Audio Steganography with Embedded Text

Chua Teck Jian, Chuah Chai Wen, Nurul Hidayah Binti Ab. Rahman and Isredza Rahmi Binti A. Hamid

Information Security Interest Group (ISIG), Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia, Batu Pahat, Johor, Malaysia

Corresponding author: ai130242@siswa.uthm.edu.my, cwchuah@uthm.edu.my, hidayahar rahmi@uthm.edu.my

Abstract. Audio steganography is about hiding the secret message into the audio. It is a technique uses to secure the transmission of secret information or hide their existence. It also may provide confidentiality to secret message if the message is encrypted. To date most of the steganography software such as Mp3Stego and DeepSound use block cipher such as Advanced Encryption Standard or Data Encryption Standard to encrypt the secret message. It is a good practice for security. However, the encrypted message may become too long to embed in audio and cause distortion of cover audio if the secret message is too long. Hence, there is a need to encrypt the message with stream cipher before embedding the message into the audio. This is because stream cipher provides bit by bit encryption meanwhile block cipher provide a fixed length of bits encryption which result a longer output compare to stream cipher. Hence, an audio steganography with embedding text with Rivest Cipher 4 encryption cipher is design, develop and test in this project.

1. Introduction

Cryptography is about scrambled the structure or sequence of message into a structure that unreadable by human, namely ciphertext. Two types of algorithm are symmetric and asymmetric. Symmetric cipher is using single key in encryption and decryption such as Advanced Encryption Standard (AES) and Rivest Cipher 4 (RC4). The symmetric cipher contains two types of cipher which are block cipher and stream cipher. Asymmetric cipher is using public key in encryption and private key in decryption such as public key cryptosystem. Encryption is the process of encrypting plaintext into ciphertext. Decryption is the process of decrypting ciphertext back to plaintext.

Steganography is about hiding a secret message into another media file. The secret message and the media file can be the form of image, video or audio. The goal of steganography is to hide the secret message from a third party. Commonly, method for hiding message into image is using least significant bits. Audio steganography and video steganography also using least significant bits (LSB) algorithm. However, what makes audio and video steganography different from image steganography is audio and video steganography need masking. Human eye and ear are sensitive to any change in audio and video. Masking can exploit the properties of human sensory organ to hide information unnoticeably.

Audio steganography software is available on the web. Example of audio steganography softwares are Mp3Stego and DeepSound. Most audio steganography uses LSB algorithm. Secret message is embedding in a faint sound meanwhile the larger audible sound in the same audio act as a cover. Hence, human could not detect audible change in the audio.

To date most audio steganography use block cipher such as AES or Data Encryption Standard (DES) to encrypt the secret message. It is a good practice for security. However, the encrypted message may become too long to embed in audio and cause distortion of cover audio if the secret message is too long. Hence, there is a need to encrypt the message with stream cipher before embedding the message into the audio. This is because stream cipher provide message length encryption meanwhile block cipher provide a fixed length of bits encryption that result a longer output compare to stream cipher. Besides that, stream cipher is faster than block cipher. Encryption of message is to protect the confidentiality of the message. Which mean, only the authorised people can retrieve the secret message from the audio together with secret key.

2. Related Works

In this section, background materials that are used to build up the audio steganography with embedding text with RC4 encryption cipher are presented.

2.1. Cryptograpgy

Cryptography is the art and study of analysing secret writing [1]. The goal of cryptography is to hide the meaning of the secret message. The secret message is restructuring into an unreadable message with the help of an algorithm together with a secret key. The algorithm is public known. The secret key is private. A simple cryptosystem includes encryption of plaintext into ciphertext and decryption of ciphertext into plaintext. A basic cryptographic system consists of five elements that are plaintext, ciphertext, key, encryption algorithm and decryption algorithm. Plaintext is the secret message in its original form. Ciphertext is the mangled secret message. Encryption is a process that transform plaintext (or cleartext) into ciphertext (or cryptogram) [1]. Decryption is the reverse process of encryption. The algorithm of encryption and decryption is public known by anyone. However, the key is keep secret to everyone except for the sender and receiver. There are two types of ciphers in cryptosystem, namely symmetric cipher and asymmetric cipher [3].

