THE PUBLISHING HOUSE
OF THE ROMANIAN ACADEMY

PROCEEDINGS OF THE ROMANIAN ACADEMY, Series A,
Volume 16, Special Issue 2015, pp. 299-312

IS HIDDEN DATA SAFE?
ANALYSIS OF THE PUBLIC CRYPT&HIDE-STEGANOS APPLICATION

Relu TATARU[1], Adriana VLAD[2], Catalin GUGU[3]

[1]Faculty of Electronics, Telecommunications and Information Technology, Romania
[2]The Research Institute for Artificial Intelligence, Romanian Academy, Romania
[3]Independent researcher
E-mail: avlad@racai.ro

This paper addresses a practical way of analyzing a security application that aims to protect user's sensitive data using both data encryption and data hiding techniques, to conceal the existence of sensitive data within multimedia files. Crypt&Hide application, targeted in our analysis, is a part of Steganos Privacy Suite. The application was tested to establish the security level assured for the sensitive data files hidden into JPEG images. Our analysis featured both encryption and steganographic layers of the application, with emphasis on the data hiding component. Certain modifications occurred during the steganographic embedding process raise suspicions about the authenticity of carrier files, even for a primary metadata analysis. The experimental results from our testing scenarios show that stego images can be identified with a high accuracy during the classification process. The existence of such situations may lead to the failure of the application in terms of data hiding and data protection.

Key words: cryptography, steganography, steganalysis, JPEG, AES, F5, ensemble classifier, SVM.

1. INTRODUCTION

Information security is one of the greatest challenges nowadays in communication networks. The exponential growth of WEB and mobile applications encouraged users to share data online. Exchanging a large amount of data every second on unsecured channels requires the use of cryptography for keeping the transmitted and stored data secure.

Modern cryptography is focused on developing efficient algorithms which are intended to work in secure communication protocols used in the online environment and also locally (on personal computers), aiming to protect user's sensitive data. Although cryptographic algorithms (symmetric or asymmetric) currently provide a high level of security, it does not mean that information released in the online environment is really secure. Attackers have been always capable to find flaws in the cryptographic applications and protocols that implement cryptographic algorithms. Attacks were often found for cryptographic applications many years after they were released. Effective cryptographic algorithms may fail in their usage, due to subtle implementation flaws in the cryptographic applications.

Steganography represents another way to assure data protection and is frequently used along with cryptography to reach its goal *i.e.* hiding information inside a seemingly innocent digital content. Steganography deals with hidden information in multimedia objects (text, image, video or audio files) in such a manner that only the sender and receiver know about the presence of the secret message into the media object. Message protection using cryptographic tools is obvious and the exchange of secret information is clear for an attacker when intercepting the communication flow. Steganography may be considered a complement and not a substitution for cryptographic applications. A steganographic algorithm is effective and is called secure if it fulfills the main requirement of steganography, namely *undetectability*.

Generally, the main operations for steganographic algorithms are *embedding i.e.* message insertion in cover object to create the *stego object* and *extraction i.e.* the extraction of the hidden message from the *stego object*. Both operations, *i.e. embedding* and *extraction* are accomplished by using a steganographic key.

The steganographic algorithm is featured by an *embedding capacity i.e.* the maximum number of bits that can be hidden in a given cover object (to obtain the so called *stego object*) and also by a *steganographic capacity i.e.* the maximum number of bits that can be embedded so that the probability of detection is negligible [1]. The later must be sufficiently large to comply with the requirements of a good practical steganographic application.

Our study is focused on both cryptography and steganography with emphasis on the analysis of a public steganographic application - the Crypt&Hide (C&H) tool from the German software suite, Steganos Privacy Suite 14. The main goal is to present the risks of using this steganographic tool for hiding sensitive information on the personal computer or for sending stego objects over the network. The investigation field is the JPEG domain for digital color images. Digital images are used as hosts for information hiding by using the C&H German tool and represent the most analyzed multimedia source in the data hiding field.

The motivation behind using Steganos as a target for our analysis and security investigations was based on the following:

- The application deals with the sensitive fields of data encryption and data hiding;
- The German company has an important background in information security, built over many years; Their product's quality is mainly a result of the reputation and experience of this company;
- JPEG files used as carriers within the application are the most popular file format in Data Hiding community;
- A primary comparison with other steganographic applications (generally free), indicates a trustful steganographic approach, due to elements such as: secure data encryption of hidden files, efficient hiding and providing a powerful tool for secure data deletion (mandatory for removing the original secret file);
- If such an application is used further than data storage (e.g. secret communication on online channels), its possible vulnerabilities may represent a major risk not only for the secret data, but also for the users (sender and recipient trying to communicate secretly).

All these features make the Steganos/Crypt&Hide application a suitable candidate for security investigation. Steganography usage as an additional security layer is generally exclusivist. Data protected with this kind of tools, clearly have a special importance. Thus, a steganographic application must implement a secure steganographic technique *i.e.* the resulted carrier files should be very unlikely to detect.

This paper is organized as follows. Section 2 presents digital images with emphasis on JPEG images and lossy compression (readers already familiarized with JPEG compression may skip this section). Section 3 describes the software Steganos Pivacy Suite, highlighting the capabilities and characteristics of the Crypt&Hide tool. In Section 3 we also expose our experimental results when analyzing the performances of the stego application. Section 4 concludes our work.

2. DIGITAL COLOR IMAGES AND JPEG LOSSY COMPRESSION

2.1. Color images and Compression

Photos taken by people with their own digital photo cameras and images from the Internet are all digital images, mostly represented in the Red-Green-Blue (RGB) classical color space. Depending on the application, there are also other color spaces e.g. CMYK, CIE, YCbCr, used for image representation. Digital images are typically represented by a bit depth ranging up to 24, but could be higher. 24-bit images are the most popular in the online environment and are normally represented in the RGB color space (each pixel in the digital image is represented by a 3-tuple containing one value for each color channel, RGB). Different combinations of these values produce various colors.

Classical RGB color images are mostly represented by three matrices with 8-bit values (in the [0, 255] range), one for each color channel. The file size in bytes for a 24-bit digital image considering a matrix representation, is computed by multiplying the size of the matrix (height x width) by the bit-depth and

dividing the result by 8. This representation leads to significant file sizes that are not reliable for storage, processing or transmission.

