Reconstructive Steganalysis by Source Bytes Lead Digit Distribution Examination

A. Zaharis[1], A. Martini[2], T. Tryfonas[3], C. Illioudis[4] and G Pangalos[5]

[1]University of Thessaly, Greece
[2]SIEMENS SA, Greece
[3]University of Bristol, UK
[4]ATEI of Thessaloniki, Greece
[5]Aristotle University of Thessaloniki, Greece
e-mail: alexzaharis@gmail.com, dimart@gmail.com, theo.tryfonas@bristol.ac.uk, iliou@it.teithe.gr; pangalos@auth.gr

Abstract

This paper presents a novel method of JPEG image steganalysis. Our approach is driven by the need for a quick and accurate identification of stego-carriers from a collection of files of different formats, where there is no knowledge of the steganography algorithm used, nor previous database of suspect carrier files created. The suspicious image is analysed in order to identify the encoding algorithm while various meta-data is retrieved. An image file is then reconstructed in order to be used as a measure of comparison. A generalisation of the basic principles of Benford's Law distribution is applied on both the suspicious and the reconstructed image file in order to decide whether the target is a stego-carrier. We demonstrate the effectiveness of our technique with a steganalytic tool that can blindly detect the use of JPHide/JPseek/JPHSWin, Camouflage and Invisible Secrets. Experimental results show that our steganalysis scheme is able to efficiently detect the use of different steganography algorithms without the use of a time consuming training step, even if the embedding data rate is very low. The accuracy of our detector is independent of the payload. The method described can be generalised in order to be used for the detection of different type images which act as stego-carriers.

Keywords

Steganalysis, Image Reconstruction, JPEG, Benford's Law, lead digit distribution.

1. Introduction

Data hiding has always been a major part of Computer Forensics, as many cases have been solved because of hidden data retrieved by experts. Data hiding in an information system can be performed for various reasons including potential malware attacks, hiding data for later use in a compromised environment by an attacker, exchanging secret information via the Internet, or when an offender hides useful information in his personal computer.

There are numerous methods that can be used in order to hide data from potential interception. One of them is steganography (Anderson et al. 1998; Kessler 2004) on image files, which is a common technique among personal computer users. This technique has been well described, and is well known to forensic investigators. Different tools have been developed to computerize the process of locating suspect carrier files of different file types using visual, protocol compatibility or statistic analysis attacks (Fridrich and Goljan 2002). Most of these techniques concentrate and actually work against specific steganography algorithms/tools. While others that are used for universal blind steganalysis need a training step for agents to be more efficient in locating statistic anomalies on carrier files (Barbier et al. 2007) These techniques are of great performance when the training step includes a large number of true positive carrier files to be examined but can be very time consuming. On the other hand the above techniques mentioned have low hit rate for no training step.

In order to speed up the process of steganalysis without sacrificing high detection rates, we are going to present a less common technique of detecting image steganography carrier files. Our method concentrates on reconstructing (Nosratinia 2001) a reconstructed 'original' image in order to use it as a comparison measure against the original possibly stego-carrier file.

This paper deals with:

- Benford's Law, along with the reasons of choosing this kind of metric as a detection schema.
- The presentation of the process of creating a reconstructed image, resembling the data structure of the original image file before embedding any hidden data in it.
- The design and usage of a forensic tool utilizing the above mentioned technique to blindly detect image carrier files via a single comparison of file structure and not via a time-consuming training step of a decision agent.
- Finally, hit ratio results are presented along with time analysis of the detection process in order to prove the unique use of this steganalytic tool.

The contribution of this paper to the Forensics community concentrates on the presentation of an out of the box steganalytic technique that minimizes computation time along with the creation of forensic tool that implements a well known statistical analysis method (Benford 1938). This tool can be extended in order to be applicable to other image file types while complying with the known computer forensic policies.

2. Steganography Concepts and Tools

We selected a variety of steganography algorithms and tool implementations that hide information inside different parts of JPEG image files. In particular we focus on JPHSWin's (Latham 1999) Least Significant Bit (LSB) option, Invisible Secrets' Fuse method (2011) as well as Camouflage's Fuse method (2011), in order to examine our approach's ability to detect suspicious files in different hiding locations inside the JPEG format.