2.2. Symmetric Cipher

Symmetric cryptosystem or private-key cryptosystem, both sender and recipient share a same secret key [3]. During encryption process, sender encrypts the secret message with the key. Recipient uses the same key to recover the secret message. Symmetric system consists of two categories which are stream ciphers and block ciphers. Example of stream ciphers are Trivium [4] and RC4 [5]. Example of block ciphers are Advanced Encryption Standard (AES) [6] and Rivest Cipher 5 (RC5) [7].

2.2.1 Stream Cipher

Stream cipher is a symmetric cipher that binary message combine with pseudorandom key stream using combination operation exclusive-or (XOR) [1]. A secret key is the initial state of a keystream. The key together with initial vector are inserted into pseudorandom byte generator that produce arbitrary length of keystream. The plaintext is also convert into stream of bits. Combination operation XOR of key stream and plaintext stream output the ciphertext stream. Stream cipher requires shorter time to encrypt a message than block cipher. Therefore, stream cipher is suitable for operation that need fast encryption.

Rivest Cipher 4 (RC4) is one of the keystream generator which generates pseudorandom key stream [1]. As with any stream cipher, the generated key stream is XORed with the binary message to produce the ciphertext. RC4 consists of two major parts namely key scheduling and pseudorandom generation as shown in Figure 2. The key scheduling algorithm permute a key which typically length between 40 and 2048 bits. The permutation is denoted as “S” as in the Figure 1. “T” is the temporary

vector. After completed the key scheduling, the key stream is generated using pseudorandom generation algorithm. The key stream is denoted as "K".

Key Scheduling Algorithm	**Pseudorandom Generation Algorithm**

for i = 0 to 255 do
S[i] = I;
T[i] = K [i mod keylen];
j = 0;
for i = 0 to 255 do
j = (j + S[i] + T[i]) mod 256;
Swap (S[i[, S[j]];
	i, j = 0
	while (true)
	i = (i + 1) mod 256;
	j = (j + S[i]) mod 256;
	Swap (S[i], S[j]);
	T = (S[i] + S[j]) mod 256;
	K = S[t];

Figure 1. The Key Scheduling Algorithm and Pseudorandom Generation Algorithm [1].

2.3. Asymmetric Cipher

Asymmetric cryptosystem or public-key cryptosystem consist of a key pair which is public key and private key [3]. For instance, sender require public key of the recipient to encrypt the secret message. After recipient receive the ciphertext, recipient can compute the private key with ciphertext to recover the plaintext. These two keys are different and computationally infeasible determine from each other. Example of asymmetric cryptosystem are ElGamal encryption [8] and Diffie-Hellman Algorithm [9].

2.4. Steganography

Steganography is the art of concealing information [10]. The embedding formula of a stenographic system may consist of cover medium, embedded message and stego key. Cover medium file type mainly consist of image, video and audio. Digital image steganography is most popular due to its frequency on the web [10]. The most common algorithm for image steganography is least significant bits algorithm. This algorithm make use of the redundant bits in an image. Function of the redundant bits in an image is to provide far more accurate details than necessary. Thus, human can alter the secret message in redundant bits without being detected easily. A successful steganography may not make attacker raise any suspicious of hidden information.

2.5. Audio Steganography

Audio steganography is a technique to hide information in audio by exploit weakness of human auditory system [11]. Audio steganography can be a more difficult task than image steganography that is perceptually invisible from user. Alteration of audio bits might change the quality sound. A successful audio steganography may produce an output that is similar with original audio and not able to detect the modification by human ear. LSB is one of the technique to hide information in digital audio. The characters of secret message are convert into binary values. Then Least Significant Bit algorithm embed each bit of message into digitized cover audio.

2.5.1Mp3Stego

Petitcolas creates an audio steganography software called Mp3Stego [12]. This software allow user to embed text file into another audio file. This software contain four modules that are encryption module, decryption module, embedding module and extracting module. The method for embedding process is LSB. The method for encryption and decryption is Triple DES (3DES).

2.5.2 DeepSound

DeepSound is another audio steganography software that allow hiding secret data into audio files [13]. The method for embedding process is LSB. The method for encryption and decryption is AES-256 bit. This software contains four modules that are encryption module, decryption module, embedding module and extracting module. The encryption algorithm used in this software is block cipher

algorithm. Block cipher is a double-edged sword. It can provide high security to data but it also can increase the file size of steganography product. It might attract others suspicious if there is change in file size.