Lossless compressions (*i.e.* original image can be entirely reconstructed from compressed data) and *lossy* compressions (*i.e.* some information from original image is lost during compression) are the main solutions to reduce the image file size diminishing the existing redundancy from the image. JPEG lossy compression method is by far the most popular lossy compression method due to the overwhelming percentage of color images represented using *.jpg* extension on Internet. JPEG (Joint Photographic Experts Group) – the group who created the standard, proposed two main formats: EXIF (Exchangeable Image Format) which is mainly used in digital cameras nowadays and JFIF (JPEG File Interchange Format) which is the minimal JPEG format that enables image exchanging on the Internet. JPEG compression provides a selectable degree of compression level: higher image qualities (obtained for smaller degree of compression) enlarge the file size while lower image qualities (obtained for higher degree of compression) represent an option to achieve smaller file sizes. Usually, the visual effects on the compressed digital image are zero or insignificant, for a reasonable quality factor (larger than 75).

The JPEG loosy compression standard has several variations. One of the most common and simple modes is briefly presented below. Compression steps are mandatory for the understanding of most of the JPEG based steganographic tools that implement steganographic algorithms. The steganalysis of digital images (*i.e.* the art proposing techniques to detect the presence of hidden messages inside digital images) is strongly influenced by the image format, thus by the compression method.

2.2. JPEG compression for digital color images

1. Color space transformation

The digital image is converted from RGB to YCbCr color space (Y – luminance, representing the brightness from the image, Cb and Cr – chrominances for blue and red), eq. 1. This color space conversion (RGB to YCbCr) allows larger compressions without affecting the image quality. Luminance is visually more important than chrominance due to the Human Visual System (HVS) sensitivity to the intensity of colors. Thus, luminance channel is less compressed in order to avoid image quality loss and chrominance channels allow larger compression [2].

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.5 \\ 0.5 & -0.419 & -0.081 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (1)$$

2. Downsampling

Downsampling of the chrominances is applied due to the smaller sensitivity of the human eye when changing these two color spaces. An averaging for each chrominance channel is applied at block level. Usually the reduction applied is by a factor two and an immediate reduction in file size is achieved [3].

3. Discrete Cosine Transformation

Luminance and chrominance channels are grouped into 8x8 pixel blocks. These pixel matrices are transformed from spatial (pixel) domain to frequency domain, using the Discrete Cosine Transform (DCT) [4]. DCT changes the information distribution for each block and separates it into three frequency bands: low, medium and high. The first top-left coefficient of the block is called DC term and represents the average value in the block. All other coefficients are called AC terms and represent the frequency of changes in the image blocks. For each color channel, a DCT representation is obtained at the block level. These blocks are the usually called non-quantized DCT blocks.

4. Quantization

The quantization is the compression step where an important amount of information is discarded [4]. The 8×8 DCT blocks are quantized using an 8×8 quantization matrix corresponding to a certain quality factor. Each DCT coefficient is divided by the correspondent quantization coefficient from the quantization matrix and rounded to an integer which is referred as the quantized DCT coefficient. Large quantization coefficients cause more data loss, hence a smaller quantized DCT value. Low frequency AC terms are less

quantized comparing to middle and high AC frequencies. At this step, the quality factor decides the JPEG image quality.

5. DCT Encoding

The quantized DCT coefficients are selected in Zig-Zag order and compressed using dual Huffman/RLE encoding techniques. The concatenation of the specific JPEG header information and the codewords obtained after this step forms the JPEG data file.

All steps presented above represent the JPEG compression steps and are also briefly illustrated in Fig. 1. When displaying a JPEG image, all these reversible steps are inversed and the spatial domain representation of the image is shown to the user. This process is called decompression and is used by all JPEG editors to display images. It may also be used to reverse the JPEG compression method and obtain the raw form of the image.

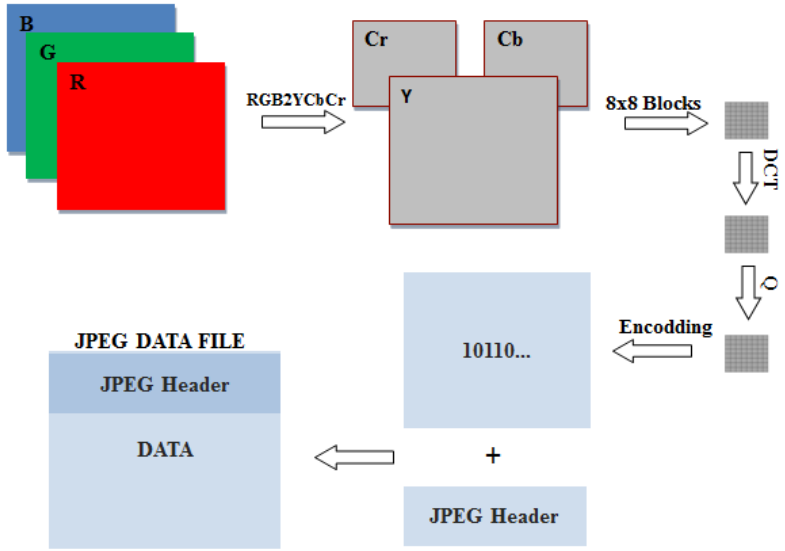


Fig. 1 – JPEG compression.

JPEG compression is essential for a deep understanding of steganography and steganalysis working on JPEG images.

3. STEGANOS PRIVACY SUITE

3.1. Description of Steganos Privacy Suite

Steganos Privacy Suite was developed by the German company Steganos as a tool with multiple functionalities with the aim of protecting user's sensitive data. This company has an important security background and provides a variation of security solutions for many years. The security suite was developed as an extension of antivirus programs, which do not provide the absolute protection of the user data. The last version of Steganos Privacy Suite is dated on 16 September 2014 and performs 384-bit AES-XEX (Advanced Encryption Standard – Xor-Encrypt-Xor Mode) data encryption [5].

Encrypted data can be stored on personal computers, on mobile devices and even in Cloud. Steganos Privacy Suite 16 has the following components: Steganos Safe, Password Manager, Portable Safe, Private Favorites, TraceDestructor, Traceless Drives, Shredder and Crypt & Hide.

- *Steganos Safe* is able to keep safe sensitive data on a personal computer, on mobile devices and in Cloud;
- *Password Manager* creates strong passwords which are automatically inserted on different websites, where user credentials are required. The user only needs to memorize one password to get access to the Password Manager;

- *Portable Safe* archives sensitive data on an USB stick, CD, DVD or Blu-ray disc;
- *Private Favorites* saves browser bookmarks that contain sensitive content;
- *E-mail Encryption* allows encrypted transmission of e-mails. Due to password based protection, only the receiver can read the email contents;
- *Shredder* deletes files performing a clean and secure data deletion;
- *TraceDestructor* erases traces of user activity from the personal computer;
- *Traceless Drives* runs programs on temporary drives. Data disappear when the computer is shut down or when the user commands this action;
- *Crypt&Hide* hides encrypted sensitive data in pictures and audio files.

