

Cyber Forensics:
Identification and analysis of least significant bit
steganography artifacts in image, audio and video files.

Author: Jean-Michel Batista

Supervisor: Dr Yasir Khan

Moderator: Dr James Shuttlewort

Abstract

Steganography enables secrecy and deceit by fooling an adversary's sensory systems or, on a larger scale, an adversary's intrusion detection system. Steganography has become a prime factor in obfuscation techniques. Whether for pure or malicious intents, it is a potential risk for users and organizations. Literature reveals that, there is not enough research on identifying the effects of the least significant steganographic algorithms on audio or video carriers. This project aims to provide an in-depth analysis of image, audio, and video files that have undergone a least significant bit steganography process and examine them within a digital forensics environment. The primary research plan of this project will use the experimental method to systematically and scientifically approach the problems in question. Results reveal that least significant bit algorithms, whether random or sequential, will attempt to overwrite bits that are not relevant to the resulting steganographic carrier's visual or auditory quality. This project's experiments demonstrate that despite a least significant bit embedding algorithm's efforts to obfuscate data behind a carrier there will always be changes caused by the steganographic process. Modern steganographic algorithms will have requirements to ensure that their output's quality can bypass a human's sensory system, because of this cyber forensic examiners must dig deeper into the binary composition of a file.

Table of Contents

Abstract	3
Research Questions	3
Table of Contents	4
1.1 Overview	7
1.2 Aims	7
1.3 Objectives	7
1.4 Tools	8
1.5 Document Structure	8
2 Literature Review	9
2.1 Introduction	9
2.2 Stegomalware	10
2.3 Data Exfiltration & Video Steganography	11
2.4 Spatial Domain Steganography - LSB	12
2.5 Steganography in Cyber Forensics	15
2.6 Steganalysis	16
2.7 Conclusion of Literature Review	21
3 Research Method	22
3.1 Least Significant Bit Algorithms	22
3.2 Examination of steganographic carriers	23
3.3 Carriers	23
3.4 Steganographic Content	24
3.5 The testing environment	25
4 Requirements	25
4.1 Experimental Requirements	27
5 Experimental Investigation	27
5.1 Experiment 1: Image Cover Carrier vs. Image Steganographic Carrier.	27
5.2 Experiment 2: Audio Cover Carrier vs. Audio Steganographic Carrier.	37
5.3 Experiment 3: Video Cover Carrier vs. Video Steganographic Carrier.	41
6 Discussion of Results & Solutions	45
6.1 Is it possible to identify key indicators and digital forensic artefacts on a media file that has gone through a steganographic process?	45
6.2 How does the steganographic process affect the morphing of media and how can this information be harvested to progressively raise a culture of security awareness and improve cyber defensive postures?	45
6.3 Solutions	46
7 Recommendations and Guidance for Future Research	47
8. Conclusion	48
9. Project Management	48

Figure 39 - Project Summary	48
Figure 40 - Gantt Chart of The Project Management Based on Project's Summary	49
9.1 Personal Reflection	49
10. Bibliography	50

1 Introduction

1.1 Overview

Steganography is an integral part of our modern world's technology. It is a time-honored science that has widely developed in both offensive and defensive cyberspace within the last decade, attracting the attention of digital forensics and cybersecurity experts in the field. In practice, steganography enables secrecy and deceit by fooling an adversary's sensory systems or, on a larger scale, an adversary's intrusion detection system. The popular digital steganographic processes involve obfuscating binary information within a carrier, with today's most common carriers being image, audio, and video files. Due to the sheer amount of volume these filetypes have over the internet, users and organizations need to be concerned in maintaining a cyber defensive posture towards any file that is not their own. However, identifying files that have undergone a steganographic process remains critical to uncover hidden information or prevent cyber-attacks.

In this project, we identify forensic artifacts left behind by steganography tools in image, audio, and video files to discover steganographic patterns that can ease the time invested in a forensic investigation or improve an organization's cyber defense posture. Identifying forensic artifacts in a forensic environment will provide more information than what the naked eye can see; tracking this automated process will give us a better understanding of the steganographic process and the post-steganographic process analysis. Furthermore, the information gathered will be helpful to explore the asymmetrical relationship between the steganographic algorithm and the carrier and its hidden contents. This report will detail the process of analyzing stego-files in the image, audio, and video category.

The post-steganography analysis revolves around one forensic toolkit; the steganographic process itself was achieved by using multiple third-party tools chosen to meet the objectives of this project; the combination of these tools were made part of this project to create the results following our experiments.

1.2 Aims

This project aims to provide an in-depth analysis of image, audio, and video files that have undergone a least significant bit steganography process within a digital forensics environment. In order to achieve the project's aims, a set of experiments followed covering the least significant bit steganography method, how to identify it, and how to defend against it. Based on our findings, we will identify patterns within the steganographic process's remaining forensic artifacts and answer the proposed research questions.

1.3 Objectives

The main objective is to identify and analyze steganography artifacts in image, audio, and video files with the help of automated tools that can perform specific steganographic

processes and a digital forensics application. During the lifespan of this project, other objectives we will need to achieve are:

- Discover existing research made on the subject.
- Discover past similar projects and learn from their results.
- Develop a personal skill set with the tools that can perform steganography.
- Comprehensive Literature Review.
- Justify methodology chosen.
- Creation of experiments.
- Evaluation of findings.
- Final review of the project.

1.4 Tools

The tools used to achieve the completion of this project were the following:

Tool used for steganographic analysis:

- The Sleuth Kit / Autopsy (Digital Forensics Platform)

Tools used for steganography:

- Steghide
- DeepSound
- OpenPuff

1.5 Document Structure

The report structure in place will aim to present the project coherently and logically. This section of the report introduces the project to the reader, presenting the problem that our research will seek to answer. It is also where we present the aims and objectives of the project and the tools used to achieve the completion of this project. We will then include a background & literature review to support our project with the state of scholarship in steganography and cyber forensics, here we will discuss and analyze each source's central point and arguments to convey their strengths and weaknesses while providing arguments that justify the need for this project. Next, we will discuss the methodological approach taken for this project to clarify our primary research plan, followed by the requirements put in place in order for this project to be successful. Additionally, we will provide an in-depth analysis of the experiments made in this project along with the findings and conclude this report by sharing our thoughts, giving recommendations for future research in the subject, and sharing the project management approach taken for the lifespan of this project.

2 Literature Review

2.1 Introduction

Steganography is an obfuscation technique that traces back to over a thousand years in history, but it is a technique that is relevant to this modern-day and age due to its potential in digital media. Unlike cryptography, the science of steganography consists of hiding the existence of information by exploiting holes in the human sensory system (HSS) (Hamid et al., 2012)[1]. The use of computational devices and the internet has allowed for never seen before access to worldwide communications where cryptography and steganography methods can secure the transit of practical information. In addition, a better understanding of digital file composition has allowed the development of sophisticated steganography tools that require not only high perception to identify but also a good understanding of computing concepts. Due to these strengths, steganography has become a prime factor in obfuscation techniques. Whether for pure or malicious intents, it is a potential risk for users and organizations.

While the majority of research focuses on image steganography, Kaur & Behal (2014)[2] have counter-argued that audio and video steganography is as much as a concern, and the ability to hide data in a variety of carriers (filetypes) creates a large variety of different techniques, each with their advantages and disadvantages. Some techniques may be easier to identify and reverse engineer than others due to multiple reasons, such as being available to the public or having a unique digital signature in their steganographic process. On the other hand, other techniques might prove very difficult to identify or reverse engineer due to other reasons, such as the complexity of the steganographic process or not well-known steganography techniques. In cyberspace, the presence of various steganography techniques requires digital forensics investigators, cybersecurity experts, and organizations to stay updated as lacking the knowledge to properly analyze or identify an obfuscated digital file could potentially have legal repercussions in a trial or investigation or pose a critical risk to an organization's assets. However, a zero-day steganography technique could have a high chance of fooling the eyes or ears of an experienced forensic investigator or a robust intrusion detection system as they will not have a complete understanding of what is in front of them nor pre-existing digital signatures that could identify the steganographic carrier as a threat. Therefore, our project will seek to identify patterns related to the post-steganographic analysis across the chosen carriers within a digital forensics environment and attempt to bridge their similarities as a result.

2.2 Stegomalware

We define *malware* as malicious software. These are commonly known but not limited to trojans or ransomware. From stealing sensible information to encrypting the contents of a drive and asking for a ransom, the sole intent of malware is to perform harmful actions towards a target's computer system. However, as the use of malware often leads to criminal activity, the ability to hide information in benign-looking digital objects has attracted many attackers who have sought to illegally bypass security systems without detection in order to continue malicious routines with a low profile. According to McAfee Labs threats report (2017)[3], steganography is one of the most accessible techniques to implement in cyber-attacks while at the same time remaining one of the most enormously tough to detect techniques. Cyber-threats that use steganography stand amongst the most challenging to uncover because they subtly arrive as zero-day threats, which is difficult for a human to detect, but it is also tricky for antivirus and intrusion detection system tools that rely on signature databases. Due to this nature, stegomalware is more likely to remain undetected within a compromised system for more extended periods.

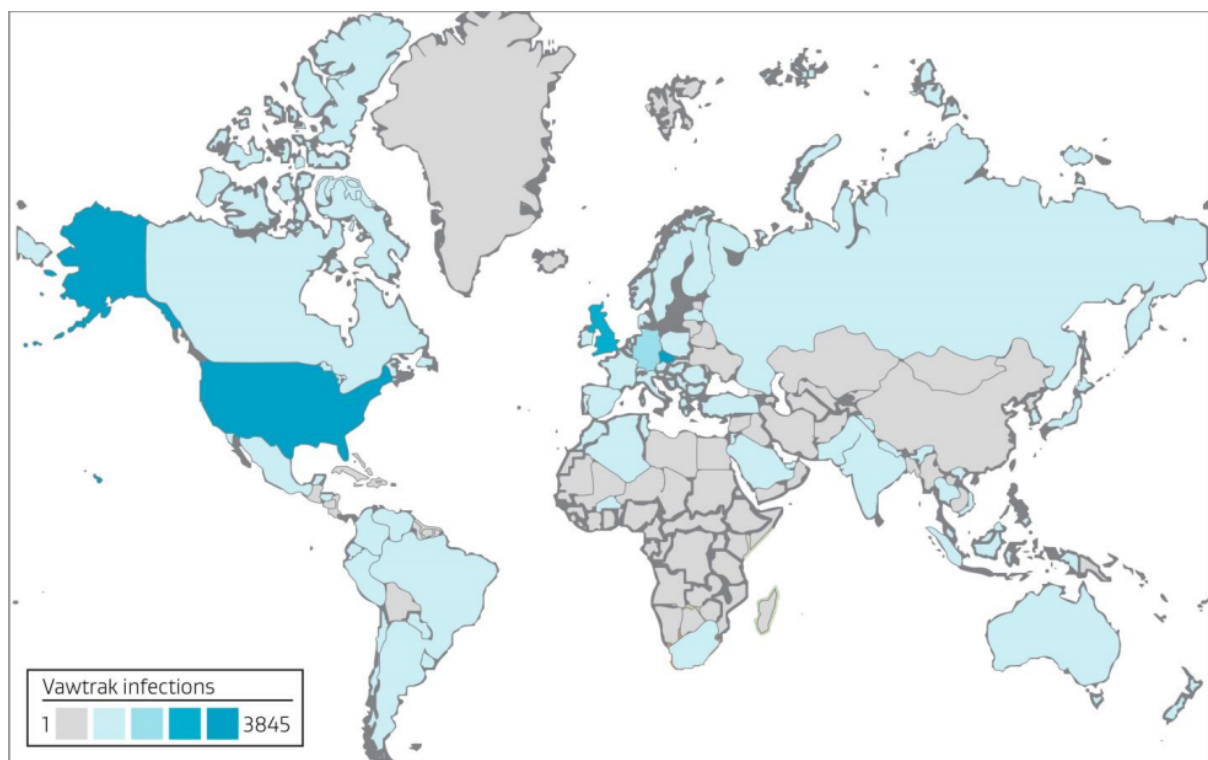


Figure 1 - Countries most affected by the spreading of Vawtrak in Q1 2015 (Křoustek, 2015)[4].

As it is easy to convince users to open downloaded files without raising much suspicion, malware authors have adopted steganography as a core trait of their craft. Cyber-threat records show how steganography has given malware the ability to hide commands sent to infected systems or hide malware update files in favicons as the trojan Vawtrak did back in 2015, where thousands of computer systems had fallen victim to the malware, as seen in Figure 1. Users and organizations remain at risk as new stegomalware approaches could surface anytime, requiring organizations to raise awareness, create new policies, rules and provide training to their staff to maintain their cyber defensive posture towards these types of attacks. However, possibilities within the steganographic spectrum are bound to creativity and not limited to stegomalware.

2.3 Data Exfiltration & Video Steganography

Records of cyber threat incidents where attackers have bypassed the cyber defense systems used by an organization to exfiltrate data using video steganography prove that automated image steganography detection is not the only solution. In addition, organizations that focus on detecting one specific carrier file type are more likely to be vulnerable against other unmonitored carriers. Therefore, this project's scope has not been limited to images only but has been expanded to cover video and audio carriers as well.

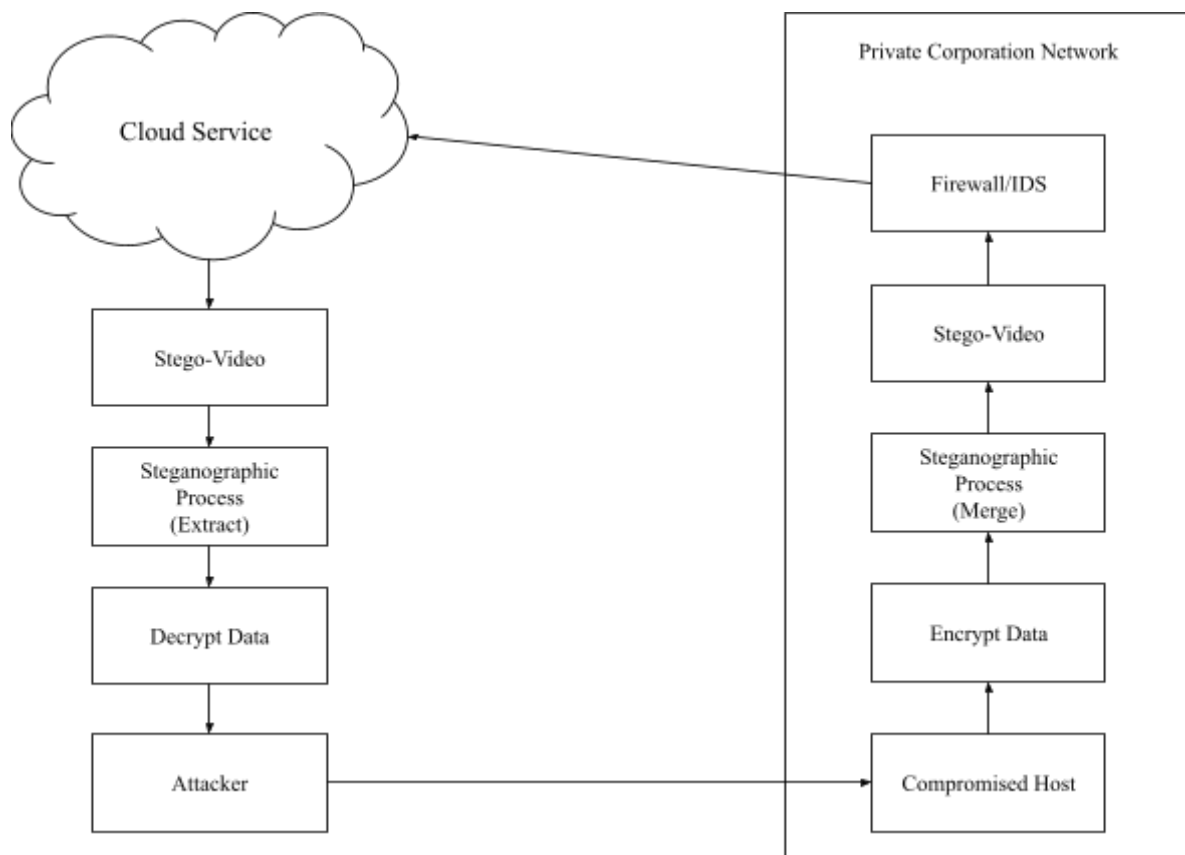


Figure 2 - Threat Model of Data Exfiltration Attack Using Video Steganography

As seen in Figure 2, we r

Computer science researchers Shantala & Viswanatha (2015)[5] proposed a secure software-defined network framework to solve data exfiltration via video steganography. Their solution transformed network nodes into reporting tools by programming a distributed set of SDN switches with the goal of filtering packets. The packets header fields would be compared to flow table entries for hidden data. If a match were found, the packet would be dropped, and an alert would be sent to the network administrator. Unfortunately, while this solution proved efficient in detecting video steganography, it did not work for all video formats. If an attacker changed their format of choice to one the solution did not support, the solution would turn obsolete against video steganography. In turn, Hernandez-Casto & Sloan (2015)[6] research shifted its focus to identifying forensic artifacts left behind by video steganography tools. Their steganalysis proved that the embedding algorithm used during the steganographic process done by tools allowed for signature identification based on hexadecimal values within the carrier file. This signature steganalysis approach accurately identifies the presence of steganography while revealing the tool used, a good and scalable solution that has seen its implementation in signature databases as scripts can be used to automate the signature identification process. However, while steganalysis is a good practice in cyber forensics and has its strengths, its weakness lies in requiring a range of data samples created with the same steganographic tool, thus making it weak against sophisticated and targeted attacks that use undocumented digital signatures or embedding algorithms that shuffle their digital signature as a concealment method.

