Al al-Bayt University

Prince Hussein Bin Abdullah College of Information Technology

Computer Science Department

Thesis Title

Hiding Encrypted Data in MP3 Frames Using Steganography

By

Mohammad Wahead Al–Qassas

2010

Title of Thesis

Hiding Encrypted Data in MP3 Frames Using Steganography

By

Mohammad Wahead AL-Qassas

Supervisor: Mamoun AL-Rababaa

A Thesis to the
Scientific Research and Graduate Faculty in Partial Fulfillment of the
Requirements for the Degree of Master of Science
In Computer Science

Members of the Committee	**Approved**
Dr. Mamoun S. AL-Rababaa (Supervisor)	
Dr. Jehad Odeh	
Dr. Venus W. Samawi	
Dr. Essam F. Al-Daoud	

Al al-Bayt University
Mafraq, Jordan
2010

ACKNOWLEDGMENTS

In the name of Allah, the most gracious and the most merciful

First, I would like to thank to Allah SWT for giving me the ability and strength to complete this thesis in its good way.

My appreciation, thanks and gratitude to my supervisor, Dr. Mamoun S. AL-Rababaa for his complete suggestion, guidance and helpful support to carryout the different stages of my research with great deal of success during the implementation this research.

I would like to thank my father and my mother who always been there for me, and also to my brother, Dr. Wael and whole of my family.

Table of Contents

List of Tables

List of Figures

List of graphs

List of Abbreviations

AES	Advanced Encryption Standard
APFs	all-pass digital filters
ABR	Average Bit-rate
BAF	before all MP3 frames
BF	between MP3 frames
B	Byte
CBR	Constant Bit-rate
CRC	Cyclic Redundancy Checksum
DES	Data Encryption Standard
DSSS	Direct Sequence Spread Spectrum
db	Decibel
XOR	exclusive or
FHSS	Frequency Hopped Spread Spectrum
GSM	Global System for Mobile communication
gr.	granule
Hz	Hertz
HIA	Hiding In Ancillary
ISO	International Organization for Standardization
kbps	kilo bit per second
KB	Kilo Byte
L	Layer
LSB	Least Significant Bits
MB	Mega Byte
ID3	metadata
ID3v	metadata version
MPEG	Moving Picture Coding Experts Group
m-peg	Moving Picture Coding Experts Group
MP3	MPEG-1 or MPEG-2 Audio Layer 3 (or III)
N	Noise
OTP	one-time pad
Pa	Pascal
RAF	Replace in All Frame
RMD	Replace in Main Data
RMD-s	Replace in Silent frame
reserv.	reservation
RSA	Rivest, Shamel and Aldeman
RSA	Rivest, Shamir and Adleman
rms	root mean square

scfsi	Scale Factor Selection Information
S	Signal
SNR	Signal-to-Noise Ratio
S/N	Signal-to-Noise Ratio
sync.	synchronization
VBR	Variable bit-rate
V	Version
WDT	Wave Differential Technique
WBSS	Wave-Based Steganography System

List of Concepts

Concept	Meaning in Arabic
Cryptography	علم التشفير
Steganography	علم الاخفاء
Encryption	التشفير
Decryption	فك التشفير
Cover file	غطاء الملف
Examination	الفحص
Extraction	استخراج
Capacity	السعة
Robust	القوة
Security	الحماية

Abstract

Steganography is a form of data hiding; it is a way for protecting information by embedding data into digital media for the purpose of identification, copy right and secret message.

Multimedia files are widely used as cover data in steganography, some steganography technique depend on changing the least significant bits (LSB), others try to add stegano data within the multimedia frames.

Hiding data within MP3 files has its own difficulty due to the nature of the MP3 format structure. Data in MP3 files is represented using Huffman coding, however MP3 files can be efficient since the MP3 file size could be large enough to carry the needing information.

In this study we hide data in the MP3 frames using two approaches; hiding the data in ancillary part in the MP3 frames (HIA) and by replacing the audio frames by data. The second approach has to sub methods: Replace main data (RMD) which hides data by replacing the main audio data in some frames. Or replace audio frames, (RAF) which hides data by replacing the whole fields of frames that has been selected except the frame header. The methods tried to have minimum effect on audio quality or the MP3 file size.

First method hides data by replacing the last bytes of frames by the encrypted data. To evaluate this method we calculate the signal to noise ratio (SNR) due to embedding process for different sizes of data hidden in the MP3 file. The size of data that can be hidden using this method is about 0.2% of the cover MP3 file size.

In this method we were capable at hiding up to 14 KB of data in a 14 MB of MP3 file with an SNR (Signal to Noise Ratio) value about 30 db.

It was found that increasing the hiding rate causes a decrease SNR value.

Second approach replaces the main data of a frame each 38 audio frame. The size of data that can be hidden is about 2.2% of the cover file size.

In this method we were capable at hiding up to 124 KB of data in a 5 MB of MP3 file with an SNR value about 34 db.

Modifying the second approach by replacing a complete frame every 38 audio frame. Increases the hiding rate to 2.6% of the cover file size. However this has a drawback on SNR values.

In this method we were capable at hiding up to 217 KB of data in a 8 MB of MP3 file with an SNR value about 31 db.

Keyword: Steganography, MP3, Encryption, Decryption, SN

1. Introduction

It is often thought that communications may be secured by encrypting the traffic, but this has rarely been adequate in practice for that many researchers concentrated on methods for hiding messages rather than encipher them [1].

The study of communications security includes not just encryption but also traffic security, whose essence lies in hiding information. This discipline includes such technologies as spread spectrum radio, which is widely used in tactical military systems to prevent transmitters being located; temporary mobile subscriber identifiers, used in digital phones to provide users with some measure of location privacy; and anonymous retailers, which conceal the identity of the sender of an email message. An important sub discipline of information hiding is steganography [2].

While cryptography is about protecting the content of messages, steganography is about concealing their very existence. It comes from Greek roots (στεγαυοζ,γραφεω), literally means 'covered writing' [3], and is usually interpreted to mean hiding information in other information.

Examples include sending a message to a spy by marking certain letters in a newspaper using invisible ink, and adding sub-perceptible echo at certain places in an audio recording.

Modern Steganography was quietly created in 1985, with the advent of the Personal Computer. A problem existed where information needed to be sent safely and securely between parties across restrictive communications channels [4].

Conventional cryptography is like shipping a safe in an armored car with a regiment of soldiers around it. Everyone knows that there's something secret inside. 'R. J. Anderson' [4] describes steganography, the science of hiding or disguising information so it can only be found by someone who knows where to look.

Until recently, information hiding techniques received much less attention from the research community and from industry than cryptography, but this is changing rapidly (table 1.1), and the first academic conference on the subject was organized in 1996 [4]. The main driving force is concern over copyright; as audio, video and other works become available in digital form, the ease with which perfect copies can be made may lead to large-scale unauthorized copying, and this is of great concern to the music, film, book, and software publishing industries. There has been significant recent research into digital 'watermarks' (hidden copyright messages) and 'fingerprints' (hidden serial

numbers); the idea is that the latter can help to identify copyright violators, and the former to prosecute them.

TABLE 1.1 Number of publications on digital watermarking during the past few years according to INSPEC, January 99. Courtesy of J.-L. Dugelay [4].

Year	1992	1993	1994	1995	1996	1997	1998
Publications	2	2	4	13	29	64	103

Finally, there are a number of non-competitive uses of the technology, such as marking audio tracks with purchasing information so that someone listening to a piece of music on his car radio could simply press a button to order the CD [1]. Fabien A. P. Petitcolas et. al. Paper [1] organized as follows. Firstly, he will clarify the terminology used for information hiding, including steganography, digital watermarking and fingerprinting. Secondly the researcher will describe a wide range of techniques that have been used in a number of applications, both ancient and modern, which he will try to juxtapose in such a way that the common features become evident [1].

Then, the researcher will describe a number of attacks against these techniques; and finally, he will try to formulate general definitions and principles. Moving through the subject from practice to theory may be the reverse of the usual order of presentation, but appears appropriate to a discipline in which rapid strides are being made constantly, and where general theories are still very tentative [1].

2. Research Objectives

1. Building a software module to hide a large amount of confidential information encoded to be circulated between the peoples and by using audio media container by implementing information within MP3 frames.
2. Keeping the multimedia file (MP3 file) away of any change of its size or its quality, which may be leads to discovered out the encoding procedure.
3. The ability to restore the encoded data from MP3 files in its original form.

3. Scope of the Study

This study aims to hide encrypted data in MP3 files (MPEG_version1_layer3). The methods used will aim to replace parts of MP3 frames with out affects on the size of the cover files. Evaluating for the processes used in this research will be done by measuring the noise appeared to the MP3 file after embedding data and calculating the SNR.

4. Related Work

The scientific study of steganography has been dealt a number of researches. Every one has adopted a different way of hiding data.

- **Hosei Matsuoka (2006) [5]** Presented spread spectrum audio data hiding method. This method is based on dealing with the original Sound by sub-band phase shifting. It introduces phase shifting in audio signals to reduce the correlation with PN (Pseudo-random Noise) signal per each sub-band. The method calculates the frequency masking threshold using psycho acoustic model, data signal is spread by a M-sequence code, and the spread signal is embedded in audio below the frequency masking threshold.
- **Jessica Fridrich et. al. (2006) [6]** Use matrix embedding method in steganography to improve their embedding efficiency in the number of message bits embedded using random linear codes for large payloads and simple code to find closest codeword to embedding messages
- **Rashid Ansari et. al. (2004) [7]** Exploit the lower sensitivity of human auditory system (HAS) to phase distortion in audio compared with magnitude distortion. And the data embedding with a controlled alteration of phase in the selected sub-band signal by simply passing the signal through a suitable configuration of APFs (all-pass digital filters) with distinct patterns of pole-zero pairs using suitable all pass digital filters.
- **Diqun Yan et. al. (2008) [8]** Propose a reversible data hiding method for digital audio using prediction error expansion technique in four steps. In the first the prediction error of the original audio is obtained by applying an integer coefficient predictor. Then a location map is set up to record the expandability of all audio samples, and then it is compressed by lossless compression coding and taken as a part of secret information. And the reconstructed secret information is embedded into the audio using 1bit from 16bit/sample to embedding the data in different type of audio file.
- **Beixing Deng et. al. (2007) [9]** Proposed a novel information-hiding capacity. This method modified some quantized spectrum values of audio layer-III such as increase resolution of frequency to embed secret information into audios without any difference between original audio and the audio with secret information.
- **Doua. A. Nassar (2003) [10]** Proposed a new approach under the rubric of Wave-

Based Steganography System (WBSS). This system consists of two techniques: Firstly the Wave Differential Technique (WDT). It is used to decrease the limitation and complexity of embedding huge amount of data (text, image) into wave-cover media. The second technique is used to embed and extract wave into wave-cover media.

- **Nedeljko Cvejic (2004) [11]** Presents a novel high bit rate for LSB (least significant bits) audio data hiding method that reduces embedding distortion of the host audio. Using the proposed two-step algorithm, hidden bits are embedded into the higher LSB layers, resulting in increased robustness against noise addition. In addition, listening tests showed that perceptual quality of watermarked audio is higher in the case of the proposed method than the perceptual quality obtained using the standard LSB method.
- **Mohammad Al-Atoom (2009) [12]** Use steganography for embedding data in MP3 files by inserting data between MP3 frames using two way to hide data in MP3 file, before all MP3 frames (BAF) and between MP3 frames (BF) after using RSA (Rivest, Shamel and Aldeman) algorithm to encrypted data.

5. Thesis Outline

In this section, the contents of the remaining chapters of this thesis are briefly reviewed.

Chapter 2: Theoretical Concept

This chapter introduces the basic concept of steganography, MP3, encryption algorithm and SNR measurement.

Chapter 3: Methodology

This chapter presents the design and implementation of proposed methods.

Chapter 4: Experimental Result and Evaluation

This chapter presents the evaluation of results for the methods implemented in this research and comparison with result from previous researches.

Chapter 5: Conclusion and Future Work

This chapter presents the conclusions and suggestion for future works.

2. Theoretical Concept

2.1 Introduction

This chapter describes the structure of the MP3 file which is a subtype of the MPEG format to put a light on the parts that will be used in this research to hide information. The knowledge of each field in the MP3 file structure is essential for the implementation part of this research. It also clarifies the SNR measurement concept that has been used in this research as an evaluation for the techniques used. It covers the theories, and concepts of hiding data, which has been used in this research.

2.2 Information Hiding Techniques

There are many techniques to send a secret message in multimedia files on the form of hide information such as covert channels, steganography, anonymity, and copyright marking [13]. In this research we implement steganography on MP3 files.

2.2.1 Steganography

Steganography is the art of hiding information of sensitivity or peculiarity inside other data so that to be invisible to others. In other words, it means "covered or hidden writing. The objective of steganography is to send a message through some innocuous carrier to a receiver while preventing anyone else from knowing that a message is being sent at all. Computer based stenography allows changes to be made to what are known as digital carriers such as images or sounds. The changes represent the hidden message, but result, if successful, in no discernible change to the carrier. The information may have nothing to do with the carrier sound or image, or it might be information about the carrier such as the author or a digital watermark or fingerprint [11].

