Volume 3, Issue 2, February 2013 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

A Conceptual Study of Various Data Hiding Techniques – A Review

Saket B. Parmar, Piyush P. Pokharna, Abhay B. Patil

Electronics Department

Bharati Vidyapeeth Deemed University

Deemed University, India

Abstract— In today's world maintaining the secrecy of information is of prime concern. Moreover it is sometimes not enough to keep the contents of the message secret but it may also be important to keep the mere existence of the message itself a secret. Earlier Cryptography was the technique which was used for securing crucial information which used 'scrambling' process i.e., the message was scrambled, so when tapped the message would not make any sense. But now, steganography is being used as the technique for data hiding which uses a unique technique of making the message itself seem 'invisible' and thus it provides better protection for the secret data. This paper presents a composite study of various such methods involved in data hiding. In steganography, data can be hidden in otherwise unsuspicious file formats such as audio file, image file etc. As for audio file or image file in which the data is hidden will be perceived indifferently, without the hidden data being noticeable and decipherable only to the desired receiver.

Keywords: Steganography, Cryptography, Data Hiding, Secret Data.

I. Introduction

The wide usage of internet and conversion of data into digital format has caused major revolution. The availability of the software and reducing prices of digital devices have made it possible for the consumers from all over the world to create, edit and exchange multimedia data. This multimedia data can be transmitted in an errorless way over the internet. But, we know that the internet is being used by millions of people around the world, hence sensitive information cannot be sent directly over this public channel (internet). In order to secure this sensitive information a special technique is used which is called 'Steganography'. The word *steganography* is of Greek origin and means 'covered writing'. The objective of steganography is to hide a secret message within a cover-media in such a way that others cannot make out the presence of the hidden message. The following formula provides a very generic description of the pieces of the steganographic process:

cover_medium + hidden_data + stego_key = stego_medium

In this context, the *cover_medium* is the file in which we will hide the *hidden_data*, which may also be encrypted using the *stego_key*. The resultant file is the *stego_medium* (which will, of course. be the same type of file as the cover_medium). The cover_medium (and, thus, the stego_medium) are typically image or audio files. [1]

II. Cryptography

It is science of writing data in secret code. Cryptography not only protects data from theft or alteration, but can also be used for user authentication. There are, in general three types of cryptographic schemes used to accomplish these goals, secret key (symmetric key), public key (asymmetric) cryptography and hash function each of which is described below. In all cases the initial unencrypted data is referred to as plaintext. It is encrypted into ciphertext which in turn be decrypted into usable plain text.

A) ***Types of Cryptographic Algorithms:*** There are several ways of classifying cryptographic algorithms. For purpose of this paper, they are classified on the basis of number of keys that are employed for encryption and decryption and further defined by their application and use. The three types of algorithms that will be discussed are:-

Secret Key Cryptography (SKC): With secret key cryptography, a single key is used for both encryption and decryption. The sender uses the key (some set of rules) to encrypt the plaintext and sends the ciphertext to the receiver. The receiver applies the same key (ruleset) to decrypt the message and recover the plaintext. Because a single key is used for both functions, secret key cryptography is also called symmetric encryption. With this form of cryptography, it is obvious that the key must be known to both the sender and the receiver; that, in fact, is the secret. The biggest difficulty with this approach, of course, is the distribution of the key.

B) ***Public-Key Cryptography (PKC):*** It describes a two-key crypto system in which two parties could engage in a secure communication over a non-secure communications channel without having to share a secret key. PKC

depends upon the existence of so-called one-way functions, or mathematical functions that are easy to compute whereas their inverse function is relatively difficult to compute.

C) Hash Functions: Hash functions are also called the message digests and one-way encryption, are algorithms that, in some sense, use no key. Instead, a fixed-length hash value is computed based upon the plaintext that makes it impossible for either the contents or length of the plaintext to be recovered.

Hash functions are well-suited for ensuring data integrity because any change made to the contents of a message will result in the receiver calculating a different hash value than the one placed in the transmission by the sender. Since it is highly unlikely that two different messages will yield the same hash value, data integrity is ensured to a high degree of confidence.

III. Steganography

Steganography is the art and science of writing hidden messages in such a way that no one, apart from the sender and intended recipient, suspects the existence of the message .The advantage of steganography, over cryptography alone, is that messages do not attract attention to themselves. because cryptography encodes the secret message in unreadable form for third party but steganography hides the secret message behind some text ,image, video etc so that third party is Unaware of secret hidden message.

