International Journal of Computer Applications in Engineering Sciences
[VOL I, ISSUE II, JUNE 2011] [ISSN: 2231-4946]

Hiding Data in Audio Using Audio Steganography

P. Ramesh Yadav[1], V. Usha Shree[2], K. Padmapriya[3]

[1]*Vaagdevi institute of technology & science, Proddatur, Kadapa(DT) A.P, India.*
[2]*Dept of ECE,SRIT Engineering college, Anantapur, Anantapur (DT),A.P.*
[3]*Dept of ECE, JNTUA College of Engineering, Anantapur, Anantapur (DT), A. P.*

[1]ramesh0436@gmail.com, [2]valasani_usha1@yahoo.com, [3]kesaripadmapriya@rediffmail.com

***Abstract*- Steganography is an art of sending hidden data or secret messages over a public channel so that a third party cannot detect the presence of the secret messages. In this paper we proposed a steganographic method of embedding textual information in an audio file. In the proposed technique, first the audio file is sampled and then an appropriate bit of each alternate sample is altered to embed the textual information. As a steganographic approach the perceptual quality of the host audio signal was not to be degraded.**

***Keywords*- Steganography, hiding data, audio file, Audio steganography.**

I. INTRODUCTION

Have you ever wanted to hide something from your friends, family or government? If the answer is yes, then you need to learn about STEGANOGRAPHY.

Steganography, coming from the Greek words 'stegos', meaning roof or covered and 'graphia' which means writing, is the art and science of hiding the fact that communication is taking place. To give a more formal definition, the Merriam Webster Dictionary defines steganography as: "The Art or practice of concealing a message, image, or file." Steganography and cryptography are closely related. Cryptography scrambles messages so they cannot be understood. Steganography on the other hand, will hide the message so that they cannot be seen.

Using steganography, you can embed a secret message inside a piece of unsuspicious information and send it without anyone knowing of the existence of the secret message. As the field of steganography has progressed, people have become increasingly interested in being able to detect these hidden messages inside media. The field of steganalysis has emerged to meet this need. Steganalysis can be defined as, "the art and science of detecting steganography".The main goals of steganalysis are to detect steganography and to detect what method (or piece of software) was used to hide the information.

Watermarking is a technique to embed some symbol data (or evidence data) in some valuable data such as an image, voice, music, video clip, etc. It protects the original data from alteration. The embedded data can be small, but it must be very protective (robust) to filtering, re-sampling and other data alteration techniques. In watermarking the "external data" is valuable, but the embedded data is not so valuable. While in steganography, the embedding capacity should be large enough. But, the embedded data can be fragile rather than robust. This is because the embedded data is secret. It should be better destroyed by attacking than robust enough to stay long. The external data is just a dummy. So, the requirement and the objective of watermarking and steganography are very opposite even if they belong to the same information hiding technique.

II. AUDIO STEGANOGRAPHY

Like the document images, the sound files may be modified in such a way that they contain hidden information, like copyright information; those modifications must be done in such a way that it should be impossible for a pirate to remove it, at least not without destroying the original signal. The methods that embeds data in sound files use the properties of the Human Auditory System (HAS). The HAS perceives the additive random noise and also the perturbations in a sound file can also be detected. But there are some "holes" we can exploit. While the HAS have a large dynamic range, it has a fairly small differential range. As a result, loud sounds tend to mask out quiet sounds. And there are also some distortions that are so common that the HAS ignores them. Suppose if we observe the audio wave file before embeddeding and after embeddeding secret data respectively in Fig 2.1 and Fig 2.2 the human auditory system can't recognize the small change.

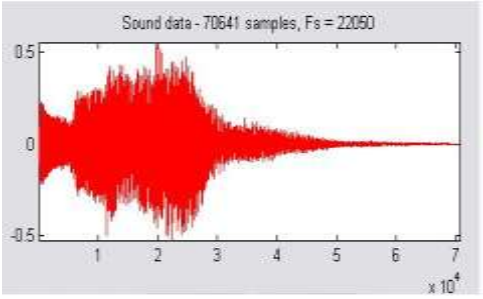

Fig 2.1: Before embedding the text

The digital sound is obtained from the analog sound by converting it to digital domain. This process implies two sub processes: sampling and quantization. Sampling is the process in which the analogue values are only captured at regular time intervals.

