Journal of Thai Interdisciplinary Research
Volume 14, Number 1, Pages 29 - 34

The prototype development of traceability system of LSB steganography in image files with Base64 and MD5 encoding

Paisan Simalaotao

Computer Science, Faculty of Science and Technology, Nakhon Pathom Rajabhat University, Nakhon Pathom 73000, Thailand

Abstract

The objective of this research is to study and develop the prototype of traceability system using LSB steganography in image files with Base64 and MD5 encoding. The research methodology follows SDLC process composed of problem investigation, system analysis and design, system development and system testing. Research tool is the prototype of traceability system using LSB steganography in image files with Base64 and MD5 encoding, developed from HTML5, JAVA and PHP scripts connecting to MySQL database on Apache Web Server. Files were converted to ASCII binary bits and encoded with Base64 and MD5 sequentially, then transformed to binary bits again to replace the last bit of each pixel which was converted from the original image file bit by bit.

From the result of the experiment on the image files which have different resolution (72 pixel/inch), the file size of converted image files were not different from the original image files. When unhidden and decoding, the complete original files were discovered and traceable. It can be concluded that the prototype of traceability system using LSB steganography in image files with Base64 and MD5 encoding is efficient in terms of processing time and non-different file size from original.

Keywords: traceability system, cryptography, steganography, least significant bits, MD5, base64

Article history: Received 19 June 2018, Accepted 28 February 2019

1. Introduction

Information service on the Internet is one of the most important communication channel but there is a critical problem of the distribution of untruthful information leading to loss of credibility because it cannot be verified to the original source of information. Therefore, traceability is a guideline to verify and certify the trustworthy of information.

From the issue mentioned above, [1] information was encoded and hidden in the forms of watermark on an image in order to be decodable to read hidden information. Steganography technique was used to securely hide digital information in image files [2]. It can also be applied in internet communication [3]. We choose encoding and steganography techniques on image files to trace and verify the authenticity of the original. This prototype was developed on the web application service by blending original bits to pixel bits of image files. Users can view images exactly as the original one, and can also read the hidden message and trace back to the source and decide whether the received message can be trusted.

2. Materials and Methods

2.1 Objectives

The objective of this research is to develop the prototype of traceability system using LSB steganography in image files with Base64 and MD5 encoding.

2.2 Literature reviews and related works

2.2.1 Steganography is a secure technique to invisibly embed message in the original file. It is difficult to get the hidden information with unknown cryptography. There are several techniques in steganography, for example, batch steganography, permutation steganography, least significant bits (LSB), bit-plane complexity segmentation (BPCS) and chaos based spread spectrum image steganography (CSSIS) [4].

The standard and popular technique of steganography is LSB which transforms original file bits encoding with 64 maximum numbers and 128 maximum randomizations to produce 256! possibilities which are difficult to decode hidden information without unknown exactly key. This is a streaming encoding technique which takes times depending on file size and encoding numbers. The benefit of using key is even a few key modification makes changes of encoding and decoding

* Corresponding author; e-mail: paisan.smlt@gmail.com

processes. Therefore, it is appropriate in steganography which provides more secure if information has been compressed before embedded in the image file [5].

2.2.2 MD5 is an algorithm improved from MD4 developed by Ronald Rivest [6] following Merkle-Damgard. The weakness of MD4 founded by Den Boer and Bosselaersand published in 1991 [7] and formally published in 1995 by Hans Dobbertin [8]. Rivest improved MD4 functions by increasing cycles of MD4 algorithm. The 2nd cycle was modified from XY V XZ V YZ function to multiple XZ V Y (z') function [6] and the order of the 2nd and 3rd cycle access were switched. There are modifications of the numbers of shifting and different constants. The result of previous cycle was used. These mentioned modifications improve efficiency in collision management in MD4 [9]. The adaption of MD4 to MD5 shows that MD5 generates more secure and efficiency in cryptography than MD4 [10]. The MD5 Algorithm are described as follows:

a) message was divided into bits

$$m\_0\ m\_1\ ...\ m\_\{N\text{-}1\} \quad (1)$$

b) Append Padding Bits: append 1 bit once and 0 bits to set the corresponding bit length

$$Input\ stream = 448\ mod\ 512 \quad (2)$$

c) MD5 uses buffer constructed from 4 words 32 bit length: A, B, C and D

word A: 01 23 45 67

word B: 89 ab cd ef

word C: fe dc ba 98

word D: 76 54 32 10 (3)

d) MD5 uses table replaced Ki numbers calculated in advance to improve speed and time of processing.

