AN AUTHENTICATED BSS METHODOLOGY FOR DATA SECURITY USING STEGANOGRAPHY –JPEG-BMP

Ravi Kumar. B[1], Murti. P.R.K.[2], Hemanth Kumar. B.

[1] Department of Computer and Information Sciences, University of Hyderabad, P.O. Central University, Gachibowli, Hyderabad 500 046, A P, INDIA
[1]ravi_budithi@yahoo.com & [2]murti.poolla@gmail.com

[3] Department of IT, R.V.R.& J.C. College of Engineering, Guntur, A P, INDIA.
[3] bhkumar_2000@yahoo.com

ABSTRACT

The goal of Steganography is to conceal information, in plain sight. Providing security to the data means the third party cannot interpret the actual information. When providing authentication to the data then only authorized persons can interpret the data. This system deals with secure transmission of data. In computer system to represent a printable character it requires one byte, i.e. 8 bits. So a printable character occupies 7 bits and the last bit value is 0 which is not useful for the character. In BSS method we are stuffing a new bit in the place of unused bit which is shifting from another printable character. To provide authentication a four bit dynamic key is generated for every four characters of the encrypted data and the key is also maintained in the data itself. In this system we implement security using steganography. ie . hiding large amount of information in an image without disturbing the image clarity and its pixels.

KEYWORDS

Bit Shifting, Steganography, Security, Authentication

1. INTRODUCTION

Steganography is an ancient art that has been reborn in recent years. The word *steganography* comes from Greek roots which literally means covered writing [1], and is usually interpreted to mean hiding information in other information. Markus Kuhn, a steganography researcher has submitted the modern definition of steganography, as "art and science of communicating in a way which hides the existence of the communication" [2]. The goal is to conceal, in plain sight, information inside other innocent information to disallow an outsider or adversary the opportunity to detect that there is a second secret message present One of the primary drivers of the renewed interest in steganography is for mitigating copyright abuses. As audio, video and other works become more readily available in digital forms, the ease with which perfect copies can be made may lead to large-scale unauthorized copying. This type of copying is naturally of great concern to the music, film, book, and software publishing industries. There has been significant recent research into digital watermarks or hidden copyright messages and digital fingerprints or hidden serial numbers. The idea is for file fingerprinting to be used to help identify copyright offenders and then potentially prosecute them with the digital watermark [1].

DOI : 10.5121/ijma.2012.4203

2. RELATED WORK

Images provide excellent carriers for hidden information. Many different steganographic techniques exist, but most can be grouped into two domains: the image domain and the transform domain. Image domain tools encompass bit-wise methods that implement least significant bit insertion and noise manipulation. These approaches are prevalent in steganographic systems and are characterized as simple systems [2]. The formats (image) used with such steganography methods are lossless; the data can be directly manipulated and recovered easily. The transform domain category of tools includes those that manipulate algorithms and image transforms such as discrete cosine transformation. These methods conceal information in significant areas of the cover and may alter image properties such as luminance. Watermarking tools usually fall in this domain. Typically, these methods are more robust than bit-wise techniques. However, a consideration must be taken as to the benefit of added information to the image versus the extra robustness obtained. Many transform domain methods are unconstrained to image format and may remain persistent for lossless to lossly, or vice versa, conversions. Some techniques share both image and transform domain characteristics. These may employ patchwork, pattern block encoding, spread spectrum methods, and masking which all can add redundancy to the hidden information. These combined approaches may help protect against some image processing techniques such as cropping and rotating. For example, the patchwork method uses a pseudo-random selection technique to mark multiple image sections (or patches). Each patch may include the watermark, so if one section is destroyed or cropped, then others may persist [3].

