Contents lists available at ScienceDirect

Forensic Science International: Reports

journal homepage: www.elsevier.com/locate/fsir

Digital Forensics

Detecting fingerprints of audio steganography software[☆]

Chen Gong[a,b,*], Jinghong Zhang[a,b], Yunzhao Yang[a,b], Xiaowei Yi[a,b], Xianfeng Zhao[a,b], Yi Ma[c]

[a] *State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China*
[b] *School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100093, China*
[c] *Beijing Information Technology Institute, Beijing 100094, China*

ARTICLE INFO

Keywords:
Audio steganography
Steganalysis
Fingerprint

ABSTRACT

Steganography has experienced rapidly growth with spreading access to the Internet in recent decades. Such low cost, simple process and easy promotion steganographic softwares pose serious and growing threats and challenges to network security. The fingerprint of steganography has proven itself an effective solution. However, large gaps exist between experiments results and practical applications. Few attention has been paid on the fingerprint of steganography. In this paper, we provide highly effective solutions to identify the fingerprints of audio steganography. Based on the analysis of audio stego produced by three kinds of audio steganographic softwares, including Xiao Steganography, Invisible Secrets, Deep Sound, we successfully detect audio stego across WAV files. Besides, we summarize a general approach of fingerprint extraction, and its effectiveness has been confirmed through experiments. The results solve an urgent need for further research on reverse engineering steganography softwares and detecting audio stego.

1. Introduction

Steganography is the technique and art of sending a message into an innocent digital media while hiding the secret communication itself [1]. As the contrary of Steganography, the goal of steganalysis is to analyze whether the secret message exists in the media on public channels [2]. In the past decades, steganography and steganalysis have keep a good circulation for mutual utilization and mutual promotion. Moreover, audio steganography has gained extensive attention during recent years for widely spread in social media [3].

There are two different types approaches to identify stego. The first is the targeted steganalysis which aims to detect a particular steganographic algorithms in advance [4–8], In the second, the universal steganalysis may be potentially capable of detecting many forms of steganographic schemes without knowing the steganographic algorithms [9–13]. Unlike the previous steganalytic approaches, the fingerprint of steganography is an efficient, quick, and safe method. It is based on the patterns which are in stego files, such as a specific embedded fingerprint or locations of added information, thus the steganalytic feature will be left in stego when using steganographic tools. Swaminathan et al. [14] observed that the processing operations for digital image leave distinct intrinsic characteristic, so these intrinsic fingerprints can be used to verify the integrity of digital image. Liu et al. [15] design a method by counting the number ‘0’ and ‘1’ in the fixed positions after re-embedding using the same steganography software. The proposed method in [16] found that some unusual palettes exist in the stego produced by the GIF steganography tool. While fingerprint is quite effective and fast, both miss detection and false alarm are quite low, or even extracting the whole embedded message.

Xiao Steganography, Invisible Secrets 4 and Deep Sound among those popular steganographic softwares, represent typical audio steganographic softwares which can securely embed message into audio files. In this paper, we make great efforts to the fingerprint technique. Such steganographic tools were based on simple least significant bits (LSBs) [17,18] changes placed in a fixed position of the audio file that provide little security, because the unique trace existed in the modified carrier file. It usually be used for copyright protection or information extraction. These are opportunities to make stego detection easier. Based on the analyzing the stego files, we summarized up an effective approach to obtaining the fingerprint of steganography. The principle way is to compare cover with stego all through, then search for the abnormal data or location between them. Next, we extract the common values from abnormal data in stego as fingerprints. To evaluate the efficiency of the proposed method, experiments have been done to verify the extracted fingerprints for each steganography.

[☆] This work was supported by NSFC under 61902391, 61972390 and U1736214, and National Key Technology R&D Program under 2019QY0700 and 2016QY15Z2500.

[*] Corresponding author at: State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China.
E-mail addresses: gongchen@iie.ac.cn (C. Gong), zhangjinghong@iie.ac.cn (J. Zhang), yangyunzhao@iie.ac.cn (Y. Yang), yixiaowei@iie.ac.cn (X. Yi), zhaoxianfeng@iie.ac.cn (X. Zhao), mayi_5501@126.com (Y. Ma).

http://doi.org/10.1016/j.fsir.2020.100075
Received 5 December 2019; Received in revised form 6 February 2020; Accepted 11 February 2020
Available online 2 March 2020

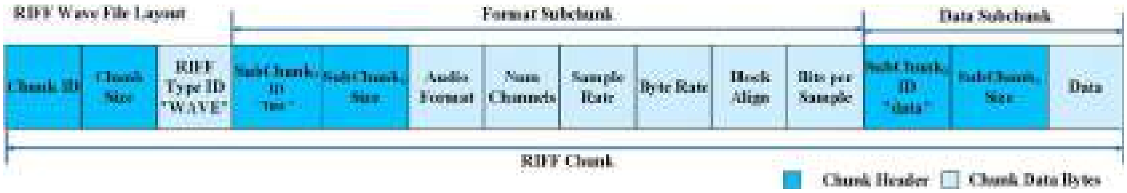


Fig. 1. A typical structure of the RIFF chunk consists of format and data sub-chunks.

