****

**What a Digital Forensics Investigator Should**

**Know About Steganalysis of Digital Content**

**Year 3 Computer & network Forensics**

**Submitted by:**

**James Finglas – B00094138**

**Word count: 4152**

**Submission date**

**24/11/2018**

Contents

[Abstract 3](#_Toc530386059)

[Introduction 3](#_Toc530386060)

[Steganography/Steganalysis Injection.Dection Examples 5](#_Toc530386060)

[Steganalysis Approaches 7](#_Toc530386061)

[Chain of Custody and Documentation of Methodology 8](#_Toc530386062)

[Steganography Signatures 9](#_Toc530386063)

[Raw Steganography 9](#_Toc530386064)

[Xiao Steganography 2.6.1 11](#_Toc530386065)

[Hide’N’Send 16](#_Toc530386066)

[rSteg 20](#_Toc530386067)

[OpenStego 23](#_Toc530386068)

[Conclusions 26](#_Toc530386069)

[References 27](#_Toc530386070)

# Abstract

Steganalysis has become of interest to law enforcement services, the legal professions, intelligence agencies and the corporate world. With the rise of digital child pornography websites, terrorism, corporate espionage not to mention a growing intelligence in ordinary criminal circles it is important for a digital forensic investigator to be able to place some context to Steganalysis.  
  
This report covered a brief history of Steganalysis as well as the importance of Steganalysis in modern digital forensics and what the investigator should know about it. Some examples of how Steganalysis were carried out in an order to display methods of identification of Steganographic signatures.  
  
The report will cover a broad analysis of Steganalysis principals rather than specifics in order give as comprehensive an overview as possible.

# Introduction

The investigator has the responsibility of determining if any files on the object or data stream he or she has been asked to examine might contain a hidden message, so how would hide such a message and how would the investigator go about discovering it?

Steganography has been around for hundreds of years stretching back to the times of the ancient Greeks. Leonardo Da Vinci himself employed some earliest forms of both steganography and cryptography, as did Galileo. So how is this relevant to Steganalysis? Well, one must ask a simple question. Since it is known that church was widely aware of the activities of Galileo, and deeply suspicious of Da Vinci why then did the church not discover the hidden writings of these masters?  
  
The answer is simple. To be able to decipher the hidden messages one must be aware that somebody might be trying to hide a message in the first place. One must also be aware of the methods employed and have such an understanding as to be able to uncover and/or decipher the message. Without knowledge of these things, one might simply find themselves looking for a needle in a haystack that may not even be there.  
  
As human technology developed in the modern age and Steganography moved into the digital age, the ability to identify that a message may be hidden grew more difficult. While it was still possible to extract messages hidden using digital images, it was common practice for each image to have be analysed with a view to every file within a given disk image having the possibility of containing a hidden message. Naturally, in the face of the vast number of files typically contained on a hard disk just in Operating System files alone, this could be an extremely daunting task taking a vast amount of time and resources.

In the early 90’s **[1]** a new branch of research began to emerge that aimed at reducing this problem by developing methods of identifying the use of Steganography. The goal of the research was to analyse as many methods of Steganography as possible to identify the signatures left by the application of Steganographic programs.

This is Steganalysis. The ability to identify the application of Steganography to a file via signatures. The

advantages of this is that a file can quickly be identified as having a hidden message. This process can also be automated.

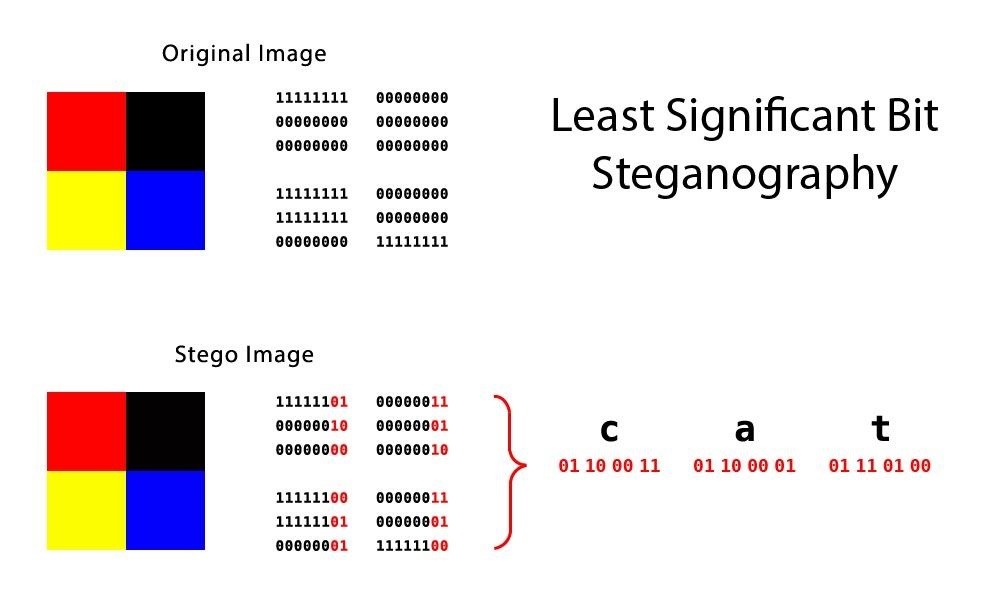
Steganalysis can be broken down into two sub categories.