With all these tools, Steganos is a complex and powerful suite performing data protection. Our paper illustrates a depth analysis of the Crypt&Hide (C&H) tool, very close to a real use scenario. This application is available for Steganos Privacy Suite 16 and also for previous versions of the security suite. Crypt&Hide uses a steganographic algorithm to hide sensitive data files in media objects such as JPEG or BMP images or audio WAV files. The investigation in what follows aims to identify possible vulnerabilities of Crypt & Hide tool when embedding data in JPEG images (a file format type much more popular than the alternatives BMP or WAV).

3.2. Crypt&Hide for JPEG files

C&H embeds a data file inside a JPEG image and creates the stego file such as a data archive. Firstly, the application allows the selection of the secret file and secondly the selection of the host image. To complete the steganographic process, the user password is required. After finishing the data concealing process, the stego image is obtained. The original image is overwritten with the stego image. Sensitive data within the secret file remains available on disk. To ensure complete data protection, the user can use a specialized tool such as Steganos/Shredder to remove permanently the file with sensitive data. The only way to access the sensitive data is by using the carrier file (stego object) within the steganographic application, with the correct user password. The knowledge of the carrier file is mandatory. If the sensitive data file is too large to be embedded in one image, the steganographic tool C&H does not allow the user to conceal the message in multiple images accessing one single operation. In this case (when the data file is too large to be hidden into a cover image), the user may split manually the secret file into multiple files and hide them in multiple images, using several operations. C&H was programmed to warn the user when the secret file is too large for the host image.

This steganographic application was developed to hide data locally, but people may be tempted to use it also for sharing secret data over data networks. Such approach is only valid over channels that do not perturb the content of the carrier files using image processing operations (e.g. image scaling, image compression, image enhancement etc).

The German company Steganos did not provide any information about the hiding technique, hence the steganalyst knows nothing about the steganographic algorithm behind C&H. This scenario is frequently encountered in real situations for such security products. Producers often choose to keep secret the hiding algorithm for this kind of application, in order to make it harder for the attacker to break their steganographic application.

The analysis aims at highlighting vulnerabilities of C&H application, by showing several experimental results obtained in practical situations that generate unsecure hiding of the sensitive data file. Skillfully exploited by the attacker, these vulnerabilities make it possible to identify the stego objects. The consequences of this risky usage of the tool may strongly disturb the user, depending on the attacker's intention.

Note that we will distinguish in this paper the notions of attacker and steganalyst. We suppose that an attacker wants to compromise the steganographic application to gain benefits using the user’s application or carrier files. On the other side, the steganalyst targets only the steganographic application and the stego files. His main purpose is to evaluate a detection rate for images modified with the steganographic application. Here, the authors follow the steganalyst role, noticing the existence of vulnerabilities that may be exploited by an attacker. When steganography fails to achieve its main goal (undetectability), carrier detection may endanger hidden data. Without the user password, an attacker is not able to extract the sensitive information,

not even the encrypted content (see subsection 3.3). However, when the attacker manages to identify the stego images, he may be able to cause serious issues to the user such as: stealing the carrier files and asking some benefits in return, corrupting the content of the stego images causing data loss or recovering sensitive data using dictionary attacks.

3.3. Crypt&Hide Analysis

The investigation was conducted using Steganos Privacy Suite 14. C&H can be used either for data encryption generating Steganos Encrypted Data Files (using only the Crypt component from the Crypt&Hide) or for both data encryption and data hiding (using both features of Crypt&Hide tool). To gather information about the hiding technique used within the application, cover/stego image pairs were created and analyzed. Image analysis was performed for the two JPEG formats: JFIF (JPEG File Interchange Format) and EXIF (EXchangeable Image Format). JFIF is a simple format with a minimal header containing essential metadata necessary for image editors to decode and display JPEG images; this format is mostly used for low resolution images exposed on the WEB. Unlike JFIF, EXIF provides a complex header being used by most digital cameras nowadays. EXIF header contains important data such as: author name, camera identification, location etc.

Our investigation concerning the C&H application was focused on JPEG images for both essential image components: header information (mostly for EXIF format) and content information (both EXIF and JFIF).

To analyze header information for JPEG images modified with C&H, we used authentic photos taken with our digital cameras and smartphones. Secret data files were hidden inside these original images, obtaining stego images. The headers of both authentic and stego images were analyzed and compared.

The image content, *i.e.* the byte stream representing the encoding of the pixels from the image, was also analyzed (the analysis is valid for both EXIF and JFIF formats). Most of the images used in this part of investigation were downloaded from the Break-Our-Steganographic-System (BOSS) competition website [6]. The images were downloaded in CR2 (Cannon Raw) format. To make them useful, we converted them into BMP file format; we resized them to 782×521 and converted into JPEG format. This small resolution was used to facilitate the creation of stego files using C&H application and to attain a good speed when analyzing image content using statistical tools. Such an approach is common in the literature when dealing with large image databases. Also, this resolution is larger than some existent resolutions of different smartphones (e.g. Samsung's minimum resolution for many smartphones is 640×480).

C&H application firstly compresses the sensitive data file and afterwards encrypts it using AES256. Our experiments proved the existence of compression by noticing important size differences existing between the original data files (English and Romanian natural texts) and the corresponding encrypted data files (EDF – *i.e.* Steganos Encrypted Data File extension). Message compression is used in steganographic applications because of the limited capacity of digital images to host the secret message. We further denote by the secret message the final bit stream hidden in the image, representing the message compressed and encrypted by Steganos in the Encrypted Data File (EDF).

The functional scheme of C&H is illustrated in Fig. 2. The Steganos Encrypted Data File (EDF) generated by the application is hidden into the JPEG image using a steganographic technique that works on quantized DCT. When data hiding process is complete, the stego image provided by the application is just like an archive. This archive is a JPEG file, which can be displayed with any JPEG editor. Visually, the carrier JPEG file and the original JPEG file are identical. Sensitive data hidden inside the image can be extracted only using C&H.

For investigation purposes we used in the hiding process text files with dimensions: 0Kb, 1Kb, 2Kb and 3Kb, containing random data (excepting the 0Kb file which was completely empty). Small sizes of data files were chosen such that the application does not reject the files (for our low resolution host images *i.e.* 782×521).

Analyzing the Steganos EDFs obtained from empty data files, it can be noticed that they contain a header component and a content component. More precisely, the size of the EDF file corresponding to an empty data file is around 400 bytes (the exact size of the header remains unknown). The header component influences the encryption process. Each time a data file is encrypted, a new version of the EDF file (with the

same size) is obtained. The encryption protocol was not published by the producers, being unknown to the authors of this paper. This application justifies the presence of the empty files in the analysis of carrier files.

