Image Based Steganography Using LSB Insertion Technique

M. S. Sutaone
Assistant Professor in E &T/C Dept
PIET's College of Engineering, Pune
Email : mssutaone@extc.coep.org.in
Phone : 9823796788

M.V. Khandare
PIET's College of Engineering, Pune
Email : meghamw@rediffmail.com
Phone : 9881404121

Abstract: Steganography is the art and science of hiding the existence of communication. The techniques used in steganography make it difficult to detect that there is a hidden message inside an Image file. This way we not only hide the message itself, but also the fact that we are sending the message.

In this paper a steganography system is designed for encoding and decoding a secret file embedded into an image file using random LSB insertion method in which the secret data are spread out among the image data in a seemingly random manner. This can be achieved using a secret key.

The key used to generate pseudorandom numbers, which will identify where, and in what order the hidden message is laid out. The advantage of this method is that it incorporates some cryptography in that diffusion is applied to the secret message.

1. Introduction:

Ensuring data security is a big challenge for computer users. Businessmen, Professionals, Military and home users all have some important data that they want to secure from others[3]. There are number of ways for securing data. Steganography is one of them.

Cryptography and steganography are often lumped together even though they are very unalike, yet complementary, technologies with different purposes. Cryptography is a technique to scramble confidential information to make it "unreadable." It is already commonly used in Internet communications. Encrypting a message can hide the content of the secret message effectively, but it cannot hide its presence. The location of the secret message is very obvious. This is the reason why an encrypted message can easily be targeted by attackers.

Steganography is the art and science of communicating in a manner such that an observer would not be able to detect that communication is occurring [7]. Steganography is useful for hiding messages for transmission. Classically this is achieved by secretly embedding the communication within another unsuspicious communication.

There are various techniques to achieve steganography like Least Significant Bit Insertion, Masking & Filtering and Algorithms & Transformations.[4] Each of these techniques can be applied, with varying degrees of success, to different image files. Least Significant Bit Insertion is a common, simple approach to embedding information in a cover file.

Section 1 gives the introduction. While Section 2 deals with representation of image types. The basics of LSB embedding and the analysis based on LSB are explained in section 3. Algorithm and implementation details are given in section 4. Section 5 gives the experimental work and results.

2. Representation of image types:

In a computer, images are represented as array of numbers that represent intensities of the three colors R(ed) G(reen) and B(lue), where a value for each of the three colors describes a pixel. Through varying the intensity of the RGB values, a finite set of colors spanning the full visible spectrum can be created. In an 8-bit image, there can be 2^8 =256 colors and in a24-bit bitmap, there can be 2^{24} = 16.7million colors. Large images are most desirable for steganography because they have the most space to hide data in .The best quality hidden message is normally produced using a 24-bit bitmap as a cover image.

A 24-bit bitmap is a bitmap header,

10010101 00001101 11001001 10010110

00001111 11001011 10011111 00010000

followed by the pixels' data. Each pixel is represented by three bytes, representing the red, green and blue color values for that pixel. The higher the number, the more intense that color is for that pixel. The cons to large images are that they are cumbersome to both transfer and upload, while running a larger chance of drawing an "attacker's" attention due to their uncommon size. As a result, compression is often used [4].

In the implementation however, one should be aware of a particular detail. In 24-bit bitmaps, the number of bytes per row is always end-padded with zeros to be a multiple of four. Although initially one may think to use these extra bytes to store hide additional information that would be unwise. Since these bytes are supposed to contain zeros, any alteration would be easily detectable. Thus, in order for the image to remain inconspicuous, only the LSBs of the actual pixel data should be altered.

3. Basic of LSB Embedding:

The concept of LSB Embedding is simple. It exploits the fact that the level of precision in many image formats is far greater than that perceivable by average human vision. Therefore, an altered image with slight variations in its colors will be indistinguishable from the original by a human being, just by looking at it. By using the least significant bits of the pixels' color data to store the hidden message, the image itself is seemed unaltered [4].

