AN OVERVIEW ON
HIDING AND DETECTING
STEGO-DATA IN VIDEO STREAMS

ALEXANDRE MIGUEL FERREIRA
Alexandre.MiguelFerreira(at)os3.nl

March 23, 2015
Abstract

As steganography becomes more common today, new techniques to hide data in large amounts of data streams and new challenges come along every day. Video steganography is one of them. Steganalysis algorithms become then more important.

This paper has as goal to offer a critical review of the steganalysis techniques used today, mainly focusing on how they can be applied for (real-time) steganography detection on video streams. This paper also intends to give an overview on how these detection algorithms can be prevented.
# Contents

List of Figures .......................................................... 3

Glossary ......................................................................... 4

1 Introduction ................................................................ 5

2 Background .................................................................. 6
  2.1 What is Steganography? ........................................... 6
      2.1.1 History .............................................................. 6
      2.1.2 Prisoners’ Problem ............................................ 7
      2.1.3 Steganography vs Watermarking ..................... 7
      2.1.4 Steganography vs Cryptography ...................... 7
  2.2 What is Steganalysis? ............................................... 7
      2.2.1 Types of Attacks .............................................. 8
          2.2.1.1 Stego Only Attack .................................... 8
          2.2.1.2 Known Cover Attack ................................ 8
          2.2.1.3 Known Message Attack ............................ 8
          2.2.1.4 Chosen Stego Attack .............................. 8
          2.2.1.5 Chosen Message Attack ......................... 8
      2.2.2 Visual Attacks ................................................. 8
      2.2.3 Structural Attacks ......................................... 9
      2.2.4 Statistical Attacks ....................................... 9
          2.2.4.1 Chi-Square Attack .................................. 9

3 Literature Study ......................................................... 10
  3.1 Steganographic Techniques ....................................... 10
      3.1.1 Injection ......................................................... 10
      3.1.2 Substitution .................................................... 10
          3.1.2.1 List Significant Bits Manipulation ............ 10
      3.1.3 Transform Domain .......................................... 11
          3.1.3.1 Discrete Cosine Transform ..................... 11
          3.1.3.2 Discrete Wavelet Transform ................... 11
  3.2 Video Steganography ............................................... 11
      3.2.1 Spatial Domain ............................................... 12
      3.2.2 Frequency Domain ....................................... 12
  3.3 Video Container Format .......................................... 12
  3.4 Compression ........................................................ 13
      3.4.1 MPEG Compression ....................................... 13
      3.4.2 H.264 Compression ..................................... 14

4 Analysis ...................................................................... 15
  4.1 OpenPuff ................................................................ 15
      4.1.1 OpenPuff Stego-analyzed ............................ 16
          4.1.1.1 Visual Attack ......................................... 16
          4.1.1.2 Statistical Attack .................................. 16
          4.1.1.3 Structural Attack .................................. 18
  4.2 Anti-Forensics ....................................................... 20
      4.2.1 Deniable Steganography ............................... 20

5 Conclusion .................................................................. 21
# List of Figures

2.1 Usual steganographic system \(^1\) ............................................... 6  
2.2 Prisoner’s Problem approach \(^2\) ................................................ 7  
2.3 Visual attack example \(^3\) .......................................................... 8  

3.1 LSB using one least significant bit \(^3\) ........................................ 10  
3.2 Typical structure of a video container \(^4\) ...................................... 12  
3.3 A typical sequence with I, B and P-frames \(^5\) ............................... 13  

4.1 OpenPuff carrier bit encoding \(^6\) .............................................. 15  
4.2 Original file frame ................................................................. 16  
4.3 Stego-file frame ................................................................. 16  
4.4 ent command results of the original file ...................................... 16  
4.5 ent command results of the stego-file ....................................... 17  
4.6 File type header hexdump from the original file ............................ 18  
4.7 File type header hexdump from the stego-file .............................. 18  
4.8 Original file hexdump ............................................................ 18  
4.9 Stego-file hexdump ............................................................... 18  
4.10 Original file MOOV box hexdump ........................................... 19  
4.11 Stego-file MOOV box hexdump ............................................. 19
## Glossary

