# Chapter 21
# Steganography and Steganalysis


**Learning Objectives**
The objectives of this chapter are to:

- Understand basic concept of steganography and steganalysis
- Explore about steganography techniques and steganalysis techniques
- Perform steganography by using steganography tool
- Perform steganalysis by using steganalysis tools

Since the mere fact that two people communicating can bring on suspicion by association, there exists extremely high utility value in keeping the communication itself hidden. It should come as little surprise that those who tend to engage in subversive activities will also utilize all of the tools available to keep their actions (and associations) private.

Steganography, literally meaning "covered writing", is an art and science of communicating information in a covert manner such that the existence of this communication is not detectable. The purpose of steganography is to hide the existence of a message in an appropriate carrier, e.g., image, audio, and video files, from a third party. Steganography can be employed in various useful applications such as copyright control of materials, enhancing robustness of image search engines, smart IDs as well as video-audio synchronization [1]. While steganography may seem to be an excellent apparatus for the exchange of sensitive information in a concealed manner, it can also be used in ways that are counter productive to our security measures, e.g., hiding records of illegal activity, financial fraud, industrial espionage, and communication among members of criminal or terrorist organizations [2].

From the view point of computer forensics, it is not only necessary for the investigators to understand the basis behind steganography and explore steganography techniques, but also it requires them to understand the means by which an



X. Lin, *Introductory Computer Forensics*,
https://doi.org/10.1007/978-3-030-00581-8_21

adversary can defeat against steganographic systems. This practice of detecting messages hidden using steganography is referred to as steganalysis. Additionally, the investigators need to recover the hidden data from the carrier. It is more challenging to uncover hidden data from the carrier than attempting to recover plaintext from ciphertexts. The later is often stored in plain sight. Files that contain hidden data are not labelled as such. It is firstly necessary to determine if files contain hidden information.

Moreover, software systems have also been developed to implement steganography and steganalysis. There are a number of tools available on the Internet for anyone to download. This makes the use of steganography much easier which may be abused for illegal activities. Therefore, the use of steganalysis is likely to increase in computer forensics in the near future. There is significant research being conducted in academic circles on steganographic and steganalytic techniques.

Due to the fact that multimedia including image, audio, and video are widely used as the main carries of steganography techniques, we explore steganography and steganalysis techniques by considering data hiding and detection in multimedia in this chapter. Specifically, we firstly describe steganography and steganalysis basis from the aspects of basic concept, methods, classifications and application. Then, we review steganography and steganalysis techniques in image, audio, and video. Also, we present the typical steganography and steganalysis tools in multimedia.

## 21.1 Steganography and Steganalysis Basis

In this section, we describe steganography and steganalysis basis to clarify the concepts, features and applications of steganography and steganalysis.

### *21.1.1 Steganography Basis*

Steganography can be simply explained as the embedding of one information source into another. It differs from cryptography, the art of secret writing, which intends to make a message unreadable by a third party but does not hide the existence of the secret communication. In steganography, the message is hidden so the third party has no knowledge of existence of the message. Sending an encrypted message gives rise to suspicion while an “invisible” message will not. Although steganography is separate and distinct from cryptography, there are many analogies between the two, and some authors categorize steganography as a form of cryptography since hidden communication is a form of secret writing.

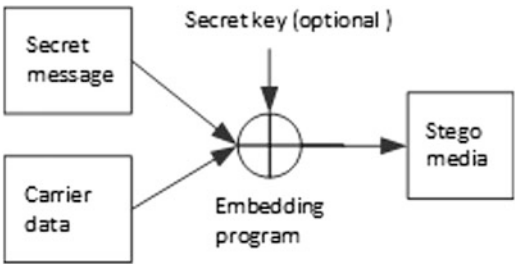



**Fig. 21.1** The steganographic embedding process [4]

Example of Encryption for "Attack at dawn":

Attack at dawn $\bigoplus$ password = 000a0c070ada00

Example of its Steganography:

Avoid The Tarts At Candy's Kitchen And The Deserts At Wilson's Neighbour.

Above are crude examples of cryptography and steganography. Here we use bold text to denote the intended hidden message. Real world examples of data hiding use more sophisticated methods and complex retrieval methods. Data is often stored in media files like images, audio and video files, also referred to as cover media or the host media/signal [3].

The embedding program should produce no obvious artifacts in the resulting stego-media, which would bring suspicion on the media. Modern steganographic techniques may combine both science (steganography and cryptography) to produce better protection of the message. It requires an additional stegonographic key, which is used for encryption of the hidden message and/or for randomization in the steganography scheme. The process to hide data can be understood as: The secret message is embedded into a second digital file called the carrier data. The result is the stego-media that is perceptually identical to the carrier, as shown in Fig. 21.1.

In this case, when the steganography fails and the message is detected, it is still secure as it is encrypted using standard or modified cryptographic techniques. Without knowing the secret key, the secret message can not be accessed.

Various features are used to characterize the strengths and weakness of the steganographic techniques [5]. The features are described as follows:

- *Invisibility or undetectibility*. Steganography is used to transmit a secret message, keeping it inside a cover medium, so invisibility of a steganographic algorithm is the first and foremost requirement. The steganographic encoding is considered failed if the adversary draws suspicion of the presence of the hidden data even though it is unable to extract the message. The embedding of the message in the cover should occur without significant degradation or loss of perceptual quality of the cover such that it can not be noticed by human eyes.
- *Hiding capacity.* The size of information can be hidden relative to the size of cover is known as hiding capacity. Larger hiding capacity allows the use of smaller cover, and thus decreasing the bandwidth required to transmit the

stego-media. Notably, hiding of more data should not affect the quality of the cover medium.

- *Robustness*. The ability of embedded data remains intact if the sego-media undergoes transformations such as addition of random noise, filtering, scaling and rotation, and so on, is defined as robustness. Robustness is critical for copyright protection watermarks because filtering is attempted to destroy any watermarks.
- *Tamper resistance*. The difficulty for a pirate to alter or forge embedded message in stego-media is referred as tamper resistance. In applications, where high robustness is demanded, requires a strong tamper resistance.

The ultimate intent of steganography is to maximize the communications bandwidth, minimize the perceptibility of the communication and ensure robustness of the embedding. There usually exist trade-offs between them. By constraining the degree of host signal degradation, a data-hiding method can operate with either high embedded data rate, or high resistance to modification, but not both. In any system, you can trade bandwidth for robustness by exploiting redundancy. The quantity of embedded data and the degree of host signal modification vary from application to application. Consequently, different techniques are employed for different applications [6].