* Passive Steganalysis: This branch, the main branch of steganalysis is concerned only with the identification of a hidden message.  
    
  This area is principally used in the law enforcement arena, and the corporate business arena; with the retrieval of the message being passed on to other team members.
* Active Steganalysis: The secondary branch, a relatively new area of steganalysis moves beyond detection; and concerns itself with retrieval and even modification of a hidden message.  
    
  This area is principally used in the areas of both intelligence agencies and corporate espionage. An intelligence agency may for example, implant a tracking algorithm r malware program to track targets of investigations. A Corporation may implant false data to obfuscate the information a competitor was attempting to steal.

In summary, Steganalysis is an attempt to defeat the very purpose of Steganography by reducing the ability to hide a message by offering methods of rapid identification of a hidden message. This is merely the first step in this area of digital forensics, but it is by its very nature, the most important in the area of steganography. After all, if a hidden message cannot be located, it cannot be revealed.

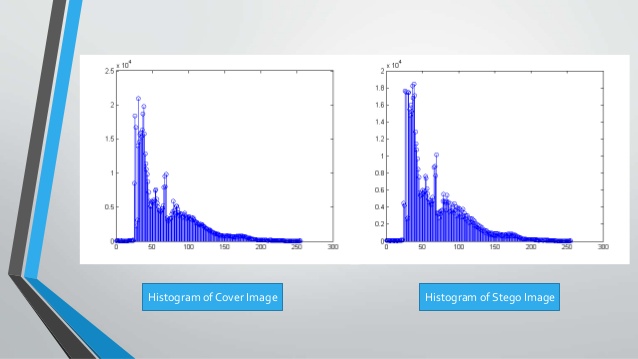
An investigator beginning a Steganalysis must therefore be aware of certain things when examining a file. A file may or may not contain hidden data, but it is the signatures he or she is looking for, NOT the data. The data may have been encrypted before being inserted into the carrier file, therefore if active Steganalysis is being pursued a working knowledge of cryptanalysis may be required. A hidden message may have been obfuscated by noise or have extra irrelevant data added to make a steganalysis more difficult or time consuming. One does not need to retrieve the data to consider a passive Steganalysis of a file a success, one only needs to prove a message exists. **[2]**  
Steganography/Steganalysis Injection/Detection Examples

There are many methods of injecting a stego object into a carrier file. There is Inner and outer loop injection, LSB injection, bright/dark pixel injection. A file may be disguised to appear as something else. A files extension may be broken to make it unreadable, and Noise injection.  
  
There are many ways of detecting a stego objects injection, such as wave form analysis, noise detection, least significant bit examination, spread spectrum analysis, visual analysis histogram analysis. The list is quite extensive. Here we will examine one injection type, and one detection method.



*Here we see an example of least significant bit injection*

Above, we see an example of least significant bit injection, one of the most common forms of steganography injection. The bit of the stego object are broken up and stolen in the least significant bits of the bytes that make up the file.



*Here we see an example of steganalysis via histogram analysis.*

Here we see an example of a histogram comparison between the original carrier file in its un altered state, and the new file with the stego object injected. As we can see, in the areas where the file has been altered it creates a vast differential in the peaks and curves.  
  
What we should notice from these examples is a recurring theme. In each case. We must have previous knowledge of a message being hidden and be in possession of the original unaltered carrier file to perform the analysis.

# Steganalysis Approaches

There are two principal approaches to analysing a file or data stream suspected of being a carrier file. The first type, is known as the “Blind” approach. In this case, nothing is known about the type of Steganography deployed on the suspect carrier file. This method is very thorough however it is very time consuming. It is a systemic attempt to manually Identify known signature types in an effort to break the Steganography. It is here, in the blind approach that the investigator is most likely to encounter many of the draw backs of Steganalysis as previously mentioned.  
  