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCT</td>
<td>Discrete Cosine Transform</td>
</tr>
<tr>
<td>DWT</td>
<td>Discrete Wavelet Transform</td>
</tr>
<tr>
<td>EOF</td>
<td>End of File</td>
</tr>
<tr>
<td>LSB</td>
<td>Least Significant Bit</td>
</tr>
<tr>
<td>MSB</td>
<td>Most Significant Bit</td>
</tr>
<tr>
<td>NFI</td>
<td>Nederlands Forensisch Instituut</td>
</tr>
<tr>
<td>PoV</td>
<td>Pairs of Values</td>
</tr>
<tr>
<td>PRNG</td>
<td>Pseudo Random Number Generator</td>
</tr>
<tr>
<td>QP</td>
<td>Quantization Parameter</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

The rapid increase of information sharing between people causes a variety of security problems. New security breaches are coming on a daily basis occurrence. One of the ways to offer security in information communication is by means of steganography. Steganography is the art and science of hiding one piece of data within another in such a way that the cover data is perceived not to have any embedded message for its unplanned recipients. Its purpose is to make communication undetectable. However, the increase in availability, sophistication and popularity of steganography programs also increases the potential opportunities for crime, being industrial espionage or criminal coordination among them. This is where steganalysis come up. Steganalysis is the mechanism of detecting information hidden using steganography. A steganographic system is hacked when it is understood that the file is carrying secret information.

There are three well-known cover medias: image, audio and video. Video steganography is emerging as a sub-field of digital steganography.

Research question

- Which methods are available for (real-time) steganalysis on a video-stream and how can these be prevented?
  - Which are the steganography methods available for video-stream?
  - Which are the steganalysis methods available for video-stream?
  - How can steganography be prevented on a video-stream?

The approach to this subject was to analyze one of the available stego-tools, namely OpenPuff [8], to conclude if it is possible to do steganalysis on video-streams. Anti-forensics was also considered, i.e. the possibility of avoiding steganalysis. Moreover, Defraser [7] (a tool developed by the NFI) was also assessed to decide whether stego-videos created by OpenPuff can be identified by this tool.
Chapter 2

Background

In this section the meaning of steganography and steganalysis is explained. Also there is a review of the state of the art and previous work done.

2.1 What is Steganography?

Steganography is the art and science of hiding communication or, in other words, the technique of hiding information within a carrier where no one, except the intended recipient, have knowledge of the existence of hidden information. The word originates from the ancient Greek words steganos (covered) and graphein (writing), literally meaning ‘covered writing’ [5].

Figure 2.1 represents a usual steganographic system.

![Usual steganographic system](https://bitsofbinary.wordpress.com/2011/10/18/an-introduction-to-lexical-steganography)

2.1.1 History

The earliest recordings of steganography come from the Greek historian Herodotus [5]. In his Histories, dated back to 440 BC, he recorded two different steganographic techniques used in Greece. The first stated that King Darius of Susa shaved the head of one of his prisoners and tattooed a secret message on his scalp. After the prisoner’s hair grew back he was sent undetected. On the second story, Demaratus needed to send a warning to Sparta about a forthcoming attack to Greece. To do so he wrote the message on a wooden backing of a tablet before applying its wax surface. Also this message was sent undetectable.

During the XV century Johannes Trithemius, in his works Polygraphiae and Steganographia, wrote on steganographic techniques such as invisible inks, coding techniques for text or hidden messages in music.

More recently, during World War II steganography was also used to send hidden messages. Concepts such as null ciphers, image substitution or microdots were introduced during this time.

---

1Image source: https://bitsofbinary.wordpress.com/2011/10/18/an-introduction-to-lexical-steganography
2.1.2 Prisoners’ Problem

This Prisoners’ problem is commonly used as an example of the need of techniques for sending information in a cover way. This was introduced by Simmons in 1983 [11]. It tells the story of 2 prisoners working on an escape plan. Although they are allowed to communicate, their communications pass through the prison’s warden which will attempt to find any hidden communication between them. The prisoners know that the warden will stop the communications if he discovers the secret channel.

The difficulty of the warden’s task depends on the complexity of the steganographic algorithm as well as his prior knowledge.

Figure 2.2 represents the described problem, where Alice and Bob are the prisoners and Wendy the warden.