$$Ki = abs(sin(i + 1)) * 2^{32} \quad (4)$$

e) MD5 uses supplementary function to transform 32 bits 3 words to 32 bits one word by and, or, not, and xor operations

F(X,Y,Z) = (X and Y) or (not(X) and Z)

G(X,Y,Z) = (X and Z) or (Y and not(Z))

H(X,Y,Z) = X xor Y xor Z

Table 1 Base64 characters

Value	Char	Value	Char	Value	Char	Value	Char
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	A	42	q	58	6
11	L	27	B	43	r	59	7
12	M	28	C	44	s	60	8
13	N	29	D	45	t	61	9
14	O	30	E	46	u	62	+
15	P	31	F	47	v	63	/

$$I(X,Y,Z) = Y\ xor\ (X\ or\ not(Z)) \quad (5)$$

Contents of buffer (A, B, C, and D) will be mixed with supplementary function words (F, G, H and I) by 4 cycles, 16 basic operations in each cycle. The results are buffer A, B, C and D composed of MD5 fragments from the input original message. [5, 10].

2.2.3 Base64 is the algorithm for encoding and decoding in ASCII format, A - Z, a – z, 0-9, / and +, totally 64 characters [11].

a. transform each character to ASCII code

b. transform ASCII code to 8 bits of base 2, including the last 8 bits, totally 24 bits

c. divide 24 bits to 6 bits and convert to base 10

d. convert base 10 to character in Base64 as shown on Table 1

Base64 cryptography can protect the modification of the original message by transforming and converting between binary bits and ASCII code. MD5 makes the original message having their own code [12].

Base64 and MD5 encoding were chosen in the development of a prototype of traceability system for information security. LSB (Least Significant Bits) steganography technique was applied to transform the last bit of color with is the least significant to store the hidden message. One pixel of color in Image file needs 3 byte to store each byte of R, G, B. Each character was hidden in 3 pixels of the image file [13].

3. Research Methodology

3.1 Sample

The experimental image files are JPEG/JPG files which are commonly used and distributed on the Internet. JPEG/JPG file format supports 24 bits of colors (16 million colors), best for photographs and paintings of realistic scenes with smooth variations of tone and color. JPEG's compression benefits make JPEG popular especially in reducing the amount of data used for an

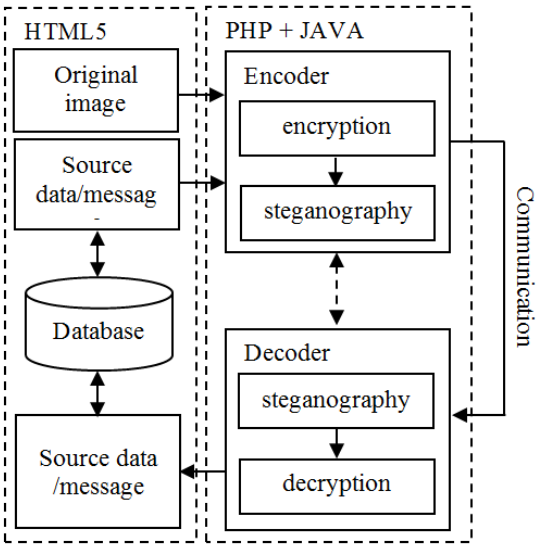

Figure 1 Conceptual framework

image in terms of responsive time of presentation on the web.

3.2 Research tool

Research tools is the prototype of traceability system using LSB steganography in image files with Base64 and MD5 encoding, developed from HTML5, JAVA and PHP scripts connecting to MySQL database on Apache Web Server.