To achieve this goal, the original media (cover) is imperceptible modified to embed encrypted messages by using a shared key, and the receiver can extract and decrypt messages from the modified carriers (steganogram). Audio steganographic systems can be categorized into these types [14]:

1. Low-bit encoding substitutes the least significant bit of the information in each sampling point with the binary representation of the secret message.
2. Phase coding replaces the phase of initial segment of the audio and modifies the subsequent segment to preserve the pattern of relation between segments.
3. Spread spectrum encoding embeds data across a large range of the frequency spectrum.

4. Echo data encoding injects data by varying parameters of echo and generates an imperceptible echo of the original signal.

Spread spectrum appeared in 1930's. First patent 1941 for Hedy Lamar (actress) and George Antheil (composer), secret communications system used by the military.

Spread spectrum techniques divide into three types, as follow [14]:

1. Direct Sequence Spread Spectrum (DSSS)

   Data to be transmitted is divided into small pieces and each piece is allocated to a frequency channel across the spectrum.
2. Frequency Hopped Spread Spectrum (FHSS)

   Data carrier frequency is periodically modified (hopped) across a specific range of frequencies (spreading).
3. Time Hopped Spread Spectrum

   Short information burst (chirp) transmitted with pseudorandom pulse duration, or transmitted in a random position.

2.3 MPEG

MPEG (pronounced M-peg), which stands for Moving Picture Coding Experts Group, is the nickname given to a family of international standards used for coding audio-visual information in a digital compressed format. The MPEG family of standards includes MPEG-1 (July 1989), MPEG-2 (July 1991) and MPEG-4 (July 1995), formally known as ISO/IEC-11172, ISO/IEC-13818 and ISO/IEC-14496. MPEG is originally the name given to the group of experts that developed these standards. The MPEG working group (formally known as ISO/IEC JTC1/SC29/WG11) is part of JTC1, the Joint ISO/IEC Technical Committee on Information Technology. The convener of the MPEG group is Leonardo Chiariglione also known as the father of MPEG, who founded the group in January 1988 with the first meeting consisting of about 15 experts on compression technology [15,16].

2.4 MPEG-1

MPEG-1, which describes the compression of audio signals, specifies a family of three audio coding schemes, simply called Layer-I, II and III, with increasing encoder complexity and performance (sound quality per bit-rate).

In general, the complexity for all MPEG-1 codec increases while moving from Layer-I to Layer-III, as follow [15,16]:

- Layer-I possesses the lowest complexity and is specifically targeted to applications where the complexity of the encoder plays an important role.

- Layer-II requires a more complex encoder as well as a slightly more complex decoder. Compared to Layer-I, Layer-II is able to suppress more redundancy in the signal and applies the psychoacoustic model in more efficient way.
- And the third type of MPEG-1 is layer-III (MP3), is once again of an increased complexity and is targeted to applications needing the lowest data rates, by its suppression of the redundant signal and its improved extraction of feebly audible frequencies using its filter. And it is a highly compressed digital audio format. It was standardized in 1991 by the Moving Pictures Expert Group, popularly referred to as MPEG [17].

2.5 MP3

MP3 is an audio compression technique used to store voice and music using different bit-rate varies between 32 kbps up to 320 kbps to achieve the desired quality of sound [18, 14].

2.5.1 Type of MP3 files:

MP3 files can be encoded with three methods [19]:

1. **Constant Bit-rate (CBR):** This is the default encoding mode, and also the most basic. In this mode, the bit-rate will be the same for the whole file. It means that each part of MP3 file will be using the same size of number of bits. The musical passage being a difficult one to encode or an easy one, the encoder will use the same bit-rate, so the quality of MP3 is variable. Complex parts will be of a lower quality than the easiest ones. The main advantage is that the final files size will not change and can be accurately predicted.
2. **Variable bit-rate (VBR):** In this mode, we can choose the desired quality on a scale from 9 (lowest quality/biggest distortion) to 0 (highest quality/lowest distortion). Then encoder tries to maintain the given quality in the whole file by choosing the optimal number of bits to spend for each part of the music. The main advantage is that we are able to specify the quality level that we want to reach, but the inconvenient is that the final file size is totally unpredictable.
3. **Average Bit-rate (ABR):** In this mode, we choose the encoder which will maintain an average bit-rate while using higher bit-rates for the parts of your music that need more bits. The result will be of higher quality than CBR encoding but the average file size will remain predictable, so this mode is highly

recommended over CBR. This encoding mode is similar to what is referred as VBR in AAC or Liquid Audio (2 other compression technologies).

2.6 MP3 file format

MP3 files are composed from sequence data frames; padded with headers, also two main headers called ID3-Tag in the first and end of MP3 file, it consist of some meta-data.

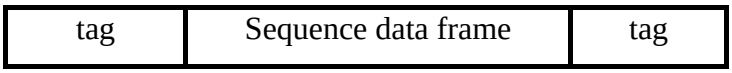


Figure 2.1 MP3 file format

2.6.1 ID3-Tag

There are two types of these tags [20]:

- ID3v2 tag
- ID3v1 tag

ID3v2 Tag is the first field of MP3 file, and it is composed of their own frames which store various bits of information. There is no set size limit on ID3v2 tags so in theory they can grow indefinitely [20].

The ID3v2 tag header, which should be the first information in the file, is 10 bytes as follows:

- ID3v2/file identifier "ID3"
- ID3v2 version 03 00
- ID3v2 flags abc00000
- ID3v2 size 4 of 0xxxxxxx

The first three bytes of the tag are always "ID3" one byte for each letter to indicate that this is an ID3v2 tag, directly followed by the two version bytes. The first byte of ID3v2 version is its major version "03", while the second byte is its revision number "00", each two number is one byte and the numbers in hexadecimal system [18].

ID3v2 flags field is one byte of which currently only three flags is used *a*, *b* or *c* and 5bit = zero, as follow [20]:

a - Un-synchronization

Bit 7 in the 'ID3v2 flags' indicates whether or not un-synchronization is used; a set bit indicates usage.

b - Extended header

The second bit (bit 6) indicates whether or not the header is followed by an extended header.

c - Experimental indicator

The third bit (bit 5) should be used as an 'experimental indicator'. This flag should always be set when the tag is in an experimental stage.

All the other flags should be cleared. If one of these undefined flags are set that might mean that the tag is not readable for a parser that does not know the flags function.

The ID3v2 tag size is encoded with four bytes where the most significant bit (bit 7) is set to zero in every byte, making a total of 28 bits. The zeroed bits are ignored, so a 257 bytes long tag is represented as 00 00 02 01 [20].

Byte	#7	#8	#9	#10
	0xxx xxxx	0xxx xxxx	0xxx xxxx	0xxx xxxx

Figure 2.2 ID3v2 size form

The ID3v2 tag size includes the padding without the header (total tag size - 10) [18].

To calculate ID3v2.3.0 size we look to last 4 bytes of first 10 bytes. The formula equation for ID3v3.3.0 size is:

Get byte number 7, Size = int(byte 7)

Get byte number 8, size = size * 128 + int(byte 8)

Get byte number 9, size = size * 128 + int(byte 10)

ID3v1 tag is the last field of MP3 file. It is composed of their own frames which store various bits of information. The size of ID3v1 tags is fixed and it is occupy 128 bytes or 227 bytes of MP3 file [20].

The ID3v1 tag header is 4 bytes. The 3 bytes is ID3v1 identifier "ID3" [20].

The fourth byte determines the size of ID3v1.

- 227 bytes "+"
- 128 bytes another character

2.6.2 MP3 sequence frames

Between tags there is a sequence of frames. Each frame stores 1152 audio samples and lasts for 26 ms. This means that the frame rate will be around 38 fps for 44100Hz bit sample rate [21].

2.6.3 MPEG (MP3) frame format [22, 23]

A frame consists of five parts: Header, Cyclic Redundancy Checksum (CRC), side information, main data and the ancillary data. In MP3, each frame contains information to produce 1152 output samples. A graphical view of the bit-stream format can be found in Figure 2.3.

This format for all MPEG types, especially for MP3 files.

Header	CRC	Side Information	Main Data	Ancillary Data

Figure 2.3 The frame layout

1. **MPEG Frame Header**

The header field contains information about the format of the frame and it starts with a synchronization word that is used to find the beginning of a frame.

The header information in the bit-stream of is 32 bits (4 bytes)

- **Frame sync:** the first 11 bits of the frame header are always set to 1
- **MPEG Audio Version ID:** 2 bits specify the audio version.

  Consist of four types of versions:

  - 00 - MPEG Version 2.5 (later extension of MPEG 2)
  - 01 – reserved
  - 10 - MPEG Version 2
  - 11 - MPEG Version 1

- **Layer description:** 2 bits specify the layer description.

  Consist of four types of Layers

  - 00 – reserved
  - 01 – Layer-III
  - 10 – Layer-II
  - 11 – Layer-I

- **Protection:** 1 bit.

  - If 0 - Protected by CRC (16bit CRC follows header)
  - Or 1 - Not protected

- **Bit-rate index:** specify the MPEG version and layer which has 16 different values of bit-rate in kbps.

Table 2.1 Bit-rate index

bits	V1,L1	V1,L2	V1,L3	V2,L1	V2, L2 & L3
0000	free	Free	free	free	free
0001	32	32	32	32	8
0010	64	48	40	48	16
0011	96	56	48	56	24
0100	128	64	56	64	32
0101	160	80	64	80	40
0110	192	96	80	96	48
0111	224	112	96	112	56
1000	256	128	112	128	64
1001	288	160	128	144	80
1010	320	192	160	160	96
1011	352	224	192	176	112
1100	384	256	224	192	128
1101	416	320	256	224	144
1110	448	384	320	256	160
1111	bad	Bad	bad	bad	bad

"0000" in table 2.1 means free format. If the correct fixed bit-rate (such files cannot use variable bit-rate) is different than those presented in upper table it must be determined by the application. This may be implemented only for internal purposes since third party applications have no means to find out correct bit-rate. However, this is not impossible to do but demands lot's of efforts.

"1111" is not an allowed value.

- Sampling rate frequency: 2 bits specify the rate frequency for sample

Table 2.2 Sampling rate frequency index

bits	MPEG1	MPEG	MPEG
00	44100 Hz	22050 Hz	11025 Hz
01	48000 Hz	24000 Hz	12000 Hz
10	32000 Hz	16000 Hz	8000 Hz
11	reserv.	reserv.	reserv.

-Padding: bit.

- If 0 - frame is not padded
- Or 1 - frame is padded with one extra slot

Padding is used to exactly fit the bit-rate. As an example: 128kbps 44.1kHz layer-II uses a lot of 418 bytes and some of 417 bytes long frames to get the exact 128k bit-rate. For Layer-I slot is 32 bits long, for Layer-II and Layer-III slot is 8 bits long

In this stage we can define the frame size.

- Private bit. It may be freely used for specific needs of an application

- Channel mode: 2 bits.

- 00 - Stereo
- 01 - Joint stereo (Stereo)
- 10 - Dual channel (2 mono channels)
- 11 - Single channel (Mono)

Dual channel files are made of two independent mono channels. Each one uses exactly half the bit-rate of the file.

- Mode extension: 2 bits. Mode extension is used to join information if no use for stereo effect

- In Layer-III specify the stereo mode, switched on or off
- In Layer-I and II specify the frequency range (bands)

- Copyright: 1 bit.

- If 0 - Audio is not copyrighted
- Or 1 - Audio is copyrighted

i.e. it is illegal to copy the contents if the bit is set.

- Original: 1 bit.

- If 0 - Copy of original media
- Or 1 - Original media

The original bit indicates, if it is set, that the frame is located on its original media.

- Emphases: 2 bits.

- 00 – none
- 01 - 50/15 ms
- 10 – reserved
- 11 - CCIT J.17

The emphasis indication is here to tell the decoder that the file must be de-emphasized, i.e. the decoder must 're-equalize' the sound after a Dolby-like noise suppression. It is rarely used.

2- CRC

Cyclic Redundancy Checksum is an optional feature, and can be omitted. When it is present, the decoder should calculate a CRC on the header and compare it with the CRC in the frame. If they do not match, the decoder should start looking for a new sync-word [22].

This field will only exist if the protection bit in the header is set and makes it possible check the most sensitive data for transmission error

3- Side Information

This field contains information for decoding and processing the main data. The size depends on the encoded channel mode. If the channel mode is a single channel bit-stream the size will be 17 bytes, if not, 32 bytes are allocated. The different parts of the side information are presented in figure 2.4, described in details below.

Main_data_begin	Private_bits	Scfsi	Side_info gr.0	Side_info gr.1

Figure 2.4 Side information form

Table 2.3 gives a clear picture of how the side information is organized both for single and dual channel modes in block type 0, 1 and 3. and table 2.4 presented the organization of side information for block type field equal 2 is in. Lengths of each term mentioned in the tables are indicated in bits and terms involved in the tables 2.3 and 2.4 are explained following the tables [24].

