MP3Stego: Hiding Text in MP3 Files

Steganography techniques involving audio file formats appear to be increasing in popularity. This paper discusses MP3Stego, a steganographic tool that operates within the MP3 encoding process.
MP3Stego: Hiding Text in MP3 Files
Mark Noto
September 15, 2001
Practical Version 1.2f

With the widespread popularity of the MP3 file format and its ubiquitous presence on storage media and the Internet, it was only a matter of time before it joined ranks alongside other steganographic mediums. Johnson defines steganography as “the art of passing information in a manner that the very existence of the message is unknown.” Unlike cryptography, which is easily detected (but hopefully difficult to decipher), steganography is an attempt to conceal the very presence of the hidden message. When the two techniques are combined, as they are in MP3Stego, they present a formidable challenge.

Jossi points out that steganography “has become a popular tool for secretly sending and receiving digital messages.” The popularity of steganography is poignantly driven home when Jossi notes that terrorist Osama bin Laden “reportedly relies on stego files to transmit maps, photos and instructions outlining future targets via pornographic bulletin boards and sports chat rooms.”

Audio Steganography Basics

Weeks notes that:

Digital steganography is based on the fact that artifacts like bitmaps and audio files contain redundant information. That’s why lossy compression techniques such as JPEG and MP3 work. Such techniques eliminate part of the redundancy, allowing the image or audio file to be compressed. The idea behind [digital] steganography is that instead of eliminating [all of] the redundant information, you replace [some of] it with other data.

One should not underestimate the difficulty of hiding data in audio signals. Bender explains the depth of the problem when he notes that “…the human auditory system (HAS) operates over a wide dynamic range. The HAS perceives over a range of power greater than one billion to one and a range of frequencies greater than one thousand to one.” Thankfully, or perhaps not depending upon whether the good guys or bad guys are sending messages, the HAS is fairly poor with respect to its’ differential range. This means that loud sounds tend to mask out softer sounds easily. Psychoacoustic models (as described later in the paper) can exploit this weakness. This information should

1 Johnson, p. 273
2 Jossi, p. 1
3 Ibid
4 Weeks, p. 2
5 Bender, p. 323
also be useful in helping one to select appropriate audio for encoding; Megadeath or NIN may be a much better choice of audio source than the Kronos String Quartet!

Hiding data in audio can be done a number of ways. One such method is low-bit encoding. When Jossi incorrectly defines steganography as “the practice of encoding data into the least significant bit of another file,” he is actually describing the low-bit encoding method. Low-bit encoding replaces the least significant bit of information in each sampling point with a coded binary string. While this method can be efficiently employed to encode fairly large amounts of hidden data in a given audio signal, it does so at the expense of introducing significant noise at theoretical upper limits.

A second method of data hiding in audio is phase coding. Shankaranarayanan explains that phase coding “...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.” While phase coding is very quiet in comparison to low-bit encoding, it is also a more complicated method.

The third method of data hiding in audio is spread spectrum. This method spreads the encoded data across as much of the frequency spectrum as possible, making it very difficult for an adversary to find the data unless they have access to, or can reconstruct, the psuedorandom signal used to spread the data across the frequency range.

The last method of data hiding in audio is echo data hiding. Sellers notes that “Echo data hiding embeds data into a host signal by introducing an echo. The data are hidden by varying three parameters of the echo: initial amplitude, decay rate, and offset. As the offset between the original and the echo decreases, the echo blends.” In a method similar to vocal overdubbing, where two tracks of vocals are combined to “fatten” up a vocal, this echo blending is perceived as part of the original sound, not an adjunct sound. At some point, the echo and original sound are not perceived as “separate” by the human ear.

Each of these methods relies upon exploiting weaknesses of human auditory perception, in either the acoustic or psychoacoustic domains. As we will see in the following section, ISO-MPEG Audio Layer-3 uses these same weaknesses to hide the fact that it drops significant amounts of audio information.