2.4 Spatial Domain Steganography - LSB

We describe the digital steganographic process as embedding data within a carrier file using an embedding algorithm. There is a massive scope in hiding data within carriers as digital media can be megabytes or more, and we can hide files or information of megabytes or more in them using various techniques. The most common form of steganography in media is the least significant bit steganography process. This process works on all data formats and commonly works by embedding the least significant bits with extra data. When it comes to an image, it is possible to change the lowest one or two bits of every single byte to be our message or payload while having an almost imperceptible change (Singh et al., 2015)[7]. For example, if we have a 4 bytes long standard pixel and only the last two bits in that byte change, the image would still look the same, only to the naked eye.

In recent years the large number of approaches proposed for spatial domain digital steganography has turned its academia into a cat and mouse game where researchers seek to improve the stealthiness of data embedding algorithms and, in turn, improve the detection mechanisms for such algorithms. The research of Al-Omari & Al-Taani (2017)[8] showed that it is possible to adaptively isolate and prioritize eligible pixels of an image for least significant bit modifications based on their adjacent RGB threshold values. By excluding pixels that can cause image distortion or reduce their quality, steganographic images can gain resistance against regular-singular (RS) statistical attacks such as the one seen in Figure 3.

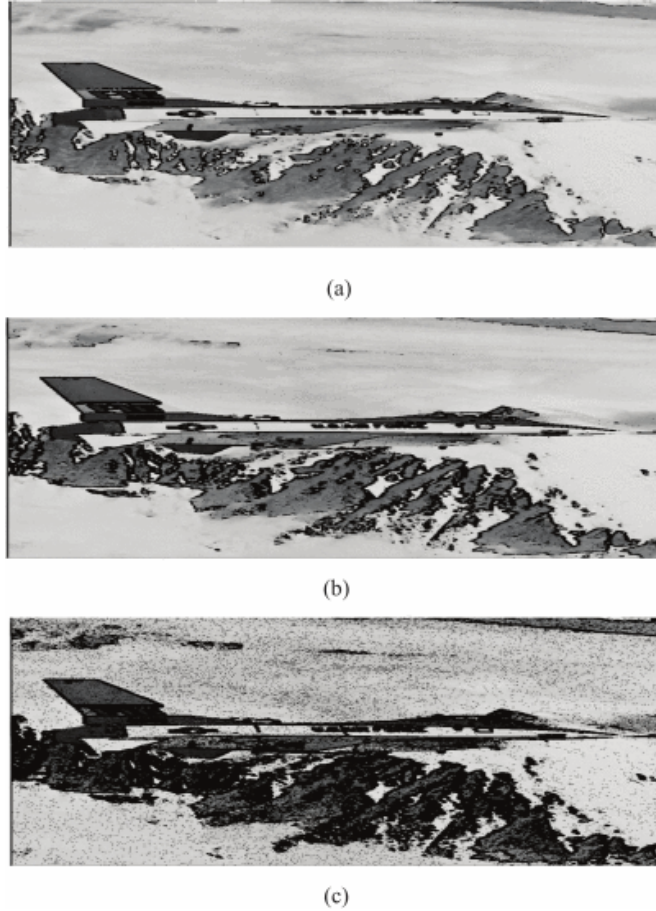


Figure 3 - Steganographic image modified for message. Size 8k (a), 16k (b), 32k (c). (Fridrich et al., 2001)[9]

Ever since, better approaches (Vishnu et al., 2020)[10] have surfaced within the spatial domain category where the obfuscated data is embedded into the edges of an image using a combination of the pixel value differentiating algorithm (PVD) proposed by Wu & Tsai (2003)[11] and Canny's (1986)[12] edge detection algorithm.

Wu & Tsai (2003)[11] had previously discovered that the edge elements or pixels of an image could embed more amount of data compared to other regions of the image and thus devised a PVD algorithm that did not alter grayscale proportions, which plays a prominent role in the response of the human vision to the gray value[10]. However, despite multiple advances adding complexity to spatial and transform steganography, the least significant bit technique has remained popular over other techniques due to how easy and efficient it is to hide data using this technique and how hard it remains to identify without steganalysis. Therefore, researchers keep improving least significant bit methods that maximize the image quality of the resulting steganographic image in contrast to the amount of data they can carry. For

example, experimental results shown by Kuo et al., (2015)[13] achieved a large embedding capacity and high imperceptibility in their steganographic image by implementing arithmetic division in their least significant bit embedding algorithm operations and combining it with a generalized exploiting modification direction (GEMD) scheme as seen in Figure 4. Additionally the approach proved viable against bit-plane attacks as the steganographic images used in the experiment did not reveal any secret information post bit-plane analysis. Due to the nature of least significant bit steganography and the way it embeds data into a carrier, we can assure that the same principles apply to audio and video, but literature does not clarify the impacts of spatial domain steganography in these types of carriers.

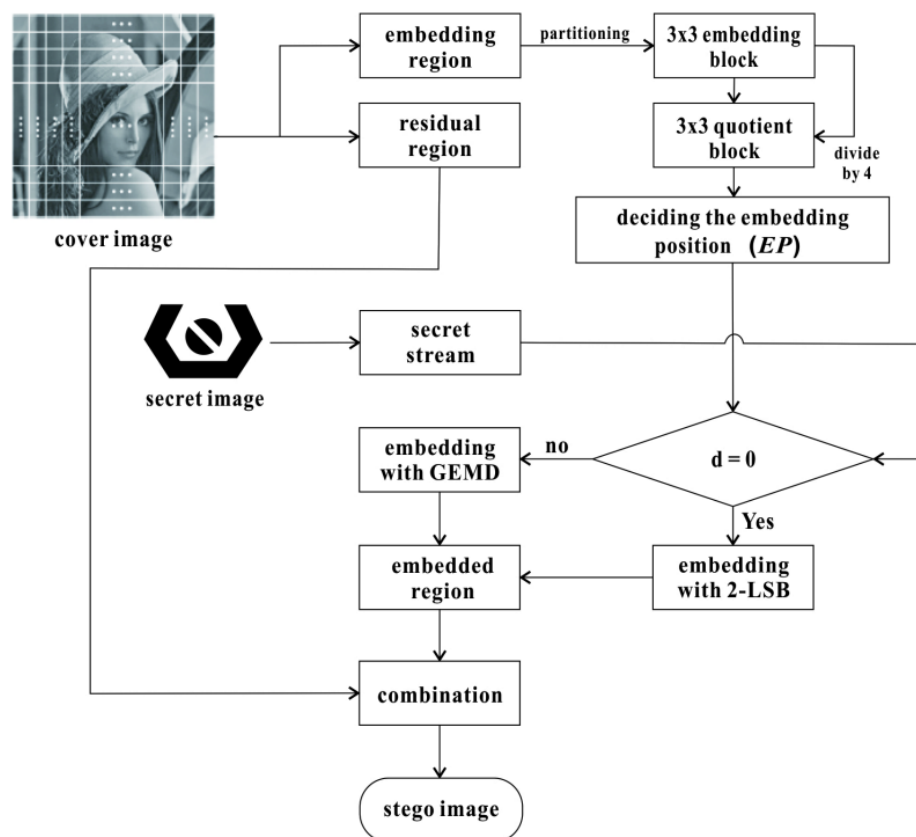


Figure 4 - Framework of the least significant bit DA-GEMD embedding process. (Kuo et al., 2015)[13]

2.5 Steganography in Cyber Forensics

Cyber forensics is still relatively known as one of the newer fields in computing and holds core properties found in traditional forensic science. These core properties include acquiring, preserving, and analyzing digital evidence through scientifically derived and proven methods. When combined with other properties such as the interpretation, documentation, and presentation of digital evidence, cyber forensics holds the power to reconstruct events of criminal activity and discover newer evidence that could make or break a case of jurisdiction

(Palmer, 2001). Considering the following hypothetical scenario. An intelligence group is assigned the task of investigating members of a terrorist organization. Without delay, communications are intercepted, and monitoring systems are put into place. The suspect leader of the terrorist group is known for sharing a picture along with the address of the organization's meeting location via WhatsApp. An incident occurs where the address matches the picture's location. Law enforcement forces are deployed to act at once to find no one at the meeting location. Without further analysis, it is impossible to know whether the organization's leader might have known about the presence of intelligence services and instead sent out a steganographic image informing the rest of the organization. These types of scenarios and ones like them are valid methods of operation for criminal organizations.

It is tough to know the statistics regarding the use of steganography by criminals or terrorists as steganography will not be found if it is not being looked for under an active investigation. As an old rule of thumb, Kessler (2004)[15] argued that despite the ever-growing steganography challenge, ignoring its significance due to lack of statistics could only achieve security through denial, a bad long and short-term security strategy. Without considering stegomalware, steganography has seen state-threatening cyber warfare uses within the context of terrorism. For example, in 2012, a man named Maqsood Lodin, suspected of involvement with the terrorist group Al Qaeda was apprehended in Berlin. During the investigation, German police confiscated a digital storage device that contained a pornographic video named "Kick-Ass," after further stego-analytic investigation, the video was discovered to possess over 100 Al Qaeda documents, including future attack operations and training manuals (Robertson et al., 2012)[16]. The United Kingdom legal system dictates that evidence will be admissible at common law when the presented evidence is reliable and of assistance to the court (CPS.gov.uk, 2019)[17]. Therefore, cyber forensic investigators are free to choose the tool and method best fit for the investigation's purpose as there are no legal requirements that dictate a specific tool or method. However, the most robust methods to defeat steganography techniques are attributed to steganalysis and its wide range of steganalysis approaches proposed over the past years.

2.6 Steganalysis

2.6.1 Visual Analysis

More than one steganalysis technique exists to identify the presence of steganography on a cover medium. Visual steganographic attacks are the simplest form of steganalysis. These types of attacks are represented by the examination of a suspicious image with the naked eye. Poor steganographic algorithms create visual artifacts that are easy to identify if the original image is known to the examiner. Figure 5 shows an unaltered cover image, while Figure 6 is

the same image with an embedded text tile, visual artifacts have been highlighted within red circle areas for easier detection. (Karampidis et al., 2018)[18].

Figure 5 - Unaltered Cover Image.[18]



Figure 6 - Steganographic Image with embedded text file.[18]

2.6.2 Digital Signature Analysis.

However, because sequential least significant bit embedding algorithms are easily detectable by visual attacks, newer steganographic software that use randomized least significant bit embedding have surfaced, making the presence of visual artifacts a lot more subtle thus harder to detect by the human eye. Therefore, a different approach to steganalysis is to identify the repetition of patterns created by the embedding algorithm of the steganographic software. This technique searches for signature patterns to determine the presence of steganography [18]. A known steganographic attack is when the steganographic tool and both the cover and steganographic image are available for analysis. In addition, practical examinations that target security vulnerabilities in the embedding algorithm can determine whether a specific steganographic software has been used. For example, despite the embedded contents, the Hiderman steganography software always attaches the hexadecimal values 43, 44, and 4E in sequence at the end of the steganographic file. These hexadecimal values translate to the characters "CDN" as seen in Figure 7, a prime vulnerability under the context of signature steganalysis.

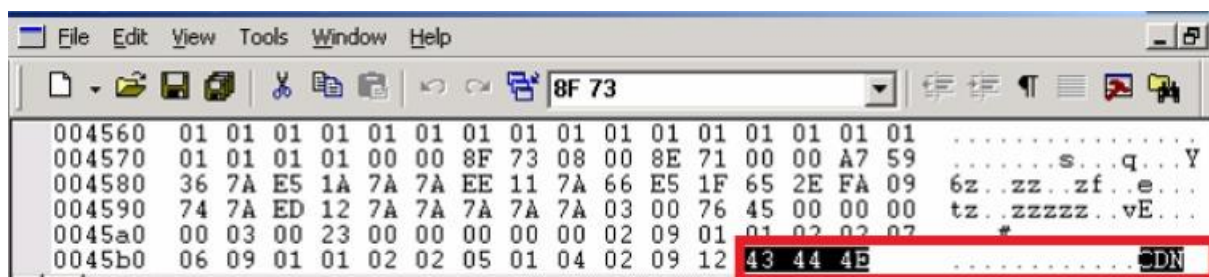


Figure 7 - The Hiderman Steganography Software Signature [18].

Another steganalysis technique is to analyze the embedding algorithm and determine statistics modified as a result of the steganographic process. Proposed statistical approaches focus on two factors, the replacement of least significant bits and the matching of least significant bits in carrier files.

2.6.3 Statistical Analysis - Least Significant Bit Matching

For steganographic carriers to avoid specific statistical attacks, the least significant bit matching steganographic techniques compare each bit of the content against the least significant bit of the corresponding carrier's cover byte. As long as the compared bits match, no change is made to the carrier[18]. However, if a mismatch is found, the carrier's bit plane is increased or decreased at random to avoid detection. For example, figure 8 shows the embedding process for LSB matching (Sharp, 2001)[19], where C is the cover image, C_i is the i th least significant bit, M is the hidden message, M_i is the i th bit of M , S is the resulting steganographic carrier, and S_i is the i th least significant bit of the steganographic carrier.

$$S_i = \begin{cases} C_i, & \text{if } M_i = C_i \\ C_i - 1, & \text{if } M_i \neq C_i \text{ and } C_i \neq 0 \\ C_i + 1, & \text{if } M_i \neq C_i \text{ and } C_i = 0 \end{cases}$$

Figure 8 - LSB Matching Embedding Process

In order to combat these types of techniques, Chen et al. (2016)[20] proposed an approach to calculate the differences between pixel pairs based on discriminative features based on the difference histogram characteristic function (DHCF). Their results demonstrated that the DHCF function was helpful in the detection of steganographic carriers that used least significant bit matching when calculated against non-adjacent pixels. Additionally, the experiments also proved that the presence of steganographic noises smoothes the histogram of difference values. The following year Sandoval et al. (2017)[21] proposed a different scheme based on the probability density function (PDF) of adjacent pixels and the co-occurrence matrix of the image. The experiment used a progressive support vector machine (SVM) to distinguish steganographic images from their respective cover, and its results showed an increase of 87.2% in effectiveness when compared to previously proposed methods. Despite the success of these proposed methods against image steganography, they would still fall short against audio and video steganography, not making them the all-around best approach for the identification of steganography.