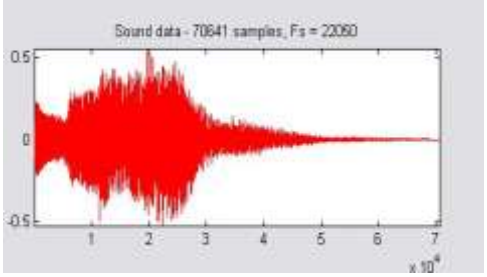

Fig 2.2: After embedding the text

Quantization converts each input value into one of a discrete value. Popular sampling rates for audio include 8 kHz, 9.6 kHz, 10 kHz, 12 kHz, 16 kHz, 22.05 kHz and 44.1 kHz. The most popular file formats for sounds are the Windows Audio-Visual (WAV) and the Audio Interchange File Format (AIFF). There are also compression algorithms such as the International Standards Organization Motion Pictures Expert Group-Audio (ISO MPEG-AUDIO). When developing a data-hiding method for audio, one of the first considerations is the likely environments the sound signal will travel between encoding and decoding. There are two main areas of modification which we will consider. First, the storage environment, or digital representation of the signal that will be used, and second the transmission pathway the signal might travel [3, 4]. In order to conceal secret messages successfully, a variety of methods for embedding information in digital audio have been introduced. These methods range from rather simple algorithms that insert information in the form of signal noise to more powerful methods that exploit sophisticated signal processing techniques to hide information. This section of the paper is organized as follows. First, the clarification of the Audio Environment. Secondly, this section describes a one of the wide range of techniques that have been used in Audio Steganography.

III. TECHNIQUE FOR DATA HIDING IN AUDIO

Least significant bit (LSB) coding is the simplest way to embed information in a digital audio file. By substituting the least significant bit of each sampling point with a binary message, LSB coding allows for a large amount of data to be encoded. The following Fig 3.1 illustrates how the message 'HEY' is encoded in a 16-bit CD quality sample using the LSB method:

In LSB coding, the ideal data transmission rate is 1 kbps per 1 kHz. In some implementations of LSB coding, however, the two least significant bits of a sample are replaced with two message bits. This increases the amount of data that can be encoded but also increases the amount of resulting noise in the audio file as well. Thus, one should consider the signal content before deciding on the LSB operation to use. For example, a sound file that was recorded in a bustling subway station would mask low-bit encoding noise. On the other hand, the same noise would be audible in a sound file containing a piano solo.

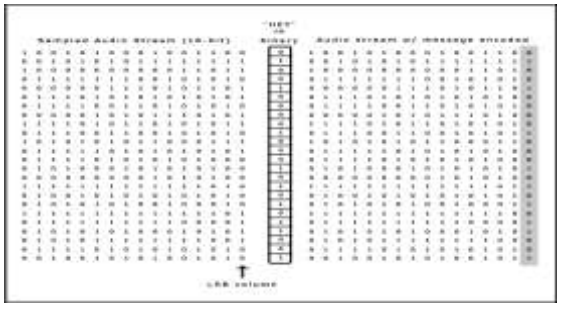

Fig 3.1: Embedding secret text in LSB position of Audio Stream

To extract a secret message from an LSB encoded sound file, the receiver needs access to the sequence of sample indices used in the embedding process. Normally, the length of the secret message to be encoded is smaller than the total number of samples in a sound file. One must decide then on how to choose the subset of samples that will contain the secret message and communicate that decision to the receiver. One trivial technique is to start at the beginning of the sound file and perform LSB coding until the message has been completely embedded, leaving the remaining samples unchanged. This creates a security problem, however in that the first part of the sound file will have different statistical properties than the second part of the sound file that was not modified. One solution to this problem is to pad the secret message with random bits so that the length of the message is equal to the total number of samples. Yet now the embedding process ends up changing far more samples than the transmission of the secret required. This increases the probability that a would-be attacker will suspect secret communication.

