Hidden Messages in JPEG Image Using Steganography

Mafaz Mohsin Khalil **Sadoon Hussin Abdullha**
Department of Biology / College of Science
University of Mosul

Received **Accepted**
29 / 04 / 2009 **06 / 10 / 2009**

الخلاصة

علم الإخفاء هو عملية إخفاء أحد أوساط الاتصال (نص، صوت، صور) ضمن الآخر. نلاحظ في السنين القليلة الماضية ظهور تقنيات لعلم الإخفاء وتحليل الإخفاء موثقة في محاضرات. تستخدم الطرق الحديثة لعلم الإخفاء صفات الوسط نفسه لنقل الرسالة . طمر الرسائل رقميا في وسط آخر مثل النصوص المجردة، النصوص الفائقة، الصوت والفيديو، الصور الثابتة، الشبكات المرورية.

تلقي هذه الدراسة الضوء على بعض المفاهيم العامة والأفكار التي تطبق في مجال إخفاء الصور وتحليلها . كذلك نستعرض ونناقش الأفكار العامة حول امنية وسعة نظام الإخفاء الصوري. الهدف من هذا البحث هو ت حديد قدرات نظام الإخفاء في مجال صور الـ JPEG باستخدام صفات معاملات DCT ومن ثم تحليل النتائج . و نوجز أيضا بعض تقنيات علم الإخفاء.

Abstract

Steganography is the process of hiding one medium of communication (text, sound or image) within another.

In the last few years, we have seen many new and powerful steganography and steganalysis techniques reported in the literature. The modern Methods of Steganography use the properties of the media itself to convey a message. Digitally embedding messages in other media, such as: Plain Text, Hypertext, Audio / Video, Still Imagery, Network Traffic.

In the following tutorial we go over some general concepts and ideas that apply to image steganography and steganalysis. We review and discuss the notions of steganographic security and capacity. The goal of

this paper is to determine the steganographic capacity of JPEG images by using DCT coefficients properties and static analysis for them. We illustrate also some techniques for image Steganography.

Introduction

Steganography is the art and science of hiding communication; a steganographic system thus embeds hidden content in unremarkable cover media so as not to arouse an eavesdropper's suspicion. In the past, people used hidden tattoos or invisible ink to convey steganographic content. Today, computer and network technologies provide easy-to-use communication channels for steganography.[2][5]

The word Steganography comes from the Greek steganos (covered or secret) and graphy (writing or drawing) and means, literally, covered writing. We will look at history of Steganography through to current day uses and advancements in the area, explaining how the digital age has seen the possible rebirth of Steganography and Steganlysis, the process of finding the hidden information.[4][8]

The goal of Steganography is to avoid drawing suspicion to the transmission of a hidden message. If suspicion is raised, then this goal is defeated.[7]

History of Steganography

The first recorded use of Steganography is from the Histories of Herodotus, where in ancient Greece text was written on wax covered tablets. Herodotus describes how Demeratus wanted to warn Sparta on an imminent invasion from Xerxes. In order to hide the message he scraped wax off a tablet and wrote a message. The tablet was then covered with wax again. Upon inspection by enemy soldier the tablets appeared blank and were allowed to pass. Other ancient methods include tattooing messages on a courier's head, and allowing their hair to grow, thus hiding the message and allowing the courier to deliver their message unhindered (although obviously their hair had to be removed again upon deliverance). [5]

From the medieval period through to the renaissance many complex ciphers were being developed and used so also was Steganography.