**MP3 in a Nutshell**

In order to understand how MP3Stego works, we need to discuss ISO-MPEG Audio Layer-3 (or MP3 to the rest of us). ISO-MPEG Audio Layer-3 was a result of work done

---

6 Jossi, p. 1  
7 Shankaranarayanan, p.4  
8 Sellers, sec. 8.2.4
by Fraunhofer IIS-A in the area of perceptual audio coding. Some explanation of perceptual audio coding is in order:

In a perceptual audio coder, the codec does not attempt to retain the input signal exactly after encoding and decoding, rather its goal is to ensure that the output signal sounds the same to a human listener. The primary psychoacoustic effect that the perceptual audio coder uses is called 'auditory masking', where parts of a signal are not audible due to the function of the human auditory system. The psychoacoustic model analyses the input signals within consecutive time blocks and determines for each block the spectral components of the input audio signal by applying a frequency transform. Then it models the masking properties of the human auditory system, and estimates the just noticeable noise-level, sometimes called the threshold of masking. In its quantization and coding stage, the encoder tries to allocate the available number of data bits in a way that meets both the bitrate and masking requirements.9

In layman’s terms this means that the audio signal contains a significant portion of information that can be discarded without the average listener noticing a change.

Two nested iteration loops are utilized in the Layer-3 encoder; the inner iteration loop (rate loop) and the outer iteration loop (noise control/distortion loop). Without going into a long, detailed explanation of the encoding process, these two loops are involved in a “negotiation” for digital real estate. The encoding process is illustrated below:


9 Thom, p.2
The steganographic tool discussed in this paper, MP3Stego, operates within the MP3 encoding process. It takes advantage of this process to insert hidden data during data compression.

**MP3Stego**

Fabien A. P. Petitcolas, creator of MP3Stego, notes that “When looking at the steganographic tools available on the Net, it occurred to me that nothing had been done to hide information in MP3 files, that is sound tracks compressed using the MPEG Audio Layer III format.” Considering the fact that steganography is most effective when it doesn’t raise suspicions, a ubiquitous file format such as MP3 is highly desirable as a host for hidden data. The fact that MP3 employs a lossy compression algorithm also contributes to its’ effectiveness as a steganographic medium.

MP3Stego works in the following manner:

MP3Stego will hide information in MP3 files during the compression process. The data is first compressed, encrypted and then hidden in the MP3 bit stream. The hiding process takes place at the heart of the Layer III encoding process namely in the inner_loop. The inner loop quantizes the input data and increases the quantizer step size until the quantized data can be coded with the available number of bits. Another loop checks that the distortions introduced by the quantization do not exceed the threshold defined by the psychoacoustic model. The part2_3_length variable contains the number of main data bits used for scalefactors and Huffman code data in the MP3 bit stream. We encode the bits as its parity by changing the end loop condition of the inner loop. Only randomly chosen part2_3_length values are modified; the selection is done using a pseudo random bit generator based on SHA-1.

Referring back to the diagram of the MP3 encoding process on page 3, we can see that MP3Stego operates within the boxed “iteration loops” process. Anderson and Petitcolas co-authored a paper that discusses the “power of parity.” This approach is explained as follows:

We use our one-time pad (or keystream generator) to select not one pixel but a set of them, and embed the ciphertext bit as their parity. This way, the information can be hidden by changing whichever of the pixels can be changed least obtrusively. There is an interesting tradeoff: the more bits in the selection channel (i.e., the greater the value of $k$), the more bits we can hide in the ciphertext. In practice our selector channel will be a cryptographic

---

10 Petitcolas, p. 1
11 Ibid
psuedorandom number generator, and we can draw from it as many bits as we like.\textsuperscript{12}

This concept is applied practically in MP3Stego.

\textbf{MP3Stego in Action}

MP3Stego, version 1.1.15 for Windows 95/98/NT, operates via command line interface. Both MP3Stego and a separate graphic interface are available at \url{http://www.cl.cam.ac.uk/~fapp2/steganography/mp3stego/index.html}. The command line interface mode allows the user to both encode and decode files. To encode a file the following syntax is used:

```
encode –E secret_text.txt –P passphrase hells_bells.wav hells_bells.mp3
```

This compresses hells.bells.wav along with secret_text.txt into hells_bells.mp3, encrypting the hidden text using the passphrase “passphrase.”

