Secret Sharing in Audio Steganography

Ka Fai Peter Chan
Defence, Peace, Safety & Security (DPSS)
Council for Scientific and Industrial Research
Pretoria, South Africa
kchan@csir.co.za

Abstract— **This paper demonstrates the feasibility of combining the use of secret sharing in conjunction with audio steganography to provide a robust way to store and transmit secret data. Storage of secret information is a constant security concern in the corporate and military environment. Cryptography and steganography are two known techniques that deal with this concern.**

Recently, the combination of the two techniques has been used in conjunction to improve data security. However, a weakness of this technique is that storage is centralized. The hidden information is stored in a single information carrier of location. If the information carrier or location becomes inaccessible, the hidden information becomes irretrievable.

This article proposes the use of a threshold scheme in conjunction with steganography to provide a robust method of hiding information in MP3 audio files. This approach addresses the weakness existing in storage and transmission of secret data as abovementioned. The article concludes with the results of such an application and recommendation for future work.

Keywords-Steganography; cryptography; information carrier; threshold sharing scheme; mp3 audio files

I. Introduction

Secret information is handled in the military and corporate environments on a daily basis. Such information includes company strategies or national security documents that should not be publicly disclosed. Storage and transmission of secret information is a constant concern. Two techniques that increase the secrecy of information are cryptography and steganography.

Cryptography involves the process of encrypting data to conceal its contents. In order to view the contents, the user is required to decrypt the data with a key. There are many cryptographic algorithms to secure data [1], but sometimes just hiding the information is not enough. Steganography is the art of hiding information within information. Unlike cryptography, steganography focuses on hiding the presence of data rather than the data contents. When the two techniques are combined, it offers a high level of security [2].

However, a weakness of the combined technique is that the secret data is centralized. All the secret data reside in a single location or within a single information carrier. If the information carrier becomes corrupted or modified, all the secret data becomes irretrievable [3]. Having the secret data residing in one location is prone to the threat of intrusion. If an attacker manages to get hold of the information carrier, revealing its contents becomes easy. This article proposes the use of a secret sharing scheme to address the mentioned weakness. The secret shared data is then hidden in audio files to increase the level of security.

II. Background of Secret Sharing

The idea of secret sharing was first proposed by Shamir [4] and Barkley [5]. The concept of secret sharing is based on a (t, n) threshold scheme, whereby secret data is split into n parts where at least t parts are required to reconstruct the data. Following Shamir's threshold scheme, the application proposed in this article needs to satisfy the following properties:

1. The secret data can generate n parts, where $n > 0$ and is any real number.
2. Any t number of pieces, where $t > 1$, can reconstruct the original data.
3. Any $t-1$ parts cannot reconstruct the original data.

An additional requirement for the purpose of this article is to be able to disperse any data format files into n pieces.

Although the idea of secretly sharing data in steganography is not new [3, 6, 7, 8], there has been relatively little research done on audio steganography. Transmission of images and other smaller information mediums are faster due to their size, but with today's internet bandwidth, transmission of audio files have become rather popular. MP3 is a popular audio file format. This article proposes using secret sharing in conjunction with MP3 steganography as a technique to store and transmit secret data.

The rest of this article is organized as follows. A brief overview of Shamir's threshold scheme will be discussed in Section III. Section IV looks at audio steganography how it applies in MP3 files. Section IV introduces the application of Shamir's threshold scheme and experiment set-up. Experimental results will be discussed in Section VI. Section VII provides some recommendation for future work and Section VIII concludes the article.

The author would like to thank the CSIR for the opportunity to pursue this research. A special thanks goes to Marthie Grobler for her comments on the first draft of this paper and to Bradley Streak for his assistance in developing the threshold sharing application. The finance support from the University of Pretoria is hereby gratefully acknowledged.

III. Shamir Secret Sharing Scheme

The secret sharing scheme used in this article is based on Shamir's sharing scheme. The idea behind such a scheme is to take a piece of data *D* and split it into *n* number of parts $(n > 1, n \varepsilon R)$ and a threshold *t* $(0 < t \leq n)$ is required to reconstruct the data.