Steganography provides some very useful and commercially important functions in the digital world, as described in the followings [7, 8].

- *Secret communication*. It can be used by intelligence agencies across the world to exchange highly confidential data in a covert manner. For example, a secret agent can hide a map of a terrorist camp in a photograph by using image steganographic software. The photograph can be posted on a public discussion board or forum. An officer from the head office can download the photograph from the forum and easily recover the hidden map.
- *Secure and invisible storage of confidential information*. Confidential information like patents or trade secrets can be securely stored in steganographic hard disk partitions. Such partitions are invisible and can only be accessed by its owner. Even the existence of such partition is unknown to others. No one can access the confidential information stored in the partition without a proper file name and associated password.
- *Digital watermarking*. In this application, the embedded data are used to place an indication of ownership in the host signal and/or to ensure the integrity of the content. It serves the same purpose as an author's signature or a company's logo. Although conceptually similar to steganography, digital watermarking usually has different technical goals. It is not necessary to hide the watermarking information. Generally, only a small amount of repetitive information is inserted into the carrier.
- *Tamper-proofing*. It is used to indicate that the host signal has been modified from its authored state. Modification to the embedded data indicates that the host signal has been changed in some way.

- *Feature location*. This is usually used in image steganography. In this application, it enables one to identify individual content features, e.g., the name of the person on the left versus the right side of an image. Typically, feature location data are not subject to intentional removal. However, it is subjected to image modification such as scaling, cropping, and tone-scale enhancement. As a result, feature location data-hiding techniques must be immune to geometrical and non-geometrical modifications of a host signal.

Unfortunately, steganography can also be used by criminals to exchange information or perform malicious actions. In the aftermath of September 11, 2001, a number of articles appeared suggesting that al Qaeda terrorists employ steganography. The threat not only exists in national security, but also in the financial and commercial markets. Information regarding money laundering, insider trading, the illegal drug trade, the distribution of child pornography and trafficking in humans can all be concealed using steganography [4]. Although it is hard to know how widespread the use of steganography is by criminals and terrorists, it is certain to draw a growing attention. Steganography may pose a hurdle for law enforcement and counterterrorism activities. The increased availability of steganographic tools introduces a new threat to the forensic investigators by hiding information in seemingly innocuous carriers. Forensic investigators have to be concerned with information that cannot be readily apparent. They must keep an eye out for subtleties that may point to hidden information. Consequently, ignoring the significance of steganography is not a good strategy [9].

### *21.1.2 Steganalysis Basis*

The ease use of abundant steganography tools and the possibility of hiding illicit information via web page images, audio, and video files have raised the concerns of law enforcement. Steganalysis is used to recover hidden information from these steganography files. While it is relatively easy to hide a secret message in a cover, the detection of an embedded message, i.e., steganalysis is challenging due to many different methods used in steganography and the evolution of the steganography algorithms. It is quite complex to detect hidden information without knowing which steganalytic technique was used or if a stego key was used. The major challenge for Steganalysts lies in that the priority of steganography is to ensure that others do not know that file exists.

Steganalysis broadly follows the way in which the steganography algorithm works. It is a fairly new practice and requires much work and refinement. Efforts have been made to develop steganalysis algorithms, which include passive and active steganalysis. Passive steganalysis simply tries to detect the presence of a message while active analysis attempts to extract the secret message itself. In some cases, steganography detection and extraction is generally sufficient if the purpose is evidence gathering related to a past crime. While during an on-going investigation of

criminal or terrorist groups destruction, detection of hidden data may not be sufficient. The steganalyst may also want to disable the hidden message so that the recipient cannot extract it, and/or alter the hidden message to send misinformation to the receiver.

Detecting steganography is based on the combinations of carrier, stego-media, embedded message, and steganography tools known by the analyst. The associated attacks are steganography–only attack, known-carrier attack, known-message attack, chosen-steganography attack, chosen-message attack, and known-steganography attack. Steganalysis techniques can be classified in a similar way as cryptanalysis methods, largely based on how much prior information is known, described as follows:

- Steganography-only attack: The steganography medium is the only item available for analysis.
- Known-carrier attack: The carrier and steganography media are both available for analysis.
- Known-message attack: The hidden message is known.
- Chosen-steganography attack: The steganography medium and algorithm are both known.
- Chosen-message attack: A known message and steganography algorithm are used to create steganography media for future analysis and comparison.
- Known-steganography attack: The carrier and steganography medium, as well as the steganography algorithm, are known.

These attacks may be applied with varying results depending upon the characteristics and availability of the steganography components. The use of steganalysis is likely to increase in computer forensics in the near future. Notably, the battle between steganography and steganalysis is never-ending. New, more sophisticated steganographic methods will require more refined approach for detection. There are significant researches being conducted in academic circles on steganographic and steganalytic techniques, which are illustrated in the following sections.

## 21.2 Steganography Techniques and Steganography Tools

The goal of steganography is to avoid drawing suspicion to the transmission of a hidden message. In other words, a good steganographic method should have acceptable statistical imperceptibility and a sufficient payload, while these two objectives are generally conflicting with each other for a given algorithm. Currently, lots of steganography tools have been explored to carry out steganography.

### *21.2.1 Steganography Techniques*

In modern steganography, numerous attempts have been made to achieve steganography. Generally, steganography techniques can vary greatly depending on the carrier media. Currently, image, audio and video files remain the easiest and most common carrier media on the Internet. Moreover, these files posses a large amount of redundant bits which can be used for steganography. As many image steganography techniques can be used in audio and video steganography, this section will focus on steganography techniques in image. We especially concentrate on two typical and popular methods: LSB (Least Significant Bit) approaches and DCT based image steganography.

#### 21.2.1.1 LSB Approaches

The LSB embedding is the most widely used technique to hide data in spatial domain in image. The basis behind LSB is to insert the secret information in the least significant bit of the pixel values. The changes resulting from the LSB insertion algorithm are not visible to the human eye. Notably, this method uses bits of each pixel thus it can be easily destroyed by compressing, filtering, or cropping the image [10]. Therefore, LSB algorithms are usually used in lossless compression format such as BMP images.