In our work we are going to distinguish four types of image files:

- The original file, which in our case would be a JPEG image file created/saved with MS Paint.
- The carrier file, which in our case is going to be the result of steganography applied on the original file with either, JPHSWin, Camouflage or Invisible Secrets.
- The reconstructed original file, which in our case is going to be generated by our tool in order to simulate the original MS Paint JPEG file.
- The suspect file, which can be either an Original file or a Carrier file. Our purpose is to identify its exact type.

A crucial point of this work is to generate a reconstructed image file as similar as possible to the original file. To do so, we initially have to identify the software or encoding algorithm of the suspect JPEG image (EXIF Make). We are therefore going to use an open source image analysis tool, namely "JPEGSnoop" (2011), which can determine the various settings that were used by the digital camera when taking the photo (EXIF metadata, IPTC), but can also extract information that indicates the quality and nature of the JPEG image compression used by the camera while saving the file.

Moreover, one of the features of JPEGsnoop, is an internal database that compares an image against a large number of compression signatures. JPEGsnoop reports what digital camera or software was likely used to generate the image. This is extremely useful in determining whether or not a photo has been edited / tampered in any way. This type of feature is sometimes referred to as Digital Image Ballistics / Forensics.

In our efforts to recreate a JPEG encoding algorithm (JPEG 2011, JPEGClub 2011) we chose to simulate the MS Paint software algorithm using JAVA. Thus a tool has been created that acted partially as an Ms Paint Simulator to use as our cloning machine in an attempt to recreate an almost identical original file (reconstructed file) when given a suspect file. This software can be easily extended in order to simulate different types of JPEG encoding implementations.

3. Benford's Law

Benford's law, also known as the first digit law or significant digit law, is an empirical law. It was first discovered by Newcomb in 1881 and rediscovered by Benford in 1938. It states that the probability distribution of the first digits, x (x = 1, 2,..., 9), in a set of natural numbers is logarithmic. More specifically, if a data set satisfies Benford's law, its significant digits will have the following distribution:

$$p(x) = \log 10\ (1 + 1/x),\ x = 1,2,\ldots,9 \qquad (1)$$

Where p(x) stands for probability of x.

Although counterintuitive, validity of Benford's law has been demonstrated in various domains. While the naturally generated data obey the Benford's law well, deliberately altered data and random guess data do not follow this law in general. This property has been widely used in fraud detection in accounting area (Buck et al. 1993, Nigrini, 1996). However, the applications of Benford's law in the image processing field have only recently been explored.The most common reason to use Benford's Law in digital image Forensics is for identifying whether the most significant digits of the block-DCT coefficients follow Benford's law and determine possible carrier files (Hernandez et al. 2000; Jolion 2001; Acebo & Sbert 2005; Perez-Gonzalez et al. 2007). We, on the other hand, are going to use a Generalisation of Benford's Law basic principles ("GBL") against byte array statistics of the file examined.

4. A Generalisation of Benford's Law

Steganography on a JPEG image can cause major alteration throughout the byte array structure of the file depending on the steganography algorithm used. Therefore an initial idea was to use Benford's Law to detect such alterations on byte values of a suspect file. We found however that Bendford's Law cannot be applied conclusively on the byte arrays of a file and no safe conclusions can be made as far as a suspect file is a stego-carrier.

We noted however that on a byte array sequence of a file, it is important to detect changes on the least and most significant digit of each byte, as those are mostly affected by popular methods of steganography. Differences on the rest of the digits are of minor importance and can be omitted. We are going to use this generalisation of Benford's Law (GBL) in an algorithm to detect deliberate modifications due to steganography. This method can differentiate between bit alteration as resulted through the purposeful process of Steganography and unavoidable bit alteration resulted by image reconstruction.

4.1. Proposed Algorithm

Given a Suspect Image file F:

1. Calculate its byte array sequence F[file.length()].
2. Calculate the probability x=1,2,…,9 of the first and the last digit of the byte array sequence F[file.length()] for the Suspect File.
3. Calculate the probability x=1,2,…,9 of the first and the last digit of the byte array sequence F[file.length()] for the reconstructed 'original' file.
4. Compare the results of step 2-3 and using a predefined threshold, decide whether the suspect file is a carrier file or not.

The procedure of the determination of the similarity threshold, used in step 4 is in described in detail in Section V (Detection Algorithm Section).