4.6.4 Statistical Analysis - Least Significant Bit Replacement

Two different embedding techniques fall under the category of least significant bit replacement algorithms. The first technique aims to sequentially substitute bits of the carrier file by starting at the top of the bit plane and then downwards until the entire steganographic content is embedded as seen in Figure 9 & 10. On the other hand, the second technique randomly substitutes bits of the carrier file by dispersing the position of the least significant bits within the bit plane at random as seen in Figure 11 & 12. Researchers Fridrich et al. (2013)[22] proposed an adaptive machine learning detection solution to identify replacement algorithms based on co-occurrences of neighboring noise residuals and pixel parity. At the time, the calibration by parity and parity-aware residuals was considered two key novel concepts as they could be used in a binary classifier and perform better than other least significant bit replacement identification solutions such as the extraction of feature vectors derived from gray-level co-occurrence matrices (GLCM) proposed by Kekre et al.(2011)[23].

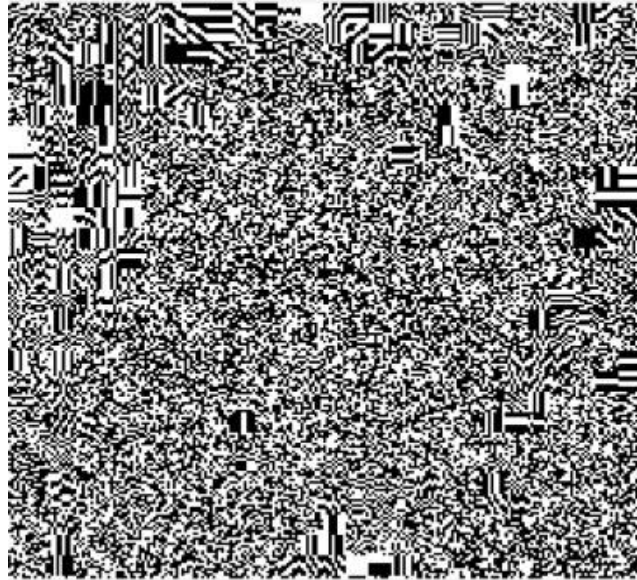


Figure 9 - Sequential LSB Unaltered Image Bitplane.[18]

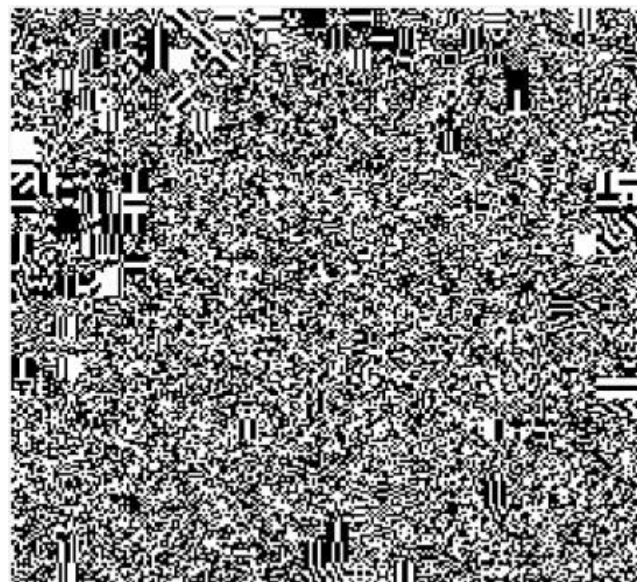


Figure 10 - Sequential LSB Steganographic Image Bitplane.[18]

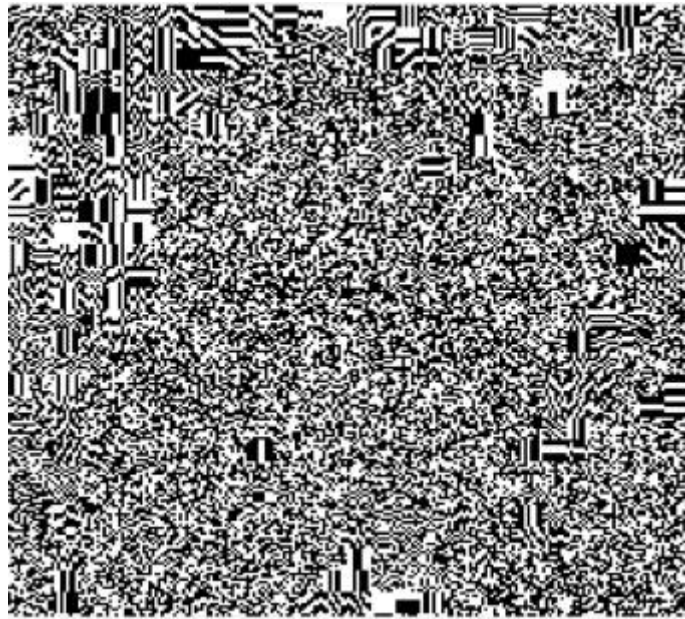


Figure 11 - Random LSB Unaltered Image Bitplane[18]

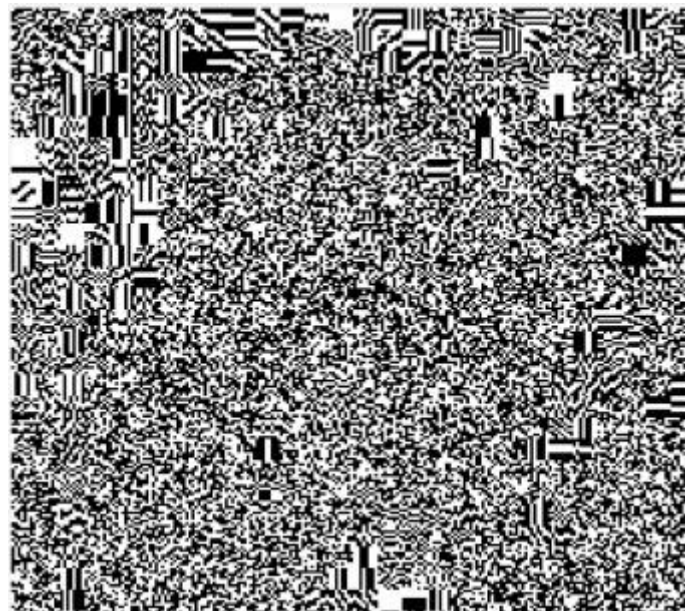


Figure 12 - Random LSB Steganographic Image Bitplane[18]

2.7 Conclusion of Literature Review

Our literature review shows a wide range of approaches that counter the least significant steganographic techniques on image carriers, the relevance of steganography in cyber forensics, and how steganography can be leveraged to pose a threat against individuals and organizations. However, what would the results of these proof of concepts be under the context of other steganographic carriers? Unfortunately, there is not enough research on identifying the effects of the least significant steganographic algorithms on audio or video carriers, and more research will be necessary to differentiate the original carrier from the steganographic carrier. Due to the nature of the least significant bit steganography and the way it embeds data into a carrier, we can assure that the same principles apply to audio and video, and identifying files that have undergone a steganographic process remains critical to uncover hidden information or prevent cyber-attacks. To support the objective of answering our research questions, we propose a set of experiments that cover the identification of the least significant bit steganographic method on image, audio, and video files.

3 Research Method

Our study will consider the range of research methodologies used on previously conducted experiments in the field. The steganalysis method chosen is based on the known steganographic attack where the carrier and steganography medium and the steganography algorithm are known[15]. Furthermore, we expect the produced data to have qualitative properties due to the nature of this project and propose a solution based on the results. Requirements for the experiments and the results will be presented in the requirements section. The research topic examines the asymmetrical relationship between the steganographic algorithm and the carrier and its hidden contents. We attempt to bridge differences addressed by the cause and effect of the least significant bit steganography process on image, audio, and video files. The results of this project are targeted at the digital forensics community. Much research done in the scope of this topic is heavily invested in image files and not as much on audio or video files. Thus, having first-hand knowledge available can benefit both new researchers in the field and digital forensic investigators looking to identify if a file is under the influence of a steganographic technique. The primary research plan of this project will use the experimental method to systematically and scientifically approach the problem in question. Data samples will be produced using the steganography mentioned earlier and manipulated with the stated digital steganography techniques in a controlled environment to produce a range of data to analyze cause and effect relationships. This methodological approach best suits our research goal as the results will be

managed with a deductive approach. We will support the results section through experiments where we will control the steganography conditions files are exposed to in a way accurate measurements are possible. Because controlled experiments attempt to be easy to replicate through a standardized procedure and allow for precise control of extraneous and independent variables (Steganographic Algorithm). Results produced by this project obtained through experimental samples will be compared against the original control sample. The presented experiments were planned thoughtfully to make results reproducible. All steganographic software and forensic toolkit of choice are available to the public.

3.1 Least Significant Bit Algorithms

The achievements this research needs to accomplish for its success are identifying key indicators and digital forensic artifacts on the selected media files that have gone through the least significant bit steganography process and how it affects its carrier. The least significant bit algorithms used are available through the use of the following applications, Steghide[22] for image steganography, DeepSound[23] for audio steganography, and OpenPuff[24] for video steganography. These make part of our methodology as they will be used to create the necessary steganographic carriers. The software were selected based on their popularity as software that can perform steganography under the assumption that users will commonly base their software of choice and invest their trust in an application if it is popular. Therefore, popularity in specific steganographic software should be a target of interest for cyber defensive countermeasures.

3.2 Examination of steganographic carriers

The examination of the steganographic carriers post steganographic process will be done within the environment of the digital forensics platform Autopsy[25]. This software will allow us to view the hexadecimal composition of the steganographic carriers. It will also help us preview any changes made by the steganographic algorithms as we will have the cover carrier and the steganographic carrier side by side. We will leverage this tool to identify and analyze potential steganographic artifacts leftover by the least significant bit embedding algorithms to answer our research questions. The software Autopsy was chosen amongst other digital forensic platform competitors due to its cost-effectiveness. It offers the same core features as other forensic tools while remaining free and accessible to the public.

3.3 Carriers

The carriers used were created using a Samsung S8+ mobile phone then transferred to the testing environment. Creating our carriers allows us to guarantee they are free of external compression artifacts and unaltered, making them more suitable for our experiment. We created one of each individual carrier, an image carrier, an audio carrier, and a video carrier, respectively, in order to experiment with the least significant bit algorithms available for

each. Throughout our literature review, we mentioned the importance of individuals and organizations attributing resources to the detection of other carriers other than images as they could prove as much as a threat. Thanks to this argument, we are including audio and video files in our analysis.

3.3.1 Cover Image



r

Name: Sky.jpg

Size: 1.30 MB

Dimensions: 4032x2268

Horizontal resolution: 72 dpi

Vertical resolution: 72 dpi

Bit depth: 24

Colour representation: sRGB

Compressed bits/pixel: 0

5.3.2 Cover Audio

Available for download at:

https://drive.google.com/file/d/1HR2qK0uB-cWOtTrx81hq8tJ6M_dMSDN-/view?usp=sharing

Name: A New Beginning.mp3

Size: 26 MB

Duration: 27 seconds

5.3.3 Cover Video

The video is available at: <https://www.youtube.com/watch?v=J1o1SmesIMA>

Name: rain.mp4

Size: 92.6 MB

Duration: 10 minutes 4 seconds

Dimensions: 1280x720

Bitrate: 1280 Kbps

Frame rate: 30 frames/second

Channels: 2 (stereo)

Audio Bitrate: 127 Kbps

Audio Sample rate: 44.100 kHz

3.4 Steganographic Content

The steganographic content embedded within the carrier files through the least significant bit steganography process will be the works of Shakespeare ``All's Well That Ends Well"[26] in its entirety as it is available for free to the public. This text was chosen because it can represent the potential real scenario payload because of its length and file size and small enough to fit inside each carrier. However, it is important to note that the steganographic content within a carrier is not limited to text and can be anything, including parts of stego malware, other images, audio, or video files depending on the size of the carrier. A bigger carrier will be able to carry more data over a network or the internet.

Name: Shakespeare.txt

Size: 130 KB

Words: 24,261

3.5 The testing environment

The machine in which the experiments were tested possesses the following specifications:

Processor: Intel(R) Core(TM) i9-10900K CPU @ 5.0GHz

Memory: 32 Gb RAM

Storage: 1TB M.2 SSD

External Storage: 120GB Hitachi USB SSD

Operating System: Windows 10 Pro

This machine has a lot of computing power. However, not a lot of computing power is needed to carry out these experiments. The only factor that may vary is the time the ingest modules of Autopsy take on different computational systems. The time for embedding algorithms to embed the steganographic contents within a file is almost negligible for this project's scope or

for individuals seeking to replicate the experiments. An external SSD containing the cover carriers was used to transfer the carriers from the mobile phone to the testing environment.

4 Requirements

The project's main requirement is to use popular steganography tools mentioned above to create digital "evidence" and then analyze them using a forensic toolkit to identify steganographic behavioral patterns on image, audio, and video files. This project will also require its results to be replicable through publicly available tools to reach the widest audience possible. Additionally, the results of this project target the digital forensics community. A basic understanding of computing concepts and how steganography relates to them is required for this project to be beneficial.

In order to proceed with the experiments, it is a requirement to have the steganographic software listed above as each steganographic software owns a different embedding algorithm. All the software listed is available on the Windows operating system; users using other operating systems will have limitations on what steganographic software or forensic platform is available. Embedding the carriers with steganographic contents will also require a basic understanding of the Windows command line. Steghide does not possess a graphical user interface (GUI) and thus must be interacted with through the command line in the parent directory of the binary to provide the necessary parameters to embed a file with its least significant bit algorithm successfully. DeepSound and OpenPuff do provide GUIs, so they are not difficult to interact with.

Steghide Usage example:

```
steghide embed -cf carrier.jpg -ef content.txt
```

Optional Arguments:

-ef, --embedfile	select file to be embedded
-ef <filename>	embed the file <filename>
-cf, --coverfile	select cover-file
-cf <filename>	embed into the file <filename>
-p, --passphrase	specify passphrase
-p <passphrase>	use <passphrase> to embed data
-sf, --stegofile	select stego file
-sf <filename>	write result to <filename> instead of cover-file
-e, --encryption	select encryption parameters
-e <a>[<m>] <m>[<a>]	specify an encryption algorithm and/or mode
-e none	do not encrypt data before embedding
-z, --compress	compress data before embedding (default)
-z <l>	using level <l> (1 best speed...9 best compression)

-Z, --dontcompress do not compress data before embedding
-K, --nochecksum do not embed crc32 checksum of embedded data
-N, --dontembedname do not embed the name of the original file
-f, --force overwrite existing files
-q, --quiet suppress information messages
-v, --verbose display detailed information

The chosen carrier files are optional. However, it is a requirement that they are free of external compression artifacts so we can use clean carriers in the experiments as compression artifacts that could change the file's composition before the analysis could thwart the integrity of the experiments. It is also required for the carriers to have a larger file size than the content intended for embedding, as modern embedding algorithms will not accept contents that will compromise the carrier's visual or auditory quality. The chosen forensic platform could be up for discussion as each individual platform has strengths and weaknesses. As a minimal requirement, the forensic platform of choice should be capable of using an EXIF data ingest module and provide a preview of the hexadecimal binary streams that compose the files. We expect that the least significant bit algorithms will overwrite descriptive metadata as these properties don't play a role in the media's quality.

4.1 Experimental Requirements

Additional requirements are necessary for the success of our experimental investigation section. We are required to provide the details of each investigative step carried out in our experimental procedure. Every investigative experiment will have a set of sub-sections created to make its contents and procedures more comprehensible to the reader. Our first sub-sections communicate the purpose of the experiment and its goal. The second subsection will require us to detail the methodological steps used to produce our results briefly. The third subsection will require an in-depth discussion explaining the experimental procedure carried out and how the data is collected. Finally, we will conclude every experimental investigation with a fourth subsection where it will be a requirement to analyze and discuss the experiment results.

5 Experimental Investigation

5.1 Experiment 1: Image Cover Carrier vs. Image Steganographic Carrier.

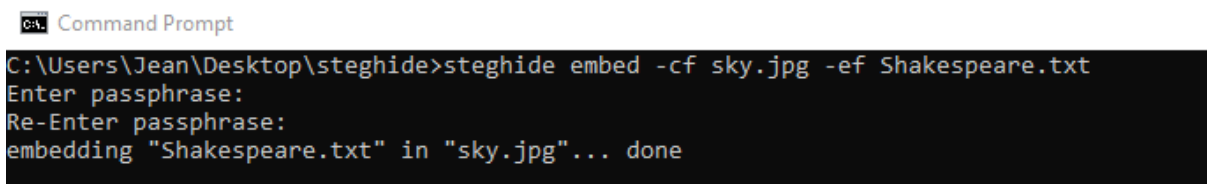
Purpose of the experiment: The goal of this experiment is to determine through conventional dead analysis changes made by the least significant bit steganography algorithm of steghide to the original image cover carrier after embedding our steganographic content.

