NOTTINGHAM TRENT UNIVERSITY

SCHOOL OF SCIENCE AND TECHNOLOGY

Concealing Data in Pixels Using Steganography

by

Manjot Singh Dhillon

in

2024

Project report in part fulfilment

of the requirements for the degree of

Bachelor of Science with Honours

In

Computer Systems (Cyber Security)

I hereby declare that I am the sole author of this report. I authorise Nottingham Trent University to lend this report to other institutions or individuals for the purpose of scholarly research.

I also authorise Nottingham Trent University to reproduce this report by photocopying or by other means, in total or part, at the request of other institutions or individuals for the purpose of scholarly research.

Signature

Manjot Singh Dhillon

ABSTRACT

The following project aims to address the issue of data security breaches within companies by proposing a steganography tool as a solution. Through extensive research, it was identified that many companies suffer from data losses, suggesting the need for robust security measures. Additionally, the tool's application extends to aiding fair journalism and reporting from regions with information restrictions, enhancing accessibility to vital information.

The research method comprehensively investigates various steganographic algorithms and encoding formats. Specifically, the study focussed on the working of algorithms such as Least Significant Bit (LSB) and Discrete Wavelet Transform (DWT). Subsequently, an application has been developed to enable the encoding of textual data within images and text files, enhanced by encryption to fortify security measures.

The employed methodology encompassed rigorous exploration of different algorithms, evaluating their efficacy and applicability to the project's objectives. Through careful development and testing, a user-friendly Graphical User Interface (GUI) was designed to facilitate usage.

Key findings indicate the successful implementation of the steganography tool, demonstrating its effectiveness in concealing textual data within various media formats. Incorporating encryption algorithms enhances the security and robustness of the tool, mitigating the risk of data breaches. In addition, the algorithms underwent assessment using performance evaluation metrics such as Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM).

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to Ms. Ismahane Cheheb, my project supervisor, for their exceptional guidance, support, and invaluable insights throughout this project. Their mentorship has been invaluable in shaping the trajectory of this work.

I would like to thank my family and friends for their support, encouragement, and understanding throughout this journey. Their belief in me has been a constant source of motivation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF DIAGRAMS

CHAPTER 1

INTRODUCTION

1.1 Introduction

In today's age, there has been a digitalisation of assets, therefore, everything is digitally stored and processed. All kinds of data are collected and handled by companies, including individuals' data. As technological advancements are made in securing these assets with the implementation of cryptographic practises and the development of security-related hardware systems (IDS/IPS, firewall), the progress made by bad actors to breach the security placed by companies is also substantial. According to Sobers' (2023) research, over 51.4 million healthcare data records were breached within the US alone between 2017 and 2022. It is reasonable to assume that this number is several times larger globally.

The Information Commissioner's Office (2023) recommends that companies use electronic security measures such as solid user passwords, firewalls and anti-virus software. Although these security measures are effective, they cannot be relied on heedlessly; hence, companies often add layers of security, such as encryption and hashing algorithms, that are meant to safeguard the integrity of the data if it falls into the hands of an unauthorised individual. However, it is essential to note that even though these methods are highly secure, the data is not unbreakable, as techniques exist to break through encryption and hashing, which can be used to access leaked data. Steganographic algorithms can provide an additional layer of security to make data more difficult to access.

1.2 Steganography

Steganography is a technique that focuses on the concealment of information within other innocuous-looking data. Nowadays, the data used to contain secret information varies from a range of multimedia documents, varying from audio to video. Although seemingly modern, its creation dates to 440 BC, during ancient Greece, with several iconic applications, such as text hidden under hair (Burnley, 2018). Today, this technology has seen its digitalisation, involving the concealment of information within a digital object to reduce the chances of detection by an unauthorised third party (Evsutin, Melman and Meshcheryakov, 2020).

The terminology for this technology derives from the Greek words: "Stegos" and "Graphia", which mean "Covered Writing" (Majeed et al., 2021). Steganography is a technique that has been used for centuries. Its first recorded use was by Demeratus, the King of Sparta. He wrote a message on a wooden block and covered it in wax to warn his people. However, a more famous use of steganography belongs to the Greek nobleman Histaeus. He shaved the head of one of the enslaved people he trusted, tattooed a message on his head, and sent the enslaved person to the recipient once the hair had grown back (Şahin, Çevik and Takaoğlu, 2021). Steganography has been used in various historical events, including World War 1, World War 2, and the Vietnam War. In World War 2, the British and Germans used steganography, albeit in different forms. The British used smuggled board games to show escape routes and aid prisoners in escaping, while Nazi Germany developed the Microdot technique, which involved inputting pixel-sized pictures within a regular post (Burnley, 2018). Morse code was also used during the Vietnam War to send a message back to the US from a captive.

1.3 Project

Steganography has been digitised to conceal electronic data within multimedia files. This technology can be proven effective in helping secure data, but not many applications are available compared to encryption and anti-virus software. Therefore, the project described in this report follows the progress of software development from start to finish, which allows data to be concealed to enhance information security further at rest or in transit. The primary objective is to develop user-friendly software with a graphical user interface (GUI) that enables the secure storage or transmission of data. This approach will ensure accessibility for non-technical users, allowing them to use the software and benefit from its advantages.

Throughout the project, careful consideration has been given to legal, social, ethical, and professional issues.

CHAPTER 2

CONTEXT

2.1 Introduction

Steganography is an ancient art of concealing data within seemingly innocuous-looking carriers. Today, this has evolved in the digital age, finding applications across various domains, from safeguarding confidential information to protecting data (Evsutin, Melman and Meshcheryakov, 2020). The following section examines the various applications and techniques of steganography, exploring its utilisation in image, video, audio, text and network domains.

Steganography applications vary, from confidential communications and digital watermarking to potential malicious uses. Businesses and militaries utilise steganography for secure communication, embedding secret data within multimedia files transmitted across networks. Digital watermarking is another crucial application that ensures the authenticity and ownership of digital content by embedding identifying information within multimedia files. However, the technique can be exploited by malicious actors to conceal malware within seemingly harmless carriers, posing significant cybersecurity risks.

Steganography employs various techniques for different carrier types. Image steganography conceals data within digital images; video steganography extends this concept to digital videos. Text steganography conceals data within text files by altering their structure and semantics, while audio steganography embeds information within audio signals. Network steganography conceals data within network protocols.

This section explores techniques employed within each domain, such as transform domain methods in video steganography, temporal and transform domain techniques in audio steganography, and format-based and linguistic methods in text steganography. Finally, this section compares steganography with cryptography and watermarking, highlighting their distinct approaches to securing information.

2.2 Applications of Steganography

Steganography serves various purposes, with applications ranging from constructive to deconstructive uses. Typical applications include confidential communication, digital watermarking, data protection, and potential malicious uses. In confidential communication, secret data is hidden within multimedia files and sent across networks for extraction by the recipient, typically utilised by businesses and militaries (Simplilearn, 2023). Digital watermarking is another crucial application where data is embedded into a digital signal, making it challenging to erase; this is commonly used to prove ownership or for descriptive purposes. If the signal is copied, the embedded data persists within the copy (Ginni, 2022). Steganography also plays a role in protecting data from unauthorised alterations and access. Information is embedded within multimedia files, enhancing its efficacy as a tool for safeguarding user data (Sahu and Sahu, 2020). Businesses implement steganography to secure data for access control. Access keys are provided only to authorised personnel to extract data from the carriers (Kawaguchi, 2023). However, despite its legitimate uses, steganography can also be exploited by malicious attackers to conceal malware within seemingly innocuous carriers and carry out various malicious activities (Chowdary, 2023).

2.3 Types of Steganography

Steganography, the field dedicated to concealing information, uses various methods to conceal information within carriers, like images, videos, audio, text, and network protocols (Tidmarsh, 2023). Image steganography involves subtle modifications in a digital image's colour palette or the hex value of individual pixels, making detection almost impossible for the naked eye (Simplilearn, 2023). On the other hand, video steganography embeds data within digital videos, leveraging the extensive data videos contain, to hide information effectively; this can be thought of as a combination of image and audio steganography (Choudary, 2019). Text steganography hides data within text files by altering the text's physical structure or semantics (Gardiner, 2012). Audio steganography uses audio files as carriers, embedding information by changing their binary sequences (Simplilearn, 2023). Lastly, network steganography conceals data using network protocols, facilitating secret communications over a network without drawing attention (Tidmarsh, 2023).

2.3.1 Image Steganography

Image steganography is a technique for concealing data within a cover image such that the embedded content, whether text, images, audio, or binary information, remains invisible to the naked eye (Iwugo, 2023). This method is an effective tool for concealed communication, allowing the discreet transmission of confidential information without drawing the attention of unauthorised individuals.

Beyond its primary role in covering communication, Bachchas (2023) states that image steganography also finds application in digital watermarking. This application involves embedding specific copyright details or authentication codes into an image. The embedded data helps track and safeguard intellectual

property, ensuring that digital content's ownership and usage rights are protected and easily verifiable.

The methods used in image steganography are categorised into three types: Traditional-Based, CNN-Based and GAN-Based Steganography Methods (Subramanian et al., 2021).

2.3.1.1 Traditional-Based Methods

Traditional methods use classic data embedding techniques that do not involve machine learning or deep learning algorithms. Among these, the Least Significant Bit (LSB) method is widely used for its simplicity and effectiveness (Qin et al., 2019). The LSB technique operates on the principle that the human eye cannot detect minor changes in pixel values. Data is embedded by converting it into binary form and imperceptibly altering the least significant bits of the image's pixels (Bachchas, 2023). The selective alteration minimally impacts the image's appearance, rendering the embedded data undistinguishable to unauthorised individuals. The LSB technique has inspired other data embedding algorithms like HUGO, WOW, UNIWARD, and more (Qin et al., 2019).

Swain (2019) explains in his research that Pixel Value Differencing (PVD) is another notable method, which works by adjusting the differences between consecutive pixel values, ensuring that the changes blend seamlessly with the original picture's texture and patterns. This method combines LSB for the initial two bits and PVD for the remaining bits to balance data concealment and image integrity.

2.3.1.2 CNN-Based Methods

CNN-based methods use Convolutional Neural Networks for advanced and secure data embedding. CNNs are a Deep Learning system capable of processing input images, assigning weights and biases to various visually identifiable features, and identifying objects. When extracting features from an input image, this approach maintains the pixel relationships by analysing image properties using small squares of input data (Thenmozhi, 2022).

The application of CNN models in image steganography draws inspiration from the encoder-decoder architecture, in which both the cover image and confidential information serve as inputs for the encoder to produce a stego image, which then serves as input to the decoder to uncover the concealed data. While various algorithms have different architectures, they adhere to the same fundamental principle (Subramanian et al., 2021).

2.3.1.3 GAN-Based Methods

General Adversarial Network (GAN) based methods, introduced by Goodfellow et al. (2014), use a type of CNN for advanced data concealment. This method consists of two networks: a generator and a discriminator. The generator creates images closely resembling the input data, aiming to produce images that are as accurate as possible. Meanwhile, the discriminator's role is to differentiate between the generated and authentic images. This process involves training the generator to replicate the input data with minimal changes while the discriminator is trained to identify fake images accurately. In image steganography, a third network called a steganalyser is sometimes incorporated. This new network checks for the presence of hidden data in the images. These three networks compete to create images as close as possible to the original (Subramanian et al., 2021).

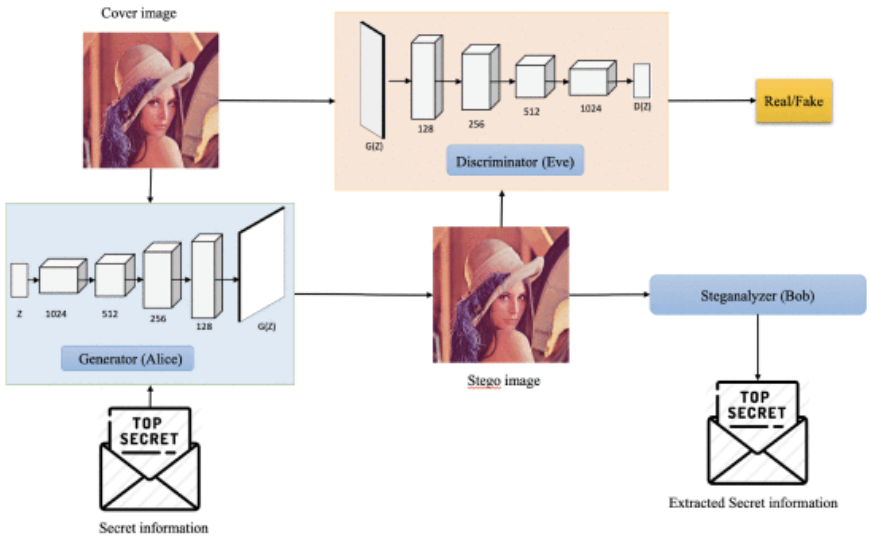


Figure 1: General Working Overview of GAN (Subramanian et al., 2021)

2.3.2 Video Steganography

Video steganography is a technique for hiding confidential information in videos. The concealed information can be any type of media or textual data, and the carrier video can be in any format, raw or compressed. Kunhoth et al. (2023) noted in their research that different methods for hiding data are grouped based on specific criteria. One way to classify these is by looking at the video type used: raw or compressed. The techniques are further classified into two domains for raw videos: spatial and transform domain. An example of a spatial technique is the Least Significant Bit (LSB) method, while transform techniques include methods like the Discrete Wavelet Transform (DWT) and Discrete Cosine Transform (DCT). The main difference between raw and compressed domains is that videos are compressed and made smaller in the compressed domain before the confidential data is added, saving storage space. Within the compressed domain, some commonly used techniques are Motion Vectors, intra-prediction modes, entropy coding modules, and DCT/DST (Fan, Zhang and Zhao, 2022).

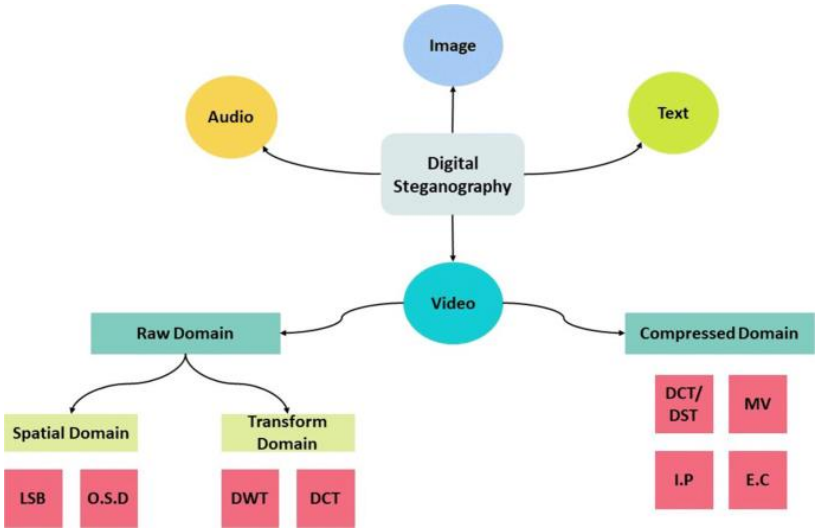


Figure 2: Hierarchical classification of video steganography (Kunhoth et al., 2023)

2.3.2.1 Raw Domain

The raw domain approach treats the cover video as a sequence of frames in the same format. In their paper, Mstafa and Elleithy (2017) explain that a digital video is initially converted into individual frames, which are then independently used as carriers for concealing data. After embedding, all frames are combined to create the stego video. In this domain, confidential information can be hidden by hiding it directly within the image or by converting the cover file into the frequency domain to conceal the data (Kunhoth et al., 2023). To guarantee the safety of this sensitive data, it undergoes pre-processing before being hidden. This pre-processing ensures that the data remains secure, even if the cover image is subjected to attacks or some frames are lost during transmission.

2.3.2.1.1 Spatial Domain

Numerous steganographic methods use the spatial domain, where pixel intensities are used to hide secret messages. In a study by Zhang et al. (2012), they proposed an approach using BCH coding to embed data by concealing it within a block of the carrier object. This technique conceals information within a block of the carrier object by altering specific coefficients in the input values. This technique improves the capacity for embedding data and reduces the time it takes to execute compared to other methods, effectively transforming the algorithm's complexity from exponential to linear.

Alternatively, the LSB algorithm stands out as a widely employed technique; in this method, the video is converted into frames, each being treated as an individual image. Since each image in colour consists of three channels (RGB – Red, Green and Blue), the secret bits are embedded in the last bit of each channel. To store one byte of data, with a minimum of three pixels is necessary (GR and RB, 2021).

2.3.2.1.2 Transform Domain

The transform domain involves converting cover frame blocks into the transform domain and embedding secret data in the least significant bits of transform coefficients. The commonly used functions in this domain are DWT and DCT (Korgaonkar and Gaonkar, 2017).

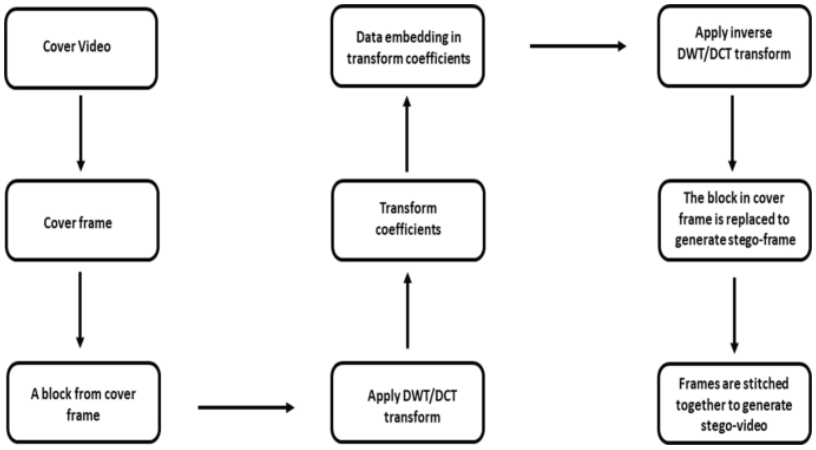


Figure 3: General workflow of data hiding in the transform domain (Kunhoth et al., 2023)

Kunhoth et al. (2023) explain that signals can be represented in the time or frequency domain, with each domain capturing essential characteristics. In stationary signals, frequency components remain constant, while non-stationary signals experience frequency changes. The DWT breaks down signals into sets containing significant and insignificant information. The significant information, referred to as low-frequency DWT coefficients, represents the overall characteristics of the signal. Conversely, the insignificant information reflects the signal behaviour, known as high-frequency coefficients. Achieving this decomposition involves passing a single signal through a series of filters and splitting it into two parts.

The Discrete Cosine Transform (DCT) is an alternative to the Discrete Wavelet Transform (DWT) for image processing. This algorithm creates more frequency

bands, providing better frequency resolution. In video processing, the DCT is separately applied to each frame, dividing it into low, middle and high-frequency bands. Data is concealed within the transform coefficients of one or more bands (Mstafa and Elleithy, 2017).

In a related study, Huang and Shi (2002) introduced an algorithm for hiding information using DCT and the communication theory. This approach incorporates BCH codes and soft decision decoding and is tested against joint signal processing operations and the Stir Mark attack. The confidential information is embedded within the DCT coefficient with the highest energy and low-efficiency AC coefficients.

2.3.2.2 Compressed Domain

The compressed domain involves concealing data within compressed video files; this approach offers advantages such as reduced memory consumption and increased data transfer speed across networks. This process can be achieved by using various techniques, including MPEG, AVC (H.264) and HEVC (H.266) video codecs (Patel, Lad and Patel, 2021).

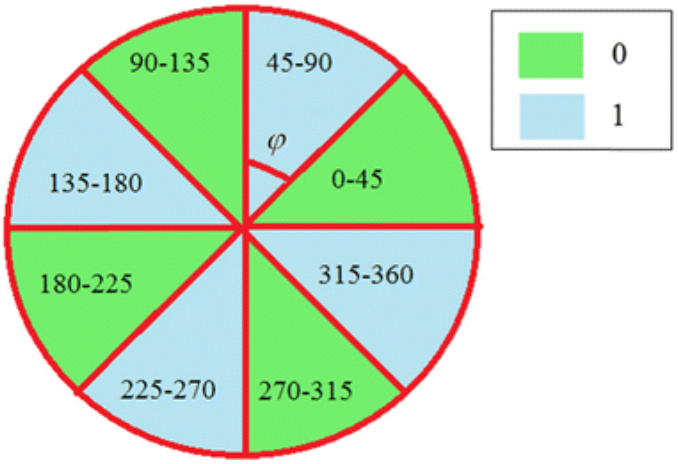


Figure 4: Motion vector representation (Pan et al., 2010)

In a study conducted by Pan et al. (2010), a steganography approach was introduced within the H.264 video standard, utilising motion vectors and linear block codes. This method involves embedding data using motion vectors of inter-frame macroblocks and discarding surrounding macroblocks. Motion vectors are chosen within each video inter-frame based on a predefined threshold, and their values are determined by calculating the phase angles obtained from the arctangents of vertical and horizontal motion vector components. The concealed information is integrated into the motion vector array using the linear block principle, minimising vector alterations and maximising embedding capacity (Mstafa and Elleithy, 2017). The algorithm demonstrates that within every 6 bits of the motion vectors array, 4 bits of data

can be concealed. However, the number of motion vectors available limits the method's capacity.

2.3.3 Audio Steganography

Audio steganography emerges as a sophisticated technique for concealing data within audio signals, an approach using the masking effect of the Human Auditory System (HAS), a concept detailed by Jian et al. (2017). A key focus in implementing this technique lies in its robustness, particularly its resilience to noise, compression and signal manipulation. According to Bender et al. (1996), achieving a high level of robustness presents a significant challenge, as it often necessitates a trade-off with the capacity for data hiding; enhancing robustness reduces the amount of data concealable.

The shift towards adapting audio files as a medium for steganography is primarily motivated by the extensive efforts directed at digital images in the field of steganalysis, leading to an initial neglect of audio steganography. However, Djebbar et al. (2012) pointed out that audio steganography presents challenges, for which the sensitivity of the HAS plays a critical role. It can tolerate minor changes within specific differential ranges and often mask softer sounds with louder ones, which typically goes unnoticed by individuals.

Various categories are present within this technique to facilitate the concealment of messages. These domains include the Temporal and Transform domains, with the latter including three subdomains.