A) Types of Steganographic Algorithms: There are various types of steganographic algorithms used today. For simplification these steganographic algorithms are classified based on the medium (cover) they use to embed secret data. The following are the various types of Steganographic algorithms

1) Text Steganography Algorithms:

a) Line-shift encoding: It involves actually shifting each line of text vertically up or down by as little as 3 centimeters. Depending on whether the line was up or down from its initial position it would equate to a value that would or could be encoded into a secret message.

b) Word-shift encoding: It works in much the same way that line-shift encoding works, the only difference is that, in this the horizontal spaces between words is used to equate a value for the hidden message. This method of encoding is less visible than line-shift encoding but requires that the text format should support variable spacing. Text based encoding methods require either the original file or the knowledge of the original files formatting to be able to decode the secret message of the human visual system (HVS). Almost any plain text, cipher text, image and any other media that can be encoded into a bit stream can be hidden in a digital image.

2) Image Steganography algorithms:

In the digital world of today this type of encoding is widely accepted. This is because it can take advantage of the limited power of human vision. When dealing with digital images for use with Steganography, 8-bit and 24-bit per pixel image files are typical. Both have advantages and disadvantages, as we will explain below. 8-bit images are a great format to use because of their relatively small size. The drawback is that only 256 possible colors can be used which can be a potential problem during encoding. 24-bit images offer much more flexibility when used for Steganography. The large numbers of colors (over 16 million) that can be used go well beyond the human visual system (HVS), which makes it very hard to detect once a secret message, has been encoded. The other benefit is that a much larger amount of hidden data can be encoded into a 24-bit digital image as opposed to an 8-bit digital image. The one major drawback to 24-bit digital images is their large size (usually in MB) makes them more suspect than the much smaller 8-bit digital images (usually in KB) when sent over an open system such as the Internet. Digital image compression is a good solution to 24-bit digital images but it has drawback that the possibility of the uncompressed secret message to lose parts of its contents.

a) Least significant bit (LSB) encoding: It is by far the most popular of the coding techniques used for digital images. By using the LSB of each byte (8 bits) in an image for a secret message, you can store 3 bits of data in each pixel for 24-bit images and 1 bit in each pixel for 8-bit images.

b) Digital Watermarking: This technique actually extends an images data by masking the secret data over the original data as opposed to hiding information inside of the data[2].

3) Audio Steganography Algorithms:

A number of different cover objects (signals) can be used to carry hidden messages. Data hiding in audio signals exploits imperfection of human auditory system known as audio masking. In presence of a loud signal (masker), another weaker signal may be inaudible, depending on spectral and temporal characteristics of both masked signal and masker [4]. Masking models are extensively studied for perceptual compression of audio signals [3] In the case of perceptual compression the quantization noise is hidden below the masking threshold, while in a data hiding application the embedded signal is hidden there. Data hiding in audio signals is especially challenging, because the human auditory system operates over a wide dynamic range. The human auditory system perceives over a range of power greater than one billion to one and a range of frequencies greater than one thousand to one. Sensitivity to additive random noise is also acute. The perturbations in a sound file can be detected as low as one part in ten million (80 dB below ambient level). However, there are some "holes" available. While the human auditory system has a large dynamic range, it has a fairly small differential range. As a result, loud sounds tend to mask out quiet sounds. Additionally, the human auditory system is unable to perceive absolute phase, only relative phase. Finally, there are some environmental distortions so

common as to be ignored by the listener in most cases [4]. Now we will discuss many of these methods of audio data hiding technology.

a) Parity Coding: One of the prior works in audio data hiding technique is parity coding technique. Instead of breaking a signal down into individual samples, the parity coding method breaks a signal down into separate regions of samples and encodes each bit from the secret message in a sample region's parity bit. If the parity bit of a selected region does not match the secret bit to be encoded, the process flips the LSB of one of the samples in the region. Thus, the sender has more of a choice in encoding the secret bit, and the signal can be changed in a more unobtrusive fashion [6]. Figure 1, shows the parity coding procedure.

b) Phase Coding: The phase coding method works by substituting the phase of an initial audio segment with a reference phase that represents the data. The phase of subsequent segments is adjusted in order to preserve the relative phase between segments. Phase coding, when it can be used, is one of the most effective coding methods in terms of the signal-to perceived noise ratio. When the phase relation between each frequency component is dramatically changed, noticeable phase dispersion will occur. However, as long as the modification of the phase is sufficiently small (sufficiently small depends on the observer; professionals in broadcast radio can detect modifications that are unperceivable to an average observer), an inaudible coding can be achieved [5]. . Phase coding relies on the fact that the phase components of sound are not as perceptible to the human ear as noise is. Rather than introducing perturbations, the technique encodes the message bits as phase shifts in the phase spectrum of a digital signal, achieving an inaudible encoding in terms of signal-to-perceived noise ratio [6].

