Rochester Institute of Technology
RIT Scholar Works

Presentations and other scholarship | Faculty & Staff Scholarship

7-2012

Audio Steganography Using High Frequency Noise Introduction

David Wheeler
Rochester Institute of Technology

Daryl Johnson
Rochester Institute of Technology

Bo Yuan
Rochester Institute of Technology

Peter Lutz
Rochester Institute of Technology

Follow this and additional works at: https://scholarworks.rit.edu/other

Recommended Citation
Wheeler D., Johnson D., Yuan B., and Lutz P. Audio Steganography Using High Frequency Noise Introduction. In SAM'12 - The 2012 International Conference on Security and Management (Las Vegas, NV, USA, July 2012).

This Conference Paper is brought to you for free and open access by the Faculty & Staff Scholarship at RIT Scholar Works. It has been accepted for inclusion in Presentations and other scholarship by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.

Audio Steganography Using High Frequency Noise Introduction

David Wheeler, Daryl Johnson, Bo Yuan, Peter Lutz
B. Thomas Golisano College of Computing & Information Sciences
Rochester Institute of Technology, Rochester NY
{dbw3113,daryl.johnson,bo.yuan,peter.lutz}@rit.edu

***Abstract*—This paper presents a new method of audio steganography that allows character data to be encoded into audio in a way that is indiscernible to prying third-parties. Unlike typical methods of audio steganography that propose storage by modifying the least significant bits or the phase of the audio data, this approach makes use of frequency ranges that are undetectable to the human ear. The method proposed in this paper provides for a reasonably high-bandwidth and is resistant to common detection and prevention techniques.**

I. Introduction

The act of being able to hide imperceptible information within digital media has become an area of increasing interest in the computing world. Data hiding techniques have many potential applications such as covert communication, hiding executable data, watermarking and digital rights management [2]. The method used to conceal data for all of the above situations is called steganography. Steganography, which literally means "concealed writing", is a method of covert communication that has existed for thousands of years. Today steganography has been adapted to the digital era and can be implemented in pictures, audio, text and even other forms of digital multimedia as well [3].

Digital steganography involves the use of two entities that make up the transfer file. The first entity is the cover object, which is the overt data being sent, and the second entity is the stego object, which is the secret or covert message embedded in the cover object. In addition, digital steganography has two rudimentary requirements that must be fulfilled. The first requirement is that the stego object is virtually imperceptible to any third-parties who may obtain the file or files, whereas the second requirement is that there must be a reasonably high bandwidth for the stego-data.

In this internet era, digital media is commonly transferred over the internet both through individual file transfer and streaming of raw data. Given these new developments there has been an increasing focus on embedding hidden information into audio. There are several classic ways that are used to hide information in audio. Least Significant bit (LSB) encoding, echo hiding, phase coding, and spread spectrum coding are among the most common techniques used [3]. This paper proposes a novel concept of audio steganography that encodes binary information into high-frequency signals. The proposed steganography implementation is a sort of hybrid mix of several of the more common methods, and combines a known technique with a new one. The maximum range that a human ear can hear is between the frequencies of 20 Hz and 20KHz. However, natural high-frequency hearing loss over time and the lack of speaker fidelity makes this perceptible frequency much lower [9]. This means that while audio files carry digital information all the way up to 20 KHz, some of the highest frequencies will be completely imperceptible to the human ear. The method illustrated in this paper will be both imperceptible to humans and reasonably resistant to preventative software techniques.

II. Related Work

As mentioned above there are four main categories of audio steganography that are commonly used to hide information into auditory data: least significant bit coding, echo coding, phase coding and spread spectrum coding. Each method varies in implementation, bandwidth, and covertness. They all have advantages and disadvantages and are typically used for differing applications.

A. Least Significant Bit Encoding

As the name implies least significant bit coding (LSB encoding) deals with modifying the least significant bit of each audio frame in order to encode binary information. This is an inherently simple task and has the advantage of high bandwidth but is unfortunately easy to prevent. Small format changes that occur during file conversion, compression, or through preventative techniques, can easily contaminate the hidden data [3]. There have been proposed LSB coding methods, however, that utilize higher level bits in same fashion as the LSB method that are robust against some issues that are present in this category of audio steganography [2].

B. Echo Hiding

The process of echo hiding involves inserting echoes with varying characteristics into discrete audio signals. Three echo parameters, amplitude, decay rate and offset (essentially the delay time of the echo) are applied in varying ways in order to successfully encode binary information. This method is very covert as each echo occurs below the audible limit of the human ear. The disadvantage of this technique is that the process can sometimes yield a noticeable mix of echoes which increases the risk for detection [3].

