Advances in Science and Technology
Research Journal
Volume 10, No. 31, Sept. 2016, pages 31–39
DOI: 10.12913/22998624/64016

Research Article

DEVELOPMENT AND IMPLEMENTATION OF HASH FUNCTION FOR GENERATING HASHED MESSAGE

Erfaneh Noroozi[1], Amir Ghaeedi[2]

[1] Young Researchers and Elite Club, Dariun Branch, Islamic Azad University, Dariun, Islamic Republic of Iran, e-mail: erfanehnoroozi@iaudariun.ac.ir

[2] Islamic Azad University of Dariun Branch, Dariun, Islamic Republic of Iran

Received: 2016.05.06
Accepted: 2016.07.04
Published: 2016.09.01

ABSTRACT

Steganography is a method of sending confidential information in a way that the existence of the channel in this communication remains secret. A collaborative approach between steganography and digital signature provides a high secure hidden data. Unfortunately, there are wide varieties of attacks that affect the quality of image steganography. Two issues that required to be addressed are large size of the ciphered data in digital signature and high bandwidth. The aim of the research is to propose a new method for producing a dynamic hashed message algorithm in digital signature and then embedded into image for enhancing robustness of image steganography with reduced bandwidth. A digital signature with smaller hash size than other hash algorithms was developed for authentication purposes. A hash function is used in the digital signature generation. The encoder function encoded the hashed message to generate the digital signature and then embedded into an image as a stego-image. In enhancing the robustness of the digital signature, we compressed or encoded it or performed both operations before embedding the data into the image. This encryption algorithm is also computationally efficient whereby for messages with the sizes less than 1600 bytes, the hashed file reduced the original file up to 8.51%.

Keywords: digital signature, steganography, bandwidth, authentication, hash message.

INTRODUCTION

Since people have been succeeded in making connections among themselves, the issue of confidential (private) connection came to the attention, too. At first; the application of confidential connection was mostly in martial issues. With the development of civilization, the use of ciphering in issues like politics became essential. Public key encryption technique is the desired encryption technique that each input bytes into exactly one byte as output. The secret key can be any string such as a word, a number, or just a string of random letters. Then the secret key is used to change the content of the information in the method. One of category of hiding information is steganography that is a way of inserting information in host image and its end is protecting copyright law, validation and legitimization of image [Shin and Ruland, 2013]. In all the cases, manipulation of image for inserting information should be in a licensed limit, so that there would be no damage to the image. Nowadays, various applications of steganography like monitoring the way of product distribution, ownership validation, copy control and concealed communication are introduced [Saadi et al., 2009].

Digital signature scheme and steganography are the popular techniques available to hide data securely. In fact, in a communication channel, steganography is a method of sending confidential information in a way that the existence of the channel in this communication remains secret [Thomas and Singh, 2013]. The three most important parameters for image steganography are: i) payload, ii) imperceptibility, iii) robust-

ness [Makbol and Khoo, 2013], and imperceptibility should be observed in applying the techniques which attempt to enhance the payloads or robustness. A collaborative approach between steganography and cryptography is suggested by Islam et al. [2010]. Using Public Key Infrastructure (PKI) method, the approach provides a high secure hidden data, although the size of the cipher-text is a genuine problem for the steganography. In order to preserve integrity, PKI encryption has been proposed by Wang et al. [2009]. By using a hash function of digital signature instead of PKI, we can obtained faster processing and less size for authentication of the message. In steganography, the original message remains without any change and it is just concealed by using an embedding technique into a cover medium. By doing the reverse function, we can retrieve the original data.

In computer networks, bandwidth or data transfer rate is the amount of information that can be transmitted from sender to the receiver side in the specific given time [Chen et al., 2001]. In digital signature scheme after generating digital signature, original file with the digital signature have to send to the receiver side, separately. Consequently, a high bandwidth [Jansirani et al., 2011] is required. Thus, for solving this issue, the data was transformed into encoded format and then these data were embedded in an image file, finally the image file with the much lower bandwidth is transmitted. By this scheme not only the authenticity and the integrity of images can be verified, but also the illegal modifications can be located.