A more sophisticated approach is to use a pseudorandom number generator to spread the message over the sound file in a random manner. One popular

approach is to use the random interval method, in which a secret key possessed by the sender is used as a seed in a pseudorandom number generator to create a random sequence of sample indices. The receiver also has access to the secret key and knowledge of the pseudorandom number generator, allowing the random sequence of sample indices to be reconstructed. Checks must be put in place, however, to prevent the pseudorandom number generator from generating the same sample index twice. If this happened, a collision would occur where a sample already modified with part of the message is modified again. The problem of collisions can be overcome by keeping track of all the samples that have already been used. Another approach is to calculate the subset of samples via a pseudorandom permutation of the entire set through the use of a secure hash function. This technique ensures that the same index is never generated more than once.

There are two main disadvantages associated with the use of methods like LSB coding. The human ear is very sensitive and can often detect even the slightest bit of noise introduced into a sound file, Second disadvantage however, is that this is not robust. If a sound file embedded with a secret message using either LSB coding was resample, the embedded information would be lost. Robustness can be improved somewhat by using a redundancy technique while encoding the secret message. However, redundancy techniques reduce data transmission rate significantly.

Advantages of Proposed Method:

- Secrecy - The embedding secret text is known to the sender and the receiver only.
- Imperceptibility – The medium after being embedded with the covert data is indiscernible from the original medium. One cannot become suspicious of the existence of the covert data within the medium.
- High capacity - The maximum length of the covert message that can be embedded as long as possible depending on the size of the covering medium (audio).

A. LSB based Audio Steganography

- In the current endeavor, an audio file with ".wav" extension has been selected as host file.
- And assumed that the least significant bits of that file should be modified without degrading the sound quality because those LSB positions have a very little contribution in the audio perception.

B. Embedding Secret Text in Audio

The process of embedding the secret text in an audio file is as follows.The input audio file is a Windows Audio-Visual (WAV) file, which has 16-bit linear quantized digital audio samples.

Then separate the header and data parts.

(1)Store the size of the secret text in the header part.

(2)Store the LSB positions of the data part in alternate samples of LSB data part.

Thus Stego audio file created, which is a WAV file having hidden text but with out any changes in audibility of the Cover file.

Embedding Process:

The steganography technique used is LSB coding.

The audio file consists of data in bytes.

To encode the message, we first find the length of the string.

The offset in the original file, from which the encoding process must start, is by default set to 500. This is done because, the WAV file has a header in the initial offsets and if that header is tampered with, the destination file will not be able to access its header in the appropriate format.

Encode that length which can be upto 256 characters into the 1st 8 bytes of the audio file. This will assist us in the decoding process.

Take each character from the message string, convert it into byte and change the LSB of the next 8 bytes of the audio file as per each of the bit of the character type.

Repeat the same procedure till the message string gets exhausted.

Thus on writing byte after byte to the new file, we get a new audio file ―output.wav having message hidden into it which can be sent to the receiver without any fear of eavesdropper.

C. Extraction of Secret Text from StegoAudio

The Process of extracting the hidden text from WAV audio file is as follows.Input file is Stego audio (WAV) file and then separated header and data parts. Header consists of size of secret text. Store LSB of data part and perform left shift of previous bit. Then convert binary to ASCII values. Thus secret text can be extracted.

Extraction Process:

The audio file ―output.wav which has the message hidden into it.

Select From the offset that was specified at the sending side(i.e. 500) , take the LSB of the next 8 bytes to get the length of the message (that was encoded in the first 8 bytes from the given offset) which will help us to get the encoding message only from the next 8 * length bytes of audio file.

Create a byte from the LSB of the next consecutive 8 bytes and go on printing each of the character of the message string in the textbox.