2.6. Comparison Between Existing System and Proposed System

Table 1 shows the comparison of existing audio steganography with the proposed audio steganography. Proposed software allow user to input maximum 2048 bits of key size meanwhile both existing software is limited at maximum key size of 168 bits and 256 bits. Mp3Stego and DeepSound use block ciphers to encrypt the text meanwhile the proposed software choose stream ciphers to encrypt the text due to the speed. According to Stallings, performance of stream cipher is faster than block cipher. Stream cipher encrypt plaintext into bit size but block cipher encrypt the plaintext into block size that may also lead to larger output file size than original file size. When users want to have a secure communication with another, but he need to send many data in a short time. Stream cipher works well with lots of data in short time. However, there is a trade-off between the performance and security. Block ciphers provide higher security than stream ciphers because encrypt the secret data into an arbitrary block of data. Mp3Stego produce a minimum of 64 bits of output block size and DeepSound provide a minimum of 256 bits of output block size.

Table 1. Comparison similar audio steganographys.

Elements/Software	Mp3Stego [12]	DeepSound [13]	Proposed system
Supported Medium	Audio file (.mp3)	Audio file (.wav,.flac, .ape)	Audio file (.wav)
Secret Message	Text file	Text file	Text file
Encryption Algorithm	TDES	AES-256	RC4
Maximum key size	56, 112 or 168 bits	256 bits	2048 bits
Graphic User Interface (GUI)	No	Yes	Yes
Programming Language	C	C#, Winforms	C#

3. System Methodology

According to Dundas, this methodology has four main processes which are Object-Oriented Analysis Phase, Object-Oriented Design Phase, Object-Oriented Implementation Phase, Testing and Maintenance phase [14].

3.1. Analysis

Object-oriented analysis is the first phase in a software development. This phase is mainly about determine system requirements and function by collecting user's needs. This analysis phase helps to develop a software that is fulfil the requirements of the user's needs. Background study on audio steganography software is able to help in identify the functionalities that should be included in proposed software. To identify the security element and steganography algorithm in proposed software, stream cipher and steganography algorithm are investigated. The result of this phase leads to the following design phase.

3.2. Design

Object-oriented design phase is the second phase in software development. In this phase, all the requirements in the analysis phase is transformed into a design. It decides how the software may work with users. The primary goal of this design phase is identify and design classes for proposed system. The interaction and relationship between classes are modelled into use-case diagram, class diagram and sequence diagram using Unified Modelling Language (UML). An activity diagram is also designed to illustrate the flow of the software and how it works. Software and hardware requirement are determined after determination of system flow.

3.3. Implementation
Object-oriented implementation phase is the third phase of software development. This phase is to include all details for software to execute and develop the software. All the interaction between classes are translated into programming language. C# object-oriented programming language is chosen to develop the proposed system. The user interface is also developing to connect with functionalities of program. Every module must connect well with the user interface.

3.4. Testing and Maintenance
Testing phase is working concurrently with object-oriented implementation phase or might as well the software is in work. Errors or bugs may occur during or after implementation phase. Testing phase is to ensure the interaction between objects and classes is going well. User may test the software to ensure the quality of software and it meet user needs. Maintenance is another phase after implementation phase. A good software provides constant bug fix and function improvement from time to time. This application is constantly monitored during this phase. Remaining errors is handled during this phase.

4. Analysis and Design
This section presents the process of analysis and design of proposed application. UML diagram such as activity diagram, sequence diagram and class diagram are discussed to provide better understanding of software flow.

4.1 Sequence Diagram
Sequence diagram explains the script in the use case diagram. The sequence diagram for user to embed secret message in audio is shown in Figure 2. User inputs message, audio and password before encryption start. The message is encrypted with the password. Encrypted text is embedding into cover audio. The encrypted audio is creating and saved in computer.