The increasing number of steganographic softwares pose a higher requirement on steganalysis, there is a strong need for an identifiable features to increase the reliability of stego recognition. The operation of steganographic embedding can be regarded as clues for steaganalyzers, the fingerprints could use the operation traces and vulnerabilities of steganography software accordingly. So, the fingerprint is the identifiable features left in stego files by steganographic softwares. Highly accurate detection of steganographic software has been proven experimentally on a number of audio files.

The structure of this report is as follows. Section 2 introduces some related background knowledge. Section 3 presents analysis of three cases and gives a general approach of fingerprint extraction for each case. The performance of the approaches will be measured and evaluated in Section 4. We conclude the report in Section 5, where we present some final thoughts, insights and discuss future avenues of work.

2. Preliminary

In this section, we firstly include a brief knowledge of the structure of WAV, highlighting the basic background related to better understand the fingerprints analysed.

2.1. WAV data structure

WAV, Waveform Audio File Format, is a native audio file format standard for storing digital audio bitstream. WAV is developed by Microsoft and IBM. Since the popularity of Windows OS and so many programs designed for the platform, WAV is one of the most widely supported digital audio file formats. The following specification introduces the structure of WAV.

WAV is an instance of the Resource Interchange File Format (RIFF) structure which organize the files contents in "chunks". The chunk is a description about the file, including certain metadata or actual data. Each chunk serves a specific purpose and is structured very specifically (order matters). Different types of chunks may contain sub-chunks. A typical structure of RIFF chunk is illustrated in Fig. 1.

A standard RIFF file format based WAV file itself consists of mainly three sub-chunks, the RIFF chunk is used to determine whether the file is a RIFF-based file, the FORMAT chunk is used to specify parameters such as byte rate and the DATA chunk is used to contains the actual data. An actual WAV sample is illustrated in Figs. 2 and 3. The sample WAV is an 8-bit, mono, 8 KHz. Fig. 2 describes the WAV sample in the time domain, and Fig. 3 describes the basic RIFF WAVE structure of the WAV sample to see how data is stored.

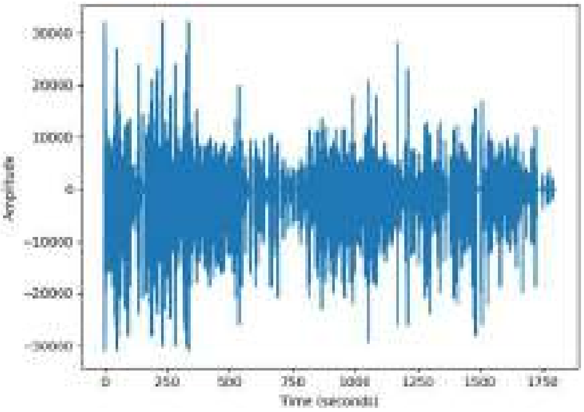


Fig. 2. The waveform in the time domain.

2.2. Embedding communication protocol

In order to extract message correct from the steganographic carriers, each steganographic tool usually defines its own embedding protocol. Thanks to the embedding protocol, the user can determine what steganographic softwares have processed the stego carrier, realize the function of the message extracting correctly. A typically structure of the embedding protocol is shown in Fig. 4. The header start tag and header end tag are used to mark the beginning and end of embedding process. The information of software, including software identifier and software version, this part is mainly used for copyright protection and message extraction. Others are about the secret message, content start tag and content end tag, file name, file length, encryption information and message. Besides, different steganographic software typically employs a customized way of embedding communication protocol. Such embedding communication protocol leave trace (fingerprint) for steganalyzer to identify the authenticity of these suspicious files.

2.3. Xiao Steganography description

Xiao Steganography is a lightweight and free multi-platform software which is designed for hiding private files in BMP images or WAV files. The use of this tool is very simple. All you have to do is run this program, load any BMP image or WAV file in the program interface, and then add the files you want to hide. It also provides encryption so you can choose from a variety of encryption algorithms. The software screenshot is shown in Fig. 5.