A. Sharing a Secret

In order to split the data *D* into parts, *D* must first be a polynomial. To split *D* into *n* parts, $(D_{1,\ \ldots,}D_n)$, a random coefficient *a* and a *t-1* degree polynomial is generated such that the following function is obtained:

$$F(x) = a_0 + a_1x + a2x^{2\ +}\ a3x^3 + \ldots. + a(t-1)x^{t-1} \tag{1}$$

In which a0 = D. Then evaluate:

$$D_1 = f(1), D_2 = f(2),\ \ldots, D_n = f(n). \tag{2}$$

Data parts can then be dispersed to different locations or parties. Application of such technique addresses the problem of securing data in a single location or host whereby a single misfortune can corrupt the data. The proof of the scheme can be found in Shamir's original article [4].

B. Revealing a Secret

The reconstruction process involves the Lagrange polynomial interpolation. The polynomial interpolation allows users to obtain a function with a common coefficient – the common coefficient being the secret data, from a set of distinct *x* and *y* values. The following formula shows the interpolation function.

$$\mathrm{F}(\mathrm{x}) = \sum_{j=0}^{k} \mathrm{y}_j\, \mathrm{l}_j(x) \tag{3}$$

$$lj(x) := \prod_{\substack{0<m<k\\ m\neq k}} \frac{x - x_m}{xj - x_m} = \frac{(x-x)}{(x_j - x_0)} \cdots \frac{(x - x_{j-1})}{(x_j - x_{j-1})} \frac{(x - x_{j+1})}{(x_j - x_{j+1})} \cdots \frac{(x - x_k)}{(x_j - x_k)} \tag{4}$$

Where j is the recovery threshold, x is the piece number and y is the value of the function 3.

Such a technique is ideal for organizations where hierarchy is important. The threshold required to reconstruct the data can increase inversely proportional to the rank of the piece holder. It also enforces the decision to reveal the secret data once more than one interested party agrees upon it.

IV. Audio Steganography and Mp3

Audio steganography is about hiding data in audio files by looking at redundant bytes and replacing them. The type of audio file that this article focuses on is the MP3 audio file format. This section briefly discusses the structure of an MP3 audio file and illustrates the places where data can be hidden.

A. MP3 Background

MP3 is the abbreviation for MPEG-1 or MPEG-2 audio Layer 3. The Moving Picture Experts Group (MPEG) first designed the concept MPEG1 & 2 in 1991. The ISO/IEC Committee published the standard for MPEG2 audio layer 3 in 1993 as ISO/IEC 11172-2:1993 [9].

MP3 is currently a popular data format to store audio data. They preserve a decent sound quality even after they are highly compressed. The encoding process of MP3 files from original sound files (such as .wav) is a complex process involving human auditory system properties [10]. The encoding process is beyond the scope of this article.

The result of the encoding process is a set of data frames. The set of data frames is assembled to form an MP3 audio file. Figure 1 illustrates a MP3 data frame.

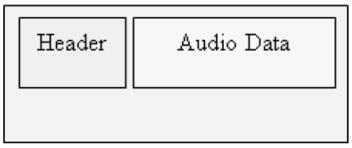


Figure 1. A MP3 Data Frame

The number of frames that make up a MP3 file varies. The number of frames produced depends on the number of sounds produced per interval. For example, a solo vocal section would contain fewer frames than an interval where there are many instruments playing. Figure 2 illustrates a MP3 file structure. ID3 version 2 tags contain the MP3 files metadata, such as the album it belongs to, the artist and track number.

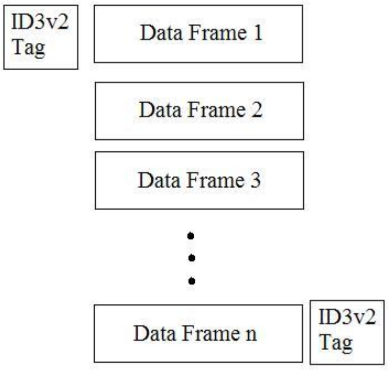


Figure 2. MP3 file structure

B. Hiding Data in MP3

There are primarily two techniques to hide data in MP3 files. The one technique is to hide during the encoding of the MP3 file and the other is post encoded (after the encoding process). The method for hiding data during the encoding of MP3 is complex and requires the original .wav audio file. Having to obtain the original audio file makes the hiding process tedious. Post encode steganography methods only require a MP3 file and will return a MP3 file after the hiding process.

This article proposes the use of byte stuffing to hide shared data pieces in MP3 files. The use of this method was first implemented by Maciak, Ponniah and Sharma [11] in 2001. They used such a technique to embed text into MP3 files. Since then, there has been very little research performed on this technique.