The ability to create dynamic digital signature is highly related with the integrity and robustness of the image steganography. As a significant verification method, digital signature algorithm introduces a technique to endorse the contents of the message. This message has not been altered throughout the communication process [Filler et al., 2009]. Thus, it increases the receiver confidence that the message was unchanged. Two drawbacks when using digital signature schemes are extra bandwidth and large file size during transmission. Implementing an encryption algorithm in the spatial domain steganographic method can contribute to increasing the degree of security. Unfortunately, there is wide variety of attacks that effect on quality of image steganography, although there are methods for data hiding but they are still very weak in resisting these attacks. The aim of the research is to propose a new method for producing a dynamic hashed message algorithm in digital signature and then embedded into image for enhancing robustness of image steganography with reduced bandwidth.

Research objectives is to analyze the robustness of dynamic digital signature for authentication purpose by image steganography.

POPULAR TECHNIQUES AVAILABLE TO HIDE DATA

Steganography, cryptography and watermar-king are the popular techniques available to hide data securely. Steganography except for its different purpose is similar to watermarking. Hiding the existence of a message with high data capacity is the aim of steganography while the concentration of digital watermarking is mostly on the robustness of embedded message rather than capacity or concealment. The purpose and application of the first is augmenting capacity while the second's focus is on robustness. The first uses a communication channel to transfer

Table 1. Comparison of techniques available to hide data securely

Criterion/Method	Steganography	Watermarking	Cryptography
Carrier	Any digital data	Mostly image/audio	Usually text based
Secret data	Payload no changes to the structure	Watermark no changes to the structure	Plain text changes the structure
Key	Optional	Optional	Necessary
Objective	Secrete communication	Copyright preserving	Data protection
Concern	Delectability/ capacity	Robustness	Robustness
Type of attacks	Steganalysis	Image processing	Cryptanalysis
Fails when	It is detected	It is removed/replaced	de-ciphered
Relation to cover	The message is more important than the cover	The cover is more important than the message	N/A

information in an undetectable way, while the second focuses on copyright protection and looking after the legal application of a particular software or media [Mazumder and Hemachandran, 2012]. The human eye cannot detect the difference between before and after the process of embedding. Since the different between the original image and watermarked image can be easily detected, we can distinguish visible watermark easily. Watermarking for binary and halftone images is suggested by [Lee et al., 2009; Kim et al., 2006]. Their approach is using data hiding by self-toggling (DHST) scheme that is known as authentication watermarking by self-toggling (AWST) especially proper for dispersed dot halftone images than for binary images. The comparison shows that steganography can be applied in any kind of media such as image, audio and video, while cryptography is only used for text-based data.

Steganography algorithm

The most widely used algorithms for embedding data into images are the JSteg and F5 spatial domain algorithms.

Spatial domain technique

The commonly-used technique to embed message bits in DCT coefficients is the least significant bit (LSB) embedding technique [Goyal and Chutani, 2012; Ahmed et al., 2010]. The LSB insertion method alternatives the least significant bits in each pixel with the secret data; this technique is effective, easy and has high embedding capacity [Aarumugam and Rajan, 2011].

JSteg algorithm

The JSteg algorithm (Fridrich and Kodovsky, 2011) was one of the first JPEG steganography algorithms. Developed by Derek Upham, JSteg embeds message bits in the LSB of the JPEG coefficients. In this method, the index of the JPEG coefficients is not randomized to embed the message bits. JSteg embeds no message with the values 0 and 1 in DCT coefficients. This is to avoid changing too many zeros to ones because the number of zeros is extremely high [Fridrich et al., 2003]. If we compare it to the number of ones, the number of zeros that will be altered to ones is more than the number of ones being changed to zeros. As in the JSteg algorithm, the data are concealed in the Blowfish algorithm by substituting the LSB of the DCT coefficients but there is a difference which is that here all the coefficients (even those with values of 0 and 1) are used [Westfeld, 2009]. Visual changes detectable by the human eye are the result of concealing more data. Generally, the messages are embedded in lossy compressed JPEG images by the JSteg steganographic algorithm [El-Ghoneimy, 2008].

F5 algorithm

One of the most popular steganographic algorithms is F5 [Westfeld et al., 2010], which is undetectable using the chi-square technique [Filler et al., 2009]. To encode message bits, F5 uses matrix encoding. The changes are distributed evenly throughout the stego-image with the help of permutated straddling [Filler et al., 2010]. By changing only one of the places, matrix encoding can embed k bits. Any DC coefficients and coefficients with a zero value are not changed by the F5 algorithm. Since zero value coefficients are not considered in decrypting, the same message bit should be embedded in the subsequent coefficient until its value becomes non-zero (Andrew, 2009), but the histogram of JPEG coefficients will be modified by this technique in a way that can be predicted [Goljan et al., 2009].