2.4. Invisible Secrets 4 description

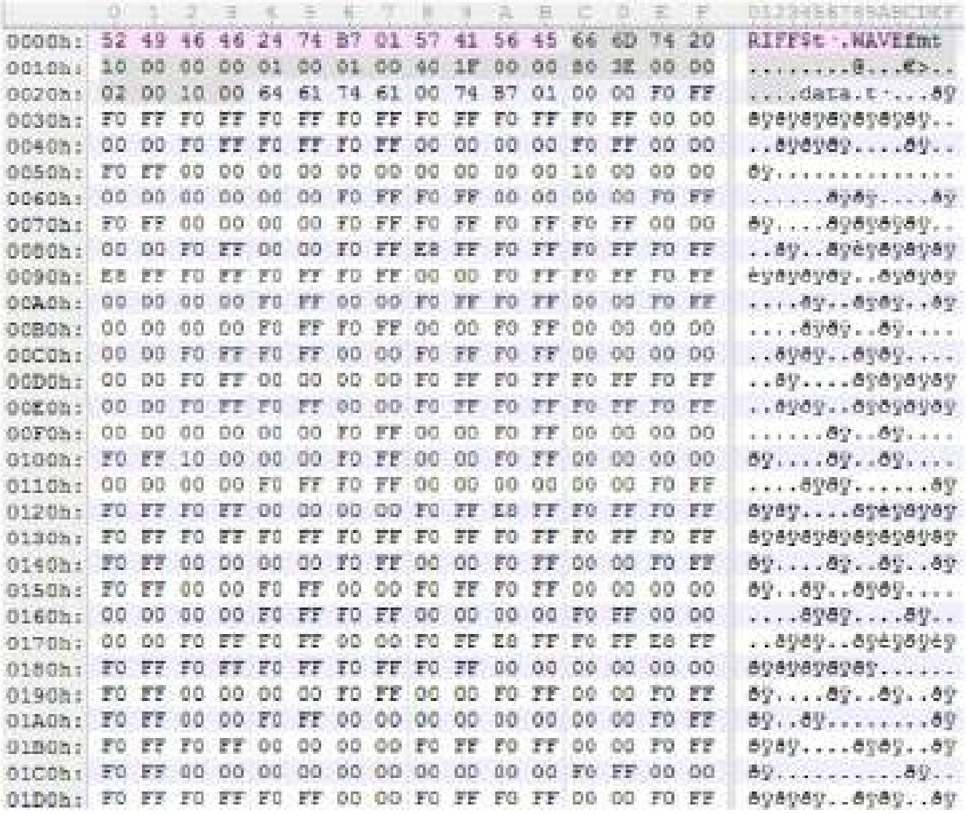
Invisible Secrets 4 is a very powerful file encryption and hiding software. The software not only allows user to encrypt the file and folder structure containing secret message, but also hide files. Invisible Secrets support a variety of file types, including: video, audio and image. The software screenshot is shown in Fig. 6.

2.5. Deep Sound description

Deep Sound is capable of hiding secret message into songs. Moreover, Deep Sound adopt AES256 256-bit encryption. Beside, the application additionally run in a user-friendly way so that audio converter function can encode different audio formats (FLAC, MP3 and APE) into other formats (FLAC, MP3 and APE). The software screenshot is shown in Fig. 7.

3. Fingerprints of steganographic tools

In this section, we present our analysis on the operation of three steganographic tools over WAV carrier. Specially, we will detail the fingerprints to these steganographic tools into perspective. Please notice that all of samples in this section are using the same cover WAV file.



```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F   0123456789ABCDEF
0000h: 52 49 46 46 24 74 B7 01 57 41 56 45 66 6D 74 20   RIFF$t·.WAVEfmt 
0010h: 10 00 00 00 01 00 01 00 40 1F 00 00 80 3E 00 00   ........@...€>..
0020h: 02 00 10 00 64 61 74 61 00 74 B7 01 00 00 F0 FF   ....data.t·...ðÿ
0030h: F0 FF F0 FF F0 FF F0 FF F0 FF F0 FF F0 FF 00 00   ðÿðÿðÿðÿðÿðÿðÿ..
0040h: 00 00 F0 FF F0 FF F0 FF 00 00 00 00 F0 FF 00 00   ..ðÿðÿðÿ....ðÿ..
0050h: F0 FF 00 00 00 00 00 00 00 00 00 00 10 00 00 00   ðÿ..............
0060h: 00 00 00 00 00 00 F0 FF F0 FF 00 00 00 00 F0 FF   ......ðÿðÿ....ðÿ
0070h: F0 FF 00 00 00 00 F0 FF F0 FF F0 FF F0 FF 00 00   ðÿ....ðÿðÿðÿðÿ..
0080h: 00 00 F0 FF 00 00 F0 FF E8 FF F0 FF F0 FF F0 FF   ..ðÿ..ðÿèÿðÿðÿðÿ
0090h: E8 FF F0 FF F0 FF F0 FF 00 00 F0 FF F0 FF F0 FF   èÿðÿðÿðÿ..ðÿðÿðÿ
00A0h: 00 00 00 00 F0 FF 00 00 F0 FF F0 FF 00 00 F0 FF   ....ðÿ..ðÿðÿ..ðÿ
00B0h: 00 00 00 00 F0 FF F0 FF 00 00 F0 FF 00 00 00 00   ....ðÿðÿ..ðÿ....
00C0h: 00 00 F0 FF F0 FF 00 00 F0 FF F0 FF 00 00 00 00   ..ðÿðÿ..ðÿðÿ....
00D0h: 00 00 F0 FF 00 00 00 00 F0 FF F0 FF F0 FF F0 FF   ..ðÿ....ðÿðÿðÿðÿ
00E0h: 00 00 F0 FF F0 FF 00 00 F0 FF F0 FF F0 FF F0 FF   ..ðÿðÿ..ðÿðÿðÿðÿ
00F0h: 00 00 00 00 00 00 F0 FF 00 00 F0 FF 00 00 00 00   ......ðÿ..ðÿ....
0100h: F0 FF 10 00 00 00 F0 FF 00 00 F0 FF 00 00 00 00   ðÿ....ðÿ..ðÿ....
0110h: 00 00 00 00 F0 FF F0 FF 00 00 00 00 00 00 F0 FF   ....ðÿðÿ......ðÿ
0120h: F0 FF F0 FF 00 00 00 00 F0 FF E8 FF F0 FF F0 FF   ðÿðÿ....ðÿèÿðÿðÿ
0130h: F0 FF F0 FF F0 FF F0 FF F0 FF F0 FF F0 FF F0 FF   ðÿðÿðÿðÿðÿðÿðÿðÿ
0140h: F0 FF 00 00 00 00 F0 FF 00 00 F0 FF 00 00 F0 FF   ðÿ....ðÿ..ðÿ..ðÿ
0150h: F0 FF 00 00 F0 FF 00 00 F0 FF F0 FF 00 00 00 00   ðÿ..ðÿ..ðÿðÿ....
0160h: 00 00 00 00 F0 FF F0 FF 00 00 00 00 F0 FF 00 00   ....ðÿðÿ....ðÿ..
0170h: 00 00 F0 FF F0 FF 00 00 F0 FF E8 FF F0 FF E8 FF   ..ðÿðÿ..ðÿèÿðÿèÿ
0180h: F0 FF F0 FF F0 FF F0 FF F0 FF 00 00 00 00 00 00   ðÿðÿðÿðÿðÿ......
0190h: F0 FF 00 00 00 00 F0 FF 00 00 F0 FF 00 00 F0 FF   ðÿ....ðÿ..ðÿ..ðÿ
01A0h: F0 FF 00 00 F0 FF 00 00 00 00 00 00 00 00 F0 FF   ðÿ..ðÿ........ðÿ
01B0h: F0 FF F0 FF 00 00 00 00 F0 FF F0 FF 00 00 F0 FF   ðÿðÿ....ðÿðÿ..ðÿ
01C0h: F0 FF 00 00 00 00 00 00 00 00 00 00 F0 FF 00 00   ðÿ..........ðÿ..
01D0h: F0 FF F0 FF F0 FF 00 00 F0 FF F0 FF 00 00 F0 FF   ðÿðÿðÿ..ðÿðÿ..ðÿ
```