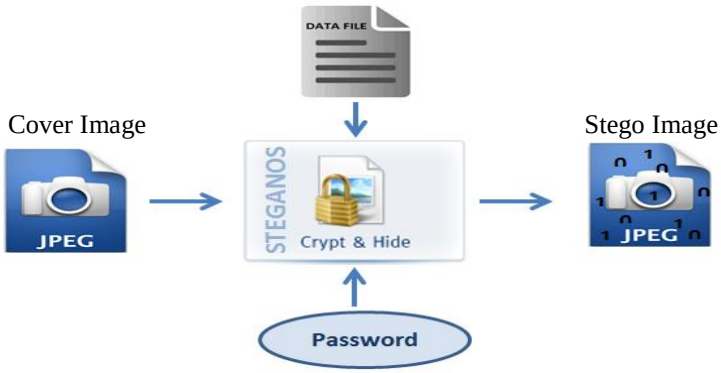


Fig. 2 – Crypt&Hide.

Our motivation for using random data files as sensitive information for the steganographic process was the better control of the EDFs file through the elimination of the compression step.

3.4. Metadata Analysis

A steganographic application that aims to provide undetectable stego images after data hiding process should assign to carrier files authentic header information. The simple analysis of an image set (containing both original and stego files) should not draw attention to carrier files due to unexpected header information. The images used to analyze the influence of C&H at header level were in EXIF/JPEG format and were provided by the following digital cameras and smartphones: Sony DSC-W200, Panasonic Lumix DMC-TZ3, Canon A550, Samsung Galaxy S3 Neo and HTC ONE M7. Comparing header information of original and stego images, we observed that they are mostly similar, but with at least one important difference: the presence of the JFIF specific marker APP0 (0xFF 0xE0) inside the header of carrier files. The difference between an authentic image (photo taken with HTC ONE M7) and a stego image obtained using Crypt&Hide (empty text file was hidden) at header info level is shown in Fig. 3 in two ways. Figs. 3a) and 3b) illustrate the differences using a hex viewer. In Fig. 3c) and 3d) the differences are illustrated using the JpegSnoop EXIF editor [7]. Both testing scenarios prove the presence of the APP0 marker, followed by a specific byte stream with length 16, only for the stego image. We repeated our tests for all our cameras and smartphones and we obtained the same behavior.

The presence of two types of header for images provided by the same image source may be noticed by an attacker. In this case, digital images containing both JFIF and EXIF markers may be considered suspicious and they represent in our opinion an important vulnerability for the user who believes that his sensitive data is untraceable. The same vulnerability remains for EXIF/JPEG stego images shared by the user in the online environment (in environments that preserve the image header information).

Another issue identified for JPEG stego images created with C&H is the modification date ("Modified:"), which is changed by the steganographic application.

Note that modification date of the image is available on most of the operating systems by right clicking the image file and checking the image properties. When an image set (stored in a directory from the hard drive) contains images taken in the same period and location (e.g. on the same day and in the same city during holiday, which happens every day for people all over the world) but with different modification date, we normally deal with a modification process. In this situation, pictures with different modification date were normally modified with software, other than JPEG encoder of the photo camera.

Also, note that original date time, *i.e.* the moment when the image was digitized for the first time, is available in the EXIF header. C&H does not change this date, so the original date is still available when analyzing a stego image. Using JpegSnoop, original date time is available by regarding the values for *DateTime* (in EXIF IFD0), *DateTimeOriginal* and *DateTimeDigitized* (in EXIF IFD1); note that IFD stands for Image File Directory.

Image creation and image access dates (“Created:” and “Accessed:”) are modified by a simple transfer, which is normal for users moving pictures from one location to another (e.g. from camera card to personal computer). This type of operation does not affect the modification date, which is normally changed only when the encoded representation of the image is modified. Although there are plausible situations of changing the modification date (e.g. image rotation, image enhancement, brightness adjustment, image transfer on channel that re-encodes the JPEG images etc.), the attacker’s attention may be caught by the changes in the modification date and why not avoid that? An attacker dealing with such scenario (*i.e.* similar images taken in the same place, in the same period with major differences between modification dates) may have necessary tools to investigate if differences were caused by natural or steganographic processing. Generally, photo editors used to process images (e.g. Photoshop) put their signature on the image file, leaving obvious traces in image header and content.

In our opinion, the clues that we described above (additional APP0 marker specific for JFIF/JPEG and different modification date) represent a good start for an attacker to begin a much more powerful analysis at the image content level, using steganalysis tools to decide on the authenticity of the targeted images.