By the 1940's Steganography was called upon again to hide secret messages. World War Two is better known for the birth of hardcore encoding (E.g. the German Enigma) and the computer to crack this code.
Other forms include Great Britain's S.O.E. (Special Operations Executive) passing messages to agents in occupied Europe written in invisible inks, these messages could appear as simple blank pieces of paper or another letter upon inspection but could contain vital

communication written between the lines, only made visible in a given solution.[5][8]

Steganography Terms

- Cover-Medium – The medium in which information is to be hidden. Also sometimes called "cover-image/data/etc."
- Stego-Medium: A medium within which information is hidden.
- Message: The data to be hidden or extracted.
- Redundant Bits: Bits of data in a cover-medium that can be modified without compromising that medium's integrity

The basics of embedding

Three different aspects in information-hiding systems contend with each other: capacity, security, and robustness. Capacity refers to the amount of information that can be hidden in the cover medium, security to an eavesdropper's inability to detect hidden information, and robustness to the amount of modification the stego medium can withstand before an adversary can destroy hidden information.

Information hiding generally relates to both watermarking and steganography. A watermarking system's primary goal is to achieve a high level of robustness that is, it should be impossible to remove a watermark without degrading the data object's quality. Steganography, on the other hand, strives for high security and capacity, which often entails that the hidden information is fragile. Even trivial modifications to the stego medium can destroy it.

Essentially, steganographic communication senders and receivers agree on a steganographic system and a shared secret key that determines how a message is encoded in the cover medium. To send a hidden message, for example, Alice creates a new image with a digital camera. Alice supplies the steganographic system with her shared secret and her message. [10]

Steganographic system uses the shared secret to determine how the hidden message should be encoded in the redundant bits. The result is a stego image that Alice sends to Bob. When Bob receives the image, he uses the shared secret and the agreed on steganographic system to retrieve the hidden message. Figure1 shows an overview of the encoding step; as mentioned earlier, statistical analysis can reveal the presence of hidden content.[8]

In general, the information hiding process consists of the following steps:

1. Identification of redundant bits in a cover medium. Redundant bits are those bits that can be modified without degrading the quality of the cover medium.

2. Selection of a subset of the redundant bits to be replaced with data from a secret message. The stego medium is created by replacing the selected redundant bits with message bits.

The modification of redundant bits can change the statistical properties of the cover medium. As a result, statistical analysis may reveal the hidden content.

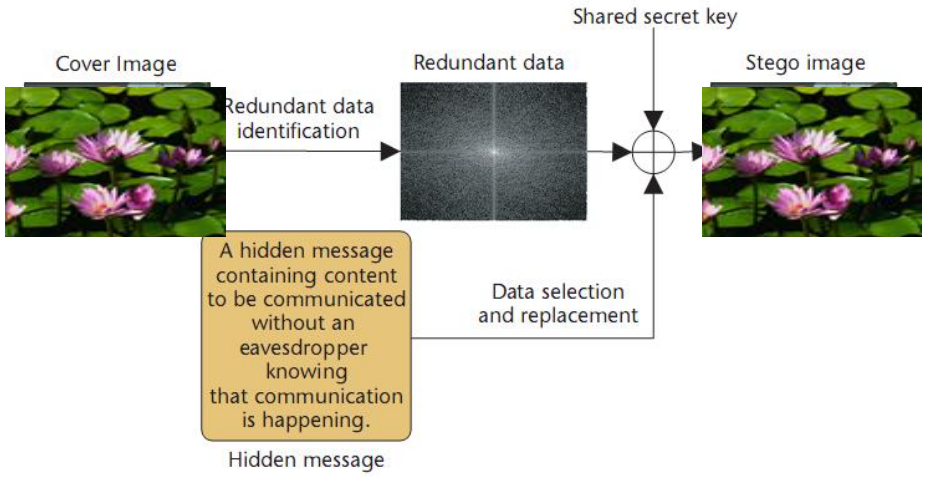


Figure 1. Modern steganographic communication. The encoding step of a steganographic system identifies redundant bits and then replaces a subset of them with data from a secret message.

Techniques for Image Steganography

Given the proliferation of digital images, and given the high degree of redundancy present in a digital representation of an image (despite compression), there has been an increased interest in using digital images as cover-objects for the purpose of steganography. Therefore we have limited our discussion to the case of images for the rest of this tutorial. We should also note that there have been much more work on embedding techniques which make use of the transform domain or more specifically JPEG images due to their wide popularity. Thus to an attacker the fact that an image other that of JPEG format is being transferred between two entities could hint of suspicious activity.