*Example 21.1 (Example of LSB) Assume the original raster data (assuming no compression) for 3 pixels (9 bytes) is:*

*(00100111 10101001 10001001) (00100110 11001001 11101101) (11001010 00100100 11001000)*

*The first bit to the left is the most significant digit and the first bit on the right is the least significant digit. Hide the letter B in the three pixels with LSB algorithm.*

*Solution. The binary value for letter B is 01000010. Inserting the binary value for B in the three pixels would result in*

*(00100111 11101000 11001001) (00100110 11001000 11101000) (11001000 00100111 11101000)*

*The underlined four bits are the actually changed bits in the 8 bytes used.*

In order to embed a larger message, information is sometimes hidden in the second and third bits or more bits of each pixel, as changes made to the second and third or more LSBs of each pixel are also not noticeable to the human eye with a well-chosen image. It means that large amount of information can be embedded per image thus the LSB algorithm has a high capacity. There is a tradeoff between steganography capacity and invisibility.

The simple algorithm described above inserts the bits of the hidden message sequentially into the cover image. As a result, it is easy to detect and extract the message. One variation of LSB insertion uses the random pixel manipulation technique by utilizing a stego key. The stego key provides a seed value for a random number generator. Using the seed value, random pixels in the image are selected for

embedding the message. Even if an adversary suspects that LSB steganography has been used, he has no idea which pixel to target without the secret key. Although inserting the message in random pixels makes it harder to detect and extract the hidden message, the steganography can still be destroyed by compression and other image manipulation such as filtering or cropping [10].

Another alternative algorithm is the LSB matching (LSBM) algorithm, which improves the undetectibility of the stego-image. It does not substitute the least significant bits in the stego-image such as in case of LSB algorithm. The LSBM adds $-1$ or $+1$ ($\pm 1$ schema) randomly to the value of the stego-image when the secret information bit does not match the LSB of the stego-image. For example, the pixel value 63 with the binary number (00111111) and a secret bit 0. The algorithm randomly adds 1 and it becomes 64 (01000000) after embedding the secret bit. Statistically, the probability of increasing and decreasing for each modified pixel is the same. Thus, it will eliminate the asymmetry artifacts produced by the LSB algorithm [11, 12]. However, Harmsen [13] finds that the center of mass (COM) of the histogram characteristic function can be exploited to detect LSBM, where the cover images contain more high-frequency component compared to its stego-image histogram. Subsequently, Mielikainen [14] proposes LSB matching revisited (LSBMR) algorithm to resist this attack.

Unlike the LSB algorithm, the LSBMR algorithm uses two pixels of the cover image as the embedding unit to convey the secret message: First pixel ($x_j$) is used to embed the secret message bit ($m_j$); The binary relationship between both pixels value $x_j$ and $x_{j+1}$ is used to embed another message bit ($m_{j+1}$). In [14], the relationship between both pixels is based on the following binary function:

$$f(x_j, x_{j+1}) = LSB\left(\left\lfloor \frac{x_j}{2} \right\rfloor + x_{j+1}\right)$$

For embedding a unit of two consecutive pixels, there are four cases for LSBMR as followings [12]. In the LSBMR algorithm, it takes a unit composes of pair of cover image pixel ($x_j$,$x_{j+1}$) and message M bits ($m_j, m_{j+1}$) as input. After embedding ($m_j, m_{j+1}$) into ($x_j$,$x_{j+1}$), the algorithm produces the stego-pixels ($y_j$, $y_{j+1}$) as output.

*Case 1*: if($m_i = LSB(x_i)$) & if($m_{i+1} \neq f(x_i, x_{i+1})$), $(y_i, y_{i+1}) = (x_i, x_{i+1} \pm 1)$

*Case 2*: if($m_i = LSB(x_i)$) & if($m_{i+1} = f(x_i, x_{i+1})$), $(y_i, y_{i+1}) = (x_i, x_{i+1})$

*Case 3*: if($m_i \neq LSB(x_i)$) & if($m_{i+1} = f(x_i - 1, x_{i+1})$), $(y_i, y_{i+1}) = (x_i - 1, x_{i+1})$

*Case 4*: if($m_i \neq LSB(x_i)$) & if($m_{i+1} \pm f(x_i - 1, x_{i+1})$), $(y_i, y_{i+1}) = (x_i + 1, x_{i+1})$

The pairs of pixels are selected randomly by using PRNG seeded with a shared stego-key. The algorithm checks if the first message bit ($m_j$) matches the LSB pixel ($x_j$) of the first cover image, then the stego pixel $y_j = x_j$ ($x_j$ remains unchanged); otherwise the stego pixel $y_{j+1} = x_{j+1}$ ($x_{j+1}$ remains unchanged). In case $m_j = LSB$ ($x_j$) and $m_{j+1}$ does not matches the binary function $f(x_j, x_{j+1})$ then $y_{j+1} = x_{j+1} \pm 1$. The algorithm either increases or decreases by one based on even- and odd- valued regions. In addition, it would not introduce the LSB approach asymmetry property.

**Fig. 21.5** The results of the quantizer

$$\begin{bmatrix} 204 & -1 & -3 & -24 & -8 & -4 & -4 & -1 \\ -8 & -3 & -2 & -14 & -7 & -4 & 0 & 0 \\ 0 & -3 & 0 & 10 & 4 & -1 & 0 & 0 \\ 0 & -11 & 0 & 10 & 4 & 0 & 0 & 0 \\ -30 & -7 & 6 & 7 & 0 & 0 & 0 & 0 \\ -7 & 2 & 12 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ -12 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

**Table 21.1** *An example of embedding letter "B" using LSBMR*

| $m_j$ | $m_{j-1}$ | $x_j$ | $x_{j+1}$ | $y_j$ | $y_{j+1}$ |
|---|---|---|---|---|---|
| *0* | *1* | *51* | *61* | *52* | *61* |
| *0* | *0* | *22* | *12* | *22* | *13 or 11* |
| *0* | *0* | *31* | *11* | *30* | *11* |
| *1* | *0* | *12* | *41* | *11* | *41* |

*Example 21.2 (Example of LSBMR) Assume that the letter "B" is required to embed as a secret data into a cover image. "B" has the binary value 01000010. Thus, 4 units of 2 consecutive pixels are selected randomly by PRNG. The selected pixels pair are (51, 61), (22, 12), (31, 11) and (12, 41).*

*Solution. The detail of embedding first two bits* $(m_j, m_{j+1}) = (0, 1)$ *of the letter "B" into the cover pixel value* $(x_j, x_{j+1}) = (51, 61)$ *is shown in Table 21.1. As the LSB (51) does not match* $m_j$ *and* $f(x_{j-1}, x_{j+1})$ *does not match* $m_{j+1}$*, case 4 is invoked and the stego-image pixels* $(y_j, y_{j+1}) = (51, 61)$ *is produced. With the same method, Stego-image pixels are provided in* Table 21.1 *after applying LSBMR algorithm.*