The second and more commonly used approach, is known as the “Analytical” approach. The approach was developed principally by the ‘Steganographic Analysis and research Centre” (SARC), who have more recently been incorporated into BackboneSecuruity.com. The analytical approach was developed by storing a large database consisting of almost every known Steganographic program and the signatures they created on a carrier file. This allowed for rapid signature detection.  
To date the SARC database contains 960+ applications and their associated known signature types. It is commonly used by both national (US) and international government and intelligence agencies and private sectors. Repositories also include the world’s largest set of known Hashes applicable to Steganography. **[3]**The SARC project has led to the development of one of the most comprehensive File analysing programs. Not to mention real time data stream detection and signature analysis which of course aims at not only recognizing current signatures but newly burgeoning signatures. SARC has also developed a comprehensive and internationally recognized training certification program. This is highly beneficial in a field of expertise whose value is only now beginning to be fully recognized. **[4]**It I interesting to note the N.I.S.T. (The National Institute for Science and technology) who play a lead role in all things science related, including Digital based, take a back Seat in the creation of tools to fight Steganography. In fact, if one were to google ‘NIST Steganalysis’ you would find that they recommend two of the stools developed by S.A.R.C. **[5]**There are many new approaches to Steganalysis being explored every day, so many they far exceed to scope of this report. But there is one that deserves a brief mention. Audio Steganalysis. With the recent rise in popularity of Podcasting as a form of media, not to mention the popularity of the MP3 file format. It was inevitable that audio steganography use would rise. Using complex waveform, inner loop, noise increase, block size variance and several other methods this method aims at detecting audio Steganography. Two major benefits of audio Steganography are that the message can be broken up and stored in multiple pieces and that there are several injection types available in an audio file. **[6]**

If one employs the Blind method to a Steganalysis of a file this can be further broken down into one of six basic types of Steganalysis attack **[7]**:

1. **Stego only attack**: In this case only the suspect data stream or file object is available for analysis
2. **Known message attack**: The original virgin (Pre-Steganography file) carrier file is available as well as the Stego object. A direct comparison can be performed to detect abnormalities.
3. **Known message attack**: The hidden message and the corresponding carrier file are known. The type of attack may not seem useful at first hand since everything is provided for a full analysis, however this does help establish known patterns and hashes for future analysis.
4. **Known Stego attack**: Both virgin and carrier files are available as well as the type of steganography being known.
5. **Chosen Stego Attack**:The Steganography program and carrier file are known.
6. **Chosen Message attack**: In this case, the investigator will use Steganographic tools to create Stego objects, to compare the resulting signatures against suspect data streams or carrier files.

# Chain of Custody and Documentation of Methodology

It cannot be underestimated the importance of maintaining a proper chain of custody of evidence, secure facilities; a well-maintained lab and a thorough procedure for documenting each step taken in the analysis of any file or data stream. This is the back-bone of any investigation. One can easily envision a case where a lengthy investigation is undertaken, and crucial evidence is found. However, the defence asks one simple question. “Did you document your work?”. If even one step has not been documented, doubt can be introduced.

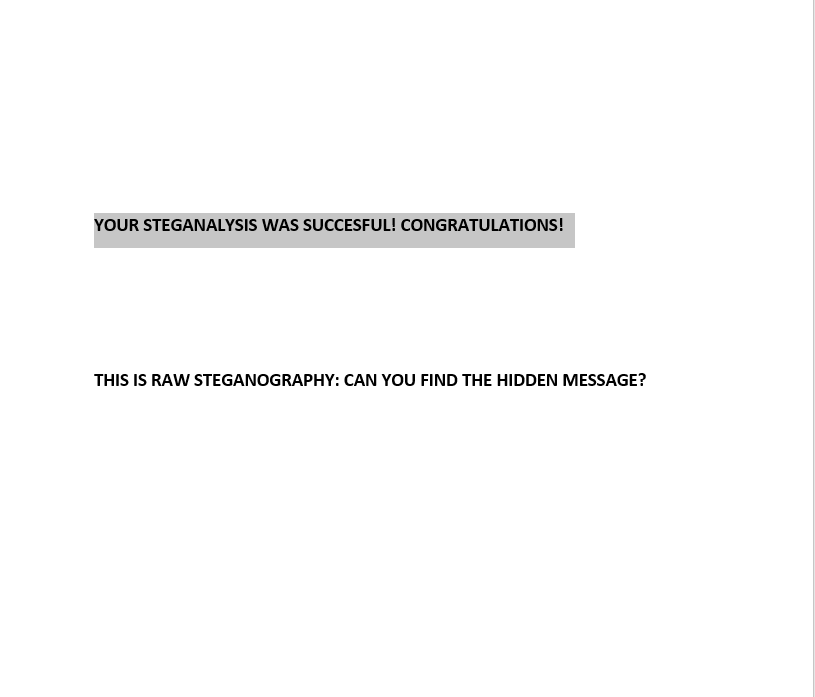
# Steganography Signatures

## Raw Steganography

Raw steganography is perhaps one of the oldest forms of digital steganography. However, when executed with care and skill it can be one of the most effective methods of message obfuscation.  
  
For in Raw Steganography there is no signature change, so no comparison is possible, as nothing is implanted into the file.

  
  
Here we see an example of raw steganography.

Above we see an example of raw steganography. Since no signature change occurs, only a visual examination can be performed. However, even visually, we see nothing to give us reason to believe that a message may have been hidden. The investigator must be prepared to think in very old-school terms here. How could someone hide a message, without altering the files MD5, hex value or histogram?  
  