It was observed that by applying GBL on the byte arrays of a suspect and a reconstructed file the GBL results are crucially different if examining a true stego-carrier and its reconstructed file as opposed to examining an original file and its reconstructed file. Thus GBL can be a valuable metric to detect such anomalies leading to determining if a file is a true stego-carrier.

4.2. Advantages against other Detection Algorithms

The advantages of the applying GBL as a steganography detection algorithm are:

- High detection rate (Par VII, Figure 8)
- Fast detection, only needed digits of a byte array sequence are examined minimizing detection time. No previous database of suspicious files, or training step is needed.
- Extensible algorithm, the GBL algorithm can be used in order to detect steganography inside different image formats other than JPEG. GBL is not a format-dependent steganography detection algorithm as described on Section VI.

5. Reconstruction of Original JPEG Image

In order to design a quick and efficient way of reconstructing an original JPEG image from a suspicious one, the use of known image processing tools along with Java image libraries were utilized. For the purposes of our research, original JPEG images were created with MS Paint and were used to prove the concept of our method.

The process of steganalysis that is going to be described deals with suspect files, about which the steganalyst:

1. Doesn’t know whether the file is a stego-carrier file or not.

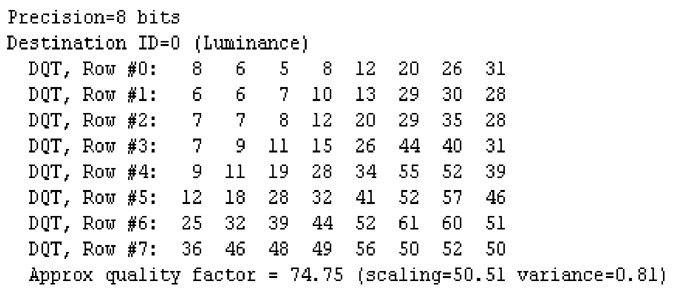
2. Has no information about the hidden message or file type or the steganography algorithm used to embed that message, provided the file is a stego-carrier file.

5.1. Step 1 – JPEGSnoop

Given a suspect file < S >, the process of reconstruction begins by obtaining useful information about the file after examining it with the JPEGSnoop tool. The information obtained by JPEGSnoop is:

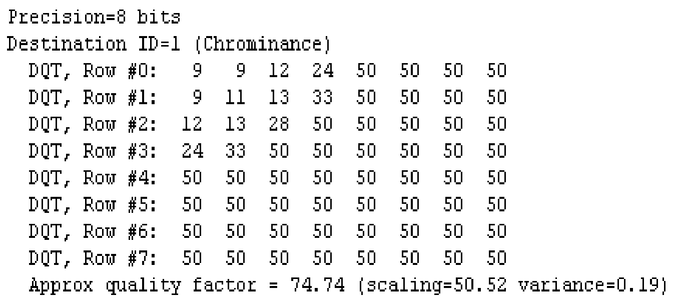
1. The Quantization Table (Quality Factor)
2. The “EXIF Make” or Software Signature that created the picture examined.

It is important to highlight the fact that all three steganography tools examined (JPHSWin, Camouflage, Invisible Secrets) do not re-encode the JPEG image. Parts of the image are altered in order to embed information but the EXIF Make or Software Signature does not change at all. Taking the above mentioned aspect into consideration, one can safely use the JPEGSnoop tool to identify the software that created the original File even if the only file in hand was the suspect file. Knowing the software that created the original file helps in deciding on how to alter the encoding algorithm in order to create a recinstructed file that resembles the original file as close as possible.



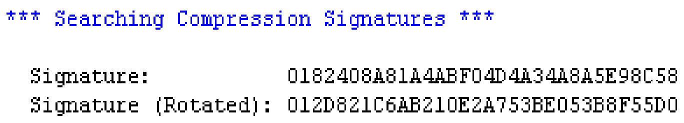
```
Precision=8 bits
Destination ID=0 (Luminance)
  DQT, Row #0:   8   6   5   8  12  20  26  31
  DQT, Row #1:   6   6   7  10  13  29  30  28
  DQT, Row #2:   7   7   8  12  20  29  35  28
  DQT, Row #3:   7   9  11  15  26  44  40  31
  DQT, Row #4:   9  11  19  28  34  55  52  39
  DQT, Row #5:  12  18  28  32  41  52  57  46
  DQT, Row #6:  25  32  39  44  52  61  60  51
  DQT, Row #7:  36  46  48  49  56  50  52  50
  Approx quality factor = 74.75 (scaling=50.51 variance=0.81)
```