The LSBMR method allows an embedding of the same amount of information into the stego image as LSB matching. At the same time, the number of changed pixel values is smaller, thus it has better invisibility compared to the LSB algorithm. Additionally, the LSBMR method does not have the asymmetric property of LSB replacement method. Therefore, it is immune against steganographic attacks that utilize the asymmetric property. Moreover, it could be used for any discrete-valued cover medium, not just images. However, the LSBMR algorithm does not consider the difference between a pair of pixels, while not all pixels are suitable to be modified, as mentioned in [15]. The Human eye becomes more sensitive and may appear more suspicions in case of modifying bits in smooth area. In LSBMR, pixel pair is also selected by PRNG without considering the relationship between the message size and the content of cover image. Therefore, LSBMR may change the least significant bits of some part of the image such that it can easily be noticed when analyzing the LSB plane of the stego-image. Hence, the LSBMR algorithm is not strong against visual attacks [12].

In summary, the main advantage of LSB manipulation is that it is a quick and easy way to hide information. Its disadvantages are mainly due to the fact it is vulnerable to small changes resulting from image processing or lossy compression. Thus, LSB based steganography techniques are usually applied in BMP as well as GIF, while

their resistance to statistical counter attacks and compression are reported to be weak. Consequently, other spatial domain techniques are also explored for image steganography.

#### 21.2.1.2 DCT Based Image Steganography

DCT is an advantage transformation in image thus data can be hidden by modifying the DCT coefficient values in the frequency domain. After DCT transformation, the image has low, high and middle frequency components. The low-order DCT coefficients correspond to large features of pixels and high-order coefficients correspond to fine features. So the high-order coefficients are selected for embedding secret information. These techniques are normally applicable to JPEG images because JPEG images are stored as DCT coefficient values. As JPEG images dominate the image format, we focus on the basic logic and methods of DCT based JPEG image steganography in this chapter.

In JPEG image, blocks of $8 \times 8$ pixels are transformed into 64 DCT coefficients by using the DCT. The DCT coefficients are quantized using a 64-element quantization table. JPEG suggested Luminance Quantization Table used in DCT lossy compression as shown in Table 21.2.

The bits of a hidden message can then be embedded in the least significant digits of the quantized DCT coefficients. In practical JPEG steganography, the hidden message is usually encrypted before being embedded in the coefficients to enhance the security performance. The process is shown in Fig. 21.2 [1]. Moreover, the quantization table is modified in order to improve the embedding capacity.

Given below are the steps of a JPEG steganographic method based on quantization table modification [16]. The modified quantization table is shown in Table 21.3. In this table, the 30 coefficients located in the middle part are set to be one. Based on this quantization table, the secret message is embedded in the middle frequency part of the DCT coefficients.

**Table 21.2** Luminance quantization table

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

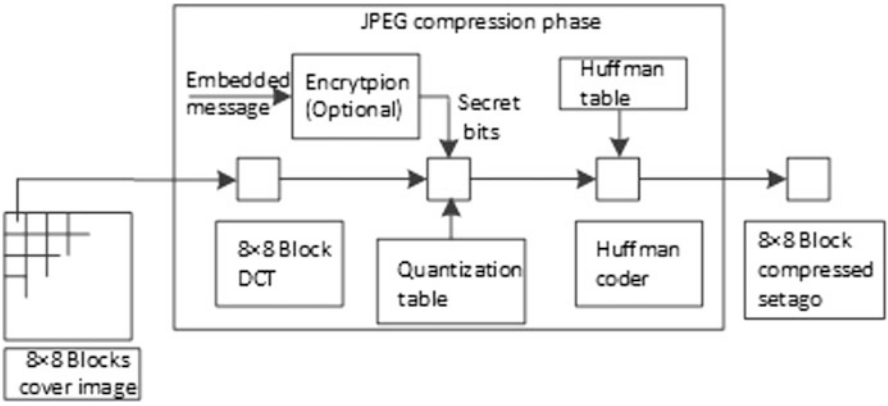



**Fig. 21.2** Data flow diagram showing the general process of embedding in the DCT domain

**Table 21.3** Modified quantization table

$$P = \begin{bmatrix} 16 & 11 & 10 & 1 & 1 & 1 & 1 & 1 \\ 12 & 12 & 1 & 1 & 1 & 1 & 1 & 55 \\ 14 & 1 & 1 & 1 & 1 & 1 & 69 & 56 \\ 1 & 1 & 1 & 1 & 1 & 87 & 80 & 62 \\ 1 & 1 & 1 & 1 & 68 & 109 & 103 & 77 \\ 1 & 1 & 1 & 64 & 81 & 104 & 113 & 92 \\ 1 & 1 & 78 & 87 & 103 & 121 & 120 & 101 \\ 1 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

- Step 1. A cover-image O with size N × N pixels is partitioned into non-overlapping blocks $\{O_1; O_2; O_3; \ldots; O_{N/8 \times N/8}\}$. Each $O_i$ contains 8 × 8 pixels.
- Step 2: Use DCT to transform each block O*i* into DCT coefficient matrix $F_i$, where $F_i[a, b] = DCT(O_i[a, b])$. Here, $O_i[a, b]$ is the pixel value in O*i*, $0 \leq a, b \leq 7$.
- Step 3: Use modified quantization table P to quantize each $F_i$. The result can be represented as $C_i[a, b] = truncate(F_i[a, b]/P[a, b])$.
- Step 4: Apply an encryption method with secret key k to encrypt the message M. The resulted message is $S = \{s_1, s_2, \ldots, s_m\}$, where $s_i$ is a secret bit and m is the length of S .
- Step 5: Select $C_i[a, b]$ to hide S respectively, where P[a,b] = 1. Each Ci [a,b] embeds two secret bits into it. The embedding order is shown in Fig. 21.3.
- Step 6: Apply JPEG entropy coding, which contains Huffman coding, Run-Length coding, and DPCM, to compress each block $C_i$. Collect the above results

$$C_i = \begin{bmatrix} C(0,0) & C(0,1) & C(0,2) & C(0,3) & C(0,4) & C(0,5) & C(0,6) & C(0,7) \\ C(1,0) & C(1,1) & C(1,2) & C(1,3) & C(1,4) & C(1,5) & C(1,6) & C(1,7) \\ C(2,0) & C(2,1) & C(2,2) & C(2,3) & C(2,4) & C(2,5) & C(2,6) & C(2,7) \\ C(3,0) & C(3,1) & C(3,2) & C(3,3) & C(3,4) & C(3,5) & C(3,6) & C(3,7) \\ C(4,0) & C(4,1) & C(4,2) & C(4,3) & C(4,4) & C(4,5) & C(4,6) & C(4,7) \\ C(5,0) & C(5,1) & C(5,2) & C(5,3) & C(5,4) & C(5,5) & C(5,6) & C(5,7) \\ C(6,0) & C(6,1) & C(6,2) & C(6,3) & C(6,4) & C(6,5) & C(6,6) & C(6,7) \\ C(7,0) & C(7,1) & C(7,2) & C(7,3) & C(7,4) & C(7,5) & C(7,6) & C(7,7) \end{bmatrix}$$