The answer is very simple!



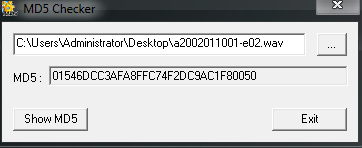
Here we see the message revealed!

Here we see the message revealed simply be scanning the document for text with the colour set to white. This many seem laughable to some, but this method is successful more often than not and this same method is used to trick students studying security courses as well participants in some of the worlds hardest security capture the flag events. In fact, this method has been used against our own institutions capture the flag team, who did discover the steganography but only after hours of exhausting many tools to try to break algorithm used… when none had been used.

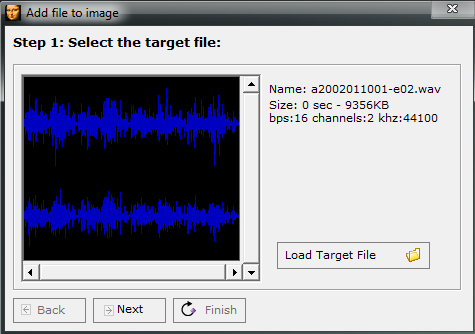
*Here we see a much more advanced example with binary data hidden directly on the picture in very tiny text.*

## Xiao Steganography 2.6.1

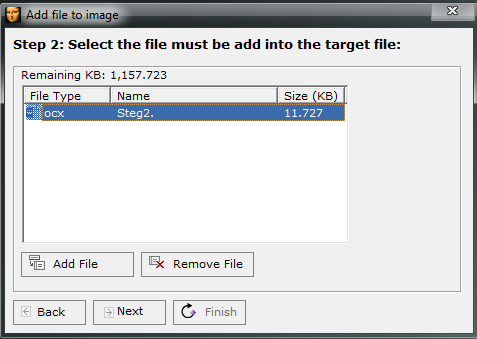
This is a popular program used to embed steganography into image and wave files. The program uses only the least significant bit(LSB) method as other audio formats apply only to audio. For the sake of file type cross compatibility between .img and .wave files is maintained through the restriction to the LSB injection method.



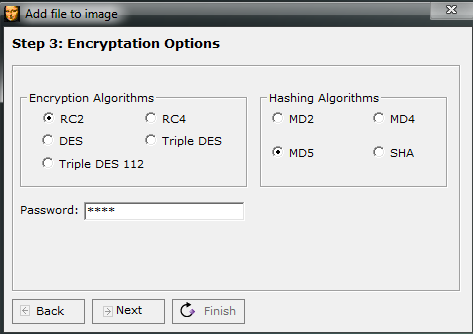
*here we have the MD5 HASH of our original wave file (a2002011001-e02.wave)*



*Here we have selected our target (carrier/virgin) file*

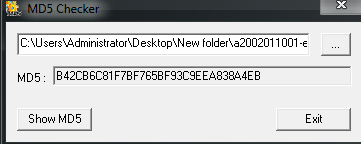
**

*Here we have selected our target (Stego Object) file.*

**

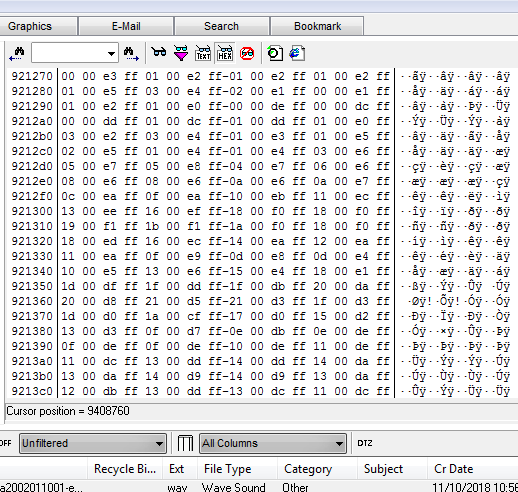
*Here, see the available algorithmic options, as well as the various hash option followed by a password option.*Clicking next at this stage completes the process and provides us with a completed Stego Object, which I have placed in a new folder and given the same name as the original file. Now begins our attempt to analyse the file.

Our step is to compare MD5 Hashes. If we do not see a matching MD5 this tells us immediately we have a suspect file.

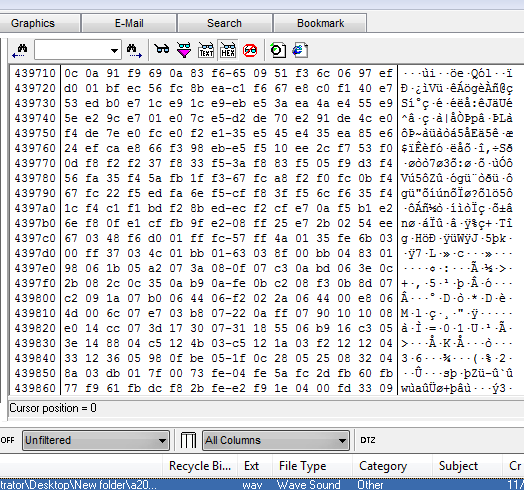


*MD5 HASH of suspect file.*

As seen above, our MD5 HASH values do not match, so we have identified this file warrants further investigation. Our next step is to compare hex values of the files, starting first with our original wave file.