Figure 1: Luminance Quality Factor



```
Precision=8 bits
Destination ID=1 (Chrominance)
  DQT, Row #0:   9   9  12  24  50  50  50  50
  DQT, Row #1:   9  11  13  33  50  50  50  50
  DQT, Row #2:  12  13  28  50  50  50  50  50
  DQT, Row #3:  24  33  50  50  50  50  50  50
  DQT, Row #4:  50  50  50  50  50  50  50  50
  DQT, Row #5:  50  50  50  50  50  50  50  50
  DQT, Row #6:  50  50  50  50  50  50  50  50
  DQT, Row #7:  50  50  50  50  50  50  50  50
  Approx quality factor = 74.74 (scaling=50.52 variance=0.19)
```

Figure 2: Chrominance Quality Factor



```
*** Searching Compression Signatures ***

  Signature:            0182408A81A4ABF04D4A34A8A5E98C58
  Signature (Rotated):  012D821C6AB210E2A753BE053B8F55D0
```

Figure 3: The MS Paint Signature

When the Software Signature (in our case MS Paint) and quantization table are extracted, we proceed to step 2.

5.2. Step 2 – Reconstructive Image Encoding

In this step our goal is to achieve a simulation of the software JPEG encoding system (MS Paint) extracted on Step 1. We have already specified on the previous step the quantization tables used and we can assume that the Huffman tables and header syntax are specific for every JPEG file created by specific software. The only other part missing is the actual source / image RGB / YCC values which are going to be used as an input to our JPEG encoding system.

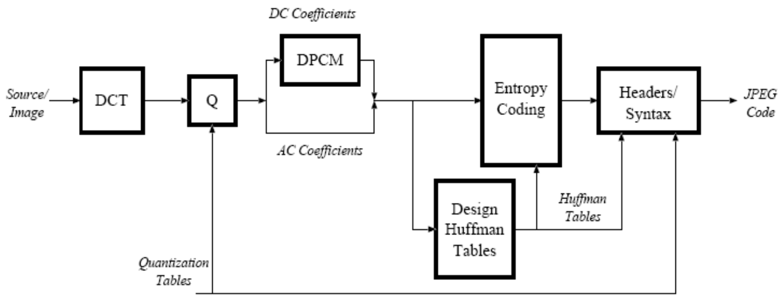


Figure 4: The JPEG Encoding System (Nosratinia 2001)

Different methods have been developed in order to retrieve RGB / YCC values (MATLAB 2011) with the best possible precision in order for the reconstruction to be lossless. We are going to use a less accurate method of Image RGB / YCC data retrieval by utilizing the JAVA Advance Imaging API (JAI). By using standard methods one can retrieve the wanted RGB /YCC values with great precision and minimum time loss.

Using those data as an input to our Dynamic JPEG encoding system, also programmed using JAVA, a reconstructed 'original' JPEG image <P> is created. Step 2 concludes the process of a reconstructed image construction. The output of this process will be used in our GBL in order to detect whether the suspect file used on Step 1 is actually a carrier file.

6. Proposed Steganalysis Method and Detection Algorithm

6.1. The Steganalysis Method

Given a Suspect file <S> :

Step 1 – Data Retrieval

1. The quantization table (quality factor) is extracted from <S>.
2. The "EXIF Make" or Software Signature that created the picture is identified.
3. Headers are extracted from <S>.
4. RGB / YCC values are extracted from <S> using JAVA Advanced Imaging API.

Step 2 – Image Reconstruction

1. Retrieved software signature of Section 1 is used in order to specify the structure of the image file to be reconstructed.
2. According to the structure identified above the following dynamic parameters of JPEG encoding system are set :
    1. Quantization tables
    2. Huffman tables
    3. Headers
3. Reconstructed image file <P> is encoded.