Method of the experiment: We embed the sky.jpg image with 24,261 words using the steghide LSB embedding algorithm. The image cover carrier and the steganographic carrier are then both introduced to the digital forensics platform for analysis.

Detail of experiment: A sky.jpg image with a size of 1.30 MB possessing EXIF data was chosen as our image carrier, the dimensions of the image were suitable for this experiment because the larger the pixel count of the carrier the more data it can hold. The importance of our image carrier possessing EXIF data is based on the assumption that least significant bit algorithms, whether random or sequential, will attempt to overwrite bits that are not relevant to the resulting steganographic carrier's image quality. We then embed our image cover carrier with a 24,261 words text file using steghide least significant bit embedding algorithm with the following command line:

```
steghide embed -cf sky.jpg -ef Shakespeare.txt
```

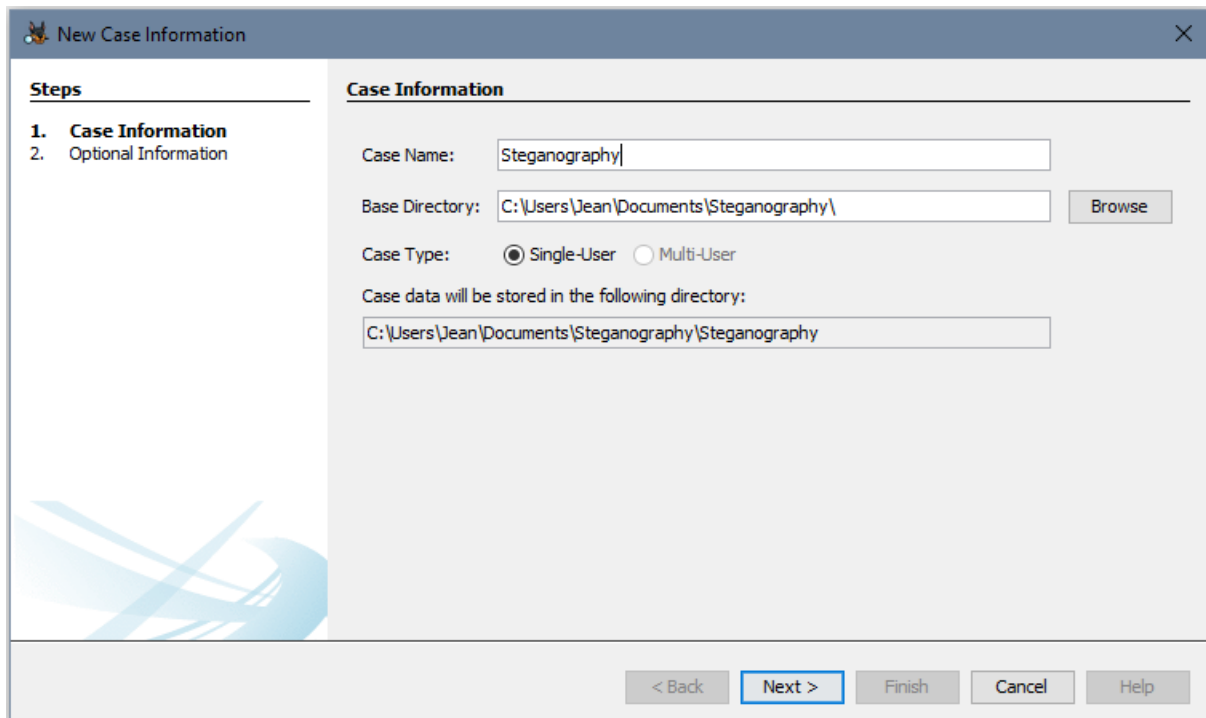
The flag -cf is used to specify the carrier file and the flag -ef is used to specify the content to be embedded. By default steghide suggests the user to enter a password to add a layer of cryptography on top of the steganographic algorithms, however, we did not use a password as cryptography algorithms also modify the contents of a file to an extent.



```
C:\> Command Prompt
C:\Users\Jean\Desktop\steghide>steghide embed -cf sky.jpg -ef Shakespeare.txt
Enter passphrase:
Re-Enter passphrase:
embedding "Shakespeare.txt" in "sky.jpg"... done
```

Figure 13 - Steghide LSB Embedding

The steganographic carrier was renamed to sky_stego to differentiate between the cover carrier and the stego carrier. Both files were introduced to the forensic platform environment for further inspection. In Autopsy we created a new case called Steganography and allocated a base directory to hold the case's information. We did not add any optional information as it is not necessary to use the software's features.



New Case Information

Steps

- 1. Case Information**
- Optional Information

Case Information

Case Name:

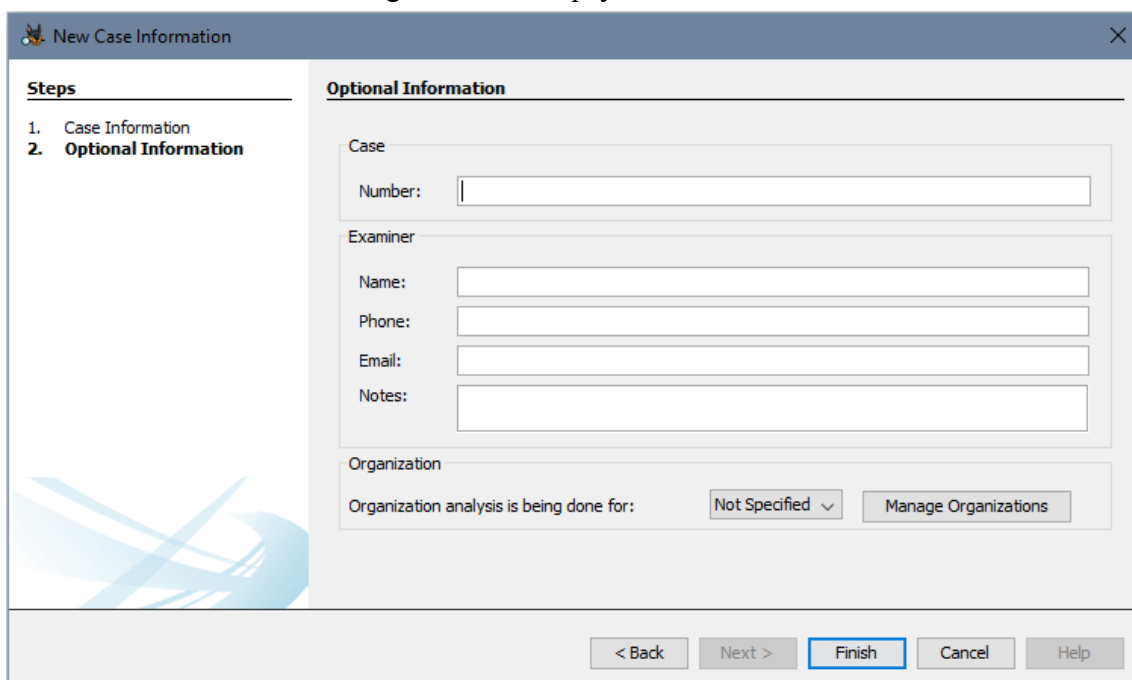
Base Directory:

Case Type: ☒ Single-User ☐ Multi-User

Case data will be stored in the following directory:

< Back **Next >** Finish Cancel Help

Figure 14 - Autopsy Case Information



New Case Information

Steps

- Case Information
- 2. Optional Information**

Optional Information

Case

Number:

Examiner

Name:

Phone:

Email:

Notes:

Organization

Organization analysis is being done for:

< Back Next > **Finish** Cancel Help

Figure 14 - Autopsy Optional Information

In order to add the data sources intended for analysis we generate a new host name based on the data source name and specify the data's source type as logical files. This will allow us to analyze individual folders or files without needing a source to image from. In this case our data source will exist as a folder containing all cover carriers and steganographic carriers.

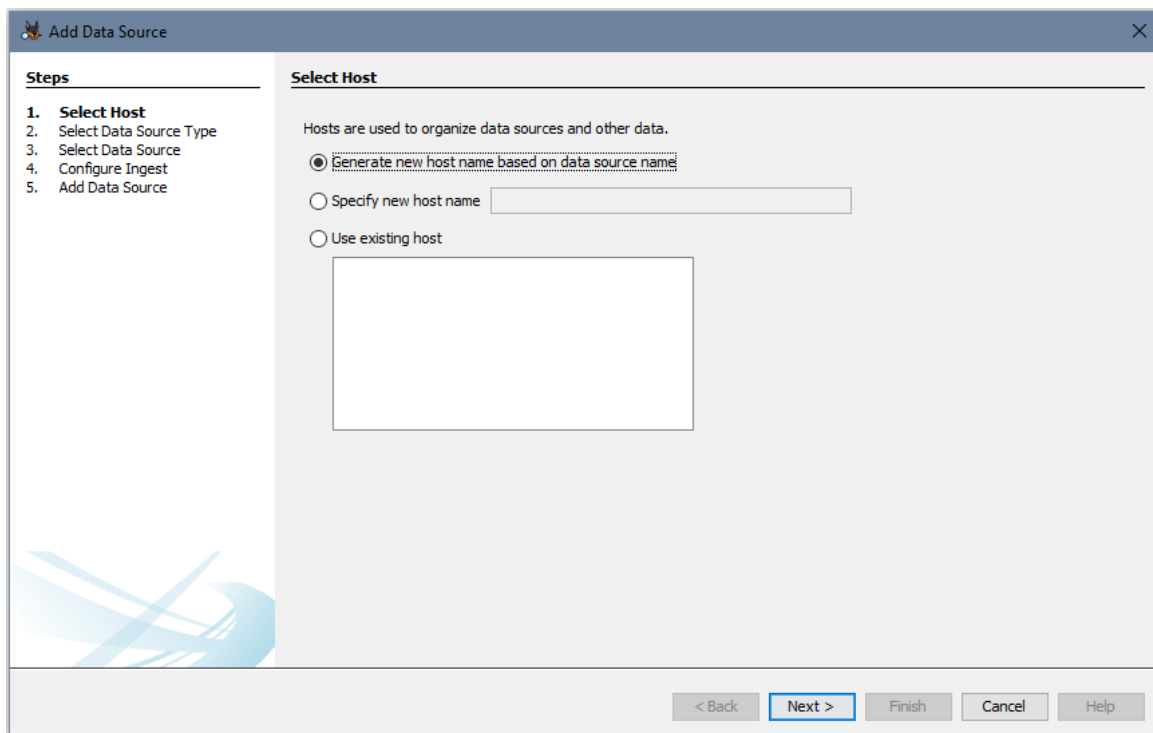


Figure 15 - Autopsy Host Selection

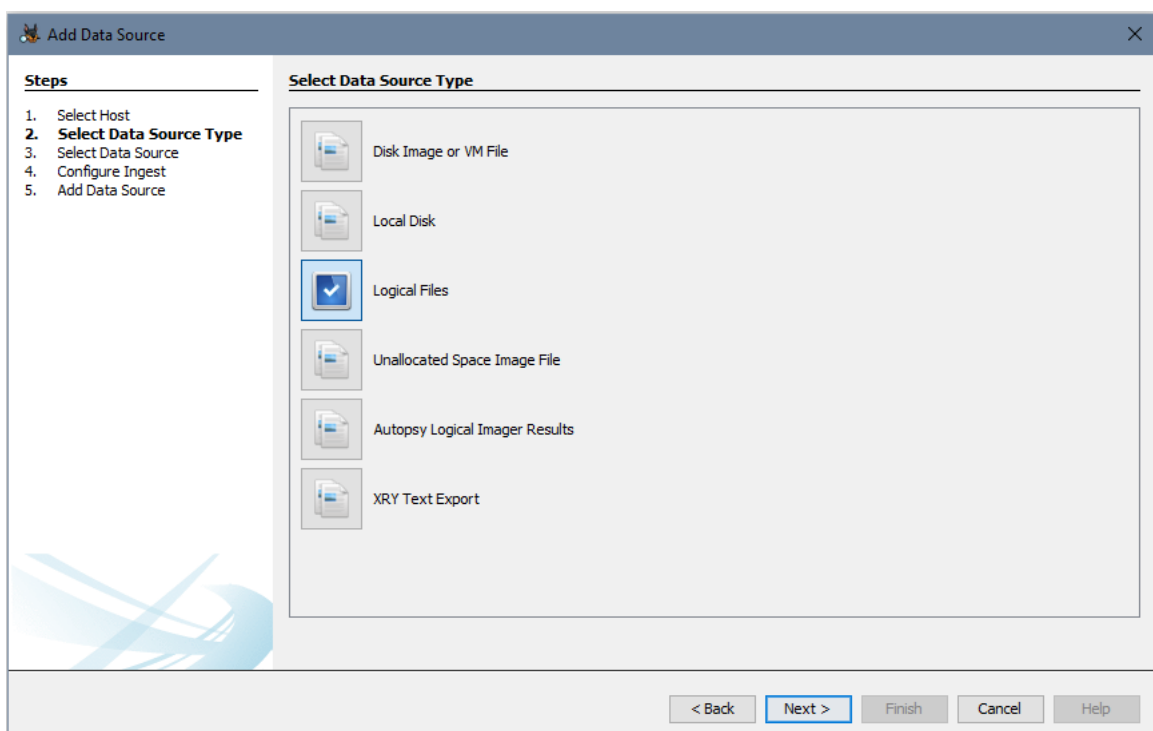


Figure 16 - Autopsy Data Source Type

The next step was to choose the ingest module configurations for the investigation. Here we chose the ingest modules based on their importance and relevance to media analysis. These included but were not limited to Picture Analyzer, Data Source Integrity and Hash Lookup.

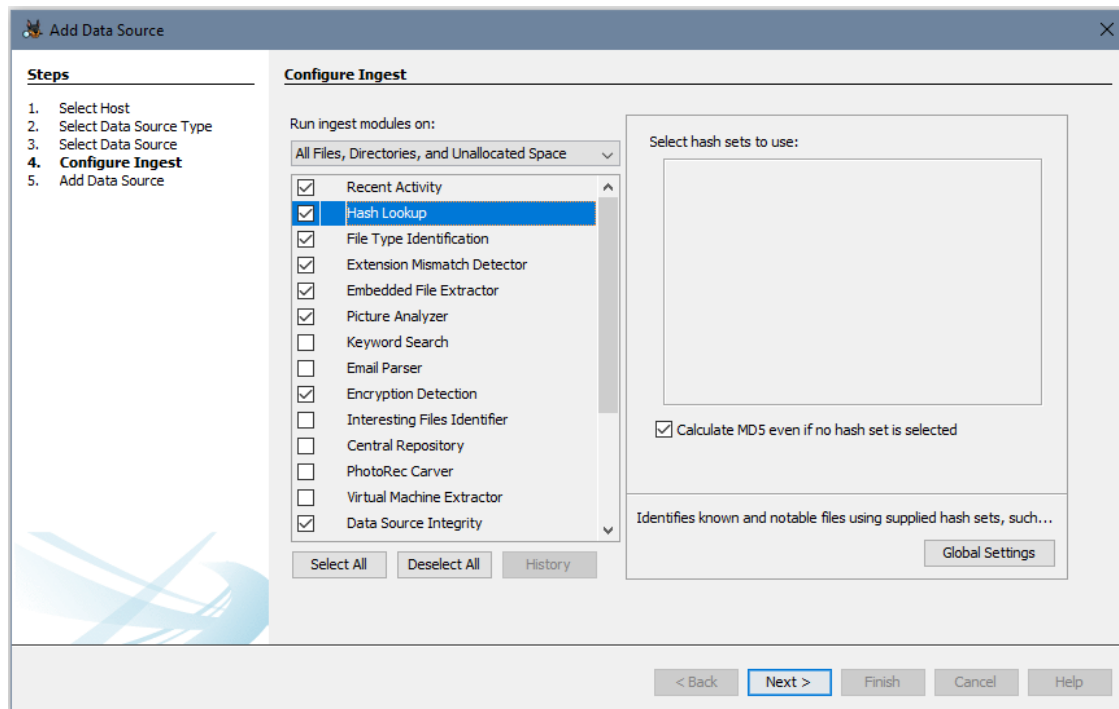


Figure 17 - Autopsy Ingest Modules

Autopsy proceeds to process the ingest modules on the selected files and makes them available to us through the application from which we will draw our results.