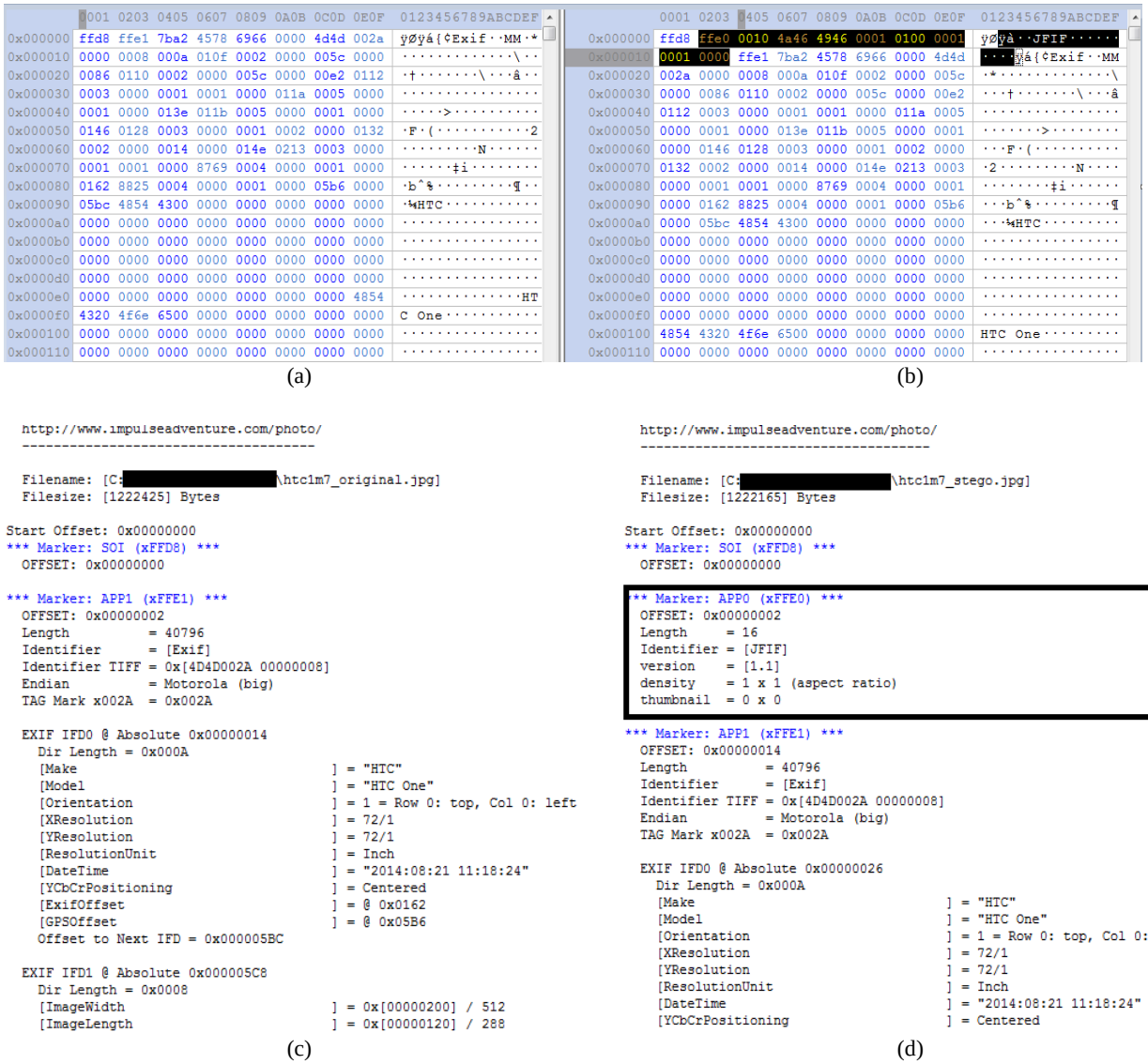


Fig. 3 – EXIF header modification of the stego image:
(a) partial original header hexview; (b) partial stego header hexview;
(c) partial original header JpegSnoop; (d) partial stego header JpegSnoop.

3.5. Content Analysis

The information concerning the C&H application (regarding hiding domain and steganographic distortion function) was obtained by analyzing cover/stego image pairs. We started our content analysis by checking differences between quantized DCT coefficients of cover/stego image pairs, relying on previous steganographic algorithms working at this level [8-12]. To achieve this, cover and stego JPEG images were decoded until obtaining the quantized DCT coefficients. By direct comparison at DCT level, we obtained the distortion function expressed by eq. 2.

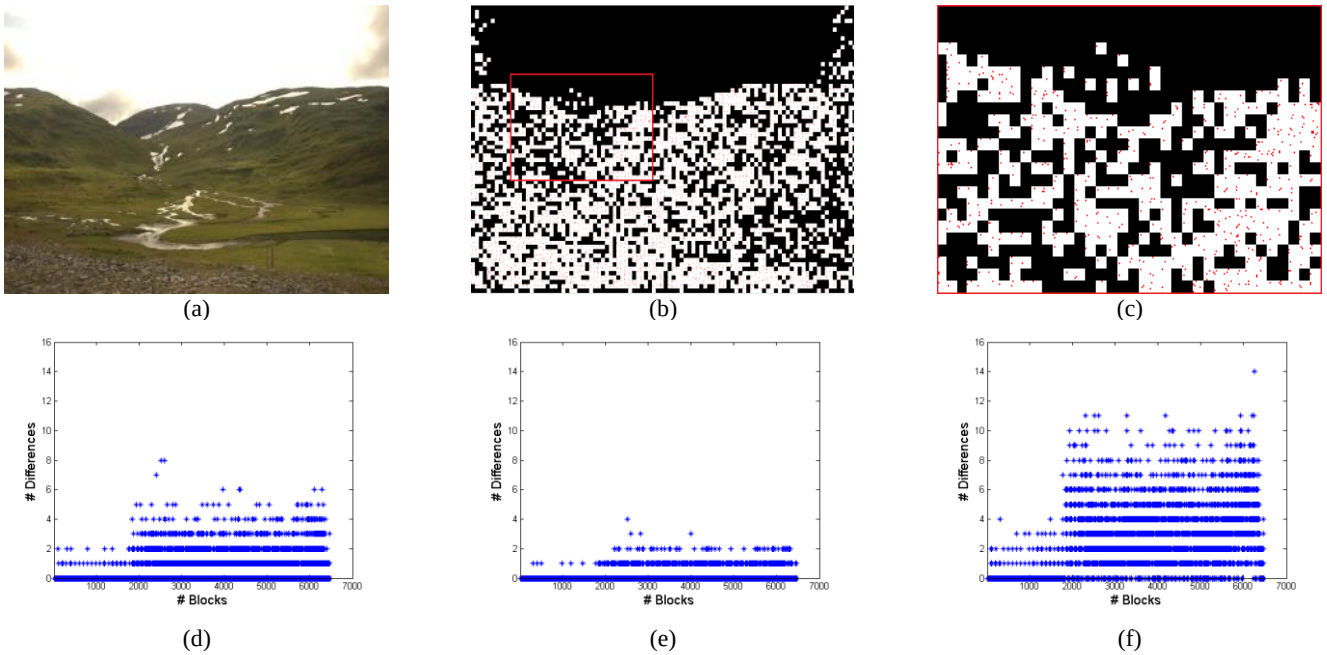
$$f(x) = sign(x) \cdot (|x| - 1) \quad (2),$$

where x is the value of DCT coefficient in original image, $|\cdot|$ represents the absolute value, and $f(x)$ is the modified DCT coefficient from the stego image. The distortion function, *i.e.* the technique used to change DCT coefficients in order to hide secret message bits, is the same as the one used for F5 algorithm [9]. We also observed by analyzing cover/stego image pairs that DC coefficients and null AC coefficients are not modified during the embedding process, which is another similarity with the F5 algorithm. The identified clues made us suppose (without being certain!) that C&H uses the F5 algorithm or a similar algorithm to hide messages with restricted sizes with respect to the host image. Although our analysis identified the distortion function, we could not determine the encoding technique that uses DCT coefficients to hide secret message bits. However we provide in this section important clues about its behavior in this section. We targeted our analysis on the following topics: hiding domain; hiding efficiency; embedding distribution and DCT histograms; statistical analysis and classification using CC-PEV image features.