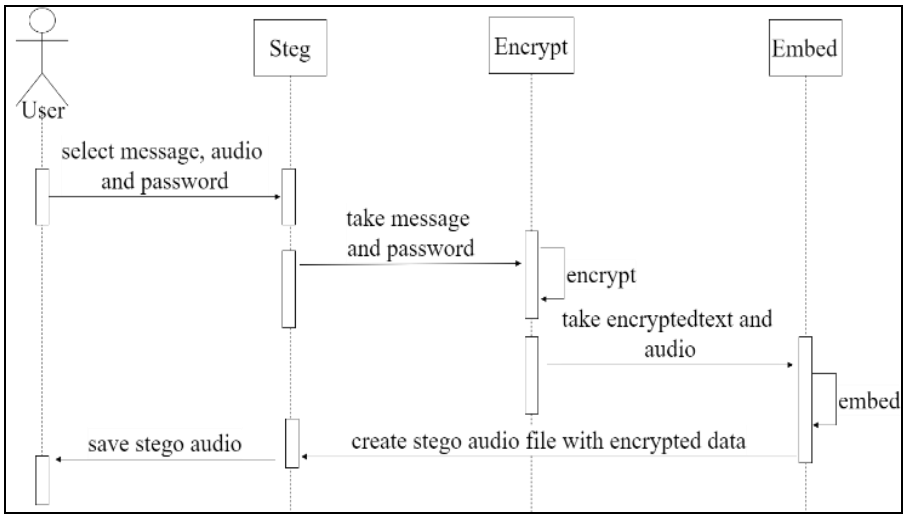

Figure 2. Sequence Diagram for user to embed secret message in audio.

Figure 3 shows the sequence diagram for user to extract and retrieve secret message from stego audio. User inputs the stego audio and password. Encrypted message is extracted from stego audio. Encrypted message is then decrypt with the password. The decrypted text is displayed to receiver.

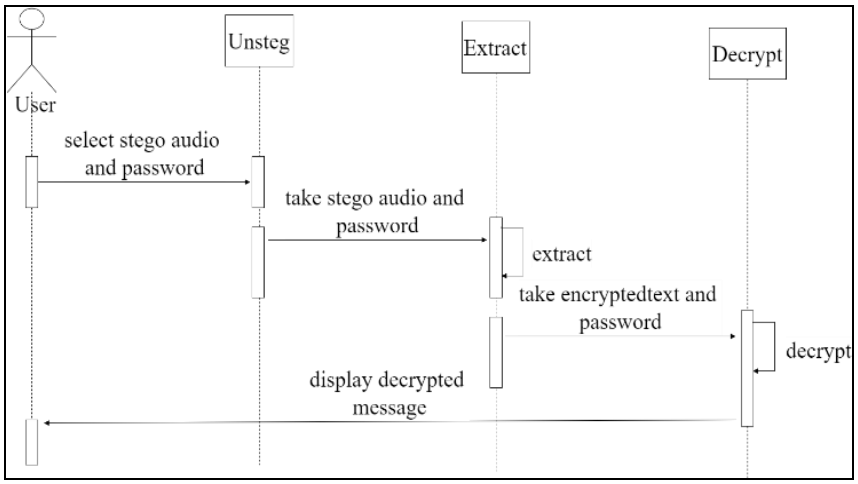

Figure 3. Sequence Diagram for user to retrieve message.

4.2 Activity Diagram

An activity diagram is used to visually present the data flow behind the system as illustrated in Figure 4. Firstly, the user starts with either input the secret message or audio file. Secret message is validated to determine whether the secret message is entered or not. Audio is validated to determine whether the audio is in wav format or not. The system only proceed if it receive the correct data. Key and cover audio is needed to generate stego audio. Key is also validated whether the key is entered or not. After user select the output folder, the system may generate the stego audio. Meanwhile, stego audio and the same key is needed to retrieve the message back. If the key is not matched, the process end, else the secret message is displayed.

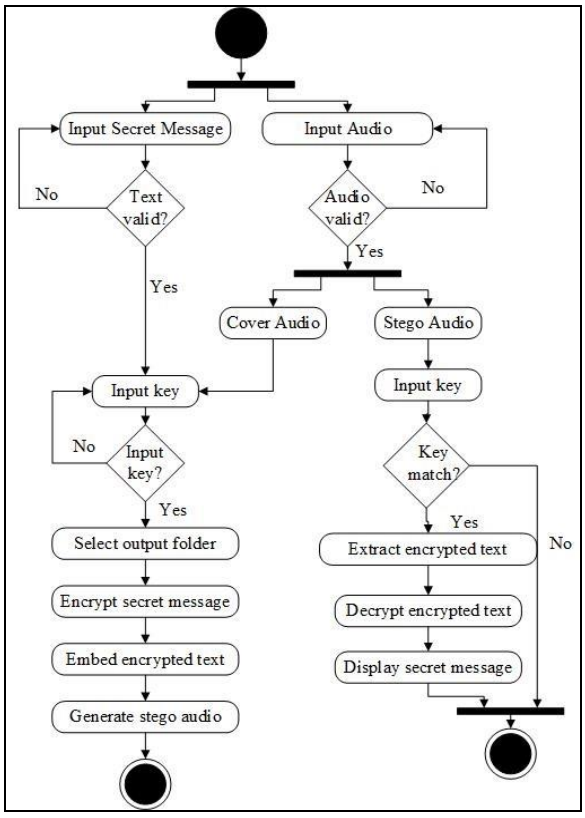

Figure 4. Activity Diagram for proposed system

4.3 Class Diagram

Class diagram describe the relationships between classes. Figure 5 shows the class diagram of proposed system.