![Figure 2.2: Prisoner’s Problem approach](http://studentweb.niu.edu/9/Z172699/Organisation.html)

2.1.3 Steganography vs Watermarking

Watermarking is similar to steganography, however there are some key differences. While on steganography the data embedded should be covert and undetectable, on watermarking it does not matter if the hidden information is easy to detect, the important factor any attempt to removing a watermark should result in significant degradation of the quality of the carrier file. Watermarking is commonly used to help trace the origin of files.

2.1.4 Steganography vs Cryptography

Steganography and cryptography are different. Cryptography is the study of the ways in which messages can be sent in a disguised form so that only the intended recipients can read the message. Steganography is often confused with cryptography since both are used to protect confidential information. The difference between the two is that cryptography scrambles a message so it cannot be understood, steganography hides the message so it cannot be seen.

2.2 What is Steganalysis?

The security of a steganographic system is defined by its strength to defeat detection. Steganalysis is the practice of detecting the presence of messages that have been hidden using steganography. Although it is not the goal of steganalysis, ideally the content of the hidden message is also determined.

---

2.2.1 Types of Attacks

Steganalysis attacks can be active or passive. In the active attacks a steganalyst can manipulate the data while in the passive attack the steganalyst is only able to analyze the information without changing it.

The following are types of attacks used by steganalysts to detect steganography on files [10].

2.2.1.1 Stego Only Attack

The attacker has intercepted only the stego data and is able to analyze it. For example, only the stego-carrier (picture, video, etc) and hidden information are available.

2.2.1.2 Known Cover Attack

The attacker has intercepted the stego-file and knows which cover file was used to create this stego-file. This provides an advantage over the stego-only-attack for the attacker. For example, the original image and the image containing the hidden information are available and can be compared.

2.2.1.3 Known Message Attack

The attacker has intercepted the stego-file and knows both the cover file which was used to create the stego-file as well as the message that is embedded in this stego-file. Although the message is known, this attack might be very difficult to perform.

2.2.1.4 Chosen Stego Attack

The algorithm used as well as the stego-carrier are known. For example, the steganography tool, the image and hidden information are known.

2.2.1.5 Chosen Message Attack

The aim of this attack is to find patterns of the stego-object that can point to specific steganography tools or algorithms. To do so, the steganalyst generates stego-objects from some steganography tools or algorithms of a chosen message.

2.2.2 Visual Attacks

Visual attacks are the simplest form of attacking a steganographic system. This attack is based on the visual analysis of the image and if there are noticeable differences between the carrier and stego image, it is probable that the suspected image/video carries hide information. However, if an embedding is not detected by the observer, the bit planes of the image are then analyzed, beginning with the least significant plane.

Figure 2.3: Visual attack example

A successful visual attack shows also how the stego-system operates when embedding the hidden message. If the carrier is not known this attacks becomes very hard to perform.

---

3Image source: http://www.aaronmiller.in/thesis/
2.2.3 Structural Attacks

In a structural attack the steganalyst tries to find known properties of the algorithms used to hide the message. If they contain any properties of these algorithms they are analyzed further. Due to false positives, this attack is used to highlight images which show signs of possible embedded data. As in the visual attacks, the possibility for a structural attack to be successful depends a lot on if the carrier file is known or not.

2.2.4 Statistical Attacks

In statistical attacks an statistical analysis of the images is done using mathematical formulas. Depending on the results it can be determined if there is hidden information on the file or not. Statistical attacks are much more effective than the visual or structural attacks.

2.2.4.1 Chi-Square Attack

The Chi-square attack, developed by Westfield and Pfitzmann [15], is a statistical test to measure if a given set of observed data and an expected set of data are similar or not. In this attack Pairs of Values (PoV) expected frequencies are compared to the observed frequencies of the file being analyzed, this gives the probability of hidden data. PoV are values where if the LSB is changed in one value they are transformed to another value and vice-versa. For example, if a pixel as the value 23 and a bit with value 1 is inserted into the LSB it will be changed to the value 24, the same way, if a bit with value 0 is inserted into the LSB of a pixel with value 24, it will be changed to the value 23.

This attack is successful even without knowing the carrier file, however it fails to determine the hidden data’s size.
Chapter 3

Literature Study

There are various techniques for hiding data in a digital storage file. Although the aim of this project is video steganography, there are also steganography techniques applied to images and audio which are relevant to video file formats. Moreover, video can be divided into audio and image streams. In this chapter there is a discussion on audio and image techniques since it is important to understand them to be able to work with video steganography.