METHODOLOGY

Operational framework

The overall aim of the research is to propose a new method for producing a dynamic hash message algorithm in a digital signature and then embedding it into an image in order to enhance the robustness of image steganography. This is accomplished by implementing the operational framework as shown in Figure 1. Phase 1 and each of the subsequent phases are explained in detail in the following sections.

The operational framework consists of three phases. In the first phase of investigation, a review of the literature was conducted on digital signatures and image steganography schemes, and previous related studies on authentication were compared. As discussed, the second phase consisted of three activities: placing the dynamic hash message in the digital signature, encrypting the dynamic hash message, and embedding the

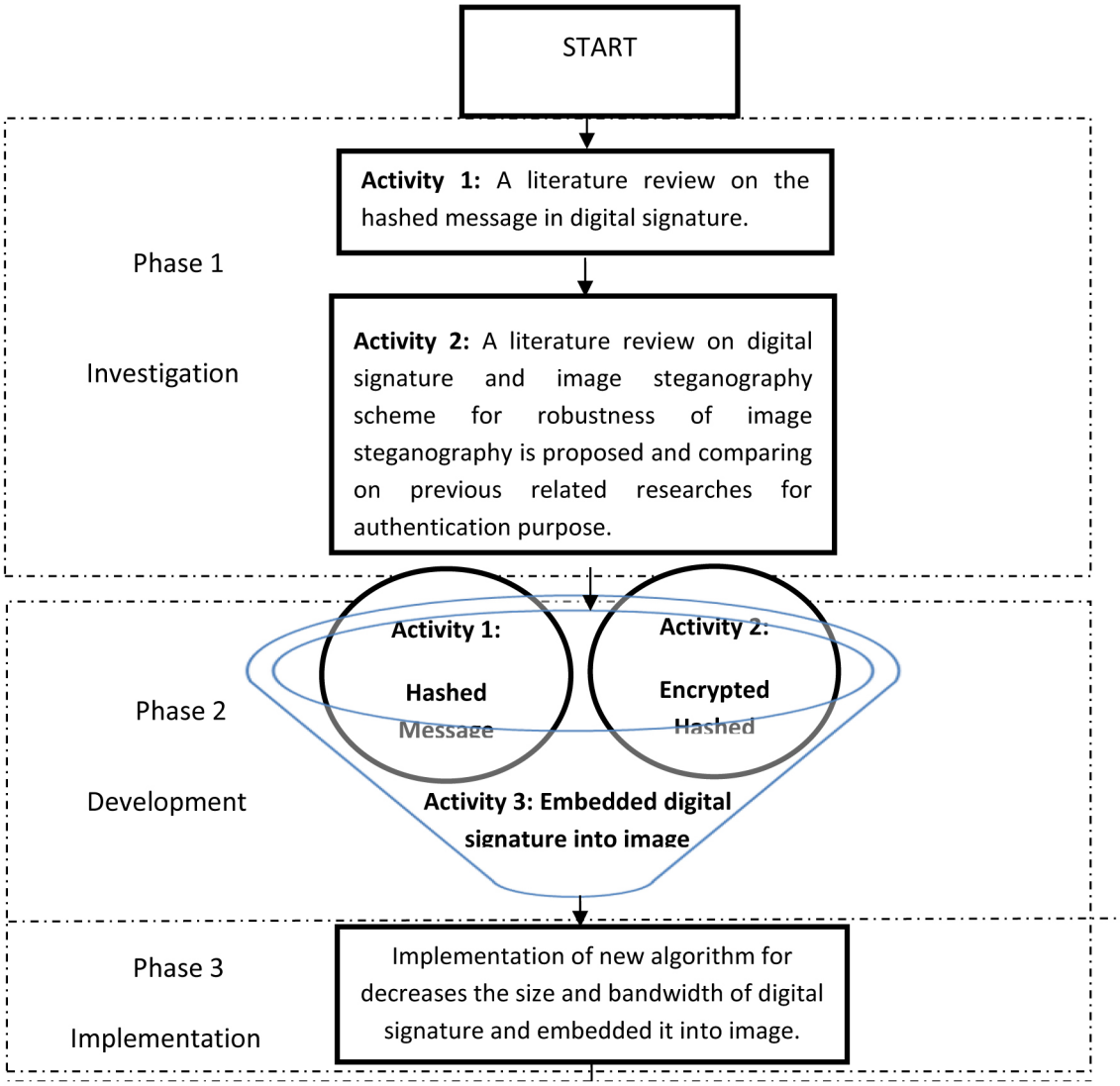

Fig. 1. Operational framework

data. The third phase involves implementation of the proposed algorithm.