**Fig. 21.3** Embedding sequence

**Fig. 21.4** An example of JPEG steganography (**a**) A block of 88 pixel values; (**b**) The DCT coefficients

$$\begin{bmatrix} 124 & 130 & 135 & 138 & 140 & 141 & 140 & 140 \\ 129 & 140 & 141 & 143 & 147 & 143 & 143 & 143 \\ 135 & 140 & 150 & 151 & 140 & 139 & 139 & 139 \\ 140 & 145 & 146 & 140 & 140 & 142 & 142 & 142 \\ 140 & 146 & 147 & 147 & 147 & 140 & 140 & 140 \\ 150 & 150 & 150 & 150 & 149 & 149 & 148 & 148 \\ 151 & 151 & 150 & 152 & 148 & 148 & 149 & 149 \\ 150 & 150 & 149 & 149 & 150 & 150 & 150 & 150 \end{bmatrix}$$

(a)

$$\begin{bmatrix} 3261 & -10 & -31 & -24 & -8 & -4 & -4 & -1 \\ -101 & -33 & -27 & -14 & -7 & -4 & 0 & 1 \\ 1 & -32 & 0 & 10 & 4 & -1 & 0 & -2 \\ -6 & -11 & 0 & 10 & 4 & -5 & -3 & 1 \\ -30 & -7 & 6 & 7 & -4 & -7 & 1 & 5 \\ -7 & 2 & 12 & -1 & -4 & 9 & 7 & -4 \\ 0 & 4 & -4 & -4 & 2 & 10 & 8 & -3 \\ -12 & 10 & -16 & -12 & 7 & 8 & -1 & -2 \end{bmatrix}$$

(b)

and generate a JPEG file E that contains the quantization table p and all the compressed data.

- Step 7: Transfer the secret key k and the JPEG stego-image E to the receiver.

Assume that the original message is $1000110010100101100101101000011_2$ and it is encrypted as 10101100110100100101001010011102 with secret key $k$. Figure 21.4a lists a block of 8 × 8 pixels in the original cover-image. By using DCT, the block is transformed into DCT coefficients, as listed in Fig. 21.4b.

Before embedding the message in the cover-image, the quantization table P is used to quantize the DCT coefficients. The results of the quantized coefficients are listed in Fig. 21.5. Then, the secret message is embedded in the middle-frequency part of the quantized DCT coefficients, i.e., [0,3], [0,4], [0,5], [0,6], [0,7], [1,2],

**Fig. 21.6** The results of the block after embedding the message

$$\begin{bmatrix} 204 & -1 & -3 & -26 & -10 & -7 & -4 & -3 \\ -8 & -3 & -1 & -12 & -6 & -5 & 1 & 0 \\ 0 & 0 & 2 & 10 & 5 & -2 & 0 & 0 \\ 0 & -11 & 0 & 10 & 4 & 0 & 0 & 0 \\ -30 & -7 & 6 & 7 & 0 & 0 & 0 & 0 \\ -7 & 2 & 12 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ -12 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

[1,3], [1,4], [1,5], [1,6], [1,2], [2], [2,3], [2,4], [2,5], [3,0], [1,3], [2,3], [3], [3,4], [4,0], [1,4], [2,4], [3,4], [5,0], [1,5], [2,5], [6,0], [1,6], and [7,0]. The result is shown in Fig. 21.6.

Notably, which values in the 8x8 DCT coefficients block are selected to be altered is very important as changing one value will affect the whole $8 \times 8$ block in the image. However, careful consideration must be given to the sensitivity of DCT coefficients when selecting coefficients. Otherwise, it could result in distortion of the resulted stego-image, and some artefacts will be noticeable.

In summary, these DCT transforms convert the pixels in such a way as to give the effect of spreading the location of the pixel values over part of the image. The secret information is embedded in the LSB of the coefficients. Comparing to the steganography techniques in the spatial domain, they are complex but also more robust.

### *21.2.2 Steganography Tools*

There are a number of steganography tools available on the Internet [17], each with its own supporting one or more specific types of carrier file (or cover media) to embed hidden data inside it, such as an image, audio or video, and later extract that data. Few tools can hide data behind any file, and some even offers encryption before hiding the data to reduce the risk of data leaks.

Some of the popular tools to perform steganography include:

- Camouflage (available at: http://camouflage.unfiction.com/Download.html)
- OpenStego (available at: http://sourceforge.net/projects/openstego/files/)
- S-Tools (available at: http://www.cs.vu.nl/~ast/books/mos2/zebras.html)

Among the above tools, S-Tools, The Steganography Tools, is an easy-to-use yet powerful tool to hide data into audio and image files. Figure 21.7 is a screenshot of the main window of S-Tools with a bmp image file (“original-zebras.bmp”) opened.

Suppose that you want to hide a secret message into the opened bmp image file, and a secret message is saved in a text file called “secret.txt”. S-Tools is a drag and drop software so you can simply drag “secret.txt” from the directory where it resides into the S-Tools program.