There have been a number of image steganography algorithm proposed, these algorithm could be categorized in a number of ways:

- Spatial or Transform, depending on redundancies used from either domain for the embedding process.
- Model based or ad-hoc, if the algorithm models statistical properties before embedding and preserves them, or otherwise.
- Active or Passive Warden, based on whether the design of embedder-detector pair takes into account the presence of an active attacker.

Spatial Domain Embedding

The best widely known steganography algorithm is based on modifying the least significant bit layer of images, hence known as the LSB technique. This technique makes use of the fact that the least significant bits in an image could be thought of random noise and changes to them would not have any effect on the image. This is evident by looking at Figure 2. Although the image seems unchanged visually after the LSBs are modified, the statistical properties of the image changes significantly. We will discuss in the next section of this tutorial how these statistical changes could be used to detect stego images created using the LSB method.

In the LSB technique, the LSB of the pixels is replaced by the message to be sent. The message bits are permuted before embedding, this has the effect of distributing the bits evenly, thus on average only half of the LSB's will be modified. Popular steganographic tools based on LSB embedding, vary in their approach for hiding information. Some algorithms change LSB of pixels visited in a random walk, others modify pixels in certain areas of images, or instead of just changing the last bit they increment or decrement the pixel value.[11]

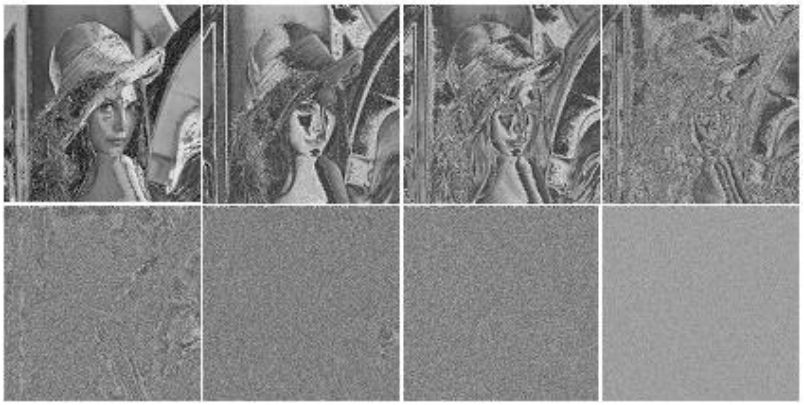


Figure 2: Bitplane decomposition of image Lenna.

Transform Domain Embedding

Another category for embedding techniques for which a number of algorithms have been proposed is the transform domain embedding category. Most of the work in this category has been concentrated on making use of redundancies in the DCT (discrete cosine transform) domain, which is used in JPEG compression. But there has been other algorithms which make use of other transform domains such as the frequency domain.[6]

Embedding in DCT domain is simply done by altering the DCT coefficients, for example by changing the least significant bit of each coefficient. One of the constraints of embedding in DCT domain is that

many of the 64 coefficients are equal to zero, and changing two many zeros to non-zeros values will have an effect on the compression rate. That is why the number of bit one could embed in DCT domain, is less that the number of bits one could embed by the LSB method. Also the embedding capacity becomes dependent on the image type used in the case of DCT embedding, since depending on the texture of image the number of non-zero DCT coefficients will vary.

Although changing the DCT coefficients will cause unnoticeable visual artifices, they do cause detectable statistical changes. In the next section, we will discuss techniques that exploit these statistical anomalies for steganalysis. In order to minimize statistical artifacts left after the embedding process, different methods for altering the DCT coefficients have been proposed, we will discuss two of the more interesting of these methods, namely the JSTEG and Outguess algorithms.[6]

JPEG FORMAT

The JPEG compression method was developed by the Joint Photographic Expert Group with the aim of compressing images with little or no noticeable degradation in quality. The process of compression consists of four steps – initial input, Discrete Cosine Transformation, quantization and encoding.