Results: The output of the embedding algorithm after embedding the text file to the image is shown in Figure 18. It is interesting to witness how far least significant bit steganographic algorithms have evolved as the visual artifacts present in the steganographic image could be categorized as almost nonexistent. To the human eye the steganographic image holds an noticeable resemblance to its cover carrier and only by other computational means could we tell they are not the same image despite looking the same.

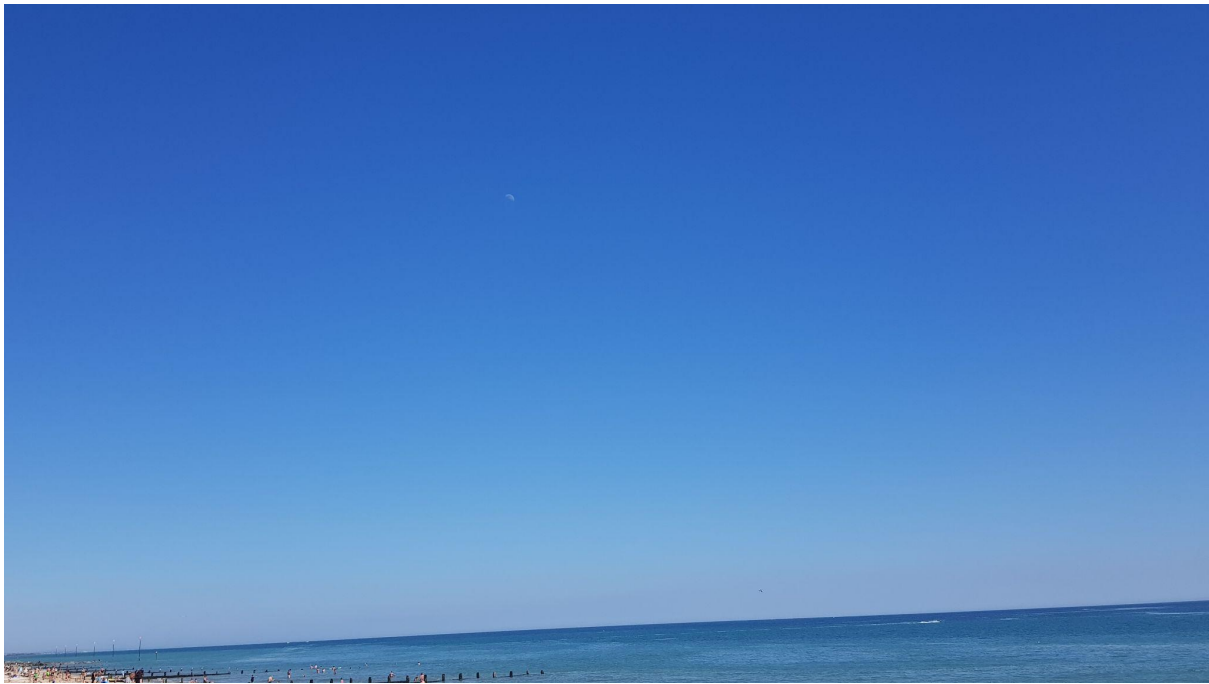


Figure 18 - Steganographic Image Carrier

Name: sky_stego.jpg

Size: 1.29 MB

Dimensions: 4032x2268

Horizontal resolution: 96 dpi

Vertical resolution: 96 dpi

Bit depth: 24

Colour representation: unavailable

Compressed bits/pixel: unavailable

Based on the image properties we can deduce steghide's least significant bit steganography algorithm has compressed the image while adding additional artifacts to the image, the image size has been reduced by 0.01 MB while its dots per inch (DPI) have increased to 96 dpi from 72. Additionally, the hash sets created by Autopsy guarantees that sky.jpg and sky_stego.jpg are not the same image. Hash functions map data of arbitrary size to fixed-size values and are commonly used for the undeniable authentication of a file in digital forensics. We used Autopsy to calculate the MD5 Hash and SHA-256 Hash of both the cover carrier and the steganographic carrier to prove their difference.

sky.jpg

MD5 Hash: f79f2cc57484a3e66d2006c630397547

SHA-256 Hash: c79bab0c6b5a08b7f8d1747848eb7c42b963f742afd218ea88971ec4e980f2f1

sky_stego.jpg

MD5 hash: 8ac406013c23ef44e2b65eb1149cd6dc
SHA-256 Hash: 82ca34ea625eb6dbcbf4bde0ade04be467c894d5da57bd8372bf2e7f3646426f

Under further inspection, only the EXIF metadata of the cover carrier image could be generated (Figure 19). We expected the embedding algorithm to overwrite data it deemed unnecessary to maintain the quality of the image due to the nature of the least significant bit process. A side by side comparison of the hexdump of both the cover carrier and the steganographic carrier allowed us to find critical differences in both files composition. Binary streams that make up the EXIF information of the image were in fact overwritten, these were notably identified as the date of creation, the latitude and longitude of where the image was taken and the device used to take the image (Figure 20 & 21). We also identified that entire binary stream sections that were previously empty on the cover carrier were now filled with additional data (Figure 22 & 23). Additionally the signature of the file header morphed to the JFIF image file format from EXIF, any application that supports images can recognize both image formats but these two standards are mutually incompatible (Figure 24 & 25). The hex distribution statistics also indicate that the count for each hex character has massively increased compared to the cover image carrier, possibly hinting at the addition of contents of the 24,261 words text file (Figure 26 & 27).

Listing									
EXIF Metadata									
Table Thumbnail Summary									
Source File	S	C	O	Date Created	Latitude	Longitude	Altitude	Device Model	Device Make
sky.jpg			1	2020-05-30 16:18:08 BST	50.80277777777778	-0.6683333333333333	0.0	SM-G955F	samsung

Figure 19 - Autopsy Fails to Generate sky_stego.jpg EXIF Metadata

00000090	04 00 01 00 00 00 B6 03 00 00 00 00 00 00 48 00I.....H.
000000A0	00 00 01 00 00 00 48 00 00 00 01 00 00 00 73 61H.....sa
000000B0	6D 73 75 6E 67 00 53 4D 2D 47 39 35 35 46 00 00	msung.SM-G955F..
000000C0	47 39 35 35 46 58 58 53 37 44 54 41 36 00 32 30	G955FXXS7DTA6.20
000000D0	32 30 3A 30 35 3A 33 30 20 31 35 3A 31 38 3A 30	20:05:30 15:18:0
000000E0	38 00 24 00 9A 82 05 00 01 00 00 00 8C 02 00 00	8.\$.\$,.....E...
000000F0	9D 82 05 00 01 00 00 00 94 02 00 00 22 88 03 00	.,....."..."^...

Figure 20 - Cover Image Carrier - EXIF Metadata Binary Stream

00000090	08 08 08 08 08 08 08 08 08 08 08 08 08 08 08 08 FF C0ÿÀ
000000A0	00 11 08 08 DC 0F C0 03 01 22 00 02 11 01 03 11Ü.À..".....
000000B0	01 FF C4 00 1F 00 00 01 05 01 01 01 01 01 01 00	..ÿÀ.....
000000C0	00 00 00 00 00 00 00 01 02 03 04 05 06 07 08 09
000000D0	0A 0B FF C4 00 B5 10 00 02 01 03 03 02 04 03 05	..ÿÀ.µ.....
000000E0	05 04 04 00 00 01 7D 01 02 03 00 04 11 05 12 21}.....!
000000F0	31 41 06 13 51 61 07 22 71 14 32 81 91 A1 08 23	lÀ..Qa."q.2.`i. #

Figure 21 - Steganographic Image Carrier - Overwritten EXIF Metadata Binary Stream

C:\Users\Jean\Desktop\Steganography\sky.jpg																		
Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text	
000015C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000015D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000015E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000015F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001600	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001610	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001620	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001630	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001640	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001650	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001660	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001670	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001680	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001690	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000016A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000016B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000016C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000016D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000016E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000016F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001700	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001710	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001720	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001730	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001740	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001750	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001760	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001770	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001780	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001790	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000017A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000017B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000017C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000017D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000017E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000017F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001800	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00001810	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	

Figure 22 - Cover Image Carrier - Offset x000015C0 to x00001710 Binary Stream

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
000015C0	58	17	19	1C	50	57	CC	8D	64	DB	B4	7E	94	91	B9	C2	X...PWÌ.dÛ'~""'À
000015D0	06	CE	3F	A5	26	84	28	4F	E3	44	F9	3A	FB	52	CA	C1	.Î?¥\$, (OâDû:ûREÄ
000015E0	4B	C4	18	17	E3	3E	9E	BF	8D	46	FB	88	79	01	64	4C	KÄ..ä>žç.Fû'y.dL
000015F0	8E	94	E5	5C	39	FB	A8	BD	4F	AD	0B	61	8F	42	16	32	Ž"â\9û""O..a.B.2
00001600	F7	03	EE	91	B4	77	6F	EB	4C	56	65	66	DB	1F	98	0F	÷.i'woëLVerÛ.~.
00001610	61	F2	E0	D4	BB	71	26	F7	2C	57	B0	3F	78	FE	1D	A9	aòâÖ»q&÷,W°?xp.©
00001620	84	66	7C	82	47	B5	1D	00	52	86	45	C0	72	B2	35	46	„f ,Gu..R+EÄr°5F
00001630	E4	C7	C9	90	37	B7	7A	7B	EE	19	49	0E	E3	9C	63	DE	äÇË.7.z{.I.I.äæcP
00001640	A2	10	BB	30	C8	1B	07	AF	35	08	09	10	06	91	3F	77	c.»0Ë..~5....'w
00001650	96	FE	1E	C2	8F	9F	95	19	C8	6E	47	B5	2F	42	C8	19	-p.Ä.ÿ°.ËnGu/BE.
00001660	C0	FA	70	3D	29	CA	C0	64	86	00	67	E6	C7	39	AD	00	Àúp=)ÈÄdt.gæÇ9..
00001670	42	86	35	0A	57	11	B5	46	C1	90	AA	11	95	C7	DE	A7	Bt+5.W.pFÄ.°.ÇP\$
00001680	26	18	3B	6F	25	BE	EF	3D	BF	C2	9C	02	0E	37	7C	B9	æ.;o%ûi=çÄæ..7 '.
00001690	FC	AB	2B	00	C8	80	0A	C7	12	13	FA	54	A6	20	12	32	û«+.ÈË.Ç..úT .2
000016A0	4A	B0	D9	BB	E9	FF	00	D7	A4	3F	21	11	22	B1	1F	DE	ÿ°Üæÿ.×«?!."±.P
000016B0	1D	BE	94	92	73	2E	01	20	74	03	BD	6D	10	18	42	B4	.%'"s.. t.âm..B'
000016C0	AA	4C	6A	A7	6E	CE	BD	7B	D3	D6	15	52	58	E4	49	4C	*Lj\$niÿs(ÖÖ.RXaIL
000016D0	C2	92	58	82	48	E0	7A	53	9B	08	BD	0E	FF	00	6E	68	Ä'X.HâzS>\$.ÿ.nh
000016E0	19	1A	FE	EC	9C	46	CC	5B	AD	28	4C	92	87	28	83	93	..p æFÏ . (L'+(f"
000016F0	DB	9F	5A	98	13	26	E8	D7	92	47	7E	D5	0E	D2	71	E5	ÜÿZ".æè×'G~Ö.Öqâ
00001700	E5	FB	60	F4	E2	97	30	08	40	28	70	43	73	C7	D2	9B	âû`ôâ-0.0(pCsÇÖ>
00001710	E5	98	C0	C7	EF	3D	3E	95	21	19	7C	12	CA	7D	69	5D	â`ÄÇi=>!. .Ë)il
00001720	58	6D	44	00	9C	E5	69	88	64	AA	EC	E3	07	38	A3	39	XmD.æâi`d°iâ.8£9
00001730	2E	CE	17	77	41	4E	20	86	7D	E7	8E	39	1D	4D	28	1C	.Î.wAN t)çŽ9.M(.
00001740	96	07	0D	D2	80	23	DE	4A	90	41	20	F3	ED	4A	AB	E6	-..Öæ#PJA óiJææ
00001750	1C	04	0A	BD	73	EB	4F	DA	3A	0C	85	F4	F4	A9	00	24	...sæëOÛ: ...ôôö.\$
00001760	FC	BC	B6	7B	50	04	0C	18	87	E0	30	FC	EA	31	85	DC	û¼¶(P...tàOûêL..Û
00001770	87	05	78	CF	B5	4C	F1	3A	E0	1D	BE	DF	FD	7F	6A	4D	+.%ÏpLñ:â.%âÿ.jM
00001780	A0	C9	E5	A2	16	F5	3E	F4	00	C4	5D	AE	DB	00	2A	79	Éâc.ô>ô.Ä]øÛ.*y
00001790	03	FB	D5	33	80	E4	B1	8C	EF	E9	E8	69	00	5F	99	94	.ûÖ3Ëâ±Çiëei. "m
000017A0	E0	8F	E7	48	11	89	DF	19	E3	A8	07	FC	F5	A9	9C	80	â.çH.%B.â".ûöææ
000017B0	42	A5	80	D8	48	39	E6	86	F3	03	86	56	50	A3	1B	97	B¥€ØH9ætó.+VPÄ.-
000017C0	D6	94	60	95	DC	E7	79	CF	14	A5	5D	C8	67	00	1E	9E	Ö""ÜçÿI.¥]Ëg..ž
000017D0	9C	51	16	04	4C	50	B1	7C	E1	47	1C	2F	51	41	2A	18	æQ..LP± âG./QA*.
000017E0	BF	FC	B3	F4	ED	4E	7C	8D	CC	80	61	7A	60	77	A7	14	çü°ôin .Ëæaz`wS.
000017F0	62	81	B3	97	F4	34	58	42	FF	00	AB	25	79	77	23	3E	b.'-ô4XBÿ.æÿw#>
00001800	B4	A0	97	53	84	28	98	E9	E9	4F	88	90	37	34	79	18	'-S„ (~ééO^74y.
00001810	F9	49	E2	82	58	28	2C	C3	69	F4	5F	FD	0A	94	9E	83	ùIâ,X(,Äiô_ÿ."žf

Figure 23 - Steganographic Image Carrier - Offset x000015C0 to x00001710 Binary Stream

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	FF	D8	FF	E1	04	8C	45	78	69	66	00	00	49	49	2A	00	ÿøÿä..EXif...II*.
00000010	08	00	00	00	0B	00	0F	01	02	00	08	00	00	00	A2	00	

Figure 24 - Cover Image Carrier Header - 8C 45 78 69 66 EXIF Digital Signature

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	FF	D8	FF	E0	00	10	4A	46	49	46	00	01	01	00	00	01	ÿøÿä..JFIF.....
00000010	00	01	00	00	FF	DB	00	43	00	01	01	01	01	01	01	01	ÿÿÿÿ

Figure 25 - Steganographic Image Carrier Header - 4A 46 49 46 JFIF Digital Signature