DEVELOPMENT AND IMPLEMENTATION

A new digital signature that is developed by using RGB color image steganography for authentication purposes is proposed. After producing the digital signature with a small hash size, both as text and image, we can embed it into the cover image and produce the stego-image. This research is designed to fulfill the requirements of reduced bandwidth and increased robustness and imperceptibility of the embedded data.

Development of the algorithm

In the proposed algorithm, we use the first bytes of the original file by the fetching function and set the data in variable characters. A hashing algorithm is explained as the following phases: the fetch phase, compare phase and written phase. Then, the hashing function is fetched followed by the encoder function. The encoder function opens the hashing file and 16-byte private key. Once we don't reach the end of the hashed file, the first byte of the private key and the first byte of the hashing file will be XOR and the results are placed in the first byte of the key. The algorithm adopted is as follows:

```
int main (void){
int x, y; int mn; strnset (sign, passlen*2);
FILE * myfile;
Char filename [80]; text mode (C80); clrscr ( );
Printf („Signature Generator Version 1.00\n”);
Printf („Enter File Name: „);
Scanf („%s”, filename);
My file = fopen (filename,”rb”);
If (myfile==NULL) {
Printf („\nCan not Open File %s! \n”, filename);
```

```
Exit (0) ;} else {
Hashing (my file);
Printf („\hashing was Completed ...\n"); encoder ();
Printf („Result Was: %s\n", sign); fclose (my file);
Getch (); getch (); return 0 ;};
```

To convert an array of characters to the hexadecimal function and change it to hexadecimal format, two variables (Lo and Hi) are required to implement the operation from the beginning until the end of the encoded file. The algorithm for the conversion is shown as follows:

```
Int converttohex (char * data, char * result)
 {int j=0; int i=0; int hi, lo;
 For (; i<passlen; i++)
 {Hi=data[i] & 240;
 Lo=data[i] & 15;
 Hi=hi>>4;
 If (hi>9) {Hi=hi+'A"0';};
 If (lo>9) {Lo=lo+'A'-'9'-1 ;};
 Hi+='0';
 Lo+='0';
 Result [j++] =hi; Result [j++] =lo;
 }; return 0 ;};
```

Figure 2 shows the size of the original file versus the size of the hashed files. As illustrated, the average hashed file size was 8.51% less than the size of the original file. The algorithm generates a dynamic hash file. It means that the size of the hashed file depends directly on the size of the original file with an average reduction of 8.51% of the size of original file.

Implementation of the algorithm

In the implementation, the digital signature is created with a dynamic hash message and then embedded into an image as a stego-image. At the beginning of the software implementation (Excelent Modulator, Ver 1.0), a new private key is created by choosing the "new key" option from the file menu. The file, encode and decode menus will be active. The probability of producing the same keys does not exist. In the next step, we choose the "save key" option and save the private key. In the next step, the new private key with its password should be introduced to the software. As a result, we choose the "load key" option from the file menu and load the private key and the password belonging to the private key on the proposed software. The "select image option" is for choosing the RGB cover image on the system. By choosing that, the properties of the image appear such as width, height and capacity. Another option is the "import file to buffer option". At first, the input file in this software is entered in the normal buffer; then the compress buffer is created from the embedded data which reduces the size of the embedded data. To use the public-key encryption method, we need a different public key and private key. If we choose a new key in the software, we select the public key from (256!) states. In the sender section, from the encoder menu, we choose "select BMP image" that is applied for choosing the cover image. To make the stego-image, we have two kinds of buffer for handling the cover image. When we want to embed a large size file into the cover image, we should compress it and then put it into the cover image. The first state can be placed in the normal buffer into cover image by choosing the 'import file to normal buffer'. In this case, we put the embedded data into the cover image and produce stego-image. This is by choosing 'normal to bitmap' and we send it to the receiver.