2.3.3.1 Temporal Domain

The temporal domain is a collection of algorithms designed for modifying the waveform of an audio signal. The primary goal of these algorithms is embedding information while causing the slightest alterations possible to the sound's perceived quality. This approach is vital to ensuring the data-hiding process is secure and robust, as Kanhe et al. (2015) highlighted in their study. Additionally, as Djebbar et al. pointed out in 2012, a significant aspect of this domain is the reliance on a perceptual model for determining the maximum amount of data that can be embedded in the audio without noticeably affecting the quality, therefore, maintaining the audio's natural sound while covertly embedding data, thereby upholding a balance between data capacity and sound quality.

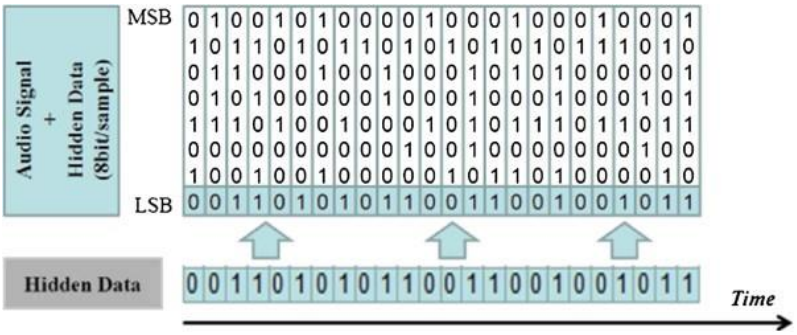


Figure 5: The least significant bit (LSB) within an 8-bit per sample signal gets replaced by a single bit of concealed data (Djebbar et al., 2012)

A practical example of how this domain operates can be seen in the utilisation of the Least Significant Bit (LSB) algorithm. This technique involves the modification of the least significant bit of each audio sample to embed the bit of the secret message (Zamani et al., 2009). Further advancing this technique, Rana and Kunwar (2016) developed a method that used the LSB algorithm while enhancing it. They replaced each audio sample's third and fourth bits with the

secret message bits and then carefully adjusted the original bits to ensure that the audio sample's statistical characteristics remained unchanged.

2.3.3.2 Transform Domain

The transform domain is a collection of techniques that use the human perceptual properties and frequency masking capabilities inherent in the Human Auditory System (HAS) (Bilal and Kumar, 2015). The process within this domain involves the conversion of the original cover object into a different subdomain, which produces a transformed coefficient that undergoes an inverse transformation process, resulting in signals that effectively carry the embedded data (Hemalatha, Acharya and Renuka, 2015).

Notably, this domain has three distinct subdomains: frequency, wavelet and codecs.

2.3.3.2.1 Frequency Domain

The frequency domain is a part of the transform domain, focussing on the techniques that manipulate the frequency elements of an audio signal to conceal confidential information (Bilal and Kumar, 2015). Research by Djebbar et al. (2012) highlights the superiority of the frequency domain over the temporal domain because it can produce better results in signal-to-noise ratio.

An example of a frequency domain algorithm is tone insertion, which uses the HAS' inability to hear quieter tones when louder ones are present. This process, known as auditory masking, was discussed by Gopalan and Wenndt (2004) in their research. To embed a single bit into an audio frame, a pair of tones are generated at specific frequencies, with their power levels carefully adjusted to a certain level relative to the overall power in the audio frame. Embedding is accomplished by inserting these low-power tones at specific frequencies. This method has a capacity of 250 bps when four tones are included in each spectrum (Djebbar et al., 2011).

2.3.3.2.2 Wavelet Domain

The wavelet domain uses mathematical tools to decompose a signal into its frequency components, allowing temporal and frequency characteristics to be analysed. A study by Ali (2015) on audio steganography utilising wavelet and Data Encryption Standard (DES) illustrates how data is hidden utilising this domain. This process involves transforming the audio file from the temporal domain to the frequency domain using a one-level linear wavelet decomposition technique. Only the high-frequency components retrieved from the audio file are utilised to conceal the confidential message, followed by the application of DES to encrypt the text, which will be concealed using the LSB algorithm.

2.3.3.2.3 Encoder Domain

The encoder domain focuses on embedding sensitive data within the audio signal. The techniques used within this domain are designed to ensure that hidden data remains unaffected and intact, even when noise is added or when audio is compressed using audio codecs like AMR (AlSabhany et al., 2020). The primary objective of this domain is to preserve the robustness and integrity of the concealed information even when the audio goes through compression and encoding.

Aoki (2008) presented a steganography method for the G.711 telephony speech coder that does not result in data loss. This technique employs a folded binary code to represent data, assigning each speech sample a value within the range of -127 to 127. One bit of information is embedded into every 8 bits of speech data with zero amplitude. The potential capacity for data hiding varies, offering a range from 24 to 200 bps, which depends on the number of speech samples with zero amplitude.

2.3.4 Text Steganography

Text steganography was historically a prevalent method for concealing information, but nowadays, it has witnessed a decline in its significance. This decline can be attributed to the relative ease with which such methods can be deciphered. Text files exhibit more redundant data than digital media, posing significant challenges in employing text steganography (Din, Utama, and Mustapha, 2018). A critical limitation of this approach lies in its vulnerability to detection. If a detector identifies alterations within the text or if it undergoes modification, the embedded information is likely to be irreversibly lost. Past research has highlighted the two primary issues identified as related to steganography. The first issue pertains to the effectiveness of developing robust methods for text-based steganography (Reddy, Nagaraju, and Subramanyam, 2014), and the second concerns the low level of security provided when attempting to conceal information within a text (Meng, Ye, and Hang, 2010).

There are two main types of text-based steganography, with further subcategories:

- Format-based method:
    - Word shift encoding
    - Line shift encoding
- Linguistic steganography:
    - Syntax
    - Sematic

2.3.4.1 Format-Based Steganography

Format-based steganography utilises the physical attributes of text characters to alter them and subtly conceal covert messages. This technique involves substituting specific elements of existing data, such as adjusting the font size or margin width slightly, to embed secret information. The critical aspect of this method is its finesse: the modifications made are so insignificant that they remain imperceptible to an unaided visual inspection (Şahin, Çevik, and Takaoğlu, 2021). Consequently, the steganographic content is embedded in a way that evades detection without altering the text's readability or layout. This approach to data hiding is beneficial in scenarios where hidden messages must remain undetectable to casual observers.

Majeed et al. (2021) highlighted the effectiveness and discretion of this method, demonstrating its viability in secure communication applications.

2.3.4.1.1 Word Shift Encoding

Krishan, Thandra, and Baba (2017) describe word-shifting as a novel method of concealing information within a text document. This technique involves minimal manipulation of the spaces between words, allowing for embedding hidden messages without altering the text. The text is divided into lines, and within each line, words are grouped into sets of three. The positioning of the middle word in each trio is crucial to this method. The binary bit '0" is encoded by shifting the middle word slightly to the left while shifting it to the right encodes a binary bit "1". This discreetly embeds information using binary coding.

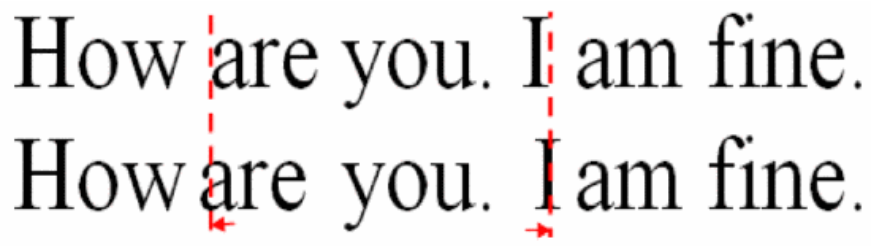


Figure 6: Word Shifting (Krishan, Thandra and Baba, 2017)

The document is formatted to justify the content and ensure that any alterations go unnoticed. Justification, a typical text alignment method that aligns text evenly along both the left and right margins, helps to mask any irregular spacing caused by word shifts. This technique minimises any visual anomalies that might arise from the word-shifting process, reducing the likelihood of detection. The document may appear ordinary at first glance, but it contains a hidden layer of information that can only be accessed by those aware of the encoding technique.

2.3.4.1.2 Line Shift Encoding

The line-shifting technique is a method for embedding binary data within textual documents. The technique utilises the vertical space between consecutive lines of text to encode information. Within this encoding technique, there are different algorithms one can use, one of which is line-shifted up and down (Din, Utama and Mustapha, 2018). In this, the lines of text are grouped in sets of three, and within each group, the position of the middle line is crucial: shifting the line slightly upwards encodes a binary bit "1" while shifting it downwards represents a binary bit "0" (Krishan, Thandra and Baba, 2017).

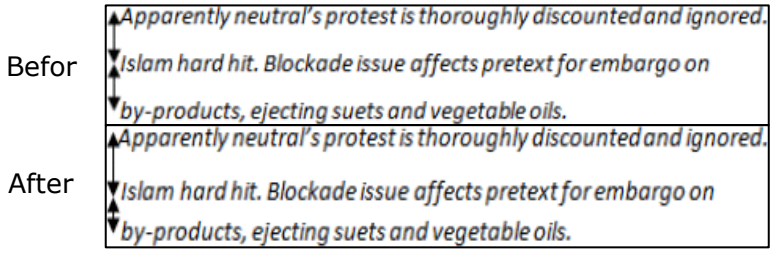


Figure 7: Line Shifting (Krishan, Thandra and Baba, 2017)

Kumar and Singh's (2020) study further explore this technique, emphasising the necessity of sophisticated differential coding methods. These methods enhance the system's robustness and ability to remain effective under various conditions and optimise its performance, ensuring the process is accurate and efficient.

One of this technique's most remarkable aspects is its data storage capacity. Kumar and Singh's study emphasises that on a standard page with 40 lines of text, it is possible to encode as much as 1,048,576 bits of binary data. To put this in perspective, this amount of data is equivalent to approximately 128 kilobytes, meaning it could include a short text, image or an audio segment.

The decoding process requires precision, as it involves meticulously measuring the vertical distances between the lines of text. This measurement is crucial for accurately interpreting the encoded data, as even a slight deviation could lead to errors in the retrieved information.

2.3.4.2 Linguistic Steganography

Linguistic steganography is the technique involving the embedding of hidden messages within a language's structure or meaning. This technique emphasises camouflaging the concealed information within the essential elements of text, such as words, phrases, and sentences (Kumar and Singh, 2020).

There are two primary strategies: syntax-based and semantic-based approaches. The syntax-based approach focuses on the arrangement and structure of words to hide messages, while the semantic approach uses the meaning and interpretation of words and phrases to encode information (Yadav and Batham, 2015).

2.3.4.2.1 Syntax

Syntax steganography conceals information by modifying the text's syntax or structure without notably changing its meaning. This approach operates on a low level, manipulating the grammatical structure of the text (Gardiner, 2012). There are various approaches to this practice, one common technique involves embedding punctuation into the cover text to encode data secretly (Yadav and Batham, 2015).

The study published by Zhang et al. (2018) described a method employing Malayalam, a language from southern India. This technique uses Malayalam's Unicode symbols to encode hidden messages by aligning them with the language's Unicode symbols.

2.3.4.2.2 Sematic

Semantic steganography is an approach that leverages the redundancies in the meanings of a word in a language to embed secret messages. This technique

exploits the various meanings of a word to conceal information, effectively camouflaging it within the regular use of a text (Vaishakh et al., 2019).

Figueira (2022) explains that a typical example of this method is replacing certain words in a text with their synonyms. While subtle, this alteration maintains the text's overall meaning, ensuring the steganographic message is not readily identifiable to unintended individuals. This method relies on the richness and complexity of a language to ensure that the altered text is unsuspicious, thus safeguarding the embedded information from detection.

2.3.5 Network Steganography

Network steganography hides information within network protocols like TCP, UDP or ICMP (Kaspersky, 2023). This method uses existing network traffic to conceal confidential data, minimising any impact on the standard data transmission. According to Mazurczyk et al. (2016), two main techniques are used in network steganography: storage and timing. Storage techniques modify protocol fields, such as unused bits in packet headers or data fields, to hide information, while timing techniques hide information within the timing of protocol messages or packets. Timing channels are less common than storage channels due to their complexity and limited user control over protocol and operating system timing events. These methods can also be combined for a hybrid approach.

However, current storage channels have two significant drawbacks: they transmit actual covert data within channels at a low transmission rate, and additionally, there is a risk to confidentiality if a third party discovers the channel since the data is embedded within the protocol (Soni, 2020).

2.4 Comparison with Cryptography

Steganography and cryptography are methods used to secure information, but they operate differently. Steganography focuses on concealing confidential information with seemingly innocuous carriers, making the data harsher to detect. On the other hand, cryptography converts information into unreadable ciphertext using mathematical algorithms to ensure the principles of cyber security are met (Saha, 2023).

While cryptography is widely known today, steganography is less prevalent. As previously discussed, steganography conceals data in various carriers like text, video, audio, image and network/protocol. In contrast, cryptography can be categorised into symmetric and asymmetric key cryptography (GeeksforGeeks, 2019).

A key distinction lies in the fact that steganography does not alter the structure of data, whereas cryptography algorithms alter the general structure (Malik, 2024). Cryptography relies heavily on mathematical algorithms for encryption, while steganography does not involve as many mathematical transformations (Saha, 2023). Furthermore, the security principles supported by these technologies differ. Steganography aims for confidentiality and authentication, while cryptography also addresses data integrity and non-repudiation (Panigrahi, 2022).

2.5 Comparison with Watermarking

Steganography and watermarking are techniques used to embed information within digital media, each serving distinct purposes and employing different methods. Digital watermarking involves embedding a visible or invisible identifier, such as a logo or text, into digital media files like images, videos, and audio (Mehta and Shetty, 2013). This process aims to identify the content

owner, protect the copyright, and facilitate tracking unauthorised distribution (Kaushik, 2012). Watermarks are designed to be robust, resisting modifications or removal attempts, and are commonly employed in fields like photography, art, and digital rights management (Mehta and Shetty, 2013). In contrast, steganography conceals the existence of a message by relying on secrecy, while watermarking embeds an identifier intended to be detected for traceability. Steganography focuses on covert communication, whereas watermarking serves copyright protection and content authentication purposes. Generally, steganography has a lower hiding capacity than watermarking techniques (Desai, 2012).

2.6 Review Existing Solutions

Several commercial and open-source applications facilitate the implementation of steganographic techniques for embedding data within multimedia files. Shankdhar (2020) lists these applications, presented in the following table, along with their strengths and weaknesses.

Table 1 - Existing Steganography Software

Software	Strengths	Weaknesses	Information
Xiao Steganography	• Simple and user-friendly GUI • Support for various file formats • Supporting	• Limited to image steganography • Using LSB algorithms making it detectable	Freeware was last updated in 2007 (Softonic, 2024)

	various encryption methods		
OpenPuff (Wikipedia, 2022)	• Support for various file formats • Supporting multi-layered encryption methods	• Complex GUI • Computationally heavy due to the multiple encryption layers	Closed-source software was last updated in 2018 (Wikipedia, 2022)
Steghide (Steghide, 2003)	• High efficiency due to it being run on a command line • Supporting strong encryption algorithms such as AES • Offering compression	• Lack of GUI • Limited file formats	Freeware was last updated in 2003 (Steghide, 2003)
Crypture	• Integration with Windows Shell • Easy to use	• Limited platform support (only Windows) • The algorithm used is not	Freeware was last updated in 2016

	GUI	known	
Steganograp hX Plus	• Offers password protection • Easy to use	• Low software maintenance	Freeware was last updated in 2010 (Softpedia, 2024a)
rSteg	• Resistance to image cropping and modifications	• Embedding is fragile; changes to the carrier might break the encoded data	Open-source software was last updated in 2017 (Robertson, 2021)
SSuite Picsel	• Hiding data within the pixel structure of the image	• Slow encoding and decoding speed • High increase in image file size	Closed Source Freeware was last updated in 2018 (SSuite Office, 2018)
Our Secret	• Easy to use • Hiding data	• Easily detectable	Closed Source

	within text files		Freeware was last updated in 2013
Camouflage	• Easy to use • Allows creation of decoy carriers	• Usage of known algorithms • A limited range of files supported	Open-source software was last updated in 2020 (Srivastav, 2024)
OpenStego (OpenStego, 2021)	• Open-Source • Watermarking capabilities	• Complex design • Low focus on encryption	Open-source software was last updated in 2023
SteganPEG	• Minimised image distortion	• Limited to JPEG files • Exposure to steganalysis tools	Closed Source software was last updated in 2011 (Softpedia, 2024b)

Hide'N'Seek	• Large hiding capacity • Ability to split hidden data across multiple carriers	• Low speed • Raised detectability	Open-source software was last updated in 2020 (Swamy, 2023)

After analysing existing solutions, it became clear that each software has advantages and disadvantages. One notable observation is that most available applications primarily focus on image encoding using the LSB algorithm. While this method effectively conceals secret messages within multimedia files, it represents a basic steganography approach, rendering it easily identifiable using steganalysis tools and techniques. Further examination reveals that some available applications have a limited range of files supported. Among the more robust software, such as OpenPuff and Steghide, there are instances where the graphical user interface is either complex or absent, making these applications less accessible for individuals with lower levels of technical expertise, indulging in a steep learning curve. Moreover, most applications reported in the table have not been updated in at least three years, making them outdated for today's threats and requirements.

CHAPTER 3

NEW IDEAS

3.1 Introduction

The following section explores new ideas for overcoming the limitations present in existing solutions. The proposed method aims to present a comprehensive solution for disguising data within multimedia files, integrating advanced algorithms while prioritising key factors such as security, robustness and user-friendliness. This approach aims to provide a seamless user experience while upholding high confidentiality and data integrity standards.

Subsequently, the section describes the initial concept of the proposed solution, highlighting its versatility in data hiding by supporting various carrier files, including images and text files. It outlines the scope and objectives of the proposed solution, provides insights into early design prototypes and the implementation strategy, and highlights the software development approach.

Additionally, the section discusses evaluation criteria and potential impacts, underlining the significance of the proposed method in enhancing information security.

3.2 Proposed Method Overview

The proposed steganography software is designed to tackle the shortcomings evident in existing solutions by presenting a comprehensive approach to concealing sensitive data within multimedia files. This approach integrates advanced and basic algorithms while emphasising key factors such as security, robustness, and user-friendliness. The proposed method offers a versatile and

efficient solution for data-hiding tasks. By prioritising these essential elements, the software aims to provide users with a seamless experience while maintaining the highest confidentiality and data integrity standards. Additionally, the proposed method aims to function efficiently on slower machines, ensuring users are not limited by owning older systems. This improves the accessibility of the software on a global scale, recognising that not all users can afford high-performance machines.

3.2.1 Objectives

The proposed steganography software is designed to ensure high-level of security when concealing sensitive data within multimedia files. This security is achieved by integrating multiple algorithms, such as Least Significant Bit (LSB) and Discrete Wavelet Transform (DWT). By leveraging the capabilities of these algorithms, the core aim is to strengthen the protection of embedded data, thereby safeguarding it against unauthorised access and detection.

Moreover, the software's objective is to ensure the durability and robustness of the data encoded. This is accomplished by employing advanced embedding algorithms, with a specific emphasis on using the DWT algorithm. The goal is to elevate the resilience of the encoded information against standard steganalysis methods, thereby maintaining the integrity of the concealed data.

Furthermore, the software is designed to offer versatility in data hiding by offering support for various carrier files, including images and text files. This flexibility enables users to conceal information within carriers, making the software adaptable for various applications. The comprehensive security measures and adaptability to offer different carrier types, position the software as a robust solution for concealing sensitive data across diverse contexts.

3.2.2 Scope

1. Text Steganography using LSB
    a. The proposed method introduces a robust approach for concealing textual information within text files using the LSB technique. This technique ensures a secure means of embedding data within the least significant bit of the zero-width characters present in the cover text file.
2. Image Steganography using LSB and DWT
    a. Two encoding algorithms are provided for embedding textual content within carrier images, and the user can choose between a basic level of encoding and an advanced level:
        i. Basic encoding: this method utilises the LSB algorithm to embed textual information directly into the least significant bit plane of the cover image. Despite its simplicity, this approach maintains a low computational overhead, making it ideal for scenarios where efficiency is vital.
        ii. Advanced encoding: this technique introduces the DWT algorithm, which offers significant advantages in terms of security and robustness. By leveraging this algorithm, the concealment of textual data into cover images is enhanced, ensuring increased imperceptibility of the hidden data while augmenting resistance against detection by unauthorised users.
3. User-Friendly Graphical User Interface (GUI)
    a. The steganography software aims to provide a user-friendly GUI designed to simplify embedding and extracting text from text files and images. The design aspires to cater to users with diverse

levels of technical knowledge, ensuring accessibility for both non-technical and technical users. The interface aims to enhance usability and mitigate the learning curve for users.

4. Compatibility and Ease of Use
    a. Beyond innovative techniques, the proposed method prioritises smooth user interaction by ensuring compatibility with widely used file formats. The commitment to compatibility enhances the accessibility and usability of the software.
5. Security Measures
    a. In addition to its user-friendly interface and compatibility features, the proposed method aims to integrate robust security measures to safeguard embedded data. Through encryption of the hidden information, the method ensures an additional layer of protection and contributes to the overall confidentiality and integrity of the information.

3.2.3 Potential Target Audience

The proposed method can benefit a diverse range of users. Firstly, in countries where freedom of the press is limited, journalists frequently encounter surveillance, censorship, and harassment. The proposed software offers a vital solution for securely communicating and sharing sensitive information, such as reports on government corruption or human rights abuses, without the risk of interception or detection by authorities. By embedding messages within seemingly harmless carriers, journalists can maintain the confidentiality of their information and protect themselves from retaliation. Moreover, activists seeking to evade government surveillance and censorship can employ this software to organise protests and coordinate campaigns, requiring a secure means of communication to safeguard themselves from persecution. Whistle-blowers often face substantial risks to their safety and sometimes livelihood, depending on the

organisation. This approach can empower them with a secure channel to transmit sensitive information without compromising its integrity, evading detection, and safeguarding themselves from retaliation. Lastly, many individuals seek ways to protect themselves amidst growing concerns about privacy, particularly regarding online surveillance and data breaches. This tool can conceal sensitive messages, enhancing online communications' security.

3.3 User Interface and Experience Design

The software is designed with a focus on simplicity to improve user experience. The flowcharts below provide an overview of the encoding and decoding processes, ensuring clarity on the steps for concealing messages within various types of carriers. Additionally, early design prototypes demonstrate a clean and intuitive GUI with buttons for file selection and encoding/decoding options. These design elements aim to improve user interaction, enabling seamless execution of tasks and easy navigation.

