SUDAN UNIVERSITY OF SCIENCE AND TECHNOLOGY

COLLEGE OF GRADUATE STUDIES

تطوير نظام متعدد الطبقات لإخفاء المعلومات مدعوم بضغط البيانات

Development of a Compression Aided Multi-level Steganography

A Thesis Submitted in Partial Fulfillment of the Requirements of Master Degree in Computer Science

By:

Mohamed Yousif Abdalla Elmahi

Supervisor:

Dr. Talaat Wahby

August 2014

الآيه

بِسۡمِ ٱللَّهِ ٱلرَّحۡمَٰنِ ٱلرَّحِيمِ ١

ٱلۡحَمۡدُ لِلَّهِ رَبِّ ٱلۡعَٰلَمِينَ ٢

ٱلرَّحۡمَٰنِ ٱلرَّحِيمِ ٣ مَٰلِكِ يَوۡمِ ٱلدِّينِ ٤

إِيَّاكَ نَعۡبُدُ وَإِيَّاكَ نَسۡتَعِينُ ٥ ٱهۡدِنَا

ٱلصِّرَٰطَ ٱلۡمُسۡتَقِيمَ ٦ صِرَٰطَ ٱلَّذِينَ أَنۡعَمۡتَ

عَلَيۡهِمۡ غَيۡرِ ٱلۡمَغۡضُوبِ عَلَيۡهِمۡ

وَلَا ٱلضَّآلِّينَ ٧

صدق الله العظيم

DEDICATION

To my Mother, for teaching me how to believe in myself, keep my dream alive and follow my passion.

To my Father, for your support.

Dearest Mazin, you are the best.

To my Supervisor.

AKNOWLEDAGEMENT

I would like to thank my family for the great support that you gave me.

"A teachers affects eternity; they can never tell where his influence stops." Thanks to all my teachers especially my supervisor for teaching me how to be accurate in all the things that I do.

Thank to all my friends especially (Mutwkil Faisal, Suhaip Ali, Afaf Yousif, Lobaba Elteyeb, and Hind Abdalla) for the unconditional support along the way.

TABLE OF CONTENTS

TABLE OF FIGURES:

LIST OF TABLES

ABSTRACT

A lot of techniques are used to protect the information such as information hiding. Steganography is one of the information hiding Techniques that hide a message inside another message without drawing any suspicion.

In this research, there is a combination of both text steganography (level one in our model) and image steganography (level two in our model). Level one uses a new approach that is based on the Pseudo Random Number Generation (PRNG) to embed the secret message into Randomly Cover-text generated. Level Two uses a cover image to embed the message by using the LSB insertion technique that produces the Stego-image. Between these levels the compression process (Huffman Compression Algorithm) receives the output of level one (Stego-Text) and compresses it to reduce the size and then passes it as an input to level two. Two secret keys (Hiding Key, & Extraction Key) for authentication at both ends in order to achieve high level of security. At the receiver side different reverse operations have been carried out to get back the original information.

The analysis shows that the model has a good measure of hiding the secret message, and good image quality that carried the information.

المستخلص

توجد مجموعة من التقنيات لحمايه المعلومات مثل إخفاء المعلومات. Steganography هي واحده من تقنيات إخفاء المعلومات حيث تقوم بإخفاء المعلومات أو الرساله في داخل رساله أخري بدون لفت الانتباه لذلك.

يقوم هذا البحث علي اخفاء المعلومات السريه في عدة طبقات ، الطبقه الاولي هي عباره طريقه جديده تعمل علي اخفاء ومزج الرساله السريه في نص أخر مولد عشوائيا بإستخدام (PRNG) Pseudo Random Number Generation،وفي الطبقه التانيه يقوم النظام بالإخفاء في الصور علي طريقة (LSB Insertion Techniques) ، وما بين تلك الطبقتين يقوم النظام بضغط (أوتصغير حجم) بإستخدام خوارزميه (Huffman Compression Algorithm) لمخرجات الطبقه الاولي لتقليل حجم الملف ثم تمريره ليكون كمدخلات للطبقه التانيه ، حيث يتم أرسال الصوره للطرف الأخر ليقوم بعكس العمليات لإستخراج الرساله الاصليه مره أخري .

وقد أظهر التحليل للنموذج مقاييس جيده من حيث إخفاء الرساله السريه ، ومقاييس جيده ايضاً من حيث جوده الصوره الناتجه منه.

CHAPTER 1

INTRODUCTION

1.1 Introduction

Information Security is the practice of defending information from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction. It is a general term that can be used regardless of the form the data may take [1]. And because the information it is very valuable to the companies it should be strong protected and /or hidden by variety of technologies and techniques to ensure that it is not been reach to any unauthorized people.

Information hiding is one of the powerful techniques used in the information security. One of the chief mechanisms for hiding information is encapsulation or combining elements to create a larger entity. Any unauthorized user can then focus on the new object without knowing about the hidden details or the real message. Information hiding is an emerging research area, which encompasses applications such as copyright protection for digital media, watermarking, fingerprinting, and steganography. All these applications of information hiding are quite diverse [2].

In watermarking applications, the message contains information such as owner identification and a digital time stamp, which usually applied for copyright protection.

Fingerprint, the owner of the data set embeds a serial number that uniquely identifies the user of the data set. This adds to copyright information to makes it possible to trace any unauthorized use of the data set back to the user.

Steganography hides the secret message within the host data set and presence imperceptible.

Steganography is the art and science of communicating in a way which hides the existence of the communication such as hiding a message inside another message without drawing any suspicion to others so that the message can only be detected by its intended recipient [3].This approach of information hiding technique has recently become important in a number of application areas [4]. The word steganography comes from two

roots in the Greek language, “Stegos” meaning hidden/covered/or roof, and “Graphia” simply meaning writing [5]. The history of steganography can be traced back to around 440 B.C., where the Greek historian Herodotus described in his writing about two events : one used wax to cover secret messages, and the other used shaved heads[6].

Steganography can be classified into image, text, audio and video steganography depending on the cover media that used to embed secret data (as in figure 1).

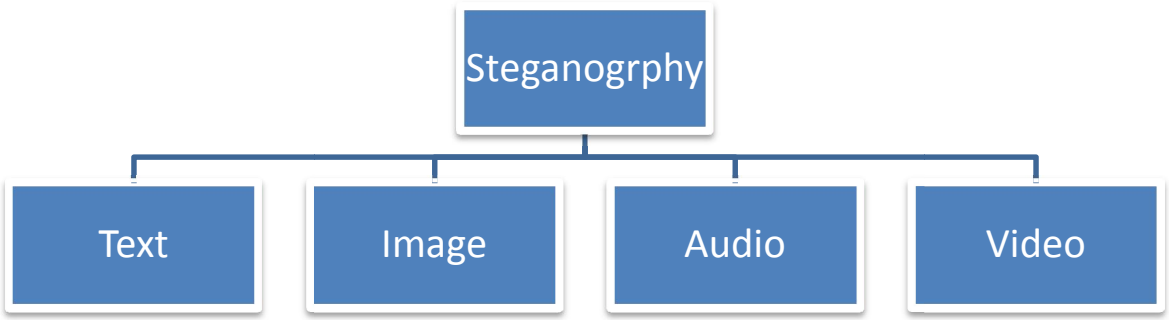


Figure 1.1: Steganography Types depends on the cover media.

1.4 Research Scope

The main area of this research will be steganography, hiding information in texts as well as images techniques, although the focus will be on the compression types and methods to be applied between the text & image of this model.

1.3 Problem Statement

Encryption draws the attention of others to attack the encryption algorithm, to eradicate that attention we use Steganography. Although text steganography is easy to detect and had very simple and weak approaches to conceal data. In addition to that the size of the massage that been hidden (in text) is very simple compared with the cover media .Also text steganography had many constraints such as the languages, grammars and others.

Hence, an enhancement is required to address those issues. This thesis proposes a multi-level steganography system to deal with these problems.

1.4 Objective of the Research

We propose to apply multi level steganography in texts and after that in image along with compression algorithm to maximize the data carried by the cover media. The objective of that will be:

- Enhancing the security of the sensitive data by using multi-level steganography in one system.
- Proposing two lines of defense by hiding the message in two cover media, and enhance the message size by compression.
- Developing one system for better confidentiality and security.

1.5 Research Methodology

By applying deep study to both text and image steganography related work to develop a system that can hold the data and hide (or embedded) it first in texts, in

addition the compression of both the original and the cover text done after the first step. The output of those operations will be used as input to the second hiding operation in the image. The extraction operations work as an opposite way to embedding operations.

1.6 Research Questions

- How can sensitive information be hidden in two covers media & how the original information can be extracted?
- Is Multilevel Steganography enhancing the security?
- Can the proposed model conceal the data in text without being noticed by unauthorized users, furthermore could the human visual system (HVS) noticed the change in the images after embedding the message?

1.7 Research Organization

Chapter one is an introduction which highlight a brief history of steganography, chapter two is the Literature Review beside the related work, chapter three Proposed Model of the multi-level steganography, chapter four will introduce the Result of the research outputs, The last chapter gives Conclusion & Future Work.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

Variety of study done in the information hiding produces huge number of techniques and methods to secure the valuable information from any unauthorized users, and although the covers media (text, image, audio, video) gave the steganography great attention of the researchers and scientists and it become a vast area of research, and a lot of researches has been made depend on the cover media. Even if we look at just the researches in text steganography as example, we will find that different methods or approaches and algorithms had been invented to enhance the concealing of the message inside the image cover.

All of these works must be classified depend on strong metrics to evaluate it, so each steganography approach has its advantages in terms of the typical steganography metrics: security, capacity, and robustness. Tradeoffs between the metrics in the approaches are to be considered.

2.2 Related Work

A lot of techniques & methods used to conceal different type (or shape) of secrete messages into different covers media; the following will be some of the work done in this area.

Hiding information into the target media requires following elements:

The cover media (text, image, audio, video) that will hold the hidden data.

The secret message, may be plaintext, ciphertext or any type of data.

The stego function and its inverse.

An optional stego-key (K) or password may be used to hide and unhide the message.

2.2.1 Text Steganography

Text steganography can involve anything from changing the formatting of an existing text to changing words within a text to generating random character sequences or using context-free grammars to generate readable. Text steganography is believed to be the trickiest due to deficiency of redundant information which is present in image, audio or a video file .Storing text file require less memory and its faster as well as easier communication makes it preferable to other types of steganographic methods [3]. Text steganography can be broadly classified into three types: Format based, Random and Statistical generation, Linguistic methods.