JPEG COMPRESSION PROCESS

The initial image data (usually an uncompressed bitmap image) is cut into groups of 8x8 pixels. The resulting 8x8 matrixes containing the image data are fed into the DCT algorithm. The purpose of the Discrete Cosine Transformation (DCT) is to represent the image data in a different domain (the brightness values in the 8x8 matrix are converted into the frequency domain). After the DCT step, each 8x8 matrix consists of 64 DCT coefficients. The JPEG compression has quantization tables that are calculated based on the image quality specified by the user. Each DCT coefficient is then divided by its corresponding quantization constant and rounded off to the nearest integer. This process has the effect of eliminating smaller and unimportant coefficients and causing larger coefficients to lose unnecessary precision. This step is the reason why JPEG is known as a lossy file format. The Huffman encoding scheme is commonly used in the encoding step to further compress the image. This process utilizes the Huffman tree (a weighted, binary tree) to store the quantized values from the previous step.[3]

Figure 3. JPEG Compression Process

Information Hiding in JPEG Images

JPEG images are commonly used on Internetweb sites. This section briefly explains the JPEG format and how it can be used for information hiding.

The JPEG image format uses a discrete cosinetransform (DCT) to transform successive 8×8-pixel blocks of the image into 64 DCT coefficients each. The least-significant bits of the quantized DCT coefficients are used as redundant bits into which the hidden message is embedded.[3]

In some image formats, e.g. GIF, the visual structure of an image exists to some degree in all bit-layers of the image. Steganographic systems that modify least significant bits of these image formats are often susceptible to visual attacks.

This is not true for the JPEG format. The modification of a single DCT coefficient affects all 64 image pixels. For that reason, there are no known visual attacks against the JPEG image format.

Discrete cosine transform

For each color component, the JPEG image format use a discrete cosine transform (DCT) to transform successive 8 × 8 pixel blocks of the image into 64 DCT coefficients each. The DCT coefficients F(u, v) of an 8 × 8 block of image pixels f(x, y) are given by

$$F(u,v) = \frac{1}{4}C(u)C(v)\left[\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y) * \cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}\right],$$

where C(x) = 1/√2 when x equal 0 and C(x) = 1 otherwise. [3][8]

Other Images Steganographic Algorithm

Four image steganographic techniques have been studied for the purpose of this research. They are F5, Jsteg, JPHide and Outguess. All these algorithms utilize Transform Domain tools and work only on JPEG images. The information is always hidden in the LSB of the DCT coefficients, the exception being the F5 algorithm. The hiding of the information is done between the quantization and encoding steps of the JPEG compression.[1][10]

1. F5

The F5 algorithm was developed by Andreas Westfield with the aim of having better capacity (maximal safe bit-rate) and embedding efficiency (number of bits embedded per change), while being resistant against visual and statistical attacks. As mentioned above, the F5 algorithm does not manipulate the LSB of the DCT coefficients when hiding information. Instead, it decrements the coefficients' absolute value when the LSB does not match the bit to be embedded. To preserve statistical properties of the JPEG file, the F5 algorithm maps negative DCT coefficients to inverted steganographic values. That is to say, for

negative coefficients, even numbers represent a steganographic one while odd numbers represent a steganographic zero, and vice versa for positive coefficients.[1]

2. JSTEG

Jsteg is a steganographic algorithm developed by Derek Upham. Jsteg is designed to be resistant against visual attacks while offering a better-than-average capacity of 12.8%. While the F5 algorithm decrements the absolute values of the DCT coefficients, Jsteg replaces the LSB of the coefficients with the information bits. The algorithm skips all DCT coefficients with values of zero and one.[1]

JPEG-JSteg algorithm is a typical steganographic algorithm using JPEG file as carrier-image. JSteg is first proposed by D. Upham. In this method, after quantization of DCT coefficients, JPEG-JSteg replaces the least significant bits (LSB) of the quantized DCT coefficients by the secret message bits. The embedding mechanism skips all coefficients with the values 0 or 1. The steganographic tool Jsteg embeds messages in lossy compressed JPEG files. It has a high capacity—e.g., 12 % of the steganogram's size—and, it is immune against visual attacks. However, a statistical attack discovers changes made by Jsteg . [3]