As a simple example of least significant bit substitution, imagine "hiding" the character 'G' across the following eight bytes of a carrier file (the least significant bits are underlined):

A 'G' is represented in the American Standard Code for Information Interchange (ASCII) as the binary string 01000111. These eight bits can be written" to the least significant bit of each of the eight carrier bytes as follows:

1001010*0* 00001101 1100100*0* 10010110

0000111*0* 1100101*1* 1001111*1* 0001000*1*

In the sample above, only half of the least significant bits were actually changed (shown above in italics). This makes some sense when one set of zeros and ones are being substituted with another set of zeros and ones.

3.1. Analysis Based on LSB:

"Another variation would be random LSB, in which the secret data are spread out among the image data in a seemingly random manner. This can be achieved if both the sender and receiver share a secret key. They can use this key to generate pseudorandom numbers, which will identify where, and in what order the hidden message is laid out. The advantage of this method is that it incorporates some cryptography in that diffusion is applied to the secret message. However, it goes beyond just making it difficult for an attacker knows that there is a secret message to figure out the message. It also makes it harder to determine that there was a secret message in the first place. The reason is because the randomness makes the embedded message seem more like noise statistically than in the sequential method".

Generally this involves changing redundant bits within the cover object. However, there are slight variations on this theme [2].

4. Algorithm & Implementation details:

- Open the files
- Open original Bmp/ Tiff/ Jpeg file (cover file) in read mode.
- Open the Text / Word / Excel (carrier file) which we want to hide in read mode.
- Open the output file called "stego object" in write mode. This file contains stego data.
- Read header of cover file and write it into output file.
- Copy reserved 512 bytes of user's permission into output file.
- Read each character from carrier file and read a character from cover file and Ex-OR each bit of character of cover file whose position is indicated by the digit of key (Digit value 0-7) and write the resultant character into output file.
- At the end of carrier file close all the files (cover, carrier and output files).

For example hide a character 'A' (carrier file) into cover file.

Bit format of letter 'A' → ASCII 65

```
(Binary)        0 1 0 0 0 0 0 1
Key value       5 2 4 1 3 6 4 0
```

Each bit of character 'A' is hidden at key position of byte of cover file. So for hiding 8 bits of character 'A' requires 8 bytes of cover file. As shown in the example below:

Cover file format is

```
   5                 2                 4
00001001          10101111          00111011
       1                3                6
11010101          00001111          01101011
     4                  0
00010101          01101010
```

After hiding the bits of character 'A' contents into cover file

```
00001001          10101111          00101011
11010101          00000111          00101011
00000101          01101011
```

Hiding Process (Sender)

- Initially Sender and Receiver communicate with each other and decide the " Stego Key " and "Password" used for hiding and retrieving the secret data.

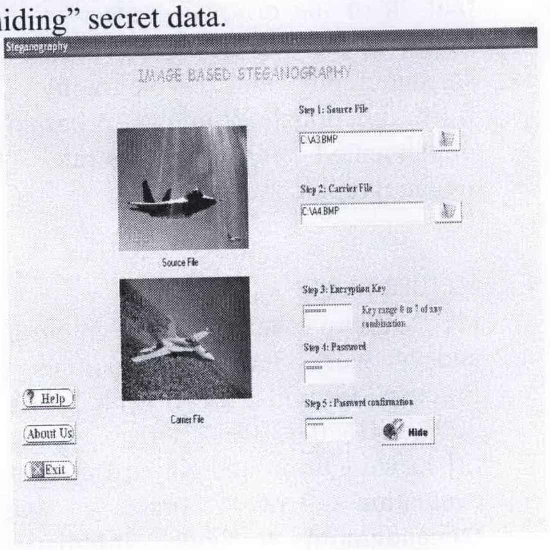
- Sender selects the secret file i.e. the data to be hidden.
- Then Sender selects cover file which is used for hiding the secret data.
- With the help of key, the secret data embedded into cover file and then applies password.
- After hiding process sender gets the "Stego Object " which is sent to the receiver.

Unhiding Process (Receiver):

- Receiver selects the Stego Object.
- Applies key and password, to retrieve data, which was communicated between sender and receiver initially.
- After retrieving process receiver gets the secret data.

5. Experimental Work & results:

It may be helpful to look at some screen shots of the program at various points in its progress. This is the main form of "Image Based the Steganography Using LSB" software. It is used for both processes "hiding and unhiding" secret data.



