib. Atzori, Luigi. Mahmood, Waqar (2016). Internet of Multimedia Things:Vision andChallenges, Ad Hoc Networks, 2016

[4] Efremov, S., Pilipenko, N., Voskov, L. (2015). An Integrated Approach to Common Problems in the Internet of Things, Procedia Engineering, V 100, p. 1215-1223.

[5] Sujitha, R., Vijaya Raghavan, N., Suganya, K. S., Devipriya A. (2014). A Novel Survey On Internet Of Things And Its Application, *IJAICT* 1 (8) 2348 – 9928.

[6] Cisco VNI, (2014). The Zettabyte Era: Trends and Analysis, June 2014.

[7] Asghar, M N., Ghanbari, M., Fleury, M., Reed, M.J. (2014). Sufficient encryption based on entropy coding syntax elements of H.264/SVC. *Multimedia Tools and Applications*, 74 (23) July 2014.

[8] Saha Arunabh n.d. *Overview of Multimedia Security*, s.l.: s.n. [Online] Avaiable at: http://www.academia.edu/8199308/Overview_of_Multimedia_Security [Accessed 15 11 2015].

[9] Thuraisingham Bhavani., 2007. *Security and privacy for multimedia database management systems*, pp. 14-29. [Online] Available at: https://www.utdallas.edu/~bxt043000/Publications/Journal-Papers/DAS/J33_Security_and_privacy_for_multimedia_database_management_systems.pdf [Accessed 1.11.2015]

[10] Amit Pande; Prasant Mohapatra; Joseph Zambreno, n.d. Securing Multimedia Content using Joint Compression and Encryption, s.l.: s.n. [Online] Available at: http://spirit.cs.ucdavis.edu/pubs/journal/amit-multi.pdf [Accessed 15 10 2015].

[11] Sonal Guleria, Sonia Vatta, 2013. *TO ENHANCE MULTIMEDIA SECURITY IN CLOUD COMPUTING ENVIRONMENT USING CROSSBREED ALGORITHM*, 2(6), pp. 562-568. [Online] Available at: http://www.ijaiem.org/Volume2Issue6/IJAIEM-2013-06-26-083.pdf [Accessed 11 10 2015].