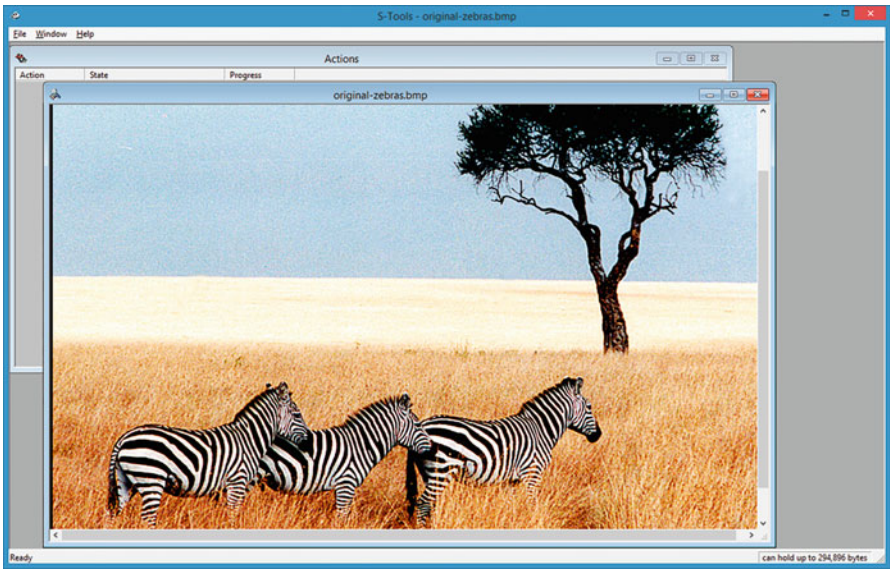



**Fig. 21.7** S-Tools

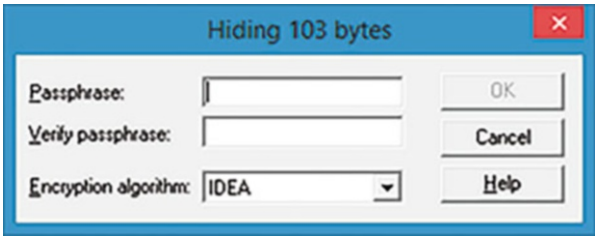



**Fig. 21.8** Hiding data using S-Tools

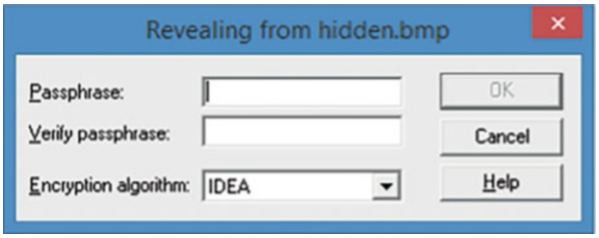
It is worth noting that S-Tools offers encryption before hiding the data. Afterwards, a pop-up dialog appears, showing the total size of the data (103 bytes in our case) that is hidden and also asking you to enter a secret key used by your chosen encryption algorithm to protect your hidden data (Fig. 21.8). Finally, the hidden data is encrypted and hidden into the file original "zebras.bmp". Next, we take a look at two images, the original zebras image and the one with a hidden secret message. Apparently, we cannot see any difference between two images with the naked eye, as shown in Fig. 21.9.

Also, you can simply extract the hidden data by the following

- Drag the image with the hidden data into the main window of S-Tools. Right click on the pictures, and then choose Reveal from the menu.
- Enter the pass phrase (secret key) twice and select the encryption algorithm used for hidden data into a pop-up dialog, shown in Fig. 21.10.


**Fig. 21.9** Hiding data using S-Tools. (**a**) Original image. (**b**) The image with hidden secret data

**Fig. 21.10** Revealing hidden data using S-Tools




- Wait until the Revealed Archive dialog box appears, where all the extracted hidden files are listed. Note that the user cannot open the hidden file from the S-Tools program.
- Right click on any hidden file retrieved and select Save As from the menu to save the file. Then, a **Save As** dialogue box will appear. Enter a valid file name, and select the working directory and click on the "Save" button. Repeat for the other ones. These are the hidden files.

## 21.3 Steganalytic Techniques and Steganalytic Tools

Steganalytic techniques can vary greatly depending on what information is known about the carrier, the message, and the algorithm used to embed the hidden message. These factors introduce a great complexity in designing a reliable steganalytic algorithm. Generally, the steganalytic techniques are classified under two categories: Specific approaches and universal approaches, based on whether the technique targets a specific steganographic method or can target most of the steganographic techniques. In the instances where steganographic techniques cannot be figured out,

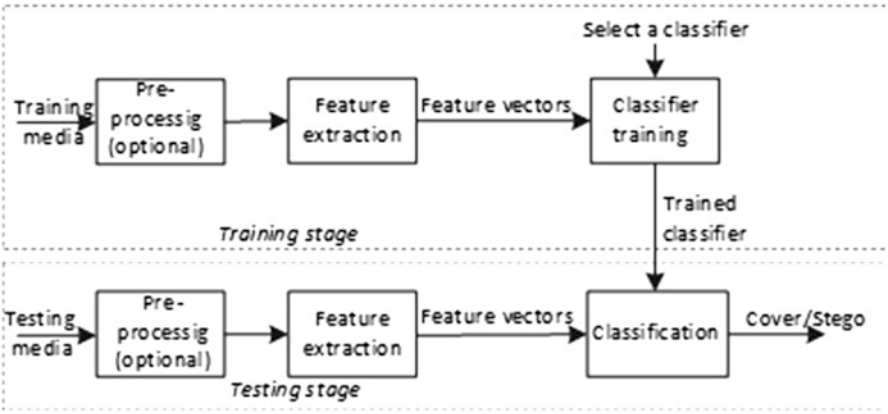



**Fig. 21.11** The process of a universal steganalytic method [18]

it is a challenge to come up with a detection mechanism that will work on all different steganography techniques. For that reason, we mainly focus on the basis of universal steganalytic techniques in this book.

### *21.3.1 Steganalytic Techniques*

A universal steganalytic approach usually takes a learning based strategy which involves a training stage and a testing stage, as illustrated in Fig. 21.11 [18]. In some techniques, medias are pre-processed for feature extraction. For example, images steganalytic techniques convert the RGB image into the grayscale image. In feature extraction, an input media from a high-dimensional space is mapped to a low-dimensional feature space. It is used both in training and testing stage. By classifier training, a trained classifier is obtained. Then, the trained classifier is used to classify an input image as either cover or a stego in the test process. Notably, some specific steganalytic methods may also take a similar learning based process. The difference between them lies in whether the features are effective in detecting a wide range of steganographic techniques.

The main differences among the techniques lie in the features selected for identifying the hidden messages. Also, they use different classifiers.

#### 21.3.1.1 Feature Extraction

Selection of statistic features is a key concern for designing a universal steganalysis algorithm. The extracted informative features should be sensitive to message embedding. Generally, good features own the characteristics: Accuracy, consistency and monotonicity [19]. Notably, detection accuracy should be consistency for a large

range of image sets. In other words, features should be independent to image's size, type, texture, settings, and access methods. Additionally, feature vector should be monotonic for the embedding ratios in stego images [20].