First format based methods use and change the formatting of the cover-text to hide the data. **Second** Random and Statistical generation to avoid comparison with a known plaintext, steganographers often resort to generating their own cover [3]. One method is concealing information in random looking sequence of characters. **Third** linguistic steganography specifically considers the linguistic properties of generated and modified text, In this method the synonym of words for some pre-selected are used. The words are replaced by their synonyms to hide information in it [7]. The problem in this approach is that replacement of synonyms may change the meaning or structure of the sentence [8].

There are some of the others popular approaches of text steganography. Such as **Line Shift** which the secret message is hidden by vertically shifting the text lines to some degree [10]. To hide bit 0, a line is shifted up and to hide bit 1, the line is shifted down. **Word Shift**, In this method the secret message is hidden by shifting the words horizontally, i.e. left or right to represent bit 0 or 1 respectively . **Syntactic Method** This technique uses punctuation marks such as full stop (.), comma (,), etc. to hide bits 0 and 1. But problem with this method is that it requires identification of correct places to insert punctuation marks [9].**White Steg** This technique uses white spaces for hiding a secret message. There are three methods of hiding data using white spaces. In Inter Sentence Spacing, we place single space to hide bit 0 and two spaces to hide bit 1 at

the end of each terminating character [10]. In End of Line Spaces, fixed number of spaces is inserted at the end of each line. For example, two spaces to encode one bit per line, four spaces to encode two bits and so on. In Inter Word Spacing technique, one space after a word represents bit 0 and two spaces after a word represents bit 1. **Feature coding**, the secret message is hidden by altering one or more features of the text. Finally the **Secret Steganographic Code for Embedding (SSCE)** it is first encrypts a message using SSCE table and then embeds the cipher text in a cover file by inserting articles a or an with the non specific nouns in English language using a certain mapping technique [11]. Most of the text steganography can be lost if the document been retyped.

The following paragraph is about some of the work which had been done in text steganography:

In the first paper [12] Shirali-Shahreza, M.H. and M. Shirali-Shahreza who deal with the issue of the text steganography, the model focuses on the letters that have got points on it (example English Language had two letters i,j. while Arabic language has 15 pointed letters out of its 28 alphabet letters). Point steganography hides information in the points of the letters. To be specific; they hide the information in the points' location within the pointed letters. First, the hidden information is looked at as binary with the first several bits (for example, 20 bits) to indicate the length of the hidden bits to be stored. Then, the cover medium text is scanned. Whenever a pointed letter is detected its' point location may be affected by the message bit hidden info bit. If message hidden value bit is one the point of the letter is slightly shifted up; otherwise, the concerned cover-text character point location remains unchanged, "In order to divert the attention of readers, after hiding all information, the points of the remaining characters are also changed randomly" The advantages of this method are very secure and hide big mount of the message into the cover text.

In [13] Gutub, A. and M. Fattani. *A,* "That Benefiting from Shirali-Shahreza [12] point steganography and trying to overcome the negative robustness aspect, it proposes a new method to hide information in any letters instead of pointed ones only. This model

use the pointed letters with extension to hold secret bit ‘one’ and the un-pointed Letters with extension to hold secret bit ‘zero’. This proposed steganography method can have the option of adding extensions before or after the letters. To be consistent, however, the location of the extensions should be the same throughout the complete steganography document. The location in the model is after the letters, and the last letter in every words of the cover-text cannot have an extension if a letter cannot have an extension or found intentionally without extension it is considered not holding any secret bits. “Note that letter extension doesn’t have any affect to the writing content. It has a standard character hexadecimal code: 0640 in the Unicode system. In fact, this Arabic extension character in electronic typing is considered as a redundant character only for arrangement and format purposes”.

In [14] Bhattacharyya, S., I. Banerjee, and G. Sanyal the paper link between three variables, first even and odd word size, second the bits (two bits) from the secret message, and finally the space after word. They propose a new method of information hiding in a text by inserting extra blank space (one or two spaces) between the words of odd or even size according to the embedding sequence, and also in some cases the blank spaces in between the words of the original cover text may be used for mapping each two bit of the embedding sequence. The message converted into binary numbers, then encoded by SSCE encrypting algorithm, and it divided it into group of two bit at a time. The probability of the two bit are (00,01,10,11), (00,01) for even word size and (10,11) for odd word size. If the secret message equal ’11’ then find out the word of size odd from the cover-text and check whether one space is there after the word, if not put only one space. Else If the secret message equal ’10’ then find out the word of size odd, from the Text cover and check whether two space is there after the word, if not put only two space. Else If the secret message equal ’01’ then find out the word of size even, from the text cover and check whether one space is there after the word, if not put only one space. Else If the secret message equal ’00’ then find out the word of size even ,from the text cover and check whether two space is there after the word, if not put only two space. There are more restrictions on the cover-text should be considered.

In [11] Banerjee, I., S. Bhattacharyya, and G. Sanyal This study is the same like the previous, except it focuses on the first character of the words in the text cover, if it is vowel or consonant instead of odd and even size. The input messages can be in any digital form and it often treated as a bit stream. The input message is first encrypted using a new code generation technique SSCE. Before embedding into cover text a checking has been done to find out whether the vowels and consonants are placed in the cover text as per the grammatical order, if not place it in proper order. and then secret message has been embed to the cover text by inserting indefinite articles a or an in conjunction with the non-specific or nonparticular nouns in English language based on the mapping information. Check the message sequence and pick first two bit sequence. Starting from the first word of the cover text .If the message equal '11' then find out the word (an) from the cover text and check whether the next word's first character is vowel. Else If the message equal '10' then find out the word (an) from the text cover and check whether the next word's first character is vowel. Change (an) to (a).Else If the message equal '01' then find out the (a) from the cover text and check whether the next word's first character is consonant. Change (a) to (an).Else If the message equal '00' then find out the word (a) from the cover text and check whether the next word's first character is consonant. The repetition to the operation above till the all message embeds to the cover text. The embedding position will be saved in a separate file and encode it with SSCE value and send it to the receiver separately, by reversing the step above "the extraction algorithm" in the receiver site to obtain the message again. This approach is capable of secure transfer of the message compared to earlier techniques. The compression technique on the secret message can be made to maximize the size of the message.

2.2.2 Image Steganography

The most widely used technique today is hiding of secret messages into a digital image. This steganography technique exploits the weakness of the human visual system (HVS). An images represented by collection of the color pixels. For example: a 24-bit

bitmap will have 8 bits, representing each of the three color values (red, green, and blue) at each pixel. If we consider jus the blue there will be 2^8 different values of blue. The difference between 11111111 and 11111110 in the value for blue intensity is likely to be undetectable by the human eye. Hence, if the terminal recipient of the data is nothing but human visual system (HVS) then the Least Significant Bit (LSB) can be used for something else other than color information. This technique can be directly applied on digital image in bitmap format as well as for the compressed image format like JPEG [15].

There are many steganography techniques which are capable of hiding data within an image. There are many ways to hide information in digital images. We look at the following approaches:

Least significant bit insertion, the rightmost bit is called the LSB because changing it has the least effect on the value of the number.

Masking and filtering, Masking and filtering techniques hide information by marking an image and is usually restricted to 24-bit and gray-scale images.

Algorithms and transformations, it hides data in mathematical functions that are in compression algorithms.

Each of these techniques has varying degrees of success [16].

The fallowing paragraph is about some of work that done in image steganography:

In [17] Kumar, A. and R. Sharma proposed a new method of information hiding in an image by encrypt the text message by the RSA algorithm (RSA algorithm is used the recipient public key to encrypt secret data. It provides security by converting secret data into a cipher text, which will be difficult for any intruder to decrypt it without the recipient private key), and embedding the encrypted message into hash-LSB (Hash-LSB is a technique for steganography in which position of LSB for hiding the secret data is determined using hash function) . In this technique the secret message

is converted into binary bits; hash function it will select the positions to embedding each 8 bits at a time in least significant bits of RGB pixel values of cover image in the order of 3, 3, and 2 respectively. According to this method 3 bits are embedded in red pixel LSB, 3 bits are embedded in green pixel LSB and 2 bits are embedded in blue pixel LSB) of the image. The process is continued till entire message of bits will got embedded into the cover image; the reversing of it is considered the extraction function. The performance of the Hash-LSB technique has been evaluated and graphically represented on the basis of two measures are – Mean Square Error (MSE) and Peak Signal to Noise Ratio (PSNR) and obtained values are much better than existing techniques. The results for all stego images using Hash-LSB with RSA technique have been compared to simple LSB substitution with RSA technique which gives very lesser MSE values and higher PSNR values.

2.2.2 Compression Overview

Data compression schemes can be divided into two broad classes: lossless compression schemes, lossy compression schemes, which generally provide much higher compression than lossless compression [18].

Lossy compression techniques involve some loss of information, and data that have been compressed using lossy techniques generally cannot be recovered or reconstructed exactly [18].

Lossless compression techniques, as their name implies, involve no loss of information. If data have been losslessly compressed, the original data can be recovered exactly from the compressed data [18].

Huffman coding is an entropy encoding algorithm used for lossless data compression. The term refers to the use of a variable-length code table for encoding a source symbol (such as a character in a file) where the variable-length code table has been derived in a particular way based on the estimated probability of occurrence for each possible value of the source symbol. It was developed by David A. Huffman [19].

The Huffman Compression Algorithm considered a lossless algorithm. The algorithm constructs a tree that is used to represent the character in the file to be compressed; the tree representation is the same like ASCII and Unicode of character coding and representation.

Huffman Using a tree (a binary tree) all characters are stored at the leaves of a complete tree. See figure 3.4, the tree has eight levels meaning that the root-to-leaf path always has seven edges. A left-edge (black in the diagram) is numbered 0, a right-edge (blue in the diagram) is numbered 1. The ASCII code for any character is 8 bits but in Huffman representation each character has an associated weight equal to the number of times the character occurs in a file.

Huffman's algorithm assumes that we're building a single tree from a group (or forest) of trees. Initially, all the trees have a single node with a character and the character's weight. Trees are combined by picking two trees, and making a new tree from the two trees. This decreases the number of trees by one at each step since two trees are combined into one tree.