Fig. 3. The WAV file with bytes shown as hexadecimal numbers(A part structure of the “RIFF” chunk consists of “fmt” and “data” sub-chunks).

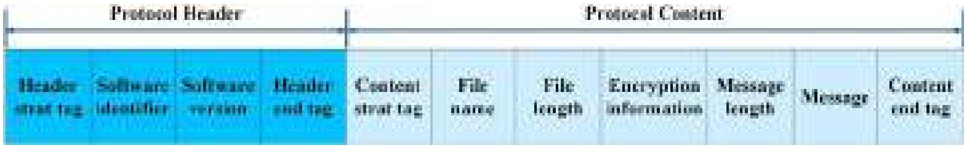


Fig. 4. A typical structure of embedding communication protocol. Different steganographic software typically employs a customized way of embedding communication protocol.

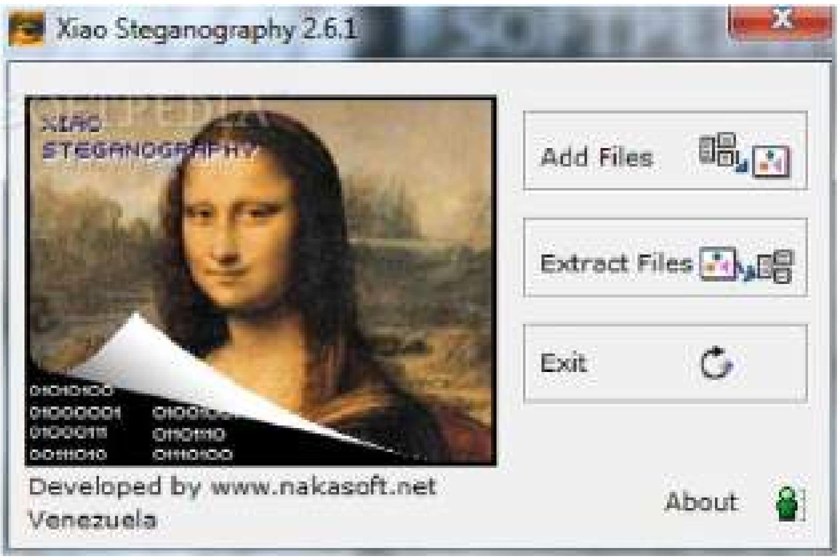


Fig. 5. The user interface of Xiao Steganography.

Fig. 6. The user interface of Invisible Secrets 4.