The JSteg algorithm. As it runs, the algorithm sequentially replaces the least-significant bit of discrete cosine transform (DCT) coefficients with message data. It does not require a shared secret.[8]

Input: message, cover image
Output: stego image
while data left to embed **do**
    get next DCT coefficient from cover image
    if DCT ≠ 0 and DCT ≠ 1 **then**
      get next LSB from message
      replace DCT LSB with message LSB
    end if
    insert DCT into stego image
end while

Figure 4 shows three images The uncompressed original image has a size of almost 45.7 Kb, while the secret has 9.95 KB, and the result image shown are about 46.3 Mb. The one to the left is unmodified and the one to the right contains the secret image, After compression, it is not possible for the human eye to find a visual difference between the two of them.

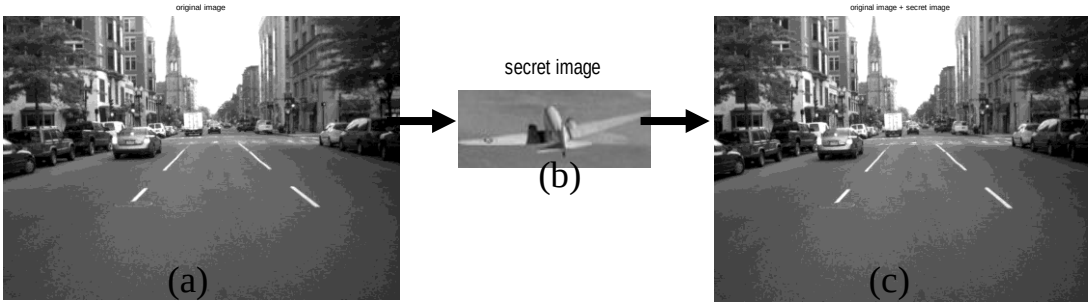


Figure 4: Embedded information in a JPEG.(a) The unmodified original picture; (b) The Secret Image; (c)The result with The Secret Image embedded in it.

Statistical Analysis

Statistical tests can reveal if an image has been modified by steganography by testing whether an image's statistical properties deviate from a norm. Some tests are independent of the data format and just measure the entropy of the redundant data.[1]

Instead of measuring the color frequencies, we analyze the frequency of the DCT coefficients. Figure 5 shows an example where embedding a hidden messages causes noticeable differences to the DCT coefficient histogram.[6]