Continue this process till the length of the string is reached. Hence finally we get the hidden message from the received audio file into the provided textbox.

- Thus we have achieved the process of decoding a message from the audio file.

IV. RESULT AND ANALYSIS

The results are taken as screen shots using MATLAB GUI tool.

Step1: Select 'Hide Text' radio button, type the text in text field of below GUI window which we want to hide

Fig 4.1: GUI window with secret text

After compiling the program in MAT Lab, a GUI (Graphical User Interface) will appear in which two selection fields 'Hide Text' and 'Recover Text' and also a button 'Select wav File'. It also has an inactive button 'Hide The Text' as shown in Fig 4.1. Select 'Hide Text' radio button and then put the data to be hidden, which is a secret text has to be received by only the intended community.

Step2: Select any '.wav' file in which we want to hide the secret text. Here 'windows XP Shutdown.wav' is selected as carrier to hide text

Fig 4.2: Selecting carrier audio file

After putting the text in text editor then click on the button 'Select wav File' and choose any WAV file, in which the secret text has to be hidden and open that file as shown in Fig 4.2.

Step3:After selecting carrier wav file 'Hide The Text' button will be highlighted

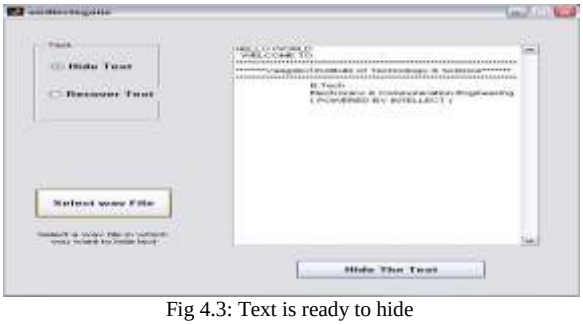

Fig 4.3: Text is ready to hide

Before selecting WAV file the button 'Hide The Text' be inactive but after selecting the WAV file in which secret data has to be hidden, then the 'Hide The Text' button highlighted.

Step4: Now by pressing 'Hide The Text' button a new audio file with random named will generate. Here 'new1901.wav'named wave file generated which have the hidden secret text which can be transmitted to remote destination.

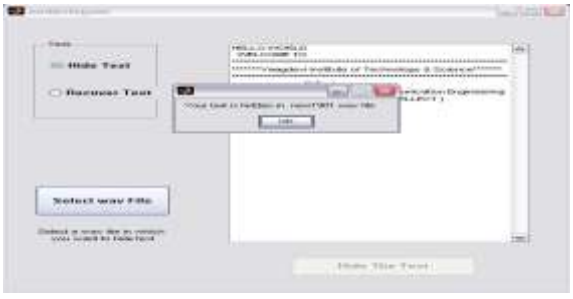

Fig 4.4: Generated stegoaudio .wav file

Now click on the button 'Hide The Text' then a WAV file, which has the secret text inherently will be created. This WAV file can be created in the folder of the source WAV file and it has to be transmitted to remote destination manually.

Step5: If observe the audio wave files before and after steganography the wave sound files are as shown below