Phase coding is explained in the following procedure:

i. The original sound signal is broken up into smaller segments whose lengths equal the size of the message to be encoded.

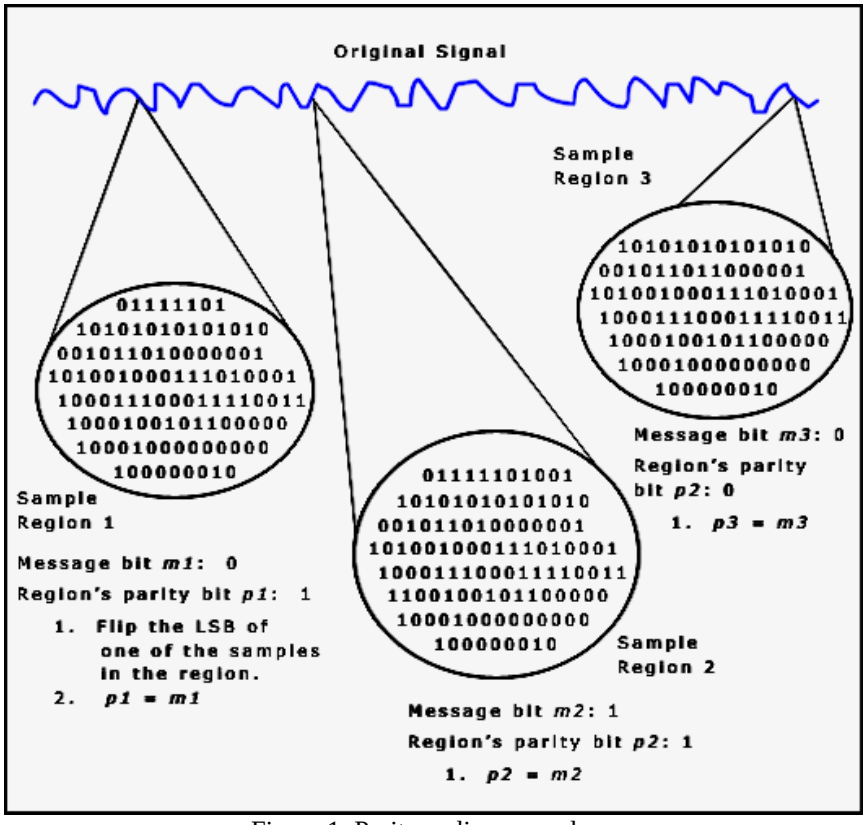

Figure 1: Parity coding procedure

ii. A Discrete Fourier Transform (DFT) is applied to each segment to create a matrix of the phases and Fourier transform magnitudes.

iii. Phase differences between adjacent segments are calculated.

iv. Phase shifts between consecutive segments are easily detected. In other words, the absolute phases of the segments can be changed but the relative phase differences between adjacent segments must be preserved. Therefore the secret message is only inserted in the phase vector of the first signal segment as follows:

v. A new phase matrix is created using the new phase of the first segment and the original phase differences.

vi. Using the new phase matrix and original magnitude matrix, the sound signal is reconstructed by applying the inverse DFT and then concatenating the sound segments back together.

c) Spread Spectrum: In a normal communication channel, it is often desirable to concentrate the information in as narrow a region of the frequency spectrum as possible in order to conserve available bandwidth and to reduce power. The basic spread spectrum technique, on the other hand, is designed to encode a stream of information by spreading the encoded data across as much of the frequency spectrum as possible. This allows the signal reception, even if there is interference on some frequencies. While there are many variations on spread spectrum communication, we concentrated on Direct Sequence Spread Spectrum encoding (DSSS). The DSSS method spreads the signal by multiplying it by a chip, a maximal length pseudorandom sequence modulated at a known rate. Since the host signals are in discrete-time format, we can use the sampling rate as the chip rate for coding. The result is that the most difficult problem in DSSS receiving, that of establishing the correct start and end of the chip quanta for phase locking purposes, is taken care of by the discrete nature of the signal. Consequently, a much higher chip rate, and therefore a higher associated data rate, is possible. Without this, a variety of signal locking algorithms may be used, but these are computationally expensive [5].

$$phase\_new = \begin{cases} \pi/2 & if \quad message \quad bit = 0 \\ -\pi/2 & if \quad message \quad bit = 1 \end{cases}$$

To extract the secret message from the sound file, the receiver must know the segment length. The receiver can then use the DFT to get the phases and extract the information (consider Figure 2 for phase coding procedure).