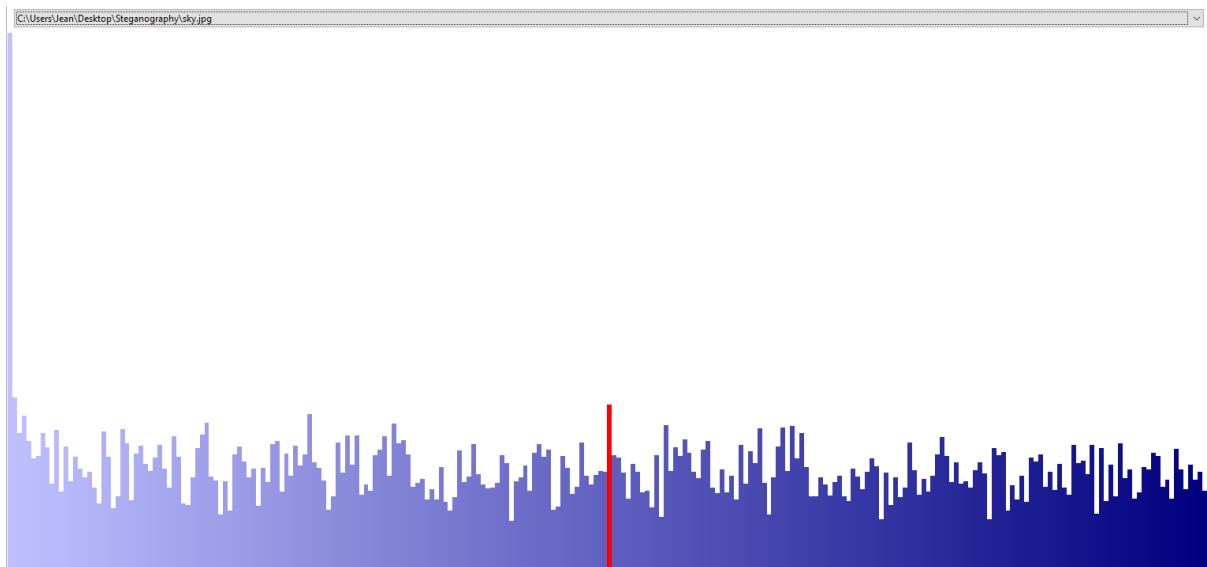


Figure 26 - Cover Image Carrier - Hex Distribution Statistics

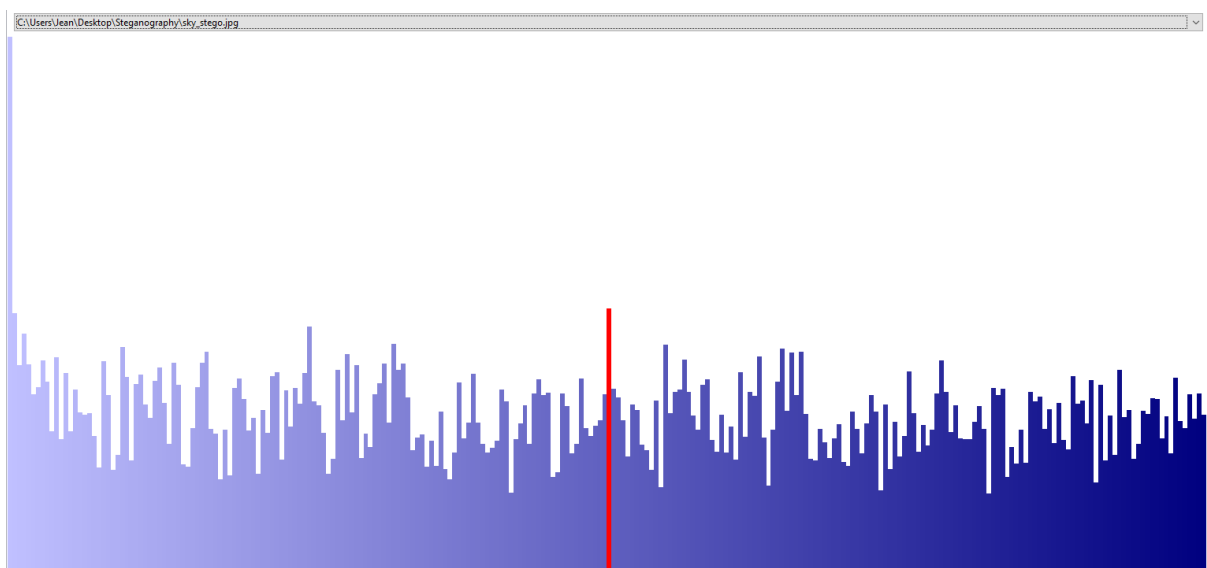


Figure 27 - Steganographic Image Carrier - Hex Distribution Statistics

The major highlight of this experiment is that while looking for additional artifacts in the steganographic image carrier we discovered that despite Steghide advertising its least significant bit algorithm embedding process to replace LSB bits at random, it is not entirely random. This observation began with the idea to try and match the embedded text file's name "Shakespeare " by calculating ASCII to Hexadecimal.

Shakespeare = 53 68 61 6b 65 73 70 65 61 72 65 0a

We attempted to match the hexadecimal values to values present in the steganographic image carrier with no luck, meaning that Steghide's algorithm is not truly sequential. Then we scaled down our search to pairs bytes to disprove the possibility of getting a random single character match. To our surprise Steghide's LSB embedding algorithm is not truly random either. The algorithm does seem to allocate replaced bytes at somewhat random locations but it does so by sequentially pairing two bytes together. Due to this vulnerability in the software's embedding algorithm we were able to reconstruct not only the file name hidden by Steghide but also entire portions of text present inside the Shakespeare.txt file (Figure 28). Steghide is considered a top 10 steganography software but it raises many security concerns as it promises steganography resistance to first-order statistical tests.

```

00007870 1B 25 2A 53 68 1C 71 DE 82 62 C7 E4 E4 B0 E4 74 .%*Sh.qP,bÇää°ät
001273C0 A9 EE 79 FD 6A A2 36 CB 76 FA 2E 9E 61 6B 5B 8B @iyyjç6Ėvú.žak[<
00012A60 FC 34 0C 6E 0E C2 58 65 73 F9 53 C8 0B B4 8C E4 ü4.n.ÄXesàSÈ.‘Gä
00005B90 19 18 F4 A9 92 27 21 90 80 17 FC F3 42 18 D7 70 ..ô@'!'!€.üóB.*p
00005BA0 65 25 FA 67 D4 50 03 76 81 D3 39 E5 68 2A 77 94 ešúgÔP.v.Ó9âh*w"
00036950 65 98 8E 07 F7 A9 65 04 61 72 03 0E 7F 0A 00 48 e~Ž.-@e.ar.....H
00032F60 B4 61 43 65 0A 85 A5 55 93 92 0B 28 F5 F5 A9 3C 'aCe...ŸU"".(ôô@<

```

Figure 28 - Steganographic Image Carrier - Reconstruction of the embedded file name

5.2 Experiment 2: Audio Cover Carrier vs. Audio Steganographic Carrier.

Purpose of the experiment: The goal of this experiment is to determine through conventional dead analysis changes made by the least significant bit steganography algorithm of DeepSound to the original audio cover carrier after embedding our steganographic content.

Method of the experiment: We embed the A New Beginning.mp3 audio file with 24,261 words using the DeepSound LSB embedding algorithm. The audio cover carrier and the steganographic carrier are then both introduced to the digital forensics platform for analysis.

Detail of experiment: The audio file A New Beginning.mp3 was chosen as our audio carrier, the file size (2.06 MB) of the audio file was suitable for this experiment under the assumption that the same principles of image carriers apply to other file types. Based on our literature review the larger the spatial domain the more data it is possible to embed inside the carrier. Audio carriers do not possess EXIF data but least significant bit algorithms, whether random or sequential, will still attempt to overwrite bits that are not relevant to the resulting steganographic carrier's audio quality. We then embedded our cover audio carrier with a 24,261 words text file using DeepSound least significant bit embedding algorithm. Within the software we open our audio carrier of choice then provide the text file we want the LSB algorithm to embed inside our carrier.

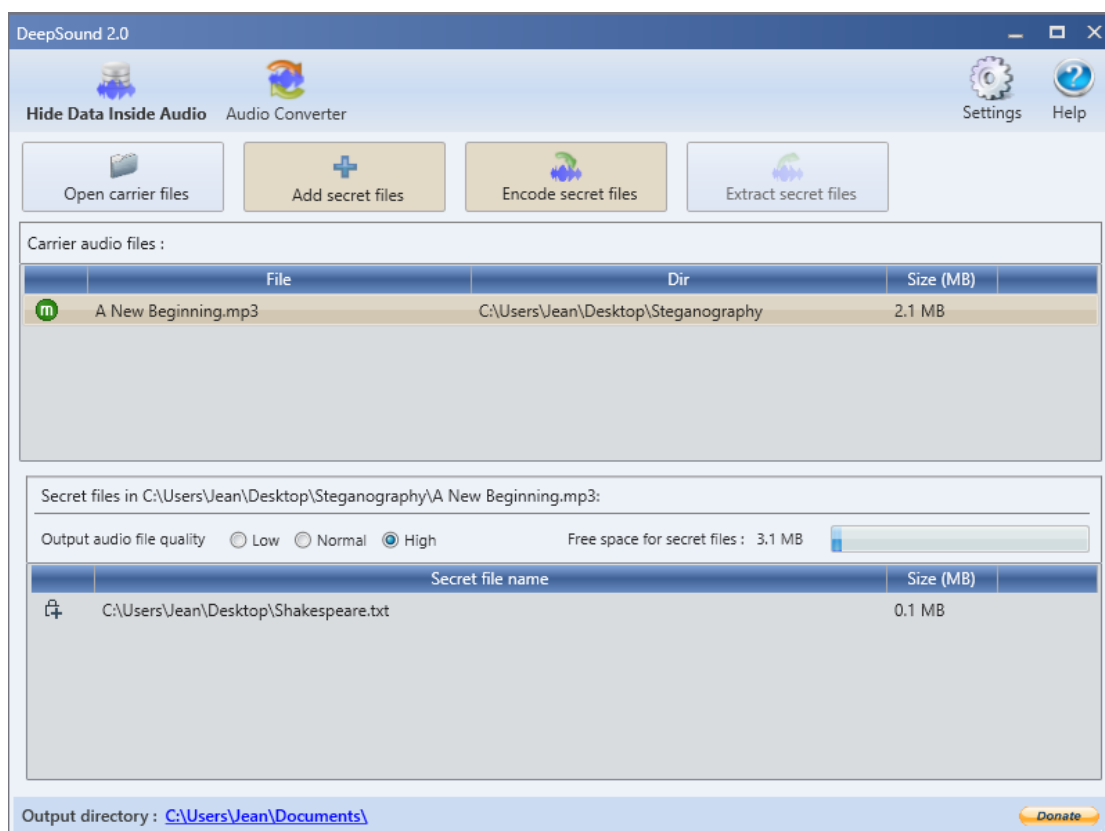


Figure 29 - DeepSound LSB Encoding

The steganographic carrier was renamed to A New Beginning_stego to differentiate between the cover carrier and the stego carrier. Both files were introduced to the forensic platform environment for further inspection. The process of adding data sources to the forensic platform remained unchanged, please refer to Experiment 1 for reference.

Results: The output of the embedding algorithm after embedding the text file to the audio carrier is available using the following link:

<https://drive.google.com/file/d/1KzYlxtiPUcgNKsC-2DZAgH-YkixaeSJk/view?usp=sharing>

To the human ears the steganographic audio carrier also holds a noticeable resemblance to its cover carrier and only by other computational means could we tell they are not the same audio file despite sounding the same.

Name: A New Beginning_stego.wav

Size: 26 MB

Duration: 2:34

If we take a look at audio file properties the only significant change is that DeepSound's least significant bit algorithm decompresses the .mp3 file type and converts it into the .wav file type which is a raw audio format. Additionally, the hash sets created by Autopsy guarantees that A New Beginning.mp3 and A New Beginning_stego.wav are not the same audio file. We used Autopsy to calculate the MD5 Hash and SHA-256 Hash of both the cover carrier and the steganographic carrier to prove their difference.

A New Beginning.mp3

MD5 Hash: 8d647f68d81408dde5c4e1ed7dfb8527

SHA1 Hash: 8c861242a5dd94b21f2d61ebe172b919d46a51768ae325ce70021a8504bf6f5f

A New Beginning.wav

MD5 Hash: 1d9752707bcd524b4cb7bfdbb72559b7

SHA1 Hash: bb0b31b93cd784ecbda5793d43601cb701a14432eaa56aa22d729feff0002e9e

We expected the embedding algorithm to overwrite data it deemed unnecessary to maintain the quality of the audio due to the nature of the least significant bit process but instead DeepSound decompresses the mp3 format which may not seem like much at first but by doing so mp3 artifacts are purged from the audio file expanding its writable spatial domain without increasing the file size. The phenomenon can also be seen through the hex distribution statistics proving our previous assumption. (Figure 30 & 31). A side by side comparison of the hexdump of both the cover carrier and the steganographic carrier shows us that binary stream sections that previously contained data on the cover carrier were now zeroed out on the steganographic carrier (Figure 32 & 33). We could not reconstruct the carrier's contents based on artifacts created by DeepSound's embedding algorithm as we did for the image. The analysis of audio encoding remains very challenging in comparison to image analysis.

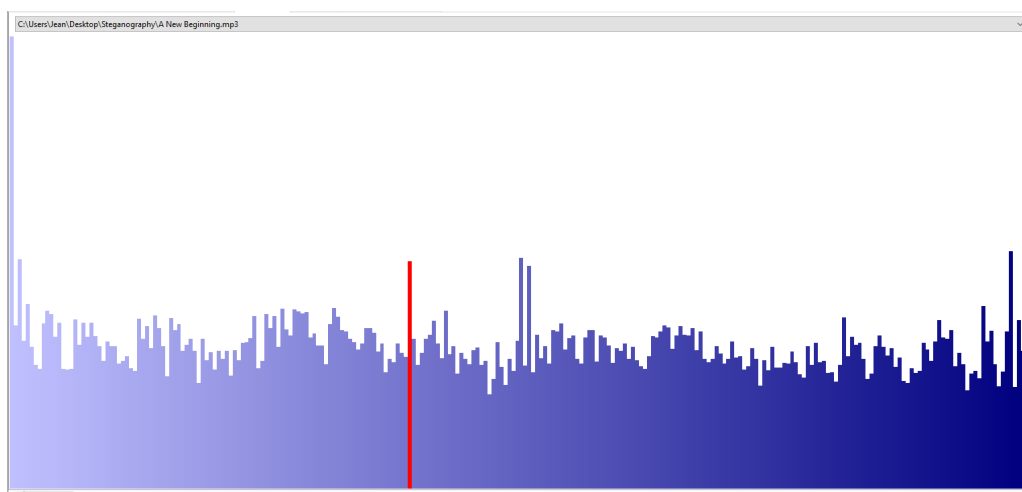


Figure 30 - Cover Audio. mp3 Carrier - Hex Distribution Statistics



Figure 31 - Steganographic Audio. wav Carrier - Hex Distribution Statistics

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00211940	61	85	A0	2C	79	52	C5	48	84	C6	50	15	8B	01	7E	02	a...yRÄH,EP.<~.
00211950	11	15	22	76	2A	31	1B	06	7A	80	A2	50	E9	D0	E8	6C	..v*1...z€cPéDèl
00211960	80	1B	64	BE	9C	AD	C6	A1	15	21	8E	43	0E	AD	17	45	€.d%æ.Ej;!ŽC...E
00211970	FA	D1	F6	0C	5A	3E	C1	9A	70	99	9F	F0	90	15	83	12	úNö.Z>Äsp™Yø...f.
00211980	48	2A	E5	12	D4	96	EA	46	5B	FD	5A	49	05	1F	58	09	H*Ä.Ö-èF[yZI..X.
00211990	3F	C5	95	4C	41	4D	45	33	2E	39	39	2E	35	55	55	55	?Ä•LAME3.99.5UUU
002119A0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
002119B0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
002119C0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
002119D0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
002119E0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
002119F0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211A00	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211A10	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211A20	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211A30	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211A40	55	55	55	55	55	55	55	55	55	55	FF	FB	82	44	9E	0F	UUUUUUUUUUyâ,Dž.òž
00211A50	36	C0	50	C1	1C	68	32	80	08	6D	04	23	01	00	00	01	èÄPÄ.h2€..m.#....
00211A60	A4	00	00	00	20	00	00	00	34	80	00	00	04	55	55	55	U... ..4€...UUUU
00211A70	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211A80	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211A90	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211AA0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211AB0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211AC0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211AD0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211AE0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211AF0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211B00	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211B10	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211B20	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211B30	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211B40	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211B50	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211B60	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211B70	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211B80	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211B90	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211BA0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUUUUUUUUUUUUUU
00211BB0	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	55	UUUUUU

Figure 32 - Cover Audio. mp3 Carrier - End of file binary stream

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
01A11F70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A11F80	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A11F90	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A11FA0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A11FB0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A11FC0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A11FD0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A11FE0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A11FF0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12000	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12010	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12080	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12090	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A120A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A120B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A120C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A120D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A120E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A120F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12100	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12110	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12120	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12130	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12140	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12150	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12160	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12170	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12180	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A12190	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A121A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A121B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A121C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A121D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
01A121E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Figure 33 - Steganographic Audio. wav Carrier - End of file binary stream

5.3 Experiment 3: Video Cover Carrier vs. Video Steganographic Carrier.

Purpose of the experiment: The goal of this experiment is to determine through conventional dead analysis changes made by the least significant bit steganography algorithm of OpenPuff to the original video cover carrier after embedding our steganographic content.