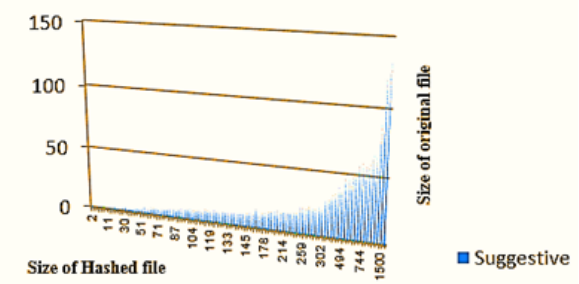

Fig. 2. A sample of 100 hashed files in digital signature

In the second state, the embedded file size is large and we want to compress it and embedded it into the image. In this case, we choose the "create compress from normal" menu. After compressing, the embedded file size and capacity decreases to 100 000 bytes and we can then easily embed it into the cover image by choosing the "put compress to bitmap" option, without causing any change in the quality of the cover image (Figure 3).

In the third state, the proposed software is able to encode the embedded data in order to elevate the security. Next, we put the encoded data into the colour RGB cover image. To operate this system, we need to choose the "import file to normal buffer" option, and then put the embedded file into the buffer. In the next step, we choose "encode to normal buffer" and encode the embedded data. Lastly, to produce the stego-image, we choose the "put normal to bitmap" option and send it to the receiver without making any changes in the image appearance. The user needs to be