The way in which byte stuffing works is rather straightforward. MP3 files often have data frames that are padded. These padding contain empty bytes used to even out or balance the frame rate. Since the padding bytes are empty, they can easily be replaced with hidden data. Figure 3 illustrates the padding in a MP3 data frame.

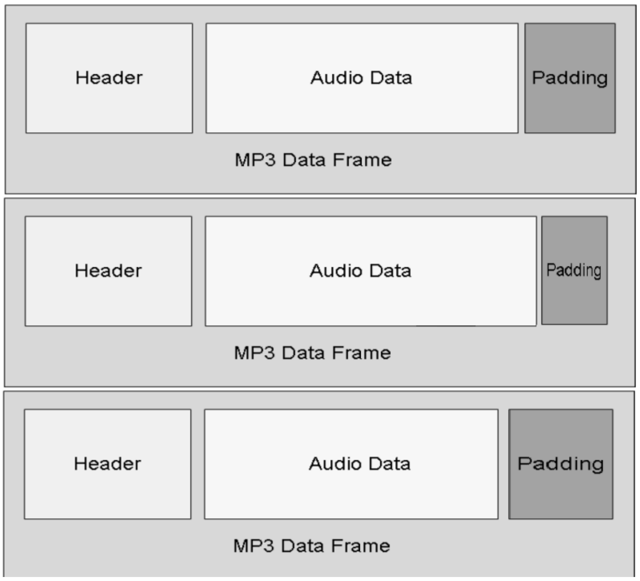


Figure 3. Padding in MP3 Data Frames

Due to the sheer amount of frames that make up a MP3 file, the potential is there to embed large files in them. However, the varying data frame size does not provide a consistent number of bytes that are available to be used.

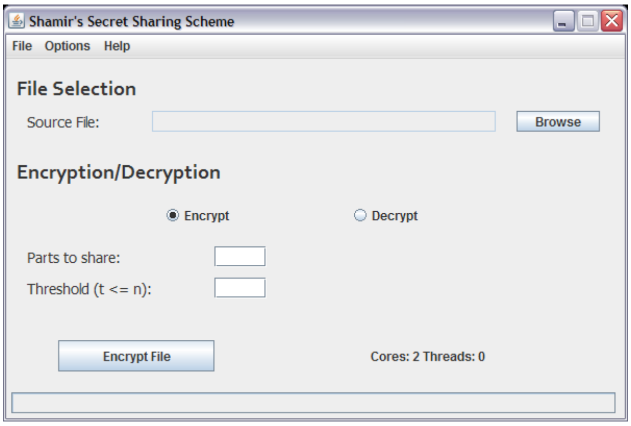
The following section describes the proposed application and how the embedding algorithm operates.

V. Proposed Application

The application of the secret sharing section was implemented according to Shamir's threshold scheme as discussed in Section III. Using Java's type conversation capabilities, different files are read into a byte array and converted to integer type. The integer array forms the a_0 of Equation 1 and is evaluated. Figure 4 shows a screen shot of the application.

The steganography part of the application uses a linear operation that reads in the MP3 audio file and attempts to insert bytes into unused padding space. The main operation of embedding data contains the following five steps:

1. For each frame in MP3 file,
2. Skip the header part of the frame,
3. Read bytes of the frame,
4. If there is free byte, insert byte of hidden data,
5. Flag used bytes.



File Selection

Source File:

Encryption/Decryption

● Encrypt ○ Decrypt

Parts to share:

Threshold (t <= n):

Cores: 2 Threads: 0

Figure 4. Shamir Secret Sharing Scheme Implementation

Each data frame in a MP3 file can be identified with "FFF" in hex values and the first 32 bits of data makes up the header of the frame.

Table 1 and 2 shows the files used to perform the experiments.

TABLE I. Types of Secret Data

Data File	Size in kilobytes
Readme.txt	2kb
Pic.jpg	20kb
CV.pdf	201kb

Secret data varying file size and type is used to test for the application's ability to handle different file types and size.

TABLE II. Information Carrier Mp3 Files

Name of MP3 file	Size in kilobytes	Data Frames
Track01.mp3	9406	18434
Track02.mp3	12708	20756
Track03.mp3	5998	14694
Track04.mp3	3769	9232
Track05.mp3	6968	17071
Track06.mp3	6117	14985

The following section shows the experimental results and provides a brief analysis of the application.