Step 3 – The Detection Algorithm

1. Generalised Benford's Law is applied on Image <S> as described on paragraph III.
2. Image size and hash value of Image <S> is calculated.
3. Generalised Benford's Law is applied on Image <P> as described on paragraph III.
4. Image size and hash value of Image <P> is calculated.
5. Comparing results of Steps 1 & 3 Section 3, updating similarity factor.
6. Comparing results of Steps 2 & 4 Section 3, updating similarity factor.
7. `if (similarity_factor> similarity_ threshold){ "Suspect File <S> is not Carrier File" } else {"Suspect File <S> is Carrier File" }`

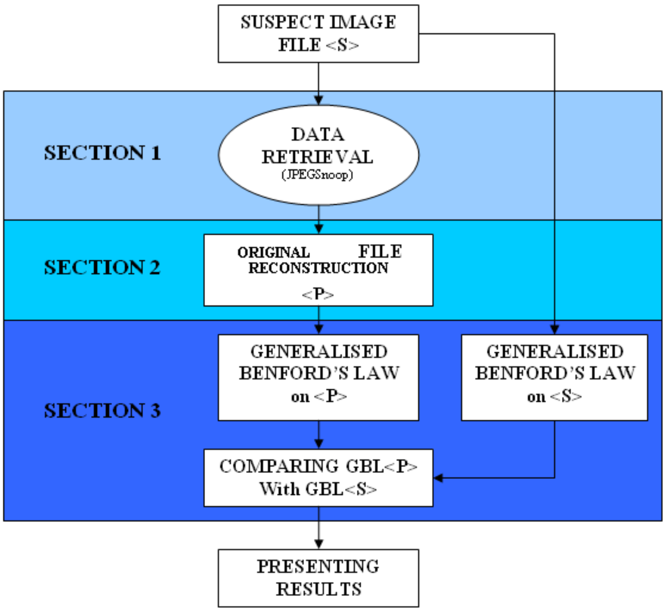

SUSPECT IMAGE FILE <S>
SECTION 1
DATA RETRIEVAL (JPEGSnoop)
SECTION 2
ORIGINAL FILE RECONSTRUCTION <P>
SECTION 3
GENERALISED BENFORD'S LAW on <P>
GENERALISED BENFORD'S LAW on <S>
COMPARING GBL<P> With GBL<S>
PRESENTING RESULTS

Figure 5: Three Steps of the Proposed Steganalysis Method.

Similarity Factor is a parameter that indicates how similar two files are. Similarity Factor values range from zero (0) to nine (9) which is the highest degree of similarity. Its value is calculated by the following algorithm for LSB steganography:

```
Similarity_factor=0;
for (int i=1;i<=9;i++){
if(Math.abs(<S>GBL(i) - <P>GBL(i))<=2.5)
Similarity_factor++;
}

if(Math.abs(<S>.size()-<P>.size())<=800){
Similarity_factor++;
}
```

While for Fuse steganography the following algorithm is used:

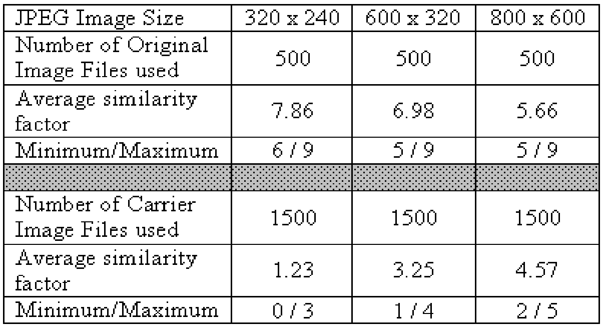
```
Similarity_factor=9;
Size_dif=Math.abs(<S>.size()-<P>.size());

for (int i=1;i<=9;i++){
if(Math.abs(<S>GBLp(i) - <P>GBLp(i))<=15)
Similarity_factor++;
 }

if(Math.abs(<S>.size()-<P>.size())<=800){
Similarity_factor++;
}
```

<S>GBL(x) stands for Generalized Benford's Law on file data distribution for the first digit and last digit with value x where x = 1,2…9.

<S>GBLp(x) stands for Generalized Benford's Law on part of the file data distribution for the first digit and last digit with value x where x = 1,2…9.The size of the part examined is equal to: Size_dif=Math.abs(<S>.size()-<P>.size());

Similarity Threshold is a predefined constant value that has been calculated after applying Generalised Benford's Law on large number of Reconstructed Files created by the previously stated process applied on original image files. This value is encoding specific, so MS Paint has a certain Similarity Threshold while Photoshop 9 has a different one.

After thorough examination of a large number of JPEG image files it has been identified that the Similarity Threshold for MS Paint has a value ~five (~5).The following table depicts the experimental results in scale of 1500 JPEG image files of different dimensions and size.