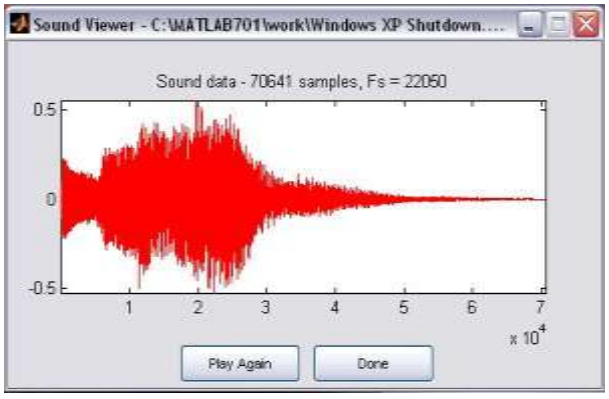

Fig 4.5: Original .wav audio structure

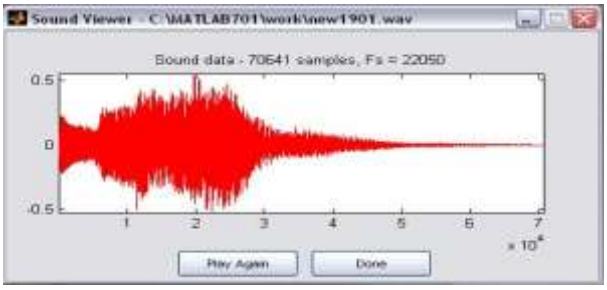

Fig 4.6: Generated Stegoaudio .wav file structure

If we observe the above original and generated Stego WAV file structures, we can't recognize any difference between them.

Step6: To extract secret text from stego audio select 'Recover Text' radio button and select the stego audio file here 'new1901.wav' by pressing 'Select wav File' button

Fig 4.7: Retrieving secret text from stegoaudio

Now compile the program in MAT Lab then again a GUI will appear then select the field 'Recover Text' and click on the button 'Select wav File'. Choose the WAV file in which secret data had been hidden inherently.

Step7: Finally the secret text has been successfully extracted from'new1901.wav' audio file, which is displayed on same text field as shown below.

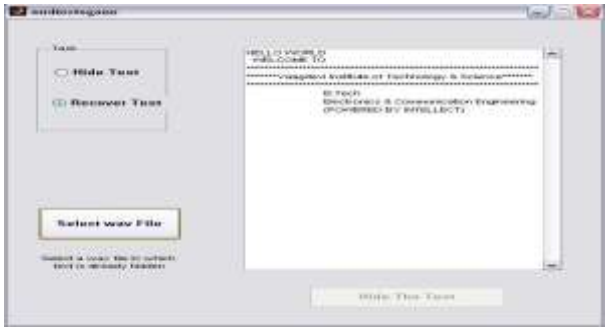

Fig 4.8: Extracted secret text from stegoaudio

Then after selecting WAV file which has hidden data, the secret text had been generated as shown in Fig 4.8. Thus secret data is difficult to intruders for even imagination of getting the information.

V. CONCLUSION

We proposed a lossless audio steganography system, in which the LSB technique was used to get high data hiding capacity and low perceptibility. So by using this technique the capacity of the data to be hiding is increased and also the clarity of the covering medium (.wav audio) doesn't change though it has hidden text. This is an advanced technique to image steganography.

Future Scope:

Security to this system can be boosted by using a concept called cryptography (i.e., encryption and decryption) to the text data. The system can be further developed to hide secret image in cover audio.

REFERENCE

[1] Hiding in Plain Sight: Steganography and the Art of Covert Communication Eric Cole Ronald D. Krutz, Consulting

[2] Editor Eric Cole, Hiding in Plain Text, Wiley Publishing, Inc.:2003

[3] W. Bender, D. Gruhl, N. Morimoto and A.Lu, "Techniques for data hiding," *IBM Systems Journal*, Vol. 35, Nos. 3 & 4, pp. 313-336, 1996.

[4] M.D. Swanson, M. Kobayashi, and A.H. Tewfik, "Multimedia data-embedding and watermarking technologies," Proc. IEEE, Vol. 86, pp. 1064-1087, June 1998.

[5] K. Gopalan, S. Wenndt, A. Noga, D. Haddad, and S.Adams, "Covert Speech Communication Via Cover Speech By Tone Insertion," Proc. of the 2003 IEEE Aerospace Conference, Big Sky, MT, Mar. 2003 (on CD).

[6] K. Gopalan, et al, "Covert Speech Communication Via Cover Speech By Tone Insertion," U.S. Patent applied for, Oct. 2003.

[7] R.J. Anderson and F.A.P. Petitcolas, "On the limits of steganography," IEEE J. Selected Areas in Communications, Vol. 16, No. 4, pp.474-481, May 1998.