3. OUR PROPOSED SYSTEM

As computers continue to permeate millions of people's daily routines, their use as steganography instruments makes perfect sense. People use steganography as a means to reduce the casual interception of private information. The increase in steganography usage is due to the cover space abundance provided by digital media, particularly within the various computer file formats (e.g. BMP, GIF, JPG, PDF, WAV, HTML, TXT etc). With these almost perfect digital media and the many continuous technology advancements, there has been a rising concern for copyright abuses. This has driven much of the steganography advancements with a immense focus on digital watermarking. This promising technology is proclaimed by industry as an excellent anti-fraud and forgery mechanism. The music and movie industries have invested millions of dollars on techniques to conceal company logos and other proprietary markings in digital images, videos, and music recordings. The interest in creating a robust, tamperproof digital fingerprint has been the focus of much of the academic research in steganography. Although steganography differs from cryptography, many of the techniques and wisdom from the more thoroughly researched discipline can be borrowed. Secure information is not necessarily covert and covert information is not necessarily secure.

Past cryptography history has shown that the adversary usually knows that communication is occurring and is able to intercept it. The adversary is often aware that the information is encrypted and that in most cases will break the encryption algorithm at any cost. Thus, cryptography's underlying security is based on the difficulty of breaking the encryption algorithm. With sufficient time and resources, this decryption task has usually been achieved.

In contrast, steganography assume the adversary can intercept the cover, but cannot perceive any information besides the original cover content. The information is concealed and may have no additional security besides the actual message embedding. However, we combining the two methods for security as shown in Figure 1and 2 can be implemented by using our proposed method. In our proposed system, we have presented new algorithms named Bits Shifting and Stuffing (BSS) methodology [4, 5]. This system is hiding large amount of encrypted and

authenticated data irrespective of the size, dimensions of the image and without disturbing the clarity of the image [10].

4. METHODOLOGY

I. **Sender**
- Data encryption
- Generating key
- Authentication for encrypted data
- Data hiding in an image

II. **Receiver**
- Retrieving data from the image
- Data authentication
- Decryption

The embedded data is the message that one wishes to send secretly. It is usually hidden in an innocuous message referred to as a cover-text, or cover-image or cover-audio as appropriate, producing the stego-text or other stego-object. To restrict detection and/or recovery of the embedded data to parties who know it (or who know some derived key value) A stego-key is used to control the hiding process.

4.1 Encoding

Architecture for Encryption and constructing stego image.

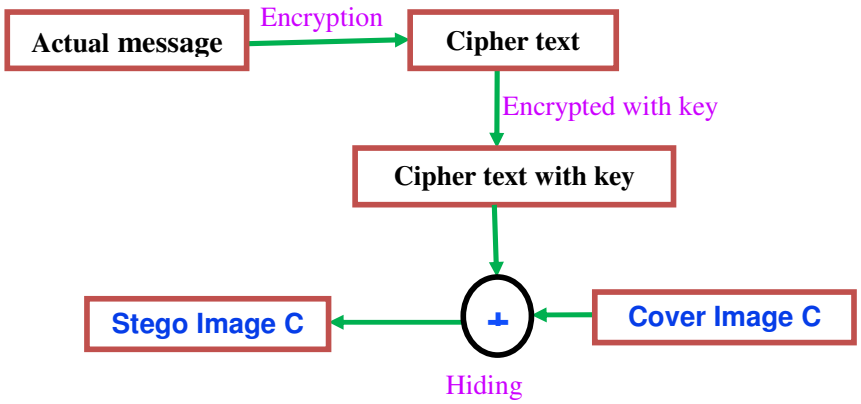

Figure 1: Encoding steps of Data hiding system in an

4.2 Encryption

In the encryption process from the message all printable characters occupies 7 bits and the last bit value is 0. In the first step I am taking 8 characters at a time from the data file and the last character 7bits are adjusted to its previous 7 characters and these 7 characters are maintained separately in another file. Like that every 8 characters are converted in to 7 characters and all these encrypted characters are maintained separately in another file (say encr1). After that by using a random function we generate a 4 bit polynomial. Take 4 characters from the encrypted file and by using this polynomial perform modulo- 2 division operation on these 4 characters then we will get a remainder of 3bits. In the second step of encryption these 4 characters, polynomial and the remainder are adjusted in 5 bytes. These five bytes are maintained separately in another

file (say encr2). Like for every 4 characters of first encrypted file after performing modulo-2 division operation, 5 bytes are maintained in the file encr2.