Usually, Mean square error, mean absolute error and weighted mean error are used as distortion metrics [21]. Also, The Probability Density Function (PDF) moment and Characteristic Function (CF) moment are two typical kinds of statistic features commonly used in universal steganalysis techniques [20]. analyzes the change trends of the statistic distribution parameters of various frequency sub-bands before and after message embedding such that providing a theoretical basis for the steganalysis feature selection and extraction. This work provides valuable information to researchers or engineers working in the field of steganography forensics or steganalysis.

#### 21.3.1.2 Classifier

Based on the extracted features, select and design the classifier is another important step for universal steganalytic techniques. Many effective classifiers, such as Fisher linear discriminant (FLD), support vector machine (SVM), neural network (NN), etc., can be selected.

We denote different classes by $w_i$, where each $w_i$ correspond to a different stego method, $1 < i < M$. Here $M$ refers to the existed number of classes. We denote the L dimensional feature vector by $\mathbf{X}$ [21],

$$p(\mathrm{X}/w_i) = \frac{1}{(2\pi)^{1/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(\mathrm{X}-\mu_i)^T \Sigma_i{}^{-1}(\mathrm{X}-\mu_i)\right) \tag{21.1}$$

where $\mu_i = E[\mathrm{X}]$ is the mean value of the $w_i$ class. $\Sigma_i$ is the covariance matrix defined as

$$\Sigma_i = E\left[(\mathrm{X}-\mu_i)(\mathrm{X}-\mu_i)^T\right]. \tag{21.2}$$

Where, $|\Sigma_i|$ denotes the determinant of $\Sigma_i$ and $E[\bullet]$ denotes the expected value.

Notably, if the number of training samples is limited, the high dimensionality of the problem adversely affects the classifier performance, which is sensitive to acquisition noise. One promising solution is to project the feature vector onto proper subspace.

After training the classifier by using the known types of media in the training media set, the parameters of classifier can be adjusted. Under the set threshold, the media can be classified by the trained classifier. Thus, judgments can be made to decide whether the images contain embedded messages or not.

### 21.3.2 Steganalysis Tools

Steganalysis tools have also been developed to detect stego messages embedded in digital media using steganography [7]. These tools are limited in their capabilities and target one or few specific cover objects. An example of such software is StegDetect. Stegdetect performs image steganalysis using statistical tests to determine if steganographic content is present. It can be used to detect jpeg images that have been altered using Steg, JPhide, Invisible Secrets, Outguess, F5, and others. StegDetect can be downloaded in DOS form as free ware from the Internet, available at http://www.brothersoft.com/stegdetect-download-306943.html.

**Review Questions**

1. LSB steganography are usually implemented in which of the following image formats? _____

   (a) GIF (b) JPEG (c) PNG (d) BMP

2. Describe in your own words, how do LSB image steganography techniques work?
3. List at least three typical steganography tools.
4. Described in your own words, what are the processes of universal steganalytic techniques? Conclude the main features and classifiers used in universal steganalytic techniques.
5. List at least two typical steganalytic tools.

## 21.4 Practice Exercises

The objective of this exercise is to perform steganography by using steganography tool. Specifically, we will use OpenStego to hide data into a cover file (e.g. an image file) and then extract hidden data.

### 21.4.1 Setting Up the Exercise Environment

- Download and install OpenStego
  Go to the following website: https://www.openstego.com/
  Click on the Download link on the upper right-hand side of the webpage
  Download "openstego-0.7.3.zip" and unzip it
- Start OpenStego (Fig. 21.12)
  Change directory to extracted folder, for example, c:\openstego-0.7.3
  Launch OpenStego by opening a Command Prompt and executing "openstego.bat"

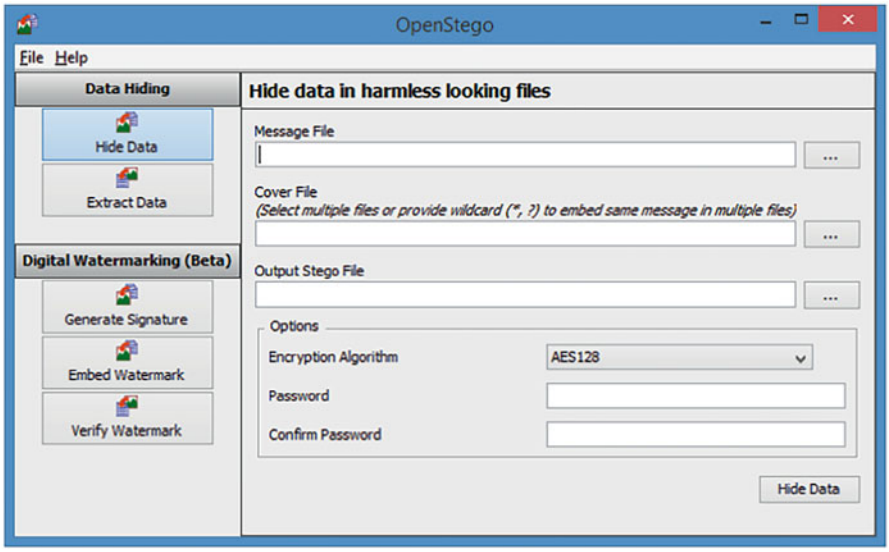



**Fig. 21.12** OpenStego user interface

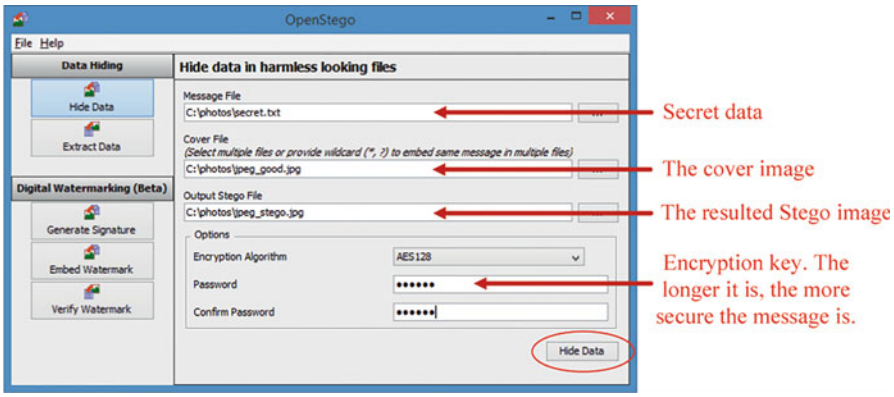
## *21.4.2 Exercises*

### Part A: Hiding Data

Download an image (e.g., a jpg file) from Google Images at https://www.google.com/, which is your cover file (or cover image).