Hiding Domain. The host image is decoded by C&H and quantized DCT coefficients are obtained for Y, Cb and Cr color channels. Only non-zero AC coefficients from these channels are used to hide sensitive information for security reasons. Most of the information is embedded in the luminance channel (Y), due to the larger number of non-zero AC coefficients from this channel. Normally, the embedding process of one or more secret bits is not accomplished when a non-zero AC coefficient has become zero after embedding. This happens because the extraction process would ignore zero AC coefficients of the stego image. The application continues with the hiding of the same secret bits, using other non-zero AC coefficients, until performing a valid modification. This phenomenon is called shrinkage and occurs every time the distortion function (from eq. 2) is applied to the values +1 and –1. Shrinkage was defined in steganography by Andreas Westfeld in [9], where he also proposed the F5 steganographic algorithm. This phenomenon represented the main weakness and led to the breaking of F5 in [13]. In order to quantify the shrinkage for C&H, we considered it as the ratio between the number of non-zero AC coefficients that become zero after embedding and the total number of AC modifications.

Our experimental results prove the spreading of steganographic modifications on almost the entire image surface, for both short and long hidden messages. Fig. 4b) illustrates the differences between a cover and her correspondent stego image (from Fig. 4a) at DCT block level on the luminance channel. The image used in this experiment has resolution 782×521. The length of the hidden secret file was 1Kb (random data file). DCT blocks altered by the steganographic process are marked with white color. For a better illustration of steganographic modifications, we emphasize in Fig. 4c) some modified DCT blocks, by cropping the difference image from Fig. 4b) (see the red rectangle). The cropped difference image shows more clearly the 8×8 DCT blocks used in the steganographic process. Red points mark the position of steganographic modifications in the block. Figs. 4a) and 4b) show higher modification rates for transitional/edge areas of the image compared to smooth areas. Fig. 4d) illustrates the number of modifications on DCT blocks when inserting the file with size 1Kb. The graphic was built considering a raster scan of the difference image. Similarly, Figs. 4e) and 4f) provide the distribution of steganographic modifications when inserting 0Kb and 3Kb files starting from the same host image shown in Fig. 4a). As expected, the number of steganographic modifications per block is highly affected by the message length.

Steganography proved over the years that hiding data in edge areas of the image is harder to detect for both spatial and frequency domain. The variation from the edge zones is much more intense compared to the variation from smooth zones, being more complicated to predict and from here comes the difficulty for steganalysis. This also explains why the sky from the cover image was not used in the embedding process.



(a) (b) (c)

(d) (e) (f)

Fig. 4 – (a) Original Image; (b) DCT block differences (Y) cover vs. stego (1KB); (c) cropped Block Differences; (d) differences per 8×8 block on Y(1KB); (e) differences per DCT block on Y(0KB) (f) differences per 8×8 block on Y (3KB) .

Hiding Efficiency. The hiding (embedding) efficiency (*i.e.* the average number of bits embedded per change of DCT coefficient) is the ratio of the hidden EDF length to the total number of modified DCT coefficients. Our analysis shows that C&H uses a message encoding technique to increase the embedding efficiency.

Note that message encoding represents a mathematical function which is based on a distortion function, to hide E bits by doing only one modification. E value is the embedding efficiency. Secret message encoding is performed aiming to increase the number of bits embedded for each non-zero AC modification. This is another similarity with the F5 algorithm, which uses matrix encoding to insert the message into the image. The use of encoding techniques is very popular in data hiding, because it significantly reduces the amount of distortion in the image. More popular and efficient techniques for secret message encoding are presented in [14] and [15]. Analyzing C&H, we noticed that both embedding efficiency and the number of non-zero AC coefficients are influenced by image quality, image resolution and message length; all these elements are reported to the same reference image. The evolution of embedding efficiency and shrinkage with respect to these elements is reflected by Table 1.

Table 1

Reference Image	Resolution	Quality	Message Length	Shrinkage	Efficiency
Ct.	Ct.	Ct.	↗	↗	↘
Ct.	Ct.	↗	Ct.	↘	↗
Ct.	↗	Ct	Ct.	↘	↗

Table 1 presents some intuitive results obtained experimentally for multiple images. In our measurements, the steganographic process was varied by modifying only one element from the following: resolution, quality factor and message length. Efficiency and shrinkage were computed keeping all other elements constant.

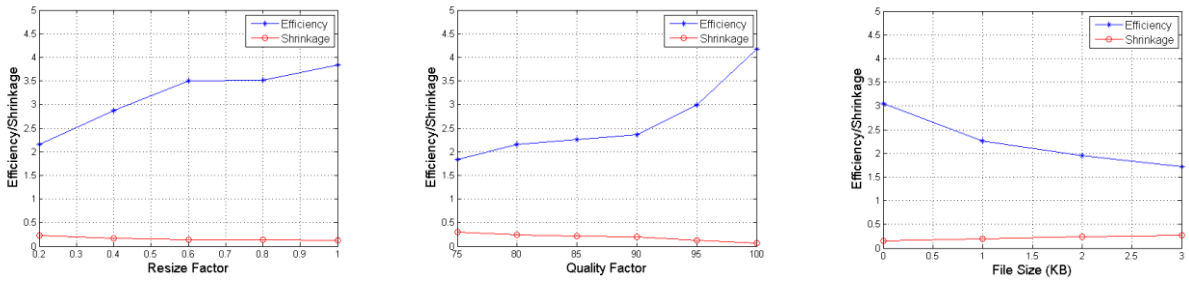
The situation with resolution increasing (line 3, Table 1) is complementary to the situation when message length is increased (line 1, Table 1). When increasing the resolution, the hiding domain is enlarged, thus the same message can be embedded more effectively, leading to a higher efficiency and a smaller shrinkage. In Fig. 5a) we illustrate the embedding efficiency for different resolutions of the image shown in

Fig. 4a). The initial image of resolution 3906×2602 was resized with the following scale factors: 0.2, 0.4, 0.6 and 0.8. A 1Kb data file with random data was used for embedding with C&H. The evolution from Fig. 5a) confirms our hypothesis regarding the increase of efficiency and decrease of shrinkage when increasing the image resolution. Increasing the message size for the same image causes the decrease of embedding efficiency and the increase of the shrinkage. Our claim was confirmed by multiple experiments but we only illustrate in Fig. 5c) the evolution for the case when different message files with sizes 0Kb, 1Kb, 2Kb and 3Kb were hidden in the same cover image of resolution 782×521 from Fig. 4a). Note that embedding efficiency was computed with respect to the length of Steganos EDF and not to the hidden data file, due to message processing (compression and encryption) before the steganographic process, which modifies the input message length. The input data file and the corresponding EDF lengths are given in Table 2.