4.3 DATA EMBEDDING INTO THE IMAGE

This method deals with identifying the (encrypted data) **cipher text** with key (encr2.) and the image to embed the data before it can be transmitted. Open the given image file in the binary mode and find the size of the original image. This size is maintained in the image itself by using a special signature which is useful to retrieve the data from the image. Now add **cipher text** from the file encrc2.cmp to the image. Now the image is ready to transmit. If the image already contains some data you cannot add some more data for the same image. So before embedding data check whether the image contains data or not.

Algorithm represents that each image contains single message. Embedding multiple times is not acceptable with this algorithm.

1. Open the image file in the binary format
2. Check whether the signature " @!~(= " is existing or not
3. If signature found , the given image has already contains some hidden data, so select another image
4. Else find the size of the image file.
   SIZE← size of the image.
5. Open encrypted data with key in the binary mode and append each character to the end of the image.
6. At the end append the signature " @!~(= "
7. Convert each digit of the size of image into character and append after the signature to the image file.

   Conversion of size into characters.

   a. TOTAL ← SIZE
   b. While (TOTAL / 10)
      X ← TOTAL mod 10
      Char CH← (char) X
      Append this CH to the end of the image
      TOTAL ← TOTAL / 10
      End while
8. CH ← (char) TOTAL // last digit.
   Append this CH to the end of the image.
9. End.

4.4 Decoding

Architecture for Extracing data from Stego image and Decryption

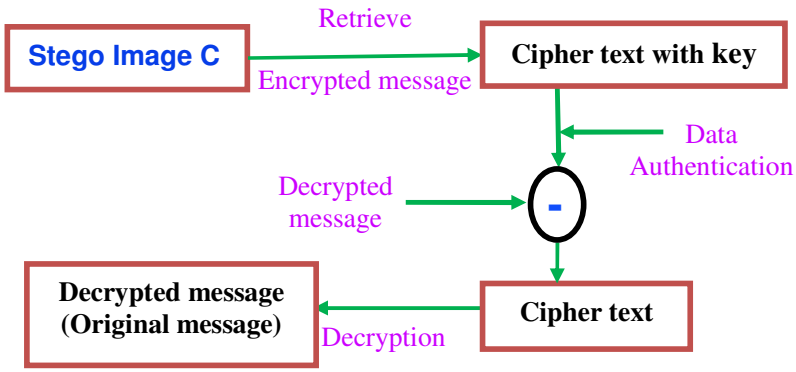

Figure 2: Decoding steps of Stego image

4.5 Decryption

Take 7 bytes from the encrypted file decypt1. and decrypt these bytes in such a way that take last bit from these 7 bytes and arrange them in a new byte. Make the last bit value of these 7 bytes as well as the new byte as 0. Now maintain these 8 bytes in special file say **data1.** This is the decrypted data. Like that for every 7 bytes from the encrypted file form 8 bytes and maintain these bytes in the file **data1**. This file contains the original data.

4.6 Data Retrieve from the image

After receiving the image through some media, the end user also opens the image in the binary mode and checks whether the image contains the special signature or not. If signature found then image contains data. Then get the original size of the image from that signature. Now except original data of the image extract data from the image up to the special signature. This is encrypted data (cipher text). This data is maintained in a file say decypt2.cmp.For this data check authentication whether data is corrupted or not and then decrypt the data to get the original message.

4.7 Decoding Algorithm

First check whether the image contains any hidden data or not. If data found then retrieve the data from the image.

1. Open the image file in the binary format
2. Check whether the signature " @!~(= " is existing or not
3. If signature not found then
   No hidden data in the image
   End
4. Else extract each character starting immediately from the signature and converting them in to digits to find the size of image.
   Conversion of last characters into digits and finding the size of image.
   a) N ← number of characters after signature.
      Character array TEMP[N]
   b) Take the last characters after the signature into the array TEMP
   c) Long Integer SIZE , B

```
Integer  A,I
SIZE ← 0, A ← 1, N← N - 1
d)  For I  0 to N
    B ← (Integer) TEMP[I]
    B ← B x A
    SIZE ← SIZE + B
    A ←  A x 10
    End  For loop.
```

5. Find position the character SIZE + 1 from the 1st character of the image and mark this position as M
6. Find total size including size of image ,data, signature and last characters after the signature. Let it be TOTAL.
7. Find the position of the last character the data. Let it be D
   N ← N + 1
   D ← TOTAL – 5 (size of signature) – N
8. Now extract characters from M to D. This is the embedded data.
   End.