Create a text file (e.g., "secret.txt"), which contains secret data to be hidden into the jpg image you just downloaded.

Launch OpenStego to hide secret file into the jpg image. The secret data will be protected by AES with 128-bit key.



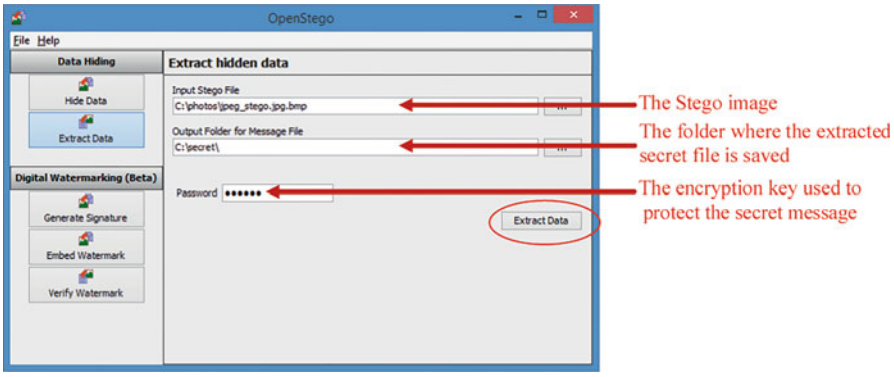

Click Hide Data

### Part B: Compare the Cover and Stego Images

Use any Image View Software (e.g., Windows Photo Viewer) to take a look at two images, the original image, which is the one before the steganography was done and the one with a hidden secret message, which is the one after the steganography was done. This will help you decide if there are any differences between the two images as seen with the naked eye.

### Part C: Extracting Hidden Data

Extract data hidden inside Stego file by using the same AES-128 key.




Click Extract Data.

## References

1. A. Cheddad, J. Condell, K. Curran, P. M. Kevitt, "Digital image steganography: Survey and analysis of current methods." Signal Processing, 2010, vol. 90, no. 3, pp. 727–752, March 2010
2. C. Hosmer and C. Hyde. Discovering covert digital evidence. Digital Forensic Research Workshop (DFRWS) 2003, August 2003 [Online]. (January 4, 2004). Available: http://www.dfrws.org/dfrws2003/presentations/Paper-Hosmer-digitalevidence.pdf
3. Jordan Green, Ian Levstein, Robert J. Boggs, Terry Fenger. Steganography Analysis: Efficacy and Response-Time of Current Steganalysis Software. http://www.marshall.edu/forensics/files/GreenJordan_Research-Paper_08_07_20141.pdf
4. A. Whitehead, "Towards Eliminating Steganographic Communication", Proc. International Conference on Privacy, Security and Trust (PST), October 12-14, 2005, New Brunswick, Canada
5. A. K. Shukla, "Data Hiding in Digital Images", A Review[C] STEG'04: Pacific Rim Workshop on Digital Steganography, 2004
6. W. R. Bender, D. Gruhl, N. Morimoto, "Techniques for data hiding." IS&T/SPIE's Symposium on Electronic Imaging: Science & Technology International Society for Optics and Photonics, 1995, vol. 35, NOS. 3&4, pp. 313-336
7. P. Hayati, V. Potdar, E. Chang, "A Survey of Steganographic and Steganalytic Tools for the Digital Forensic Investigator." http://www.pedramhayati.com/images/docs/survey_of_steganography_and_steganalytic_tools.pdf

8. W. R. Bender, D. Gruhl, N. Morimoto, A. Lu, “Techniques for data hiding”, IBM Systems Journal, vol. 35, no. 3.4, pp. 313-336, 1996
9. G. C. Kessler, “An Overview of Steganography for the Computer Forensics Examiner”, Forensic Science Communications, 2004
10. M. Bachrach and F. Y. Shih, “Image Steganography and Steganalysis”, Wiley Interdisciplinary Reviews Computational Statistics, 2011, vol. 3, pp. 251-259
11. G. L. Smitha, E. Baburaj, “A Survey on Image Steganography Based on Least Significant Bit Matched Revisited (LSBMR) Algorithm.” Proc. International Conference on Emerging Technological Trends, 2016
12. W. Luo, F. Huang, J. Huang, “Edge Adaptive Image Steganography Based on LSB Matching Revisited.” IEEE Transactions on Information Forensics & Security, 2010, pp. 201-214
13. J. Harmsen, W. Pearlman, “Steganalysis of additive-noise modelable information hiding”, Spie Processing, 2003, 5020:131-142
14. J. Mielikainen, “LSB Matching Revisited”, IEEE Signal Processing Letters, 2006, vol. 13, no. 5, pp. 285-287
15. R. J. Anderson, “Stretching the Limits of Steganography.” International Workshop on Information Hiding Springer-Verlag, 1996, vol. 1174, no. 4, pp. 39-48
16. C. C. Chang, T. S. Chen, and L. Z. Chung. “A steganographic Method Based Upon JPEG and Quantization Table Modification.” Information Sciences, 2002, vol. 141, no. 1–2, pp. 123-138
17. List of 10 Best Steganography Tools to Hide Data. https://www.geekdashboard.com/best-steganography-tools/#openstego
18. B. Li, J. He, J. Huang, Y. Shi, A survey on image steganography and steganalysis, Department of Computing, vol. 2, no. 3, pp. 288-289, 2011
19. I. Avcibas, N. Memon, and B. Sankur, “Steganalysis using image quality metrics”, IEEE Trans. Image Process., vol. 12, no. 2, pp. 221-229, 2003
20. X. Luo, F. Liu, S. Lian, C. Yang, S. Gritzalis, “On the Typical Statistic Features for Image Blind Steganalysis.” IEEE Journal on Selected Areas in Communications, 2011, vol. 29, no. 7, pp. 1404-1422
21. M. U. Celik, G. Sharma, and A. M. Tekalp, “Universal Image Steganalysis Using Rate-Distortion Curves”, Proc. Security, Steganography, and Watermarking of Multimedia Contents VI, vol. 5306, pp. 467-476, San Jose, California, USA, 2004