3.3.1 Flowchart

The flowchart below provides an essential representation of how the software operates on a user level without going into details of the encoding handling.

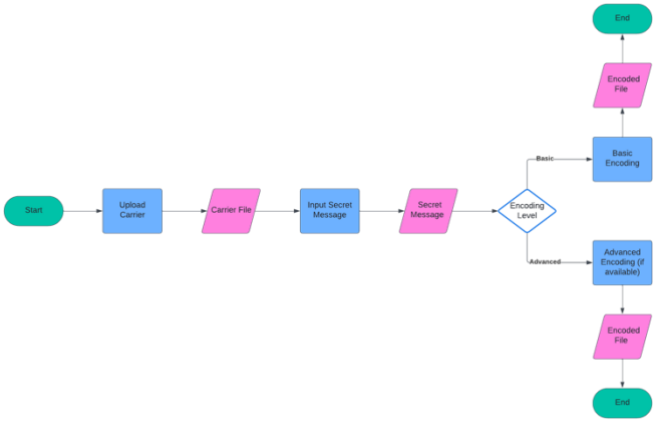


Diagram 1 - General Flowchart

Diagram 1 illustrates the inputs and outputs of the software. Users are prompted to upload the carrier file and the secret message to be embedded. Subsequently, they must select the desired encoding level, determining the algorithm employed.

3.3.1.1 Image Steganography with LSB

The following flowchart outlines the encoding of a secret message within an image carrier using the Least Significant Bit (LSB) algorithm.

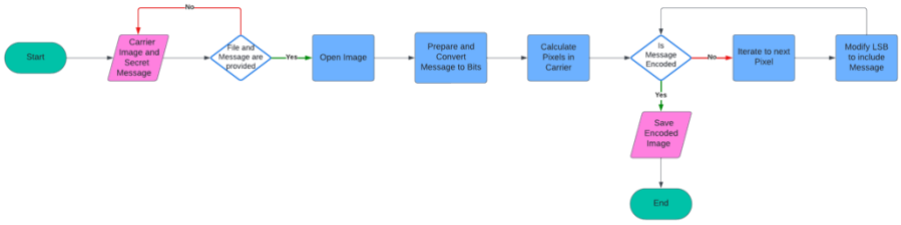


Diagram 2 – Image Least Significant Bit Encoding Flowchart

The carrier image and secret message are first uploaded, and a check is performed to ensure they have been received. Then, the image is opened, and the secret message is prepared and converted into bits. Next, the number of pixels within the carrier is calculated, and each pixel is iterated by modifying the least significant bit of each pixel until the secret message is fully encoded. Finally, the encoded image is presented as output to the user.

3.3.1.2 Image Steganography with DWT

The following flowchart outlines the process of encoding a secret message within an image carrier using the Discrete Wavelet Transform (DWT) algorithm.

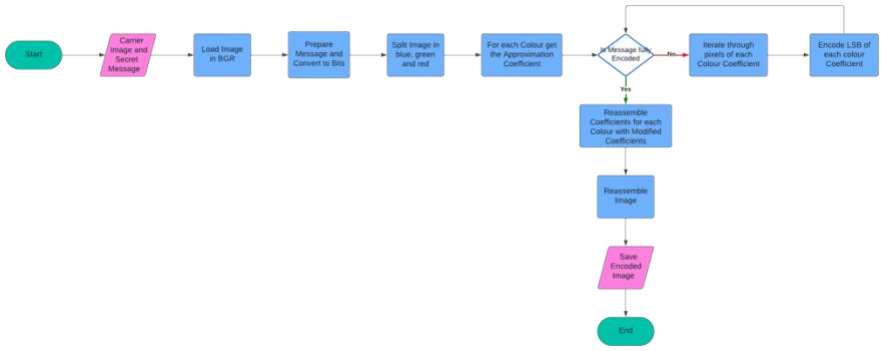


Diagram 3 – Image Discrete Wavelet Transform Encoding Flowchart

Firstly, the carrier image and secret message are uploaded by the user. Once received, the image is loaded in BGR format while concurrently preparing and converting the message into bits for later encoding. Subsequently, the colour channels are extracted from the carrier, and each channel's approximation coefficient is obtained. Following this, the pixels of the coefficients are calculated and iterated through. Each colour channel’s coefficient pixels are processed, with each pixel’s least significant bit encoding part of the secret message. Once fully encoded, each channel's coefficients are reassembled with the modifications. The reassembly of the image with the modified coefficients follows this. Finally, the encoded image is presented as an output to the user.

3.3.1.3 Text Steganography with LSB

The following flowchart outlines the encoding of a secret message within a text file using the LSB algorithm.

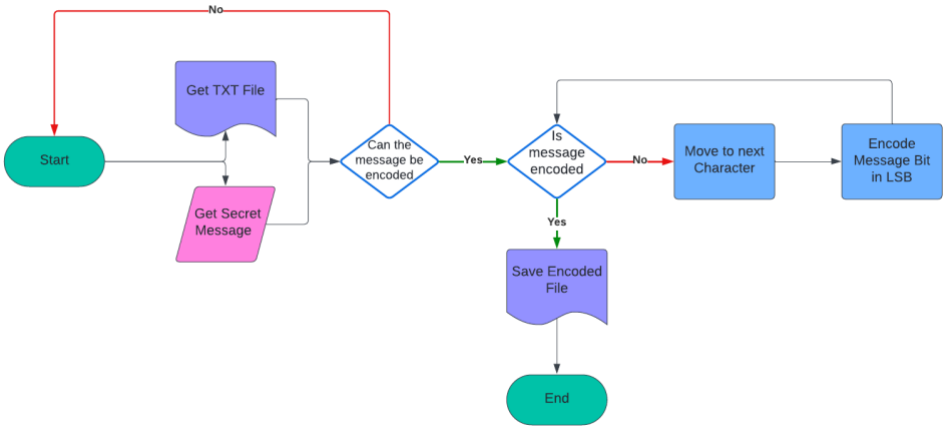


Diagram 4 - Text Least Significant Bit Encoding Flowchart

Initially, the software acquires both the carrier document and the secret message from the user. Based on its text volume, the message undergoes a check to determine if it can be encoded within the file. If encoding is possible, the program iterates through each character within the file, concealing one bit of the secret message within the least significant bit of each character where possible. Upon successfully encoding the entire message, the software generates an encoded text file for saving.

3.3.2 Early Design

The following figure represents an early prototype of the GUI of the home screen, allowing users to navigate between the different encoding methods.

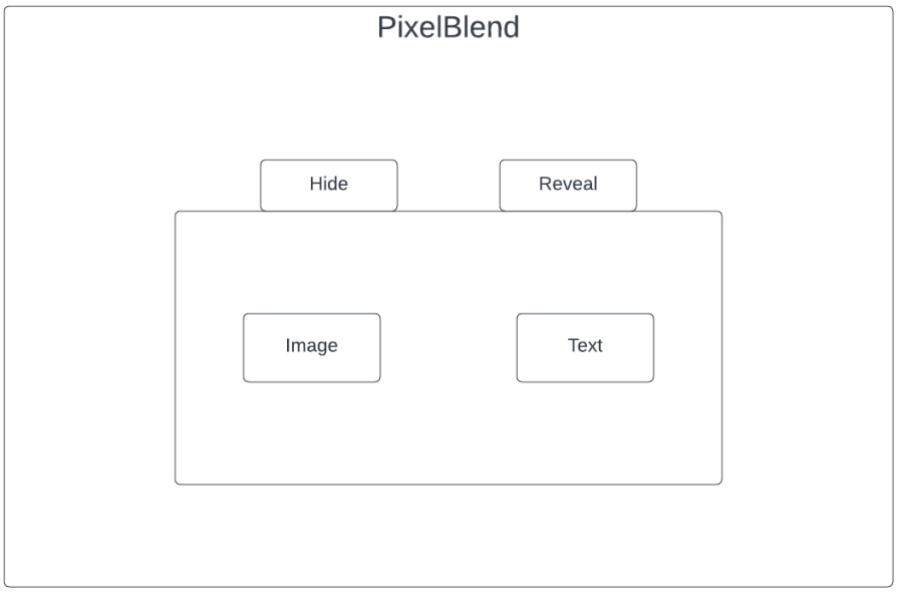


Figure 8 - Dashboard Prototype

Figure 8 displays the application's title at the top, along with two buttons. One button reveals the options for concealing text within multimedia files, while the other displays the options for extracting the encoded text from multimedia.

Once the user selects their preferred option, they can choose the type of multimedia file to encode or reveal data, initially choosing from image or text. Upon selection, a new window will appear with further instructions on encoding or revealing the data.

3.3.2.1 Encoding/Decoding GUI

The image below displays an early encoding and decoding GUI prototype to hide and reveal data.

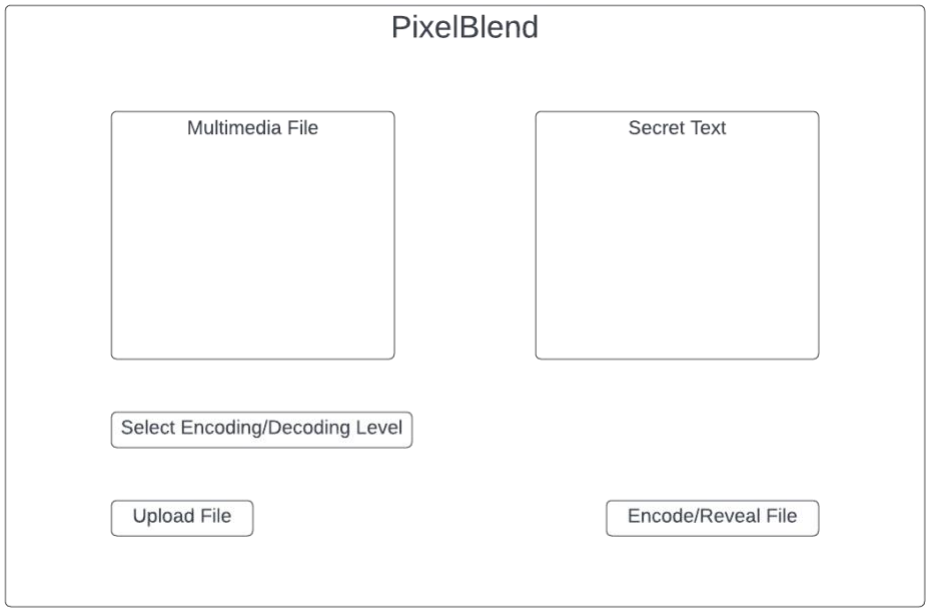


Figure 9 - Encoding/Decoding Prototype

Figure 9 displays the encoding and decoding window, featuring two primary sections. The first section displays a preview of the multimedia file, which could be an image or text extracted from a text file. The second section allows the user to input the secret data for encoding or to display the secret message once the extraction is completed. An option is provided to select the encoding/decoding level, allowing users to adjust the algorithm for varying security levels. At the bottom of the window, two buttons are available: one for uploading the multimedia file and the other for initiating the encoding or decoding process.

3.3.3 Use Case Diagram

The following diagram displays the use case diagram of the steganography software, displaying the functionalities of the proposed method. The diagram shows the two main activities of the software: encoding and decoding.

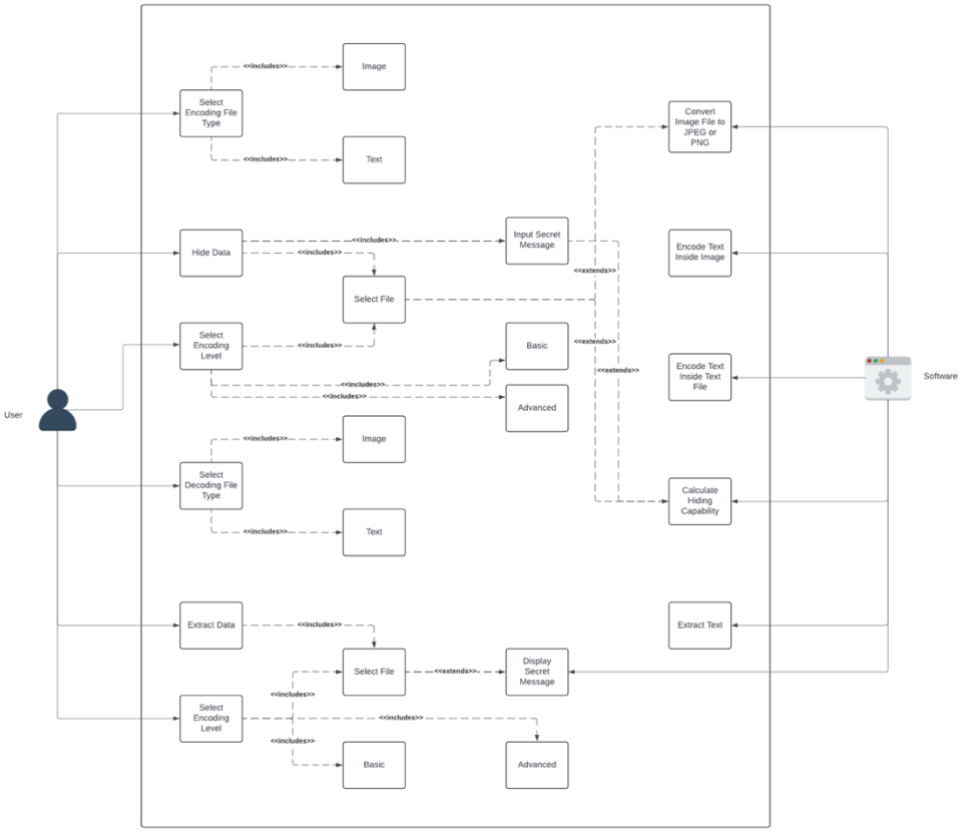


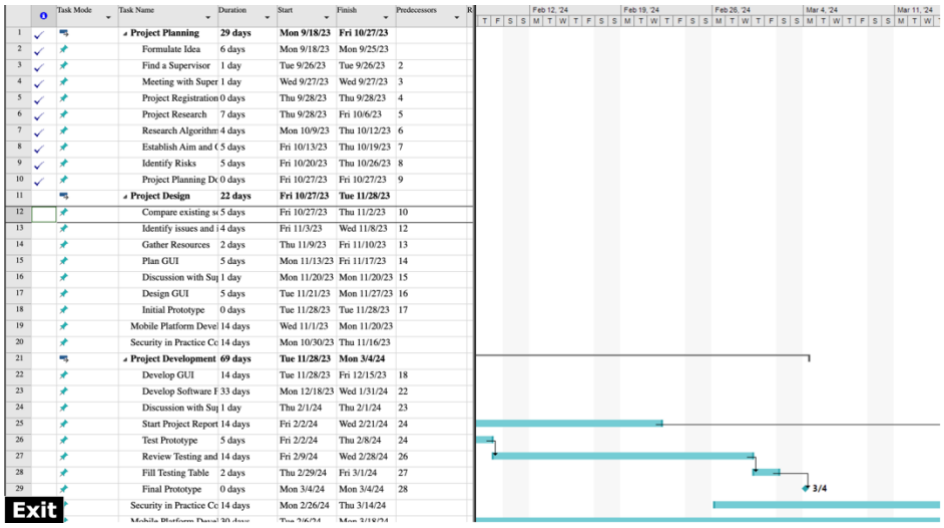
Diagram 5 - Use Case Diagram

Users can choose the type of file they want to encode, with options initially limited to images and text files. Next, depending on their previous selection, they input the message they wish to conceal within the uploaded cover file, which can be either a text or an image. Users also select the complexity level for hiding the message, with the option of basic or advanced algorithms. However, only basic encoding is available for text files. If the cover file is an image, the software may convert it to JPEG or PNG, depending on its original format. Once all selections are made, the software conceals the secret message within the cover file. Users choose the file type containing the encoded data for decoding

and upload it to the software. The complexity level to reveal the message must match the level selected during encoding. The software then extracts the hidden message from the encoded file and displays it to the user.

3.4 Gantt Chart

The image below shows the Gantt chart created during the project planning stage. It has served as a guide and will be used in the future to meet all the listed deadlines and ensure successful completion within the specified timeframe. The project is scheduled to conclude on April 19, 2024, and currently, the process is on track to achieve this goal.



	Task Mode	Task Name	Duration	Start	Finish	Predecessors
1		**Project Planning**	**29 days**	**Mon 9/18/23**	**Fri 10/27/23**	
2		Formulate Idea	6 days	Mon 9/18/23	Mon 9/25/23	
3		Find a Supervisor	1 day	Tue 9/26/23	Tue 9/26/23	2
4		Meeting with Super	1 day	Wed 9/27/23	Wed 9/27/23	3
5		Project Registration	0 days	Thu 9/28/23	Thu 9/28/23	4
6		Project Research	7 days	Thu 9/28/23	Fri 10/6/23	5
7		Research Algorithm	4 days	Mon 10/9/23	Thu 10/12/23	6
8		Establish Aim and C	5 days	Fri 10/13/23	Thu 10/19/23	7
9		Identify Risks	5 days	Fri 10/20/23	Thu 10/26/23	8
10		Project Planning Dc	0 days	Fri 10/27/23	Fri 10/27/23	9
11		**Project Design**	**22 days**	**Fri 10/27/23**	**Tue 11/28/23**	
12		Compare existing s	5 days	Fri 10/27/23	Thu 11/2/23	10
13		Identify issues and	4 days	Fri 11/3/23	Wed 11/8/23	12
14		Gather Resources	2 days	Thu 11/9/23	Fri 11/10/23	13
15		Plan GUI	5 days	Mon 11/13/23	Fri 11/17/23	14
16		Discussion with Su	1 day	Mon 11/20/23	Mon 11/20/23	15
17		Design GUI	5 days	Tue 11/21/23	Mon 11/27/23	16
18		Initial Prototype	0 days	Tue 11/28/23	Tue 11/28/23	17
19		Mobile Platform Devel	14 days	Wed 11/1/23	Mon 11/20/23	
20		Security in Practice Cc	14 days	Mon 10/30/23	Thu 11/16/23	
21		**Project Development**	**69 days**	**Tue 11/28/23**	**Mon 3/4/24**	
22		Develop GUI	14 days	Tue 11/28/23	Fri 12/15/23	18
23		Develop Software F	33 days	Mon 12/18/23	Wed 1/31/24	22
24		Discussion with Su	1 day	Thu 2/1/24	Thu 2/1/24	23
25		Start Project Report	14 days	Fri 2/2/24	Wed 2/21/24	24
26		Test Prototype	5 days	Fri 2/2/24	Thu 2/8/24	24
27		Review Testing and	14 days	Fri 2/9/24	Wed 2/28/24	26
28		Fill Testing Table	2 days	Thu 2/29/24	Fri 3/1/24	27
29		Final Prototype	0 days	Mon 3/4/24	Mon 3/4/24	28
		Security in Practice Cc	14 days	Mon 2/26/24	Thu 3/14/24	

Figure 10 - Gantt Chart

3.5 Steganography Algorithms Implementation

Several steganography algorithms are planned to focus on image and text concealment. There will be three algorithms: one for text, utilising LSB to hide zero-width characters, and two for image steganography. The first image algorithm will employ LSB to embed data within the least significant bit of each colour channel's pixel, while the second one will use DWT and LSB to conceal data within the lowest coefficient.

3.5.1 Image Steganography

Image steganography will be implemented, with basic and advanced algorithms allowing the encoding of text data within image carriers. The following sections explore the steganographic algorithms to be implemented.

3.5.1.1 Least Significant Bit (LSB)

The following algorithm will be implemented without relying on any Python libraries related to steganography, like Stegano or Stego. The decision not to use existing libraries comes from their lack of encryption features, leaving the stego files vulnerable to steganalysis techniques. The algorithm to be implemented incorporates an additional layer of security through encryption. Encryption is applied before commencing the encoding process, and upon completing the extraction process, a key is utilised to restore the original message. Figure 11 illustrates the process of encoding the secret message in the implemented algorithm. For every value of the RGB colour channels, the least significant bit of each channel is adjusted to accommodate a bit of the secret message. The LSB is altered to either 0 or 1, depending on the bit of the secret message being encoded.

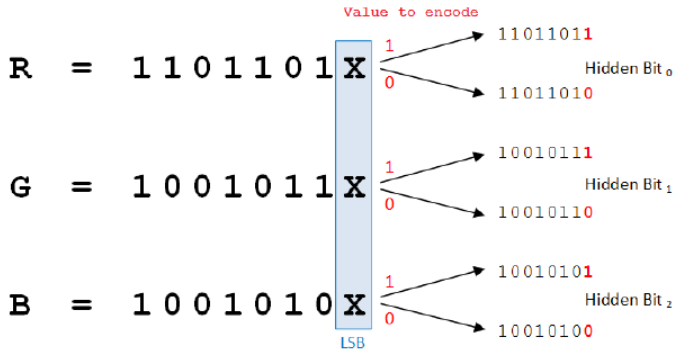


Figure 11 - LSB Encoding Visualisation (Balagyozyan and Hakobyan, 2021)

3.5.1.1.1 Hiding

The hiding process commences with selecting the image as the secret message's carrier. The image can be provided either as a file path or as an Image object,

although the application consistently only receives it as a file path. Before embedding the secret message, it is prepared by encryption using a unique key and converting its characters into their binary representation, with each character represented by a series of bits. Additionally, to ensure proper extraction, the length of the message is calculated and added at the beginning of the message, facilitating accurate extraction later. The carrier image is then processed to ensure it is in the RGB format, ensuring consistency during the embedding process. If the image is in a different format, such as grayscale, it is converted to RGB to ensure that three colour channels are available for manipulation. Subsequently, the encoding process begins by iterating over the pixels of the image. For each pixel, three bits from the message bitstream, representing the different colour components (RGB), are embedded into the least significant bit of the respective colour channels. As the LSBs contribute minimally to the overall colour value, modifying these bits results in minimal visual alterations. This process continues until all message bits have been encoded or there are no more available pixels to embed data into the image. Finally, the encoded image is returned to the user as the output, allowing it to be saved.

3.5.1.1.2 Extracting

The extraction process begins with selecting the carrier containing the secret text. Various variables are initialised to initiate extraction, including a buffer for accumulating characters, a counter to track processed bits, and lists to store binary characters and the decoded message. Additionally, a variable dynamically determines the length of the hidden message based on the value-added during encoding. The extraction process involves iterating through each image pixel, extracting the least significant bits of the red, green, and blue colour components for each pixel. These LSBs contain bits of the secret message,

enabling reconstruction of the hidden message upon analysis. The retrieved characters are stored in the buffer, and as the extraction process progresses, a check is performed for a limiter indicating the end of the message. Once the message length is determined, extraction continues until the entire message is retrieved. Lastly, the extracted message is converted back into textual characters from binary. It is then decrypted using the key provided to the user during encryption. This process reconstructs the original hidden message and presents it to the user.