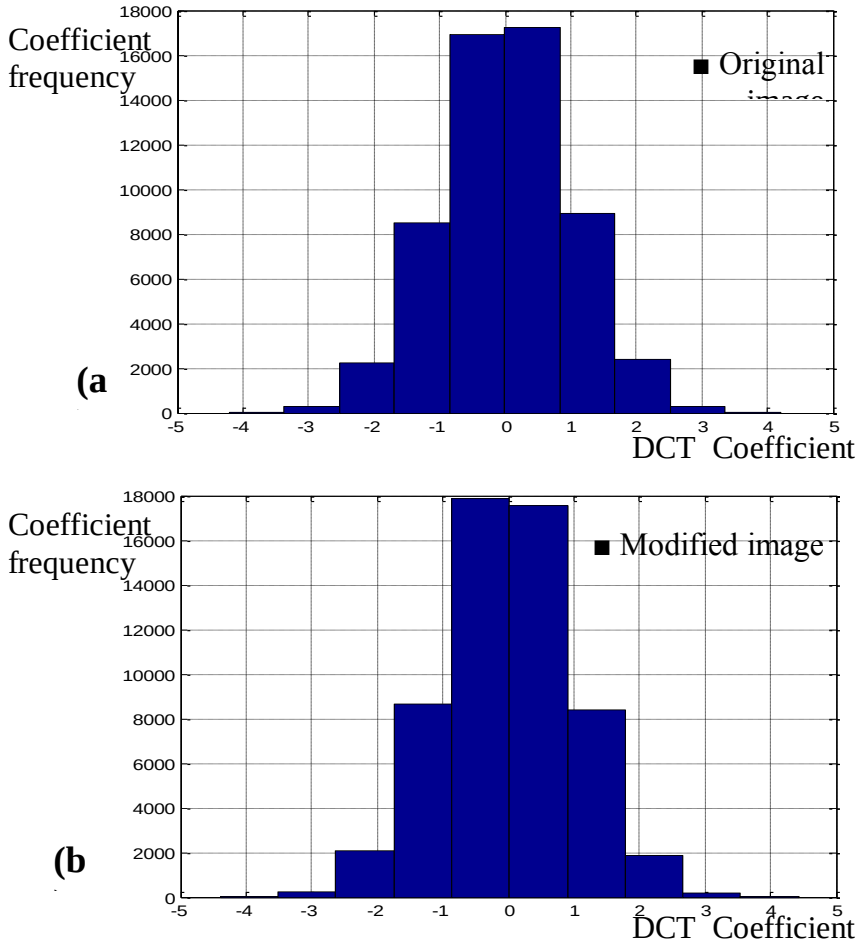


Figure 5. Frequency histograms. Sequential changes to the (a) Original and (b) modified image's least-sequential bit of discrete cosine transform coefficients tend to equalize the frequency of adjacent DCT coefficients in the histograms.

We use a $\chi 2$-test to determine whether an image shows distortion from embedding hidden data. Because the test uses only the stego medium, the expected distribution y^*_i for the $\chi 2$-test has to be computed

from the image. Let n_i be the frequency of DCT coefficients in the image. We assume that an image with hidden data embedded has similar frequency for adjacent DCT coefficients. As a result, we can take the arithmetic mean, [6] [9]

$$y_i^* = \frac{n_{2i} + n_{2i+1}}{2}$$

to determine the expected distribution. The expected distribution is compared against the observed distribution

$$y_i = n_{2i}.$$

The $\chi 2$ value for the difference between the distributions is given as

$$\chi^2 = \sum_{i=1}^{v+1} \frac{(y_i - y_i^*)}{y_i^*},$$

Where ν are the degrees of freedom, that is, the number of different categories in the histogram minus one.

3. JPHIDE

The JPHide algorithm, developed by Allan Lantham, is similar to that of Jsteg, in that it embeds the message bits in the LSB of the DCT coefficients. Unlike Jsteg, which selects the DCT coefficients continuously from the start of the image, JPHide uses a Blowfish algorithm to generate a stream of pseudo-random control bits that are used to determine which coefficients are to be used to hide the information. The result is that the information is randomly scattered around and resembles noise that is inherent to the image. This makes JPHide more resistant against visual attacks.[1]

4. OUTGUESS

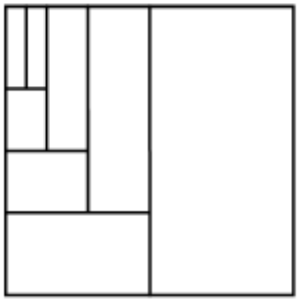
Outguess is a steganographic algorithm developed by Niels Provos. It encrypts the information before hiding and uses a pseudo-random number generator to choose the DCT coefficients in which to hide the information. What is unique to Outguess is that uses half of the available DCT coefficients to hide the information and manipulates the other half to preserve the first order distribution of the DCT coefficients.[1]

OutGuess improves the encoding step by using a pseudo-random number generator to select DCT coefficients at random. The least-significant bit of a selected DCT coefficient is replaced with encrypted message data (see Algorithm blow). The OutGuess algorithm. As it runs, the algorithm replaces the least-significant bit of pseudo-randomly selected discrete cosine transform (DCT) coefficients with message data. [7]