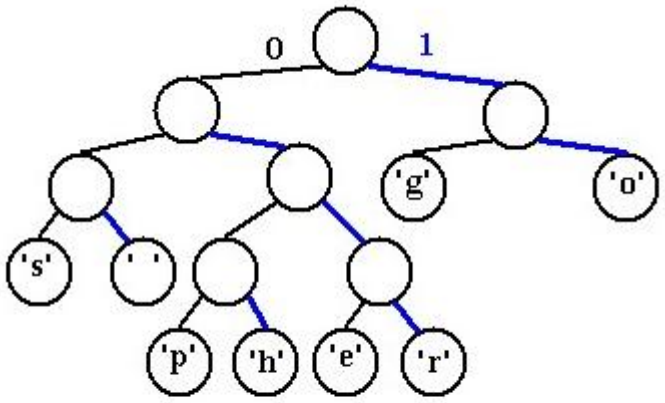


Figure 2.1: Huffman Binary Tree

2.2.3 Multi-Level Steganography

In [20] Al-Najjar, A.J. the paper about multi-level steganography, the first level is by put the text message (M) into black & white image (intermediate-object (I or D) ,the second level by take the output of the first level ”stego of the black & white image” as input to the RGB image (second cover-object (C)). The elements of M are given by the set {M} of size |M|. Each element of M is encoded using $N_b(M)$ bits. Similarly, {I} of size |I|, {C} of size |C|, and {S} of size |S| define the element sets and sizes for the intermediate- (decoy-object) cover-object and the stego-object, respectively. The intermediate-object (or decoy) {I, or D} acts as the “cover” for the “message-object {M}”, and “message” for the “cover-object {C}”. “The message {M} is passed through the transformation T_1 that can include many possibilities. It can be compression, a transform (e.g., discrete cosine transform, Fourier Transform, or Wavelet Transform), private-key or public-key encryption, etc. The transformation T_1can also be a combination of techniques, as required by the particular application. The same can be said about the other transformations T_2 and T_3”. In level one the embedding function embed the message bits to the decoy {D} or {I}, in a linear LSB order M.LSB(0) ->D.LSB(0).Level two the intermediate-object (I or D, decoy) the output from level-1 (Gray scale image Nr x Nc pixels. Embedding into RGB image is as Follows: D.LSB(0,1,2)->R.LSB(0,1,2) of Red color layer, D.LSB(3,4,5) ->G.LSB(0,1,2) of Green color layer, D.LSB(6,7) ->B.LSB(0,1) of Blue color layer. The analysis will take images of size 256 x 256 pixels (black and white image), hence an assumed 65536 pixels. The text is assumed to be embedded 1-bit per pixel. Hence, a maximum of 65536 bits of text can be embedded. 4.1 ASCII Text: With 8-bits per character, the maximum number of characters will be 8192 (65536/8) characters. Assuming 2000 characters per page (25 lines times 80 characters per line), this defines four pages. The multi-level more secure than the normal steganography, as the hidden message appears as noise, this must satisfy most hackers. The authorized recipients have both knowledge about the hidden message, as well as the keys (and other information) required to recover the message.

In [21] Bhattacharyya, S., a secret key steganographic model combining both text and image based steganography technique for communicating information more securely between two locations has been proposed which first uses a plain text as the cover data and the secret message is embedded in the cover data to form the stego text which in turn embedded into the cover image to form the stego image. The proposed text steganography scheme has been inspired by the author's previous work [11] by inserting indefinite articles 'a' or 'an' in conjunction with the non-specific or non-particular nouns in English language based on the mapping information according to the embedding sequence. Here data embedding in an image has been done through Pixel Mapping Method (PMM) within the spatial domain of any gray scale image, Embedding pixels are selected based on some mathematical function which depends on the pixel intensity value of the seed pixel and its 8 neighbors are selected in counter clockwise direction. Before embedding a checking has been done to find out whether the selected embedding pixels or its neighbors lies at the boundary of the image or not. Data embedding are done by mapping each four bits of the secret message in each of the neighbor pixel based on some features of that pixel [22]. The author incorporated the idea of secret key for authentication at both ends in order to achieve high level of security. As a further improvement of security level, the secret message has been compressed and encoded through SSCE values before embedding. This work proposes a new algorithm with higher security features so that the embedded message can not be hacked by unauthorized user. In this work an attempt has been made to increase the level of security of the steganography model by incorporating the idea of secret key along with the use of encoded form of the original message. Besides the data embedding method (PMM) used here for image steganography has been designed in such a way that it can avoid steganalysis also and it will produce a stego image with minimum degradation.

Paper Name	Authors	Year	Type	Level1	Level2	Result
A new approach to Persian/Arabic text steganography	Shirali-Shahreza, M.H. and M. Shirali-Shahreza	2006	Text Steganography			It can be used for Persian/Arabic watermarking.
A novel Arabic text steganography method using letter points and extensions	Gutub, A. and M. Fattani. A	2007	Text Steganography			This method featured security, capacity, and robustness, it useful in hidden exchange of information through text documents and establishing secret communication.
A novel approach of secure text based steganography model using word mapping method (WMM)	Bhattacharyya, S., I. Banerjee, and G. Sanyal	2010	Text Steganography			The results of the proposed method of text steganography. They are security, capacity and robustness.
Novel text steganography through special code generation	Banerjee, I., S. Bhattacharyya, and G. Sanyal	2011	Text Steganography			This approach is capable of secure transfer of the message compared to earlier techniques.
A Secure Image Steganography Based	Kumar, A. and R. Sharma	2013	Image Steganography			The combination of steganographic and

on RSA Algorithm and Hash-LSB Technique						cryptographic technique enhances the security of data and data hiding Technique. RSA algorithm has made our technique more secure for open channel. It has been used with Hash-LSB so that the original text will be embedded into cover image in the form of cipher text.
The decoy: multi-level digital multimedia steganography model	Al-Najjar, A.J.	2008	Multilevel	Image (B&W)	Image (RGB)	The proposed model adds a level of security through the main theme of steganography, The cover object usually does not invite suspicion, since it looks similar to the original object for the general observer.
Data hiding through multi level	Bhattacharyya, S.	2011	Multilevel	Text	Image	The integrated approach of SSCE,

steganography and SSCE						new method of text steganography and image based steganography using PMM has enabled the secure transfer of the message compared to earlier techniques. However to increase the security level different parameter has been considered for achieving better performance.

Table 2.1: Literature review summary

CHAPTER 3

THE PROPOSED MODEL

3.1 Introduction

This chapter describes the new model that is proposed to Development of a Compression Aided Multi-level Steganography System. In the first level hides the secret message (text) inside a cover text, and the second level the Stego text embedded into an image, between the first and second level the compression (Sender Site) and decompression (Receiver Site) will be apply to the Stego-text (as showing in figure 3.1). The model is divided into two major sites the sender site that deals with the embedding processes of the secret message by the first and the second level respectively, and the receiver site that deals with the extraction processes from the second and the first level (in reverse order) to obtain again the Secret message safely as shown in figure 3.4.

The following terms used in the model (Secret-Message) indicate to the Message ,(Cover-text) indicate to the cover of text,(Stego-text) indicate to the output of merging the message and the cover text,(Compressed) it is the output of the stego after it is compressed by a compression algorithm,(Stego-image) indicate to the output of embedding the Compressed text into an image, (Extracted) is the output (Compressed) form the image (Stego-image), and finally (Decompressed) it is output from the decompressed of the Extracted text by the same compression algorithm that is used to make the compression.

3.2 The First Level

The chapter focuses on hiding text (the Secret-Message) into another text (the Cover-text), the inspiration of this work came from the example of the simple method of steganography, that the sender sends a series of integer number (Key) to the recipient with a prior agreement that the secret message is hidden within the respective position of subsequent words of the cover text. For example the series is ‘1, 1, 2, 3, 4,2,4,’ and the

cover text is "**A** **t**eam o**f** fi**v**e men j**o**ined tod**a**y". So the hidden message is "**Atfvoa**". A "0" in the number series will indicate a blank space in the recovered message. The word in the received cover text will be skipped if the number of characters in that word is less than the respective number in the series (Key) which shall also be skipped during the process of message unhide [23]. when the model was studied, it appeared that this model had restrictions on the cover-text depend on the secret message such as the order and the number of the characters on the cover-text. Accordingly, the idea can be applied to other varieties of languages.

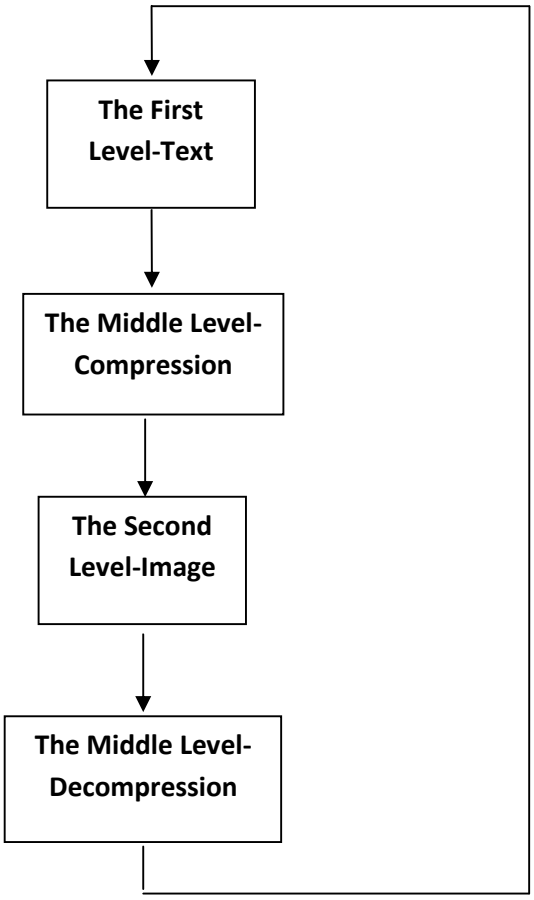


Figure 3.1: Multilevel Steganography System with Compression

To work out the previous model in order to fix the issues above, the idea in the first level or level one of the model is to produce a random Cover-text to hide The Secret-

Message randomly into the Cover-text by using the Pseudo Random Generation (PRNG). The output is seems to be a random text if you do not know about the message and the key (Key that indicates the position of the message into the random Stego-text) see figure 3.2.