JPEG Image Size	320 x 240	600 x 320	800 x 600
Number of Original Image Files used	500	500	500
Average similarity factor	7.86	6.98	5.66
Minimum/Maximum	6 / 9	5 / 9	5 / 9
▨	▨	▨	▨
Number of Carrier Image Files used	1500	1500	1500
Average similarity factor	1.23	3.25	4.57
Minimum/Maximum	0 / 3	1 / 4	2 / 5

Figure 6: Statistics of Average / Minimum / Maximum Similarity Factor.

Stego-Carrier Files were divided into three different groups of five hundred (total 1500) Original Files each. For each group JPHSWin, Camouflage and Invisible Secrets were used in order to embed the smallest in size, file possible (1kb ASCII .txt file). As it is depicted in Figure 6, a similarity factor of five (5) can be considered as a similarity threshold.

An amount of 1.5% false positive stego-carrier files was identified but this percentage decreased when the embedded file became larger (5KB ASCII .txt file).Further techniques to minimize the false positive identifications is discussed in Section VI.

The proposed method can be extended in order to detect steganography not only on JPEG image files, as described above, but also on a variety of other common image formats (BMP, TIFF, PNG).

7. A GBL STEGANALYSIS TOOL

The GBL Steganalysis Tool is going to utilize the above mentioned detection algorithm along with some Stego-tool specific detection methods in order to achieve high hit rate in minimum time. This steganalysis tool is formally named Ben-4D.

The diagram of figure 7 displays the basic methods implemented by BEN-4D tool. The core of the detection procedure used is the actual steganalysis method presented on Paragraph V. Once the suspect file has been identified as a stego-carrier file extra detection methods are applied in order to specify the steganography algorithm used. Ben-4D searches for specific patterns and signatures used by each individual steganography algorithm.

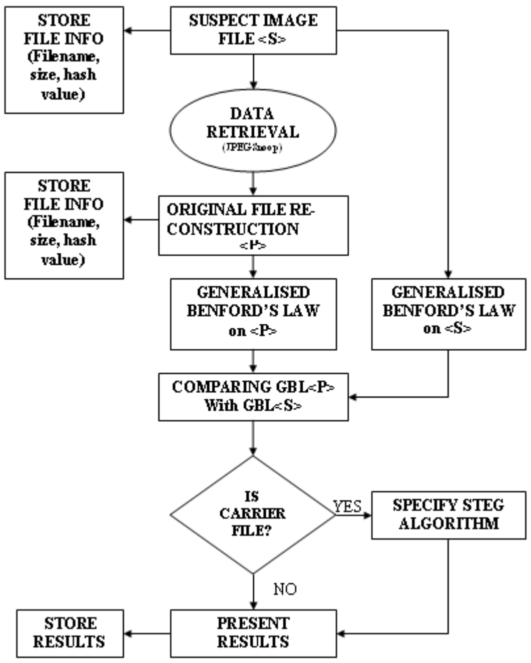

STORE FILE INFO (Filename, size, hash value)
SUSPECT IMAGE FILE <S>
DATA RETRIEVAL (JPEGSnoop)
STORE FILE INFO (Filename, size, hash value)
ORIGINAL FILE RE-CONSTRUCTION <P>
GENERALISED BENFORD'S LAW on <P>
GENERALISED BENFORD'S LAW on <S>
COMPARING GBL<P> With GBL<S>
IS CARRIER FILE?
YES
SPECIFY STEG ALGORITHM
NO
STORE RESULTS
PRESENT RESULTS

Figure 7: BEN-4D's execution flow diagram.

Stego-only attack: Only the stego-object is available for analysis. For example, only the stego-carrier and hidden information are available.