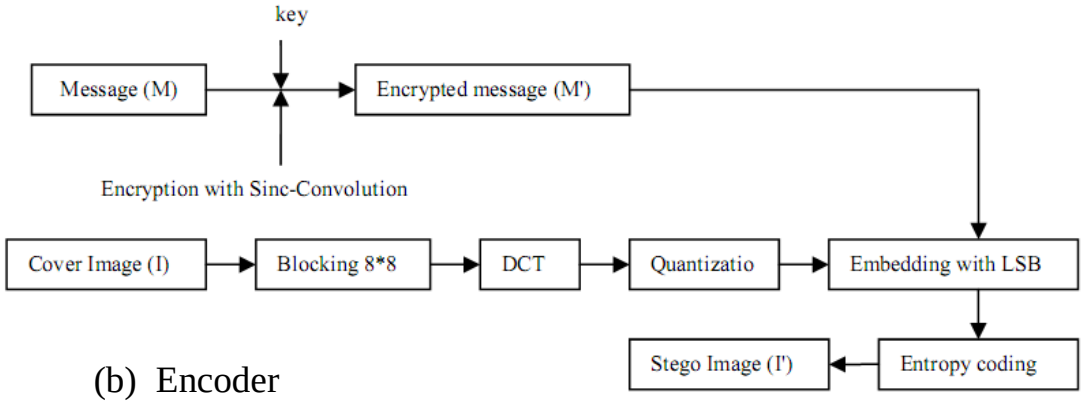
Input: message, shared secret, cover image
Output: stego image
initialize PRNG with shared secret
while data left to embed **do**
    get pseudo-random DCT coefficient from cover image
    if DCT ≠ 0 and DCT ≠ 1 **then**
        get next LSB from message
        replace DCT LSB with message LSB
    end if
    insert DCT into stego image
end while

Encryption Key

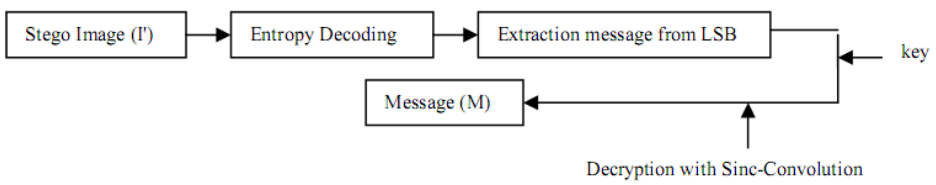
For more security, images are divided into several parts as in Figure 6 and each section is encrypted with a function. "Header" that includes the important information of the image has been divided into smaller sections.[2]



(a) Encryption Key parts



(b) Encoder



(c) Decoder

Figure 6: Encoder and Decoder.

The key used for encryption is selected to be 128 bits long and is composed of two parts:

1. Encryption information, including the number of image parts, the number of each function and its parameters.
2. Security information including the secret information.

Elements of the encryption key:

1. The first 8 bits to determine the number of image parts (at most 8 bits and at least 1 bit).
2. The second 8 to 64 bits depends on the number of image parts and determines the function number through which each part of the image is encrypted.
3. The third 16 to 48 bits determine the parameters of the encryption function (at most 6 parameters and at least 2 parameters).
4. The remaining bits for the key security. After encryption of the desired message in a form of an image, we use JSTEG steganography algorithm to hide it in a cover (host) image. The following section, describes the JSTEG algorithm.

Implementation Results

This method is secure against visual attacks, but a statistical attack can discover changes which are created by JSteg algorithm.

The kind of selected image will be more effective in getting better results after steganography. Actually this method will be broken because of the changes created in histogram of DCT coefficients, therefore by using of encryption algorithm, errors of JSteg will be eliminated and encrypted data will be hidden. "Cameraman" image encrypted with a special key is indicated in Figure. 7.[2]

encrypted image of cameraman with 170484126441922911562 1

original image of cameraman

(a): Original image and encrypted image of "cameraman"

Decryption with function key 570484126441922911562 1

Decryption with function key 170484126441922911562 1

(b): Encryption of "cameraman" image with different functions.