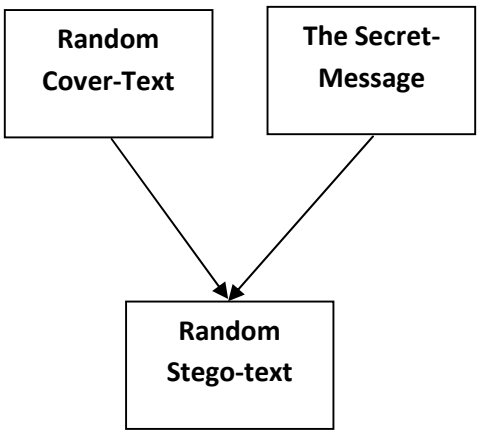


Figure 3.2: First level Embedding Algorithm at the Sender Site

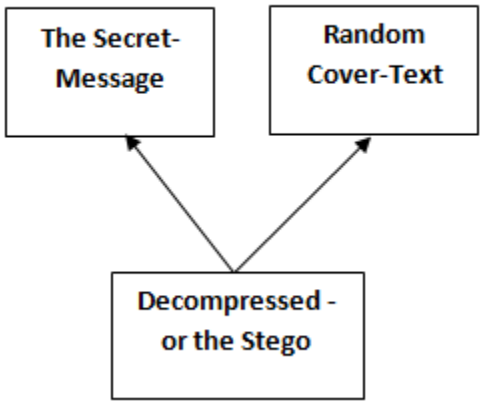


Figure 3.3: First Level Embedding Algorithm at the Receiver Site

The key (or the seed) is use by the random function to produce an array of random integer numbers, those numbers are respectively sent to another function that puts the reminder of two (mod 2) to produce a (0s, 1s) zeros & ones as a binary array, which is

used to embed the message into the random Cover-text randomly, If the element in the binary array equals 1 write one character from the message into the stego-text. Else write one character from the Cover-text into the Stego-text, until the last character from the message is embedded into the Stego-text.

The First Level Algorithm (Sender Site):

This algorithm described in Figure 3.2.

-Enter the Secret Message.

-Calculate the number of the characters in the message.

-Generate the Cover-text more than the number of the message characters.

-Enter a number to be the Hiding Key (or the Seed).

-From the key generate an array of random number by the equation S[i+1]=S[i]*A+B mod n.

S is the Seed, A,B is variable numbers n is the prime number that produces numbers less than its value.

-Generate the binary array (just zeros and ones) by the reminder (mod) of 2 of the above step output numbers.

-Generate the binary array until the ones (1s) numbers in the array equal the number of the Secret-Message characters.

-If the binary element equal 0 write one character from the Cover-text into the Stego-text, else write one character from the Secret-Message into the Stego-text. (Two texts merged into one text file randomly).

-Do until you reach the last character in the Secret-Message. The output is mixed with random Cover-text at Random Position of ever single character of the Secret-Message into the new Cover-text called Stego-text seems to be a random text.

The First Level Algorithm (Receiver Site):

This algorithm described in figure 3.3.

-Receive the Decompressed file (or the Stego-text) from Second level of the receiver site.

-Enter the Extraction key that is determined in the sender site.

-Generate array of integer from the key by the same equation of the first level in the sender site.

-Generate a binary array from the above array of integer by the same way of the first level in the sender site.

-If the binary array element equals 1 write from the stego-text to new text file called The Secret-message, else write one character from the stego-text to new text file called Cover-text2.

-Do until the last character in the stego-text.

3.3 The Compression

The middle level in the model receives the output of the first level (Stego-text) and compresses it by Huffman Compression Algorithm, the output from this compression operation is a binary output file that is ready to be used (or embedded) by the second level, second level is the second step in the model. The decompression step (Receiver Site) is used after the second level done and the Stego-image send to the receiver and the Compressed text Extracted from the Stego-image, this output (Extracted) is used as input to the Huffman Decompressed Function. The output of the decompression is the same Stego-text of the output of the First Level Algorithm (that mentioned above).

3.4 The Second Level

The main function of the second level or level two is to hide the message or the Compressed Stego-text into an image (it considered final step) in the sender site, and the receiver Extracts the compressed Stego-text from the image (the first step) in the receiver site. The LSB is used to embed the data into the image (for more details about LSB See Chapter 1,2).

The Second Level Algorithm (Sender Site):

-Receive the compressed stego from the compression algorithm.

-Read the Cover-image form the type RGB image from the hard disk.

-Calculate the pixel needed by the Compressed input.

-Calculate the height and weight and the pixel number in the image. Pixel is 3bytes (Red, Green, Blue) in RGB.

-If the number of the compressed bit is greater than number of pixel *3 quite, else go to the next step.

-Read the every pixel (3bytes, Red, Green, Blue) that needed to input the Compressed data.

-Read three bits from the compressed data and put them in one pixel in the Last Significant Bit (LSB) of every individual byte in the pixel, one in the red byte, one in the green byte, and the last one in the blue byte in the pixel.

-Do until the end of the Compressed bits.

-Write the number of the pixel that is used in the image at the first 11 pixel.

The Second Level Algorithm (Receiver Site):

-Receive the Stego-image from Sender.

-Read the number of the pixel that contains data (first 11pixel of the image).

-Read all the pixels needed to extract the compressed text that embedded by the second level in the sender site.

-Extract all the LSB from the pixels read above.

-Do until the end of all pixels.

As the Model in the figure 3.5 shows that, there are three level (or step) at the sender site, and 3 level (or step) at the receiver site. Finally the cooperation between all those levels and function must be very accurate; because any little error in one level will produce an issue of embedding and/or extracting the original message (Secret-message) again.

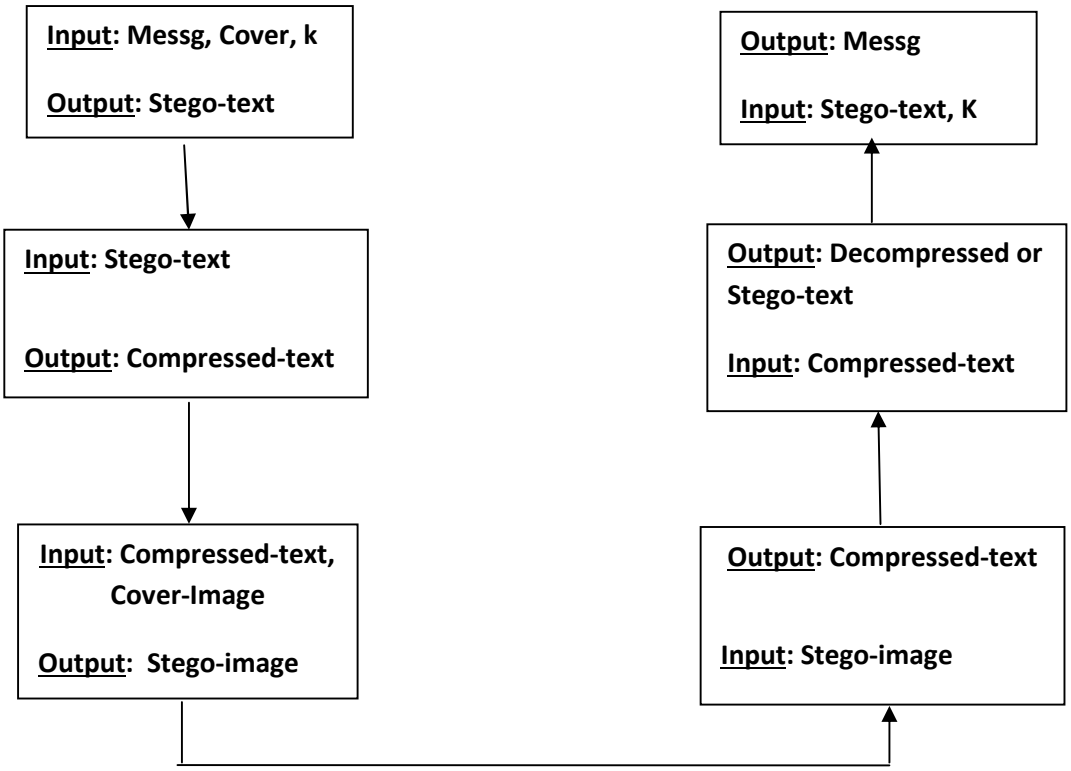


Figure 3.4: Sender and Receiver sites of the Multilevel model

CHAPTER 4

THE RESULT & DISCUSSION

4.1 Introduction

In the previous chapter the details of the proposed model were described and how the message went through different levels (text-compression-image) in the sender side, and how the message returns again from those levels by reversing the order of the levels (image-decompression-text). This chapter shows the implementation of the model in term of a Computer Program. A detailed description of how the program works – by making a number of experiments – as well as presenting the final result driven form the program operation will be included in this chapter as well.

4.2 The Implementation Tools

The implementation of this model is done by using java programming language JDK version 1.7.0 _45 with Integrated Develop Environment (IDE) NetBeans version 7.4 & TextPad under windows 7 operating system. The Image Quality (PSNR & MSE) obtained by using MATLAB.

4.3 The Implementation

The Program that implemented by the tools above is contained a three main interfaces; First one is the main window that leads you to the Hiding window (as Second window) or Extraction window (Third window).As shown in figures 4.1,4.2 and 4.3

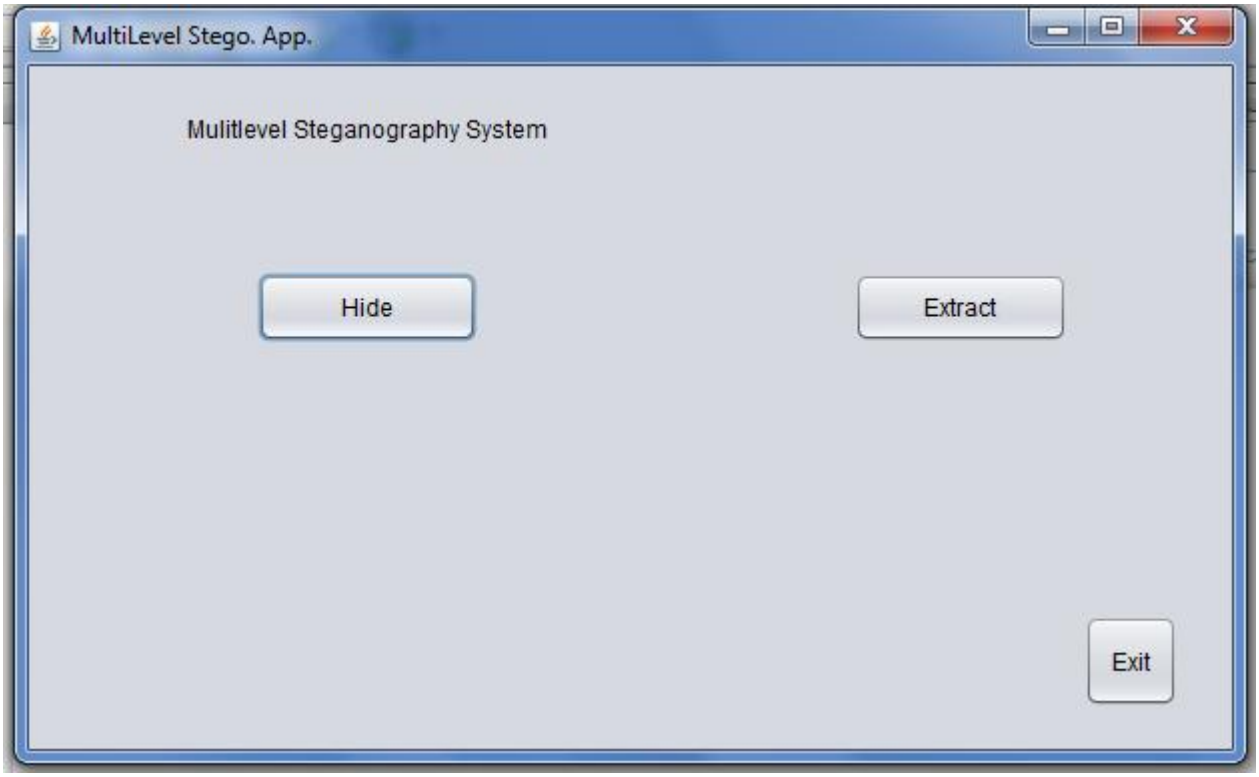


Figure 4.1: Shows the main window in the program.