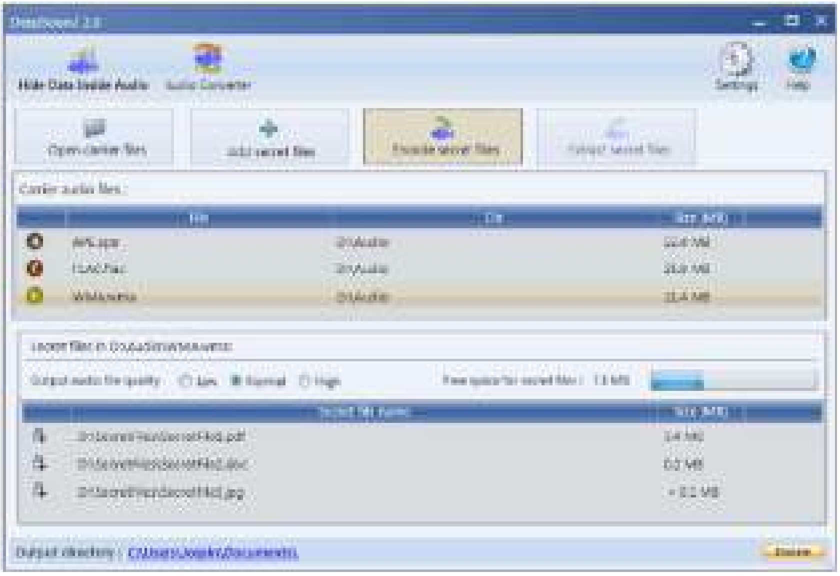


Fig. 7. The user interface of Deep Sound.

3.1. *Fingerprint of Xiao Steganography*

The embedding start position of Xiao Steganography is the starting position of the wav format audio data segment, that is data subchunk. The embedding order is the fixed embedding order. The comparison of cover and stego produced by Xiao Steganography is shown in Fig. 8. The LSB steganography algorithm is used to realized the hiding of the message. In addition, Xiao Steganography will add 1 byte of 0 × 00 into the end of the wav file. The Xiao Steganography communication protocol consists of eleven parts: flag field (0 × 54, 0 × 41, 0 × 72, 0 × CA, 0 × F5, 0 × E0, 0 × A0, 0 × E6, 0 × 78, 0 × 16, 0 × 3C), unknown field (0 × 01, 0 × 00, 0 × 00, 0 × 00), embedded message length (4 bytes), 0 × 00, 0 × 00, 0 × 00, 0 × 00/0 × FF, 0 × FF, 0 × FF, 0 × FF (the former represents normal, the latter represents encryption), encryption methods (4 bytes), HASH methods, file suffix (3 bytes), message length (4 bytes), file name (10 Bytes, extra 10 bytes will be truncated), hidden message, add 0 × 00 field is fixed at the end of the file. Normally, the hidden message is plain text; the hidden data is encrypted content when encrypted, as shown in Fig. 9.

3.2. *Fingerprint of Invisible Secrets 4*

The embedding start position of Invisible Secrets 4 is the starting position of the wav format audio data segment. The embedding order is the fixed embedding order. The LSB steganography scheme is adopted to

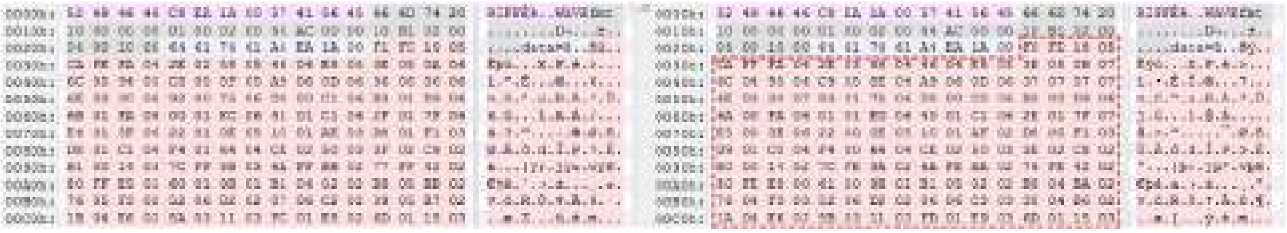
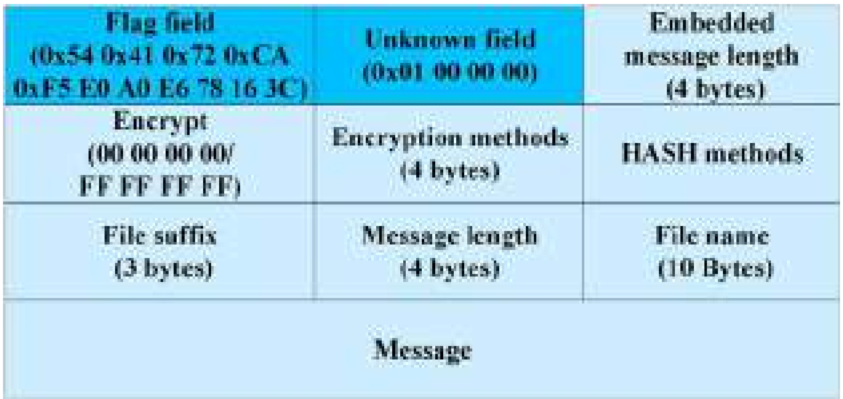


Fig. 8. Compare the cover and stego produced by Xiao Steganography in hexadecimal numbers. Left is the part of cover WAV, right is the part of stego WAV. The dashed boxes in right part indicate the embedding position.