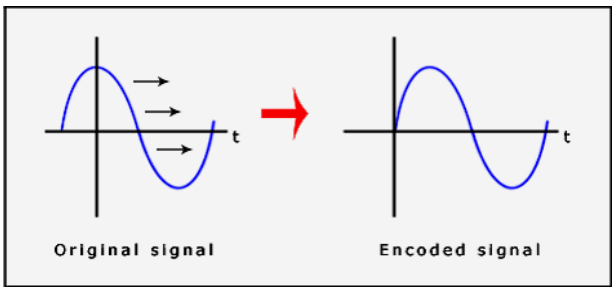

Figure 2. The signals before and after Phase coding procedure

Procedure: In DSSS, a key is needed to encode the information and the same key is needed to decode it. The key is pseudorandom noise that ideally has flat frequency response over the frequency range, i.e., white noise. The key is applied to the coded information to modulate the sequence into a spread spectrum sequence.

DSSS method: The code is multiplied by the carrier wave and the pseudorandom noise sequence, which has a wide frequency spectrum. As a consequence, the spectrum of the data is spread over the available band. Then, the spread data sequence is attenuated and added to the original file as additive random noise (see Figure 3). DSSS employs bi-phase shift keying since the phase of the signal alternates each time the modulated code alternates (see Figure 4). For decoding, phase values f0 and f0 + p are interpreted as a "0" or a "1," which is a coded binary string [5]. Spread Spectrum is shown in Figure 4.

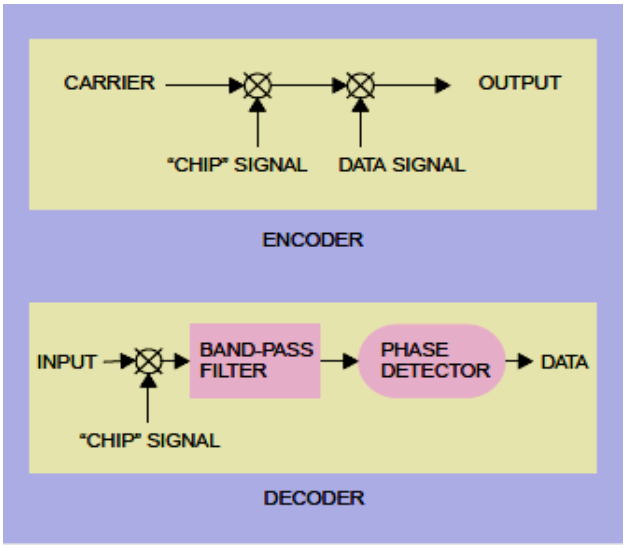

Figure 3. Spread Spectrum Encoding

In the decoding stage, the following is assumed:

I. The pseudorandom key is maximal (it has as many combinations as possible and does not repeat for as long as possible). Consequently it has a relatively flat frequency spectrum.

II. The key stream for the encoding is known by the receiver. Signal synchronization is done, and the start/stop point of the spread data is known.

III. The following parameters are known by the receiver: chip rate, data rate, and carrier frequency.

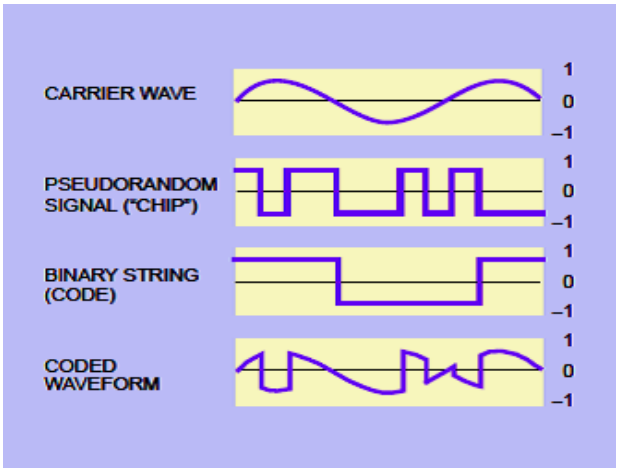

Figure 4. Synthesized spread spectrum information encoded by the direct sequence method

d) Echo hiding: In echo hiding, information is embedded in a sound file by introducing an echo into the discrete signal. Like the spread spectrum method, it too provides advantages in that it allows for a high data transmission rate and provides superior robustness when compared to the noise inducing methods. If only one echo was produced from the original signal, only one bit of information could be encoded. Therefore, the original signal is broken down into blocks before the encoding process begins. Once the encoding process is completed, the blocks are concatenated back together to create the final signal [6].