In the figure 4.2 the Hiding screen the user most enter three inputs, First select or enter his message file direction by using the browsing button, the Second input is the Key (Hiding Key), and finally select or enter the cover image direction (.png extension) and press the OK button.

After that the program will accept your inputs if they are legal, and go through all the levels that belong to the sender side (as it is shown in chapter three).

The Stego-Image directory and the Extraction Key (the output of the hiding process) will be seen (in the text area) at the bottom of the Hiding window.

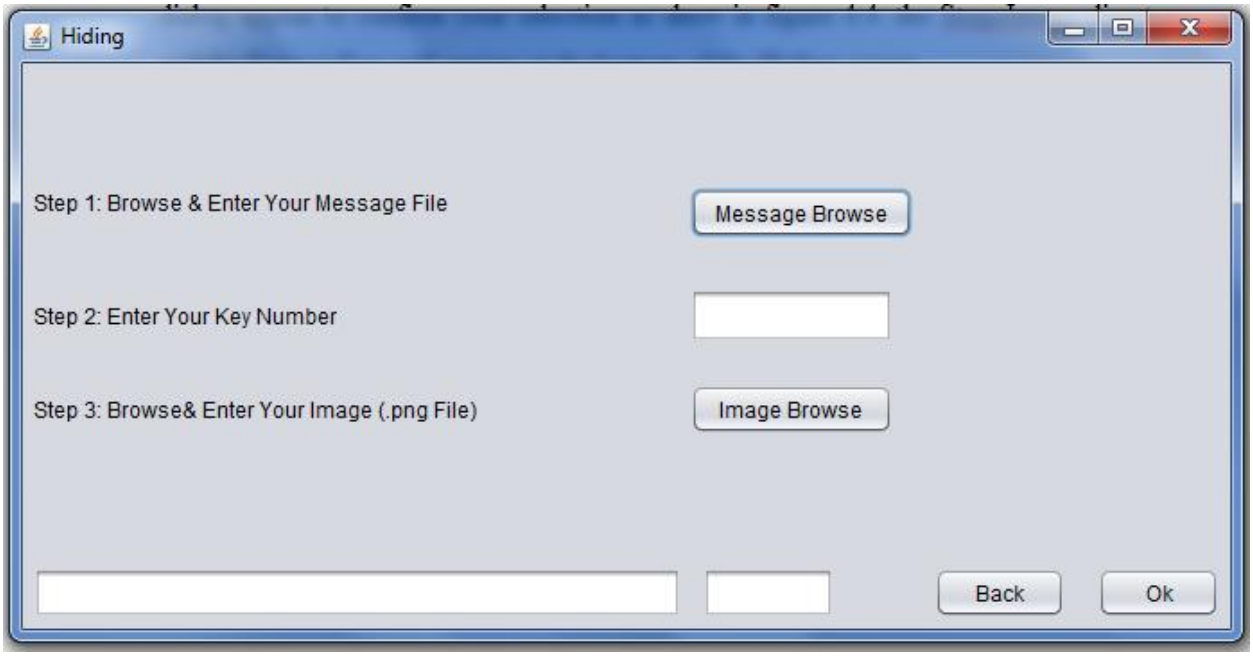


Figure 4.2: Shows the hiding window

The Extraction window will receive the outputs from the hiding process (the Stego-Image directory & the Extract Key) as inputs as shown in figure 4.3 .the Original (or the Secret) Message directory will be seen at the bottom.

Note:

The Hiding Key that the user enters in the Sender Site is different (in term of value) from the Extraction Key , the Extraction Key is generated by the Sender to protect the secret message from unauthorized user to get the message back, even if you know the Hiding Key value , such as the public key idea.

Figure 4.4 shows the browsing window that is used to select the original message & cover image in the Hiding window (sender site) and selection of the Stego-Image in the Extraction window (receiver site).

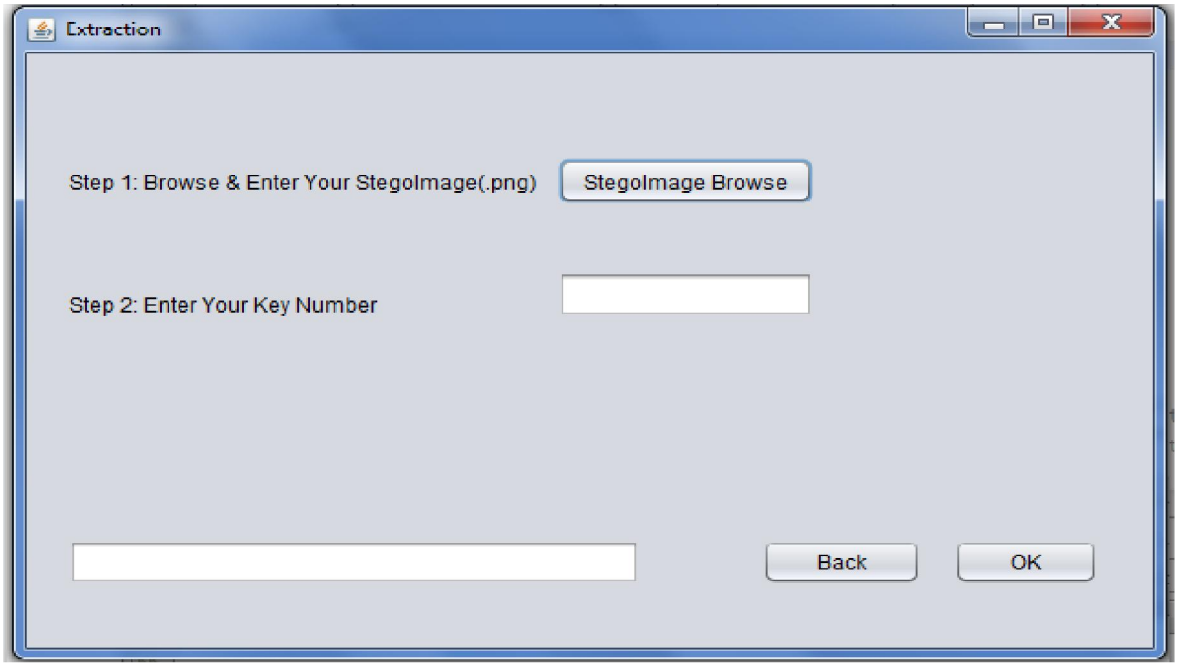


Figure 4.3:Shows the extraction window

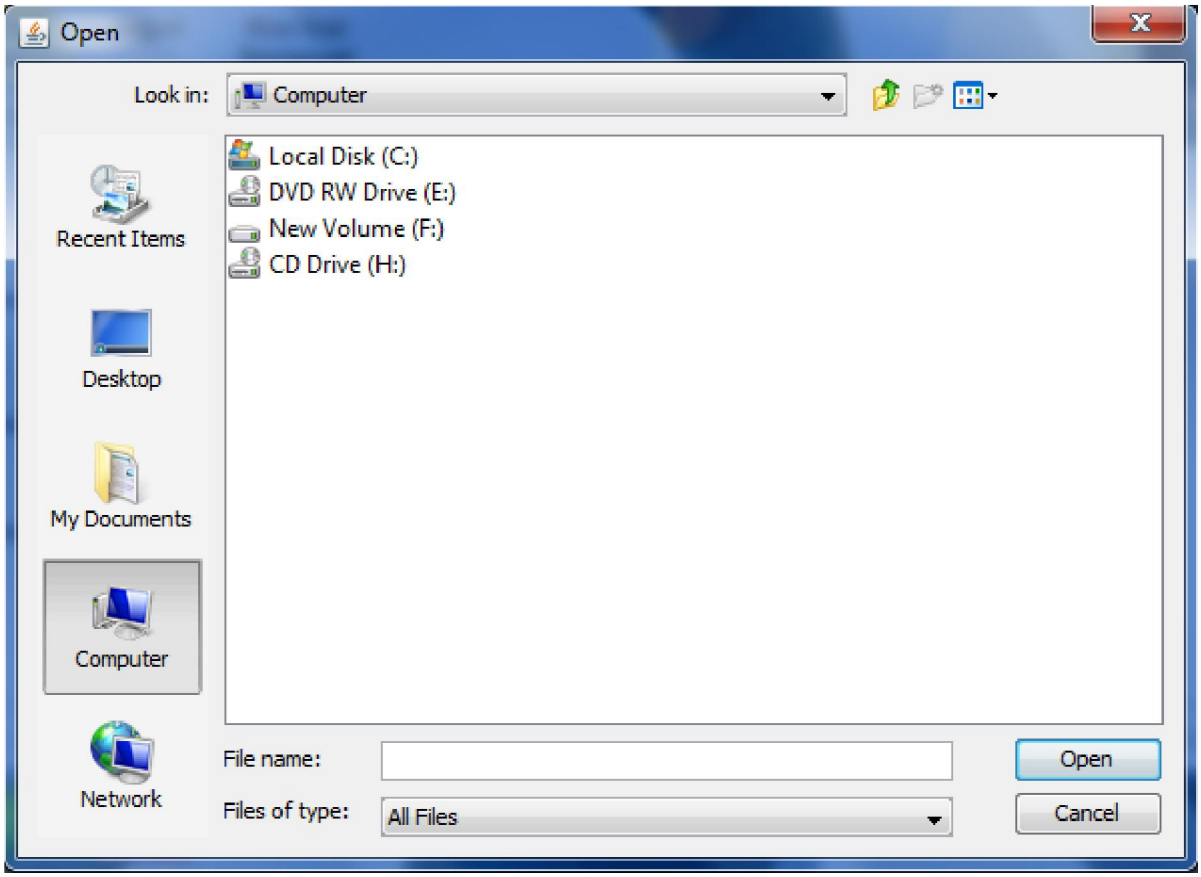


Figure 4.4: Shows the browsing window.