3.1 Steganographic Techniques

There are a collection of steganographic techniques that can be used to camouflage information in a file. The following described are examples of steganographic techniques.

3.1.1 Injection

This method is by far the simplest steganographic technique. Injection involves hiding the message in parts of a file that will be ignored by the application, such as comment tags, hidden form elements or End of File (EOF) markers, without affecting the integrity of the container file.

A drawback to this technique is that it generally makes the file larger than the original unmodified file.

3.1.2 Substitution

Substitution techniques identify areas of a file of least relevance and replace this data with the hidden information. This technique does not modify the size of the container file, therefore the steganographic capacity of the file is limited.

3.1.2.1 List Significant Bits Manipulation

The most common way to embed data in an image is to replace the least significant bits (LSB) [12].

In 8-bit images, each pixel is represented by 8 bits, such as shown on Figure 3.1.

![Figure 3.1: LSB using one least significant bit](http://lvee.org/uploads/abstract_file/file/111/2.png)

The most significant bits (MSB) are the ones to the left and the least significant bits are the ones to the right. While when changing the MSBs there is a noticeable impact on the color, changing the LSBs will not be noticeable to the human eye. Image formats commonly used in the LSB substitution are loss less and the data can be directly molded and recovered.

---

One significant advantage of this method is that it is simple to implement. However, since the data is hidden in the LSBs, these methods are vulnerable to extraction and attacks such as compression or cropping.

**LSB Sequential Insertion**

In sequential insertion, as the name implies, the message is hidden in consecutive parts of the carrier file. This technique is easy to detect and decode, since each change data bit from the message follows the next.

**LSB Pseudo Random Insertion**

In pseudo random insertion a pseudo random number generator (PRNG) is used to randomly hide the secret bits of the message into the LSB of the carrier file. In this technique a secret, known by both the sender and receiver, is used as a seed for a random number generator. This technique is difficult to be attacked both visually and statistically. It is also very hard to extract all the bits hidden within the message, since the data is hidden in a random way making it difficult to assure that the next LSB also contains hidden information.

3.1.3 **Transform Domain**

There are more complex ways of hiding secrets inside images, for instance the modification of discrete cosine transformations. Transform domain techniques are generally used on compressed container files, such as JPEG or MPEG.

3.1.3.1 **Discrete Cosine Transform**

The Discrete Cosine Transform (DCT) algorithm works by using quantization, i.e. rounding the values (for example 3.872687 is 4) of the least important parts of the image in respect to the human visual capabilities. Although doing this to each and every value produces noticeable distortions in the image, the human eye under normal conditions does not detect high frequencies in images, therefore this allows DCT to make larger modifications to these frequencies with little noticeable image distortion. The algorithm works as follows: the image is split into smaller areas (8x8 squares) which will be transformed via DCT. A quantization on the frequencies is then applied. This is the stage where the secret message is injected. Finally the image is compressed, which will not have any impact on the integrity of the secret message.

Using this technique ensures that the hidden message is distributed in an even way through the whole image [6].

3.1.3.2 **Discrete Wavelet Transform**

Although DCT is quite useful for hiding data in a compressed image, it does not well at high compressed levels. This is where the discrete wavelet transform enters. The DWT technique makes it possible to rise the level of robustness of the information being hidden. It works by taking many wavelets to encode a whole image. This allows the image to be compressed highly by storing the high frequency separated from the low frequency details. The low frequency parts are then compressed, being also possible to use quantization for further compression.

Embedding secret information with DWT works pretty much the same way it works with DCT. The drawback of this method is that if the threshold is too high the stego-file will have detectable differences [6].

3.2 **Video Steganography**

Video steganography is growing as a research area, this to the fact that a video container file has diverse advantages not presented by other container formats. To change a video file is somewhat more difficult to detect by the human eye, as frames are visible on-screen for very brief periods of time. Moreover, video file containers are quite larger than audio or images files, hence reducing the problem of steganographic capacity. There are various types of video formats, among them AVI, WMV or MP4.
Since video files are an assortment of images and audio, the techniques used on these can also be applied to video files.

### 3.2.1 Spatial Domain

In the spatial domain the image is dealt as is and the value of the pixels of the image change with respect to the scene (which can be a two- or three-dimensional scene, etc).

The techniques used in the spatial domain are based on direct manipulation of pixels in an image.