C. Phase Coding

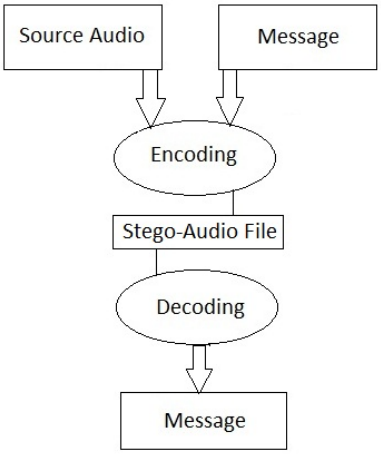
Phase Coding addresses the issue of covertness and detectability as components of sound (modified phase) are much more difficult for the human ear to perceive than the addition or subtraction of noise. To implement this method the audio file is broken down into discrete chunks, separated into phase groups and then shifted according to the binary data being encoded. The main problem with this process is that modifications of audio phase allows for a relatively low quantity of stego-data. Thus this technique is relatively low bandwidth and is typically used for applications such as watermarking or copyrighting of audio files [3].

D. Spread Spectrum Coding

The final type of audio steganography worth mentioning is Spectrum Coding. Spectrum Coding takes bits of information and randomly spreads them over the entire frequency spectrum [5]. It is similar to LSB but is more robust against steganalysis techniques. This procedure, however, is still somewhat vulnerable to detection as it can introduce noise into the audio file [3].

III. Method

Unlike some of the common steganography techniques listed above that try to mask the audio signals this approach uses more of a hidden-in-plain-sight kind of approach. While all wave files hold information between the ranges of 20 Hz and 20 KHz, not only are frequencies at the uppermost range rarely used, but they are also nearly impossible to perceive by the human ear. Given these characteristics of wave files, high frequencies can be injected into the cover audio file in order to produce a concealed binary stego signal. The basic approach to the process of embedding and reading binary information is presented below.



The method of encoding binary information in this manner is relatively simple. First, an audio file must be broken into discrete chunks known as frames. The character data to be hidden in the audio is then converted into binary information and mixed into the audio frames. The pseudo code algorithm used for mixing binary into audio is given below:

```
frame_buffer = frames.get(bufferSize)
for each binary_bit
if binary_bit == 1
   then addHighPowerHFT into frame_buffer
if binary_bit == 0r
   then noHFTInjection into frame_buffer
```

Essentially a group of frames is obtained from the audio file and then a HFT (high-frequency tone) is added to that buffer of frames depending on whether the next bit to be encoded is a 0 or a 1. One important point to note is that the size of the buffer must stay constant throughout the encoding process for the purposes of decoding. The buffer size can range anywhere from fifty to thousands of frames, which equates to between one and a hundred milliseconds. The disadvantage to increasing the buffer size is that the audio data will have a lower potential bandwidth.
The process used to encode audio is given below:

```
frames = Read_WaveFile(fileName)
characters = Read_Character_Data()
binary_bits = Convert(Characters)
foreach bit in binary_bits
   add_HFT(bit)
output_file = Write_WaveFile
```

In Fig1 a short one hundred and sixty millisecond clip of the cover audio file is shown. The graph measures the power levels at various frequencies over a span of time (160 MS). In Fig2 the same audio clip is shown once it has undergone the proposed HFT encoding. In Fig2 the levels of the HFTs would appear to be significant enough to be detected by human perception but are not, due to the fact they are located just past the peak of a humans auditory perception.