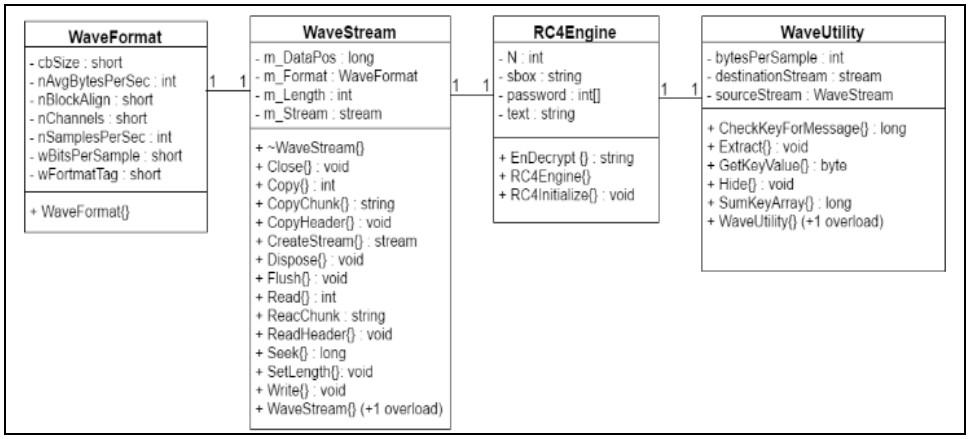

Figure 5. Class Diagram for proposed system.

In Figure 6, class diagram for proposed system is shown. There are total of four main classes in proposed system. WaveFormat class is used to define the format of .wav audio file. WaveStream class is for analysing and create a .wav audio file. Meanwhile, RC4Engine class is for user to encrypt and decrypt secret message with password. WaveUtility class is for embedding and extraction of encrypted text from stego audio.

5. Implementation of Code

In system implementation phase, C# object-oriented programming language is chosen to develop the proposed system. Figure 6 shows the RC4 encryption and decryption module implemented in proposed application.

```
public string EnDeCrypt(){
        RC4Initialize();
        int i = 0, j = 0, k = 0;
        StringBuilder cipher = new StringBuilder();
        for (int a = 0; a < text.Length; a++)  {
           i = (i + 1) % N;   j = (j + sbox[i]) % N;
           int tempSwap = sbox[i];  sbox[i] = sbox[j];
           sbox[j] = tempSwap;     k = sbox[(sbox[i] + sbox[j]) % N];
           int cipherBy = ((int)text[a]) ^ k;  //xor operation
           cipher.Append(Convert.ToChar(cipherBy)); }
        return cipher.ToString(); }
```

Figure 6. Source Code for Encryption and Decryption Module [5].

Figure 7 shows the source code for embedding module. The function of this module is to embed the encrypted message into audio. Figure 8 shows the source code for extraction module. The function of this module is to extract encrypted message from the audio.

```
public void Hide(Stream messageStream, Stream keyStream){
        byte[] waveBuffer = new byte[bytesPerSample];
        byte message, bit, waveByte; int messageBuffer, keyByte;
        while( (messageBuffer=messageStream.ReadByte()) >= 0 ){
                message = (byte)messageBuffer;
                for(int bitIndex=0; bitIndex<8; bitIndex++){
                keyByte = GetKeyValue(keyStream);
                  for (int n=0; n<keyByte-1; n++)   {
        sourceStream.Copy(waveBuffer, 0, waveBuffer.Length, destinationStream);}
                sourceStream.Read(waveBuffer, 0, waveBuffer.Length);
                waveByte = waveBuffer[bytesPerSample-1];
                bit = (byte)(((message & (byte)(1 << bitIndex)) > 0) ? 1 : 0);
                if((bit == 1) && ((waveByte % 2) == 0)){ waveByte += 1;
                }else if((bit == 0) && ((waveByte % 2) == 1)){waveByte -= 1;}
                waveBuffer[bytesPerSample-1] = waveByte;
                destinationStream.Write(waveBuffer, 0, bytesPerSample); }}
        waveBuffer = new byte[sourceStream.Length - sourceStream.Position];
        sourceStream.Read(waveBuffer, 0, waveBuffer.Length);
        destinationStream.Write(waveBuffer, 0, waveBuffer.Length);}
```