Flag field (0x54 0x41 0x72 0xCA 0xF5 E0 A0 E6 78 16 3C)	Unknown field (0x01 00 00 00)	Embedded message length (4 bytes)
Encrypt (00 00 00 00/ FF FF FF FF)	Encryption methods (4 bytes)	HASH methods
File suffix (3 bytes)	Message length (4 bytes)	File name (10 Bytes)
Message		

Fig. 9. The embedding protocol of Xiao Steganography.

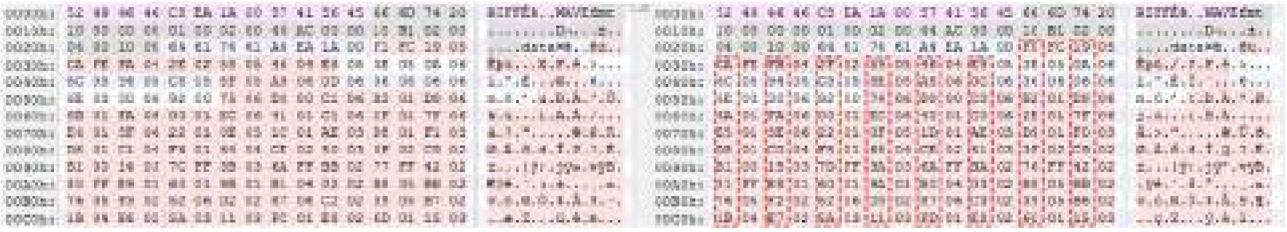
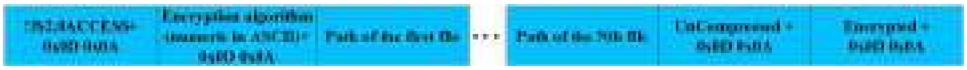


Fig. 10. Compare the cover and stego produced by Invisible Secrets 4 in hexadecimal numbers. Left is the part of cover WAV, right is the part of stego WAV. The dashed boxes in right part indicate the embedding position.



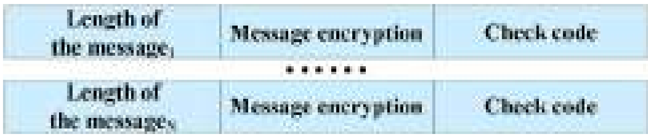
"!IS2.0ACCESS+ 0x0D 0x0A	Encryption algorithm (numeric in ASCII)+ 0x0D 0x0A	Path of the first file	...	Path of the Nth file	UnCompressed + 0x0D 0x0A	Encrypted + 0x0D 0x0A

Fig. 11. The embedding protocol header of Invisible Secrets 4.

complete the hiding of hidden information. In contrast to the Xiao Steganography software, the Invisible Secrets steganography software embeds in odd-numbered locations which only use audio data segments. For even-numbered locations, no message is embedded. The odd position of data changes, while the even position of data is the same as the original wav audio data, it is illustrated in the example of the Invisible Secrets audio data segment shown in Fig. 10. The Invisible Secrets 4 communication protocol header has five parts: the flag field ("! IS2.0ACCESS" + 0 × 0D 0 × 0A), the encryption algorithm selection field, the file path field (which may include multiple files), and the compressed field (Uncompressed + 0 × 0D 0 × 0A) and the encrypted field (Encrypted + 0 × 0D 0 × 0A). As illustrated in Fig. 11. The communication protocol content mainly includes three parts: each file includes a hidden information length, an encrypted content, and a check field (which may include multiple), as shown in Fig. 12. The Invisible Secrets WAV communication protocol consists of four parts: header length, encrypted header data, total length of hidden information, and hidden content. It is shown in Fig. 13.

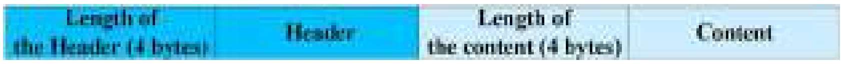
3.3. Fingerprint of Deep Sound

The embedded start position of Deep Sound is the starting position of the audio data segment. The embedding order is the fixed embedding



Length of the message$_1$	Message encryption	Check code
......		
Length of the message$_N$	Message encryption	Check code

Fig. 12. The embedding protocol content of Invisible Secrets 4. The check field existed when using encryption algorithm.



Length of the Header (4 bytes)	Header	Length of the content (4 bytes)	Content

Fig. 13. The embedding protocol of Invisible Secrets 4 designed for WAV type.