3.5.1.1.3 Pseudo Code

```
FUNCTION open image(path):
    TRY:
        RETURN image if image is already an Image
    ELSE:
        RETURN open(path)
    EXCEPT File Not Found Error
        PRINT "Could not open " + path
        EXIT
FUNCTION convert to bits(chars, encoding):
    FOR EACH character IN chars:
        CONVERT character to binary and pad it to match encoding
        APPEND padded binary character TO bits list
```

	RETURN bits list
FUNCTION set LSB (component, bit):	
	RETURN the component with the least significant bit replaced by the given bit
CLASS Basic Image Hide:	
	MAIN METHOD (image, message, encoding):
	SET index to 0
	SET image to image
	SET message to message
	SET encoding to encoding
	SET carrier image to None
	METHOD encode():
	OPEN image
	PREPARE message for embedding
	EMBED message into image
	SHOW_INFO('Success', 'Text has been encoded successfully!')
	RETURN carrier image
	METHOD open image():
	OPEN the image

	METHOD prepare message():	
		GET length of message and prepend it to message
		CONVERT message to binary bits
	METHOD embed message():	
		ITERATE over pixels in image
		ENCODE message bits into pixel colours
	METHOD encode pixel(coordinate):	
		REPLACE least significant bits of pixel colours with message bits
CLASS Basic Image Reveal		
	MAIN METHOD (image, encoding):	
		SET image
		SET ENCODINGS[encoding]
	METHOD decode():	
		OPEN image
		ITERATE over pixels in image
		DECODE message from pixel colours
	METHOD decode pixel(coordinate, image):	
		EXTRACT least significant bits from pixel colours

	ASSEMBLE characters from bits
	IF complete message is assembled, RETURN True
	ELSE, RETURN False

3.5.1.1.4 Strengths

The algorithm to be implemented provides a high level of security facilitated by the encryption of the secret message, which safeguards the encoded data against unauthorised activity. Moreover, the algorithm employs an efficient technique for concealing text within images by manipulating the least significant bits of the RGB components of image pixels. This approach allows messages to be embedded without significantly altering the visual appearance of the carrier file. Using LSB manipulation provides a layer of security for the hidden message, as the alterations have minimal impact on the image's overall colour. They remain imperceptible to the human eye, thereby safeguarding the concealed information.

3.5.1.1.5 Limitations

The LSB algorithm is vulnerable to compression algorithms, such as JPEG compression. During compression, the LSBs may be altered, resulting in data loss or potential corruption of the hidden message. This unintended loss of concealed information can occur during image processing. Moreover, the level of security provided by LSB may not be adequate to conceal sensitive data. Advanced steganalysis techniques can detect alterations to the least significant bit, potentially uncovering the hidden information.

3.5.1.2 Discrete Wavelet Transform (DWT)

The following implementation will utilise the PyWavelets library, which is accessible for Python. This library facilitates the transformation of image files into coefficients and enables the extraction of the lowest-level approximation coefficients. Figure 12 presents a visual representation of the proposed algorithm. Initially, the carrier image is transformed into the spatial domain, focusing on the lowest coefficients. Next, the LSB technique is applied to encode the secret message into the least significant bits of the lowest approximation coefficient. Finally, the image is reconstructed to generate the stego image.

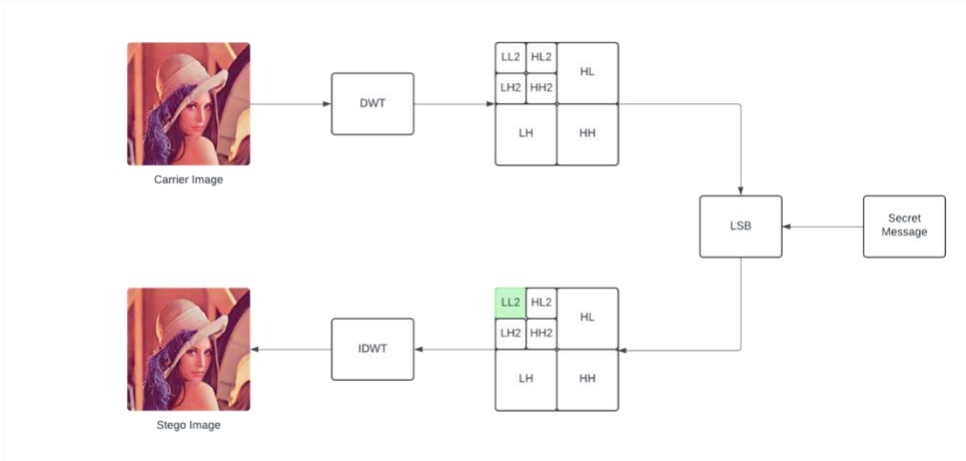


Figure 12 - DWT Encoding Visualisation

3.5.1.2.1 Hiding

The hiding process commences with encoding the message intended for concealment within the carrier image. A marker is embedded at the end of the secret message to locate its conclusion to facilitate accurate extraction during retrieval. The message is then converted into its binary representation, where each character is represented by its corresponding ASCII value in binary format. This binary representation is further segmented into individual bits, each designated for embedding into the image. The carrier image is read into memory

followed by message preparation using the OpenCV image processing library. Wavelet transformation is subsequently applied to the image, decomposing it into various frequency components. This decomposition separates the image into approximation and detail coefficients along different directions (horizontal, vertical, and diagonal). The approximation coefficients, capturing low-frequency components, and the detail coefficients, representing high-frequency components, are distinguished. The message bits are then embedded into the lowest approximation coefficients of the image using LSB substitution, replacing the least significant bit of each pixel value with a bit from the message. This minimal modification minimises visual alterations to the image, as the least significant bit makes minimal alterations to pixel values. Upon completing the embedding process, the modified approximation coefficients are transformed back into the spatial domain using inverse wavelet transform (IDWT), successfully encoding the carrier image.

3.5.1.2.2 Extracting

The extraction process initiates by loading the carrier image, which stores the secret message, into the memory. Following a similar approach to the hiding process, wavelet transformation is employed to decompose the image into approximation and detail coefficients. Subsequently, the LSBs of the pixel values within the lowest approximation coefficients are extracted. These LSBs represent the hidden message bits concealed within the image. Following the extraction, the retrieved message bits are then organised and converted back to the original message format, reversing the encoding process employed during hiding to reconstruct the message accurately. Upon message reconstruction, the marker utilised to indicate the end of the message is identified and removed, ensuring only the original message content is retrieved. Finally, the revealed message is obtained and made available for display to the user.

3.5.1.2.3 Pseudo Code

```
FUNCTION convert message to bits(message):
    CONVERT each character in message to binary representation
    RETURN list of binary bits
FUNCTION convert bits to message(bits):
    CONVERT binary bits back to characters and concatenate them
    RETURN the reconstructed message
FUNCTION encode pixel(pixel value, bit to encode):
    GET the least significant bit of the pixel value
    ENCODE the message bit using the least significant bit
    RETURN the modified pixel value
CLASS Advanced Image Hide
    MAIN METHOD (image path, message):
        LOAD image from image path
        CONVERT message to binary bits
        SET index to 0
    METHOD encode image():
        SPLIT image into colour channels
```

	APPLY wavelet transform to each channel
	ENCODE message bits into approximation coefficients
	RECONSTRUCT image from modified coefficients
	SHOW success message
	RETURN encoded image
	METHOD encode channels(red coeff, green coeff, blue coeff):
	ENCODE message bits into each colour channel
	RETURN modified coefficients for each channel
	STATIC METHOD inverse wavelet transform(encoded coeff):
	APPLY inverse wavelet transform to reconstruct image
	RETURN reconstructed image
CLASS Advanced Image Reveal	
	MAIN METHOD (image path):
	LOAD image from image path
	EXTRACT approximation coefficients from image
	METHOD extract bit sequence():
	ITERATE through approximation coefficients
	EXTRACT least significant bits and form a bit sequence

		RETURN the bit sequence
	METHOD decode message():	
		DECODE message from bit sequence
		REMOVE any trailing markers from the decoded message
		RETURN the decoded message
	METHOD get decoded message():	
		RETURN the decoded message

3.5.1.2.4 Strengths

Combining the DWT and LSB embedding can significantly enhance the security of hidden messages. The wavelet transformation disperses the message throughout the image, making it challenging to detect through visual inspection. Subsequently, LSB embedding conceals the message within pixel values, further enhancing its concealment. Moreover, the algorithm ensures robustness by employing wavelet transform, which distributes the embedded message across various frequency bands within the image. This distribution enhances the message's resilience against typical image processing operations like compression or resizing. Additionally, unlike other methods, integrating these two algorithms enables a larger volume of data to be hidden within the image without compromising its visual quality. This versatility makes the algorithms suitable for concealing messages of varying lengths.

3.5.1.2.5 Limitations

This algorithm's limitations arise from the restricted capacity for hiding information within images, influenced by factors like image size and the number of bits that can be altered without noticeable distortion. Despite the algorithm's resilience to compression, the embedded message remains vulnerable to lossy compression algorithms commonly employed in image transmission and storage.

3.5.2 Text Steganography

Text steganography, the second concealment method, will be implemented using only a basic algorithm, enabling the encoding of textual data within text files. This section delves into the algorithm to be implemented and its operational process.

3.5.2.1 Least Significant Bit (LSB)

The plan is to implement the LSB algorithm without dependency on any specific Python library for encoding and decoding. Additionally, a layer of security will be added through encryption to enhance the robustness of the encoded data.

3.5.2.1.1 Hiding

The hiding process begins with identifying the carrier text file, which will hold the secret message and the secret message. Firstly, the carrier text file is examined to determine the number of words it can contain. Once the word count is established, the application calculates the maximum concealment capacity of the carrier, referring to the maximum number of characters that can be encoded within. This step ensures the secret message is not too long to be concealed within the carrier. If the secret message exceeds the capacity of the carrier file, an error message is triggered, advising the user to either expand the carrier text or shorten the secret message to make it compatible with the carrier file.

The encoding process can be initiated, assuming the secret message fits within the carrier's capacity. Initially, the message transforms into a series of bits utilising a predefined encoding scheme, where each character of the secret message is converted into a specific sequence of bits, with distinct markers indicating whether the character is a symbol or a number. Subsequently, an XOR encryption with a designated key is applied to obscure the encoded data to enhance security. This encryption method ensures that the hidden message remains secure and is not easily deciphered without the correct key.

Finally, the encoded bits are embedded within the carrier file. To achieve this, zero-width characters, typically invisible or challenging to perceive, conceal the encoded bits within the carrier. Each character in the carrier text serves as a

container for a portion of the encoded message, ensuring the message remains hidden and intact within the carrier file.

3.5.2.1.2 Extracting

The extraction process begins by reading the encoded text file and extracting the binary data concealed within the carrier text. This extracted binary data is then processed and reconstructed to retrieve the original message, using codes to distinguish between symbols and numbers.

Following the processing phase, XOR decryption is applied with the same key used during encoding, revealing the original text. Finally, the binary data is converted into characters to reconstruct the hidden message fully.

3.5.2.1.3 Pseudo Code

```
CLASS Basic Text Hide
    MAIN METHOD (message, carrier text):
        CALCULATE word count from carrier text
        CALCULATE max insertable words from word count
        IF length of message is greater than word count:
            SHOW error message
        ELSE:
            ENCODE message into carrier text
    METHOD get carrier word count(carrier text):
```

			OPEN carrier text
			COUNT words in carrier text
			RETURN word count
		METHOD encode text(message, carrier text):	
			ENCODE message into bits
			ASK user for save path
			OPEN carrier text for reading and encoded file for writing
			EMBED encoded data into carrier text and save result
		STATIC METHOD encode message(message):	
			CONVERT message into bits with prefix and XOR
			APPEND end marker
			RETURN encoded bits
		STATIC METHOD embed data in carrier(encoded data, carrier file, encoded file):	
			DEFINE zero-width characters for embedding data
			READ carrier file and split into words
			FOR each word in carrier words:
			FOR each bit pair in encoded data:
			EMBED bit pair into word using zero-width

		characters
		WRITE word with embedded data to encoded file
CLASS Basic Text Reveal		
	MAIN METHOD (file path):	
		INITIALIZE temp and final variables
		OPEN file path for reading
		EXTRACT binary data from zero-width characters
		DECODE extracted binary data
	METHOD decode text():	
		LOOP through extracted binary data in 12-bit chunks
		EXTRACT control code and data from each chunk
		DECODE data based on control code
	STATIC METHOD binary to decimal(binary):	
		CONVERT binary to decimal and return

3.5.2.1.4 Strengths

The proposed algorithm utilises zero-width characters to conceal secret data effectively, maintaining the visual appearance of the text while rendering the encoded content invisible. This makes it challenging for unauthorised recipients to detect the presence of hidden messages. Moreover, the algorithm ensures

robustness through XOR encryption, which enhances the security of the encoded data. Deciphering the hidden message becomes difficult for unauthorised parties without knowledge of the encryption key. Integrating the algorithm into existing systems requires minimal effort, as it relies on standard Python libraries for GUI elements and file handling. This compatibility ensures its suitability across a wide range of environments. Furthermore, the proposed method incorporates error-handling mechanisms to notify users in case of errors. One example of its application is alerting the users if the secret message exceeds the carrier's capacity, helping to mitigate the risk of data loss and ensuring the reliability of the encoding process.

3.5.2.1.5 Limitations

This solution relies on a simple algorithm, which may not offer the highest level of security against advanced steganalysis methods. The employed algorithm utilises a basic embedding technique, where each carrier holds a set amount of secret data.

3.6 Evaluation Criteria

The software will be evaluated across three main dimensions: Performance, Security and Usability.

Performance assessment will incorporate several critical factors, including the speed of encoding and decoding processes, memory consumption, and scalability. Comparative benchmarking against existing software will be undertaken to measure the efficiency and efficacy of the software's data concealment and extraction techniques.

Security evaluation will prioritise the robustness of the encoding mechanism against a range of steganalysis techniques, ensuring the data remains hidden

from detection. Additionally, emphasis will be placed on the imperceptibility of visual alterations made to the carrier, enhancing the software's ability to maintain confidentiality.

Usability testing will evaluate the intuitiveness of the graphical user interface (GUI) and overall ease of use for end-users. Feedback gathered from user testing sessions will be analysed to identify potential bugs and areas for improvement, ultimately enhancing the user experience and functionality of the software.

CHAPTER 4

IMPLEMENTATION OR INVESTIGATION

4.1 Introduction

The implementation section outlines the approach and methodology adopted in developing the steganography software. It describes the implementation approach, development methodology, and specifications regarding the development environment, programming languages, and technologies. Furthermore, it discusses the design and implementation of the GUI using the Tkinter library, tailored to both technical and non-technical users. Moreover, it provides an extensive overview of the testing procedures, including unit, integration, functional, and user testing, ensuring the software's robustness and reliability. Through these comprehensive measures, the implementation section underscores the commitment to delivering a secure, user-friendly, and thoroughly tested steganography solution that meets the highest quality and performance standards.

4.2 Implementation Approach

The proposed plan follows an iterative and incremental implementation process to accommodate changes in requirements and additional implementations flexibly, should they be necessary. The development process is divided into phases, concentrating on specific features, functionalities, and software algorithms.

4.3 Methodology

The development process is effectively managed using Agile methodologies, particularly Scrum. Scrum is known for its iterative and incremental approach and facilitates the delivery of high-quality products with specified timelines. This involves organised development sprints focused on enhancing the product incrementally (Wrike, 2023). Throughout the development of the solution, short development cycles have been implemented, with frequent reviews and adjustments based on testing outcomes and user feedback. Early-stage prototypes are created to validate design models and implement features iteratively. This rapid prototyping approach ensures that the final software aligns with the objectives and meets the assigned criteria.

4.4 Development Environment

The proposed software was developed in a controlled environment optimised for software development and testing. The following section details the workstation and hardware specifications used for the implementation.

Development Workstation

- Model: MacBook Pro 2020
- Processor: Apple Silicon M1
- Memory: 8GB LPDDR4
- Storage: 256GB SSD
- Operating System: macOS Sonoma 14.4.1

Development Tools and Software

- IDE: PyCharm Professional 2023.3.4
- Programming Language: Python 3.11
- Version Control: Git 2.44.0

Testing Hardware

The steganography software was processed for heavy testing on various hardware configurations for testing and evaluation purposes. This included:

- Multiple Operating Systems: Testing on Windows and macOS.
- Different Processor Architectures: Evaluating Intel and AMD processors to ensure platform compatibility and performance.

4.5 Programming Language and Technologies

The proposed software has been developed utilising Python; a high-level, object-oriented programming language known for its interpretive capabilities. Python serves as the foundation for both the front-end and back-end development. Several essential libraries have been integrated into the development process. Here is a list of them:

- OpenCV 4.9.0.80
    - OpenCV is a library that offers various capabilities for processing images and videos, with image manipulation being the most crucial for developing the solution.
- Pillow 10.2.0
    - Pillow library enables image file opening, manipulation, and closing. Its functionalities include basic image operations and the conversion of image file formats.
- PyWavelets 1.6.0
    - PyWavelets is a library that assists in wavelet transformation and signal processing, offering multi-level wavelet decomposition capabilities.
- Tkinter

    - Tkinter is a library that enables the creation of GUIs in Python, offering a variety of widgets and tools for UI development.
- CustomTkinter 5.2.2
    - CustomTkinter is an improved version of Tkinter that includes various customisations.
- OS
    - The "os" module offers access to operating system functionalities, including file system operations, process handling, and other system-related tasks.
- NumPy 1.26.4
    - NumPy is a package that facilitates scientific computing in Python, supporting multi-dimensional arrays and offering a range of mathematical functions to manipulate these arrays efficiently.
- Cryptography 42.0.5
    - The Cryptography library facilitates the encryption and decryption of textual data using either a predefined key or a key randomly generated by an algorithm.

Git was employed for version control management throughout the software development lifecycle. This allowed tracking of changes, ensuring project integrity, and facilitating seamless integration of new features. The software development process occurred within an Integrated Development Environment (IDE), specifically PyCharm, designed for Python development by JetBrains (JetBrains, 2024). These tools and programming languages were selected based on their suitability, ease of implementing steganography algorithms, and deep understanding of Python. Using Python, the steganographic software was effectively developed while maintaining high performance, security, and usability standards.

4.6 UI Implementation

For the implementation of the Graphical User Interface, the Python library Tkinter was used, allowing efficient UI development along with a seamless integration with the back end. The design did not change much from the early prototype, except that it was polished.

4.6.1 Dashboard

The following figure showcases the GUI of the dashboard, which serves as the initial window shown to users upon launching the application. This window has been designed with non-technical users in mind while offering intuitive functionality and essential information.

Figure 13 - Dashboard

The image showcases the application name, “PixelBlend”, prominently displayed for easy readability. A tab view has also been incorporated, allowing users to

conceal or extract the secret message from the carrier. This functionality is identifiable through the tabs labelled “Hide” and “Reveal”. Within each tab, various types of steganography are listed, each represented by a corresponding button, which remains consistent between the “Hide” and “Reveal” tabs, featuring buttons for “Image”, “Text”, “Video”, and “Audio” steganography methods.

4.6.2 Video and Audio Steganography

When the user clicks on the buttons associated with audio and video steganography, a message appears, informing them that these features' encoding and decoding functionality is not yet available. Instead, it will be implemented in a future update. Figure 14 illustrates the alert displayed to the user.

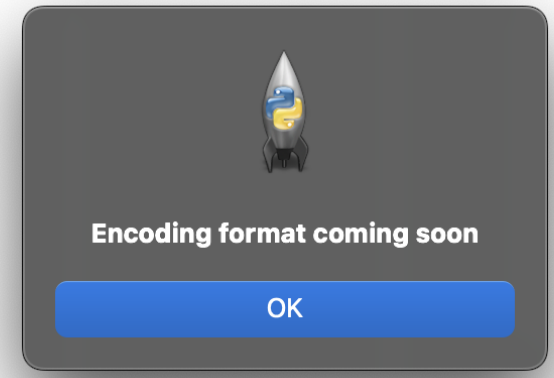


Figure 14 - Audio & Video Steganography

4.6.3 Image Steganography

The following figures showcase the GUI implemented for encoding and decoding images using various algorithms.

4.6.3.1 Hiding Window

Figure 15 presents a straightforward visualisation of the encoding process for images. The primary focus is on two prominent sections occupying most of the window space. On the left side, a preview of the uploaded image is displayed, while the right-side section serves as an input feed for entering the secret message to be encoded. A hint text has been included to guide users on where to input the secret text to enhance user-friendliness. Below the sections, a text field is provided, where the encryption key for the secret message is displayed once the encoding process is completed. This key will also be utilised during the decoding process. Therefore, the encoded data will also be lost if the key is lost. Currently, this feature is only available for encoding using the basic algorithm. Below, a list is provided to enable users to select the algorithm level for encoding the text. The available options include "Basic" and "Advanced". As previously discussed, the "Basic" option utilises the LSB algorithm, while the "Advanced" option employs the DWT algorithm. At the bottom of the window, two buttons are situated: one to upload the carrier image and the other to commence the encoding process.

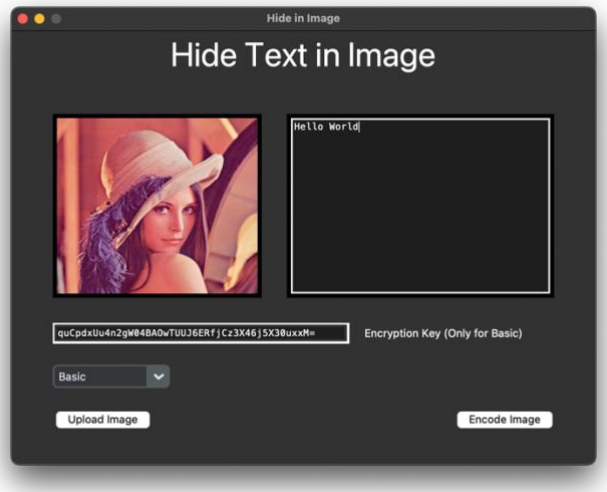


Figure 15 - Image Steganography

4.6.3.2 Image Selection

The image below illustrates the process of uploading an image to the application. When the user clicks on the “Upload Image” button, the operating system’s file manager window opens, enabling the user to select the desired image for encoding. For confidentiality reasons, the image has been pixelated to conceal sensitive data on the development machine. Additionally, at the bottom of the image, a snippet displays a pre-set list of options users can upload. This ensures that users can only upload image formats supported by the application. The available options include PNG, JPG, and JPEG formats.