Table 2.3 Fields lengths in side information for block_types 0, 1 and 3

Name	Single channel	Dual channel
main_data_begin	9	9
private_bits	5	3
share	4	4 + 4
Information for first granule:		
part2_3_length	12	12 + 12
big_values	9	9 + 9
global_gain	8	8 + 8
scalefac_compress	4	4 + 4
window_switching	1	1 + 1
For normal blocks:		
table_select	3x5	3x5 + 3x5
region0_count	4	4 + 4
region1_count	3	3 + 3
Subtotal for normal blocks	22	44
preflag	1	2
scalefac_scale	1	2
count1table_select	1	2
Subtotal for first granule	59	118
Subtotal for second granule	59	118
Total number of bits	136	256
Total number of bytes	17	32

Table 2.4 Organization of side information for block_type 2

Name	Single channel	Dual channel
For start, stop and short blocks: block_type	2	2 + 2
mixed_block_flag	1	1 + 1
Table selection for two regions	2*5	2*5 + 2*5
subblock_gain	2*5	3*3 + 3*3
Subtotal for not normal blocks	22	44

The length of each field will be specified in parenthesis together with the fieldname above the actual description. If one length value is written the field size is constant. If two values are specified the first value will be used in mono mode and the second will be used for all other modes, thus these fields are of variable length. All tables below will assume a mono mode. The tables will change depending on mode since separate values are needed for each channel [22, 24].

- Main data begin (9 bits)

  It is a pointer that points to the beginning of the main data. The variable has nine bits and specifies the location of the main data as a negative offset (jumping backwards) in bytes from the first byte of the audio sync word. The number of bytes of the header and side information are not taken into account while calculating the location of the main data. This is called bit reservoir technique and it allows the encoder to use some extra bits while encoding a difficult frame. Since it is nine bits long, it can point up to $2^9 - 1 = 511$ bytes in front of the header. If the value of main data begin is zero, then the main data follows immediately the side information

- Private bits (5 bits, 3 bits)

  Bits for private use, these will not be used in the future by ISO (International Organization for Standardization).

- Scfsi (4 bits, 8 bits)

  The Scale Factor Selection Information determines whether the same scale

factors are transferred for both granules or not. Here the scale factor bands are divided into 4 groups according to table 2.5.

Table 2.5 Scale factor groups

group	scalefactor bands
0	0,1,2,3,4,5
1	6,7,8,9,10
2	11,12,13,14,15
3	16,17,18,19,20

4 bits per channel are transmitted, one for each scale factor band. If a bit belonging to a scale factor band is zero the scale factors for that particular band are transmitted for each granule.

- Side information granule 0 and side information granule 0

  Side information for each granule:

  The last two parts of a frame have the same anatomy and consists of several subparts as shown in table 2.6. These two parts store particular information for each granule respectively.

Table 2.6 Fields for side information for each granule

Part2_3_length	Big_values	Global_gain
Scalefac_compress	Windows_switchin flag	Block_type
Mixed_block_flag	Table_select	Subblock gain
Region0_count	Region1_count	Preflag
Scale_facscale	Count1 table select	

  - Part2_3_length (12 bits, 14 bits)

    States the number of bits allocated in the main data part of the frame for scale factors (part2) and Huffman encoded data (part3). 12 bits will be used in a single channel mode whereas in stereo modes the double is needed. This field can be used to calculate the location of the next granule and the ancillary information (if used) by summation the part2 3 length in each granule to calculate the main data size. And the residual bytes in end main data called ancillary data.

– Big _values (9 bits, 18 bits)

  Contains pairs of values enhance the performance of the Huffman encoder.

– Global_gain (8 bits, 16 bits)

  Specifies the quantization step size, this is needed in the requantization block of the decoder.

– Scalefac_compress (4 bits, 8 bits)

  Determines the number of bits used for the transmission of scale factors. A granule can be divided into 12 or 21 scalefactor bands. If long windows are used (block_type = {0,1,3}) the granule will be partitioned into 21 scalefactor bands. Using short windows (block_type = 2) will partition the granule into 12 scalefactor bands. The scale factors are then further divided into two groups, 0-10, 11-20 for long windows and 0-6, 7-11 for short windows.

– Windows_switching flag (1 bit, 2bits)

  If windows switching flag is set, mixed block flag and subblock gain are used. And all remaining values not being in region0 are contained in region1 thus region2 is not used. If window switch in flag is not set then the value of block type is zero.

– Block_type (2 bits, 4 bits)

  This field is only used when windows switching flag is set and indicates the type of window used for the particular granule as follows in table 2.7, (all values but 3 are long windows).

Table 2.7 Block type definition

block_type	window type
00	forbidden
01	start
10	3 short windows
11	end

  The value 00 is forbidden since block type is only used when other than normal windows are used.

– Mixed_block flag (1 bit, 2 bits)s

  This field is only used when windows switching flag is set.

The mixed_block flag indicates that different types of windows are used in the lower and higher frequencies. If mixed_block flag is set the two lowest subbands are transformed using a normal window and the remaining 30 subbands are transformed using the window specified by the block type variable.

- Table_select ((10 bits, 20 bits) or (15 bits, 30 bits))

  Table selection indicates which Huffman table to be used when decoding the big_values as specifies at table select for normal block. This field will only be used when the window switching flag is not set. It indicates that other than normal block is in use.

- Subblock gain (9 bits, 18 bits)

  indicates the gain offset at quantization factor of 4 from global_gain for each short block. The values of the subblock have to be divided by 4^sublock_gain (window) in the quantization block of the decoder. This field is used when window_switiching_flag is set and block_type is 2.

- Region0 count (4 bits, 8 bits) and Region1 count (3 bits, 6 bits)

  Big_values that were mentioned earlier is further subdivided into three regions namely, region0, region1 and region2. This partition of the spectrum is used to enhance the performance of Huffman coder while also attaining better error robustness and better coding efficiency. The values region0_count and region1_count are used to indicate the boundaries of the regions. The region boundaries are aligned with the partitioning of the spectrum into scale factor bands. Region0_count and region1_count contains one less than the number of scale_factor bands in the regions 0 and 1 respectively.

- Preflag (1 bit, 2 bits)

  This field is never used for short blocks (i.e, block_type = 2). If it is set, then some values are added to the scale factors. This is equivalent to multiplication of the re-quantized scale factors with the values which also means additional high frequency amplification of the quantized values.

- Scalefac scale (1 bit, 2 bits)

  Indicates the quantization step size applied to scale factor.

– Count1 table select (1 bit, 2 bits)

Count1 table select Indicates which Huffman code tables of ISO standard to be used for count1 region.

If all of side bytes are one's, the frame is silent and there will not be any further data in the frame. Therefore the ancillary data begins immediately after these 32 bytes of side information (unless the following frame's main data begins there instead**).**

4- Main Data

This is the important part of MPEG frame contains the audio sound data.

Audio Sound data of main data consist of [21]

- Scalefactors
- Huffman coded are the sound data

5- Ancillary data

The ancillary data is optional and the number of bits available is not explicitly given. The ancillary data is located after the Huffman code bits and ranges to where the next frame's main-data-begin points to [22].

2.7 How to calculate frame length

First, let's distinguish two terms frame size and frame length. Frame size is the number of samples contained in a frame. It is constant and always 384 samples for Layer-I and 1152 samples for Layer-II and Layer-III. Frame length is length of a frame when compressed. It is calculated in slots. One slot is 4 bytes long for Layer-I, and one byte long for Layer-II and Layer-III. When we are reading MPEG file we must calculate this to be able to find each consecutive frame [21].

Note: Frame length may change from frame to frame due to padding or bit-rate switching.

Read the Bit-Rate bits, Sample-Rate bits and Padding bit of the frame header and find the Bit-Rate and Sample-Rate values from table 2.1 and 2.2.

For Layer-I files use this mathematical equation [22]:

$$\text{FrameLengthInBytes} = (12 \times \text{Bit-Rate} / \text{Sample-Rate} + \text{Padding}) \times 4 \quad \ldots(2.1)$$

For Layer-II or layer III files we use this mathematical equation [22]:

$$\text{FrameLengthInBytes} = 144 \times \text{Bit-Rate} / \text{Sample-Rate} + \text{Padding} \quad \ldots\ldots\ldots\ldots..(2.2)$$

Example:

Layer-III, Bit-Rate = 128 kbps, Sample-Rate = 44100 Hz, Padding = 0

==> FrameSize = 417 bytes

2.8 Assistance Techniques

In this research we add encryption stage to increase the security. For evaluation purposes we build a software module to calculate the noise added to the original file due to steganography process.

2.8.1 Encryption and Decryption Algorithms

As an addition to the Steganography, Encryption algorithm was used in this research to increase safety on the transmitted information.

- **Why we need encryption and decryption?**

Sending sensitive messages and files across the Multimedia is very dangerous as most information are transmitted in an unsecured form. And hiding process for the confidential information in multimedia file may be not enough to save the information form tampering. In this case, the sending sensitive information over the Multimedia should be encrypted at the outset [25].

By adding encryption stage to our process, we can send the confidentially information and files safely [25].

- **Encryption**

Encryption is the process of transforming information (plain text) using an algorithm to make it unreadable (cipher text) to anyone except those possessing special knowledge, usually referred to as a key [25].

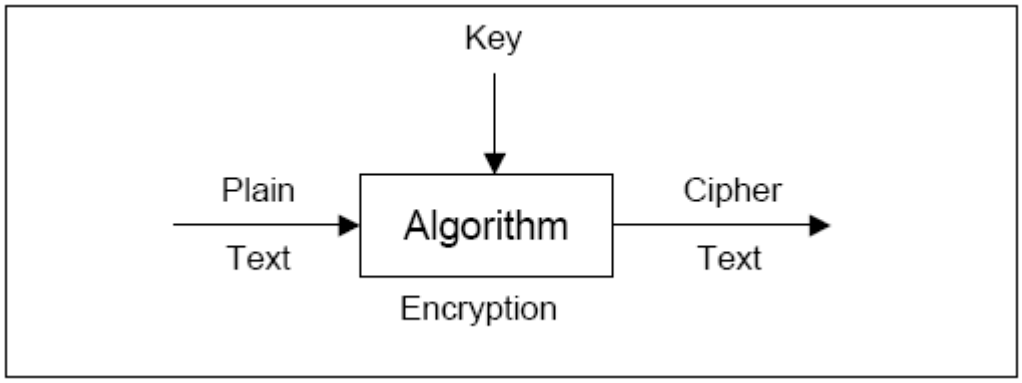


Figure 2.5 Basic encryption concepts

This changing on the information is techniques, called Cryptography.

Cryptography techniques can be used to scramble a message so that if it is discovered it cannot be read. If a cryptographic message is discovered it is generally known to be

a piece of hidden information (anyone intercepting it will be suspicious) but it is scrambled so that it is difficult or impossible to understand and de-code [13].

- **Decryption**

Decryption is reverse process of encryption. It is transforming information using an algorithm to get the original information by receiver.

2.8.1.1 Encryption Systems Categories [25]

- **Symmetric encryption.**
- **Asymmetric encryption.**

Symmetric encryption: In a symmetric encryption system, both the sender and receiver must possess the same key. The sender encrypts the message using the key and the receiver decrypts the cipher-text message using the same secret key.

Two classes of symmetric-key encryption algorithms:

Block: operates on plaintext input in blocks (usually 64 bits at a time) of bits to produce the cipher-text output; uses the key value to determine how the transformation algorithm is applied.

Stream: operates on plaintext input one bit at a time, often using a key-stream generator to produce a series of bits which are XOR'd (exclusive or) with the plaintext input.

- **Examples of Block**:

– **DES (**Data Encryption Standard**)** is a block cipher with a 64 bit block size.

– **AES** (Advanced Encryption Standard**)** is a block cipher with a 128 bit block size.

– **RSA and Diffie-Hellman** are block ciphers with variable block sizes.

- **Examples of Stream:**

– **A5**, the algorithm used to encrypt GSM (Global System for Mobile communication).

– **OTP** (one-time pad) are also stream ciphers.

- **Advantages of symmetric-key:**

– Have high rates of data throughput.

– Keys for symmetric-key ciphers are relatively short.

– Symmetric-key ciphers can be composed to produce stronger ciphers.

- **Disadvantages of symmetric-key:**

– In a two-party communication, the key must remain secret at both ends.

– In a large network, there are many key pairs to be managed.
– Digital signature mechanisms arising from symmetric-key encryption.

In this research, stream class of symmetric key encryption algorithm was used in a software module to increase the protection of the transferred information. Each byte of data was encrypted using XOR gate with three different keys.

Asymmetric encryption: Known as "Public key" encryption, each entity participating in the communication uses mathematical algorithms implemented in a software program to generate a "public key" and a "private key" which are related via the mathematical formula. The private key must be kept secret and is never disclosed; this is a requirement for the security system to function.

The public key, however, is intended to be freely distributed.

- **Public key cryptography characteristics:**

– Something encrypted with the public key can only be decrypted with the private key.
– Something encrypted with the private key can only be decrypted with the public key.