Method of the experiment: We embed the rain.mp4 video file with 24,261 words using the OpenPuff LSB embedding algorithm. The video cover carrier and the steganographic carrier are then both introduced to the digital forensics platform for analysis.

Detail of experiment: The video file rain.mp4 was chosen as our video carrier, OpenPuff requires its carriers to have a minimum of 16 carrier bytes per carrier to avoid presenting visual or auditory artifacts in the resulting steganographic video. The rain.mp4 video file was big enough to meet OpenPuff's requirements (258MB). We are also required to provide a 8 to 32 character password with the option of providing two more passwords for multi-layered data obfuscation. The video file was suitable for this experiment under the assumption that the same principles of image carriers apply to other file types. OpenPuff uses an adaptive non-linear carrier bit encoding, the algorithm will still overwrite bits that are not relevant to the resulting steganographic carrier's video quality. The main difference when compared to image and audio carriers is that LSB on videos will work similar to watermarking where the information hidden is embedded into the frames of the video. We embedded our cover video carrier with a 24,261 words text file using the OpenPuff least significant bit embedding algorithm. Within the software we provide our password of choice then provide the text file we want the LSB algorithm to embed inside our carrier as well as a suitable carrier that meets OpenPuff's requirements (Figure 34).

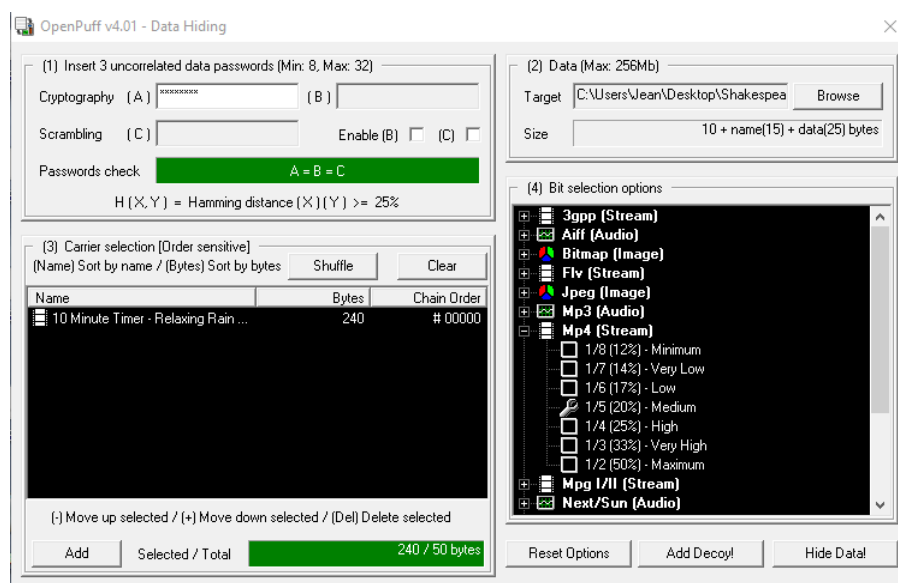


Figure 34 - OpenPuff LSB Encoding

The steganographic carrier was renamed to rain_stego to differentiate between the cover carrier and the stego carrier. Both files were introduced to the forensic platform environment for further inspection. The process of adding the data sources to the forensic platform remained unchanged, please refer to Experiment 1 for reference.

Results: The output of the embedding algorithm after embedding the text file to the video carrier is available using the following link:

https://www.youtube.com/watch?v=mvHvJ_Ka2mY

To the human sensory system the output video is not distinguishable from the original carrier in terms of audio and video and only by other computational means could we tell they are not the same video file despite sounding and visually looking the same to the naked eye.

Name: rain_stego.mp4

Size: 92.6 MB

Duration: 10 minutes 3 seconds

Dimensions: 1280x720

Bitrate: 1280 Kbps

Frame rate: 30 frames/second

Channels: 2 (stereo)

Audio Bitrate: 127 Kbps

Audio Sample rate: 44.100 kHz

Following the video's file properties there is no trace that OpenPuff's least significant bit algorithm made any change to the file at all. However, the hash sets created by Autopsy guarantee that rain.mp4 and rain_stego.mp4 are not the same video file. We used Autopsy to calculate the MD5 Hash and SHA-256 Hash of both the cover carrier and the steganographic carrier to prove their difference.

rain.mp4

MD5 Hash: e21807d18937c53f2d275b28e9de0158

SHA1 Hash: cfadd3919b62acc2bd99d55bf4a69976871f4b4796a81a9306f516499bc2c66c

rain_stego.mp4

MD5 Hash: c602b0ee5ee398f5623a63c5c6ffc824

SHA1 Hash: 04cf3674ba1768da7aaef94f18d586c6e8ec4a0b3a3eb24a57f3421290864803

After comparing and analyzing byte streams of both the original video carrier and the video steganographic carrier we can confirm that the embedding algorithm of OpenPuff remains very subtle. We expected the embedding algorithm to overwrite data aggressively similar to Steghide and DeepSound. A side by side comparison of the hexdump of both the cover carrier and the steganographic carrier shows us that binary stream sections remain almost identical except minimal byte type difference fields (Figure 35 & 36). This could be an indication that OpenPuff's LSB algorithm hides the data in redundant MP4 fields. As discussed in our literature review, least Significant Bit Matching techniques compare each bit of the content against the least significant bit of the corresponding carrier's cover byte and as long as the compared bits match, no change is made to the carrier. However, hex distribution statistics show an abnormal increase in the use of each hex value on the steganographic carrier (Figure 37 & 38). We could not reconstruct the carrier's contents based on artifacts created by OpenPuff's embedding algorithm as we did for the image. The analysis of video encoding remains very challenging in comparison to image and audio analysis.

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	00	00	00	18	66	74	79	70	6D	70	34	32	00	00	00	00	...ftypmp42...
00000010	69	73	6F	6D	6D	70	34	32	00	03	CE	BE	6D	6F	6F	76	isommp42..f%moov
00000020	00	00	00	6C	6D	76	68	64	00	00	00	00	DB	A8	80	19	...lmvhd....Û"€.
00000030	DB	A8	80	19	00	00	03	E8	00	09	3A	72	00	01	00	00	Û"€....è.:r...
00000040	01	00	00	00	00	00	00	00	00	00	00	00	00	01	00	00
00000050	00	00	00	00	00	00	00	00	00	00	00	00	00	01	00	00
00000060	00	00	00	00	00	00	00	00	00	00	00	00	40	00	00	00@...
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000080	00	00	00	00	00	00	00	00	00	00	00	03	00	02	20	60`
00000090	74	72	61	6B	00	00	00	5C	74	6B	68	64	00	00	00	03	trak...\tkhd...
000000A0	DB	A8	80	19	DB	A8	80	19	00	00	00	01	00	00	00	00	Û"€..Û"€.....
000000B0	00	09	3A	5E	00	00	00	00	00	00	00	00	00	00	00	00	...^.....
000000C0	00	00	00	00	00	01	00	00	00	00	00	00	00	00	00	00
000000D0	00	00	00	00	00	01	00	00	00	00	00	00	00	00	00	00
000000E0	00	00	00	00	40	00	00	00	05	00	00	00	02	D0	00	00@.....@..
000000F0	00	00	00	24	65	64	74	73	00	00	00	1C	65	6C	73	74	...\$edts....elst
00000100	00	00	00	00	00	00	00	01	00	09	3A	5F	00	00	02	00:.....
00000110	00	01	00	00	00	02	1F	D8	6D	64	69	61	00	00	00	20Ømdia...
00000120	6D	64	68	64	00	00	00	00	DB	A8	80	19	DB	A8	80	19	mdhd....Û"€..Û"€.
00000130	00	00	3C	00	00	8D	BE	00	55	C4	00	00	00	00	00	5F	..<...%.UÄ.....
00000140	68	64	6C	72	00	00	00	00	00	00	00	00	76	69	64	65	hdlr.....vide
00000150	00	00	00	00	00	00	00	00	00	00	00	00	49	53	4F	20ISO
00000160	4D	65	64	69	61	20	66	69	6C	65	20	70	72	6F	64	75	Media file produ
00000170	63	65	64	20	62	79	20	47	6F	6F	67	6C	65	20	49	6E	ced by Google In
00000180	63	2E	20	43	72	65	61	74	65	64	20	6F	6E	3A	20	31	c. Created on: l
00000190	30	2F	31	31	2F	32	30	32	30	2E	00	00	02	1F	51	6D	0/11/2020.....Qm
000001A0	69	6E	66	00	00	00	24	64	69	6E	66	00	00	00	1C	64	inf...\$dinf....d
000001B0	72	65	66	00	00	00	00	00	00	01	00	00	00	00	0C	75	ref.....u
000001C0	72	6C	20	00	00	00	01	00	02	1F	11	73	74	62	6C	00	rlstbl.

Figure 35 - Cover Video Carrier - Byte type difference example

	FD AO	Rain.mp4																FD AO	Rain_stego.mp4																Statistics	Statistics
Offset (h)		00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text																		
00000000		B0	00	00	18	66	74	79	70	6D	70	34	32	00	00	00	00	[...ftypmp42....																		
00000010		69	73	6F	6D	6D	70	34	32	00	03	CE	BE	6D	6F	6F	76	isommp42..f%moov																		
00000020		00	00	00	6C	6D	76	68	64	00	00	54	02	DB	A8	80	19	...lmvhd..T.Û"€.																		
00000030		DB	A8	80	19	00	00	03	E8	00	09	3A	72	00	01	00	00	Û"€.....è...r....																		
00000040		01	00	00	00	00	00	00	00	00	00	00	00	00	00	01	00																		
00000050		00	00	00	00	00	00	00	00	00	00	00	00	00	00	01	00																		
00000060		00	00	00	00	00	00	00	00	00	00	00	00	40	00	00	00@...																		
00000070		00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00																		
00000080		00	00	00	00	00	00	00	00	00	00	00	03	00	02	20	60`																		
00000090		74	72	61	6B	00	00	00	5C	74	6B	68	64	00	00	00	03	trak...\tkhd....																		
000000A0		DB	A8	80	19	DB	A8	80	19	00	00	00	00	01	00	00	00	Û"€..Û"€.....																		
000000B0		00	09	3A	5E	00	00	00	00	00	00	00	00	00	00	00	00	...:^.....																		
000000C0		00	00	00	00	00	01	00	00	00	00	00	00	00	00	00	00																		
000000D0		00	00	00	00	00	01	00	00	00	00	00	00	00	00	00	00																		
000000E0		00	00	00	00	40	00	00	00	05	00	00	00	02	D0	00	00@.....D..																		
000000F0		00	00	00	24	65	64	74	73	00	00	00	1C	65	6C	73	74	...\$edts....elst																		
00000100		00	00	08	24	00	00	00	01	00	09	3A	5F	00	00	02	00	...\$.....:....																		
00000110		00	01	00	00	00	02	1F	D8	6D	64	69	61	00	00	00	20Ømdia...																		
00000120		6D	64	68	64	00	04	04	00	DB	A8	80	19	DB	A8	80	19	mdhd....Û"€..Û"€.																		
00000130		00	00	3C	00	00	8D	BE	00	55	C4	00	00	00	00	00	5F	..<...%.UÅ.....																		
00000140		68	64	6C	72	00	04	80	00	00	00	00	00	76	69	64	65	hdlr..€.....vide																		
00000150		00	00	00	00	00	00	00	00	00	00	00	00	49	53	4F	20ISO																		
00000160		4D	65	64	69	61	20	66	69	6C	65	20	70	72	6F	64	75	Media file produ																		
00000170		63	65	64	20	62	79	20	47	6F	6F	67	6C	65	20	49	6E	ced by Google In																		
00000180		63	2E	20	43	72	65	61	74	65	64	20	6F	6E	3A	20	31	c. Created on: l																		
00000190		30	2F	31	31	2F	32	30	32	30	2E	00	00	02	1F	51	6D	0/11/2020.....Qm																		
000001A0		69	6E	66	00	00	00	24	64	69	6E	66	00	00	00	1C	64	inf...\$dinf....d																		
000001B0		72	65	66	00	01	08	80	00	00	00	01	00	00	00	0C	75	ref...€.....u																		
000001C0		72	6C	20	00	00	00	01	00	02	1F	11	73	74	62	6C	00	rlstbl.																		

Figure 36 - Steganographic Video Carrier - Byte type difference example

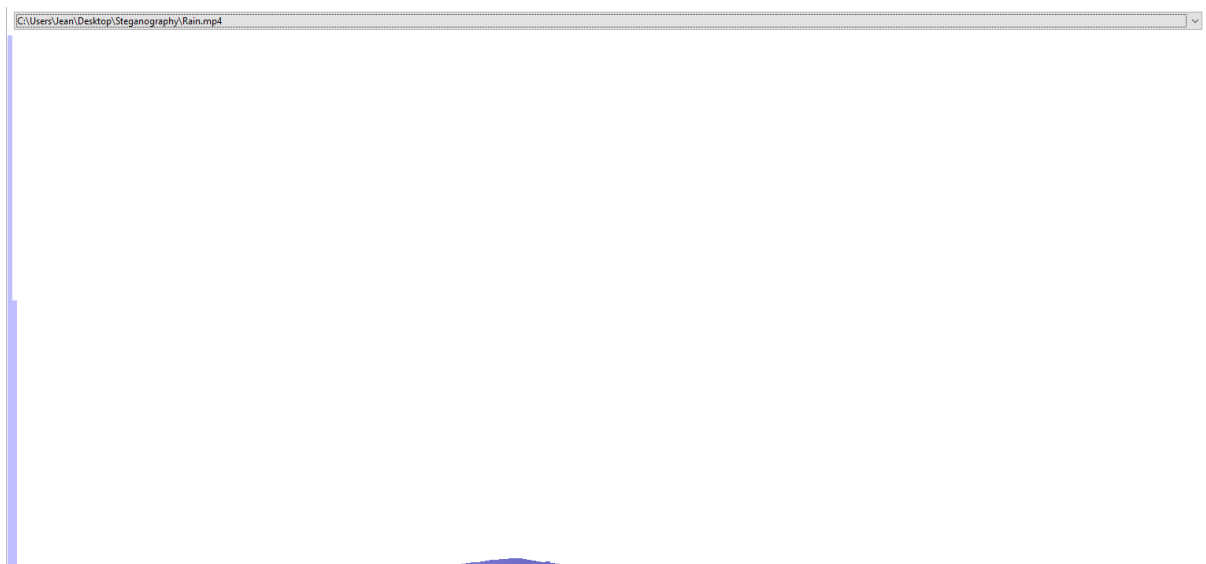


Figure 37 - Cover Video Carrier - Hex Distribution Statistics

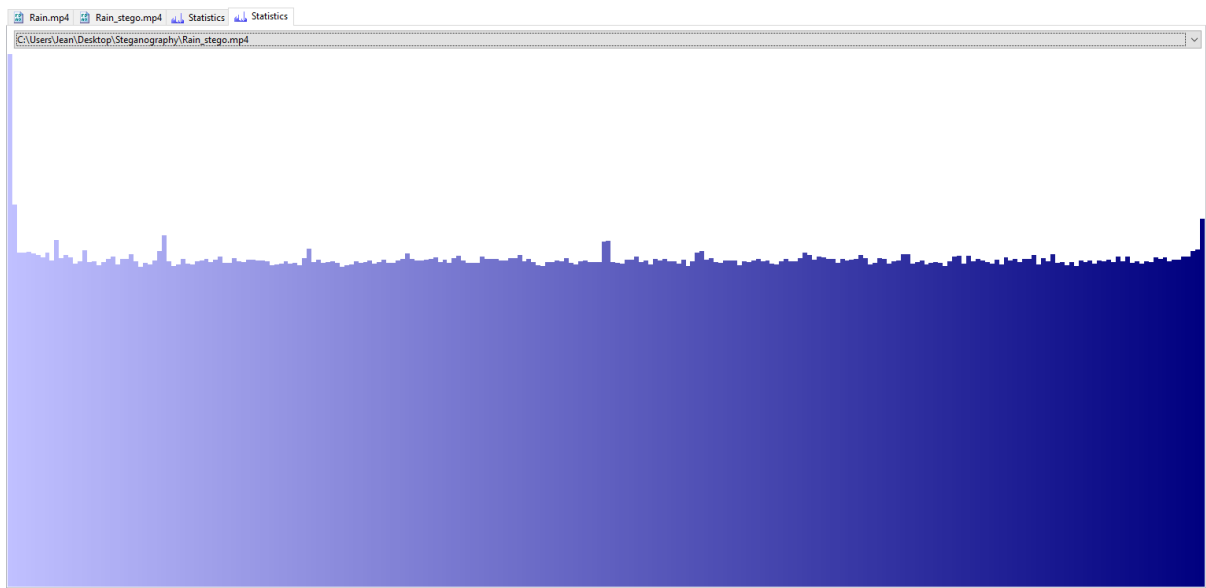


Figure 38 - Steganographic Video Carrier - Hex Distribution Statistics

6 Discussion of Results & Solutions

6.1 Is it possible to identify key indicators and digital forensic artefacts on a media file that has gone through a steganographic process?