### 3.2.2 Frequency Domain

The frequency domain is a space in which each image value at a specific image position represents the amount that the intensity values in the image vary over a specific distance related to that same specific image position. In the frequency domain, changes in the image position correspond to changes in the spatial frequency or the rate at which image intensity values are changing in the spatial domain image.

As an example, imagine that there is the value 30 at the point that represents the frequency 0.1 (or 1 period every 10 pixels). This means that in the corresponding spatial domain image the intensity values vary from dark to light and back to dark over a distance of 10 pixels, and that the contrast between the lightest and darkest is 60 gray levels (2 times 30).

When secret messages are hidden in a video it is commonly classified into temporal domain and spatial domain. The advantage of the spatial domain based methods is that it is simple and easy to implement since the pixel values of the image are directly replaced with the hidden message. The transform domain techniques are more resistant to attacks on the stego-image due to the fact that the secret information is hidden in frequency domain, which makes it very difficult to extract the message [4].

Another advantage of using video files to hide information is that it adds more security against attacks since video files are relatively more complex than image or audio files.

### 3.3 Video Container Format

A video container is usually associated to the file format. In here it is contained the various components of a video, such as the stream of images or the sound. As an example, it is possible to have multiple soundtracks and/or subtitles included in a video file, as long as the container format allows it. Popular containers are OGG, Matroska, AVI or MPEG.

Figure 3.2 shows a typical structure for a video container. No specific format is represent since details might vary from format to format.

![Figure 3.2: Typical structure of a video container](https://msdn.microsoft.com/en-us/library/windows/desktop/ee663601(v=vs.85).aspx)

---

As can be seen in the figure, the video file structure is hierarchical, with the header information appearing at the beginning of the container. This is the typical structure of container formats (most of them). Also the data section contains crossed audio and video packets, which is a common structure in media containers.

### 3.4 Compression

Video compression regards reducing and removing redundant video data. Efficient compression techniques can significantly reduce the file size with no undesirable effects on the visual quality. Different video compression standards utilize different methods of reducing data, therefore results may differ in bit rate, quality and latency.

There are two types of compression, they are lossless compression and lossy compression.

**Lossless Compression**

This technique gives preference to the original image information, meaning that every single bit of data that was originally in the file remains after the file is uncompressed. This allows the original data to be perfectly reconstructed from the compressed data.

**Lossy Compression**

Lossy compression discards the points which are difficult to identify by the human eye. The resulting image is similar to the original image but not exactly the same. This technique reduces a file by permanently eliminating certain information, especially redundant information. When the file is uncompressed, only a part of the original information is still there (however it might not be noticed by the human eye).

Lossy compression is generally used for video and sound, where a certain amount of information loss will not be detected by most users.

### 3.4.1 MPEG Compression

MPEG compression is a lossy compression and, as almost all the lossy compression techniques, it exploits the fact that human senses do not distinguish small changes in high frequency information.

3 bytes are used in video to represent every single pixel. These are either separated in the usual RGB code (red, green and blue) or in luma and two chroma components, named YUV and YCbCr. MPEG also uses the DCT technique (see Section 3.1.3.1) to convert the spatial data into the frequency domain. This is done in 8x8 blocks of pixels. The upper left corner is called the DC value. DC stands for direct current and refers to the average brightness in the block. All the other values describe the variation around the DC value. They are called AC values, or alternating current. The DCT coefficients coming from the conversion are then quantized accordingly to their importance.

An MPEG file is a sequence of three kinds of frames: I-, B- and P-frames, as can be seen on Figure 3.3.

![Figure 3.3: A typical sequence with I, B and P-frames](http://www.axis.com/products/video/about_networkvideo/compression.htm)

---

I-frames

Intra-coded frames are video frames which can be reconstructed without any reference to any other frame. A video file will always start with an I-frame and will have successive I-frames added at regular intervals. The downside to an I-frame is that they are the largest in terms of size as the whole video frame is encoded every time. They are also entirely encoded using DCT values.

P-frames

Predictive-coded frames are video frames which are forwarded predicted from I-frames or P-frames, making it impossible to reconstruct them without the data from either I or P-frames. The downside to P-frames is that they are sensitive to transmission errors because of their dependency on earlier frames.