VI. Experimental Results

The first experiment was to test the feasibility of Shamir's threshold scheme implementation. To test this, each of the files in Table I was shared with a different threshold value while the number of shares remain constant. This test provided the proof needed to satisfy the requirements as mentioned in Section I.

The result of the first part of the experiment is shown in Table III.

TABLE III. RESULT OF SHAMIR SECRET SHARING SCHEME IMPLEMENTATION

Name of MP3 file	Size in kilobytes	Size after 20kb data	Sound Distortion	MD5 match to original
Track01.mp3	9406	9406	No	Yes
Track02.mp3	12708	12708	No	Yes
Track03.mp3	5998	5998	No	Yes
Track04.mp3	3769	3769	No	Yes
Track05.mp3	6968	6968	No	Yes
Track06.mp3	6117	6117	Slight	Yes

The results in Table III show that the proposed implementation satisfies the requirements as specified in Section I.

The second part of the experiment is to take each of the pieces and embed them into different MP3 files using the algorithm described in Section V. Table IV and Table V show the result of the hiding process.

TABLE IV. RESULT OF HIDING README.TXT SHARES

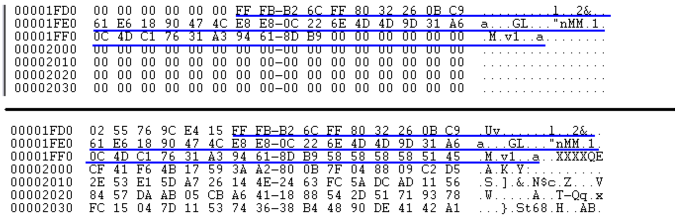
Name of MP3 file	Size in kilobytes	Size after 2kb data	Sound Distortion	MD5 match to original
Track01.mp3	9406	9406	No	Yes
Track02.mp3	12708	12708	No	Yes
Track03.mp3	5998	5998	No	Yes
Track04.mp3	3769	3769	No	Yes
Track05.mp3	6968	6968	No	Yes
Track06.mp3	6117	6117	No	Yes

TABLE V. RESULT OF HIDING PIC.JPG SHARES

Data File	Size in kilobytes	Pieces	Recovery Threshold	Size of Parts	Recover t-1	MD5 check
Readme.txt	2kb	6	2	2kb	No	Yes
		6	3	2kb	No	Yes
		6	4	2kb	No	Yes
Pic.jpg	20kb	6	2	20kb	No	Yes
		6	3	20kb	No	Yes
		6	4	20kb	No	Yes
CV.pdf	201kb	6	2	201kb	No	Yes
		6	3	201kb	No	Yes
		6	4	201kb	No	Yes

Results show that the steganography process did not alter the integrity of the host MP3 file. MD5 hash checks do not show any indication of alteration. However, the size available for embedding secret data is still a problem. CV.pdf could not be embedded in the list of host MP3 files. Although further experiment with another MP3 file allowed the 201Kb file to be embedded in it, the inconsistency of the space available to embed data could hinder the selection of MP3 hosts.

Figure 5 shows where the secret data is embedded into a frame of Track02.mp3. The original MP3 in on the right hand side and the embedded MP3 file is shown on the left. It can be seen that the empty bytes are filled with the secret shared data.



```
00001FD0  00 00 00 00 00 00 FF FB-B2 6C FF 80 32 26 0B C9  ..........1..2&..
00001FE0  61 E6 18 90 47 4C E8 E8-0C 22 6E 4D 4D 9D 31 A6  a...GL..."nMM.1.
00001FF0  0C 4D C1 76 31 A3 94 61-8D B9 00 00 00 00 00 00  .M.v1..a........
00002000  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
00002010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
00002020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
00002030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
```

```
00001FD0  02 55 76 9C E4 15 FF FB-B2 6C FF 80 32 26 0B C9  .Uv......1..2&..
00001FE0  61 E6 18 90 47 4C E8 E8-0C 22 6E 4D 4D 9D 31 A6  a...GL..."nMM.1.
00001FF0  0C 4D C1 76 31 A3 94 61-8D B9 58 58 58 58 51 45  .M.v1..a..XXXXQE
00002000  CF 41 F6 4B 17 59 3A A2-80 0B 7F 04 88 09 C2 D5  .A.K.Y:.........
00002010  2E 53 E1 5D A7 26 14 4E-24 63 FC 5A DC AD 11 56  .S.].&.N$c.Z...V
00002020  84 57 DA AB 05 CB A6 41-18 88 54 2D 51 71 93 78  .W.....A..T-Qq.x
00002030  FC 15 04 7D 11 53 74 36-38 B4 48 90 DE 41 42 A1  ...}.St68.H..AB.
```