Figure 7. Source Code for Embedding Module.

```
public void Extract(Stream messageStream, Stream keyStream){
        byte[] waveBuffer = new byte[bytesPerSample];
        byte message, bit, waveByte;
        int messageLength, keyByte;
        while( (messageLength==0 || messageStream.Length<messageLength) ){
                message = 0;
                for(int bitIndex=0; bitIndex<8; bitIndex++){
                        keyByte = GetKeyValue(keyStream);
                        for(int n=0; n<keyByte-1; n++){
                        sourceStream.Read(waveBuffer, 0, waveBuffer.Length);}
                        sourceStream.Read(waveBuffer, 0, waveBuffer.Length);
                        waveByte = waveBuffer[bytesPerSample-1];
                        bit = (byte)(((waveByte % 2) == 0) ? 0 : 1);
                        message += (byte)(bit << bitIndex); }
                messageStream.WriteByte(message);
                if(messageLength==0 && messageStream.Length==4){
                messageStream.Seek(0, SeekOrigin.Begin);
                messageLength = new BinaryReader(messageStream).ReadInt32();
                messageStream.Seek(0, SeekOrigin.Begin);
                messageStream.SetLength(0);} }
```

Figure 8. Source Code for Extraction Module.

6. Performance Analysis

Functional testing is a technique to test all the functionality of this application. This is to ensure the functions work properly by feeding the input and examine the output. Here, the testing of the main functionalities of audio steganography with embedded text application are presented. Figure 9 shows the encryption module and embedding module. User may choose cover audio, enter secret key and secret message. After clicking on Encrypt button, the secret message is encrypted. Then click on the Hide button, to hide the encrypted message into the audio. Figure 10 shows the extracting module and decryption module. First user selects the audio and enter secret key. Then click on Unhide button to recover the encrypted text and Decrypt button to retrieve back the plaintext. Both major functions are operated as expected.

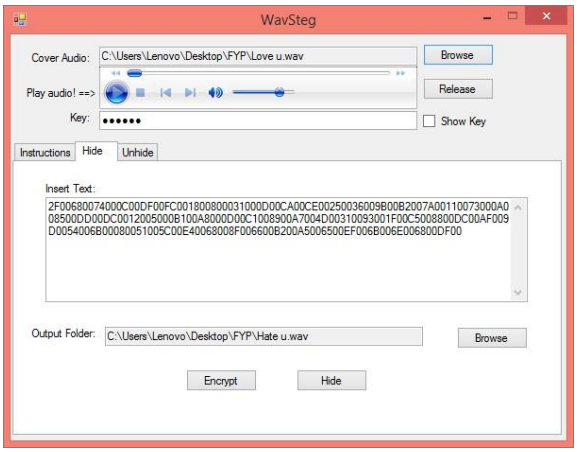

Figure 9. Encryption module and embedding module.

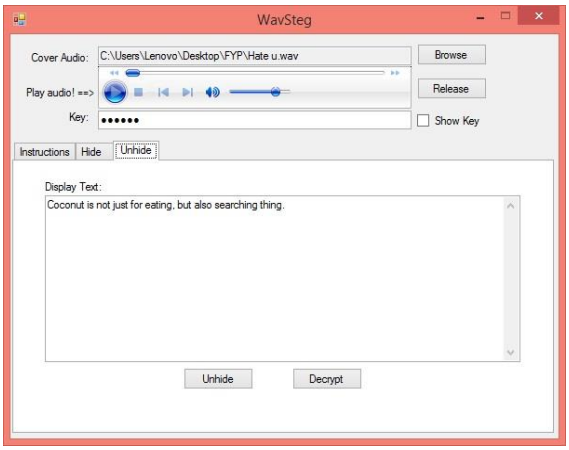

Figure 10. Decryption module and Extracting module

7. Result and Discussion

Here we provide the acceptance testing carried out with a group of 40 users. Two major feedbacks are required from the users are system functionality test (see Figure 12) and user friendly test (see Figure 13). Throughout the testing, the audio steganography with embedding text with RC4 operates 90% successfully as shown in Figure 11. The remaining 10% happened on the extraction module where users inserted incorrect password which had resulted random message extracted. Noted that the system relies on the users to insert correct password for both embedded module and extraction module. Figure 12 shows the analysis result about the audio steganography with embedding text with RC 4 design. 80% users of the users agree that the design is user friendly, 5% of users disagree and 15% users does not sure either the design is user friendly or not.