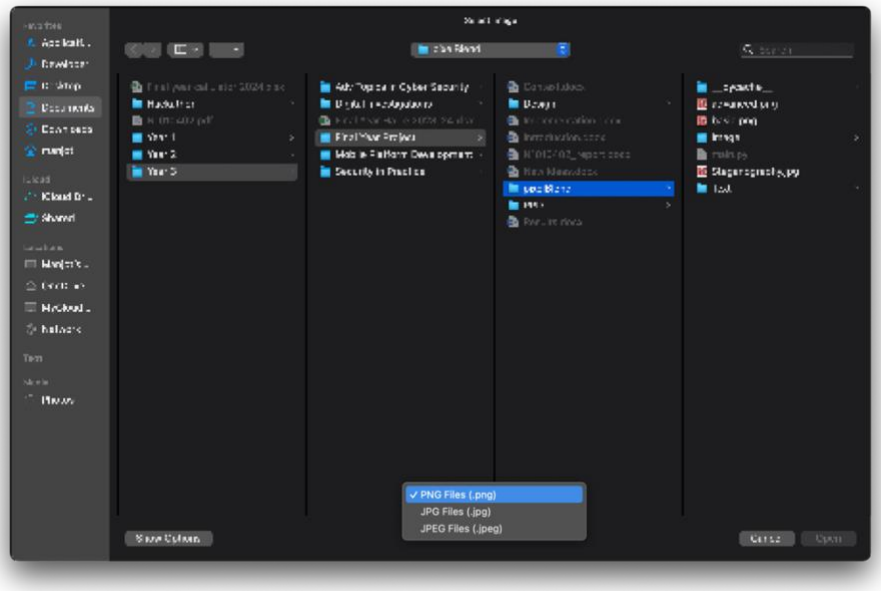


Figure 16 - Image Selection

If no image is uploaded or no message is provided, an error message appears, informing the user to upload the carrier image and secret message. Without completing these steps, the encoding process cannot be initiated.

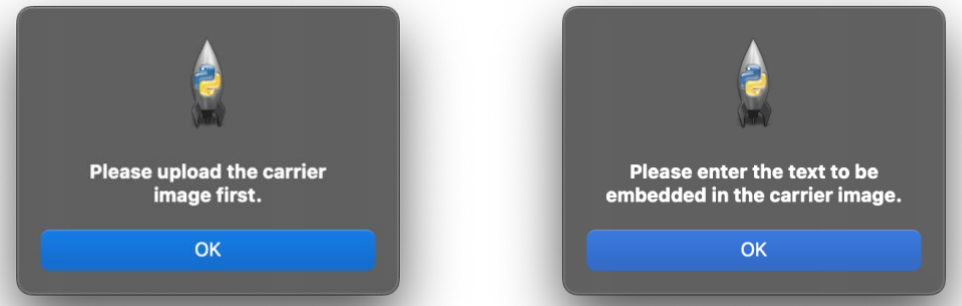


Figure 17 - Error Messages

4.6.3.3 Extraction Window

When extracting the concealed message, the window appears similar to the hiding window. However, the title can easily distinguish the windows, which reads “Extract Text from Image”. The window comprises two sections: one displaying the uploaded image and the other displaying the message extracted from the stego image. An entry for the encryption key is provided to decode images encoded using the LSB algorithm. The user must input this key and match the one provided during encoding.

Furthermore, users are presented with a selection of the algorithm level utilised during encoding, which can either be “Basic” or “Advanced”. The user must know the exact level of security chosen during encoding. Finally, two buttons are provided at the bottom of the window: one for uploading the stego image and the other to initiate the decoding process.

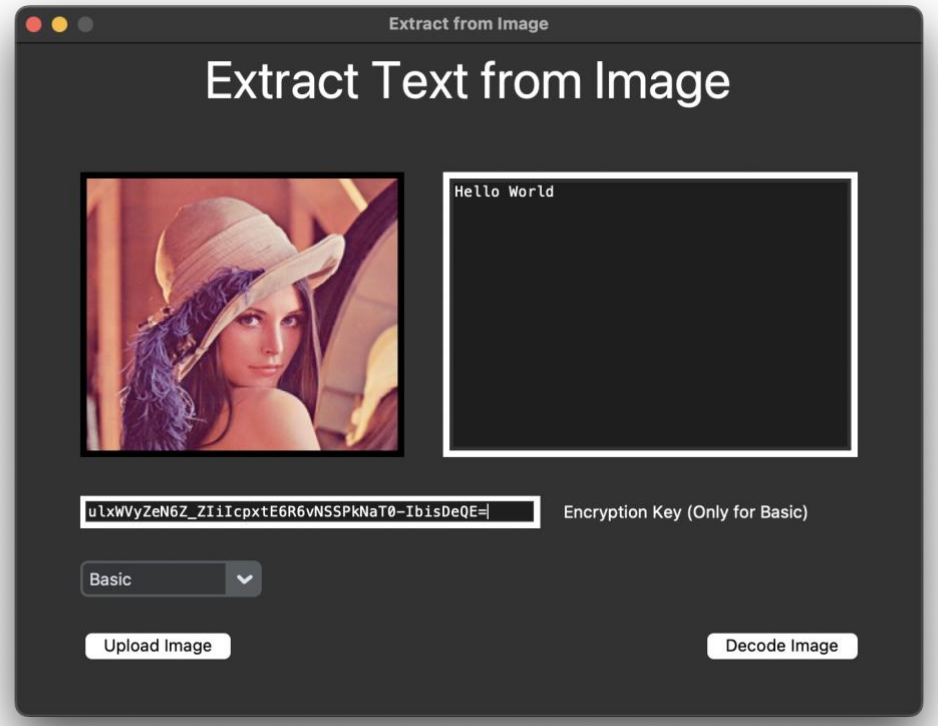


Figure 18 - Image Steganography Extraction Window

4.6.3.4 Messages

The extraction window includes similar error handling mechanisms as the encoding process, such as checking whether the image has been uploaded. Additionally, if the decoding algorithms fail to detect any message, the error message displayed in the following figure informs the user to verify the decoding level, as it might be incorrect or if the uploaded image was indeed the stego image.

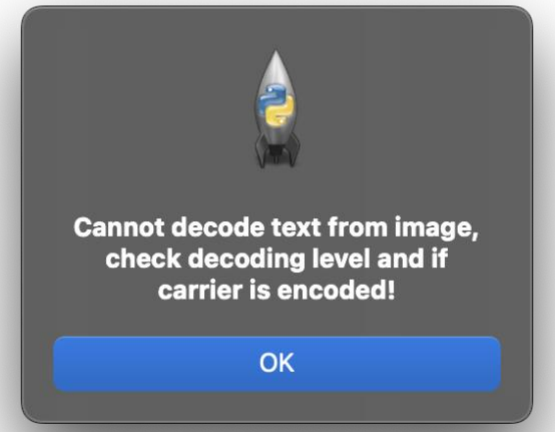


Figure 19 - Error Message

If the decryption key provided by the user is incorrect, an error message is displayed, informing the user that the provided key is incorrect. Additionally, if the user fails to provide the decryption key, another error message prompts them to input it.

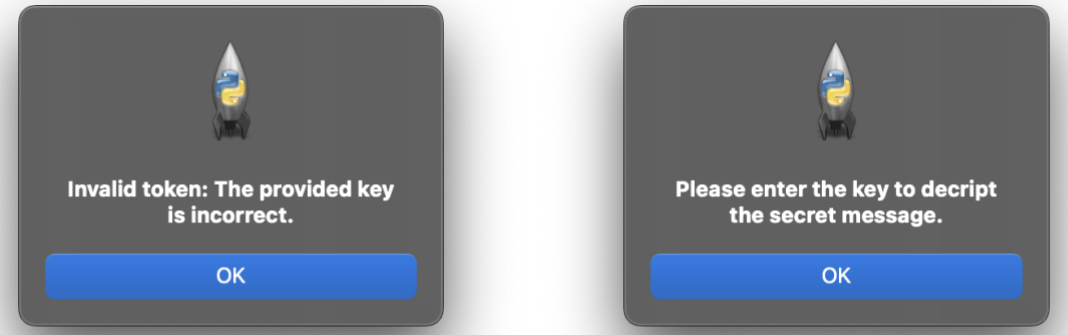


Figure 20 - Error Messages

4.6.4 Text Steganography

The following figures showcase the GUI implemented for encoding and decoding text files.

4.6.4.1 Hiding Window

Figure 21 provides an overview of the window for hiding data within a text file. The critical components of the window include two sections: one for displaying the text contained in the carrier file and the other for inputting the secret message. The latter section features a hint text suggested by a user during user testing. Below these sections, a list allows users to choose the encoding algorithm level, with options between "Basic" and "Advanced". The "Basic" option utilises the LSB algorithm, while the "Advanced" option is not implemented but is planned for future deployment. At the bottom of the window, two buttons are included: one for uploading the text file and the other to initiate the encoding process.

Figure 21 - Text Encoding Window

4.6.4.2 Messages

The error message displayed during the encoding and decoding process mirrors those previously shown, verifying whether the carrier file has been uploaded and if the secret message has been inputted. Additionally, a message box appears if the user selects the "Advanced" algorithm for encoding. This message informs the user that the selected algorithm is unavailable and will be implemented in future software updates.

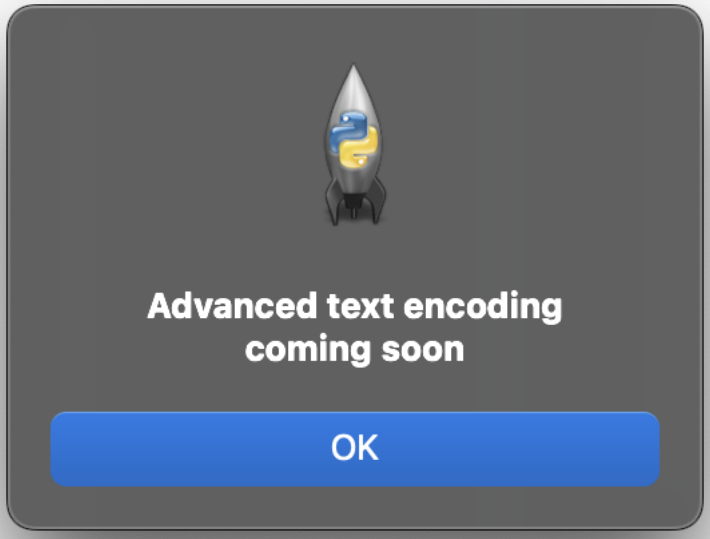


Figure 22 - Error Message

4.6.4.3 Extraction Window

When extracting the concealed message, the displayed window resembles the hiding window, but it can be easily distinguished by the title, which reads "Extract Text from Text". There are two sections: one displaying the uploaded step text file and the other displaying the message extracted from the stego file. Additionally, a selection of the algorithm level utilised during encoding is provided, with options including "Basic" and "Advanced". The user must know the exact level of security utilised during encoding. However, the "Advanced" algorithm has not yet been implemented and is planned for future updates. Finally, at the bottom of the window, two buttons are provided: one for uploading the stego file and the other to initiate the decoding process.

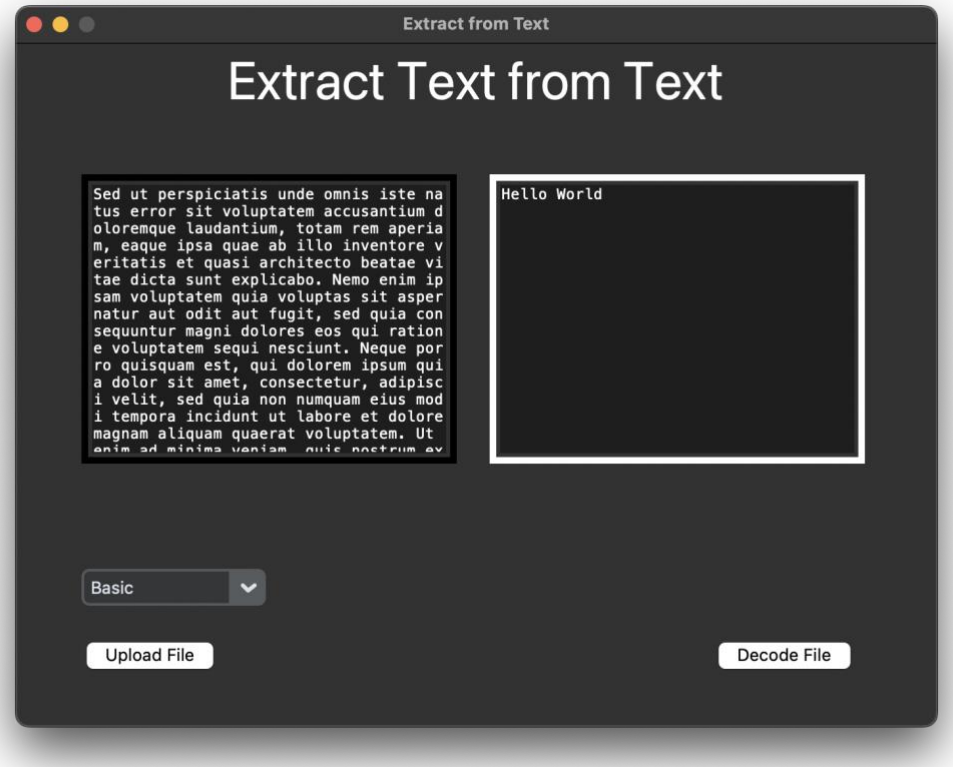


Figure 23 - Text Extraction Window

4.7 Testing

Numerous tests have been conducted to verify the smooth operation of the developed application. Various types of testing, including unit, integration, functional, and user testing, have been performed. These tests help ensure proper error handling and identify potential code script bugs.

4.7.1 Unit Testing

Unit testing assesses individual functions and modules of an application to ensure their proper functioning, including encoding, decoding, and error handling. The following table is organised into different columns: a description of the conducted tests, the expected result, the actual result, and whether the test passed or failed.

Table 2 - Unit Testing

Test ID	Description	Expected Results	Actual Results	Pass/Fail
1	Test "Basic" image encoding function with valid input	The secret message is correctly encoded in the carrier	Encoded data matches the expected output	Pass
2	Test "Basic" image encoding function with invalid input	Messagebox is shown to the user displaying an error	Appropriate error handling	Pass
3	Test the "Basic" image decoding function with valid	The secret message is correctly extracted	Decoded data matches the	Pass

	input	from the carrier file	expected output	
4	Test the "Basic" image decoding function with invalid input	Messagebox is shown to the user displaying an error	Appropriate error handling	Pass
5	Test "Advanced" image encoding function with valid input	The secret message is correctly encoded in the carrier	Encoded data matches the expected output	Pass
6	Test "Advanced" image encoding function with invalid input	Appropriate error handling	Appropriate error handling	Pass
7	Test the "Advanced" image decoding function with valid input	The secret message is correctly extracted from the carrier file	Decoded data matches the expected output	Pass
8	Test the "Advanced" image decoding function with invalid input	Appropriate error handling	Appropriate error handling	Pass
9	Test "Basic" text encoding function with valid input	The secret message is correctly encoded in the carrier	Encoded data matches the expected output	Pass

10	Test "Basic" text encoding function with invalid input	Messagebox is shown to the user displaying an error	Appropriate error handling	Pass
11	Test "Basic" text decoding function with valid input	The secret message is correctly extracted from the carrier file	Decoded data matches the expected output	Pass
12	Test "Basic" text decoding function with invalid input	Messagebox is shown to the user displaying an error	Appropriate error handling	Pass

4.7.2 Integration Testing

Integration testing evaluates the interaction between various modules and components to ensure seamless operation. It verifies that different application parts work seamlessly, including the interface between the steganography modules and the user interface. The following table is organised into different columns: description of the test, expected result, actual result, and test outcome.

Table 3 - Integration Testing

Test ID	Description	Expected Results	Actual Results	Pass/Fail
1	Test integration with error handling	The application should handle errors gracefully and provide	Appropriate error handling as expected for	Pass

		the user with informative error messages	known errors	
2	Test integration between encoding and decoding modules	The application should successfully embed and extract data from carriers	The application operates as expected	Pass
3	Test interface between the steganography module and the UI	Users should be able to select carrier files and the secret message without errors	The integration works as expected	Pass
4	Testing integration with I/O operations	The application should appropriately read and write files to the filesystem	The application works as expected	Pass
5	Test integration with image processing	The image carriers should retain quality and integrity after encoding and decoding the message	The application works as expected	Pass
6	Test integration with different encoding algorithms	The application should be able to encode and decode the carrier by utilising different algorithms depending	The application works as expected	Pass

on the input

4.7.3 Functional Testing

Functional testing examines the application's overall functionality to ensure it can effectively embed data within carrier files and extract it without loss or corruption. The table below contains the description of each test, the expected result, the actual result, and the test outcome.

Table 4 - Functional Testing

Test ID	Description	Expected Results	Actual Results	Pass/Fail
1	Test embedding data into image carrier utilising the basic algorithm	Data successfully hidden	The outcome matches the expected result	Pass
2	Test extracting data from image carrier utilising the basic algorithm	Data successfully extracted	The outcome matches the expected result	Pass
3	Test embedding data into image carrier utilising the advanced algorithm	Data successfully hidden	The outcome matches the expected result	Pass
4	Test extracting data from image carrier utilising the advanced algorithm	Data successfully extracted	Extracted data differs from the encoded data	Fail
5	Test extracting data from	Data	The outcome	Pass

	image carrier utilising advanced algorithm, after modifications	successfully extracted	matches the expected result	
6	Test embedding data into text carrier	Data successfully hidden	The outcome matches the expected result	Pass
7	Test extracting data from text carrier	Data successfully extracted	The outcome matches the expected result	Pass

4.7.4 User Testing

User testing involves interacting with the application to offer feedback on its usability, functionality, and overall experience. The table below presents the identification of each user, the test description, and the feedback received from the user.

Table 5 - User Testing

Test ID	Tester ID	Description	Feedback Received
1	User 1	Embedding data into an image file	The user liked the clean and intuitive interface design and encountered no issues during testing. Suggested adding some hint text where the secret text needs to be inputted

2	User 1	Extracting data from the image	The user found the extraction process intuitive and easy to understand. He/she encountered no issue with performance usage.
3	User 2	Embedding data into a text file	The user encountered a new error when clicking on the box displaying the carrier text.
4	User 3	The overall functionality of the application	The user found the application intuitive and straightforward in its user interface. He/she suggested the implementation of a full-sized software with a more sophisticated UI
5	User 4	Tested application, trying to find bugs	The user found a bug that does not allow the carrier text to be displayed on the screen if it has been clicked

CHAPTER 5

RESULTS / DISCUSSION

5.1 Introduction

The literature presents the findings of the steganography software, demonstrating its effectiveness in concealing information within digital images and text. The software utilises basic and advanced algorithms, and this section offers insights into their performance, capacity, robustness, and security. The application's performance is evaluated by examining the computational efficiency, showcasing the processing times for encoding and decoding using different algorithms. Capacity is assessed to demonstrate the software's capability to embed textual data within images, highlighting the maximum amount of text that can be concealed without compromising image quality. Moreover, the security and robustness are analysed to assess the software's resilience against various attacks, such as compression and manipulation. PSNR and SSIM metrics provide insights into the fidelity and similarity between original and stego images, providing quantitative measures of the algorithm's performance in preserving image quality during data embedding. Online decoders and HEX editors are utilised to analyse the robustness against external attempts to extract concealed data, emphasising the software's ability to withstand such attacks. Finally, comparative analysis visually compares the original and modified images and texts to evaluate the effectiveness of the embedding process.

5.2 Performance Evaluation

The implemented solution's performance has been evaluated across various metrics, such as computational efficiency, embedding capacity, robustness, security, PSNR, SSIM, online decoders, HEX editors, metadata comparison, and byte-by-byte comparison. This evaluation ensures the application's effectiveness in concealing secret data within a cover medium. If the hidden data is easily detectable, the algorithm is not dependable. Moreover, as steganography is primarily used for covert communications and data concealment, assessing the performance is crucial for ensuring data security.

5.2.1 Computational Performance

The encoding timings for each implemented algorithm have been calculated using the machine utilised for system development, and the results are as follows:

- LSB Image Encoding: 0.0089 seconds
- DWT Image Encoding: 0.0362 seconds
- LSB Text Encoding: 0.0003 seconds

Similarly, decoding timings for each algorithm have been calculated and reported:

- LSB Image Decoding: 0.0283 seconds
- DWT Image Decoding: 0.0256 seconds
- LSB Text Decoding: 0.0007 seconds

Despite variations in algorithm complexity, both encoding and decoding timings for each algorithm remain relatively fast, all completing processing within 1 second. These results indicate high computational efficiency in performing their respective tasks. Even with the slowest processing time of 0.0362 seconds, the

application is expected to perform well on machines with limited computational resources, ensuring tasks can be handled without excessive strain on system resources.

5.2.2 Capacity

The image steganography algorithms were tested to encode textual data containing 650 words using both LSB and DWT algorithms. The maximum amount of textual data encoded within the image using LSB or DWT was determined to be 421 words. This encoding limit does not affect the image quality but impacts the extraction process, which cannot extract data longer than this limit. Any data extracted beyond the 421st word appears unreadable, as the characters displayed cannot be recognised as readable. Additionally, text steganography was tested for its maximum capacity, which dynamically changes depending on the amount of text stored within the text file. Therefore, a specific maximum encoding number is not defined, as it constantly fluctuates.

5.2.3 Robustness and Security

The implemented image steganography algorithms underwent testing against attacks such as PNG compression. In the case of the LSB algorithm, the encoded data was completely lost during this process, indicating that the basic algorithm lacks security against deep compression. Similarly, the DWT algorithm was tested against the same compression process, resulting in the complete loss of the encoded data. However, some undecipherable characters were still decoded, none resembling the original encoded data. Regarding text steganography, the algorithm was analysed against text manipulation techniques such as copying, pasting, and editing the stego text generated by the LSB algorithm. Editing the stego results in partial loss of encoded data, depending on the extent of the

edits made. Additionally, erasing or modifying the stego file data in large quantities poses the risk of completely losing the secret message encoded.