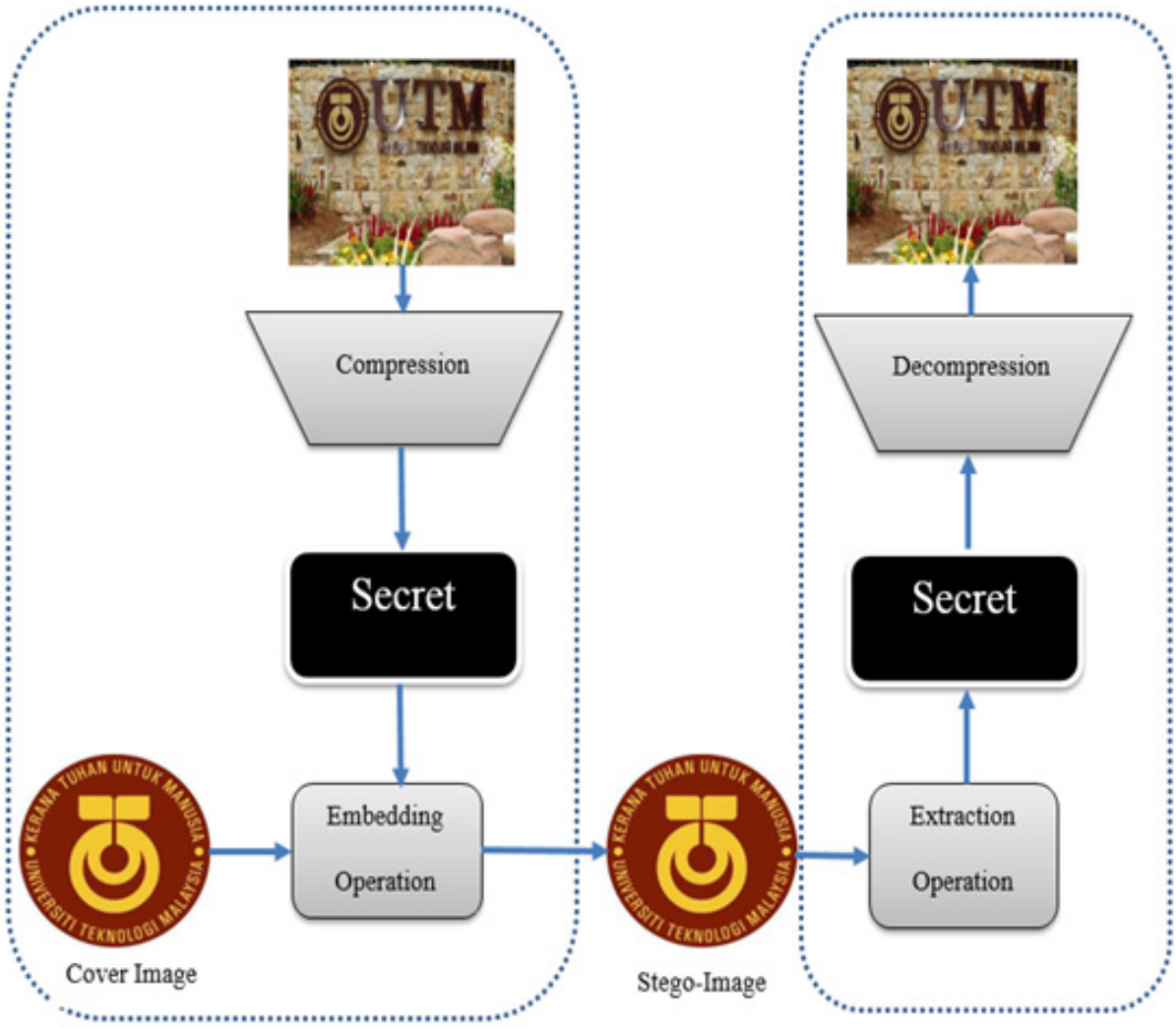

Fig. 3. Embedding in the image after compression

aware if another file is embedded as data in the cover image (Figure 4).

In the fourth state, we want to compress data with a large file size and encode it. In order to elevate security, the compressed encoded data is placed into the cover image and sent to the receiver. Next, after choosing the cover image, we select the "import file to normal buffer" option and, by doing this, we have transferred the embedded data to the normal buffer.

There are various states on the sender side, and the user is privileged to choose the states. The condition of the embedded file will determine the state that will be available on the receiver side. In the first state on the receiver side, the embedded file is in normal state without compression or encoding. In this case, at first we choose the "get normal from bitmap" option. It means that it puts the embedded image into the normal buffer. In the next step, by choosing "export file from normal buffer", we separate the embedded data from the cover image. In the second state, when the size of the embedded data is large, we choose the "get compress from bitmap" option. In this case, we fill the buffer with the compressed data and after that we choose "create uncompressed from compress" which extracts the embedded data out of the compressed state. Finally, to separate the cover image from the embedded image, we use "export file from normal buffer". In the third state, the sender has embedded the data in the encoded form in the RGB cover image; therefore, at the receiver side, "get normal from bitmap" is first applied and then the "decode normal buffer" option is selected to decode the normal buffer. The "export file from normal buffer" option is selected in order to export the embedded data from the cover image in the stego-image. In the fourth state, the sender has increased the security by using both compression and encoding and has embedded the