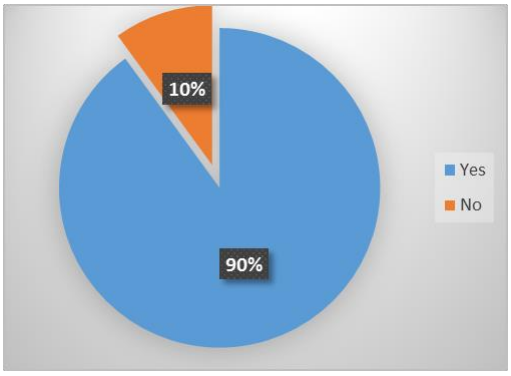

Figure 11. System Functionality Test

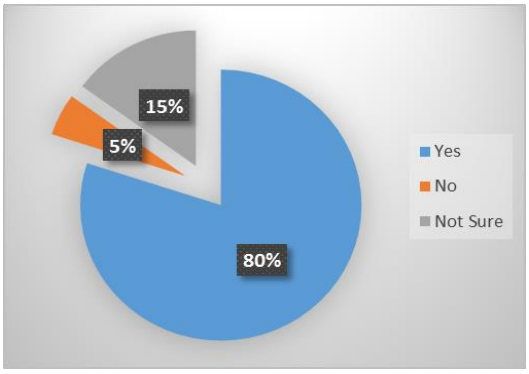

Figure 12. User Friendly

8. Conclusion

The audio steganography with embedding text with RC4 encryption cipher is designed, is developed and is tested. This application has been successfully developed and reached this targeted project aim. The application is a windows form application that enable user to hide and unhide their message in wav audio, which may provide confidentiality to the user message. However, one limitation for this application is that the application only support one audio file which is wav audio. Therefore, more variety of audio files format support such as .mp3 and .wma can be added to the application as hiding medium. This mechanism can be added through understanding and analysing the file format.

Acknowledgments
This research was supported by FRGS vot 1558, ORICC UTHM and Gates IT Solution Sdn. Bhd.

9. References

[1] Stallings, W., 2011. *Cryptography and network security: principles and practices*. Pearson Education India.
[2] Tripathi, R. and Agrawal, S., 2014. Comparative study of symmetric and asymmetric cryptography techniques. *International Journal of Advance Foundation and Research in Computer (IJAFRC)*, *1*(6), pp.68-76.
[3] Simmons, G.J., 1979. Symmetric and asymmetric encryption. *ACM Computing Surveys (CSUR)*, *11*(4), pp.305-330.
[4] Canniere, C. D., and Preneel, B., "New Stream Cipher Designs. Trivium, 244- 266. Berlin: Springer," (2006)
[5] Rivest, R.L. and Schuldt, J.C., 2014. Spritz—a spongy RC4-like stream cipher and hash function. *Proceedings of the Charles River Crypto Day, Palo Alto, CA, USA*, *24*, p.10.
[6] Daemen, J. and Rijmen, V., 1999. AES proposal: Rijndael.
[7] Rivest, R.L., 1994, December. The RC5 encryption algorithm. In *International Workshop on Fast Software Encryption* (pp. 86-96). Springer Berlin Heidelberg.
[8] ElGamal, T., 1985. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, *31*(4), pp.469-472.
[9] Diffie, W. and Hellman, M., 1976. New directions in cryptography. *IEEE transactions on Information Theory*, *22*(6), pp.644-654.
[10] Kumar, A. and Pooja, K., 2010. Steganography-A data hiding technique. *International Journal of Computer Applications*, *9*(7), pp.19-23.
[11] Divya, S.S. and Reddy, M.R.M., 2012. Hiding text in audio using multiple LSB steganography and provide security using cryptography. *International journal of scientific & technology research*, *1*(6), pp.68-70.
[12] Petitcolas, F.A., 1998. mp3stego.
[13] Batora, J., DeepSound Overview. Retrieved from http://jpinsoft.net/DeepSound/Overview.aspx.
[14] Dundas, J., 2007. The Software Development Life Cycle. Software Testing, pp. 29-58.