B-frames

Bidirectionally predictive-coded frames are both forwarded and backward predicted from the last I-frame or P-frame, i.e. they need two other frames to reconstruct them. Although using B-frames improves the prediction and the quality of decoded video it also increases the processing requirements and latency.

On steganography, the DCT values of these frame types may be changed by a limited amount when it is decoded.

3.4.2 H.264 Compression

H.264 compression introduces a new intra-prediction scheme for encoding I-frames. This technique can greatly reduce the bit size of an I-frame and, as I-frames are the largest of the video frames, this has a great impact on the overall size of the video file. H.264 achieves a smaller bit size for the frame while maintaining the quality by enabling the successive prediction of smaller blocks of pixels within each macro block in a frame.

H.264 compression works the following way: a block of residual samples is transformed using a 4x4 or 8x8 integer transform, an approximate form of the DCT. The transform outputs a set of coefficients, each of which is a weighting value for a standard basis pattern. When combined, the weighted basis patterns re-create the block of residual samples.

The output of the transform, a block of transform coefficients, is quantized, i.e. each coefficient is divided by an integer value. Quantization reduces the precision of the transform coefficients according to a quantization parameter (QP). Commonly, the result is a block in which most or all of the coefficients are zero. Setting QP to a high value means that more coefficients are set to zero, resulting in high compression at the expense of poor decoded image quality, oppositely, setting QP to a low value means that more non-zero coefficients remain after quantization, resulting in better decoded image quality but lower compression.

The H.264 baseline profile only uses I- and P-frames. Since B-frames are not used low latency can be achieved. [1]
Chapter 4

Analysis

To understand how steganography is applied on videos, one of the available stego-tools, namely *OpenPuff*, was analyzed.

4.1 OpenPuff

*OpenPuff* is a steganography tool created by Cosimo Oliboni. This tool lets the users to hide information in a wide range of carrier formats, among them videos, such as 3gp, Mp4 or Mpeg II. On top of this, the users can also hide data in more than a single carrier file, forming a carrier chain. For the hidden message to be retrieve, all the carrier files must be provided in the same order used to hide the information.

A successful steganographic system should take 2 important factors into consideration, embedding efficiency and embedding payload. If a steganographic scheme has a high embedding efficiency it means quality of stego data and less amount of the carrier file data changed [3]. As it is obvious, any kind of distortions will catch the attention of a steganalyst. Moreover, the security of a steganographic architecture is directly dependent on the embedding efficiency. [14]. A high embedding payload means that the capacity to hide secret information inside the carrier file is big.

As stated by Neils Provos in [9], a paper where *OpenPuff* is based on, steganalysis resistance and performance are incompatible trade-offs.

To make the system secure and to keep it with good performance *whitening* is used. This will provide higher data security and allows deniable steganography (which will be discussed on section 4.2). However it will require more carrier bits. i.e. the use of this technique requires larger files. On the other hand, to make the system secure and to make it steganalysis resistant *whitening* and *cryptography* are used. It will only assures higher data security with the drawback of marking the carriers more "suspicious", due to their random statistical response.

Therefore *OpenPuff* implements also 3 layers of hidden data obfuscation, being them cryptography, whitening and encoding. Figure 4.1 shows how *OpenPuff* steganographic architecture works.

![OpenPuff carrier bit encoding](https://en.wikipedia.org/wiki/File:OpenPuff_arch8.jpg)

Figure 4.1: *OpenPuff* carrier bit encoding

---

Before the data is carrier injected, it is encrypted and whitened, meaning that part of the hidden information will turn into a big block of pseudorandom suspicious data. Carrier injection will then encode it applying a non-linear covering function [2] which will take also original carrier bits as input.

This way, modified carriers will need less change and, since it will lower their random-like statistical response, it deceives various steganalysis tests [16].

4.1.1 OpenPuff Stego-analyzed

To understand and analyze OpenPuff, the approach followed was to create some stego-videos and perform known attacks (described on Section 2.2).

4.1.1.1 Visual Attack

The visual attack was performed by reproducing both the original and stego videos. Also, individual frames from the original and from the stego-file were compared and analyzed. Figures 4.2 and 4.3 show the same frames collected using Defraser. As can be seen both frames look identical.

![Figure 4.2: Original file frame](image1) ![Figure 4.3: Stego-file frame](image2)

As expected no noticeable differences were found between both files neither while reproducing the videos nor analyzing the individual frames.