5.2.4 Peak Signal-to-Noise Ratio (PSNR)

The Peak Signal-to-Noise Ratio (PSNR) measures the ratio between the maximum possible power of an image and the power of any corrupting noise affecting its representation. To calculate the PSNR of an image, it is compared to a clean image having the maximum possible power. PSNR serves as an indicator of the quality of a steganography algorithm. Higher PSNR values signify better fidelity between the original and modified images, while lower values indicate more significant distortion. PSNR values reflect how much the embedded data has impacted the visual quality of the cover image. Generally, good PSNR values vary depending on the application's requirements, but the following guidelines can be followed:

- PSNR values exceeding 30 dB typically represent a high-quality reconstruction, with differences between the original and stego images imperceptible to the naked eye.
- PSNR values between 20 and 30 dB indicate moderate to acceptable reconstruction quality, with slight modifications noticeable but not objectionable.
- PSNR values below 20 dB generally indicate poor reconstruction quality, with significant visual alterations.

Two image steganography algorithms, one advanced and one basic, were implemented and tested, yielding the following results:

- LSB Algorithm (Basic): PSNR value of 77.37 dB
- DWT Algorithm (Advanced): PSNR value of 70.88 dB

These results show that the basic algorithm achieved a higher PSNR value than the advanced algorithm, indicating better fidelity between the original and modified images in terms of PSNR alone.

5.2.5 Structured Similarity Index Measurement (SSIM)

The Structured Similarity Index Measurement (SSIM) assesses the similarity between the cover and stego images. SSIM helps assess the stego image's quality in image steganography compared to the original cover image. Unlike comparing overall image statistics, SSIM analyses similarities locally by dividing images into smaller regions and comparing their similarities. A Python script has been developed to calculate the similarities between stego images generated using the LSB and DWT algorithms and the original carrier image, yielding the following results:

- LSB Algorithm (Basic): Similarity value of 0.9999981683161097
- DWT Algorithm (Advanced): Similarity value of 0.999981396724419

The first score indicates an almost identical match between the two images, with an extremely high level of similarity and any differences likely to be negligible or imperceptible to the human eye. Similarly, the second score suggests a high similarity between the stego and original images. Although slightly lower than the first score, it is close to 1, indicating that the images are nearly indistinguishable.

5.2.5.1 LSB Image

The following figure illustrates the modifications between the original carrier image and the stego image when utilising LSB, indicating the areas where data is concealed within the image.

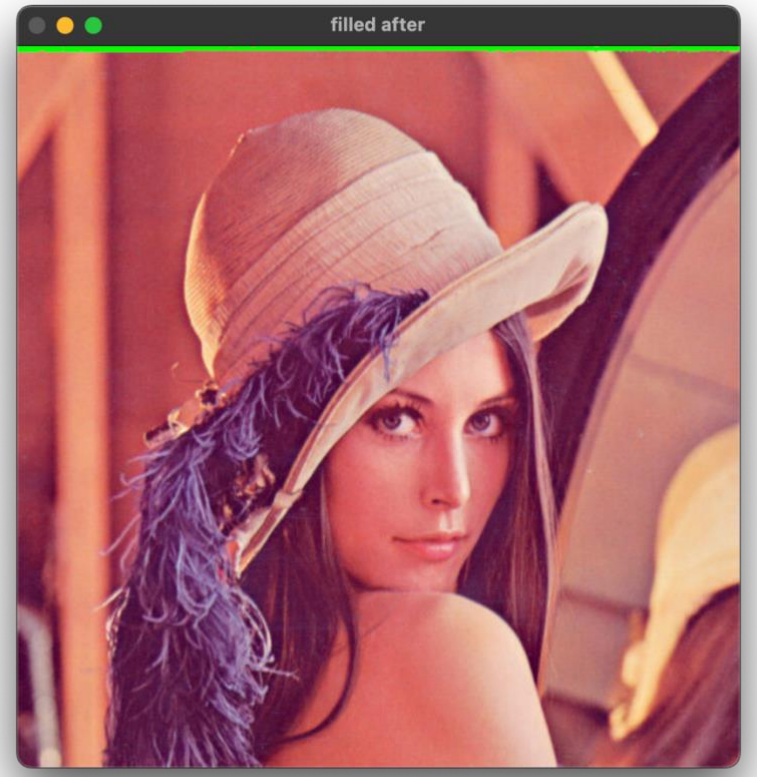


Figure 24 - SSIM for LSB

As depicted in Figure 24, the LSB algorithm modifies the image pixels to sequentially store the encoded message at the top of the image, represented by the colour green. This process begins from the first pixel and iterates sequentially until the entire message is encoded.

5.2.5.2 DWT Image

The figure below clearly displays the distinctions between the original carrier image and the stego image using DWT, indicating where data is stored within the image.

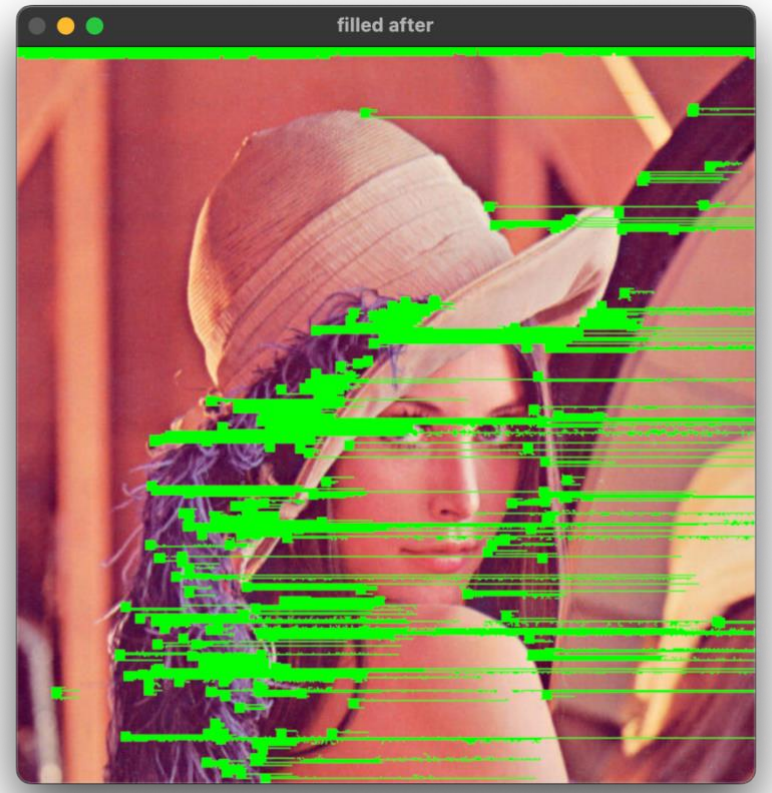


Figure 25 - SSIM for DWT

Figure 25 demonstrates that the DWT algorithm does not encode the secret message in a linear sequence from the first pixel of the image. Instead, it distributes changes across the image, implying increased alterations to the image and greater security for the encoded data. The modifications from the original image have been illustrated with a green highlight.

5.2.6 Online Decoders

Numerous websites on the internet offer tools for encoding and decoding data within multimedia files. To evaluate the security of the implemented algorithms, a few of these websites were utilised to attempt extraction of the encoded data. The first website consulted was “Aperi’Solve”, which divides the image into various colour coefficients to extract the encoded data.

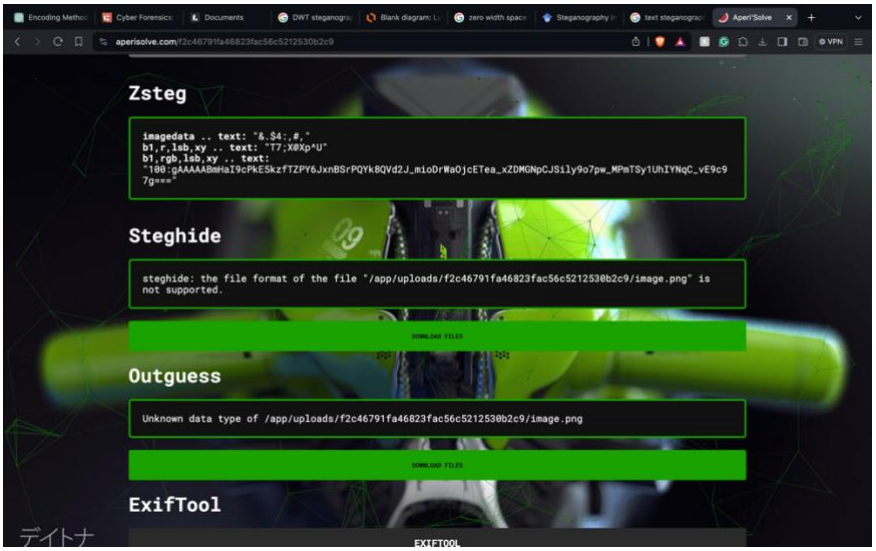


Figure 26 - Aperi'Solve for Image Encoded with LSB

Figure 26 demonstrates that the website successfully decoded the embedded data, visible under the "Zsteg" section, revealing the secret message. This indicates that a suitable online decoder can circumvent the LSB algorithm. However, while the message can be extracted through the website, encryption helps maintain its confidentiality, enhancing the overall security of the implemented algorithm.

The same website was used to test the advanced algorithm. The following figure shows the results of the analysis performed by the website.

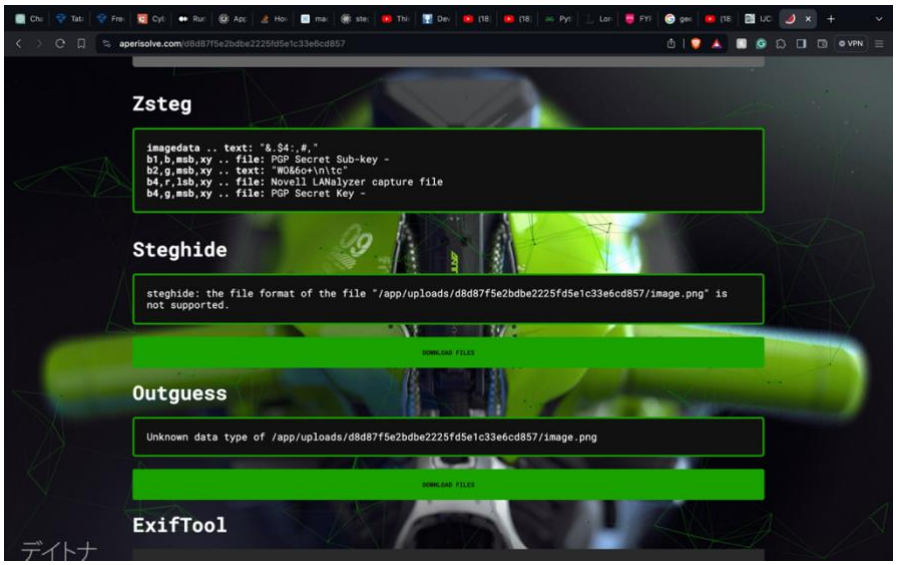


Figure 27 - Aperi'Solve for Image Encoded with DWT

As depicted in Figure 27, the website failed to retrieve the secret data stored within the stego image generated using the advanced algorithm. This indicates that the DWT algorithm offers higher security against online decoders.

The second website consulted is “stylesuxx”, which, while more superficial than the first website examined, still enables the encoding and decoding data within images.

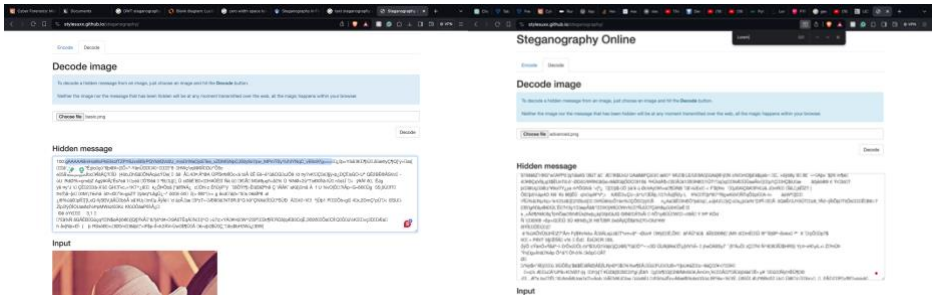


Figure 28 - Stylesuxx Decoding LSB and DWT Stego Images

In Figure 28, the website attempts to decode a stego image with embedded data using basic and advanced algorithms. The website successfully retrieves data from the image encoded with the basic algorithm, although the retrieved data is encrypted. However, images that use advanced algorithms offer higher security

and reduce the vulnerability to online decoders. This highlights the reason for implementing the DWT algorithm for sensitive data.

Examining the results from online decoders reveals potential risks tied to algorithm choice. While the LSB algorithm is vulnerable to decoding by online tools such as “Aperi’Solve”, the DWT algorithm shows higher resistance against such attacks.

5.2.7 HEX Editor

A HEX editor website, such as “HexEd.it”, was utilised to examine the header and footer of the stego files for signs of manipulation. Stego images are consistently stored in the PNG format by the application, with the header of the file represented by “89 50 4E 47” and the footer by "49 45 4E 44 AE 42 60 82" in HEX values. The most straightforward way to detect potential image manipulation is by inspecting the header and footer, as they always appear at the beginning and end of a file; any HEX values displayed before the header or after the footer indicate image manipulation, suggesting possible steganography.

5.2.7.1 LSB Encoded Image

Figure 29 displays the HEX values of the stego image encoded using the Least Significant Bit algorithm. The image reveals that the header and footer align with the necessary HEX values for PNG files, indicating that the encoded data is securely stored within the image and less vulnerable to forensic techniques employing HEX editors. This also implies that the encoded data resides within the file's body, enhancing the security of the hidden information. While this does not ensure complete security, it does increase the difficulty for unauthorised users to extract or detect the secret message.

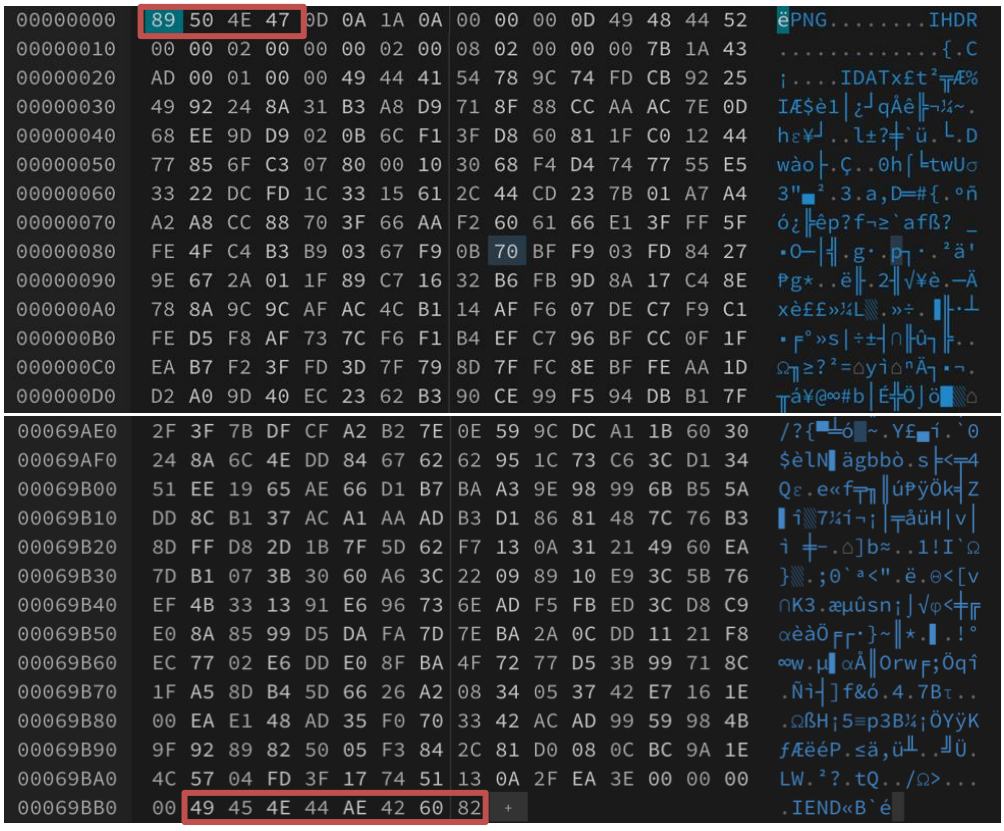


Figure 29 - Stego Image Encoded Using the LSB Algorithm

5.2.7.2 DWT Encoded Image

Figure 30 presents the stego image's HEX values encoded using the Discrete Wavelet Transform algorithm. The image indicates that the header and footer correspond to the necessary HEX values for PNG files, implying secure storage of the encoded data within the image and reduced vulnerability to detection through forensic methods using HEX editors. Furthermore, this suggests that the encoded data is embedded within the file's body, thereby enhancing the security of the hidden information. While this does not guarantee complete security, it does make it more challenging for an unauthorised user to extract or detect the secret message.

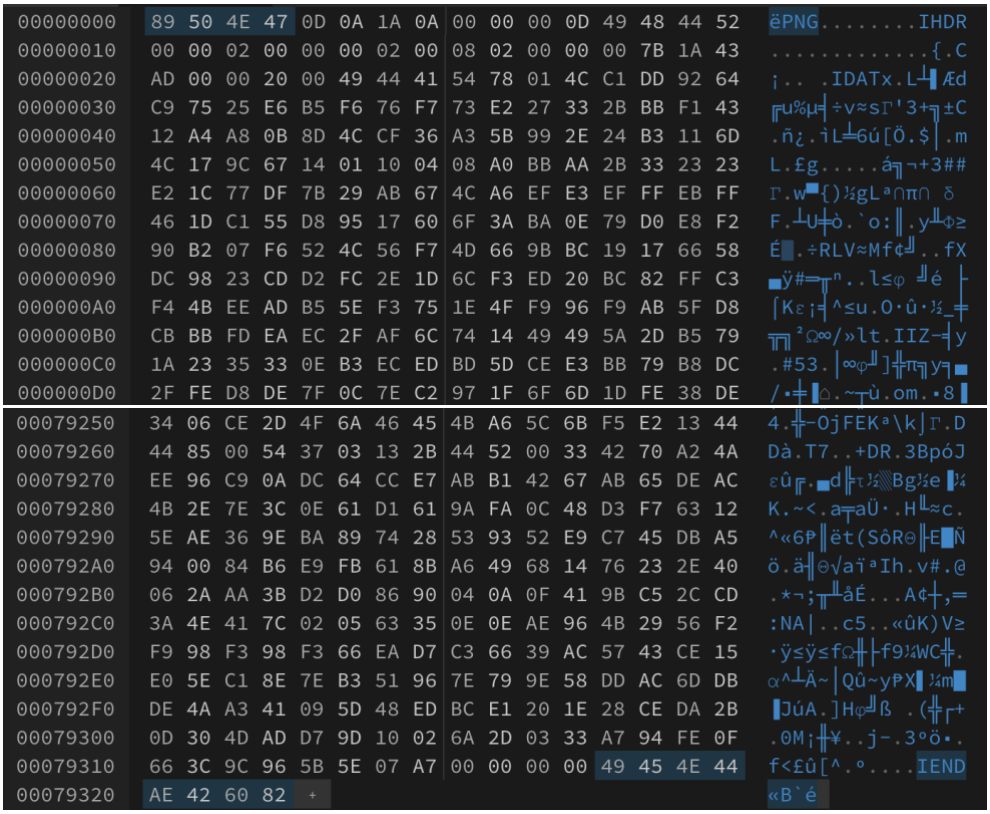


Figure 30 - Stego Image Encoded Using the DWT Algorithm

5.2.7.3 LSB Encoded Text File

The following image illustrates the HEX values of the original carrier text file and the stego file. The top represents the original carrier, while the bottom displays the stego file. Upon analysing and comparing the two HEX values, it becomes evident that alterations have occurred in the file. The header remains unchanged from the original file, but additional data has been stored in the locations where zero-width characters, such as spaces, were present. This suggests that the algorithm is vulnerable to techniques involving HEX editors. However, despite the method's ability to detect concealed information, the text retains its confidentiality due to the applied encryption.

Figure 31 - Comparison of HEX Values of Carrier and Stego File

5.2.8 Metadata and Byte-by-Byte Comparison

A thorough assessment has been conducted on the text steganography algorithm using LSB. This involves scanning metadata, such as file sizes and last modification timings. Furthermore, a detailed comparison between the two files was executed, examining the content byte-by-byte to check their similarity.

```
Metadata Scan:  False
Bytes Scan:  False
```

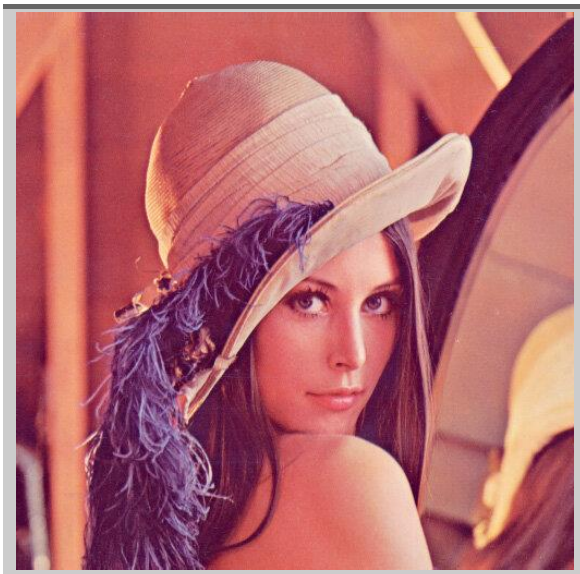
Figure 32 - Metadata and Byte-by-Byte Scan

As shown in Figure 32, the initial scan comparing metadata has yielded a "False" outcome, indicating that the original carrier and stego files differ in metadata, such as file size and last modification time. Additionally, the second scan, which examines each byte of the files, also reveals differences between them. This suggests that encoding data into the stego file has altered its content.

5.3 Comparative Analysis

The comparative analysis evaluates how well the steganography methods in this software perform by visually comparing the original cover files with the stego files created after embedding the secret message. This analysis helps us understand how effective the embedding process is and how it affects the visual integrity of the carrier file.