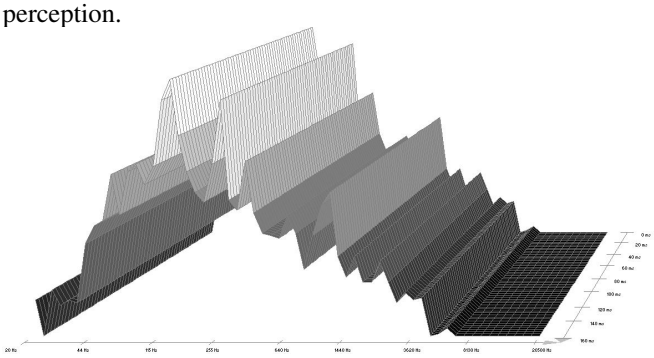


Fig1: 160 ms of cover audio

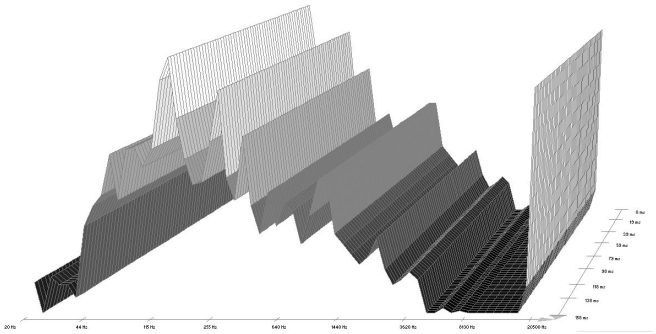


Fig2: 160 ms of audio with encoded HFTs

Decoding the stego object from the encoded wave file simply requires reversing the encoding process and applying a Fourier transform to the encoded frames. The decoding algorithm is given below:

```
frames = Read_WaveFile(stegoFileName)
fftData = FFT(frames)
data = HighPass(fftData
foreach buffer in data
   bits += get_Bit(buffer)
characters = Convert(bits)
```

In these steps the wave file with the hidden stego object is initially read into frames. In order to get the frequency breakdown of the frames a fast Fourier transform algorithm is applied to the data. A high-pass filter is applied to the frequency data which will only allow the HFT stego data to be read into the data array. Depending on the buffer size the data is read and each buffer is interpreted either as a 0 or 1 value. The binary information is then converted back into its character information and the hidden message is then fully decoded.

IV. Experimental results

The proposed method described above was implemented in a java program called AudioStego which allowed for encoding of character data into wav-forFmat audio files. The program made use of a WavFile java class for basic input and output operations of wave files [1] and was otherwise implemented with standard java libraries. A multitude of different songs and audio information were tested. Each audio clip featured different dynamic and spectral ranges. All audio was sampled at 44.1 KHz with 16 bits of depth and the length of various audio files ranged from ten seconds up to five minutes in length.

Subjective tests were done on each of the tested audio files by several individuals in order to determine how discernable the introduction of HFTs were in the cover audio. The tests compared an original cover audio file with an audio-stego version that implemented varying buffer sizes and power levels for the encoded HFTs. Table 1 illustrates the different characteristics of HFTs used and their corresponding inaudibility as reported by the listeners.

TABLE I
Subjective Listening Results

Buffer Size	Power Level	inaudibility
100	High	Sometimes
1000	High	Sometimes
100	Mid	Yes
1000	Mid	Yes
100	Low	Yes
1000	Low	Yes

In all low power and mid-power cases it was reported that the additional HFTs in the audio signal were indiscernible. In both high-power cases some subtle white noise and crackling was discernable in the audio-stego file to some of the listeners. When inserting very low-power HFTs, data could be occasionally be decoded incorrectly due to noise in the audio signal. It also appeared that size of frame buffers played no noticeable role in detection. Given this data, the best apparent characteristics for indiscernible HFT injection would occur at the mid-power level and with buffers of 100 frames (this maximizes the bandwidth of the file).

V. Detection and Prevention

There is a somewhat limited amount of research in the field of audio steganalysis (the analysis and detection of steganography). This has to do with the fact that many audio steganography schemes are quite advanced and the nature of high-bandwidth audio streams makes it difficult to produce consistent analyzing tools [4].

There are two main categories to talk about when referring to steganalysis. The first is detection and the second is prevention. Detection is particularly difficult in this case as HFT insertion is essentially in a category of its own for audio steganography methods. Prevention is much easier as any attack against the stego file that offsets the HFTs length, frequency, or power level could possibly disrupt the hidden data.

Along the lines of steganalysis, there are several major detection methods used that are worth mentioning. The first is the use of statistical distance measurements for steganalysis. This idea essentially measures the difference between cover audio and their stego-audio signal equivalents with the use of common techniques such as echo and phase coding. The statistical data that is generated is then compared against the stego-audio in question [7]. Another similar technique called audio steganalysis based on Hausdorff Distance finds the Hausdorff distance measure between a cover audio signal and its stego-audio equivalent. It decomposes the audio into wavelet coefficients and the Hausdorff distances between the wavelets are used to train a classifier about the difference between cover audio and stego-audio signals [3]. This type of steganalysis would prove to be ineffective against HFT addition since the algorithms would be gathering statistics from unrelated steganography algorithms.