Decoding the file is accomplished using:

```
Decode –X –P passphrase hells_bells.mp3
```

This uncompresses hells_bells.mp3 into hells_bells.mp3.pcm and decrypts and saves the hidden text as hells_bells.mp3.txt.

MP3Stego uses 3DES encryption and the SHA-1 hash. 3DES and SHA-1 are both widely employed and well tested. Lumb notes, “A good steganographic system should be based on the assumption that outsiders have full knowledge of its design and implementation details. The only missing information is a short exchangeable random number sequence, known as the secret key.”\textsuperscript{13} MP3Stego employs a passphrase and 3DES to protect the hidden data payload. SHA-1 is employed to generate pseudo random bits for use in the hiding process.

One particular idiosyncrasy that needs to be dealt with when using MP3Stego is the fact that the .wav file being used must be a mono 16-bit file encoded with pulse code modulation at 44100Hz. The program does not appear to provide any way to use stereo .wav files (the normal format for .wav files). In order to get much use out of the software it’s necessary to have software that can encode audio in the format that MP3Stego requires. This is somewhat inconvenient, but since the software is more of a proof-of-concept product than a well-developed offering it is not unexpected. It is likely that second generation products are, or will shortly become, available which incorporate the kinds of features that will make this software acceptable to less sophisticated users.

\textsuperscript{12} Anderson, p. 479
\textsuperscript{13} Lumb, p.1
While MP3Stego is primarily a steganographic tool, the author notes that it can also be employed as a copyright marking system for MP3 files. The watermark can be removed by uncompressing and then recompressing the bit stream, but not without severely degrading the quality of the resulting audio file. One other test of the quality of a watermark defined by deCarmo is that "All watermarks must be unobtrusive."\(^{14}\) MP3Stego meets this test.

Another important measure of the quality of a steganographic tool is its resistance to steganalysis. Steganalysis is defined by Johnson as "...the art of discovering and rendering useless such covert messages."\(^{15}\) One of the normal methods used for detecting the existence of hidden messages is to look for obvious and repetitive patterns. Considering that the hidden text is embedded by MP3Stego using SHA-1 to generate pseudo random bits, it is unlikely that obvious and repetitive patterns will be apparent to the steganalyst. Most of the existing steganalysis tools are designed to detect hidden information in image files. In fact, finding hidden data in JPEG images is generally acknowledged to be far easier than other file formats. In cases where MP3Stego is used to hide fairly small amounts of information (such as an account number, password, or other short message) it would be all but impossible to detect the presence of stegotext. My review of existing literature failed to reveal any tools designed to detect hidden data in MP3 files.

NOTE: This does not mean that these tools don’t exist—only that they are not readily available to the general public.

**Conclusions**

Steganography techniques involving audio file formats appear to be increasing in popularity. This may be attributed to the fact that current steganalysis tools available to the general public fall short when applied to audio files. Another factor that may be contributing to the increasing popularity is the widespread popularity of the MP3 file format and its ubiquitous presence on storage media and the Internet. Only adding to the appeal of audio steganography is the relatively large size of MP3 files. These large files make detection of stegotext more difficult. Until such time as effective steganalysis tools are widely available, the information security professional must pay particular attention to the kinds of audio file traffic seen on networks and also be aware of the software signatures of steganography tools. Tools that keep track of installed software such as Asset Insight can aid in detecting “budding” steganographers. In cases where users are careless about deleting “work files”, the presence of PCM files and mono .wav files (an MP3Stego signature) should arouse interest.