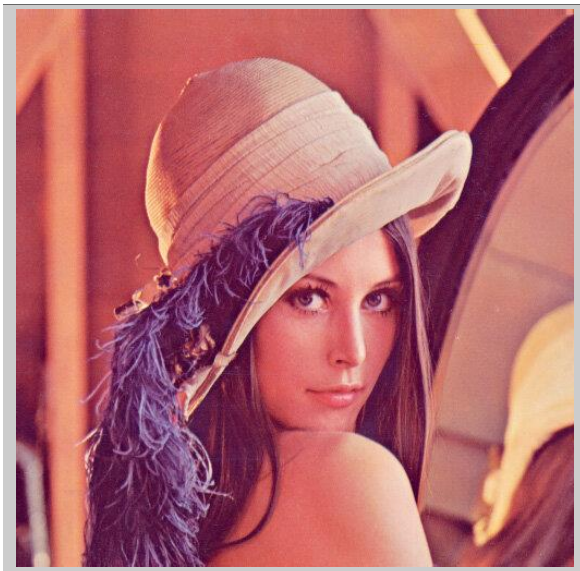
Original cover image

	File size: 78 KB

Stego image utilising LSB algorithm for encoding

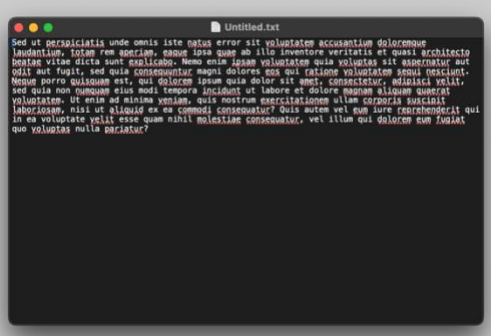
File size: 434 KB

Stego image utilising DWT algorithm for encoding

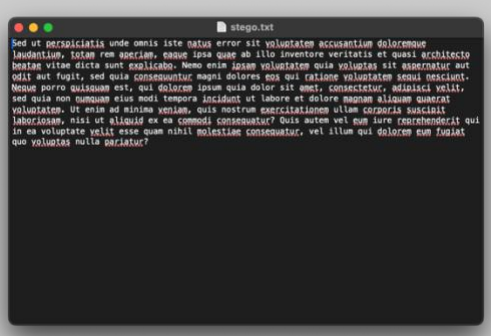


File size: 500 KB

Original cover text file

	File size: 865 bytes (4 KB on disk)

Stego file utilising LSB algorithm for encoding

	File size: 3,097 bytes (4KB on disk)

Visual comparison between the original image and LSB encoded stego image

Visual comparison between the original image and DWT-encoded stego image

5.3.1 Image Observations

Upon initial inspection, the original cover and stego images created using the LSB and DWT algorithms seem visually identical. The naked eye can detect no apparent differences in colour, texture, or pixel distribution between them. Additionally, there are no visible artefacts or distortions in the stego images that suggest the presence of hidden data.

5.3.2 Image Analysis

The visual comparison confirms that the steganography methods employed to hide data within the cover image maintain transparency and go unnoticed by human observers. The lack of noticeable differences between the original cover image and the stego images indicates that the embedding process preserves the visual quality and integrity of the cover medium. Despite no visible changes, a significant variation in file size was noticed. The original image size of 78 KB was expanded to 434 KB with the LSB algorithm and 500 KB with the DWT algorithm. This increase implies that both algorithms introduce extra data into the stego images during the hiding process. Despite the high level of transparency, which makes the stego images ideal for covert communications, the larger file size might prompt concerns regarding detectability, as the surge in file size could raise suspicion.

5.3.3 Text File Observations

Upon initial examination, the original text file and the stego text file created using the LSB algorithm seem visually indistinguishable, with no apparent alterations in characters or spacing that would indicate the presence of concealed data. Additionally, there are no symbols or unreadable characters within the stego file, indicating the algorithm's effectiveness for covert communications.

5.3.4 Text File Analysis

The visual comparison confirms that the steganography methods used to conceal data within the cover text file are undetectable to human observers. The lack of noticeable changes indicates the efficiency of the embedding process, although it remains vulnerable to potential data loss due to modifications to the stego file. While no noticeable differences in file content were observed, a significant increase in file size was noted. The original file size Is 865 bytes, increasing to 3,097 bytes after encoding data, suggesting the presence of concealed data within the file.

CHAPTER 6

CONCLUSIONS / FUTURE WORK

6.1 Conclusions

In conclusion, developing a steganography tool with algorithms to conceal data within various multimedia files has shown promising progress in advancing information security and covert communication. The software employs techniques like the least significant bit (LSB) and discrete wavelet transform (DWT) for image steganography and LSB for text steganography; encryption is also integrated into some algorithms to enhance data security. The detailed design and implementation of the software process can embed information within digital media while maintaining the integrity and confidentiality of the concealed data.

Numerous challenges were faced during the solution's development, ranging from algorithm complexities to performance optimisation. However, the obstacles were overcome through iterative improvements and thorough testing, creating a robust and reliable tool for text concealment.

A significant achievement of this project is the successful implementation of multiple encoding algorithms, particularly the DWT technique. When comparing the developed solution with others available online, none of the examined ones included algorithms beyond LSB. Leveraging this absence in other tools ensures imperceptibility and resilience against detection in the developed software.

While the project met its objectives, it is essential to recognise that the development does not stop here. Continuous improvements and innovations are vital to protect against emerging threats and evolving security requirements.

Therefore, future software releases will incorporate ongoing research and development efforts, exploring new algorithms and encryption techniques.

6.2 Future Work

Looking forward, numerous opportunities for future work could build upon this project's achievements. The developed application is a foundation for creating a comprehensive universal tool capable of encoding and utilising various algorithms across different multimedia formats. Firstly, expanding the software to support audio and video carrier files presents a significant opportunity for enhancing its versatility across diverse multimedia formats. Integrating audio and video while extending encryption to all algorithms (currently limited to a few) will enable secure communication and data concealment. A robust encryption mechanism tailored for each algorithm will enhance the software's overall security and protect against unauthorised access and detection.

Moreover, improving the capabilities of the text steganography algorithm is essential, as the current algorithm requires enhancements to reduce vulnerability to modifications and steganalysis techniques. Additionally, implementing an advanced text steganography algorithm, possibly utilising machine learning techniques, can further strengthen security and resilience against detection.

Integrating machine learning algorithms represents a revolutionary approach to enhancing implemented security measures. Utilising machine learning for embedding and detecting hidden information enables the software to adapt to newer steganalysis techniques continuously. This proactive strategy enhances the software's capability to withstand detection attempts and ensures the confidentiality of the concealed data.

Lastly, incorporating a password feature during the decoding process introduces an additional layer of security and user authentication, allowing users to set passwords for accessing encoded data, enhancing privacy and confidentiality. However, users would have a limited number of attempts, and the irreversible loss of encoded data after multiple incorrect password entries needs careful consideration to balance usability and security.

6.3 Potential Impact

The proposed method holds the potential to benefit a wide range of users, ranging from individuals to businesses, who are seeking effective ways to strengthen information security against unauthorised access. Users can easily preserve their privacy and confidentiality by offering a user-friendly solution for data concealment.

In the broader context of the security industry, the development of a steganography software contributes to the advancement of information security techniques with technology that is not widely implemented today. Implementing robust concealment techniques fortifies digital communication channels against eavesdropping and interception, strengthening security measures.

The software can serve as a secure means of communication for journalists, whistle-blowers, and human rights activists, allowing them to publicise sensitive information without fear of censorship or surveillance. Furthermore, it aids businesses and content creators protect intellectual property rights and proprietary information by embedding copyright or watermarking information within multimedia files. However, ethical considerations must be carefully addressed. While the software can empower individuals to protect their privacy and security, there is a risk of potential misuse for unlawful purposes. Therefore,

it is crucial to establish ethical standards and responsible use guidelines to mitigate risks and ensure accountability.

6.4 Legal, Social, Ethical and Professional Issues

6.4.1 Legal Issues

Table 6 - Legal Issues

Issue	Description
Data Protection Regulations	Using the software to hide sensitive personal information without consent could break data protection rules. Adhering to laws like the General Data Protection Regulations (GDPR) means being transparent and obtaining consent when dealing with personal data (GDPR, 2019).
Export Control Laws	The software's concealment of information and where it is developed and used could mean it falls under export control laws. Sending cryptographic software to specific countries without proper authorisation could have legal consequences (Dechert, 2016).
Liability and Accountability	If the software is involved in illegal activities or causes harm, liability and accountability issues may emerge. It is essential to clarify the extent of responsibility and include disclaimers in the terms of service to reduce potential legal consequences (Vardi, 2022).
Jurisdictional Variations	Countries have varying cryptography laws, data privacy, and digital communications. Before distributing globally, it is essential to research these differences in jurisdiction to

guarantee legal compliance.

6.4.2 Social Issues

Table 7 - Social Issues

Issue	Description
Trust and Reliability	Establishing trust in the software's reliability and security is vital for global acceptance. Users must feel assured that their data is well-protected and the software functions as intended without adding any vulnerabilities or risks.
Privacy Concerns	Steganography can safeguard privacy by hiding sensitive information, but it also raises worries about privacy invasion if misused. Users must understand and respect privacy rights and use the software responsibly.
Digital Literacy	Using steganography effectively demands a level of digital literacy and technical knowledge. Differences in digital literacy among different groups might impact who makes the most of or access the benefits of the application.
Technological Divide	Using steganography might widen the technology gap between those with and without access to technology, potentially affecting digital rights and cybersecurity in regions with limited technology or internet access.
Trust in Digital	Steganography can affect how trust is perceived in

Communications	digital communication. While it enhances privacy and confidentiality, widespread use could increase distrust if linked with secretive communication practices.

6.4.3 Ethical Issues

Table 8 - Ethical Issues

Issues	Description
Informed Consent	Before users begin using the software to hide data, they should give informed consent. For ethical use, users must comprehend the implications and potential consequences of using the software.
Dual-Use of the Technology	The steganography software has the potential for both beneficial and harmful applications. Distributing it raises ethical concerns, as it could be used maliciously. A set of terms and agreements should be prepared to inform users about their liabilities when using the software for such purposes.
Transparency	Maintaining ethical standards requires transparency in the development and operation of the software. This means describing the software's capabilities, limitations, and potential risks.

6.4.4 Professional Issues

Table 9 - Professional Issues

Issues	Description
Ethical Utilisation of Technology	The developer and users of the proposed solution are responsible for promoting honesty, integrity, and responsible use of technology. This includes evaluating how users' actions may affect the application's reputation and trustworthiness and upholding ethical standards in software development and deployment.
Accessibility and Inclusion	The steganography software aims to benefit society by prioritising accessibility and inclusivity. Ensuring it is accessible to individuals with diverse technical skills and compatible with various operating systems maximises its reach and usability. However, achieving accessibility in countries with limited technology access poses challenges despite the software's intended promotion of inclusivity.
Respect for Legal and Regulatory Compliance	Before deploying the steganography software, it must adhere to applicable laws, regulations, and industry standards regarding cryptography, data protection, and digital communication. This involves recognising the authority of law enforcement agencies and assisting efforts to combat illicit use.

6.5 Synoptic Reflections

Upon reflection on the project and its implications for my future career aspirations, especially in digital forensics, I am convinced of the relevance and applicability of the skills honed throughout this project. Digital forensics demands technical proficiency, analytical skills, and ethical integrity to navigate the complexities of investigating cyber incidents and digital evidence.

The development of the steganography tool has provided a strong foundation in essential aspects of digital forensics. Firstly, the project has increased my understanding of steganography and cryptography principles and techniques, which are crucial for safeguarding digital information. Exploring the intricacies of LSB and DWT embedding algorithms and encryption methods provided insights into manipulating and concealing data for covert communications. This knowledge is the foundation for conducting digital examinations and investigations involving steganographic artefacts.

Furthermore, developing the steganography software has enhanced my technical expertise in cryptography, image processing, pixel manipulation, digital signal processing, and software engineering. These skills apply to digital forensics, where the ability to analyse and interpret digital evidence is vital. Mastering techniques for detecting and extracting hidden information from digital media is a significant aspect of this skill set.

Moreover, involvement in this project enhanced critical thinking and problem-solving skills, which are essential in digital forensics. Designing, implementing, and refining the steganography software has sharpened my analytical abilities and equipped me with strategies for overcoming challenges and optimising performance. In the ever-changing landscape of digital forensics, the capacity to

adapt and respond to emerging threats and technologies is crucial, and this project has improved my capability to do so.

In the future, I see myself using the knowledge acquired from this project to pursue a career in digital forensics. My proficiency in steganography will allow me to make valuable contributions to the field.

REFERENCES

Ali, R.H., 2015. Steganography in Audio Using Wavelet and DES. Baghdad Science Journal, 12(2), pp.431-436.

AlSabhany, A.A., Ali, A.H., Ridzuan, F., Azni, A.H. and Mokhtar, M.R., 2020. Digital audio steganography: Systematic review, classification, and analysis of the current state of the art. Computer Science Review, 38, p.100316.

Aoki, N., 2008, August. A technique of lossless steganography for G. 711 telephony speech. In 2008 International Conference on Intelligent Information Hiding and Multimedia Signal Processing (pp. 608-611). IEEE.

Bachchas, K.S. (2023). Image steganography: Concealing secrets within pixels. [online] cybersecurity.att.com. Available at: https://cybersecurity.att.com/blogs/security-essentials/image-steganography-concealing-secrets-within-pixels.

Balagyozyan, L.A. and Hakobyan, R.G., 2021. Steganography in frames of graphical animation. In E3S web of conferences (Vol. 266, p. 09006). EDP Sciences.

Bender, W., Gruhl, D., Morimoto, N. and Lu, A., 1996. Techniques for data hiding. IBM systems journal, 35(3.4), pp.313-336.

Bilal, I. and Kumar, R., 2015. Audio steganography using QR decomposition and fast Fourier transform. Indian Journal of Science and Technology, 8(34), pp.1-7.

Burney, M.L. (2018). The History of Steganography and the Threat Posed to the United States and the Rest of the International Community. ProQuest Dissertations Publishing.

Choudary, A. (2019). Steganography tutorial | A complete guide for beginners. [online] Edureka. Available at: https://www.edureka.co/blog/steganography-tutorial [Accessed 24 Jan. 2024].

Chowdary, C. (2023). What Is Steganography? Types, Techniques, Applications. [online] intellipaat.com. Available at: https://intellipaat.com/blog/what-is-steganography/.

Coos, A. (2022). 5 Ways Big Companies Protect their Data. [online] Endpoint Protector Blog. Available at: https://www.endpointprotector.com/blog/5-ways-big-companies-protect-their-data/ [Accessed 16 Jan. 2024].

Dechert (2016). UK/EU Export Controls on Encryption Products. [online] www.dechert.com. Available at: https://www.dechert.com/knowledge/onpoint/2016/9/uk-eu-export-controls-on-encryption-products.html [Accessed 16 Apr. 2024].

Desai, H.V., 2012. Steganography, cryptography, watermarking: A comparitive study. Journal of Global Research in Computer Science, 3(12), pp.33-35.

Din, R., Utama, S. and Mustapha, A., 2018. Evaluation review on effectiveness and security performances of text steganography technique. Indonesian Journal of Electrical Engineering and Computer Science, 11(2), pp.747-754.

Djebbar, F., Ayad, B., Hamam, H. and Abed-Meraim, K., 2011, April. A view on latest audio steganography techniques. In 2011 International Conference on Innovations in Information Technology (pp. 409-414). IEEE.

Djebbar, F., Ayad, B., Meraim, K.A. and Hamam, H., 2012. Comparative study of digital audio steganography techniques. EURASIP Journal on Audio, Speech, and Music Processing, 2012(1), pp.1-16.

Evsutin, O., Melman, A. and Meshcheryakov, R. (2020). Digital steganography and watermarking for digital images: A review of current research directions. IEEE Access, 8, pp.166589-166611.

Fan, P., Zhang, H. and Zhao, X., 2022. Robust video steganography for social media sharing based on principal component analysis. EURASIP Journal on Information Security, 2022(1), p.4.

Figueira, J., 2022. A Survey on Semantic Steganography Systems. arXiv preprint arXiv:2203.12425.

Gardiner, J., 2012. Stegchat: a synonym-substitution based algorithm for text steganography. Submitted in conformity with the requirements for the degree of MSc Computer Security, School of Computer Science, University of Birmingham, pp.1-63.

GeeksforGeeks (2019). Difference between Steganography and Cryptography. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/difference-between-steganography-and-cryptography/.

GDPR (2019). What are the GDPR consent requirements? - GDPR.eu. [online] GDPR.eu. Available at: https://gdpr.eu/gdpr-consent-requirements/.

Ginni (2022). What are the application of Steganography? [online] www.tutorialspoint.com. Available at: https://www.tutorialspoint.com/what-are-the-application-of-steganography.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative adversarial nets. Advances in neural information processing systems, 27.

Gopalan, K. and Wenndt, S., 2004, July. Audio steganography for covert data transmission by imperceptible tone insertion. In Proc. The IASTED International Conference on Communication Systems And Applications (CSA 2004), Banff, Canada.

GR, M. and RB, S., 2021, May. Video steganography: a survey of techniques and methodologies. In Proceedings of the International Conference on Smart Data Intelligence (ICSMDI 2021).

Hemalatha, S., Acharya, U.D. and Renuka, A., 2015. Wavelet transform based steganography technique to hide audio signals in image. Procedia Computer Science, 47, pp.272-281.

Huang, J. and Shi, Y.Q., 2002. Reliable information bit hiding. IEEE Transactions on circuits and systems for video technology, 12(10), pp.916-920.

Information Commissioner's Officer (2023). Data storage, sharing and security. [online] ico.org.uk. Available at: https://ico.org.uk/for-organisations/advice-for-small-organisations/frequently-asked-questions/data-storage-sharing-and-security/#ifwere [Accessed 16 Jan. 2024].

Iwugo, D. (2023). What is Steganography? How to Hide Data Inside Data. [online] freeCodeCamp.org. Available at: https://www.freecodecamp.org/news/what-is-steganography-hide-data-inside-data/.

JetBrains (2024). PyCharm: the Python IDE for data science and web development. [online] JetBrains. Available at: https://www.jetbrains.com/pycharm/?var=1.

Jian, C.T., Wen, C.C., Rahman, N.H.B.A. and Hamid, I.R.B.A., 2017, August. Audio steganography with embedded text. In IOP conference series: materials science and engineering (Vol. 226, No. 1, p. 012084). IOP Publishing.

Kanhe, A., Aghila, G., Kiran, C.Y.S., Ramesh, C.H., Jadav, G. and Raj, M.G., 2015, August. Robust audio steganography based on advanced encryption standards in temporal domain. In 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (pp. 1449-1453). IEEE.

Kaspersky (2023). What is steganography? Definition and explanation. [online] www.kaspersky.com. Available at: https://www.kaspersky.com/resource-center/definitions/what-is-steganography.

Kaushik, N. (2012). Differences Between Watermarking and Steganography | Difference Between. [online] DifferenceBetween.net. Available at: http://www.differencebetween.net/business/product-services/differences-between-watermarking-and-steganography/.

Kawaguchi, E. (2023). Applications of Steganography. [online] datahide.org. Available at: https://datahide.org/BPCSe/applications-e.html.

Korgaonkar, V.V. and Gaonkar, M.N., 2017, May. A DWT-DCT combined approach for video steganography. In 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT) (pp. 421-424). IEEE.

Krishnan, R.B., Thandra, P.K. and Baba, M.S., 2017, March. An overview of text steganography. In 2017 Fourth International Conference on Signal Processing, Communication and Networking (ICSCN) (pp. 1-6). IEEE.

Kumar, R. and Singh, H., 2020. Recent trends in text steganography with experimental study. Handbook of Computer Networks and Cyber Security: Principles and Paradigms, pp.849-872.

Kunhoth, J., Subramanian, N., Al-Maadeed, S. and Bouridane, A., 2023. Video steganography: recent advances and challenges. Multimedia Tools and Applications, pp.1-43.

Majeed, M.A., Sulaiman, R., Shukur, Z. and Hasan, M.K. (2021). A review on text steganography techniques. Mathematics, 9(21), p.2829.

Malik, A. (2024). Educative Answers - Trusted Answers to Developer Questions. [online] Educative. Available at: https://www.educative.io/answers/what-is-the-difference-between-cryptography-and-steganography.

Mazurczyk, W., Wendzel, S., Zander, S., Houmansadr, A. and Szczypiorski, K., 2016. Information hiding in communication networks: fundamentals, mechanisms, applications, and countermeasures. John Wiley & Sons.

Mehta, M. and Shetty, A. (2013). Digital Watermarking and Steganography. International Journal of Engineering Research & Technology, [online] 2(10). Available at: https://www.ijert.org/digital-watermarking-and-steganography.

Meng, P., Ye, Y. and Hang, L., 2010, October. Steganography in Chinese text. In 2010 International Conference on Computer Application and System Modeling (ICCASM 2010) (Vol. 8, pp. V8-651). IEEE.

Mstafa, R.J. and Elleithy, K.M., 2017. Compressed and raw video steganography techniques: a comprehensive survey and analysis. Multimedia Tools and Applications, 76, pp.21749-21786.

OpenStego (2021). OpenStego. [online] www.openstego.com. Available at: https://www.openstego.com/.

Pan, F., Xiang, L., Yang, X.Y. and Guo, Y., 2010, July. Video steganography using motion vector and linear block codes. In 2010 IEEE International conference on software engineering and service sciences (pp. 592-595). IEEE.

Panigrahi, K.K. (2022). Difference between Steganography and Cryptography. [online] TutorialsPoint. Available at: https://www.tutorialspoint.com/difference-between-steganography-and-cryptography.

Patel, R., Lad, K. and Patel, M., 2021. Study and investigation of video steganography over uncompressed and compressed domain: a comprehensive review. Multimedia Systems, 27, pp.985-1024.

Qin, J., Luo, Y., Xiang, X., Tan, Y. and Huang, H., 2019. Coverless image steganography: a survey. IEEE access, 7, pp.171372-171394.

Rana, M. and Kunwar, F.B., 2016, March. A Temporal domain audio steganography technique using genetic algorithm. In 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom) (pp. 3141-3146). IEEE.

Reddy, R.P.K., Nagaraju, C. and Subramanyam, N., 2014. Text encryption through level based privacy using DNA steganography. International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), 3(3), pp.168-172.

Robertson, J. (2021). rsteg. [online] GitHub. Available at: https://github.com/xgi/rsteg.

Saha, S. (2023). Difference Between Steganography and Cryptography. [online] Scaler Topics. Available at: https://www.scaler.com/topics/difference-between-cryptography-and-steganography/.