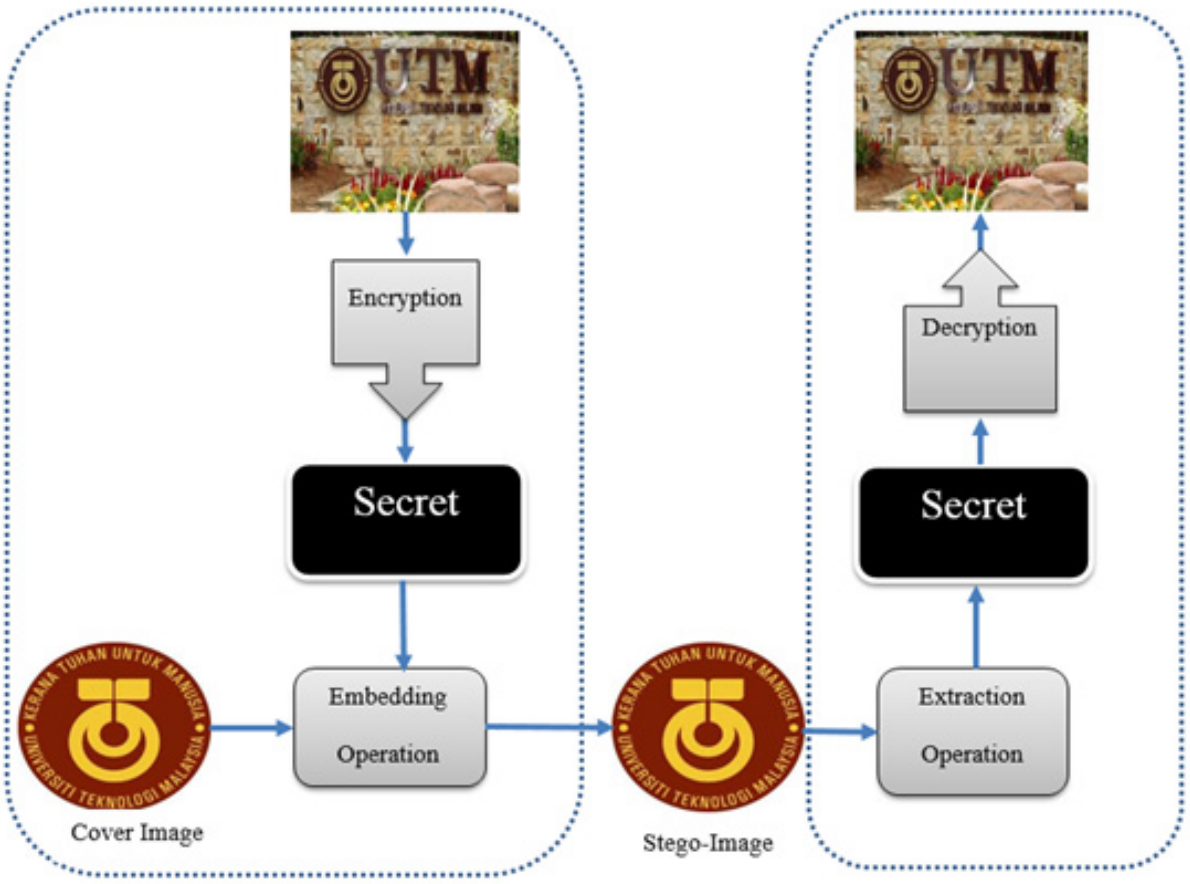

Fig. 4. Embedding in the image after encryption

encoded compressed data into the cover image. In this case, the user who receives the stego-image should use the "get compress from bitmap" option and fill the software buffer from the compressed data. In the next step, we should export it from the compressed state. We use the "create uncompressed from compress" option. In this process, we extract the embedded data out of the compressed state and now we should also export it from the encoded state. Therefore, the "decode compress buffer" option is used for decoding the uncompressed data (which is impossible without having the private key), and then the "export file from normal buffer" option is used to extract the embedded data from the cover image.

Evaluation of the hash algorithms

Some hash algorithms were compared with the proposed algorithm in terms of logical operators and the complexity of the hardware involved. The results of the comparison are presented in Table 2.

As seen in the table, it is found that more than four (4) logical operations (the rotating shift) are required than the proposed algorithm. The hardware complexity requirement for the proposed algorithm was also lower compared to the requirement for the other algorithms. Hardware complexity covers devices such as logic devices, programmable and gate arrays and application-specific integrated circuits.

The proposed algorithm had reduced significantly the sizes of the files only 8 bytes for various original file sizes compared to the other algorithms (Table 3).

CONCLUSION

One of the most important issues in digital media is the security of messages being sent throughout the network. The security provided through steganography means that unwanted users are prevented from accessing the secret data and reading the contents. There is a possibility that unwanted users can access the data and the contents, therefore techniques such as compression and encoding techniques should be used in the process of embedding and extracting data.

Table 2. Comparison of logical operations and hardware complexity

Algorithm	Logical operations	Hardware complexity
MD5 algorithm [Sinha and Singh, 2003]	AND, OR, NOT, rotating shifts	medium
SHA1 algorithm [Sternet al., 2002]	AND, OR, NOT, rotating shifts, XOR	large-scale
SHA2 algorithm [Alkhathamiet al., 2013]	AND, OR, NOT, rotating shifts, XOR	large
Proposed algorithm	OR and XOR	low

Table 3. Comparison of size of file in bytes

Size of original files (Byte)	MD5 algorithm (Byte)	SHA1 algorithm (Byte)	SHA2 algorithm (Byte)	Proposed algorithm (Byte)
14	32	40	64	8
18	32	40	64	8
72	32	40	64	8
1	32	40	64	8

* SHA is Secure Hash Algorithm.

Computer steganography methods should be essentially performed in such a way that the stego-image is not detectable from the original image. These methods of steganography can be used for different purposes. For example, in addition to secure communications, steganography can be applied to develop data banks. In this application, the task of steganography is to add accompanying information to an image in order to integrate the information and simplify saving. In these applications of steganography, the existence of the information in the image is generally clear. The system is supposed to be closed and there is no concern about an attack from outside to discover the information. In these cases, the non-resistant methods are considered suitable. One of the properties of the proposed software is that we can embed any kind of data, such as an image, text or audio file, into the cover image without making any change in the appearance or quality of the cover image. In addition, after producing the digital signature or encoded hash message, we can compress or encode it in order to have highly secure data. Another capability of the proposed software is that the embedded data can be encoded several times and, in this case, the security increases and no hacker can access the original data. Of course, on the receiver side, the same number of decoding operations must be performed.