Table 2

Data File (Kb)	Steganos EDF (Bytes)
0	408
1	1304
2	2264
3	3224

Image quality is another element that affects the evolution of efficiency and shrinkage (line 2, Table 1). The enhancement of image quality (by enlarging the quality factor Q) increases the values of DCT coefficients, which leads to decreasing shrinkage. Higher values of non-zero AC coefficients decrease the probability of making useless modifications during the steganographic process. Fig. 5b) illustrates the evolution of efficiency and shrinkage as function of the quality factor. Our test images were obtained by compressing at different qualities the same image in a raw (uncompressed) format.



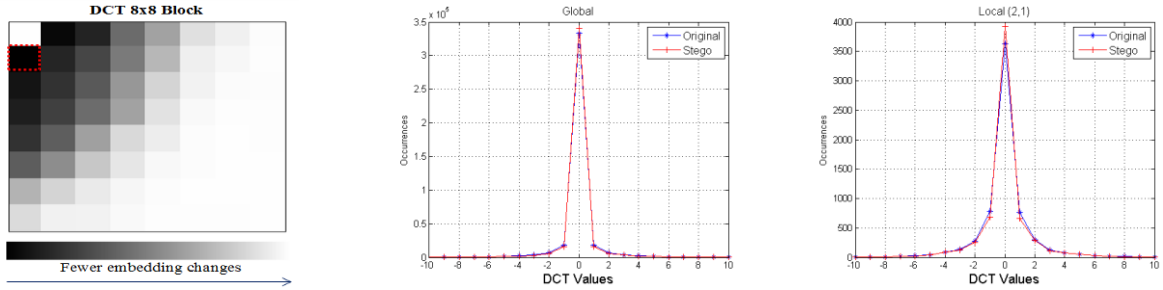
(a) (b) (c)

Fig. 5 – Efficiency and shrinkage:
(a) evolution for different image resolutions; (b) evolution for different quality factors; (c) evolution for different file sizes.

Embedding distribution and DCT histograms. Effectiveness of DCT based steganographic algorithms is directly affected by the choice of both DCT blocks (see Fig. 4b)) and frequencies from these blocks. AC low frequencies, *i.e.* frequencies from the top left side of DCT block (considering the zig-zag scan used in JPEG compression), are generally recommended for the hiding process because of their superior energy values. C&H tool uses low and middle frequencies more frequently than high frequencies. The distribution of steganographic modifications into the DCT block is illustrated in Fig. 6a); such behavior was obtained analyzing a set of 400 cover/stego image pairs, with different hiding parameters (message length, image resolution and image quality).Our discussion is valid for both luminance and chrominance, but the experimental results are illustrated only for the luminance because it hosts most of the secret information and deals with the majority of steganographic modifications (more than 80% in our measurements). Black and dark gray positions from the 8×8 DCT block represent AC coefficients with high modification probability (large number of AC modifications) during the embedding process and coincide with low frequencies of the DCT block. The percentage of information hidden in each of these positions ranges from 4% to 8%. Gray positions are not so often used, representing the middle frequencies, each hosting from 1% to 4% of the message. High frequencies represented by light gray and white are rarely used and hosts less that 1% of the

hidden information. DC term presented in the top left corner is never used in the embedding process and is represented using pure white color.

For all perturbed frequencies, the modification is made by decreasing the absolute value of the frequency with one unity. This technique increases the number of zero AC coefficients and reduces the number of +1 and –1 coefficients. However, this behavior is not highly visible for non-zero coefficients on AC histograms because of partial histogram compensation (when modifying, 1 and –1 become 0, but 2 and –2 become 1 and –1, 3 and –3 become 2 and –2 and so on). A more detailed discussion on histogram compensation is presented by Fridrich et al. in [13]. Global and local (2,1) histograms of AC terms are illustrated in Figs. 6b) and 6c). The message embedding impact is more visible on the local (2,1) histogram, because of a higher modification rate in this frequency (see (2,1) position in Fig. 6a)). Experiments were conducted using the cover image from Fig. 4a) and the 3Kb message file. This type of behavior is exploited and used to detect stego images in the next section.



(a) (b) (c)

Fig. 6 – Message distribution and histograms:
(a) stego changes density on DCT blocks; (b) global histogram on AC coefficients; (c) AC - (2,1) histogram on AC frequencies.

Statistical analysis and classification using CC-PEV image features. Steganalysis of the F5 algorithm was presented by many works in various approaches. Our analysis on C&H tool indicates the use of a similar steganographic technique as the one from F5. The German application restricts the message length to reduce the total number of modifications and hence to lower the detection probability. What we assume is that maximal message length depends on the message encoding and a pseudo-random selection of embedding positions. This is one of the main advantages of this application but is not always enough, as we show in the next paragraphs.

The most powerful steganalysis tests are currently based on image model construction using feature extractors and machine learning techniques used for image classification. This combination of elements created the image classifiers, which are steganography detectors, very effective in determining the image nature. Firstly, relevant image features are extracted from an image set originating from the same source, to construct the image model (corresponding to the image source). Another set with stego images is built and the model of stego images is similarly constructed. Extracted features for both original and stego images are introduced in the classifier for a training procedure. After the training phase, the classifier is able to decide about the nature of unknown images. This step is also known as the test phase. The requirement of using the same image source for both original and stego images is mandatory to achieve the highest accuracy for a given classifier [16]. Generally, the classifier performances depend on: image source, image resolution, hidden message length and extracted features. A detailed description of steganographic classifiers is illustrated in [17], where one of the most effective steganography detectors is also introduced: Ensemble Classifier.

C&H steganalysis was carried out using effective techniques used for the detection of steganographic algorithms. Ensemble Classifier (EC) [16] and a Support Vector Machine (SVM) based library, namely *libsvm* [18], were used to detect carrier images.

The steganalysis carried out is briefly presented in the following steps:

- An image set with original images coming from the same image source is obtained (e.g. photos taken with the same photo camera).
- A stego set is built from the previous set by embedding secret messages with the same length using C&H.

- A *feature extracto*r is used for cover and stego image sets to extract relevant features from every image.
- Feature sets are used by the *classifier* in a *training phase* to construct the image model for both cover and stego images.
- After image model construction, the *testing phase* starts. Original and/or stego images, coming from the same image source, can now be classified in two steps:
  - features are extracted from every image
  - the classifier provides a response concerning the image type (original or stego), using the extracted features; the accuracy of the classifier is measured in terms of Probability Error (PE).

In our experiments we used an original image set, containing 1000 digital images with size 782×521 and quality factor 85. The JPEG quality factor was chosen as a compromise between quality factors used by digital photo cameras (normally between 90 and 100) and quality factors used on WEB or by some photo sharing applications (usually starting from 75).