Şahin, F., Çevik, T. and Takaoğlu, M. (2021). Review of the Literature on the Steganography Concept. International Journal of Computer Applications, 975, p.8887.

Sahu, A.K. and Sahu, M., 2020. Digital image steganography and steganalysis: A journey of the past three decades. Open Computer Science, 10(1), pp.296-342.

Shams N. Abdul-wahab, Mostafa Abdulghafoor Mohammed and Omar A. Hammood (2021) "Theoretical Background of steganography", Mesopotamian Journal of CyberSecurity, 2021, pp. 22–32. doi: 10.58496/MJCS/2021/005.

Shankdhar, P., 2020. Best tools to perform steganography [updated 2020] | Infosec [online]. resources.infosecinstitute.com. Available at: https://resources.infosecinstitute.com/topics/cryptography/steganography-and-tools-to-perform-steganography/.

Simplilearn (2023). What is Steganography? Types, Techniques, Examples & Applications | Simplilearn. [online] Simplilearn.com. Available at: https://www.simplilearn.com/what-is-steganography-article [Accessed 24 Jan. 2024].

Sobers, R. (2023). 84 Must-Know Data Breach Statistics [2023]. [online] www.varonis.com. Available at: https://www.varonis.com/blog/data-breach-statistics#:~:text=days%20(IBM). [Accessed 16 Jan. 2024].

Softpedia (2024a). Download SteganographX Plus 2.0. [online] softpedia. Available at: https://www.softpedia.com/get/Security/Encrypting/SteganographX.shtml.

Softpedia (2024b). Download SteganPEG 1.0. [online] softpedia. Available at: https://www.softpedia.com/get/Security/Encrypting/SteganPEG.shtml.

Softonic (2024). Xiao Steganography. [online] Softonic. Available at: https://xiao-steganography.en.softonic.com/#:~:text=A%20free%20Security%20program%20for%20Windows&text=It [Accessed 9 Apr. 2024].

Soni, T., 2020. Moving target network steganography. Rowan University.

Srivastav, H. (2024). Honey20/Camouflage. [online] GitHub. Available at: https://github.com/Honey20/Camouflage [Accessed 9 Apr. 2024].

SSuite Office (2018). SSuite Picsel Steganography Encryption. [online] SSuite Office Software. Available at: https://www.ssuiteoffice.com/software/ssuitepicselsecurity.htm.

Steghide (2003). Steghide. [online] steghide.sourceforge.net. Available at: https://steghide.sourceforge.net/.

Subramanian, N., Elharrouss, O., Al-Maadeed, S. and Bouridane, A., 2021. Image steganography: A review of the recent advances. IEEE access, 9, pp.23409-23423.

Swain, G., 2019. Very high capacity image steganography technique using quotient value differencing and LSB substitution. Arabian Journal for Science and Engineering, 44(4), pp.2995-3004.

Swamy, A. (2023). ashwek/Hide_n_Seek. [online] GitHub. Available at: https://github.com/ashwek/Hide_n_Seek [Accessed 9 Apr. 2024].

Tidmarsh, D. (2023). A Guide to Steganography: Meaning, Types, Tools, & Techniques. [online] Cybersecurity Exchange. Available at: https://www.eccouncil.org/cybersecurity-exchange/ethical-hacking/what-is-steganography-guide-meaning-types-tools/ [Accessed 24 Jan. 2024].

Thenmozhi, S., 2022. CNN based Image Steganography Techniques: A Cutting Edge/State of Art Review. 2022 Smart Technologies, Communication and Robotics (STCR), pp.1-5.

Vaishakh, K., Pravalika, A., Abhishek, D.V., Meghana, N.P. and Prasad, G., 2019. A semantic approach to text steganography in sanskrit using numerical encoding. In Recent Findings in Intelligent Computing Techniques: Proceedings of the 5th ICACNI 2017, Volume 1 (pp. 181-192). Springer Singapore.

Vardi, M. (2022). Accountability and Liability in Computing – Communications of the ACM. [online] Communications of the ACM. Available at: https://cacm.acm.org/opinion/accountability-and-liability-in-computing/ [Accessed 16 Apr. 2024].

Wikipedia (2022). OpenPuff. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/OpenPuff.

Wrike (2023). What Is Scrum in Agile? [online] www.wrike.com. Available at: https://www.wrike.com/project-management-guide/faq/what-is-scrum-in-agile/.

Yadav, V.K. and Batham, S., 2015. A novel approach of bulk data hiding using text steganography. Procedia Computer Science, 57, pp.1401-1410.

Zamani, M., Manaf, A.B.A., Ahmad, R.B., Jaryani, F., Taherdoost, H. and Zeki, A.M., 2009, November. A secure audio steganography approach. In 2009 International Conference for Internet Technology and Secured Transactions,(ICITST) (pp. 1-6). IEEE.

Zhang, L., Wang, S., Gan, W., Tang, C., Zhang, J. and Liang, H., 2018. SLIDE: An efficient secure linguistic steganography detection protocol. In Cloud Computing and Security: 4th International Conference, ICCCS 2018, Haikou,

China, June 8–10, 2018, Revised Selected Papers, Part III 4 (pp. 298-309). Springer International Publishing.

Zhang, R., Sachnev, V., Botnan, M.B., Kim, H.J. and Heo, J., 2012. An efficient embedder for BCH coding for steganography. IEEE Transactions on Information Theory, 58(12), pp.7272-7279.

BIBLIOGRAPHY

Ali, R.H., 2015. Steganography in Audio Using Wavelet and DES. Baghdad Science Journal, 12(2), pp.431-436.

AlSabhany, A.A., Ali, A.H., Ridzuan, F., Azni, A.H. and Mokhtar, M.R., 2020. Digital audio steganography: Systematic review, classification, and analysis of the current state of the art. Computer Science Review, 38, p.100316.

Aoki, N., 2008, August. A technique of lossless steganography for G. 711 telephony speech. In 2008 International Conference on Intelligent Information Hiding and Multimedia Signal Processing (pp. 608-611). IEEE.

Bachchas, K.S. (2023). Image steganography: Concealing secrets within pixels. [online] cybersecurity.att.com. Available at: https://cybersecurity.att.com/blogs/security-essentials/image-steganography-concealing-secrets-within-pixels.

Balagyozyan, L.A. and Hakobyan, R.G., 2021. Steganography in frames of graphical animation. In E3S web of conferences (Vol. 266, p. 09006). EDP Sciences.

Bender, W., Gruhl, D., Morimoto, N. and Lu, A., 1996. Techniques for data hiding. IBM systems journal, 35(3.4), pp.313-336.

Bilal, I. and Kumar, R., 2015. Audio steganography using QR decomposition and fast Fourier transform. Indian Journal of Science and Technology, 8(34), pp.1-7.

Burney, M.L. (2018). The History of Steganography and the Threat Posed to the United States and the Rest of the International Community. ProQuest Dissertations Publishing.

Choudary, A. (2019). Steganography tutorial | A complete guide for beginners. [online] Edureka. Available at: https://www.edureka.co/blog/steganography-tutorial [Accessed 24 Jan. 2024].

Chowdary, C. (2023). What Is Steganography? Types, Techniques, Applications. [online] intellipaat.com. Available at: https://intellipaat.com/blog/what-is-steganography/.

Connolly, C., Lincoln, P., Mason, I. and Yegneswaran, V. (2014). {TRIST}: Circumventing Censorship with {Transcoding-Resistant} Image Steganography. [online] www.usenix.org. Available at: https://www.usenix.org/conference/foci14/workshop-program/presentation/connolly.

Coos, A. (2022). 5 Ways Big Companies Protect their Data. [online] Endpoint Protector Blog. Available at: https://www.endpointprotector.com/blog/5-ways-big-companies-protect-their-data/ [Accessed 16 Jan. 2024].

Dalal, M. and Juneja, M., 2016. Overview of video steganography in compressed domain. Int J Comput Technol Appl, 9(10), pp.1-11.

Dechert (2016). UK/EU Export Controls on Encryption Products. [online] www.dechert.com. Available at: https://www.dechert.com/knowledge/onpoint/2016/9/uk-eu-export-controls-on-encryption-products.html [Accessed 16 Apr. 2024].

Desai, H.V., 2012. Steganography, cryptography, watermarking: A comparitive study. Journal of Global Research in Computer Science, 3(12), pp.33-35.

Din, R., Utama, S. and Mustapha, A., 2018. Evaluation review on effectiveness and security performances of text steganography technique. Indonesian Journal of Electrical Engineering and Computer Science, 11(2), pp.747-754.

Djebbar, F., Ayad, B., Hamam, H. and Abed-Meraim, K., 2011, April. A view on latest audio steganography techniques. In 2011 International Conference on Innovations in Information Technology (pp. 409-414). IEEE.

Djebbar, F., Ayad, B., Meraim, K.A. and Hamam, H., 2012. Comparative study of digital audio steganography techniques. EURASIP Journal on Audio, Speech, and Music Processing, 2012(1), pp.1-16.

Evsutin, O., Melman, A. and Meshcheryakov, R. (2020). Digital steganography and watermarking for digital images: A review of current research directions. IEEE Access, 8, pp.166589-166611.

Fan, P., Zhang, H. and Zhao, X., 2022. Robust video steganography for social media sharing based on principal component analysis. EURASIP Journal on Information Security, 2022(1), p.4.

Figueira, J., 2022. A Survey on Semantic Steganography Systems. arXiv preprint arXiv:2203.12425.

Gardiner, J., 2012. Stegchat: a synonym-substitution based algorithm for text steganography. Submitted in conformity with the requirements for the degree of MSc Computer Security, School of Computer Science, University of Birmingham, pp.1-63.

GeeksforGeeks (2019). Difference between Steganography and Cryptography. [online] GeeksforGeeks. Available at: https://www.geeksforgeeks.org/difference-between-steganography-and-cryptography/.

GDPR (2019). What are the GDPR consent requirements? - GDPR.eu. [online] GDPR.eu. Available at: https://gdpr.eu/gdpr-consent-requirements/.

Ginni (2022). What are the application of Steganography? [online] www.tutorialspoint.com. Available at: https://www.tutorialspoint.com/what-are-the-application-of-steganography.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative adversarial nets. Advances in neural information processing systems, 27.

Gopalan, K. and Wenndt, S., 2004, July. Audio steganography for covert data transmission by imperceptible tone insertion. In Proc. The IASTED International Conference on Communication Systems And Applications (CSA 2004), Banff, Canada.

GR, M. and RB, S., 2021, May. Video steganography: a survey of techniques and methodologies. In Proceedings of the International Conference on Smart Data Intelligence (ICSMDI 2021).

Hemalatha, S., Acharya, U.D. and Renuka, A., 2015. Wavelet transform based steganography technique to hide audio signals in image. Procedia Computer Science, 47, pp.272-281.

Huang, J. and Shi, Y.Q., 2002. Reliable information bit hiding. IEEE Transactions on circuits and systems for video technology, 12(10), pp.916-920.

Information Commissioner's Officer (2023). Data storage, sharing and security. [online] ico.org.uk. Available at: https://ico.org.uk/for-organisations/advice-for-small-organisations/frequently-asked-questions/data-storage-sharing-and-security/#ifwere [Accessed 16 Jan. 2024].

Iwugo, D. (2023). What is Steganography? How to Hide Data Inside Data. [online] freeCodeCamp.org. Available at:

https://www.freecodecamp.org/news/what-is-steganography-hide-data-inside-data/.

JetBrains (2024). PyCharm: the Python IDE for data science and web development. [online] JetBrains. Available at: https://www.jetbrains.com/pycharm/?var=1.

Jian, C.T., Wen, C.C., Rahman, N.H.B.A. and Hamid, I.R.B.A., 2017, August. Audio steganography with embedded text. In IOP conference series: materials science and engineering (Vol. 226, No. 1, p. 012084). IOP Publishing.

Kanhe, A., Aghila, G., Kiran, C.Y.S., Ramesh, C.H., Jadav, G. and Raj, M.G., 2015, August. Robust audio steganography based on advanced encryption standards in temporal domain. In 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (pp. 1449-1453). IEEE.

Kao, D.Y. (2016). A Fight for Copyright Infringement: Uncovering Covert Identifiable Data behind Computer Software. Intellectual Property Rights: Open Access, 4(2). Available at: https://doi.org/10.4172/2375-4516.1000167.

Kaspersky (2023). What is steganography? Definition and explanation. [online] www.kaspersky.com. Available at: https://www.kaspersky.com/resource-center/definitions/what-is-steganography.

Kaushik, N. (2012). Differences Between Watermarking and Steganography | Difference Between. [online] DifferenceBetween.net. Available at: http://www.differencebetween.net/business/product-services/differences-between-watermarking-and-steganography/.

Kawaguchi, E. (2023). Applications of Steganography. [online] datahide.org. Available at: https://datahide.org/BPCSe/applications-e.html.

Korgaonkar, V.V. and Gaonkar, M.N., 2017, May. A DWT-DCT combined approach for video steganography. In 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT) (pp. 421-424). IEEE.

Krishnan, R.B., Thandra, P.K. and Baba, M.S., 2017, March. An overview of text steganography. In 2017 Fourth International Conference on Signal Processing, Communication and Networking (ICSCN) (pp. 1-6). IEEE.

Kumar, R. and Singh, H., 2020. Recent trends in text steganography with experimental study. Handbook of Computer Networks and Cyber Security: Principles and Paradigms, pp.849-872.

Kunhoth, J., Subramanian, N., Al-Maadeed, S. and Bouridane, A., 2023. Video steganography: recent advances and challenges. Multimedia Tools and Applications, pp.1-43.

Lubacz, J., Mazurczyk, W. and Szczypiorski, K., 2014. Principles and overview of network steganography. IEEE Communications Magazine, 52(5), pp.225-229.

Majeed, M.A., Sulaiman, R., Shukur, Z. and Hasan, M.K. (2021). A review on text steganography techniques. Mathematics, 9(21), p.2829.

Malik, A. (2024). Educative Answers - Trusted Answers to Developer Questions. [online] Educative. Available at: https://www.educative.io/answers/what-is-the-difference-between-cryptography-and-steganography.

Mandal, P.C., Mukherjee, I., Paul, G. and Chatterji, B.N., 2022. Digital image steganography: A literature survey. Information sciences.

Mazurczyk, W., Wendzel, S., Zander, S., Houmansadr, A. and Szczypiorski, K., 2016. Information hiding in communication networks: fundamentals, mechanisms, applications, and countermeasures. John Wiley & Sons.

Mehta, M. and Shetty, A. (2013). Digital Watermarking and Steganography. International Journal of Engineering Research & Technology, [online] 2(10). Available at: https://www.ijert.org/digital-watermarking-and-steganography.

Meng, P., Ye, Y. and Hang, L., 2010, October. Steganography in Chinese text. In 2010 International Conference on Computer Application and System Modeling (ICCASM 2010) (Vol. 8, pp. V8-651). IEEE.

Mstafa, R.J. and Elleithy, K.M., 2017. Compressed and raw video steganography techniques: a comprehensive survey and analysis. Multimedia Tools and Applications, 76, pp.21749-21786.

OpenStego (2021). OpenStego. [online] www.openstego.com. Available at: https://www.openstego.com/.

Pan, F., Xiang, L., Yang, X.Y. and Guo, Y., 2010, July. Video steganography using motion vector and linear block codes. In 2010 IEEE International conference on software engineering and service sciences (pp. 592-595). IEEE.

Panigrahi, K.K. (2022). Difference between Steganography and Cryptography. [online] TutorialsPoint. Available at: https://www.tutorialspoint.com/difference-between-steganography-and-cryptography.

Patel, R., Lad, K. and Patel, M., 2021. Study and investigation of video steganography over uncompressed and compressed domain: a comprehensive review. Multimedia Systems, 27, pp.985-1024.

Qin, J., Luo, Y., Xiang, X., Tan, Y. and Huang, H., 2019. Coverless image steganography: a survey. IEEE access, 7, pp.171372-171394.

Rana, M. and Kunwar, F.B., 2016, March. A Temporal domain audio steganography technique using genetic algorithm. In 2016 3rd International

Conference on Computing for Sustainable Global Development (INDIACom) (pp. 3141-3146). IEEE.

Reddy, R.P.K., Nagaraju, C. and Subramanyam, N., 2014. Text encryption through level based privacy using DNA steganography. International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), 3(3), pp.168-172.

Ridgway, J. and Stannett, M., 2014. Developing a Video Steganography Toolkit. arXiv preprint arXiv:1409.4883.

Robertson, J. (2021). rsteg. [online] GitHub. Available at: https://github.com/xgi/rsteg.

Saha, S. (2023). Difference Between Steganography and Cryptography. [online] Scaler Topics. Available at: https://www.scaler.com/topics/difference-between-cryptography-and-steganography/.

Şahin, F., Çevik, T. and Takaoğlu, M. (2021). Review of the Literature on the Steganography Concept. International Journal of Computer Applications, 975, p.8887.

Sahu, A.K. and Sahu, M., 2020. Digital image steganography and steganalysis: A journey of the past three decades. Open Computer Science, 10(1), pp.296-342.

Shams N. Abdul-wahab, Mostafa Abdulghafoor Mohammed and Omar A. Hammood (2021) "Theoretical Background of steganography", Mesopotamian Journal of CyberSecurity, 2021, pp. 22–32. doi: 10.58496/MJCS/2021/005.

Shankdhar, P., 2020. Best tools to perform steganography [updated 2020] | Infosec [online]. resources.infosecinstitute.com. Available at:

https://resources.infosecinstitute.com/topics/cryptography/steganography-and-tools-to-perform-steganography/.

Simplilearn (2023). What is Steganography? Types, Techniques, Examples & Applications | Simplilearn. [online] Simplilearn.com. Available at: https://www.simplilearn.com/what-is-steganography-article [Accessed 24 Jan. 2024].

Smith, C.B., 2019. The Comparison of Steganography and Cryptography: Concealing Information (Doctoral dissertation, Utica College).

Sobers, R. (2023). 84 Must-Know Data Breach Statistics [2023]. [online] www.varonis.com. Available at: https://www.varonis.com/blog/data-breach-statistics#:~:text=days%20(IBM). [Accessed 16 Jan. 2024].

Softpedia (2024a). Download SteganographX Plus 2.0. [online] softpedia. Available at: https://www.softpedia.com/get/Security/Encrypting/SteganographX.shtml.

Softpedia (2024b). Download SteganPEG 1.0. [online] softpedia. Available at: https://www.softpedia.com/get/Security/Encrypting/SteganPEG.shtml.

Softonic (2024). Xiao Steganography. [online] Softonic. Available at: https://xiao-steganography.en.softonic.com/#:~:text=A%20free%20Security%20program%20for%20Windows&text=It [Accessed 9 Apr. 2024].

Soni, T., 2020. Moving target network steganography. Rowan University.

Srivastav, H. (2024). Honey20/Camouflage. [online] GitHub. Available at: https://github.com/Honey20/Camouflage [Accessed 9 Apr. 2024].

SSuite Office (2018). SSuite Picsel Steganography Encryption. [online] SSuite Office Software. Available at: https://www.ssuiteoffice.com/software/ssuitepicselsecurity.htm.

Steghide (2003). Steghide. [online] steghide.sourceforge.net. Available at: https://steghide.sourceforge.net/.

Subramanian, N., Elharrouss, O., Al-Maadeed, S. and Bouridane, A., 2021. Image steganography: A review of the recent advances. IEEE access, 9, pp.23409-23423.

Swain, G., 2019. Very high capacity image steganography technique using quotient value differencing and LSB substitution. Arabian Journal for Science and Engineering, 44(4), pp.2995-3004.

Swamy, A. (2023). ashwek/Hide_n_Seek. [online] GitHub. Available at: https://github.com/ashwek/Hide_n_Seek [Accessed 9 Apr. 2024].

Tidmarsh, D. (2023). A Guide to Steganography: Meaning, Types, Tools, & Techniques. [online] Cybersecurity Exchange. Available at: https://www.eccouncil.org/cybersecurity-exchange/ethical-hacking/what-is-steganography-guide-meaning-types-tools/ [Accessed 24 Jan. 2024].

Thenmozhi, S., 2022. CNN based Image Steganography Techniques: A Cutting Edge/State of Art Review. 2022 Smart Technologies, Communication and Robotics (STCR), pp.1-5.

UKEssays. November 2018. Legal social ethical and professional issues. [online]. Available from: https://www.ukessays.com/essays/information-technology/legal-social-ethical-and-professional-issues-information-technology-essay.php?vref=1.

UK Parliament (2021). Defamation Act - Parliamentary Bills - UK Parliament. [online] bills.parliament.uk. Available at: https://bills.parliament.uk/bills/983.

Vaishakh, K., Pravalika, A., Abhishek, D.V., Meghana, N.P. and Prasad, G., 2019. A semantic approach to text steganography in sanskrit using numerical encoding. In Recent Findings in Intelligent Computing Techniques: Proceedings of the 5th ICACNI 2017, Volume 1 (pp. 181-192). Springer Singapore.

Vardi, M. (2022). Accountability and Liability in Computing – Communications of the ACM. [online] Communications of the ACM. Available at: https://cacm.acm.org/opinion/accountability-and-liability-in-computing/ [Accessed 16 Apr. 2024].

Wikipedia (2022). OpenPuff. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/OpenPuff.

Wrike (2023). What Is Scrum in Agile? [online] www.wrike.com. Available at: https://www.wrike.com/project-management-guide/faq/what-is-scrum-in-agile/.

Yadav, V.K. and Batham, S., 2015. A novel approach of bulk data hiding using text steganography. Procedia Computer Science, 57, pp.1401-1410.

Zamani, M., Manaf, A.B.A., Ahmad, R.B., Jaryani, F., Taherdoost, H. and Zeki, A.M., 2009, November. A secure audio steganography approach. In 2009 International Conference for Internet Technology and Secured Transactions,(ICITST) (pp. 1-6). IEEE.

Zhang, L., Wang, S., Gan, W., Tang, C., Zhang, J. and Liang, H., 2018. SLIDE: An efficient secure linguistic steganography detection protocol. In Cloud Computing and Security: 4th International Conference, ICCCS 2018, Haikou, China, June 8–10, 2018, Revised Selected Papers, Part III 4 (pp. 298-309). Springer International Publishing.

Zhang, R., Sachnev, V., Botnan, M.B., Kim, H.J. and Heo, J., 2012. An efficient embedder for BCH coding for steganography. IEEE Transactions on Information Theory, 58(12), pp.7272-7279.