While HFT addition is seemingly resistant to detective methods, it is much less robust against preventative meth-

ods. Since it is computationally difficult to analyze a high-bandwidth audio stream, recent research has been conducted to attack steganography by slightly distorting audio data. Some of the techniques that are being implemented are frequency shifting, white noise addition, and variable time delay [6]. All three of these techniques can potentially put the HFT method at risk. The main issue with attacking a stego-audio signal is that the attacker wishes to maintain the integrity of the audio as much as possible. This works out in favor of HFT as smaller changes to power levels, sample time adjustments, and frequency domains are less likely to distort the encoded binary information. In the case of variable time delay a common delay is no more than 10 milliseconds per second of audio data [6]. That level of change is unlikely to distort the stego-audio enough to distort the encoding. These preventative techniques pose a potential threat to the HFT method but do not completely invalidate it in many cases.

VI. Conclusion

There are a number a number of proven methods for applying steganography to hide information within audio data. In this paper a new and simple approach was investigated that made use of rarely used and difficult to detect frequency ranges in raw audio files. It was shown through implementation and subjective experimentation that this novel method can effectively transmit binary information, by way of frequency injection, unnoticed to an end user. While the proposed method suffers from several drawbacks as far as robustness, however, it also serves the user with a channel for high bandwidth data transmission.

VII. Future Work

There are several areas in which this proposed method could potentially be expanded on in the future. One particular addition that could be made to obfuscate binary data would be to apply an encryption algorithm on top of the binary data before encoding it in the audio file. A potential technique that could be applied would be to use a common encryption scheme such as AES and to apply it across the data before encoding and after decoding [8]. This would place a level of security on top of the hidden information so that any third-party that discovers the encoded bits would be unable to discern any kind of meaning from them.

Currently the proof of concept program, AudioStego, only encodes binary data to the raw wave file format. Due to the unwieldlieness of uncompressed audio data most auditory information is transmitted in compressed formats. Formats such as MPEG Audio Layer III(mp3), Windows Media Audio formats(wma) and Vorbis(ogg) are prime examples. As with many lossy formats MPEG Audio Layer III ignores potentially repetetive or unimportant frames and frequency bands [10]. Encoding data using HFTs would be harderto implement since the signal would have to be prominent enough so that it is not lost when the raw file undergoes lossy compression. There would either be a higher level of detectability or a higher error rate when decoding bits. To further complicate the issue of lossy-file conversion, formats such as MP3 significantly reduce signals present at very high frequency ranges. A quick experiment was conducted where a stego-audio file was converted into its lossy MP3 equivalent, converted back to a wav format and then was decoded. The HFTs were diminished to the point where the binary data was unable to be read. In order to avoid this conversion issue the AudioStego program would need to directly modify the converted format in order to inject HFT's in a manner in which they could be decoded.

References

[1] Dr. Andrew Greensted, http://www.labbookpages.co.uk/audio/wavFiles.html

[2] Cvejic, N., and T. Seppanen.*Increasing Robustness of LSB Audio Steganography Using a Novel Embedding Method.* In Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference On, 2:533 –537 Vol.2, 2004.

[3] Ishaque, M. Qudus Khan, F. Abdul Sattar, S. *Investigation of Steganalysis Algorithms for Multiple Cover Media.* Ubiquitous Computing and Communication Journal. Vol 6, no 5, October 2011.

[4] C. Kraetzer and J. Dittmann, *Pros and Cons of Mel-cepstrum based Audio Steganalysis using SVM Classification.* Lecture Notes in Computer Science, vol. 4567, pp. 359 – 377, January 2008.

[5] H. Matsuoka, *Spread Spectrum Audio Steganography Using Sub-band Phase Shifting.* in Intelligent Information Hiding and Multimedia Signal Processing, 2006. IIH-MSP 06. International Conference on, 2006, pp. 3 – 6.

[6] M. Nutzinger, *Real-time Attacks on Audio Steganograhy.* Journal of Information Hiding and Multimedia Signal Processing. Vol. 3, no. 1, January 2012.

[7] H. Ozer, I. Avcibas, B. Sankur and N. D. Memon, *Steganalysis of Audio based on Audio Quality Metrics.* Proceedings of the Conference on Security, Steganography and Watermarking of Multimedia, Contents V, vol. 5020, SPIE, pp. 55 – 66, January 2003

[8] Sridevi, R. Damodaram, A. Narasimham, S. *Efficient Method of Audio Steganography by Modified LSB Algorithm and Strong Encryption Key with Enhanced Security.*Journal of Theoretical and Applied Information Technology. vol. 5, no. 6, pp. 768 – 771, June 2009.

[9] hypertextbook.com/facts/2003/ChrisDAmbrose.shtml

[10] MP3 Audio Format: http://wiki.hydrogenaudio.org/index.php?title=MP3