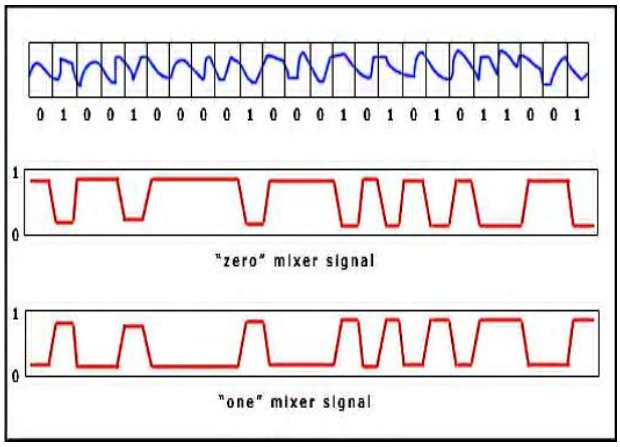

Figure 5. Echo Hiding

Echo Hiding is shown in Figure 5. Also, a message can be encoded using musical tones with a substitution scheme. For example, a Fistone will represent a 0 and a C tone represents a 1.
A normal musical piece can now be composed around the secret message or an existing piece can be selected together with an encoding scheme that will represent a message [7, 8].

e) LSB Coding: In the current endeavor, an audio file with ".wav" extension has been selected as host file. It is assumed that the least significant bits of that file should be modified without degrading the sound quality.

To do that, first one needs to know the file structure of the audio file. Like most files, WAV files have two basic parts, the header and the data. In normal wav files, the header is situated in the first 44 bytes of the file. Except the first 44 bytes, the rest of the bytes of the file are all about the data. The data is just one giant chunk of samples that represents the whole audio. While embedding data, one can't deal with the header section. That is because a minimal change in the header section leads to a corrupted audio file.

A program has been developed which can read the audio file bit by bit and stores them in a different file. The first 44 bytes should be left without any change in them because these are the data of the header section.

Then start with the remaining data field to modify them to embed textual information. For example, if the word "Audio" has to be embedded into an audio file one has to embed the binary values of the word "Audio" into the audio data field. Consider the following table:

From the table, one can come to a point that to embed the word "Audio" into the host audio file actually the corresponding eight bit binary values have to be embedded into the data field of that audio file.

To develop this algorithm multiple bits of each sample of the file have been changed or modified to insert text data in it. It has also been observed the degradation of the host audio file after modification of the bits. The bit modification was done by various ways, like 1, 2, 3, 4 bits were changed in turn. But after going through all the modification it has been observed that 1 bit change in LSB gave the best result. Thus, data can be embedded according to the following algorithm.

TABLE I
Letters with ASCII Values and Corresponding Binary Values

Letter	ASCII Value	Corresponding Binary Value
A	065	01000001
u	117	01110101
d	100	01100100
i	105	01101001
o	111	01101111

i. Algorithm (For Embedding of Data):

1. Leave the header section of the audio file untouched.
2. Start from a suitable position of the data bytes. (For the experiment purpose the present start byte was the 51st byte). Edit the least significant bit with the data that have to be embedded.
3. Take every alternate sample and change the least significant bit to embed the whole message. The data retrieving algorithm at the receiver's end follows the same logic as the embedding algorithm.

ii. Algorithm (For Extracting of Data):

1. Leave first 50 bytes.
2. Start from the 51st byte and store the least significant bit in a queue.
3. Check every alternate sample and store the least significant bit in the previous queue with a left shift of the previous bit.
4. Convert the binary values to decimal to get the ASCII values of the secret message.
5. From the ASCII find the secret message.

An audio file named "audio.wav" has been selected for this experiment. After checking the binary values of each sample, first 44 samples were left without any changes. The data embedding with LSB modification has been started after the header section. If the data embedding process is started from 51st sample then the LSB value of the 51st sample should be modified. If the binary value of the corresponding sample is "01110100" then "1" should be modified. From Table I it can be observed that to embed the letter "A", the sender has to embed the binary value "01000001". That is why according to the embedding algorithm "A" should be embedded according to Table II.