3.3 System development

From the problem and requirement investigation, we found that the common and popular media on the Internet are images. It is necessary to investigate that images are from the trustful sources, not being forged. Traceability is important to ensure the received messages. LSB steganography in image files was applied together with Base64 and MD5 encoding,

System analysis and design as shown on Figure 1 consists of the input part: image file, source message and the verification part of unhidden message.

From Figure 1, we developed the user interface part to input the cover image file, source data/message and the traceability part using HTML5 connecting to MySQL database together with PHP and JAVA scripts. It was separated into Encoder part to encrypt message and being steganography and Decoder part which read bits from image file to unhide the message (reverse steganography) and decrypt the message from an image file.

In system development, we used HTML5 together with JAVA and PHP scripts processing on Apache web server and connecting to MySQL database to verify the hidden message as follows:

1) Input part of encrypting and steganography as shown in Figure 2.

```
START
  Step1: Input Source_data, Original_image;
  Step2: For each ASCII code of Source_data
detect;
          Step2-1-1: Apply ASCII code to binary;
          Step2-1-2: Keep data to Source_array;
  END;
  Step4: Compute Enc;
  Step5: Compute LSBSteg;
  Step6: Output SteganoImage;
END.
```

Figure 2 Encryption and steganography algorithms

```
START Compute Enc;
  Step1: For each Source_array;
      Step1-1: Split binary stream by 6 bits;
      Step1-2: Convert to Decimal;
      Step1-3: Convert to Character;
      Step1-4: MD5 encryption;
      Step1-5: For each ASCII code detect;
          Step1-5-1: Apply character to binary;
          Step1-5-2: Keep data to Source_array;
      END;
  END;
  Step2: Apply Source_array;
END.
```

Figure 3 Base64 and MD5 encryption algorithms

```
START Compute LSBSteg;
  Step1: Source_array;
  Step2: For each color detect in Original_image;
        Step2-1: Extract pixel color to R,G,B;
        Step2-2: Apply R,G,B pixel color to
binary;
        Step2-3: Replace the least significant bit
                 by one bit of Source_array;
  END;
  Step3: Apply SteganoImage;
END.
```

Figure 4 steganography algorithms

Figure 2 shows the algorithms starting from input an image file and a source message, convert ASCII to binary, then encode with algorithms in Figure 3.

From Figure 3, message in binary format was encoded in Base64 by splitting binary stream into sets of 6 bits each, convert to decimal, search in Base64 table to reverse to characters and encoded by MD5, then

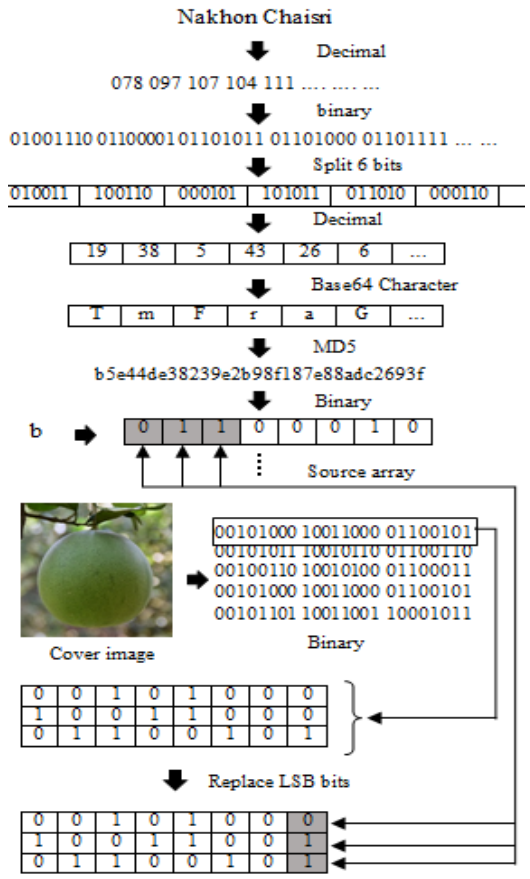

Figure 5 steganography process

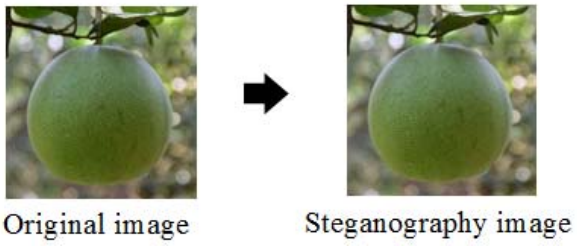

Figure 6 images before and after steganography

transform to binary to be hidden in the image file as shown in Figure 4.