5. IMPLEMENTATION RESULTS AND DISCUSSIONS

Results

5.1 Encoding *Sample Test Data*

The embedded data is the message that one wishes to send secretly. It is usually hidden in an innocuous message referred to as a cover-text, or cover-image or cover-audio as appropriate, producing the stego-text or other stego-object.

Actual message (235bytes)

Encryption

Ôhå eíâddåd äáá és ôè måsáç ôèát ïå wióèå tï såî sâãråôù® Iô é usõáìì hiddåniî án énocuoõsmessaçereæeròà to aó cïvåò-ext, ïrãoöår-éáçe oò ovår-áõio aó áprïpòéáå, pòïäãiîg ôè sôeçï-åøô oò ôhår óôçï-obêåt.

Encrypted message (208 bytes)

Authentication with key

y• +F§{'┤ k/}└ '+'I• ♡ ♡ !O¡└ ŸMO+m• EO?
››O♡ G¡└ □/}└ ¦O›K½└ K¡• /EO+™• xO+™• qO§+—-
zI│ wÉM• M• £□♡ «›®KA• gf□+##J}• sKqOM└ w
{«← {s{kŸ«~K ››/{+“/>□““/6{{¥• *}• › – {·{├┘ K-o—)I c£Å{□← —
xMk—/³K/?♡ L}• “y¯ K—+³~{O¯ m□› └ x{“‡
┘ OO“‡~} g/◘ O'□“ {?sO-y• G¡ O;/£˜OÇ-
ox□“}• ¥□+G¥┐ O§™└ “Kyo□?K§/S↕ I t

Encrypted message with key (260 bytes)

Figure 3: actual image pc1.jpg - 1024 × 768

5.2. Decoding

Figure 4: Stego image pc1.jpg - 1024 × 768

Retrieved data from stego image (**Encrypted message with key**

y• +F§{'┤ k/}└ '+'I• ▯▯ !O¡└ ŸMO+m• EO?
››O▯ G¡└ ▯/}└ ¦O›K½└ K¡• /EO+™• xO+™• qO§+—-
zI│ wÉM• M• £▯▯ «›®KA• gf▯+##J}• sKqOM└ w
{«← {s{kŸ«~K ››/{+“/>▯““/6{{¥• *}• › – {·{├ ┘ K-o—)I c£Å{▯← —
xMk—/³K/?▯ L}• “y– K—+³~{O⁻ m▯› └ x{“‡
┘ OO“‡~} g/◘ O'▯“ {?sO-y• G¡ O;/£˜OÇ-
ox▯“}• ¥▯+G¥┐ O§™└ “Kyo▯?K§/S↕ I t

Authentication

Ôhå eíâddåd äáá és ôè måssáç ôèát ïå wióèå tï sâî sâãråôù® Iô é usõáìì hiddåniî án énocuoõsmessaçereæeròå to aó cïvåò-ext, ïrãoöår-éáçe oò ovår-áõio aó áprïpòéáå, pòïäãiîg ôè sôeçï-åøô oò ôhår óôçï-obêåt.

Authenticated data (Encrypted message) (208 bytes)

Decryption

The embedded data is the message that one wishes to send secretly. It is usually hidden in an innocuous message referred to as a cover-text, or cover-image or cover-audio as appropriate, producing the stego-text or other stego-object.