According to the same way the remaining consecutive letters of the word "Audio" is embedded in the file "audio.wav." Editing of the existing binary values with the intended binary values causes a minimal change in the audio file "audio.wav" that remains almost imperceptible to anyone other than the sender. When it comes to the point of data retrieving at the receiver's end, the retrieving algorithm has to be followed: First, change the audio message into binary format that has come from the source as stego-object. Leave first 50 bytes with
no change in them

TABLE II
Samples of Audio File with Binary Values before and after Embedding

Sample No.	Binary values of corresponding sample	Binary value to be embedded	Binary values after modification
51	01110100	0	01110100
53	01011110	1	01011111
55	10001011	0	10001010
57	01111011	0	01111010
59	10100010	0	10100010
61	00110010	0	00110010
63	11101110	0	11101110
65	01011100	1	01011101

Start from 51st bit, check the least significant bit, and store it in a queue. Check every alternate sample to collect the whole messages. Like 53rd, 55th and 57th and so on. Store the least significant bits of the alternate samples in the

queue with left shift of previous bit. Convert the binary values to decimal to get back the ASCII from which the text can be retrieved. The whole retrieval process can be depicted with the following table more thoroughly:

TABLE III
Extraction of Data from Audio File

Sample No	Binary values with embedded secret data	Bits that are stored in the queue
51	01110100	0
53	01011111	01
55	10001010	010
57	01111010	0100
59	10100010	01000
61	00110010	010000
63	11101110	0100000
65	01011101	01000001

As in Table II the embedding process of the letter "A" was stated that is why, in Table III, the retrieval process of "A" is depicted. Starting from the 51st sample, every alternate sample has been checked and the least significant bit has been stored into a queue with a left shift of previous bit. After getting all the bits in the queue, start from the left hand side, take 8 bits and convert them into equivalent decimal to get the ASCII, from the ASCII retrieve the embedded textual message. From the table, it is clearly observed that after getting 01000001 in the queue it is converted into the equivalent decimal that is 65, the ASCII of "A". Thus "A" is retrieved. Like the same way, the next letters also have been retrieved and hence the complete word "Audio."[9]

IV. Conclusion

Cryptography and steganography methods studied in this paper are both self sufficient. However, cryptographic techniques cannot be replaced by steganography, but it can be used as supplement with steganography. It's because the use of steganography reduces the chances of hidden message being detected. Thus we have studied all the major data hiding techniques in brief.

Acknowledgements

The authors would like to thank Bharti Vidyapeeth College Of Engineering, for supporting our work and to Mrs. A. S. Nigade for transferring and delivering valuable knowledge.

REFRENCES

[1] Steganography: Hiding Data within Data, Gary C. Kessler, April 2002 issue of Windows & .NET Magazine .
[2] http://avidusers.blogspot.in
[3] Rade Petrovi, Kanaan Jemili, Joseph M. Winograd, Ilija Stojanovi, Eric Metois, "*Data Hiding Within Audio Signals*", June 15, 1999, MIT Media Lab, Series: Electronics and Energetics vol. 12, No.2, pp.103-122.
[4] J. Johnston and K. Brandenburg, "*Wideband Coding Perceptual Consideration for Speech and Music*". *"Advances in Speech Signal Processing"*, S. Furoi and M. Sondhi, Eds. New York: Marcel Dekker, 1992.
[5] W. Bender, W. Butera, D. Gruhl, R. Hwang, F. J. Paiz, S. Pogreb, "*Techniques for data hiding*", IBM Systems Journal, Volume 39 , Issue 3-4, July 2000, pp. 547 – 568.
[6] Samir Kumar Bandyopadhyay, Debnath Bhattacharyya, Poulami Das, Debashis Ganguly and Swarnendu Mukherjee, "*A tutorial review on Steganography*", International Conference on Contemporary Computing (IC3-2008), Noida, India, August 7-9, 2008, pp. 105-114.
[7] Robert Krenn, "*Steganography and steganalysis*", An Article, January 2004.
[8] Francesco Queirolo, "*Steganography in Images*", Final Communications Report.
[9] Pramatha Nath Basu, Tanmay Bhowmik "*On Embedding of Text in Audio – A case of Steganography*", 2010 International Conference on Recent Trends in Information, Telecommunication and Computing