Figure 4 shows algorithms of take in the result from the encryption to LSB steganography in an image file by transforming image file pixels to binary bits and replacing the encoded source message in the last bit of each pixel. It can be described as shown in Figure 5.

After encoding with Base64, MD5 and LSB steganography, a new image file was generated with a few different details on the image as shown on Figure 6.

2) The algorithms of reading message hidden in an image file and decoding shown on Figure 7.

Figure 7 shows the traceability process of steganography starting from reading the image file in order to search the hidden message as shown on Figure 8.

```
START
  Step1: Input Steganography_image;
  Step2: Compute LSBStegRev;
  Step3: Compute Dec;
  Step4: Output Source_data;
  Step5: Reverse information check;
END.
```

Figure 7 Algorithms of reading and decoding

```
START Compute LSBStegRev;
  Step1: For each color detect in Cover_image;
      Step1-1: Extract pixel color to R,G,B;
      Step1-2: Apply R,G,B pixel color to binary;
      Step1-3: Read the least significant bit
               position;
  END;
  Step2: Reverse to MD5 code;
END.
```

Figure 8 Algorithms of reading the hidden message

```
START Compute Dec;
  Step1: Reverse MD5 code to Character
  Step2: For asciicode detect;
      Step2-1: Convert to Decimal;
      Step2-2: Apply character to 6 bits binary;
      Step2-3: Apply and split to 8 bits binary;
      Step2-4: Keep data to Source_array;
  END;
  Step3: Apply Source_array to Output
Source_data;
  Step5: Reverse information check;
END.
```

Figure 9 Algorithms of decoding

The algorithms in Figure 8 show the process of reading pixels from the steganography image, converting pixels to binary bits, and searching for the bit hidden on the last bit to be decoded with MD5 and Base64 sequentially.

From Figure 9, the hidden bits were transformed to characters which are coded from MD5. Characters were converted to decimals according to Base64 table. Decimals were transformed to 6 binary bits, then grouped to 8 bits and converted to ASCII characters to generate the original message.

The decoded original message was searched to recheck with data collected in database. If matched, the system would take all attributes of that dataset to be

Table 2 Experiment on various sizes of image files

Image size	Resolution (Pixel/Inch)	Original file size (K)	Processing time (s)	Generated file size
512x384	72	576.8	1.5216	576.8
640x480	72	905.3	1.8734	905.3
720x480	72	1012.7	2.5258	1012.7
1280x720	72	2703.4	3.7125	2703.4

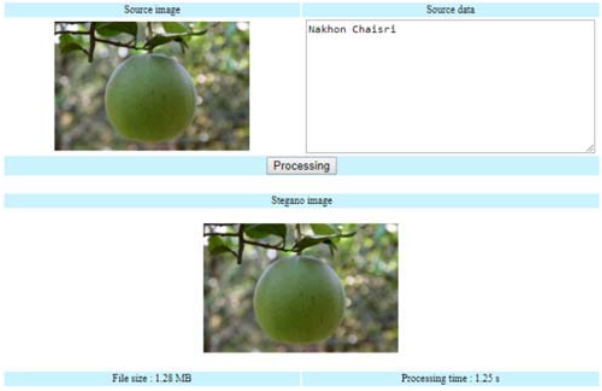

Figure 10 Testing process

Figure 11 Testing result

displayed and returned to users; otherwise, users would be notified no matched data.

Testing process was shown on Figure 10. From Figure 10, the original file and message were transformed to binary, then encoded with Base64 and MD5 to hide the massage in the image file. Figure 11 shows the output steganography image file, file size, and processing time of tracing back.