The stego images were created using C&H for three sensitive data files containing random data generated with Matlab, with sizes: 0Kb, 1Kb and 3Kb. After embedding files in our original set, we obtained three stego image sets. A separate analysis was done for each stego set. Note that some images could not be processed by the C&H tool because of the message length (especially when hiding the 3Kb file). In this case, the cover images providing a mismatch were removed.

To extract features, we used the CC-PEV feature extractor [19, 20], which extracts a feature set (with length 548) for each image, using quantized DCT coefficients; the extracted features are given in Table 3. We chose this feature set due to its proved effectiveness against F5 algorithm, which is using the same distortion function as the one implemented by C&H.

Table 3

Feature	Feature Size
Global histogram	11
5 AC histograms	5 × 11 = 55
11 dual histograms	11 × 9 = 99
Variation	1
Two types of block-ness	2
Co-occurrence matrix	5 × 5 = 25
Markov features	9 × 9 = 81
Total dimensionality:	274
*the feature set is doubled by the same features extracted from the calibrated image(see [20])	

Table 4

Message File(KB)	Image sets number	EC Detector (Average error)	SVM Detector (Average error)
0	999	0.2802	0.3050
1	880	0.0132	0.0134
3	400	0.0010	0

For the train phase we used 80% of the images and for the test phase, the rest of 20%. To illustrate the performances of the two classifiers (EC and SVM) on C&H, Table 4 presents the corresponding accuracy in terms of Average Error at detecting cover/stego images during the test phase.

Errors obtained by the two detectors for CC-PEV features clearly show that stego images created with C&H are detectable in a large percentage, even when inserting an empty file.

6. CONCLUSIONS

This paper presents a practical way of dealing with the steganalysis problem for JPEG images, using only a restricted set of information regarding a public steganographic application, namely Crypt&Hide of the German framework Steganos Privacy Suite. By looking at the experimental results only, we extracted as much information as we could about the cryptographic and steganographic components approached by the steganographic application. We used all the information we could gather during our analysis and we presented some important vulnerabilities of the application, especially on the data hiding module of the application. In this research it was proved that JPEG images processed by the application contain processing traces such as the embedded JFIF marker and the updated modification date. These clues, combined with an effective statistical analysis of the image content using the DCT hiding domain, led to a high detection rate

of the carrier files in our testing scenario. Our analysis was facilitated by numerous similarities between the steganographic application hiding technique and the well-known F5, an already broken steganographic algorithm. The exposed issues should be taken into consideration when using Crypt&Hide application. Also, the hiding technique should be improved in terms of cover image selection, message length and message encoding.

REFERENCES

1. I.J. COX, M.L. MILLTER, J.A. BLOOM, J. FRIDRICH, T. KALKER, *Digital Watermarking and Steganography*, second ed., Morgan Kaufmann, Burlington, USA, 2008.
2. E. HAMILTON, *JPEG File Interchange Format*, Version 1.02, 1992.
3. D. A. KER, *Chrominance subsampling in digital images*, The Pumpkin, **1**, November 2005.
4. G.K. WALLACE, *The JPEG Still Picture Compression Standard*, IEEE Transactions on Consumer Electronics, **38**, *1*, 1992.
5. *Steganos Privacy Suite*. Software and description available at : https://www.steganos.com/data-security/data-security/privacy-suite/features/
6. *Break Our Steganographic Challenge (BOSS*): http://agents.fel.cvut.cz/~pevnak/boss_competition
7. *JPEGsnoop 1.7.3 – JPEG File Decoding Utility*. Software avalilable at: http://www.impulseadventure.com/photo/jpeg-snoop.html
8. D. UPHAM, *Steganographic algorithm JSteg*. Software available at http://zooid.org/~paul/ crypto/jsteg
9. A. WESTFELD, *High Capacity Despite Better Steganalysis (F5–A Steganographic Algorithm)*, Moskowitz, I.S. (eds.): Information Hiding. 4th International Workshop. Lecture Notes in Computer Science, **2137**, Springer-Verlag, Berlin Heidelberg New York pp. 289–302, 2001.
10. N. PROVOS, *Defending against statistical steganalysis*, 10th USENIX Security Symposium, pp. 323–335, Washington, DC, August 13–17, 2001.
11. P. SALLEE, *Model-based methods for steganography and steganalysis*, International Journal of Image Graphics, **5**, *1*, pp 167–190, 2005.
12. J. FRIDRICH, T. PEVNY, J. KODOVSKY, *Statistically undetectable JPEG steganography: Dead ends, challenges, and opportunities*, Proceedings of the 9th ACM Multimedia & Security Workshop, pp. 3–14, Dallas, TX, September 20–21, 2007.
13. J. FRIDRICH, M. GOLJAN, D. HOGEA, *Steganalysis of JPEG Images: Breaking the F5 Algorithm*, Proc. 5th Int'l Workshop Information Hiding, Springer-Verlag, 2002.
14. J. FRIDRICH, M. GOLJAN, D. SOUKAL, *Wet paper codes with improved embedding efficiency*, IEEE Transactions on Information Forensics and Security, **1,** *1*, pp. 102–110, 2006.
15. T. FILLER, J. JUDAS, J. FRIDRICH. *Minimizing Embedding Impact in Steganography using Trellis-Coded Quantization*, Proc. SPIE, Electronic Imaging, Media Forensics and Security XII, San Jose, CA, January 18-20, 2010.
16. J. KODOVSKY, J. FRIDRICH, V. HOLUB, *Ensemble Classifiers for Steganalysis of Digital Media*, IEEE Transactions on Information Forensics and Security, **7**, *2*, pp. 432-444, April 2012.
17. J. KODOVSKY, J. FRIDRICH, *Steganalysis in high dimensions: fusing classifiers built on random subspaces*, Proc. SPIE, Electronic Imaging, Media Watermarking, Security, and Forensics XIII, San Francisco, CA, January 23–26, 2011.
18. C.-C. CHANG, C.-J. LIN, *LIBSVM: a library for support vector machines*, ACM Transactions on Intelligent Systems and Technology, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm
19. T. PEVNY, J. FRIDRICH, *Merging Markov and DCT features for multiclass JPEG steganalysis*, E. J. Delp and P. W. Wong, editors, Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents IX, **6505**, San Jose, CA, January 29–February 1, 2007.
20. J. KODOVSKY, J. FRIDRICH, *Calibration revisited*, in J. Dittmann, S. Craver, and J. Fridrich (editors), Proceedings of the 11th ACM Multimedia and Security Workshop, Princeton, NJ, September 7–8, 2009.
21. W. PRAT, *Digital Image Processing*, John Wiley&Sons, NY, 1978.