- **Advantages of Public-key**:

– Only the private key must be kept secret.
– Depending on the mode of usage, a private key/public key pair may remain unchanged for considerable periods of time.
– Many public-key schemes yield relatively efficient digital signature mechanisms.

- **Disadvantages of Public-key:**

– Slower than the best known symmetric-key schemes.
– Key sizes are typically much larger.
– No public-key scheme has been proven to be secure.

2.8.2 Audio Noise Measurement

Other technique used for this research in the evaluation results in chapter five.

In this research the noise resulted from the steganography techniques used should not be noticed by human ear, howfever measuring the noise should use technical methods that provides suitable values for this problem.

- **How do we measure noise?**

There are two main techniques used for measuring noise over audio signals. Method one is convenient to linear scale (sound pressure, μPa (Micro Pascal)) and the second method is convenient to log scale (sound pressure level, dB (decibel)) [24].

The intensity of sound is measured in dB, which signifies the force of sound waves against the ear.

Decibel is used to measure sound level, but it is also widely used in electronics, signals and communication. The db is a logarithmic unit used to describe a ratio. The ratio may be power, sound pressure, voltage or intensity or several other things [24].

To give you an idea of decibel measurements, here are approximate decibel levels for some everyday sounds [24].

Sound	Intensity (dB)
threshold of hearing	0
Whisper	30
hum of fridge	40 - 50
normal speech	50 - 60
busy car traffic	70
outboard motor	80
Jackhammer	110
rock concert	120
threshold of pain	130
Jet taking off	140

There is Six popular specifications for quantifying analog to digital converter dynamic performance to measure the noise between two sounds; SINAD (Signal-to-Noise-And-Distortion ratio), ENOB (Effective Number Of Bits), SNR (Signal-to-Noise Ratio), THD (Total Harmonic Distortion), THD + N (Total Harmonic Distortion plus Noise), and SFDR (Spurious Free Dynamic Range). In this research we used SNR measurement to measure the noise on sounds.

2.8.2.1 SNR

Signal-to-noise ratio (often abbreviated SNR or S/N) is a measure used in science and engineering to quantify how much a signal has been corrupted by noise.

In the most general case, SNR is expressed as the ratio of **rms** (root mean square) Signal level (S_{rms}, original signal) and to the rms Noise (N_{rms}, difference between the

original signal and the signal with noise after added embedding process), (SNR = S_{rms} / N_{rms}). Because many signals have a very wide dynamic range, SNRs are often expressed using the logarithmic decibel scale. In pascal, the SNR is defined as mathematical equation 2.3 [26, 27, 28, 29]:

$$SNR_{\mu pa} = 10 \log_{10} (S_{rms} / N_{rms}) \quad \text{.....(2.3)}$$

And in decibels, the SNR is defined as mathematical equation 4:

$$SNR_{db} = 20 \log_{10} (S_{rms} / N_{rms}) \quad \text{.....(2.4)}$$

The SNR in decibels is 20 times the base-10 logarithm of the amplitude ratio, or 10 times the logarithm of the power ratio [26, 27, 28, 29].

3. Methodology

3.1 Introduction

This chapter discusses the methodology of hiding encrypted data in MP3 file using a software module for encrypting and hiding data in MP3 files, encrypting algorithm will be used to encrypt information before hiding process without having any effect on the size of the MP3 file, making this process secure enough and not noticeable by some who hear or see MP3 data. And also discusses the methodology for restoring the encrypted hidden data in normal form without any corrupted in the hidden data.

The hiding process will be implemented in two major methods:

1. **Replacing**: replacing one of MP3 frames in different part of the frame, like in main data, in silent frame and also in full frame, by the desired data.

   In this method we can replace one of the MP3 audio frames by a (part) chunk of the encrypted data.

2. **Hiding**: hiding data in the Ancillary field (HIA) in MP3 frame.

   In this method the data will hide within the ancillary field if there is found in the frames of MP3 file.

3.2 Replacing method

In replacing method we can embed encrypted data in three parts as follow:

1. Main data method (RMD) replacing the data in main data field by encrypted data, without any other change on the frame.
2. Silent frame method (RMD-s) this method is similar to RMD method by replacing the data in main data field by encrypted data and with changing in inside information field order to turn the frame type from original frame to silent frame.
3. Full frame method (RAF) all bytes in the frame replaced by the encrypted data except 4 bytes for header field of MP3 frame.

Each method of replacing methods used two sub-methods to hide data:

1. Fixed allocation for the frame number.

   The encrypted data replaced in a specific location, for example, frame number x in each second (x: frame location that used for hiding information).

2. Dynamic allocation for the frame number.

   Replace the encrypted data in one of the frames each time, the location will be specified depending on an algorithm similar to random number generators as in

mathematical equation 3.5. We predefined rand pattern generated at embedding process. The same pattern is used in extraction. This pattern is generated uniformed random generator with a specific seed.

This process will add more security to the proposed methods, by a chunk of the encrypted data each second. This method is stronger than fixed allocation method since it can be covered in different frame allocation and it can't be discovered easily.

Hiding location = hiding location x RND() +hiding location / 2 ….(3.5)

3.2.1 Process phases:

Figure 3.1 shows the embedding data processing in MP3 files through the following phases; Examination, Encrypting and Embedding. In the end process we get the encrypted audio file such as the original one.

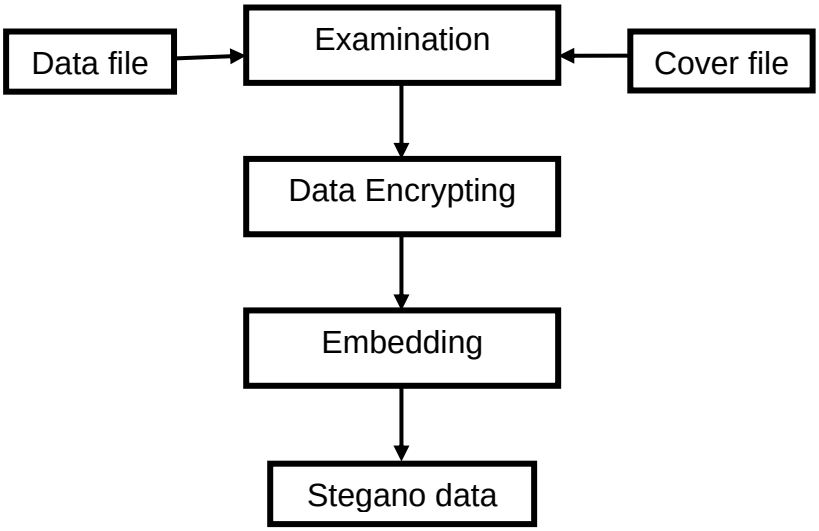


Figure 3.1 Process phase for embedding data.

Step 1: Examination

- Firstly select one of the methods for hiding encrypted data in MP3 frame:

  1.RMD in main data field of MP3 frame each x time.

  2.RMD-s in main data field for silent frame each x time.

  3.RAF in full frame each x time.

Also we must determine the hiding allocation.

– In fixed allocation for the frame number by selecting the frame number.

Or

– In dynamic allocation.

- Analyze the MP3 file depending on the methods chosen in the previous steps, that specifies the needed of cover size of MP3 file size that can be used to hide encrypted data, which also depends on the cover specifications (size of cover file, duration, Bit-rate, Sample-rate, and other specification) in specified size of MP3 file, if the cover (MP3 file) enough to hide the information. Also the analyzing process checks the file type if that MPEG-version1-layer3 file or not and we will talk about the analyzing process in detail in step 3.

Step2: Encryption

- **Header building and Encryption stage:**

A software module will be used to build the information header which contains data file. This header contains checksum of 4 bytes for data file that will be hidden and used it in checking validity of data file in extraction stage.

These information will be encrypted at the begging of data file and used in the extraction and checking stage later.

Symmetric encryption system used to encrypt data by using three different keys. The encryption process has not effect on the actual data size and it keeping the same size for data files after encryption process. All encrypted data will store in a new file (encrypted file).

Step 3: Embedding

In the allocated frame, a software module replaces the data in the proposed locations (fields) depending on the method of hiding used. The anatomy of the MP3 file and its headers in analyze stage can specify where are the targeted locations that will be used for hiding data, and the size of covered location. The encrypted data will split into data chunks to distribute on the hiding locations.

- **Analyze of MP3 file stage:**

We need to analyze the MP3 file again to determine the hiding location in the MP3 frame which is needed to determine the chunks size.

The first part that will be analyzed is ID3v2 tag in first MP3 file which contains:

– ID3v2, consist of 3 bytes (I, d, 3) which identify the MP3 file.

– ID3v2 version.

– ID3v2 size.

– Other parts contain information about MP3 file.

Specify the size of ID3v2 by reading 4 bytes from byte seventh to byte tenth in bit form in the first of MP3 file, where the most significant bit (bit 7) is set to zero in every byte and convert it to decimal numbers.

Using the formula equation for bytes form we specify ID3v2 size:

Get byte number 7, Size = int(byte 7)

Get byte number 8, size = size x 128 + int(byte 8)

Get byte number 9, size = size x 128 + int(byte 9)

Get byte number 10, size = size x 128 + int(byte 10)

After calculating ID3V2 tag size we can reach to the first frame location for MP3 file. The first 32 bits (4 bytes) of this frame is the header information, which distinguish the MP3 file type, as follows:

- The first 11bits always set to 1.
- 2bits after 11 bits specify the MPEG version (version 1).
  Value of MPEG Version 1 type is 11
- 2bits after that to specify the layers description (layer-III).
  Value of MPEG Layer-III type is 01
  These three points specify MPEG type (MP3).
- 1bit for protection to specify CRC size.
  If protection equal zero, CRC size equal 16 bit.
- 4bits, table 3.1 specify bit-rate (kbps) value as in table 3.1.

Table 3.1 Bit-rate index

Bits	V1,L3	Bits	V1,L3
0000	Free	1000	112
0001	32	1001	128
0010	40	1010	160
0011	48	1011	192
0100	56	1100	224
0101	64	1101	256
0110	80	1110	320
0111	96	1111	Bad

- 2bits to specify rate frequency for sample, each sample of frames has different frequency.

Table 3.2 Sampling rate frequency

bits	MPEG1
00	44100 Hz
01	48000 Hz
10	32000 Hz
11	reserv.

– 1bit for padding with exactly slot using to exactly fit the bit-rate.

If MPEG Layer-III slot is 8 bits long. At this point we can specify and calculate frame size.

As previous mention, to calculate frame length in chapter two we used mathematical equation 2.2 to get the size of MPEG version 1 layer 3 frame:

$$\text{FrameLengthInBytes} = 144 \times \frac{\text{BitRate}}{\text{SampleRate}} + \text{Padding}$$

In general there are many of MP3 files that spreading between people and each one of these file have different specifications where the value of bit-rate, sampling rate frequency and padding bit are different in MP3 files. These specifications determine the frames size in MP3 file; therefore the researcher present all original MP3 frame sizes and the covered sizes of MP3 frame that can be used in the proposed methods to hide information. The covered size of MP3 frame size depends on some field size (side information fields of 32bits or 17bits and CRC fields of 1bit) in that frame as we mentioned earlier in chapter two.

The tables present only the size of one frame, the total size of all frames depending on MP3 file size and duration time for that file.

Table 3.3 presents all original MP3 (MPEG version1 layer3) frame sizes depending on bit rate (kbps), sampling rate frequency (Hz) and padding bit by adding one bit for frame if used. The maximum of frame size for MP3 file is 1441 bytes and the minimum of frame size is 96 bytes.

Table 3.3 Original frames sizes for MPEG (version 1, layer 3).

	Sampling rate frequency					
	44.1 Hz	48 Hz	32 Hz	44.1 Hz	48 Hz	32 Hz
Bit-rate	Size without padding			Size with padding		
32 kbps	104	96	144	105	97	145
40 kbps	130	120	180	131	121	181
48 kbps	156	144	216	157	145	217
56 kbps	182	168	252	183	169	253
64 kbps	208	192	288	209	193	289
80 kbps	261	240	360	262	241	361
96 kbps	313	288	432	314	289	433
112 kbps	365	336	504	366	337	505
128 kbps	417	384	576	418	385	577
160 kbps	522	480	720	523	481	721
192 kbps	626	576	864	627	577	865
224 kbps	731	672	1008	732	673	1009
256 kbps	835	768	1152	836	769	1153
320 kbps	1044	960	1440	1045	961	1441
		min:	96	max:	1441	

Table 3.4 presents all the field of MP3 (MPEG version1 layer3) frame sizes that can be used to hide information depending on bit rate, sampling rate frequency and padding bit for frame without 32 bytes (side information field size). The maximum of frame size for MP3 file after removing side information (32 bytes) is 1405 bytes and the minimum of frame size is 60 bytes.

Table 3.4 Frames sizes for MPEG (v1, l3) without side information (32B).