Figure 7: Encryption and Decryption with function key.

CONCLUSIONS

The past few years have seen an increasing interest in using images as cover media for steganographic communication. There have been a multitude of public domain tools, albeit many being ad-hoc and naive, available for image based steganography. Given this fact, detection of covert communications that utilize images has become an important issue. In this tutorial we have reviewed some fundamental notions related to steganography and steganalysis.

Although we covered a number of security and capacity definitions, there has been no work successfully formulating the relationship between the two from the practical point of view. For example it is understood that as less information is embedded in a cover-object the more secure the system will be. But due to difficulties in statistical modelling of image features, the security versus capacity trade-o® has not been theoretically explored and quantified within an analytical framework.

We also reviewed a number of embedding algorithms starting with the earliest algorithm proposed which was the LSB technique. At some point LSB seemed to be unbreakable but as natural images were better understood and newer models were created LSB gave way to new and more powerful algorithms which try to minimize changes to image statistics. But with further improvement in understanding of the statistical regularities and redundancies of natural images, most of these algorithms have also been successfully steganalysed.

In term of steganalysis, as discussed earlier, there are two approaches, technique specific or universal steganalysis. Although finding attacks specific to an embedding method are helpful in coming up with better embedding methods, their practical usage seems to be limited. Since given an image we may not know the embedding technique being used, or even we might be unfamiliar with the embedding technique. Thus universal steganalysis techniques seem to be the real solution since they should be able to detect stego images even when a new embedding technique is being employed.

References

1) AARON S.C.TAN and CHANG Ee CHIEN "NUROP COGRESS PAPER DCT-Based Image Steganalysis and Steganography", http:// ntu.edu.sg/eee/urop/.../abstract/NUS_SoC/NUS-TanSuChiangAaron-SoC.pdf

2) Ahmad R. Naghsh-Nilchi, and Latifeh Pourmohammadbagher, "A New Approach to Steganography using Sinc-Convolution Method", http:// isis.poly.edu/~steganography/pubs/vlsi04.pdf

3) Dongdong Fu, Yun Q. Shi, Dekun Zou and Guorong Xuan, "JPEG Steganalysis Using Empirical Transition Matrix in Block DCT Domain,http://research.microsoft.com/workshops/mmsp06/.../CameraReady_327.pdf

4) Jessica Fridrich, Tomáš Pevný and Jan Kodovský, "Statistically Undetectable JPEG Steganography: Dead Ends, Challenges, and Opportunities", http://www.ws.binghamton.edu/fridrich/Research/fraction03.pdf

5) Jonathan Watkins, 15th December 2001, "Steganography - Messages Hidden in Bits", http://mms.ecs.soton.ac.uk/mms2002/papers/6.pdf

6) Max Weiss, "Principles of Steganography", http:// www.math.ucsd.Edu/~crypto/Projects/MaxWeiss/steganography.pdf

7) Mehdi Kharrazi and Husrev T. Sencar, Oct–Dec 2006," Performance study of common image steganography and steganalysis techniques", http:// isis.poly.edu/~steganography/pubs/bench.pdf

8) NIELS PROVOS and PETER HONEYMAN, 2003 IEEE SECURITY & PRIVACY, "Hide and Seek: An Introduction to Steganography", http://niels.xtdnet.nl/papers/practical.pdf

9) Niels Provos and Peter Honeyman, CITI Technical Report 01-11, "Detecting Steganographic Content on the Internet", http:// www.citi.umich.edu/techreports/reports/citi-tr-01-11.pdf

10) R. Chandramouli, "A Mathematical Approach to Steganalysis", http:// www.ece.stevens-tech.edu/~mouli/spiesteg02.pdf

11) Xiaojun Qi and KokSheik Wong, "AN ADAPTIVE DCT-BASED MOD-4 STEGANOGRAPHIC METHOD", http:// www.cs.usu.Edu/~xqi/Tenure/ICIP.Steganography.05.pdf