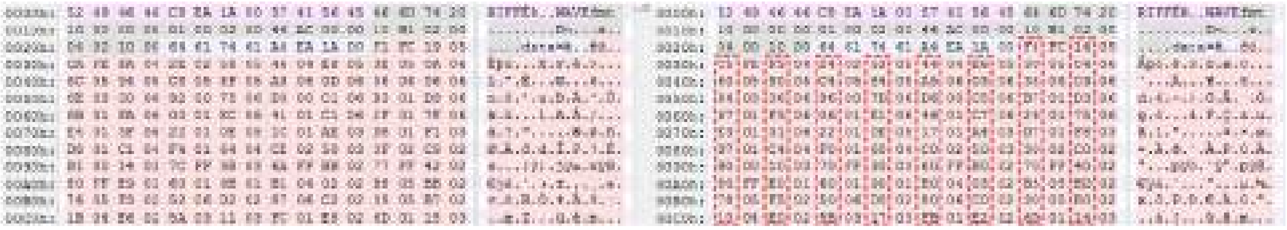
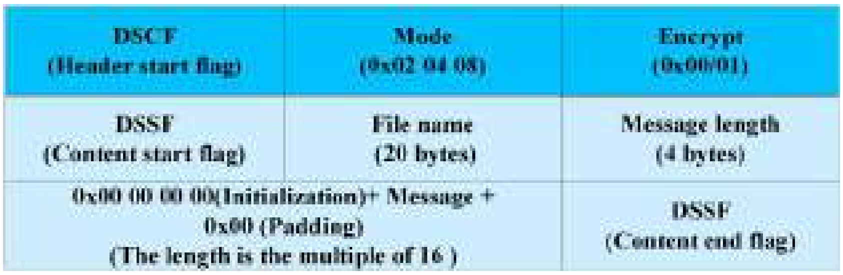


Fig. 14. Compare the cover and stego produced by Deep Sound in hexadecimal numbers. Left part is the part of cover WAV, right part is the part of stego WAV. The dashed boxes in right part indicate the embedding position.



DSCF (Header start flag)	Mode (0x02 04 08)	Encrypt (0x00/01)
DSSF (Content start flag)	File name (20 bytes)	Message length (4 bytes)
0x00 00 00 00(Initialization)+ Message + 0x00 (Padding) (The length is the multiple of 16)		DSSF (Content end flag)

Fig. 15. The embedding protocol of Deep Sound designed for WAV type.

Table 1
The execution time of 10,000 ms samples under different Steganographic tool.

Steganographic tool	Execution time (s)
Xiao Steganography	52
Invisible Secrets 4	3
Deep Sound	1

order. Like the Invisible Secrets 4, the odd position information is changed, and the even position is the same as the original wav audio data, as illustrated in Fig. 14. The Deep Sound communication protocol can be divided into two parts: the protocol header and the protocol content. The protocol header has three parts: DSCF (software start flag), 02/04/08 represent for low, normal and high modes, and 00/01 for normal mode/ encryption mode. The protocol content has six parts: DSSF (content start flag), file name (20 bytes), hidden message length (4 bytes), hidden message initialized by 4 0 × 00 and filled by padding 0 × 00, in order to make the message length is a multiple of 16, and DSSF (content end flag). Unlike the former, the embedded method of the Deep Sound protocol header is the lowest 4-bit. The embedding method for low, normal, and high mode content is the lowest 8, 4, and 2 LSB. In the Deep Sound encryption mode, the message will be encrypted, and it is difficult to restore the plain text. It is shown in Fig. 15.

Here we further compare the execution time of the software, The results are shown in Table 1. Specifically, we tested 100 samples. Each sample is 10,000 ms and 1.5 Mb txt file are used as embedding message. We can see that the differences in the execution time are obvious.

4. Experiments

4.1. Setting

This section provides experiments on stego detection. Firstly, we concluded the well-designed fingerprint of these steganography, then presented the experiment results on stego WAV produced from the these tools. To evaluate the performance, several experiments across 500 paired of cover and stego WAV files are performed in this section.

All experiments in this paper are on a public dataset published by Lin [19]. In this dataset, it contains more than 100 h speech conversation of different languages from different gender. Each sample stored in WAV format with 16 bits. Each WAV file was used to embed message by these 3 under different embedding rates from 20% to 100% in step of 40%. In the experiments, the embedding ratio represents the ratio of length of message m to the length of cover audio n. Besides, we cut samples into 1s, 2s, and 3s segments to test the model performance in different length samples. Every experiment are randomly divided into training set, testing set with the ratio of 5:5. The detection performance are measured by P_{MD}, P_{FA} and P_{TP}. Here, P_{MD} is the probability of the stego samples wrongly identified, P_{FA} is the probability of the cover samples wrongly identified, and P_{TP} is the probability of the stego samples correctly identified.

4.2. Results

From Table 2, we can see the extracting fingerprints from these steganography get a high performance to detect the 1000 ms samples under different embedding rate, including miss detection rate, true positive rate and false alarm rate. Besides, we can also observe from Table 3 that the proposed method can also highly effective in detecting various length of samples. Thus, we can conclude that the proposed method can effectively detect the these 3 steganographic tools.

5. Conclusions

In this paper, we present the detection against the WAV files produced by Xiao Steganography, Invisible Secrets 4 and Deep Sound by the extracted fingerprint. The experiment result indicates the proposed method is highly effective and practical. Specifically, the fingerprint can be used to deal with different embedding rate under different length WAV files. So the fingerprint is more convincing than the machine learning based steganalysis. This work inspires us to investigate more steganographic softwares regardless of file type. We expect our work can provide

Table 2
Detection accuracy of 1000 ms samples under different embedding rate.