	Sampling rate frequency					
	44.1 Hz	48 Hz	32 Hz	44.1 Hz	48 Hz	32 Hz
Bitrate	Size without padding			Size with padding		
32 kbps	68	60	108	69	61	109
40 kbps	94	84	144	95	85	145
48 kbps	120	108	180	121	109	181
56 kbps	146	132	216	147	133	217
64 kbps	172	156	252	173	157	253
80 kbps	225	204	324	226	205	325
96 kbps	277	252	396	278	253	397
112 kbps	329	300	468	330	301	469
128 kbps	381	348	540	382	349	541
160 kbps	486	444	684	487	445	685
192 kbps	590	540	828	591	541	829
224 kbps	695	636	972	696	637	973
256 kbps	799	732	1116	800	733	1117
320 kbps	1008	924	1404	1009	925	1405
		min	60	max	1405	

Table 3.5 presents all the field of MP3 (MPEG version1 layer3) frame sizes that can be used to hide information depending on bit rate, sampling rate frequency and padding bit for frame without 17 byte (side information field size). The maximum of frame size for MP3 file after removing side information (17 bytes) is 1424 bytes and the minimum of frame size is 79 bytes.

Table 3.5 Frames sizes for MPEG (version 1, layer 3) without side information.

	Sampling rate frequency					
	44.1 Hz	48 Hz	32 Hz	44.1 Hz	48 Hz	32 Hz
Bitrate	Size without padding			Size with padding		
32 kbps	87	79	127	88	80	128
40 kbps	113	103	163	114	104	164
48 kbps	139	127	199	140	128	200
56 kbps	165	151	235	166	152	236
64 kbps	191	175	271	192	176	272
80 kbps	244	223	343	245	224	344
96 kbps	296	271	415	297	272	416
112 kbps	348	319	487	349	320	488
128 kbps	400	367	559	401	368	560
160 kbps	505	463	703	506	464	704
192 kbps	609	559	847	610	560	848
224 kbps	714	655	991	715	656	992
256 kbps	818	751	1135	819	752	1136
320 kbps	1027	943	1423	1028	944	1424
		min	79	max	1424	

In the last two tables, we can remove 2 bytes of frame size that reserved for CRC field if the frame of the protected frames type.

After finding padding bit value we continue to analyze the next 2bit to specify the channel mode used to specify size of side information field.

00 - Stereo

01 - Joint stereo (Stereo)

10 - Dual channel (2 mono channels)

11 - Single channel (Mono)

The side information field size will be 32 byte in first three types of channel mode and it will be 17 bytes in last type.

The other bits in the frame header are not used in this analyzing.

- **Splitting stage:**

A software module split the encrypted file into data chunks to hide it in MP3 frames. The chunks size depending on the MP3 frames specifications and the method of hiding used. A software module in this stage uses three types to hide data:

1. Main data method (RMD):

Data chunks embed in main data field of MP3 frame.

The hiding data embed in main data part by replacing each main data after side information as in figure 3.2.

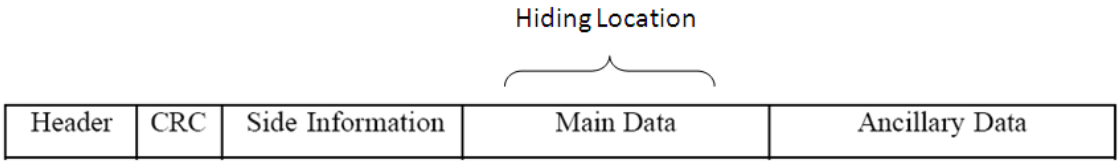


Figure 3.2 Hiding locations for RMD method

This way repeats each x number of hiding location until last frame or until the last byte in encrypted file.

2. Silent frame method (RMD-s):

Data chunks embedded in main data field of MP3 frame.

This way is similar to previous way but with difference in side information values. The side information bytes changed to one's to turn the original frame to silent frame. The encrypted data embed in main data part by replacing each main data after side information as in figure 3.3.

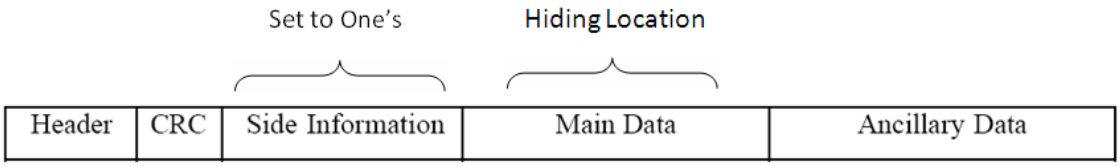


Figure 3.3 Hiding locations for RMD-s method

This way repeats each x number of hiding location until last frame or until the last byte in encrypted file.

Full frame method (RAF):

Data chunks embed in all parts of MP3 frame except the header field.

The changing occurs for all frame bytes except the header field as in figure 3.4.

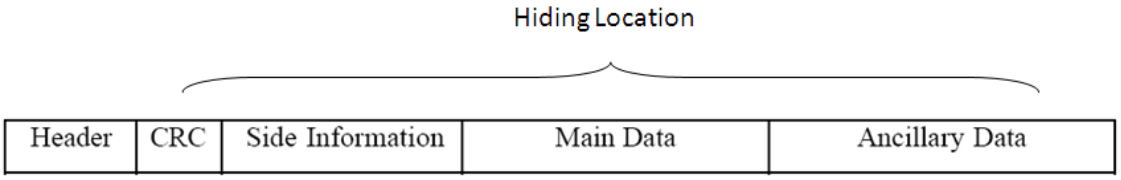


Figure 3.4 Hiding locations for RAF method

This way repeats each x number of hiding location until last frame or until the last byte in encrypted file.

3.3 Hiding method

This method uses the ancillary field in each frame to hide the encrypted data chunks. But as we mentioned earlier in chapter two, the ancillary field is optional field, not for all frames in MP3 files and the number of bits available is not explicitly given, because of these reasons we hided the data in last byte of each frame to simulate the ancillary field for MP3 frame.

By this way, the cover size that can be used to hide data will be smaller than previous used methods in this research.

3.3.1 Process phases:

Hiding data process in MP3 files for this method same as hiding data process for replacing method (as shown in figure 3.1) but it different in step 3; Embedding.

Step1: Examination

Analyzing the MP3 file in this method specifies the needed of cover size of MP3 file size that can be used to hide encrypted data, which depends on the number of frames in MP3 file**.**

Step 2: Encryption

We need to encrypt the information as replacing method (in step 2) to increase on security data.

Step 3: Embedding

Software module will replace the last byte of all frames by the encrypted data. The anatomy of the MP3 file and its headers in analyze stage can specify where are the targeted locations that will be used for hiding data, and the size of hidden data location.

- **Analyze of MP3 file stage:**

We need some steps to analyze MP3 file as in replacing method (in step 3). The ID3v2 analyze and the first 4 bytes of each frame header to specify the size of each frame. By specifying the frame size, we can specify the end of the frame that will be used for hiding encrypted data.

- **Splitting stage:**

A software module split the encrypted file into data chunks to hide it in MP3 frames. The chunk size depends on the total of ancillary field sizes in MP3 frames (total size for the last byte in each frame).

A software module embeds the data chunks in last byte of the frame after main data as in figure 3.5.

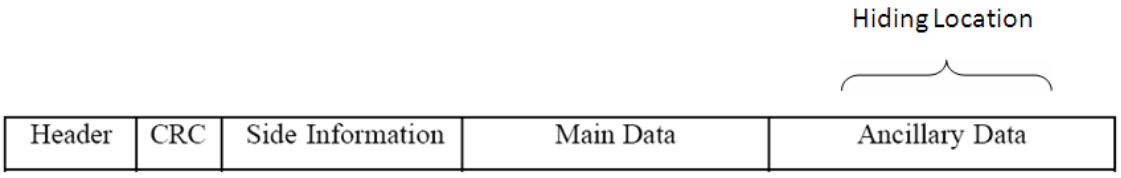


Figure 3.5 Hiding locations for SA method

3.4 Extraction and Decryption

Extraction stage:

Figure 3.6 describes the extract process for the encrypted data from MP3 file to get the hiding information. A software module will extract the hidden data depending on the selected method for data insertion method.

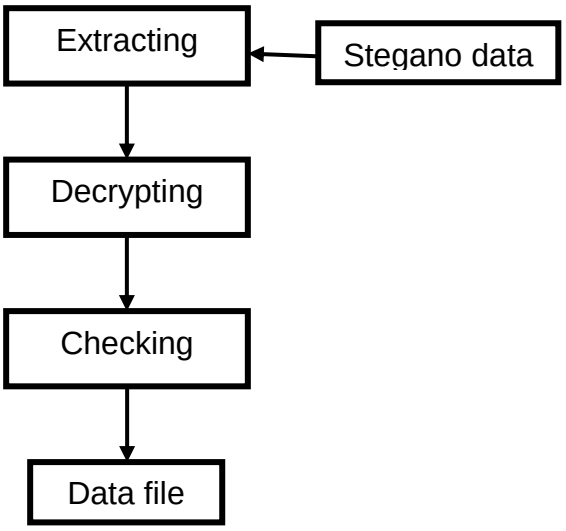


Figure 3.6 Extraction data process.

In RMD, RMD-s, and RAF method, a software module analyzes the frames depending on the hiding methods and the hiding location then extracts the data chunks.

In HIA method, a software module analyzes the frames to get last bytes of each frame and extract the data chunks.

Decrypted stage:

Depending on the stream class of symmetric key encryption algorithm a software module decrypts all data chunks from extracting process to get the original file, but after removing the first bytes in the beginning of the file which consist on the header.

Checking validity stage:

Check validity: integrity information is not undetectably altered or destroyed by unauthorized person or process. Its mean the extracted data correct or have been voluntary to active attacked.

This process uses header to check the validity of data file that will be hidden. For more protection the header built before encrypted process where encrypted with data.

The header built at the beginning of file, the header contains data size of 4byte, contains 4bytes for check sum. After checking stage, we get the original form of hidden data.

4. Experiment results and Evaluation:

4.1 Introduction:

In this chapter we present testing results for hiding process in multimedia files using three measurement forms for each method as follow:

- Hiding space size for each multimedia file.
- Time that need for hiding process.
- Measuring signal to noise ratio value (SNR measurement).

These measurement forms were used on the four proposed methods.

1. Hiding data in the Ancillary field of frames in MP3 (HIA).
2. Replacing data in main data field of MP3 frames (RMD).
3. Replacing data in silent frame of MP3 file (RMD-s).
4. Replacing data in full frame of MP3 file (RAF).

The last three methods were applied in two sub-methods; fixed and dynamic locations of MP3 frames.

4.2 Data Collection:

The methods discussed above have two main elements, the **Multimedia files MPEG files of version 1 layer 3 (MP3)** of different sizes that will be used to hide information which will be sent to receiver across different medias on MP3 form, and the **Confidential information** that will be hidden in multimedia files. This information could be any data file type. The only restriction of this information is the size which should be suitable the maximum size that the MP3 cover file can carry.

The maximum size of information that can be hidden depends on duration and other properties of cover file (MP3).

4.3 System specifications

The research was implemented on Pentium 4 computer core 2 due 2.2GHz, with 2 GB memory and 160GB hard disk.

The operating system was Windows XP Professional Service pack 3 (Works on other operating systems).

Visual studio 6.0 (VB 6.0 & VC++ 6.0) where used as implementation tools in this research. Windows media player Version 11.0.5721.5268 used to play MP3 files for testing purity sound.

4.4 Evaluation result

We tested many different files of MP3 files vary in size and content for all proposed methods in this research. And we present six files of them in the evaluation and their properties as follows:

Table 4.1 Cover files properties

Cover files	Size in bytes	Bit-rate	channel	Audio sample rate	Duration time
File 1	863,696	96 kbps	stereo	32 KHz	71 s
File 2	2,286,030	96 kbps	2 stereo	44 KHz	190 s
File 3	5,030,851	129 kbps	2 stereo	44 KHz	209 s
File 4	8,335,365	320 kbps	2 stereo	44 KHz	208 s
File 5	10,951,629	320 kbps	2 stereo	44 KHz	303 s
File 6	14,814,862	320 kbps	2 stereo	44 KHz	370 s

And we used other six files of MP3 files vary in genre (blues, classical, country, folk and popular).

We used six files vary in type, size and content for hiding process in MP3 files, as follows:

First file

Size in bytes: 2,506 Type of file: Text Document (txt)

Second file

Size in bytes: 9,693 Type of file: GIF Image (gif)

Third file

Size in bytes: 74,022 Type of file: Text Document (txt)

Fourth file

Size in bytes: 160,256 Type of file: Microsoft Word Document (doc)

Five file

Size in bytes: 257,929 Type of file: Adobe Acrobat Document (pdf)

Sixth file

Size in bytes: 375,563 Type of file: Adobe Acrobat Document (pdf)

We can use any type of data file to use it in hiding process.