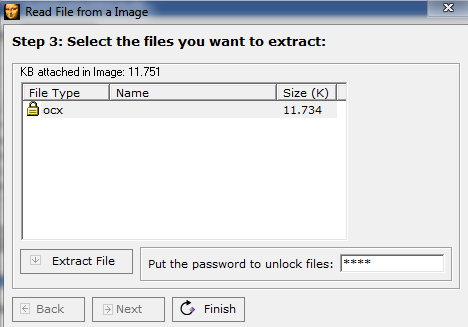


*Here we see the hex values of Original file, and as you can see, the hex structure of the file is very specific.*



*Here we see the hex values of the stego object inserted into the carrier file.*

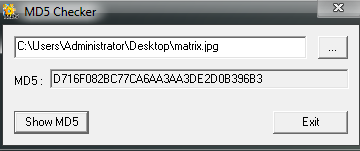
So, what we see here, is that we have two .wav files of the same name. Both work fine. However, the MD5 Hash values do not match, nor do the hex values. This would be considered a successful steganalysis as we have identified that a message likely hides inside this message. However, while we have identified that a message is hidden, we have not identified the exact method of steganography. This would need to be done during the retrieval process, if an active investigation was taking place.  
  
In the case of this example i was aware of the Steganography program used. Therefore, I simply reversed the process. One thing was interested. The file was embed using a OCX file type. When extracting the image, I needed to know the original file name and file extension. I also needed to apply the password. In this case the file name/extension was ‘steg1.txt’ and the password was ‘steg’.  
This led to the file being successfully retrieved.

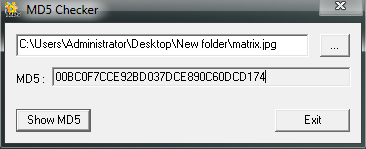


*Here we see the process being reversed*

## Hide’N’Send

Here we have a program very similar to Xiao Steganography. Again, we simply select the carrier image file in which we wish to embed our Stego object text file. Once the hide option is selected, an option to add a password is offered and the object is hidden.

  
  
*Here we see the original MD5 hash of the original carrier file.*

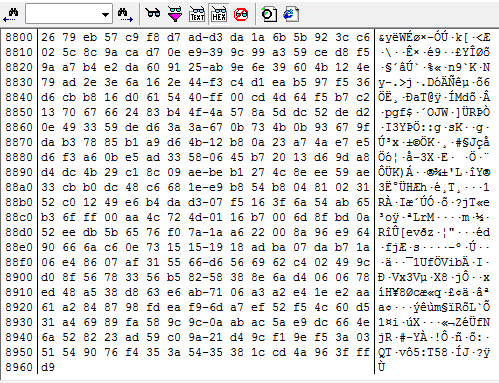


*Here we see the MD5 of the carrier file with stego object inserted.*

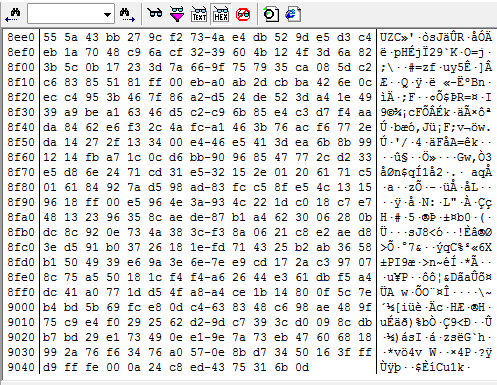


*Here we see the Stego object being imbedded into the carrier file*

In this case you will notice in the image above, the steganography has been encrypted. This will alter the signature of the new file thus making the detection of the hidden message more difficult. You may notice a different hash algorithm was selected. While this will of course also change the signature from that of a similar image hashed with an md5 algorithm it should be noted that the database of known signatures and hashes is not quite vast and while these kinds of variations might fool an investigator in a blind analysis they are far less likely to fool an automated tool with access to the latest hash and signature databases.