Figure 5. Comparison of MP3 Files

All the pieces that were stored on the different MP3 carriers could be reconstructed and the MD5 check corresponding to the original data. By using empty bytes that already exist in the MP3 file, embedding data into the file does not alter the integrity of the file. Employing a threshold sharing scheme, the corruption of one or more MP3 files still allows for the retrieval of the secret data.

However, the threshold number should not be equal to the amount of shared pieces; otherwise a corruption of one piece of the data will make it impossible to reconstruct the secret data. Having the threshold set too low, the secret data could be easily reconstructed without the proper consent. A way to determine the threshold number is to share the secret data based on the number of parties involved and their ranks.

VII. FUTURE WORK

This article proposed a very straightforward implementation of secret sharing in audio steganography. Two main limitations of the current application are the performance of the hiding process and the lack of storage space.

Future work may look into hiding larger files without changing the structure of the host MP3files. Enhancement of post encoded MP3 steganography techniques against steganalysis and better detection of secret data in MP3 files provide interesting research directions.

VIII. CONCLUSION

Storage of secret information is a constant security concern, and the reliability and integrity of this information is important. Previous works involving steganography and cryptography had one common weakness: all the secret data is stored in a single location. The corruption of the secret data's host means the corruption of the secret data.

This article proposed a simple application of threshold sharing and information hiding in mp3 audio files. The proposed application allows users to split any data file into *n* number of parts where a minimum threshold is required to reconstruct it. Each shared part can be embedded into a MP3 audio file carrier. The MP3 can be dispersed to multiple different parties. Knowledge of less than the threshold (*t*-1) shares does not review the secret data.

In terms of ease of use, the proposed implementation uses a post encoded MP3 steganography technique. This means that the user wishing to hide data in a MP3 file does not need to acquire the original wav file. The hiding process embeds in already encoded MP3 files.

Experimental results show the feasibility of such an approach. This is a robust way to secure secret data through obscurity and distribution. This technique can be beneficial to organizations where rank is important. The threshold of the shares can be increased and distributed to personnel of lower ranks. Higher ranked officers can decide to reveal the data with fewer shares where as lower ranked officers require more shares to reveal the same information.

References

[1] G. Kessler. "An Overview of Cryptography". Internet: http://www.garykessler.net/library/crypto.html. May 1998 [March 21, 2011].

[2] P. Marwaha and P. Marwaha, "Visual Cryptographic Steganography in Images", in *Proc. ICCCNT*, 2010, pp. 1-6.

[3] C.-C. Thien and J.-C. Lin, "Secret image sharing," *Computers and Graphics*, vol. 26, no. 5, pp. 765–770, 2002.

[4] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[5] G. R. Blakley, "Safeguarding cryptographic keys," in *Pro. of the National Computer Conference*, pp. 313–317, New York, NY, USA, June 1979.

[6] C.-C. Lin and W.-H. Tsai, "Secret image sharing with steganography and authentication," *Journal of Systems and Software*, vol. 73, no. 3, pp. 405–414, 2004.

[7] C.-N. Yang, T.-S. Chen, K. H. Yu, and C.-C. Wang, "Improvements of image sharing with steganography and authentication," *Journal of Systems and Software*, vol. 80, no. 7, pp. 1070–1076, 2007.

[8] C.-C. Chang, Y.-P. Hsieh, and C.-H. Lin, "Sharing secrets in stego images with authentication," *PatternRecognition*, vol. 41, no. 10, pp. 3130–3137, 2008.

[9] ISO. "ISO/IEC 11172-3:1993 - Information technology -- Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s -- Part 3: Audio".http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=22412 [March 18, 2011].

[10] S. Hacker. *MP3: The Definitive Guide*. O'Reilly Media. Sebastopol .2000.

[11] L.G. Maciak, M.A. Ponniah and R. Sharma, "MP3 Steganography Applying Steganography to Music Captioning", Internet: http://www.terminally-incoherent.com/stuff/projects/mp3stego/paper.htm, Oct. 3 2005 [March 18, 2011].