After entering the inputs in the Hiding or the Extraction window a little confirmation message dialog appear to confirm your selections (or the directory) as show in figure 4.5.

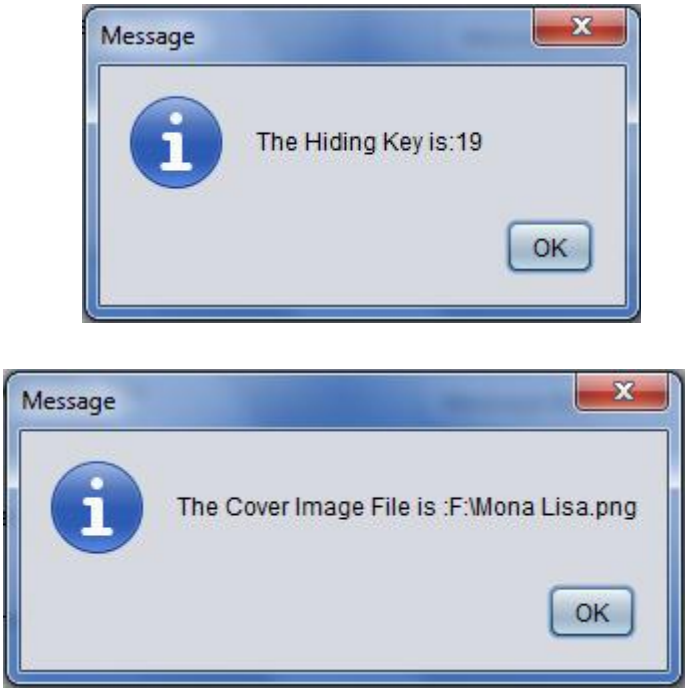


Figure 4.5:Shows the inputs confirmation message

4.4 Result

A lot of experiments were done to test variety of measures such as capacity and Quality, the experiments had been applied to three different message character files (177, 5,893 and 30,011 characters as shown in the Appendix), and although different three steganography standard images in size, height, weight and pixel numbers, the first image is Lenna 512x512 weight height and size 462KB, the second image Mona Lisa 609x709

weight height 1.092KB size, and the third image Baboon 211x239 133KB size. The experiment repeated five times to every single message of the message above, those three tables for just one image. The average table although shown. The quality measured by First Peak Signal to Noise Ratio (PSNR) is an expression for the ratio between the maximum possible value (power) of a signal and the power of distorting noise that affects the quality of its representation, and Second the mean squared error (MSE) of an estimator measures the average of the squares of the "errors", that is, the difference between the estimator and what is estimated. MSE is a risk function, corresponding to the expected value of the squared error loss or quadratic loss.

The terms in the tables below is:

- **Mess-C.No**: is an abbreviation of the Message Characters Number.
- **Cov-C.No**: is abbreviation of the Cover Characters Number.
- **Stego-C.No**: is abbreviation of the Stego-text Characters Number.
- **Comp-Bits**: number of the file bits after compression process.
- **Pixel-No**: number of the Image pixels needed to embed Compressed file.
- **LSB Modification**: is the differences on the LSB of the Stego-Image (after insert the message) and the original image.
- **MSE**: Mean Squared Error of the Stego-Image and the original image.
- **PSNR**: is Peak Signal to Noise Ratio of the Stego-Image and the original image.

Lenna Image Tables:

Image details (512x512) size 462KB tables

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
177	164	341	1,921	640.1	982 (51%)	0.016834	65.8689
177	88	265	1,380	460	681 (49%)	0.000886	78.6551
177	176	353	2,014	671.1	1024 (50%)	0.012711	67.0892
177	175	352	1,981	660.1	1008 (50%)	0.014506	66.5153
177	150	327	1,856	618.2	906 (50%)	0.008565	68.803

Average (177 message characters):

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
177	151	328	1830	609.9	50%	0.0107004	69.3863

Table 4.1: Lenna image with 177 characters of the secret message & Average Table.

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
5,893	5,651	11,544	66,811	22,270.1	33,066 (49%)	0.091193	58.5312
5,893	5,650	11,543	66,912	22,304	33,452 (49%)	0.042555	61.8412
5,893	2,948	8,841	48,510	16,170	24,249 (49%)	0.030860	63.2369
5,893	2,676	8,569	46,481	15,493.2	23,246 (50%)	0.040347	62.0727
5,893	5,892	11,785	68,410	22,803.1	34,413 (50%)	0.0094392	58.3814

Average (5,893 Message characters):

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
5,893	4563	10,456	59,425	19,808	49%	0.04287884	60.81268

Table 4.2: Lenna image with 5,893 characters of the secret message & Average Table.

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
30,011	29,398	59,409	344,727	114,909	172,284 (49%)	0.219086	54.7247
30,011	30,011	60,022	348,817	116,272.1	174,319 (49%)	0.281218	53.6404
30,011	30,008	60,019	348,679	116,226.1	174480 (50%)	0.248193	54.1829
30,011	30,012	60,023	348,898	116,299	174,120 (49%)	0.270763	53.8049
30,011	13,641	43,652	237,659	79,219.2	118,518 (49%)	0.176243	55.66.92

Average (30,011 Message characters):

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
30,011	26,614	56,625	325,756	108,585	49%	0.2391006	54.088225

Table 4.3: Lenna image with 30,011 characters of the secret message & Average Table

The Original Image of Lenna The Stego-Image

Figure 4.6:Shows the image of Lenna & Stego

Mona Lisa Image Tables:

Image details (609x709) size 1.092 MB tables

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
177	176	353	2,008	669.1	987 (49%)	0.002301	74.5120
177	194	371	2,140	713.1	1,075 (50%)	0.003580	72.5921
177	176	353	2,029	676.1	1,033 (50%)	0.002387	74.3518
177	150	327	1,802	600.2	915 (50%)	0.000935	78.4248
177	80	257	1,341	447	636 (47%)	0.000455	81.5534

Average (177 Message characters):

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
177	155	332	1,864	621	49%	0.0019316	76.28682

Table 4.4: Mona Lisa image with 177 characters of the secret message & Average Table.

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
5,893	5,893	11,786	68,397	22,799	34,169 (49%)	0.023839	64.3579
5,893	5,890	11,783	68,483	22,827.2	34,375 (50%)	0.024016	64.3258
5,893	2,948	8,841	48,438	16,146	24,263 (50%)	0.016934	65.8432
5,893	5,893	11,786	68,306	22,768.2	34,378 (50%)	0.023993	64.3300
5,893	5,425	11,318	65,215	21,738.1	32,610 (50%)	0.023646	64.3932

Average (5,893 Message characters):

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
5,893	5,210	11,103	63,768	21,256	50%	0.0224856	64.65002

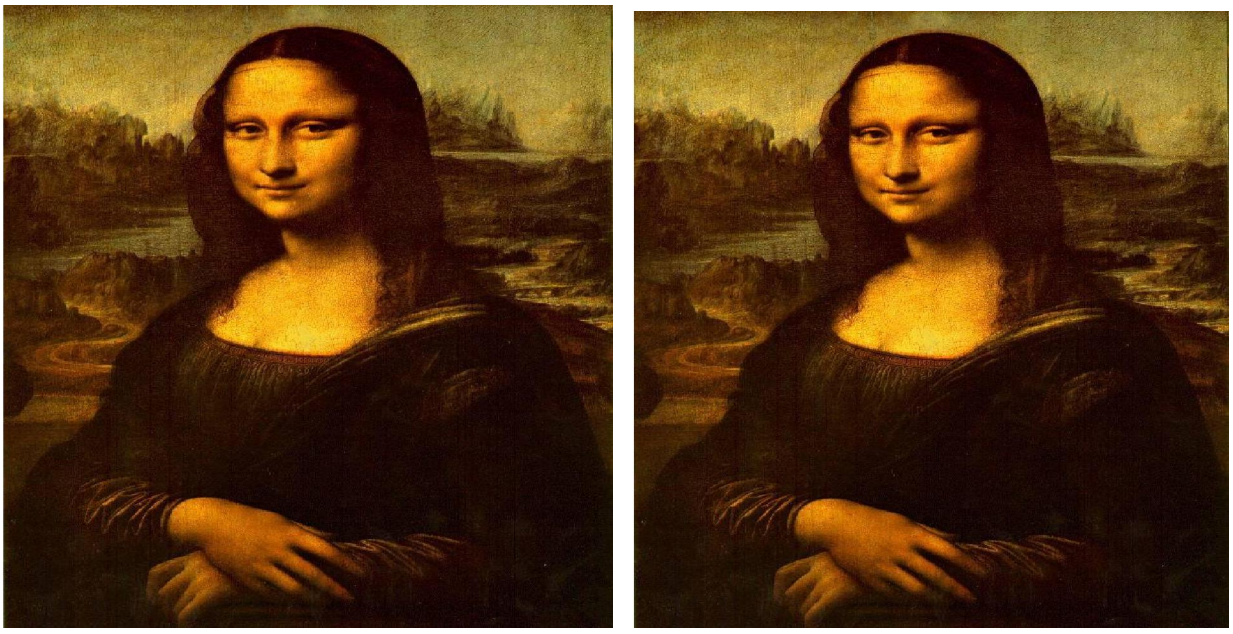
Table 4.5: Mona Lisa image with 5,893 characters of the secret message & Average Table.

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
30,011	30,011	60022	348782	11626.2	175096 (50%)	0.122898	57.2353
30,011	18006	48017	267601	89200.1	133727 (49%)	0.093693	58.4137
30,011	25394	55405	317757	105919	159809 (50%)	0.111466	57.6594
30,011	34007	64018	375291	125097	187607 (49%)	0.130853	56.9630
30,011	28,788	58,799	340,598	113,532.2	170,721 (50%)	0.122545	57.2479

Average (30,011 Message characters):

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
30,011	27,241	57,252	330,006	89,075	50%	0.116291	57.50386

Table 4.6: Mona Lisa image with 30,011 characters of the secret message Average Table.