4.4.1 Hiding and Replacing methods

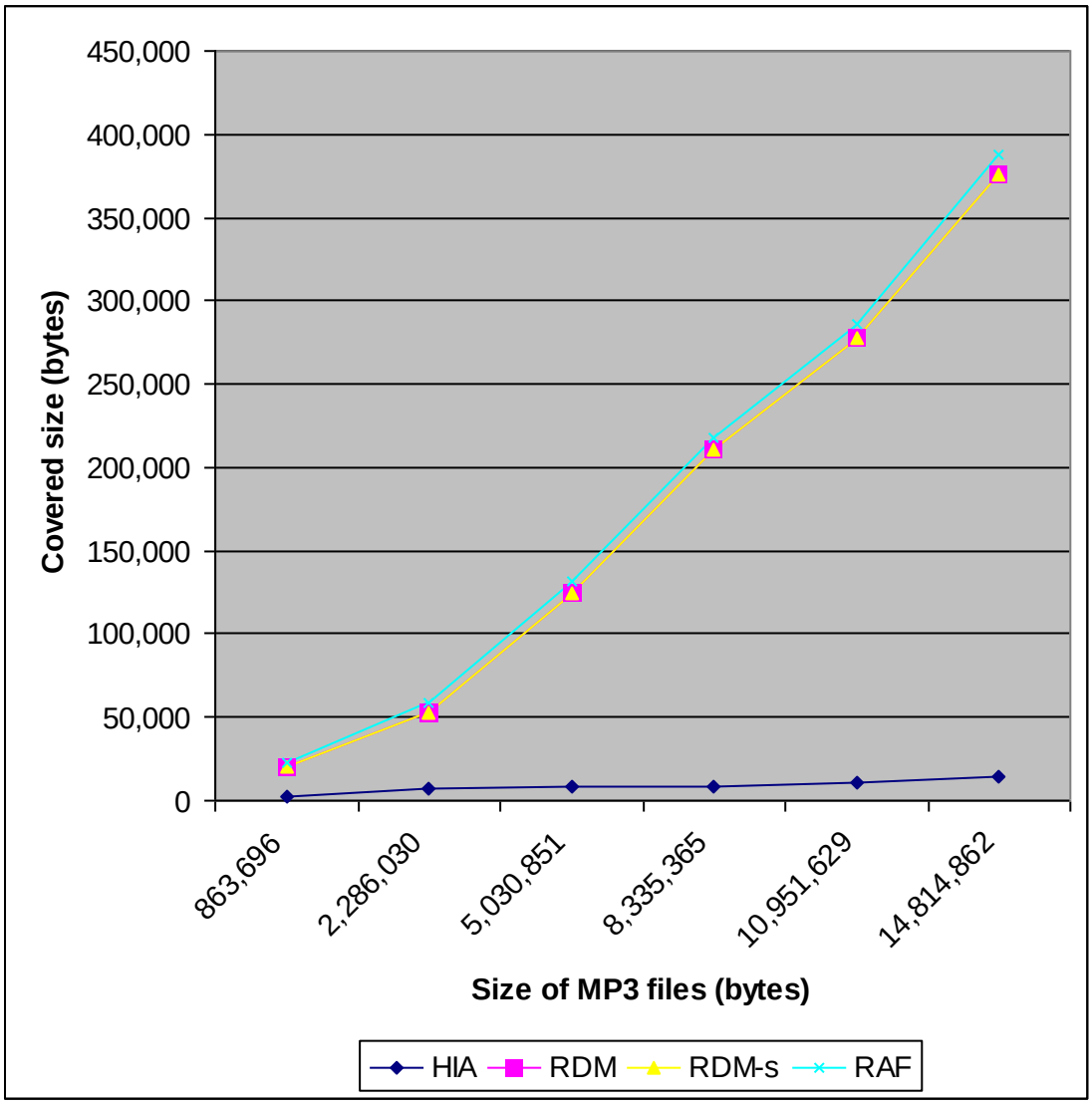
Table 4.2 presents the maximum of covered sizes in MP3 files that can be used to hide the information. The results for RMD, RMD-s and RAF methods were tested in **fixed location** of MP3 frame in each 38 frame; the value of fixed location can change with other value. The results of HIA method depend on the ancillary field if there were existed in the frame or no.

Note: Each values for tables in bytes.

Table 4.2 Maximum of embedding in fixed location.

		Methods			
MP3 files	Size of MP3 file	HIA	RDM	RDM-s	RAF
File 1	863,696	1,980	20,574	20,574	22,238
File 2	2,286,030	7,271	52,980	52,980	59,092
File 3	5,030,851	8,002	124,571	124,571	131,423
File 4	8,335,365	7,992	211,035	211,037	217,755
File 5	10,951,629	10,498	277,563	277,563	286,395
File 6	14,814,862	14,156	376,302	376,302	388,238

Graph 4.1 present an analysis of the result for table 4.2.



Graph 4.1 Maximum of embedding in fixed location.

As shown in table 4.2 the covered sizes in MP3 files that were used in RMD, RMD-s and RAF methods is about 2.4% of total size for MP3 files, which it is a good ratio and enough to hide some of data. And the covered size that was used in HIA method is much less than of the other methods (about 0.09%).

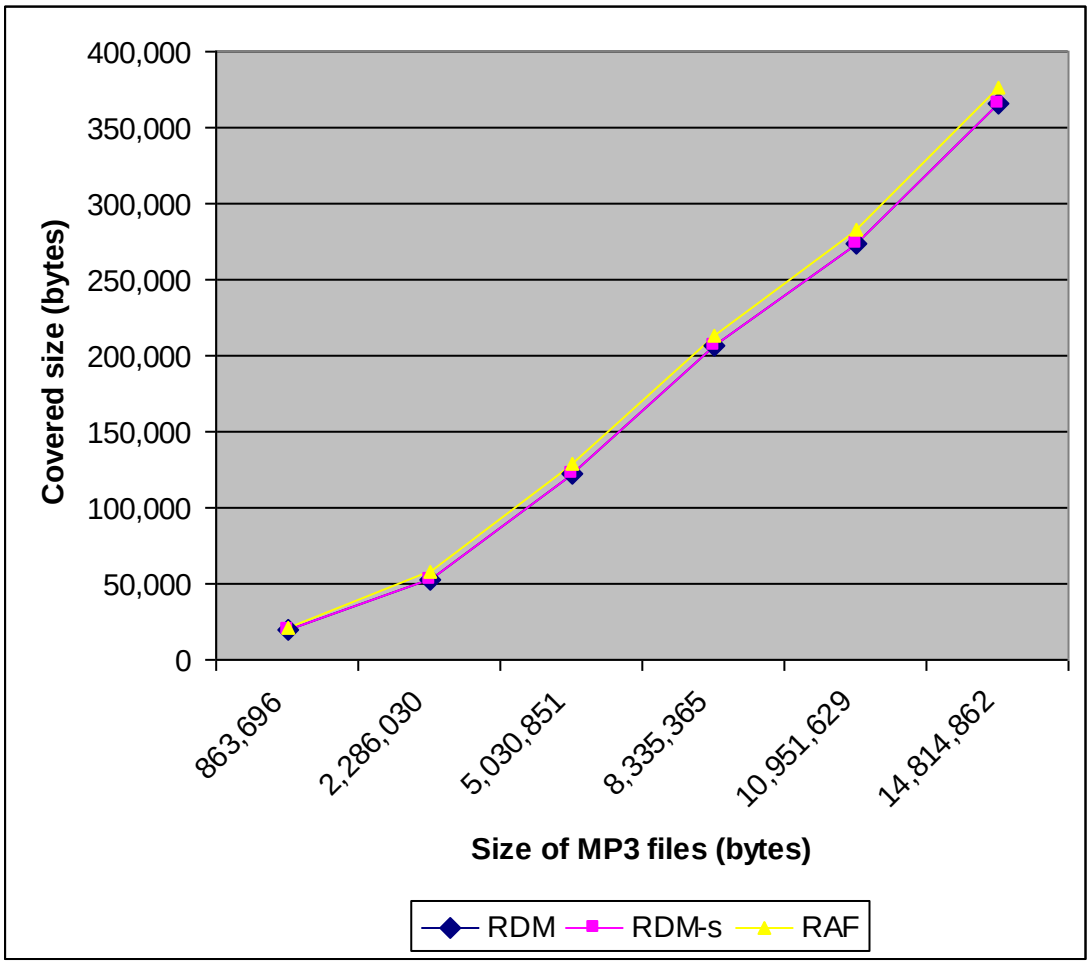
And with increasing cover size the hiding capacity increasing in linear form as shown in graph 4.1

In dynamic locations method, table 4.3 present the results of maximum of the cover sizes for MP3 files which were tested in RMD, RMD-s and RAF method.

Table 4.3 Maximum of embedding in dynamic location.

		Methods		
MP3 files	Size of MP3 file	RDM	RDM-s	RAF
File1	863,696	19,782	19,782	21,382
File2	2,286,030	52,421	52,421	58,469
File3	5,030,851	121,717	121,717	128,369
File4	8,335,365	207,003	207,003	213,595
File5	10,951,629	274,330	274,330	283,066
File6	14,814,862	365,196	365,196	376,780

Graph 4.2 present an analysis of the result for table 4.3.



Graf 4.2 Maximum of embedding in dynamic location.

As shown in table 4.2 the covered sizes in MP3 files that were used in RMD, RMD-s and RAF methods in dynamic method is slightly less than that used in fixed method because the location that used in hiding process changeable and it around 38 frame. But in general it is stay a good ratio and enough to hide some of data. And increasing of cover size is linear form as we see in graph 4.2.

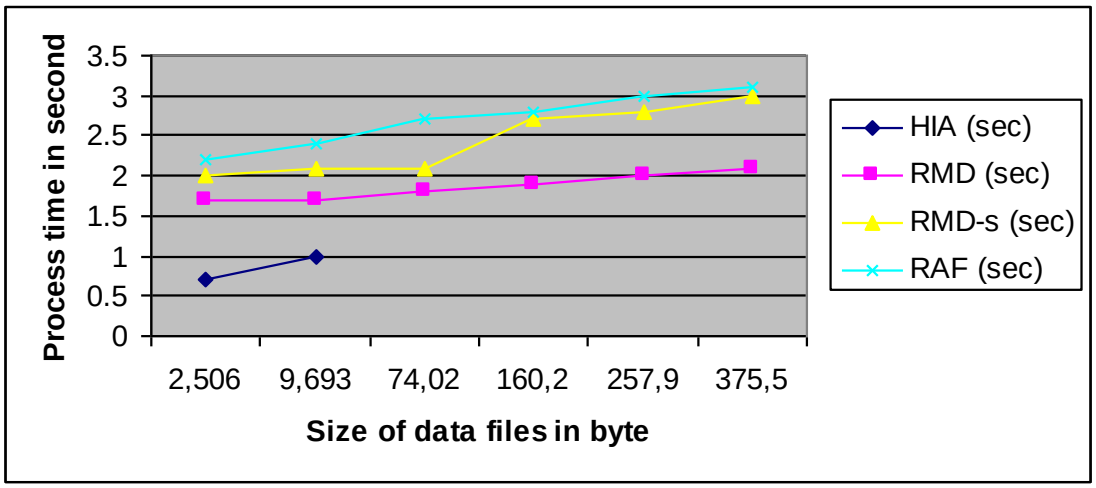
4.4.2 Result of hiding process time

Table 4.3 presents the time of embedding process for data files in **fixed location** method, as an example each 38 frame in same MP3 file with size 14,814,862 byte.

Table 4.4 Embedding time for fixed location method.

			Process time			
Data Files	Type of files	Files size (Bytes)	HIA (sec)	RMD (sec)	RMD-s (sec)	RAF (sec)
File 1	Text Document (txt)	2,506	0.7	1.7	2	2.2
File 2	Gif Image (gif)	9,693	1	1.7	2.1	2.4
File 3	Text Document (txt)	74,022	no process	1.8	2.1	2.7
File 4	Microsoft Word Document (doc)	160,256	no process	1.9	2.7	2.8
File 5	Adobe Acrobat Document (pdf)	257,929	no process	2	2.8	3
File 6	Adobe Acrobat Document (pdf)	375,563	no process	2.1	3	3.1

Graph 4.3 present an analysis of the result for table 4.4.



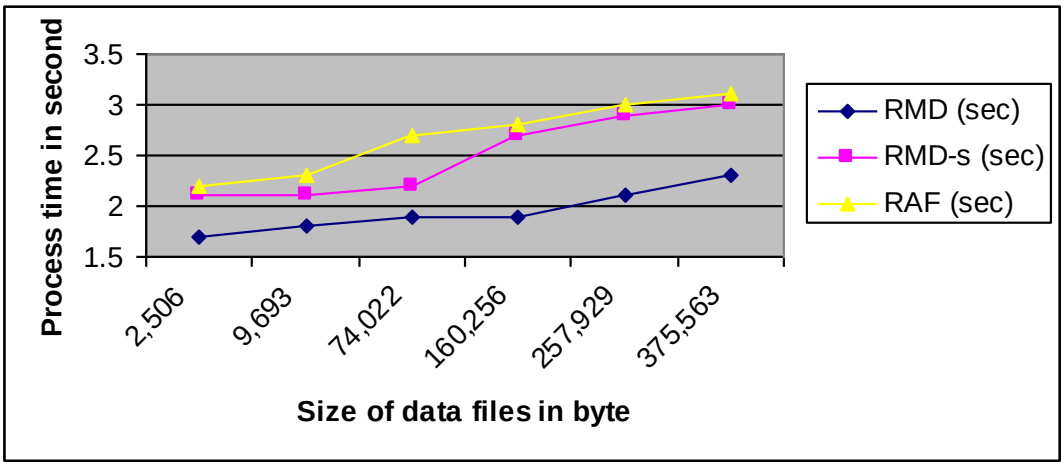
Graph 4.3 Embedding time for fixed location method

As shown in table 4.4 there is "no processing" in HIA method for last three file because the capacity of cover size decreases progressively until it reach zero. In this case the covered size not enough to hide data.