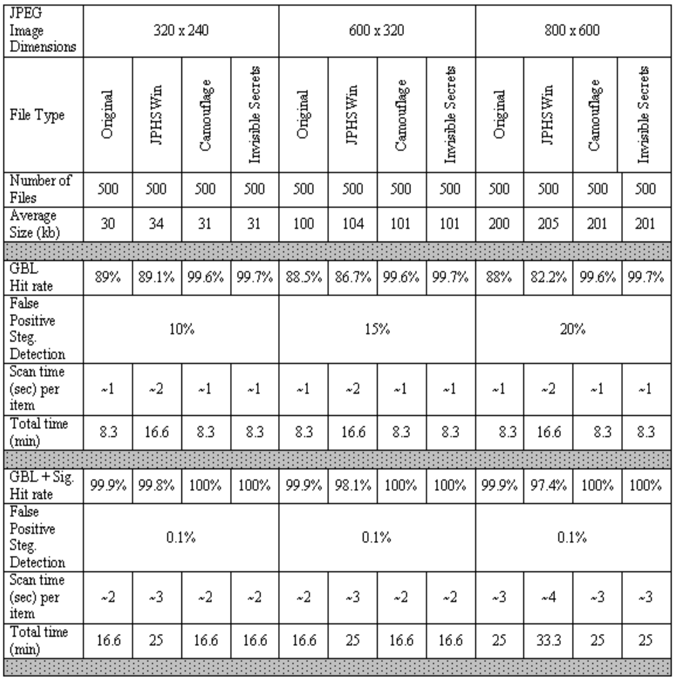
Example Steg- Algorithm specific Detection

Some detection fingerprints used to identify the existence of a specific steganography algorithm are:

1. No standard Huffman tables used (JPHSWin).
2. Noticeable difference in size, between Carrier file and Reconstructed file (Camouflage,Invisible Secrets).
3. Not standard headers (Invisible Secrets).
4. Altered bits following a specific pattern.
5. Not proprietary file termination (Camouflage).

A combination of the Generalized Benford's Law detection along with some of the detection fingerprints stated above can increase the detection hit rate of carrier files as well as decrease the False – Positive detection, mentioned on a previous paragraph, from an average 15% to 0.1 %.

Figure 8, represents the hit rates obtained using only Generalized Benford's Law detection algorithm in a comparison to using a combination of Generalized Benford's Law and other detection fingerprints.



JPEG Image Dimensions	320 x 240				600 x 320				800 x 600			
File Type	Original	JPHS Win	Camouflage	Invisible Secrets	Original	JPHS Win	Camouflage	Invisible Secrets	Original	JPHS Win	Camouflage	Invisible Secrets
Number of Files	500	500	500	500	500	500	500	500	500	500	500	500
Average Size (kb)	30	34	31	31	100	104	101	101	200	205	201	201
GBL Hit rate	89%	89.1%	99.6%	99.7%	88.5%	86.7%	99.6%	99.7%	88%	82.2%	99.6%	99.7%
False Positive Steg. Detection	10%				15%				20%			
Scan time (sec) per item	~1	~2	~1	~1	~1	~2	~1	~1	~1	~2	~1	~1
Total time (min)	8.3	16.6	8.3	8.3	8.3	16.6	8.3	8.3	8.3	16.6	8.3	8.3
GBL + Sig. Hit rate	99.9%	99.8%	100%	100%	99.9%	98.1%	100%	100%	99.9%	97.4%	100%	100%
False Positive Steg. Detection	0.1%				0.1%				0.1%			
Scan time (sec) per item	~2	~3	~2	~2	~2	~3	~2	~2	~3	~4	~3	~3
Total time (min)	16.6	25	16.6	16.6	16.6	25	16.6	16.6	25	33.3	25	25

Figure 8: BEN-4D's Hit Rate Statistics (1kb hidden data)

Figure 8 also demonstrates how the larger in dimension and size the image is, the smaller the GBL hit rate becomes. This can be explained due to the fact that the overall alteration of the byte array structure is statistically smaller thus more difficult to detect. Section 1, Step 4, YCC/RGB values are copied with a loss. The bigger the size of the embedded medium the higher the hit rates. Thus the results illustrated on Figure 8 are the worst case scenario.

Finally, scanning time is almost doubled when utilizing the full steganalysis algorithm in comparison to simply utilizing GBL. This fact will be used as a feature to give BEN-4D's user the opportunity to choose between two different scan modes a fast one and a full one.

8. CONCLUSIONS AND FUTURE WORK

Steganalysis is a serious concern within the field of Computer Forensics, as a number of cases exist that have been solved due to hidden data retrieved by experts. This paper's contribution to this is a novel method of steganalysis using Generalized Benford's Law on Reconstructed Cover Image Stego-Carrier Files.

We designed and develop a user-friendly tool with a Graphical User Interface (GUI) in order to simplify the process of steganalysis and aid the average investigator user.

Experimental statistical results presented in this paper, demonstrate that using our proposed method high rate, effective and quick detection of the given steganography algorithms can be achieved.