Embedding rate	Steganographic tool	Miss detection rate	True positive rate	False alarm rate
0.2	Xiao Steganography	0%	100%	0%
	Invisible Secrets 4	0%	100%	0%
	Deep Sound	0%	100%	0%
0.6	Xiao Steganography	0%	100%	0%
	Invisible Secrets 4	0%	100%	0%
	Deep Sound	0%	100%	0%
1.0	Xiao Steganography	0%	100%	0%
	Invisible Secrets 4	0%	100%	0%
	Deep Sound	0%	100%	0%

Table 3
Detection accuracy of 100% embedding rate under different length samples.

Sample length	Steganographic tool	Miss detection rate	True positive rate	False alarm rate
1000 ms	Xiao Steganography	0%	100%	0%
	Invisible Secrets 4	0%	100%	0%
	Deep Sound	0%	100%	0%
2000 ms	Xiao Steganography	0%	100%	0%
	Invisible Secrets 4	0%	100%	0%
	Deep Sound	0%	100%	0%
3000 ms	Xiao Steganography	0%	100%	0%
	Invisible Secrets 4	0%	100%	0%
	Deep Sound	0%	100%	0%

some inspirations to design and develop effective steganalytic techniques against the steganographic tool. In the future, more work needs to be done on finding a universal approach to extract fingerprint.

The primary contributions of our work is to propose a general framework to determine the fingerprint of the steganographic softwares. The fingerprint show that some unique fixed bits could be used to reliably construct an effective steganalytic feature. Fingerprint steganalysis provide a means to not only detect the presence of steganography but also directly relationship with the steganalytic tool. Moreover we offer a general idea to find the fingerprint by analyzing the fixed position embedding within an WAV file.

Conflict of interest statement

We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work.

References

[1] B. Li, J. He, J. Huang, Y.Q. Shi, A survey on image steganography and steganalysis, J. Inf. Hiding Multim. Signal Process. 2 (2) (2011) 142–172.

[2] Y.Q. Shi, C. Chen, W. Chen, A Markov process based approach to effective attacking JPEG steganography, International Workshop on Information Hiding (2006) 249–264.

[3] W. Luo, Y. Zhang, H. Li, Adaptive audio steganography based on advanced audio coding and syndrome-trellis coding, International Workshop on Digital Watermarking (2017) 177–186.

[4] J. Fridrich, M. Goljan, D. Hogea, Steganalysis of JPEG Images: Breaking the F5 Algorithm, (2002) .

[5] X. Yu, N. Babaguchi, Breaking the YASS algorithm via pixel and DCT coefficients analysis, 19th International Conference on Pattern Recognition, 2008, ICPR 2008 (2009) .

[6] R. Böhme, A. Westfeld, Breaking Cauchy Model-Based JPEG Steganography with First Order Statistics, (2004) .

[7] J. Fridrich, M. Goljan, R. Du, Reliable detection of LSB steganography in color and grayscale images, Proceedings of the 2001 Workshop on Multimedia and Security: New Challenges (2001) 27–30.

[8] B. Li, J. Huang, Y.Q. Shi, Steganalysis of YASS, IEEE Trans. Inf. Forensics Secur. 4 (3) (2009) 369–382.

[9] C. Chen, Y.Q. Shi, JPEG image steganalysis utilizing both intrablock and interblock correlations, 2008 IEEE International Symposium on Circuits and Systems (2008) 3029–3032.

[10] J. Fridrich, J. Kodovsky, Rich models for steganalysis of digital images, IEEE Trans. Inf. Forensics Secur. 7 (3) (2012) 868–882.

[11] V. Holub, J. Fridrich, Low-complexity features for JPEG steganalysis using undecimated DCT, IEEE Trans. Inf. Forensics Secur. 10 (2) (2014) 219–228.

[12] T. Pevny, P. Bas, J. Fridrich, Steganalysis by subtractive pixel adjacency matrix, IEEE Trans. Inf. Forensics Secur. 5 (2) (2010) 215–224.

[13] B. Li, Z. Li, S. Zhou, S. Tan, X. Zhang, New steganalytic features for spatial image steganography based on derivative filters and threshold LBP operator, IEEE Trans. Inf. Forensics Secur. 13 (5) (2017) 1242–1257.

[14] A. Swaminathan, M. Wu, K.R. Liu, Digital image forensics via intrinsic fingerprints, IEEE Trans. Inf. Forensics Secur. 3 (1) (2008) 101–117.

[15] G. Bell, Y.-K. Lee, A method for automatic identification of signatures of steganography software, IEEE Trans. Inf. Forensics Secur. 5 (2) (2010) 354–358.

[16] N.F. Johnson, S. Jajodia, Exploring steganography: seeing the unseen, Computer 31 (2) (1998) 26–34.

[17] J. Mielikainen, LSB matching revisited, IEEE Signal Process. Lett. 13 (5) (2006) 285–287.

[18] A. Westfeld, F5-a steganographic algorithm, International Workshop on Information Hiding (2001) 289–302.

[19] Z. Lin, Y. Huang, J. Wang, Rnn-sm: fast steganalysis of voip streams using recurrent neural network, IEEE Trans. Inf. Forensics Secur. 13 (7) (2018) 1854–1868.