It is possible. This project's experiments demonstrate that despite a least significant bit embedding algorithm's efforts to obfuscate data behind a carrier there will always be changes caused by the steganographic process. Modern steganographic algorithms will have requirements to ensure that their output's quality can bypass a human's sensory system, because of this cyber forensic examiners must dig deeper into the binary composition of a file. Forensic artifacts will vary depending on the steganographic carrier. For example, Steghide's LSB algorithm will replace byte sections owned by the EXIF metadata present in an image. DeepSound will decompress the cover carrier and re-encode it with the steganographic content allowing it to increase its spatial domain without increasing the file size of the steganographic carrier. And, OpenPuff's LSB algorithm will hide the data in redundant MP4 fields using a least significant bit matching approach.

6.2 How does the steganographic process affect the morphing of media and how can this information be harvested to progressively raise a culture of security awareness and improve cyber defensive postures?

The least significant bit steganography process will morph and affect a media's spatial domain. Binary stream composing a file can be transformed using two methods under the least significant bit steganography algorithms. A carrier's file composition can be altered by the replacement of bits within the carrier file or by the matching of bits within the carrier. Both methods have the ultimate purpose of embedding a secret, but there is no way to know

whether the hidden secret is benign or malicious, thus posing a risk to unaware victims of malicious steganography. Steganography does not get the attention it deserves in the cyber defenses space and even less from the public. Information on how steganography works and why it is important can be shared by creating an awareness campaign for small scale or large scale audiences with the goal of progressively raising a culture of security awareness and improving cyber defensive postures starting from the weakest link, the people.

6.3 Solutions

There is no all in one solution to identify the presence of steganography in carrier files. However, a robust solution could be designed to implement hash and software signature detection in combination with statistical steganalysis algorithms that support file types of interest. A hybrid steganography detection system has a lot of potential as it could address digital signatures at a top level and if the signatures don't match with the signature database then the carrier could be addresses by a bottom level where the carrier is examined further by statistical steganalysis, if the carrier is still not classified as clean then it could be flagged for manual analysis. A top level script implementation that could benefit such system could be thought as follows:

```
#!/bin/bash
```

```
for filename in directory
```

```
do
```

```
hexdump -p filename > directory
```

```
tr -d '\n'
```

```
if grep -c -q -"Digital Signature" filename; then
```

```
echo -e "Digital Signature Found" filename >> Directory
```

```
Else
```

```
echo -e "Digital Signature Not Found" filename >> Directory
```

```
Fi
```

```
done
```

Step 1: Create Hexdump of file

Step 2: Grep Digital Signature

Step 3: Alert if signature is found or not found

7 Recommendations and Guidance for Future Research

There is a wide range of ideas that could benefit the forensic community when it comes to research in steganography. Further experiments could be carried out on the tools presented in this project to potentially find better forensic artifacts produced by the least significant bit algorithms or propose a better solution to this project as it is not perfect.

However, we propose the following:

1. Additional research on how to identify steganography in audio and video carriers affected by least significant bit embedding without the use of statistical steganalysis. This could potentially bridge more artifact similarities found across the carriers.
2. There are tools available that can automatically detect the presence of steganography in images using brute force techniques to extract the carrier's contents. It would be really interesting to see a tool that could achieve the same capabilities against audio and video carriers.
3. The development of a hybrid steganography detection system that uses both signature steganalysis and statistical steganalysis has a lot of potential and would draw the interest of many forensic examiners of all levels.

As a short term recommendation, we recommend that users and organizations refrain from opening or accepting carriers from untrusted sources. Stegomalware is increasing in popularity as so do their number of attacks. It could also be a source of illicit content. As simple and innocent as carriers may be to the human sensory system, always be aware of malicious intents.

Long term business related recommendations are to invest in an intrusion detection system that possesses a signature database of digital steganographic signatures as a minimum.

As a short and long term recommendation, we recommend that forensic examiners stay up to date with the latest steganographic tools and techniques. There is a lot of material regarding image steganography but not as much for audio and video steganography. It should remain a priority for digital forensics firms to train their staff to overcome such challenges.

8. Conclusion

The project and study presented answered the proposed research questions. The objectives established at the beginning of the journey have been satisfied by the report. Modern steganographic techniques remain a big challenge to overcome for forensic examiners and those doing research in the field. Our experimental investigation identified and analyzed artifacts created by the least significant bit algorithms on image, audio and video files. There is no way to know whether the hidden secrets within a carrier are benign or malicious, thus posing a risk to unaware victims of malicious steganography. Information on how steganography works and why it is important can be shared by creating an awareness campaign for small scale or large scale audiences with the goal of progressively raising a culture of security awareness and improving cyber defensive postures starting from the weakest link, the people.

9. Project Management

The entirety of this project was managed using the Kanban project management methodology. This approach's philosophy is to balance the workload demands with available capacity, preventing workload bottlenecks. The Kanban framework and core practices include visualizing the workflow of the project and re-prioritize tasks if they require more time, giving us flexibility for our project. We focused on one task at a time, it allowed us to isolate individual aspects of the project in the grand scheme of things and to work towards the completion of this report.

	Project Summary		100%	06/01/21	08/13/21
●	Phase 1		100%	06/01/21	08/13/21
●	Ethics Certificate	Complete	100%	06/01/21	06/16/21
●	Project Proposal	Complete	100%	06/17/21	06/18/21
●	Introduction	Complete	100%	06/21/21	06/25/21
●	Literature Review	Complete	100%	06/28/21	07/14/21
●	Research	Complete	100%	07/14/21	07/16/21
●	Requirements	Complete	100%	07/17/21	07/20/21
●	Experimental Investigation	Complete	100%	07/21/21	07/30/21
●	Discussion of Results & Conclusions	Complete	100%	07/31/21	08/02/21
●	Recommendations & Future Work	Complete	100%	08/02/21	08/04/21
●	Conclusion	Complete	100%	08/04/21	08/06/21
●	Project Management	Complete	100%	08/07/21	08/09/21
●	Improvements on the report	Complete	100%	08/09/21	08/13/21

Figure 39 - Project Summary

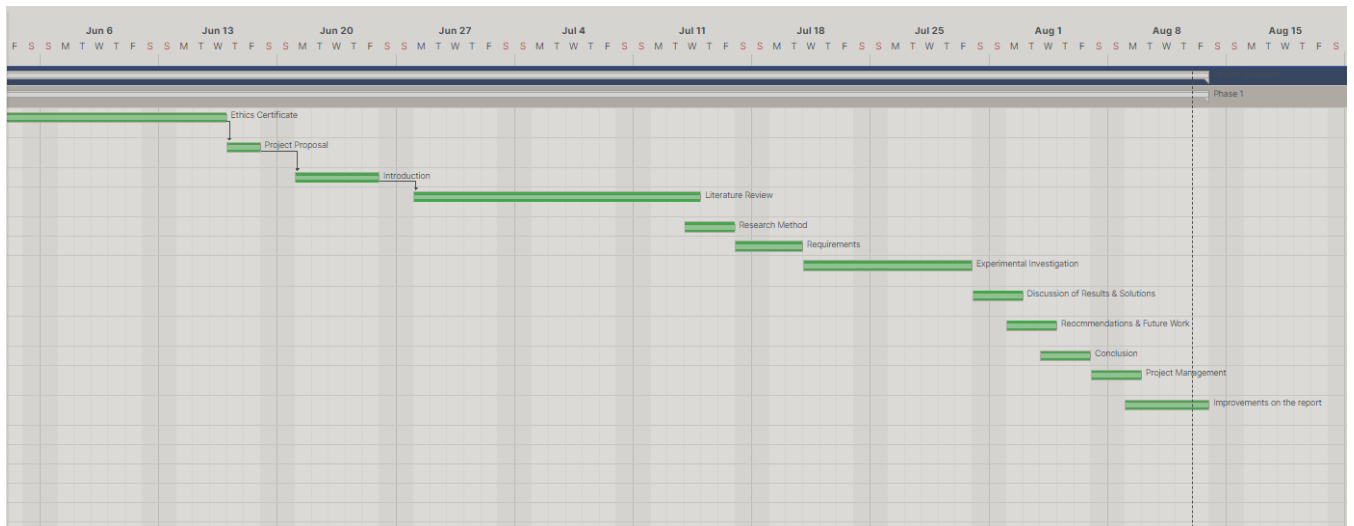


Figure 40 - Gantt Chart of The Project Management Based on Project's Summary

9.1 Personal Reflection

The original scope of the project intended to cover more steganographic algorithms but this was cut down to three LSB embedding tools due to time limitations. The project does not present the best possible approaches nor the best possible solutions despite its efforts to do so. However, a lot was learnt and exploring the rabbit hole that is steganography proved to be a challenge. It was personally fascinating to find it was possible to reconstruct sections of hidden data during the image carrier experiment.

10. Bibliography

- [1] Hamid, N., Yahya, A., Ahmad, R. B., & Al-Qershi, O. (2012). Image Steganography Techniques: An overview. *International Journal of Computer Science and Security (IJCSS)*, Volume (6): Issue 3. Available at: <https://www.cscjournals.org/manuscript/Journals/IJCSS/Volume6/Issue3/IJCSS-670.pdf>
- [2] Kaur, N., & Behal, S. (2014). Audio Steganography Techniques - A survey. Department of Computer Science, SBSSTC. Available at: <https://core.ac.uk/download/pdf/25870269.pdf>
- [3] McAfee. (2017) McAfee Labs Threats Report. Available at: <https://www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-jun-2017.pdf>
- [4] Křoustek, J. (2015). Analysis of Banking Trojan Vawtrak. AVG Technologies, Virus Lab. Available at: https://cdn2.hubspot.net/hubfs/4650993/Blog_Content/Avs/Signal/avg_technologies_vawtrak_banking_trojan_report.pdf
- [5] Shantala, C. P., & Viswanatha, K. V. (2015). Secure SDN Framework for Data Exfiltration via Video Steganography. *INTERNATIONAL JOURNAL OF COMPUTERS & TECHNOLOGY*, 14, 6067–6073. <https://doi.org/10.24297/ijct.v14i9.1847>
- [6] Sloan, T., Hernandez-Castro, J. 2015. Forensic analysis of video steganography tools. *PeerJ Computer Science* 1:e7 <https://doi.org/10.7717/peerj-cs.7>
- [7] Singh, A. K., Singh, J., & Singh, H. V. (2015). Steganography in Images Using LSB Technique. *International Journal of Latest Trends in Engineering and Technology (IJLTET)*, Volume (5): Issue 1. ISSN: 2278-621. Available at: <https://www.ijltet.org/wp-content/uploads/2015/02/60.pdf>
- [8] Z. Y. Al-Omari., A. T. Al-Taani. (2017). "Secure LSB steganography for colored images using character-color mapping," *2017 8th International Conference on Information and Communication Systems (ICICS)*, pp. 104-110, doi: 10.1109/IACS.2017.7921954.
- [9] Fridrich, J., Miroslav, G., Rui, D. (2001). "Reliable detection of LSB steganography in color and grayscale images", *Proceedings of the 2001 workshop on Multimedia and security: new challenges*, pp. 27-30.
- [10] B. Vishnu, L. V. Namboothiri, S. R. Sajeesh and L. V. Namboothiri. (2020). "Enhanced Image Steganography with PVD and Edge Detection," *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, pp. 827-832, doi: 10.1109/ICCMC48092.2020.ICCMC-000153.

- [11] D-C, Wu. W-H, Tsai. (2003). *A steganographic method for images by pixel value differencing*, vol. 24, pp. 1613-1626.
- [12] J, Canny. (1986). "A Computational Approach to Edge Detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679-698.
- Jaafar, S., Manaf, A. A., Zeki, A. M.: Steganography technique using modulus arithmetic. In: *In 2007 9th International Symposium on Signal Processing and Its Applications*, pp. 1–4. IEEE (2007).
- [13] W, Kuo. Y, Chen. C, Chuang. (2014). "High-capacity Steganographic Method based on Division Arithmetic and Generalized Exploiting Modification Direction", *Journal of Information hiding and Multimedia Signal Processing.*(ISSN 2073- 4212),Ubiquitous International Volume 5, Number 2.
- [14] Palmer, G. (2001). A road map for digital forensic research.
- [15] Kessler, G. C. (2004). An Overview of Steganography for the Computer Forensics Examiner. Available at: https://www.garykessler.net/library/fsc_stego.html
- [16] Robertson, N., Cruickshank, P., Lister, T. (2012). Documents reveal al Qaeda's plans for seizing cruise ships, carnage in Europe. CNN. Available at: <https://edition.cnn.com/2012/04/30/world/al-qaeda-documents-future/index.html>
- [17] CPS.gov.uk. (2019). *Expert Evidence | The Crown Prosecution Service*. [online] Cps.gov.uk. Available at: <https://www.cps.gov.uk/legal-guidance/expert-evidence>.
- [18] Karampidis, K., Kavallieratou, E., & Papadourakis, G. (2018). A review of image steganalysis techniques for digital forensics. *Journal of Information Security and Applications*, 40, 217–235. <https://doi.org/https://doi.org/10.1016/j.jisa.2018.04.005>
- [19] Sharp, T. (2001, April). An implementation of key-based digital signal steganography. In *International Workshop on Information Hiding* (pp. 13-26). Springer, Berlin, Heidelberg.
- [20] Chen, X., Gao, G., Liu, D., & Xia, Z. (2016). Steganalysis of LSB matching using characteristic function moment of pixel differences. *China Communications*, 13(7), 66-73.
- [21] Juarez-Sandoval, O., Cedillo-Hernandez, M., Sanchez-Perez, G., Toscano-Medina, K., Perez-Meana, H., & Nakano-Miyatake, M. (2017, April). Compact image steganalysis for LSB-matching steganography. In *2017 5th International Workshop on Biometrics and Forensics (IWBF)* (pp. 1-6). IEEE.
- [22] Steghide. (2021). Available at: <http://steghide.sourceforge.net/>.

[23] DeepSound. (2021). Available at: <https://deepsound.en.uptodown.com/>.

[24] OpenPuff. (2021). Available at: <https://embeddedsw.net/>.

[25] Autopsy. (2021). Available at: <https://www.sleuthkit.org/autopsy/>

[26] Shakespeare. (2021). All's Well That Ends Well. Available at:
<http://shakespeare.mit.edu/allswell/full.html>