The Original Image of Mona Lisa The Stego-Image

Figure 4.7:Shows the image of Mona Lisa & Stego

Baboon Image Tables**:**

Image details (211x239) size 133 KB

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
177	168	345	1,944	648	997 (51%)	0.006656	69.8985
177	177	354	2,031	677	952 (46%)	0.006359	70.0971
177	192	369	2,121	707	1,068 (50%)	0.007172	69.5745
177	176	353	1,989	663	983 (49%)	0.006597	69.9375
177	174	351	2,001	667	987 (49%)|	0.006610	69.9288

Average (5,893 Message characters):

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
177	177	354	2,017	672	49%	0.0066788	69.88728

Table 4.7: Baboon image with 177 characters of the secret message & the Average Table

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
5,893	2,948	8,841	48,436	16,145.1	24,290 (50%)	0.615367	50.2395
5,893	6,676	12,569	73,531	24,510.1	36,651 (49%)	0.275238	53.7337
5,893	5,892	11,785	68,465	22,821.1	34,355 (50%)	0.652977	49.9818
5,893	4,987	10,880	62,360	20,786.2	31,142 (49%)	0.328693	52.9629
5,893	5,650	11,543	66,875	22,291.2	33,337 (49%)	0.512377	51.0349

Average (5,893 Message characters):

Mess-C.No	Cov-C.No	Stego-C.No	Comp-Bits	Pixel-No	LSB-Modification	MSE	PSNR
5,893	5,231	11,124	63,933	21,311	49%	0.4769304	51.59056

Table 4.8: Baboon image with 5,893 characters of the secret message & the Average Table.

The Original Image of Baboon

The Stego-Image

Figure 4.8:Shows the image of Baboon & Stego

In all the tables above from left to right columns **Mess-C.No** is the values of the message characters number **Cov-C.No** is the values of the cover characters number, by adding these two values that gave us the **Stego-C.No** values (level one), after the compression of the **Stego-C.No** file the values in bits found in the column **Comp-Bits**, every 3 bits from **Comp-Bits** need 1 image pixel, that means by dividing the **Comp-Bits** by 3 that gave us the column values of **Pixel-No** (the image pixels needed to embed the Comp-Bits using LSB techniques), PSNR and MSE calculate the image quality after the embedding process (Level two).

if the values in the column **Mess-C.No** multiplied by 8, that gave us the message size in bits instead of characters number (e.g 177x8=1,416), if the comparison made between the message value in bits and the values of the column **Comp-Bits** in the same table(same row), you will find small differences may be less or greater than the message bits value. Those differences are due to two reasons, first the random generation of the cover characters, second the Compression Algorithm that depends on the character frequency in the file, repeated characters which represented as less bits.

The result shown that on average image quality between 74 PSNR of Mona Lisa image (the biggest image size) that hides 177 message characters (the smallest message characters), and the 52 PSNR of Baboon (the smallest image size) image that hold 5,893 message characters.

In all the tables above there are relationship between Cov-C.No, LSB-Modification, PSNR and MSE. If Cov-C.No is big that is a good indicator of the security(in level one) but lead to low image quality (level tow), and if Cov-C.No small (as 88 in table 4.1) that low security (in level one) but lead to high image quality (level tow) .tradeoff between the security in level one and the quality in level tow.

CHAPTER 5

THE CONCLUSION & FUTURE WORK

5.1 The conclusion

The purpose of all this work is to concealing the sensitive information from any unauthorized use by hiding the Secret Message into different levels of security with the ability of extracted the Secret Message again from all those levels. The compression algorithm in the middle of the two levels adds a good feature to the model, the main purpose of any compression algorithm that reduces the size of the files, the model work by adding the random Cover-Text to the original Secret Message, that produces a large file size (as output) needed to be compressed. After the compression process is done, the output of the compressed file is different from the original file itself, but it is near to the original Secret message in size. Any unauthorized user cannot know that it is an output of compressed file is not the file itself.

Because the sender and receiver communicate just by image (Stego-Image, the output of the last level in the sender site), so the concentration is given to that Image, and it is necessary to measure the image that carries the Secret Message without being noticed by the Human Visual System (HVS), the model gives good images quality result (for all the three images with the three different messages size).

On the other hand, level one (that hides text in text) is measured, such as the number of the Cover-Text characters that used to randomize the message inside it, and it used as rate of security (as much as the number of the Cover-Text is large that means a good security indicator). In addition to that level one Embedding operation is fast (by embedding 8bits at a time instead of 2 or 3bits at a time).

The advantages of this model are; first it adds additional level of security, by encrypting the Secret Message and the random cover characters by any encryption algorithms and passed as input to the Hiding process, so the receiver cannot obtain the Secret Message unless by the decryption of the file (Flexibility). Second if the sender decides to send the same message more than once, the output of the message will be different (because of the First level algorithm & Huffman Compression algorithm).

5.2 Recommendations

It is recommended that level one algorithm should be used (text steganography) with compression to add more security to the information traffic in the networks, or used with encrypted traffic (such as Virtual Private Network (VPN)) to add extra level of security. All that depends on the sensitivity of the information that transferred across the networks, it can be applied to either encrypted or unencrypted information.

5.3 Future work

The future work will be focused on five sites of our model, first alter the design to accept different types of messages. Second expand the range of the image extension that is used in the second level instead of just PNG extension. Third apply the random embedding operations in the level one to the binary outputs of the secret message and the cover-text to hide the data more deeply.

Fourth apply one of integrity techniques (such as MD5) to the compressed output before embedded into the image, that makes the receiver be able to check the data integrity before being accepted. Finally Encrypt pixel number in the first 11 pixel at the Stego-Image.

APPENDIX

Appendix shows many types of messages, cover, and Stego files snapshots.

Information security is the practice of defending information from unauthorized access, use, disclosure, disruption, modification, perusal, inspection, recording or destruction.

Figure A: Shows the number first secret message (177 characters).

Text & Image Steganography:
The fallowing paragraph is about some of work that done in text steganography:
In the first paper that deal with the issue of the text steganography, the mode focus on the letter that have point on it (example English Language had two letters i,j. while Arabic language has 15 pointed letters out of its 28 alphabet letters). Point steganography hides information in the points of the letters. To be specific; they hide the information in the points' location within the pointed letters. First, the hidden information is looked at as binary with the first several bits (for example, 20 bits) to indicate the length of the hidden bits to be stored. Then, the cover medium text is scanned. Whenever a pointed letter is detected its' point location may be affected by the message bit hidden info bit. If message hidden value bit is one the point of the letter is slightly shifted up; otherwise, the concerned cover-text character point location remains unchanged, "In order to divert the attention of readers, after hiding all information, the points of the remaining characters are also changed randomly" [23]. The advantages of this method are very secure and hide big mount of the message into the cover text.
"Benefiting from Shirali-Shahreza [23] point steganography and trying to overcome the negative robustness aspect, it proposes a new method to hide information in any letters instead of pointed ones only. This model use the pointed letters with extension to hold secret bit 'one' and the un-pointed Letters with extension to hold secret bit 'zero'. This proposed steganography method can have the option of adding extensions before or after the letters. To be consistent, however, the location of the extensions should be the same throughout the complete steganography document. The location in the model is after the letters, and the last letter in every words of the cover-text cannot have an extension if a letter cannot have an extension or found intentionally without extension it is considered not holding any secret bits. "Note that letter extension doesn't have any affect to the writing content. It has a standard character hexadecimal code: 0640 in the Unicode system. In fact, this Arabic extension character in electronic typing is considered as a redundant character only for arrangement and format purposes"[31].
The authors in this paper link between three variables, first even and odd word size, second the bits (two bits) from the secret message, and finally the space after word. They propose a new method of information hiding in a text by inserting extra blank space (one or two spaces) between the words of odd or even size according to the embedding sequence, and also in some cases the blank spaces in between the words of the original cover text may be used for mapping each two bit of the embedding sequence. The message converted into binary numbers, then encoded by SSCE encrypting algorithm, and it divided it into group of two bit at a time. The probability of the two bit are (00,01,10,11), (00,01) for even word size and (10,11) for odd word size. If the secret message equal '11' then find out the word of size odd from the cover-text and check whether one space is there after the word, if not put only one space. Else If the secret message equal '10' then find out the word of size odd, from the Text cover and check whether two space is there after the word, if not put only two space. Else If the secret message equal '01' then find out the word of size even, from the text cover and check whether one space is there after the word, if not put only one space. Else If the secret message equal '00' then find out the word of size even ,from the text cover and check whether two space is there after the word, if not put only two space. [56]. There are more restrictions on the cover-text should be considered.
This study is the same like the previous, except it focuses on the first character of the words in the text cover, if it is vowel or consonant instead of odd and even size. The input messages can be in any digital form and it often treated as a bit stream. The input message is first encrypted using a new code generation technique SSCE. Before embedding into cover text a checking has been done to find out whether the vowels and consonants are placed in the cover text as per the grammatical order, if not place it in proper order. and then secret message has been embed to the cover text by inserting indefinite articles a or an in conjunction with the non-specific or nonparticular nouns in English language based on the mapping information. Check the message sequence and pick first two bit sequence. Starting from the first word of the cover text .If the message equal '11' then find out the word (an) from the cover text and check whether the next word's first character is vowel. Else If the message equal '10' then find out the word (an) from the text cover and check whether the next word's first character is vowel. Change (an) to (a).Else If the message equal '01' then find out the (a) from the cover text and check whether the next word's first character is consonant. Change (a) to (an).Else If the message equal '00' then find out the word (a) from the cover text and check whether the next word's first character is consonant. The repetition to the operation above till the all message embeds to the cover text. The embedding position will be saved in a separate file and encode it with SSCE value and send it to the receiver separately, by reversing the step above "the extraction algorithm" in the receiver site to obtain the message again. This approach is capable of secure transfer of the message compared to earlier techniques [69]. The compression technique on the secret message can be made to maximize the size of the message.