This was the only attack performed where Defraser was used. It cannot be determined whether Defraser is a valuable tool to identify stego-videos, however the frames retrieved with this tool proved to be insufficient.

4.1.1.2 Statistical Attack

The statistical analysis of the stego-files created by OpenPuff was done with the program ent [13]. This tool does a collection of tests on the provided file, the output looks like represented on Figures 4.4 and 4.5.

```
Entropy = 7.902275 bits per byte.
Optimum compression would reduce the size of this 205638 byte file by 1 percent.
Chi square distribution for 205638 samples is 57547.28, and randomly would exceed this value 0.01 percent of the times.
Arithmetic mean value of data bytes is 127.0006 (127.5 = random).
Monte Carlo value for Pi is 3.025822076 (error 3.69 percent).
Serial correlation coefficient is 0.147440 (totally uncorrelated = 0.0).
```

Figure 4.4: ent command results of the original file
The following explains the meaning of each of the values output by `ent`.

**Entropy**

The entropy value represents the information density of the contents of the file and it is expressed as the number of bits per character. The result shown on Figure 4.4 indicates that the file is extremely dense in information. Therefore the compression of the files analyzed is not likely to reduce its size, since for both files it was output the value 1%.

**Chi-square Test**

This is the most common used test for the randomness of data and is extremely sensitive to errors in pseudo random sequence generators. It is calculated for the stream of bytes in the file and revealed as an absolute number and a percentage which indicates how frequently a truly random sequence would exceed the value calculated. This percentage is interpreted as the rate to which the sequence tested is suspected to be non-random. If the percentage is:

- **greater than 99% and less than 1%** - the sequence is almost surely not random;
- **between 99% and 95% or between 1% and 5%** - the sequence is considered suspect;
- **between 90% and 95% or between 5% and 10%** - the sequence is not sure to be suspect or not.

Although both files are very dense in information, as shown by the entropy value, it is far from being random, as it is exposed by the chi-square test.

**Arithmetic Mean**

The arithmetic mean is the result of the sum of all the bytes in the file divided by the file length. A random data file shall return a value around 127.5.

Both values are close to a random data file.

**Monte Carlo Value for Pi**

The Monte Carlo Value for Pi is calculated by grouping each successive sequence of six bytes from the file in a 24 bit X and Y coordinate in a square. If the randomly-generated point distance is less than the radius of a circle inscribed within the square, the six bytes sequence is considered a hit. The percentage of hits is then used to calculate the value of Pi. If the sequence is close to random, the value will approach the correct value of Pi.

Although the values presented are not the correct value of Pi, they are not far from it.

**Serial Correlation Coefficient**

The serial correlation coefficient calculates how much each byte in the file depends on the previous byte. If the sequence is random, this value will be close to zero.
Both values are pretty close to zero, meaning the file is quite random on what is related to the dependency of a byte on its predecessor.

By comparing the original file values (Figure 4.4) and the stego-file values (Figure 4.5) provided by ent, it can be seen that the values are very similar and do not raise any suspicious upon the stego-file.

If we approach this attack as a **Known Cover Attack**, and since it is based on the comparison of both the original and stego files, it can be concluded that the known stego-file contains hidden data, therefore this attack can be considered successful. However, if the original file was not known, the attack known as **Stego Only Attack** would fail.

### 4.1.1.3 Structural Attack

The structural attack is based on the comparison of the original file and the stego-file (it can either be considered a **Known Cover Attack** or a **Chosen Message Attack**).

To perform this attack an hexdump of both files was analyzed. Figures 4.6 and 4.7 represent the file type header of both files.

**Figure 4.6: File type header hexdump from the original file**

**Figure 4.7: File type header hexdump from the stego-file**

As can be seen highlighted in blue, the last four bytes of the header are changed. These bytes are an offset pointing to the beginning of the header that belongs to the MOOV box. The MOOV box defines the timescale, duration, display characteristics of the movie, as well as sub-boxes containing information for each track in the movie.

The stego-file MOOV box header offset is different from the original file MOOV box header because some bytes were inserted outside this box, as can be seen on Figures 4.8 and 4.9. This pattern is followed through out the stego-file outside the MOOV box.