Table 4.5 presents the time of embedding process for data files **in dynamic location** method in same MP3 file with size 14,814,862 byte.

Table 4.5 Embedding time for dynamic location method.

			Process time		
Data Files	Type of files	Files size (Bytes)	RMD (sec)	RMD-s (sec)	RAF (sec)
File 1	Text Document (txt)	2,506	1.7	2.1	2.2
File 2	Gif Image (gif)	9,693	1.8	2.1	2.3
File 3	Text Document (txt)	74,022	1.9	2.2	2.7
File 4	Microsoft Word Document (doc)	160,256	1.9	2.7	2.8
File 5	Adobe Acrobat Document (pdf)	257,929	2.1	2.9	3
File 6	Adobe Acrobat Document (pdf)	375,563	2.3	3	3.1



Graph 4.4 Embedding time for dynamic location method

In general the time that need for embedding processes in fixed and dynamic location method is convergent excepting the embedding time for HIA method, because the runtime in analyzing stage for all method is almost similar.

4.5 Signal to Noise Ratio (SNR) measurement:

SNR used to evaluate the quality of MP3 file after using steganography with the original MP3 file.

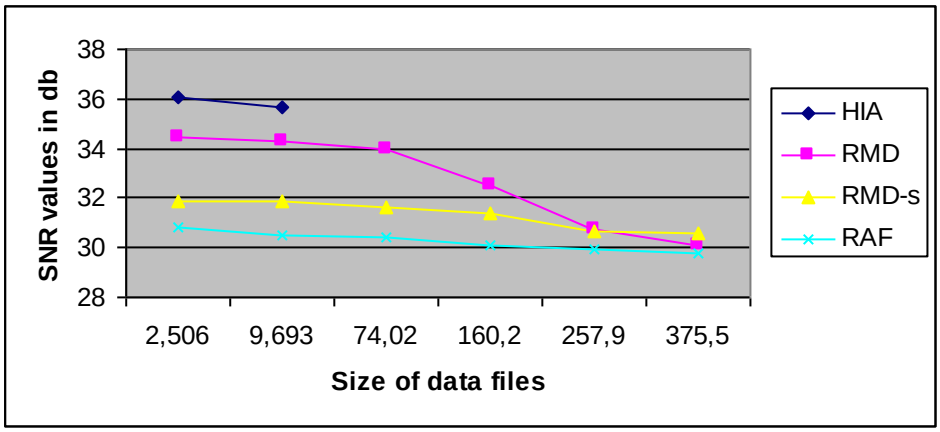
To measure SNR for MP3 files we need to convert files that encoded in manner MP3 to file of raw audio data, then calculate the noise on raw data, since the MP3 files contains compressed data using Huffman coding. To get the sound we convert MP3 files to

sound files (wav) contain voltage (amplitude) value for sound then we measure signal to noise ratio between the original MP3 file and the file that used in hiding process as in the mathematical equation 3.

Table 4.6 present SNR values of embedding the data files in MP3 file with size 14,814,862 bytes in HIA, RMD, RMD-s and RAF methods in fixed locations (each 38 frame).

Table 4.6 SNR of embedding in fixed location (38).

		SNR of methods (db)			
Data file	File size (byte)	HIA	RMD	RMD-s	RAF
File 1	2,506	35.08	34.48	31.88	30.79
File 2	9,693	34.69	34.32	31.86	30.52
File 3	74,022	no process	33.94	31.66	30.44
File 4	160,256	no process	32.52	31.42	30.12
File 5	257,929	no process	30.77	30.66	29.95
File 6	375,563	no process	30.12	30.59	29.81



Graph 4.5 SNR values for fixed location method

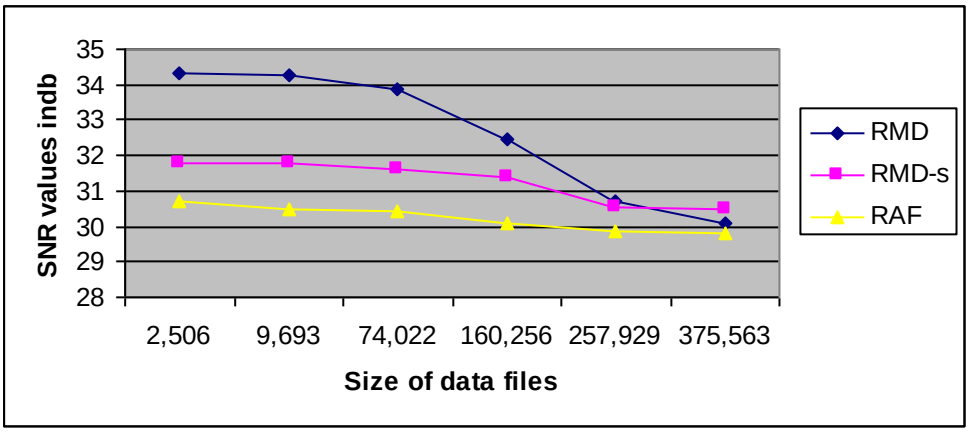
There is "no process" field in table 4.6 because the capacity of MP3 file less than data files size. SNR values decrease whenever cover size increased.

HIA method show higher values of SNR but the other hand HIA can store information less than other technique and as we see in graph 5.5 SNR values decrease gradually in RMD-s, RMD and HAF methods.

Table 4.7 presents SNR values of embedding the data files in MP3 file with size 14,814,862 bytes in HIA, RMD, RMD-s and HAF methods in dynamic locations.

Table 4.7 SNR of embedding in dynamic location.

		SNR of methods (db)		
Data file	File size (byte)	RMD	RMD-s	HAF
File 1	2,506	34.32	31.81	30.71
File 2	9,693	34.29	31.79	30.47
File 3	74,022	33.85	31.6	30.4
File 4	160,256	32.47	31.37	30.11
File 5	257,929	30.72	30.55	29.89
File 6	375,563	30.08	30.49	29.78



Graph 4.6 SNR for dynamic location method

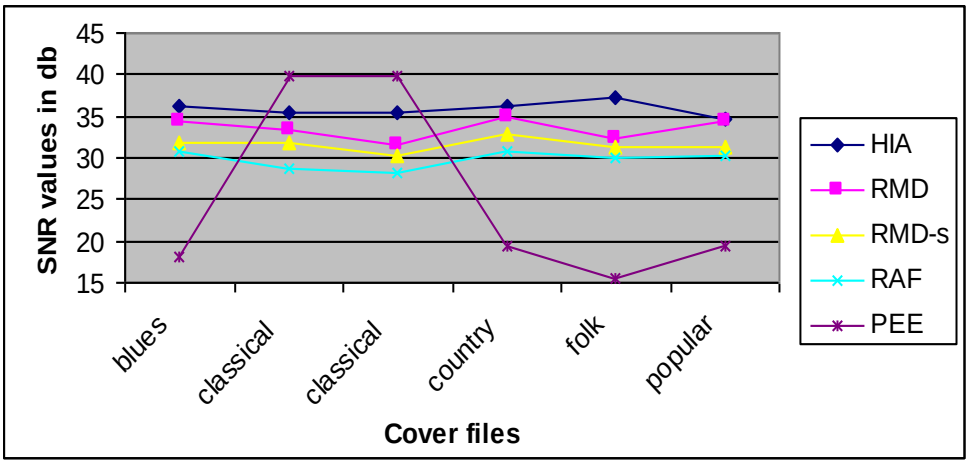
SNR values of embedding data in dynamic locations almost similar or greater than SNR values in fixed location. RMD method show higher values of SNR in dynamic location method.

In general, SNR values decrease gradually in RMD-s, RMD then HAF methods.

In this research we compare the SNR values with previous research provided by Diqun Yan et. al. [8] who propose a reversible data hiding method for digital audio using prediction error expansion technique. Table 4.7 present SNR values for five methods for embedding data in MP3 files with different type and size.

Table 4.8 SNR of embedding data.

	SNR of methods (db)				
Cover file genre	HIA	RMD	RMD-s	RAF	PEE [8]
Blues	36.08	34.48	31.88	30.79	18.22
Classical	35.46	33.42	31.76	28.68	39.75
Classical	35.33	31.67	30.24	28.18	39.75
Country	36.23	34.89	32.82	30.69	19.36
Folk	37.18	32.38	31.24	29.97	15.51
Popular	34.57	34.49	31.17	30.14	19.41



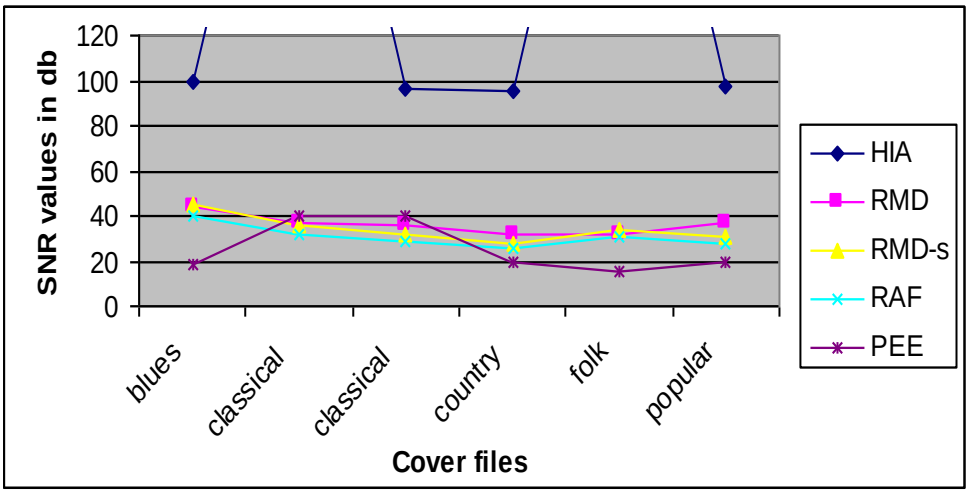
Graph 4.7 SNR of embedding data.

As we see in graph 4.7 the SNR values for the four methods that used in this research are much higher than method that used in previous research [8] except classical files of MP3 files which appears the big different of SNR values between results for this research and other researches.

Table 4.9 present SNR values for five methods for embedding data in MP3 files with same duration time (10 second) and different type of genre.

Table 4.9 SNR of embedding data in 10 second of cover files.

	SNR of methods (db)				
Cover file genre	HIA	RMD	RMD-s	RAF	PEE [8]
Blues	99.06	44.23	45.16	40.29	18.22
Classical	∞	37.13	35.45	31.77	39.75
Classical	96.33	35.69	31.45	28.79	39.75
Country	95.32	31.41	27.46	25.28	19.36
Folk	∞	31.77	34.20	30.45	15.51
Popular	97.25	36.79	30.43	28.03	19.41



Graph 4.8 SNR of embedding data in 10 second of cover files.

In general, as shown in graph 4.8 the values of SNR for HIA method or in other methods which used in this research are higher than methods that used in previous research [8].

Figure 4.1, 4.2, 4.3, 4.4 and 4.5 present the histogram for 1second of the audio file of genre type: country.

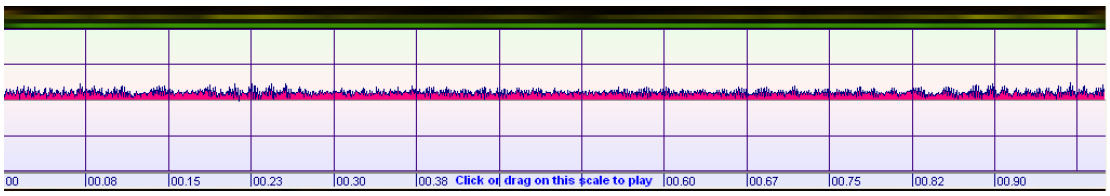


Figure 4.1 Histogram for 1 second of original sound.