Figure B: Shows the number two secret message (5,893 characters)

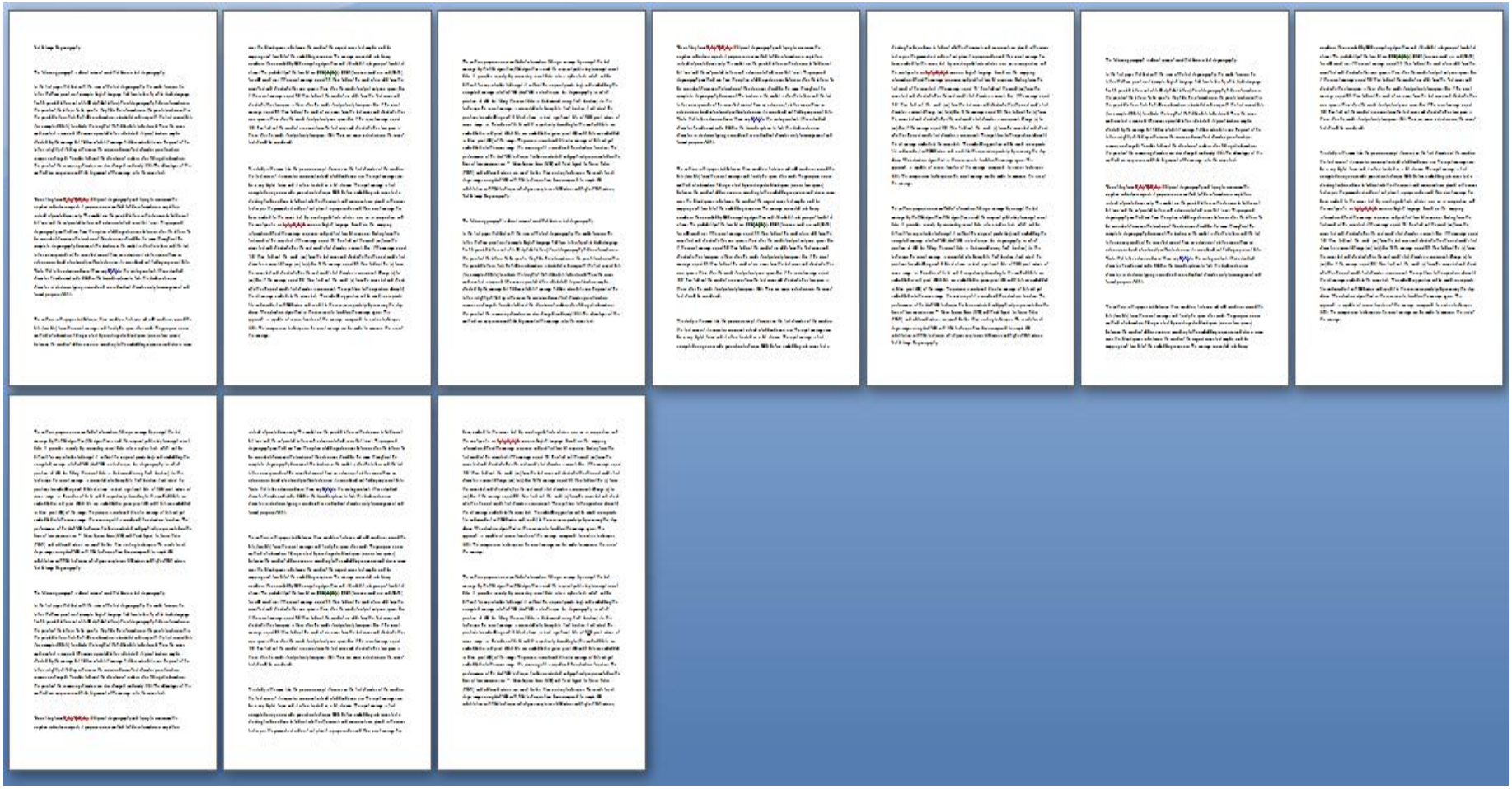


Figure C: Shows the number three secret message (30,011characte) 10 pages

```
%))5o7Q3eIq2<1<+i07!17&j3JP&OfK+066?B)+>q#NS5+!S8b4lgkF}
SENR<3%:fAdXrs55.53Ix&KXk{})J}}5nQ8SQuY(bOwd}GO4Xn3?.K<S+FY&WW0!I?Cw{5+UM1x?
7**.x+1+QD{}(tBR23{w85}ubUA5JvIw+:0>+w8#k57o858M528#xng6#6#3)AE32w?
e>E4<hqO7U#Na:>ut4<j2e6!P>lD>Q5q3D.cOO&m#r8U84Q46g.M.w<p:yrHJ%F:F8A4j:e%}
pcU.l2Dhu{X76OC<1m77OeL&0T+R{132}4UsX1OKV2n:04}7V)ryO2#>&2Ts7a}}H?>368DN517!
g<U&xF
```

Figure D: Shows random Cover-text genreted for 177 Messsage characters.

```
I%n))5foorm7aQt3ione IqSe2c<1<ur+itiy0 7is !t17he& j3JprPac&tOifce Ko+Of 6d6?
Bef)en+d>iqng #iNSnf5o+!Srm8atbi4oln fgrkFom} SENunRau<t3h%ori:zfAedd
Xrsac5ce5s.s5, u3sIxe,& KXkdi{sc}l)oJsur}e}5, ndQ8SisQruupYt(ionb,Ow mdo}
GOdi4fiXcna3tio?n.K, <pS+FerYus&aWlW, iOn!Isp?eCw{ct5io+nU,M re1cx?or7d**.inxg
+o1r+ deQsD{tr}u(tBctRio2n3.
```

Figure E: Shows Stego-text after 177 message characters hided randomly.

```
0100000001011111100110101100000100110110111110010011001001111101101011110011111100100
0010001111111110000100100101100111100001110010110001010101010001111000101110110111001 1
0111001000111001010000001000001010100110111011000110101001000010101110000111001011001 1
0010000100001101110010010101001110010110100101100001100101011101011101111111000011100 10
0000000101010010100100110111011111010001010111001101101001101100011110000001010110100 01
0110110000110000010111100101001111101110010001100110111101100011100110110110001010011 0
1100111101101101100001111100110000011010011000001100100110100000101111100111001000110 0
0101111111101111110100111011010111110110001100101000011111010011101101100101010110010 1
1111001110111100101001011011100000110111000011010111001001011011111110110100010100000000
1001101100010010100010011111101010010000000111100011000010111010101101010100101101111 1
0101010001101011011111101010010111011001101101001001001011110100101110101010110011011 1
1010011011110110101111100011001001101010101001111100111011100011111111010101101001110 01
1000110010010010110101011101010011111001111110011001010110101000110100001111101100111 0
1110111010011110111110011101110001111111110000100111010010001000010101001010011000110 1
0111111011011010000011110011111000010000110110100000101010111001110101100000011100100 01
1111011101111000001000110111111010100101101010010110101111100001110010011011001010111 0
1101001011010111101110011101101110111101010011000111001111000011101000101111010101001 1
1011100100100100000111000100111010111100010010111000111111011110011110010001111101010 1
1001001011000101100111101011000101111111111101101110111111100001100000000010001111001
0101100011100111011011111101110001101100001001011110010101101110100101000011011100011
0000010110111000111110000010111011010111100000111101000111110000011110101000111011
```

References:

1. *Information_security*. [cited 2014 16/2/].
2. Isbell, R., *Steganography: hidden menace or hidden saviour*. Steganography White Paper, 2002. **10**.
3. Agarwal, M., *TEXT STEGANOGRAPHIC APPROACHES: A COMPARISON*. International Journal of Network Security & Its Applications, 2013. **5**(1).
4. Amin, M.M., et al. *Information hiding using steganography*. in *Telecommunication Technology, 2003. NCTT 2003 Proceedings. 4th National Conference on*. 2003. IEEE.
5. Krenn, R., *Steganography: Implementation & Detection*. found online at<http://www.krenn.nl/univ/cry/steg/presentation/2004-01-21-presentation-steganography.pdf, 2004.
6. Petitcolas, F.A., R.J. Anderson, and M.G. Kuhn, *Information hiding-a survey*. Proceedings of the IEEE, 1999. **87**(7): p. 1062-1078.
7. Niimi, M., et al., *A framework of text-based steganography using sd-form semantics model*. IPSJ Journal, 2003. **44**(8).
8. Thampi, S.M., *Information hiding techniques: A tutorial review*. arXiv preprint arXiv:0802.3746, 2008.
9. Cummins, J., et al., *Steganography and digital watermarking*. School of Computer Science, The University of Birmingham, 2004. **14**: p. 60.
10. Bender, W., et al., *Techniques for data hiding*. IBM systems journal, 1996. **35**(3.4): p. 313-336.
11. Banerjee, I., S. Bhattacharyya, and G. Sanyal. *Novel text steganography through special code generation*. in *Proceedings of International Conference on Systemics, Cybernetics and Informatics (ICSCI-2011), Hyderabad, India*. 2011.
12. Shirali-Shahreza, M.H. and M. Shirali-Shahreza. *A new approach to Persian/Arabic text steganography*. in *Computer and Information Science, 2006 and 2006 1st IEEE/ACIS International Workshop on Component-Based Software Engineering, Software Architecture and Reuse. ICIS-COMSAR 2006. 5th IEEE/ACIS International Conference on*. 2006. IEEE.
13. Gutub, A. and M. Fattani. *A novel Arabic text steganography method using letter points and extensions*. in *WASET International Conference on Computer, Information and Systems Science and Engineering (ICCISSE), Vienna, Austria*. 2007.
14. Bhattacharyya, S., I. Banerjee, and G. Sanyal, *A novel approach of secure text based steganography model using word mapping method (WMM)*. International Journal of Computer and Information Engineering, 2010. **4**(2): p. 96-103.

15. Das, S., et al., *Steganography and Steganalysis: different approaches.* arXiv preprint arXiv:1111.3758, 2011.

16. Nithyanandam, P., et al., *A spatial domain image steganography technique based on matrix embedding and huffman encoding.* International Journal of Computer Science and Security (IJCSS), 2011. **5**(5): p. 456.

17. Kumar, A. and R. Sharma, *A Secure Image Steganography Based on RSA Algorithm and Hash-LSB Technique.* International Journal, 2013. **3**(7).

18. Khalid_Sayood, *Introduction_to_Data_Compression*. Fourth_Edition ed. 2012.

19. *Huffman Compression Algorithm*. [cited 2014 2/3/].

20. Al-Najjar, A.J. *The decoy: multi-level digital multimedia steganography model*. in *WSEAS International Conference. Proceedings. Mathematics and Computers in Science and Engineering*. 2008. World Scientific and Engineering Academy and Society.

21. Bhattacharyya, S., *Data hiding through multi level steganography and SSCE.* Journal of Global Research in Computer Science, 2011. **2**(2).

22. Bhattacharyya, S. and G. Sanyal. *Hiding Data in Images Using Pixel Mapping Method (PMM)*. in *Security and Management*. 2010. USA.

23. Nagaria, B., Parikh, A., EEP, M. & Shrivastav, N. *Steganographic Approach for Data Hiding using LSB Techniques*. International Journal of Advanced Computer Research 2012.