Decrypted message (235bytes)

5.3 Results analysis

We have tested our method for different sets of images of JPEG and BMP as well as messages. For each and every normal jpeg and the bmp images the proposed method is working fine. After encryption the size of the encrypted data is reduced and after the dynamic key the size increases and then after decryption the size of the decrypted data is decreased, i.e original size of the actual data. The analysis for different data is shown in the below table1

Table-1

SNO	Image type	Resolution	Image Size (bytes)	Data size (bytes)	After encryption Size (bytes)	Encrypted data with key (bytes)	Stego image size (bytes)	Stego image resolution	Data retrieved from stegoimage (After decription) (bytes)
1	BMP	128x128	32,848	3,247	2,844	3,555	36,414	128x128	3,250
2	BMP	128x128	32,848	9,741	8,524	10,655	43,514	128x128	9,741
3	BMP	128x128	32,848	29,158	25,516	31,895	64,754	128x128	29,161
4	BMP	128x128	32,848	29,162	25,516	31,900	64,759	128x128	29,165
5	JPEG	I60x120	4,608	3,247	2,844	3,555	8,173	160x120	3,250
6	JPEG	I60x120	4,608	3,248	2,844	3,555	8,173	160x120	3,250
7	JPEG	640x480	47,821	3,247	2,844	3,555	51.387	640x480	3,250
8	JPEG	640x480	47,821	9,747	8,532	10,665	58,497	640x480	9.750

Discussions

The main objective was to evaluate the performance of this algorithm in terms of data size and authentication and security. The results showed in the algorithm with dynamic key were very effective in complexity and security. We have designed a special algorithm to embed the encrypted message into the cover image and assigned a special signature to identify and to locate the information of the message from the stegano image. We have designed special algorithm for decoding process. While decoding the algorithm can identify the data with respect to the signature which is provided in the stegano image. The identified data is authenticated and decrypted. In this system the designed tool deals with providing easy and secure information. The data is encrypted with key and embedded with an Image which is ready to send through communication channels.

REFERENCES

[1] Petitcolas, F. A. P., R. J. Anderson, and M. G. Kuhn. "Information Hiding-A Survey." Proceedings of the IEEE, Special issue on protection of multimedia content, 87(7):1062-1078, July 1999.

[2] Anderson, R., and F. Petitcolas. "On the Limits of Steganography." University of Cambridge, Computer Laboratory: Cambridge, UK. September 1997. Published in IEEE Journal on Special Areas in Communications, v 16 no 4: 463-473. (May 98).

[3] Johnson, N. F. and S. Jajodia. Steganalysis of Images Created Using Current Steganography Software. Lecture Notes in Computer Science. Springer-Verlag. Vol. 1525. 1998. http://www.jjtc.com/ihws98/jjgmu.html.

[4] Ravi Kumar, B. Murti, P.R.K, Dr., "Data Encryption and Decryption process Using Bit Shifting and Stuffing (BSS) Methodology" International Journal on Computer Science and Engineering (IJCSE) Vol. 3 No. 7, pp. 2818-2827 July2011.

[5] Ravi Kumar, B. Murti, P.R.K, Dr., Hemanth Kumar, B. "An Authenticated Bit Shifting and Stuffing (BSS) Methodology for Data Security" Computer Engineering and Intelligent Systems Vol. 2, No. 3, pp. 94-104, 2011.

[6] Johnson, N. F. and Jajodia, S. Exploring Steganography: Seeing the unseen. IEEE Computer, Vol. 31, No. 2: 26-34. February 1998.

[7] Marvel. L.M., Boncelet Jr., C.G. & C. Retter, "Spread Spectrum Steganography", IEEE Transactions on image processing, 8:08, 1999.

[8] Johnson, N.and Jajodia, S. "Steganalysis of images created using current steganography software," Lecture Notes in Computer Science, Vol. 1525, pp. 273-289, 1998.

[9] Sahoo, G., Tiwari, R. K., "Designing an Embedded Algorithm for Data Hiding using Steganographic Technique by File ybridization", IJCSNS, Vol. 8, No. 1, pp. 228-233, January 2008.

[10] Ravi Kumar, B. Murti, P.R.K, Dr., Hemanth Kumar, B. "An Authenticated Bit Shifting And Stuffing (Bss) Methodology For Data Security Using Steganography " Part III Lnicst 86 proceedings, P.423