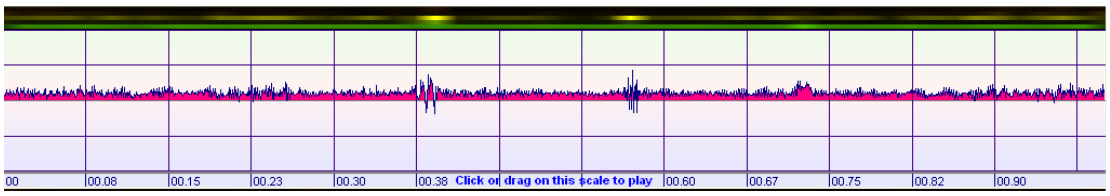


Figure 4.2 Histogram for 1 second of sound after using HIA method

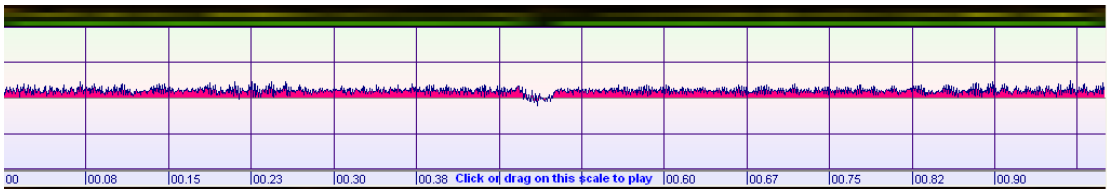


Figure 4.3 Histogram for 1 second of sound after using RMD method

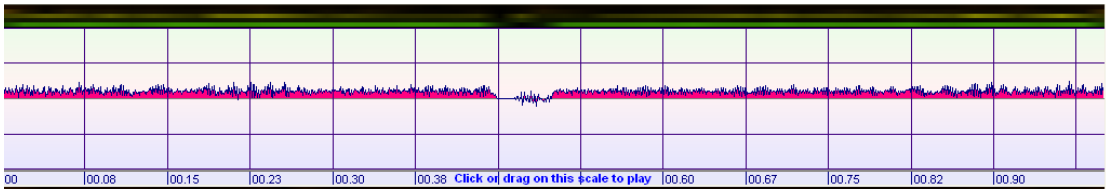


Figure 4.4 Histogram for 1 second of sound after using RMD-s method

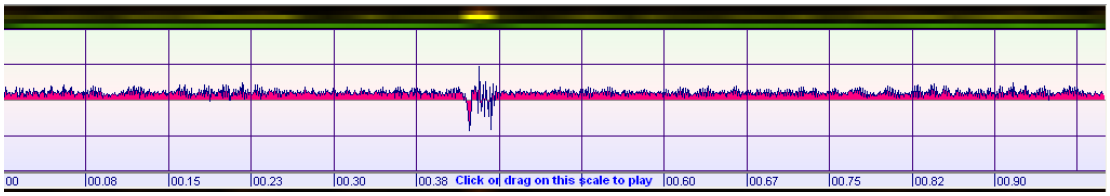


Figure 4.5 Histogram for 1 second of sound after using RAF method

As shown in figure 4.2, 4.3, 4.4 and 4.5 we can notes the effect of embedding data on mp3 file.

5. Conclusion and Future Works

5.1 Conclusion

We have a novel methods of data hiding based on replacing some bytes of audio sound in MP3 frames. Several methods were implemented to store data inside MP3 file. These methods based on the replacing method but it is using a few size of cover file to keep on the quality of MP3 file or using a large size of cover file without changing on the size of original file.

By increasing the cover size that used in hiding data we increase the data that will be stored. Hiding these large amounts of data in MP3 files (about 2.6% of the cover file size) don't affect in the size of the original file. By using four methods (HIA by hiding in last each byte of the MP3 frame, RMD, RMD-s and HAF by replacing parts of the MP3 frame each number of frames) to store data allowed sending a large of data through Multimedia files in different ways. These methods used steganography in several parts of MP3 frame that not used in other research.

Previous researches able to store up to 12 bit per frame [9] i.e. about 0.3% of the cover file size. In our research were able to store up to 2.6 of the cover file size.

5.2 Future works

As a future works, it is suggested to work on the following topics:

1. Proposing a model that can evaluate and determine the suitable frames that could be used for hiding information in order to minimize the impact of the embedding process on the quality of the sound.
2. Decrease the size of information before hiding process by using compressing algorithm.
3. Increase the cover size by using new generation of Multimedia container form that use digital audio and digital video as MP4 for hiding data.
4. Evaluate the robustness of the models proposed due to active attacks and propose new models to achieve good robustness against active attacks. The robustness was out of scope in this research.

The researcher would like to recommend that further studies be done in this regard.

References

[1] Fabien A. P. Petitcolas, Ross J. Anderson, Markus G. Kuhn, "Information Hiding-A Survey", Proceedings of the special issue on protection of multimedia content, IEEE, vol. 87, no. 7, pp. 1062-1078, Jul 1999.

[2] D. Chaum, Untraceable electronic mail, return addresses and digital pseudonyms, Communications of the A.C.M., vol. 24, no. 2, pp. 84-88, Feb 1981.

[3] **The Oxford English dictionary: being a corrected re-issue** Clarendon Press, Oxford, 1933.

[4] R. J. Anderson, "Information hiding: first international workshop", (ed.), vol. 1174 of Lecture Notes in Computer Science, Isaac Newton Institute, Cambridge, England, May 1996, Springer-Verlag, Berlin, Germany, ISBN 3-540-61996-8.

[5] Hosei Matsuoka, "Spread Spectrum Audio Steganography using Sub-band Phase Shifting", Proceedings of the International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Pasadena, CA, USA IEEE, 2006.

[6] J. Fridrich and D. Soukal, "Matrix embedding for large payloads", IEEE Trans. Inform., Forens, France, vol. 1, no. 3, pp. 390–395, Sep. 2006.

[7] Rashid Ansari, Hafiz Malik, Ashfaq Khokhar, "Data-Hiding in Audio using Frequancy-Selective Phase Alteration", the International Conference on Acoustics, Speech, and Signal Processing, vol. 5, Montreal, Canada, IEEE, May 2004.

[8] Diqun Yan, Rangding Wang, "Reversible Data Hiding for Audio Based on Prediction Error Expansion" CKC Software Laboratory, University of Ningbo. Ningbo 3115211, International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Taiwan, IEEE, pp. 249-252, 2008.

[9] Beixing Deng, Jie Tan, Bo Yang, Xing Li, "A Novel Steganography Method On Modifying Quantized Spectrum Values of MPEG/Audio Layer III", Proceedings of the

7th Conference WSEAS International Conference on Applied Informatics and Communications, vol. 7, USA, August 2007.

[10] D. Nassar, "Data Hiding In Wave Media File By Using Steganography Techniques", MSc Thesis, University Of Jordan, Jordan, 2003.

[11] Nedejko Cvejic "Algorithms for Audio Watermarking and Steganography" Department of Electrical and Information Engineering, Information Processing Laboratory, MSc Thesis, University of Oulu, Finland, 2004.

[12] Mohammad Salem Mohammad Al-Atoom "Using Steganography For Embedding Data in MP3 Files", MSc Thesis, University of Jordan, Jordan, 2009

[13**]** Manish Mahajan, Akashdeep Sharma, "Steganography in Colored Images Using Information Reflector with 2k Correction", International Journal of Computer Applications, Germany, pp. 0975 – 8887, 2010.

[14] Mengyu Qiao, Andrew H. Sung, Qingzhong Liu, "Feature Mining and Intelligent Computing for MP3 Steganalysis", International Joint Conference on Bioinformatics, Systems Biology and Intelligent Computing, Shanghai, China, IEEE, pp. 627-630, Aug 2009.

[15] Staffan Gadd, Thomas Lenart, "A Hardware Accelerated MP3 Decoder with Bluetooth Streaming Capabilities", MSc Thesis, Lunds Tekniska Högskola, Lund University, Lund, Sweden, November 2001.

[16] Karlheinz Randenburg, "MP3 and AAC Explained", 17th International Conference: High-Quality Audio Coding, Erlangen, Germany, Aug 1999.

[17] K. Brandenburg and H. Popp, "An Introduction to MPEG Layer-3", Fraunhofer Institute, EBU Technical Review, Geneva, Switzerland, June 2000

URL: http://www.mp3-tech.org/programmer/docs/trev_283-popp.pdf

[18] Scot Hacker, **MP3: The Definitive Guide**, First Edition, O'Reilly, March 2000.

[19] Vincenzo Marziale, Andrea Vitaletti, “A Framework for Internet QoS Requirements Definition and Evaluation: an experimental approach”, IST Mobile Communication Summit 2001, Barcelona, Spain, 2001.

[20] M. Nilsson, **“**ID3 tag version 2.3.0**”,** Document: id3v2.3, 3rd February 1999 URL: http://www.id3.org/id3v2.3.0.

[21] Richard Akester, “Low delay MP3 Internet Telephony”, Web Multimedia and Communications in WSEAS Multiconference, Izmir, Turkey, 2004.

[22] Rassol Raissi, “The Theory Behind Mp3”, Docment: The Theory Behind Mp3, 2002. URL: http://www.mp3-tech.org/programmer/docs/mp3_theory.pdf

[23] Krister Lagerstrom, “Design and Implementation of an MPEG-1 Layer III Audio Decoder”, MSc Thesis, Chalmers University of Technology, Gothenburg, Sweden, 2001.

[24] Praveen Sripada, “MP3 Decoder in Theory and Practice”, Department of Signal Processing and Telecommunications, MSc Thesis, Blekinge Tekniska Högskola Univercity of Blekinge, Sweden, March 2006.

[25] Chung-Ping Wu, C.-C. Jay Kuo, “Design of Integrated Multimedia Compression and Encryption Systems”, Transactions on Multimedia, vol. 7, no. 5, USA, IEEE, pp. 828 - 839, Oct 2005.

[26] Walt Kester, **Analog-Digital Conversion,** Analog Devices, Chapter 6, 2004.

[27] Walt Kester, **The Data Conversion Handbook**, Analog Devices, Chapter 2, 2004.

[28] Hank Zumbahlen, **Basic Linear Design,** Analog Devices, Chapter 6, 2006.

[29] Hank Zumbahlen, **Linear Circuit Design Handbook**, Analog Devices. Chapter 6, 2008.

الملخص

علم الإخفاء Steganography هو أسلوب من أساليب إخفاء البيانات، بحيث يشكل طريقة لحماية البيانات عن طريق تضمينها ضمن وسط رقمي لهدف التعريف وحقوق الملكية والرسائل السرية.
الملفات المتعددة الوسائط تستخدم بشكل رئيسي كغطاء حامل للبيانات في علم الإخفاء، بعض تقنيات علم الإخفاء تعتمد على تغيير الخانات الأقل أهمية بينما تعتمد طرق أخرى على إضافة البيانات ضمن أطر الوسائط المتعددة.
إخفاء البيانات في ملفات MP3 صعبة نوعا ما بسبب طبيعة صيغة هذه الملفات ومستوى التعقيد في هيكلتها. البيانات في ملفات MP3 يتم ترميزها بطريقة Huffman encoding بالمقابل ملفات MP3 يمكن ان تكون فعالة لإخفاء البيانات وذلك بسبب حجمها الكبير الذي يساعد في تضمين البيانات.
في هذه الدراسة تم إخفاء البيانات في أطر MP3 باستخدام أسلوبين؛ إخفاء البيانات في الجزء المساعد في أطر MP3 (HIA) والأسلوب الآخر بتبديل بعض الأطر بالمعلومات التي يراد إخفاؤها.
الأسلوب الثاني يستخدم أسلوبين فرعيين: تبديل البيانات الرئيسية بحيث يتم إخفاء البيانات بتبديل المعلومات الصوتية الرئيسية في بعض الأطر. اوبتبديل بعض الأطر الصوتية كاملة (HAF) بحيث يتم تبديل كل الحقول الموجودة ضمن الإطار الذي تم اختياره باستثناء ترويسة الإطار.
الأسلوب الأول يخفي البيانات عن طريق استبدال آخر بايت في الأطر الصوتية. من اجل تقييم هذا الأسلوب تم حساب نسبة الإشارة للضجيج (SNR) الناتجة لملف MP3 بسبب تضمين احجام مختلفة من البيانات في هذا الملف. حجم البيانات التي يمكن تضمينها بهذه الطريقة تصل الى ٠,٢% من حجم ملف الغطاء.
في هذا الأسلوب تم إخفاء ١٤ كيلو بايت من البيانات في ١٤ ميغا بايت من الملفات MP3 بتأثير نسبة ضجيج ٣٠ ديسبل.
الأسلوب الثاني يقوم بتبديل البيانات الرئيسية للإطار كل ٣٨ إطار (RMD). بهذه الطريقة تمكنا من تضمين بيانات بنسبة ٢,٢% من حجم ملف الغطاء.
في هذا الأسلوب تم إخفاء ١٢٤ كيلو بايت من البيانات في ٥ ميغا بايت من الملفات MP3 بتأثير نسبة ضجيج ٣٤ ديسبل.
تم عمل تعديل على الأسلوب الثاني من اجل تبديل الإطار كاملا كل ٣٨ إطار. زادت نسبة الإشارة للضجيج الى ٢,٦% من حجم ملف الغطاء على حساب نسبة الإشارة للضجيج.
في هذا الأسلوب تم إخفاء ٢١٧ كيلو بايت من البيانات في ٨ ميغا بايت من الملفات MP3 بتأثير نسبة ضجيج ٣١ ديسبل.