REFERENCES

1. Shin J., Ruland C. 2013. A survey of image hashing technique for data authentication in WMSNs. In: Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on. IEEE, 253-258.
2. Saadi K.A., Bouridane A., Guessoum A. 2009. Combined fragile watermark and digital signature for H. 264/AVC Video Authentication. In EUSIPCO, Vol. 9, 1-4.
3. Thomas P., Singh A.K. 2013. A novel steganographic approach for enhancing the security of images.
4. Makbol N.M., Khoo B.E. 2013. Robust blind image watermarking scheme based on redundant discrete wavelet transform and singular value decomposition. AEU-International Journal of Electronics and Communications, 67(2), 102-112.
5. Islam R., Naji A.W., Zaidan A.A., Zaidan B.B. 2010. New system for secure cover file of hidden data in the image page within executable file using statistical steganography techniques. arXiv preprint arXiv: 1002.2416.
6. Wang Q., Wang C., Li J., Ren K., Lou W. 2009. Enabling public verifiability and data dynamics for storage security in cloud computing. Computer Security–ESORICS 2009, 355-370.
7. Chen T., Wang J., Zhou Y. 2001. Combined digital signature and digital watermark scheme for image authentication. In: Info-tech and Info-net, 2001. Proceedings. ICII 2001-Beijing. 2001 International Conferences on, IEEE, Vol. 5, 78-82.
8. Jansirani A., Rajesh R., Balasubramanian R., Eswaran P. 2011. Hi-tech authentication for palette images using digital signature and data hiding. The International Arab Journal of Information Technology, 8(2), 117-123.
9. Filler T., Ker A.D., Fridrich J. 2009. The square root law of steganographic capacity for Markov covers. In: IS&T/SPIE Electronic Imaging (pp. 725408-725408). International Society for Optics and Photonics.

10. Mazumder J.A., Hemachandran K. 2012. Review of different techniques used in recent steganography researches. International Journal of Engineering, 1(8).
11. Lee J.W., Kim S.L., Kim C.H., Koch R.H., Lee C. U., Kim H.I., Park J.H. 2009. The sdB+ M eclipsing system HW Virginis and its circumbinary planets. The Astronomical Journal, 137(2), 3181.
12. Goyal H., Chutani S. 2012. LSB Embedding in spatial domain – a review of improved techniques. International Journal of Computers & Technology, 3(1), 153-157.
13. Aarumugam G., Rajan B. 2011. Independent domain of symmetric encryption using least significantBit: Computer vision, steganography and cryptography techniques (Doctoral dissertation, Dalarna University).
14. Fridrich J., Kodovsky J., Holub V., Goljan M. 2011. Steganalysis of content-adaptive steganography in spatial domain. In: Information Hiding. Springer Berlin/Heidelberg, 102-117.
15. Fridrich J., Goljan M., Hogea D. 2003. Steganalysis of JPEG images: Breaking the F5 algorithm. In: Information Hiding. Springer Berlin/Heidelberg, 310-323.
16. Westfeld A. 2009. Fast determination of sensitivity in the presence of countermeasures in BOWS-2. In: Information Hiding. Springer Berlin/Heidelberg, 89-101.
17. El-Ghoneimy M.M. 2008. Comparison between two watermarking algorithms using DCT coefficient and LSB replacement. Journal of Theoretical and Applied Information Technology, 4(2), 132-139.
18. Westfeld P., Maas H. G., Pust O., Kitzhofer J., Brücker C. 2010. 3-D least squares matching for volumetric velocimetry data processing. In: Proceedings of the 15th International Symposium on Appliocationsof Laser Techniques to Fluid Mechnaics (pp. 5-8).
19. Filler T., Judas J., Fridrich J. 2010. Minimizing embedding impact in steganography using trellis-coded quantization. Proceedings of Media Forensics and Security III, SPIE, 7451, 715405-1.
20. Andrew J.P. 2009. European Patent No. EP 0971544. Munich, Germany: European Patent Office.
21. Goljan M., Fridrich J., Filler T. 2009. Large scale test of sensor fingerprint camera identification. Proc. SPIE, Electronic Imaging, Security and Forensics of Multimedia Contents XI, San Jose, CA.