Step 1: Source File
C:\A3.BMP

Step 2: Carrier File
C:\A4.BMP

Step 3: Encryption Key
****
Key range 0 to 7 of any combination

Step 4: Password
******

Step 5 : Password confirmation
******

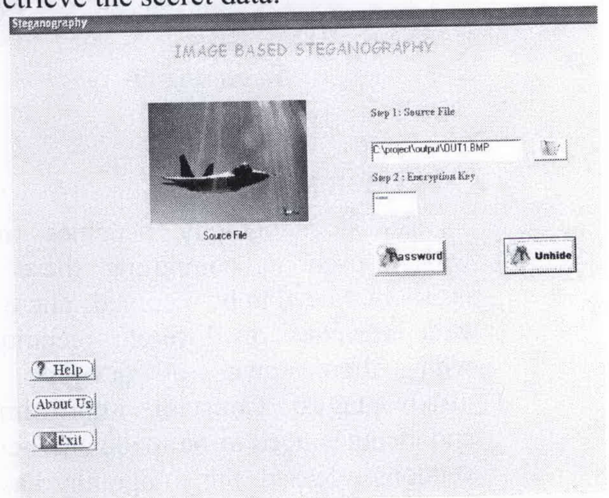
Encode Form

Hiding Process at Sender Side:

- There is folder button, which is used for selecting the source file as well as you can see the properties of the selected file.
- There is another folder button, when user clicks on this folder button the "Search" form is displayed where user can search cover file and user can see the properties.
- User can see the selected cover file in the frame.
- Then user fills the key, which is between 0 - 7 digits (length=8 bit) and fill the password which maximum length is 6 characters.
- By clicking hiding button hiding process done.

Unhiding process at Receiver Side:

- For selecting the embedded file user click on the folder button.
- Then user gives the required key and password, which is use for checking that receiver, is authenticated person or not.
- After the confirmation of receiver the button (unhide) for retrieving is visible. Then users can retrieve the secret data.



Step 1: Source File
C:\project\output\OUT1.BMP

Step 2 : Encryption Key

Decode Form

Result:
Source File – data.txt (278 KB)
Original Cover File – plane.bmp (596 KB)
Key used – 12231111
Password – omshanti
Stego Object – 596 KB (Same Size)

Original cover file

Stego object

Conclusion:

As steganography becomes more widely used in computing there are issues that need to be resolved. There are wide varieties of different techniques with their own advantages and disadvantages. Constant improvement and changes need to be made and newer versions released. Steganography has its place in security. It is not intended to replace cryptography but supplement it. Hiding a message with steganography methods reduces the chance of a message being detected.

There are infinite numbers of steganography applications. This project explores a tiny fraction of the art of steganography. It goes well beyond simply embedding text in an image. Steganography does not only pertain to digital images but also to other media (files such as voice, other text and binaries; other media such as communication channels, etc).

Random steganography using LSB with key gives us more security than simple LSB method, where it is difficult to identify the hidden data in the stego image at specific location.

Simple steganography using LSB with more than one bit used for the hidden data gives us more space to store data but here cover image will lose its visual appearance.

In short, the project is successful in that all of the original goals were met. Much is learned about the science of steganography and cryptography. A fully functional Windows program is implemented to demonstrate the steganographic system.

References:

[1] William Stallings, "Cryptography and Network Security, Principles and Practice" Edition 3rd. PRENTICE HALL 2003, ISBN 0-13-091429-0
[2] Kevin Curran and Karen Bailey "An evaluation of Image based Steganography methods" International Journal of Digital Evidence Fall 2003, Vol-2, Issue-2
[3] Dr. K. Duraiswamy and R.Umarani "Security through obscurity"
[4] Neil F. Johnson and Sushil Jajodia, "Exploring Steganography: Seeing the

unseen" IEEE transaction on Computer Practices.1998.
[5] Neil F. Johnson "Introduction to Information Hiding: Steganography, Watermarking – Attacks and Countermeasures "SANSFIRE, conference, Boston USA 2002
[6] StegoArchive
http://www.stegoarchive.com/
[7] A Detailed Look at Steganographic Techniques and their Use in an Open-System Environment
http://www.sans.org/rr/whitepapers/covert/677.php