Consequently, taking into account the potential impact of malicious use of steganography, it is essential to develop a general purpose software tool that will not only effectively locate all kinds of JPEG steganography, including double encoding algorithms such as Outguess or F5 that are not yet supported, but it will also be able to detect steganography on any possible image carrier file.

As an enhancement of our theoretical work:

1. Data loss during the reconstruction process can be diminished by using enhanced algorithms. This will result in almost identical Reconstructed Files to the Original ones or what is known as lossless transcoding (Sanchez 2006).
2. The use of more complex steganography algorithms, such as Outguess or F5, can be detected by adding new steps to the developed algorithm. These algorithms re-encode the original image in order to produce a carrier File. The encoding procedure is standard for every one of these algorithms, producing unique "EXIF Make" or Software Signature. This can be a starting point for further investigation of the suspect file.

As an improvement to the tool designed:

1. More steganography algorithms have to be detected, including the ones that re –encode the Carrier file.
2. More JPEG encoding algorithms need to be supported.
3. More image formats have to be supported including BMP, TIFF and PNG.
4. Cryptanalysis of the verified stego-carrier files can be added, in order to export the hidden information.

By both improving the method and the software implementing it, a new powerful detection tool will be produced assisting in the recovery of hidden files of potential evidential value.

9. References

Acebo, E.D. and Sbert, M. (2005), "Benford's law for natural and synthetic images," in Proc. of the First Workshop on Computational Aesthetics in Graphics, Visualization and Imaging, L. Neumann, M. Sbert, B. Gooch, and W. Purgathofer, Eds., Girona, Spain, pp. 169–176.

Anderson, R., Fabien A., Petitcolas, P. (1998), "On the limits of steganography", IEEE Journal of Selected Areas in Communications

Barbier, J., Filiol, E., Mayoura,K. (2007), "Universal Detection of JPEG Steganography", Journal of Multimedia, Vol. 2 no 2, pp. 1-9.

Benford, F. (1938), "The law of anomalous numbers," Proc. of the American Philosophical Society, vol. 78, pp. 551–572.

Buck, B., Merchant, A., Perez, S. (1993), "An illustration of benford's first digit law usinbg alpha decay half lives," European Journal of Physics, vol. 14, pp. 59–63.

Camouflage website, http://camouflage.unfiction.com/ (last accessed June 2011)

Fridrich, J., Goljan, M. "Practical Steganalysis of Digital Images – State of the Art", In Proceedings of SPIE

Hernandez, J.R., Amado, M., Perez-Gonzalez, F. (2000), "DCT domain watermarking techniques for still images: Detector performance analysis and a new structure," IEEE Trans. on Image Processing, vol. 9, no. 1, pp. 55–68.

Independent JPEG Group - http://www.ijg.org/ (last accessed June 2011)

Invisible Secrets website, http://www.invisiblesecrets.com/ (last accessed June 2011)

Jolion, J.M. "Images and benford's law" (2001), Journal of Mathematical Imaging and Vision, vol. 14, no. 1, pp. 73–81.

JPEGClub - http://jpegclub.org/ (last accessed June 2011)

JPEGSnoop, http://www.impulseadventure.com/photo/jpeg-snoop.html (last accessed June 2011)

Kessler, G. (2004), "Null Ciphers: An Overview of Steganography for the Computer Forensics Examiner", FBI's Forensic Science Communications.

Latham, A. (1999), "Steganography: JPHIDE and JPSEEK", http://linux01.gwdg.de/˜alatham/stego.html (last accessed June 2011)

MATLAB JPEG Toolbox by Phillip Sallee http://redwood.ucdavis.edu/phil/demos/jpegtbx/jpegtbx.htm (last accessed June 2011)

Nigrini, M. (1996), "A taxpayer compliance application of Benford's law", Journal of the American Taxation Association, vol. 1, pp. 72–91.

Nosratinia, A. (2001), "Enhancement of JPEG-Compressed Images by Re-application of JPEG", JOURNAL OF VLSI SIGNAL PROCESSING SYSTEMS FOR SIGNAL IMAGE AND VIDEO TECHNOLOGY, Vol. 27 No 1/2, pp. 69-80.

Pérez-González, F., Heileman, G., Abdallah, C.T. (2007) "Benford's law in image processing", in IEEE International Conference on Image Processing, San Antonio, TX, USA.

Sanchez, D. (2006), "Lossless JPEG transcoding", ECE 533 Project Report.