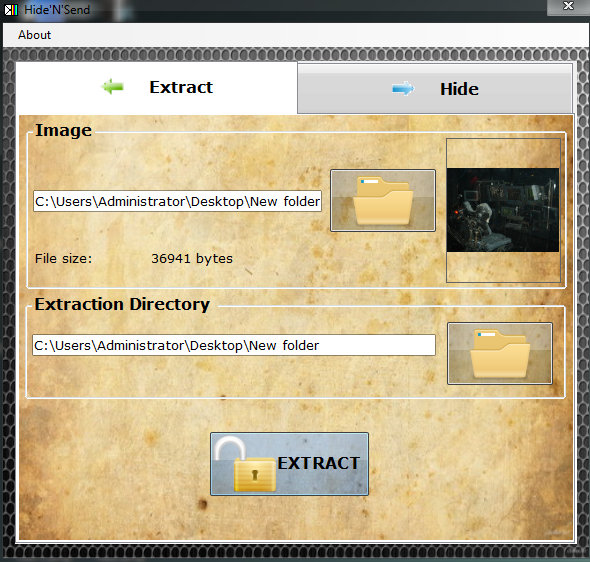


*Here we see the hex view of the original unaltered carrier file.*

**

*Here we see the hex file of the newly created stego object.*

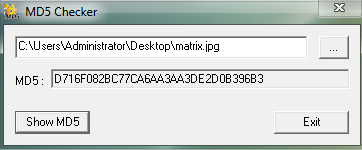
As seen above, the newly created stego object has Both a variance in the MD% hash, and the hex values when compared to the original carrier file. We have achieved another successful detection of a hidden message. Furthermore, we have begun to establish a pattern in how many common steganography programs will appear when begin analysed. However, we must not get complacent. Remember there are a vast array of uncommon programs available, and signatures will not always be easily spotted.

  
*Here we see the process being reversed and the stego object being extracted.*

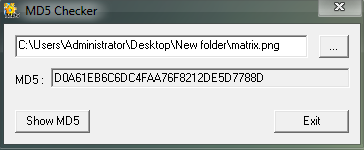
The reversal process here was also very similar, with one quality of life difference. During the extraction It was not necessary to rename the original stego object with its original file name. this was done automatically. This would be a useful feature as it removes the need for the user to know these details in advance and adds extra ease of use appeal to the program.

## rSteg

This is an interesting program which was designed by students. No information is provided about the algorithms used hashing methods or if encryption is used, however, as it is a simple .jar file, no installation is necessary. It allows for text to be directly imbedded into the LSB rather than a file. This is quite clever. While this will of course still change both the MD5 signature, it is likely to change it far less.



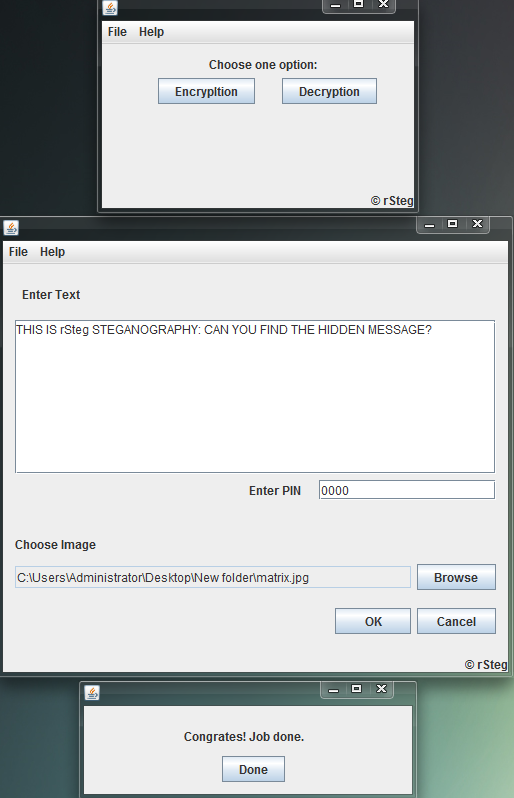
*Here we see the original MD5 hash of the carrier file.*

**

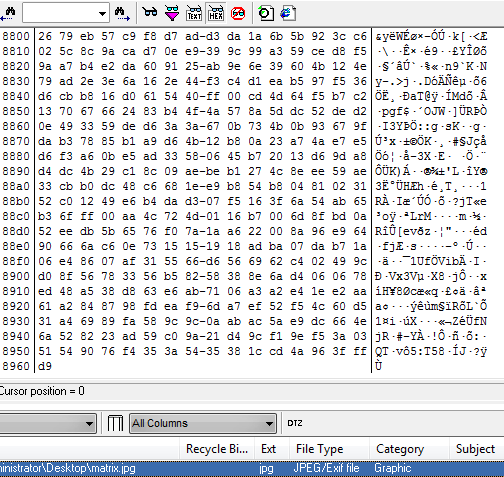
*Here we see the MD5 of the new Stego object file (notice the extension has changed).*

Interestingly, here we see that the procedure has follow the process we have come to expect, even we do not know the exact algorithms and hash types being used. We are also presented a with an option to rename the new stego object file. However, we see our first major deviation. The file extension has changed from .jpg to .png.   
  