**Figure 4.8: Original file hexdump**

**Figure 4.9: Stego-file hexdump**
Although it could not be proved, these bytes might be related to the size of the file being hidden, as well as the password(s) used to encrypt the message. This assumption is made based on [9], where it is stated that 32 state bits are hidden, 16 bits for a seed and 16 bits for an integer containing the length of the message being hidden.

It is important to notice that since the video container format may change, the optimal location of the moov box will depend on the selected delivery method. This way the MOOV box can come right away after the file header, implicating the MOOV box header offset to remain the same for both the original and the stego-file (since the bytes are being inserted only outside this box).

After the 32 state bits are hidden, the secret information is hidden inside the carrier file. In order to hide this information 2 steps are followed:

**Identification of redundant bits**

Redundant bits are bits that can be changed without noticeably degrading the carrier medium. These redundant bits are dependent on the specific output file.

**Selection of bits to hide information**

The selection of the redundant bits that will be used to hide the information is done by choosing a maximum of 50% of the redundant bits available. This is done for two specific reasons, to give the selection process a chance to find a better embedding (one which will make less changes to the carrier medium) and to preserve the frequency count base statistics.

While analyzing in detail the MOOV box, it was noticed that the bytes were modified. Figures 4.10 and 4.11 show the differences between both the original and the stego-file.

![Figure 4.10: Original file MOOV box hexdump](image)

![Figure 4.11: Stego-file MOOV box hexdump](image)

Once again, although it was not possible to prove the secret information is being hidden inside the MOOV box, it is believed this is the actual behavior.

The impossibility to determine whether the bytes being inserted are related to the size of the file being hidden, as well as the password(s) used to encrypt or not, and the impossibility to determine if the message and the bytes being modified are related to the secret information are due to two reasons: the fact that the secret information is encrypted and the use of deniable steganography techniques (see Section 4.2.1).
4.2 Anti-Forensics

Anti-forensics pursuits to make the analysis and/or examination of evidence difficult or impossible to conduct and relies on several weaknesses of the forensic process, such as the human element or the dependency on tools. Encryption and steganography are among the ways to make it successful.

However, there is always the chance of being detected using these techniques. Resisting to these unpredictable attacks is also possible, even when the user is forced (by legal or physical coercion) to provide a valid password to extract the data.

4.2.1 Deniable Steganography

As in cryptography, also in steganography there is the possibility for deniable steganography. Deniable steganography is a camouflage based technique that, even if the steganalyst is able to state that data is being hidden, allows the breaker to convincingly deny that fact.

*OpenPuff* implements deniable steganography by allowing the user to hide two different messages in the cover file. One which contains the sensitive data and one which although is plausible to be considered sensitive, the user us willingly to give away.

This method is one of the reasons why the statistical attacks are ineffective.
Chapter 5

Conclusion

The purpose of this work is to identify the available methods to do steganography on videos, as well as steganalysis. Also the available techniques to avoid steganalysis was discussed.

Throughout this project different steganographic techniques were presented and, from the literature, it can be concluded that techniques used on images and audio can also be applied to videos. The most common techniques used the spacial domain (LSB) and the frequency domain (DCT).

Even though steganography is usually undetectable by the human eye, the use of statistical analysis can reveal the presence of hidden data. However, as could be seen throughout this paper, detecting hidden data, being it done in images or videos, is a difficult process to carry out, whether the carrier files are known or not. Even if the carrier files are known it cannot be guaranteed that the hidden information can be determined and recovered. With the use of the correct techniques, hidden information tends to be nearly impossible to be undetectable. And, if it was not enough the fact that steganography is already quiet difficult to detect, new techniques, such as deniable steganography, are made available to the users.

Therefore, the best way to prevent steganography would be to alter or destroy files which are considered suspicious. The introduction of new video compression methods where less redundant bits are available is also a possibility.

Although steganography is becoming more advanced it is still a science that is not well-known.

Future Work

The attacks performed against OpenPuff during this project proved to be insufficient to determine whether there is hidden data inside a file, if the carrier file is not know. However, when the cover data is known, the analysis of the original and stego-files raise some suspicion. Although the hidden information could not be determined.

As a future project it would be interesting to assess if the hidden information can be retrieved.
I would like to thank prof. dr. Zeno Geradts from NFI for the wise counseling and time taken while my supervisor in this project, as well as to dr. Arno Bakker from UvA for his valuable input and feedback.

I would also like to thanks to my family and friends for the support given during the time this project took place.
Bibliography