Various sizes of image files were experimented for 5 times. The average of processing time and file size were shown on Table 2.

4. Results and Discussion

The experiment shows that the original message is transformed into ASCII file, encoded with Base64 and MD5, then transformed into binary file in order to hidden JPEG/JPG file with LSD by replacing the last bit of each pixel. The encoding with Base64 with MD5 and LSD steganography on various sizes of image files with the same resolution (72 pixel/inch), the result shows that the proposed system takes more processing time on the larger file than the smaller one. However, the generated files have the same size as the original file. When reading and decoding, the hidden message is shown correctly.

5. Conclusions

It can be concluded that the prototype of traceability system using LSB steganography in image files with Base64 and MD5 encoding works effectively in terms of processing time and file size. We will further our research to apply this prototype on developing the traceability to verify the original source of product images or indicate who are the owners. Furthermore, we can experiment on other encoding algorithms or on other digital formats. In the future, the results of the research can be combined with other research, such as ontology-based knowledge management [14] and teaching media with augmented reality [15].

Acknowledgement

This research has been supported by Nakhon Pathom Rajabhat University (NPRU), Thailand.

References

[1] Xijina W, Linxiub F. The application research of MD5 encryption algorithm in DCT digital watermarking, **Physics Procedia**. 2012; **25**: 1264–1269.

[2] Cheddad A et al. Digital image steganography: survey and analysis of current methods. **Elsevier B.V. Signal Processing**. 2010; **90**(3): 727-752.

[3] Sharma et al. A study of steganography based data hiding techniques. **International Journal of Emerging Research in Management & Technology**. 2017; **6**(4): 145–150.

[4] Ibrahim R, Kuan TS. Steganography algorithm to hide secret message inside an image. **Computer Technology and Application 2**. 2011; 102-108.

[5] Nath J, Nath A. Advanced steganography algorithm using encrypted secret message**. International Journal of Advanced Computer Science and Applications**. 2011; **2**(3):19-24.

[6] Rivest R. **The MD4 message digest algorithm**. Proceedings of the 10th Annual International Cryptology Conference on Advances in Cryptology. 1990; 303-311.

[7] Den Boer B, Bosselaers A. **An attack on the last two rounds of MD4. Adv Cryptol –CRYPTO 199**; [cited 13 April 2018]. Available from: http://www.springerlink.com/index/4yb85bkpxt5gemmh.pdf

[8] Dobbertin H. Cryptanalysis of MD4. **Journal of Cryptology**. 1998; **11**: 253-271.

[9] Den Boer B, Bosselaers A. Collisions for the compression function of MD5. advances in cryptology. **EUROCRYPT '93. Lecture Notes in Computer Science**. 1994; **765**: 293–304.

[10] Thomas CG, Jose RT. A comparative study on different hashing algorithms. **International Journal of Innovative Research in Computer and Communication Engineering.** 2015; **3**(7):170–5.

[11] Sumartono I, Siahaan APU, Arpan A. Base64 character encoding and decoding modeling. **International Journal of Recent Trends in Engineering & Research**. 2016; **2**(12):63-68.

[12] AlAhmad, M. A., A. Shaikhli, et al. Protection of the texts using base64 and MD5. **Journal of Advanced Computer Science and Technology Research 2**. 2012; 22-34.

[13] Chitradevi B, Thinaharan N, Vasanthi M. Data hiding using least significant bit steganography in digital images. **Statistical Approaches on Multidisciplinary Research**. 2017; 143-150.

[14] Pakdeetrakulwong U, Hengpraprohm K. An ontology-based knowledge management for organic and good agricultural practice agriculture: A case study of Nakhon Pathom Province, Thailand. **Journal of Thai Inter disciplinary Research**. 2018; **13**(4): 26-34.

[15] Sirisukpoca U, Simalaotao P. Increase student achievement of students in third grade by teaching media with the application of local wisdom on geographical maps and the content offered through augmented reality. **Journal of Thai Interdisciplinary Research**. 2016; **11**(3):24-31.