Already from the perspective we can consider this a valid signature, however one must consider how this might appear if we did not have access to the information we have such as program used, original file and MD5, knowledge of a hidden message and the contents of said message. We would have, a file, with a different extension, different file name, different MD5 hash, and signature.

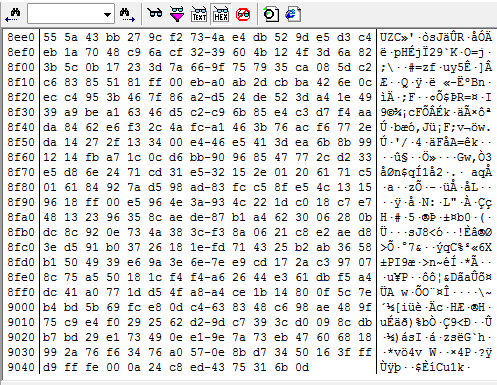
What we see here is that the change of file extension, while simple in this case, adds a very nice extra layer of obfuscation that would make a blind visual attack on the steganography very difficult indeed.



*Here we see the entire process of rSteg’s injection of text into the carrier file.*



*Here we see the hex value of the original carrier file.*

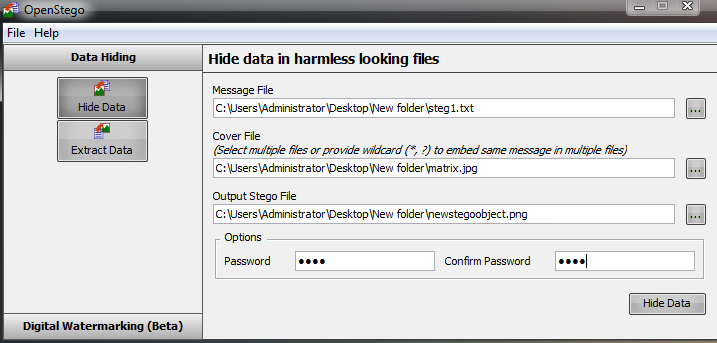
**

*Here we see the hex values of the new stego object (With file extension change).*

While we do see the expected hex value changes, we would expect to see these merely due to the different extension. Here we see the difficulty added by the file extension change. Where previously we could rely on the hex values alone when comparing two files of identical image, extension and MD5 we are now placed in a situation where we suspect a hidden message, but no singular method has provided definitive proof of a message. A blind approach would perhaps fail here, where an analytical method might succeed.

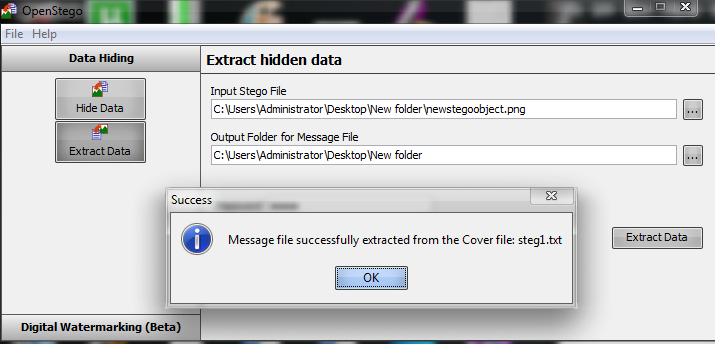
## OpenStego

This program introduces a new feature not yet encountered. The ability to inject the stego object into multiple files at once. This is rather interesting. Consider a folder of images found on a drive, where the same message has been implanted into every image altering the signatures in the exact same way. Would this make it more likely to pass a blind analysis. Or to pose the question another way, if one is looking for a tree in a forest, would one notice the entire forest is made of that exact tree? Certainly, it is an interesting extra layer of obfuscation adding multiple possibilities, even if one simple hide one thousand messages, with only one being the real message and the others being red herrings.

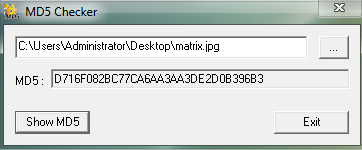


*Here we see the OpenStego injection of the message file into the carrier file. (again, please note the file extension change). Please also note the ability to select multiple cover(carrier) files.*

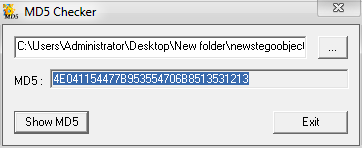
Again, we can see that the file extension has changed. We can also note that digital water marking is employed here. This is a curios choice. Digital watermarking, much as the name suggests, is a method whereby a signature, be it visual, hex or audio is added to a file to allow for identification of the file. This is our first instance of seeing a signature addition that will decrease the effectiveness of steganography once the watermark is commonly known without randomization of the watermarking methodology. In this case, no information is readily available about the methodology of OpenStego’s watermarking.