\(^{14}\) deCarmo, p.3  
\(^{15}\) Johnson, p.1
In a WiredNews article Ross Anderson notes that “There are about three or four generations of stego software. The stuff you can download is first generation and easily defeated.”\(^{16}\) His recommendation for stego software is MP3Stego. Despite the fact that he has mentored and worked with MP3Stego’s creator, I’m inclined to trust his advice, if only for the fact that we all need become familiar with it (and other steganographic tools). In the digital battlefield of the 21\(^{st}\) Century your attention to steganography is imperative. Denying adversaries like bin Laden the opportunity to use cheap and reliable covert communication channels will be crucial to protecting what we all hold dear.

**References**


\(^{16}\) McCullagh p. 3-4


Weeks, K., “Hiding in plain sight”, URL. http://community.borland.com/article/0,1410,20586,00.html
<table>
<thead>
<tr>
<th>Event Name</th>
<th>City, Country</th>
<th>Start Date - End Date</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SANS Houston 2018</td>
<td>Houston, TXUS</td>
<td>Oct 29, 2018 - Nov 03, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Gulf Region 2018</td>
<td>Dubai, AE</td>
<td>Nov 03, 2018 - Nov 15, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Sydney 2018</td>
<td>Sydney, AU</td>
<td>Nov 05, 2018 - Nov 17, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS DFIRCON Miami 2018</td>
<td>Miami, FLUS</td>
<td>Nov 05, 2018 - Nov 10, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS London November 2018</td>
<td>London, GB</td>
<td>Nov 05, 2018 - Nov 10, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Dallas Fall 2018</td>
<td>Dallas, TXUS</td>
<td>Nov 05, 2018 - Nov 10, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>Pen Test HackFest Summit &amp; Training 2018</td>
<td>Bethesda, MDUS</td>
<td>Nov 12, 2018 - Nov 19, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Mumbai 2018</td>
<td>Mumbai, IN</td>
<td>Nov 12, 2018 - Nov 17, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Rome 2018</td>
<td>Rome, IT</td>
<td>Nov 12, 2018 - Nov 17, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Osaka 2018</td>
<td>Osaka, JP</td>
<td>Nov 12, 2018 - Nov 17, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS San Diego Fall 2018</td>
<td>San Diego, CAUS</td>
<td>Nov 12, 2018 - Nov 17, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS November Singapore 2018</td>
<td>Singapore, SG</td>
<td>Nov 19, 2018 - Nov 24, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Stockholm 2018</td>
<td>Stockholm, SE</td>
<td>Nov 26, 2018 - Dec 01, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Austin 2018</td>
<td>Austin, TXUS</td>
<td>Nov 26, 2018 - Dec 01, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS San Francisco Fall 2018</td>
<td>San Francisco, CAUS</td>
<td>Nov 26, 2018 - Dec 01, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>European Security Awareness Summit 2018</td>
<td>London, GB</td>
<td>Nov 26, 2018 - Nov 29, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Khobar 2018</td>
<td>Khobar, SA</td>
<td>Dec 01, 2018 - Dec 06, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Dublin 2018</td>
<td>Dublin, IE</td>
<td>Dec 03, 2018 - Dec 08, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Santa Monica 2018</td>
<td>Santa Monica, CAUS</td>
<td>Dec 03, 2018 - Dec 08, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Nashville 2018</td>
<td>Nashville, TNUS</td>
<td>Dec 03, 2018 - Dec 08, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>Tactical Detection &amp; Data Analytics Summit &amp; Training 2018</td>
<td>Scottsdale, AZUS</td>
<td>Dec 04, 2018 - Dec 11, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Frankfurt 2018</td>
<td>Frankfurt, DE</td>
<td>Dec 10, 2018 - Dec 15, 2018</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Bangalore January 2019</td>
<td>Bangalore, IN</td>
<td>Jan 07, 2019 - Jan 19, 2019</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Sonoma 2019</td>
<td>Santa Rosa, CAUS</td>
<td>Jan 14, 2019 - Jan 19, 2019</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Amsterdam January 2019</td>
<td>Amsterdam, NL</td>
<td>Jan 14, 2019 - Jan 19, 2019</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS OnDemand</td>
<td>Books &amp; MP3s OnlyUS</td>
<td>Anytime</td>
<td>Self Paced</td>
</tr>
</tbody>
</table>