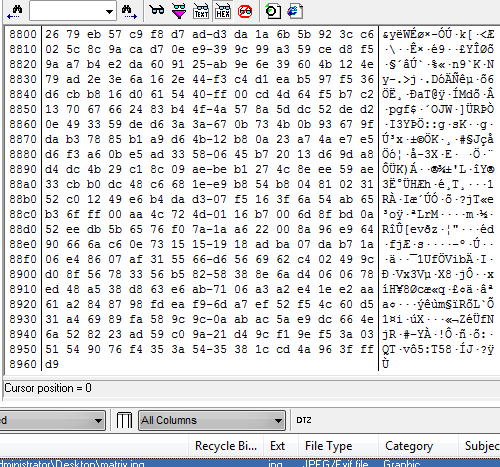
Here we see the OpenStego process reversed with the stego object extracted.



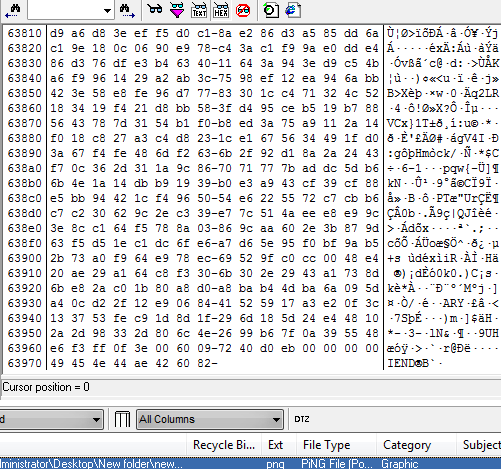
*Here we see the original MD5 hash of the carrier file.*

**

*Here we see the MD5 hash of the new stego object.*



*Here we see the hex values of the un altered carrier file.*

**

*Here we see the hex values of the new stego object*

Again, we see MD5 values, hex values and file signature have changed and we can consider this another successful signature identification and thus, a successful steganalysis.

# Conclusions

There are many different forms of steganographic injection methods. For example, you can hide stego objects on the inner and outer loops of an mp3 file. You may alter an image to appear as audio file. Arrayed against this is a rapidly growing list of detection methods.

* Histogram
* LSB
* Waveform analysis
* Visual
* Spectrum Analysis
* Noise detection

The sheer scale of the numbers of programs and methods and signatures far exceeds the scope of this report, but we can begin to gain a sense of the magnitude of the field of investigative research and why this field is likely to continue to grow.  
  
What an investigator should know therefore, is that we must rely on our tools in this field. In order to even gain the experience needed to perform a blind analysis on a file, which may take years; we must use tools to begin to build up our skills at spotting signatures.  
  
The volume of files contained in most imaged disk will prevent a manual inspection in most cases. And suspicious files which fail an automated test can always be escalated for a blind analysis. The analytical too based approach will be an investigators principal form of analysis.  
  
While standard steganalysis is only concerned with investigating whether a message exists, active analysis may require the message to be retrieved. For this, a working knowledge of cryptography and/or cryptanalysis may be needed.  
  
Because of the ever-changing nature and growth in the numbers of signatures and Hashes related to steganography, an investigator should ways make sure his or her tools are as up to date as possible. This is also doubly important due the sheer complexity involved with the varying types of injection or imbedding. We have learned that a blind analysis is extremely difficult, so as we build up our skill set, knowledge and experience we must assure that we are working with the most accurate and reliable algorithms and databases as possible. Finally, as part of the ongoing research that goes into becoming a good steganalyst, an investigator should be actively seeking to download and familiarize him or herself with the methods used, and signatures generated with said tools.

# References

1. Computer and information science magazine, Image Analysis for online Dynamic Steganography Detection, Kevin Curran, Joanne Mc Devitt, Intelligent Systems Research Centre. Published at ww.ccsenet.org/journal html, August 2008.  
     
   3.Steganalysis.  
     
   https://pdfs.semanticscholar.org/d2a4/8b4087ec079167a4e7497c7c44cb0c122ead.pdf
2. An Analytical Approach to Steganalysis, James E. Wingate, Chad W. Davis, Forensic Focus, July 10, 2011.

https://articles.forensicfocus.com/2011/07/10/an-analytical-approach-to-steganalysis/

1. Back Bone Security, who incorporated SARC.

https://www.backbonesecurity.com/Database960Applications.aspx

Original SARC blog, http://sarc-wv.blogspot.com/

1. StegAlyzerAS, StegAlyzerSS, StegAlyzerRTS & Certified Steganography Examiner Training.  
   https://www.backbonesecurity.com/SARC.aspx
2. https://toolcatalog.nist.gov/populated\_taxonomy/index.php?all\_tools=refine&ff\_id=13&1%5B%5D=any
3. Comprehensive Review of Audio Steganalysis Methods, Hamzeh Ghasemzadeh, Mohammed H. Kayvanrad, !9 January 2017. https://arxiv.org/abs/1701.05611
4. Paladion, May 2015, https://www.paladion.net/blogs/steganalysis