A Technique to Encrypt-decrypt Stereo Wave File

Jihad Nadir, Ashraf Abu Ein and Ziad Alqadi*
Computer Sngineering Department
Albalqa Applied University
Amman - Jordan
*Email: natalia_maw [AT] yahoo.com

Abstract—**Stereo wave files are now very popular type of data which transmitted and received through the internet , thus this data transmission needs a highly secure method of sending and receiving stereo wave files. This research produces a powerful technique of encryption-decryption wave files without damaging the original file and without losing any piece of information from the original file. The proposed technique is very efficient and the efficiency is achieved by decreasing the encryption and decryption times and improving the data security by making the hacking attempts impossible. The throughput of this technique will be compared with the throughputs of anther available technique of data encryption/decryption.**

Keywords: Stereo wave file, encryption, decryption, private key, throughput .

I. INTRODUCTION

Waveform Audio File Format is a Microsoft and IBM audio file format standard for storing an audio bit stream on PCs. It is an application of the Resource Interchange File Format (RIFF) bit stream format method for storing data in "chunks", and thus is also close to the 8SVX and the AIFF format used on Amiga and Macintosh computers, respectively[1]. It is the main format used on Windows systems for raw and typically uncompressed audio. The usual bit stream encoding is the linear (LPCM) format[2].

Sound is basically a pressure wave or mechanical energy having pressure variance in an elastic medium. The variance propagates as compression and rarefaction wherein compression occurs when pressure is higher than the ambient pressure and rarefaction occurs when the pressure of the propagating wave is less than the ambient pressure[3], [4].

Exactly in the same manner a WAVE file just represents the sampled sound waves which happen to be above or below the equilibrium or ambient air pressure. In this paper we will be using a "drums.wav" wave file to show the proposed algorithm of encrypting the sound file in various image formats [5],[6]. As already mentioned a wave file consists of positive and negative values over its entire range of samples.

Currently, the exchange data among users are rapidly grown so that users require to secure their data systems (i.e. video, audio, image, and text) in order to have confidentiality of the data from attackers. These security systems are widely used in the database area such as internet banking and audio communication channels. Therefore, the security systems are important aspects in information systems that many researchers continue to develop especially in the field of audio security.

In this paper, encryption technique using WAV file format that has a header and a data structure is applied. This research aims to create a system that can encrypt stereo wave files without damaging the structures of the bit file. Unlike other encryption methods that use substitution and shift, this system needs to obtain an original structure of the voice data therefore there is no reduction or increase of the structure of the bit file [7], [8], [9], [10]. The principle of this system is the use of special private key which is to be created randomly.

This key is very secure because of the following:

- It is a large 2 dimensional matrix.
- The elements of the key are double.
- The matrix is to be created randomly.
- The matrix key is known only by the sender and receiver.
- The key is to be associated by the wave file size.

Stereo wave file is a 2 dimensional double matrix with 2 columns, while a mono wave file is a one channel signal represented by one column double matrix.

II. THE PROPOSED TECHNIQUE

The proposed technique is divided in 2 phases:

- **Encryption phase**
- **Decryption phase**

The encryption phase has to implemented using the following sequence of steps:

1. Get the original stereo file (sf).
2. Find the size of sf and save it as(s).
3. Reshape sf into one column matrix(c1).
4. Find the nearest bigger square number to c1 size .
5. Convert c1 to square matrix (c2) by padding zeros to c1.
6. Generate a random double square matrix (pk) with size equal c2 size.
7. save pk.
8. Get the encrypted file(ef) by applying matrix multiplication of c2 and pk.

The decryption phase can be implemented by applying the following sequence of steps:

1. Get the encrypted file (ef).
2. Get the necessary information (pk and s).
3. calculate the inverse of pk (pki).
4. Apply matrix multiplication of ef and pki to get decrypted file (df).
5. Reshape df into one dimensional matrix (efr).
6. Omit padded zeros from efr.
7. reshape efr into 2 columns matrix to get the original file.

III. IMPLEMENTATION

The following matlab code was written to implement the proposed technique.

```
clc,clear all,close all
%get the wave file
[a fs]=wavread('C:\Users\User\Desktop\wave files\antidprs.wav');
%play the wave file
sound(a,fs)
%Change 2 channel wave file to 1 channel
b=reshape(a,121451 *2,1);
% convert wave file to 2 dimensional matrix by pading zero
for i=242903:243049
    b(i,1)=0;
end
    c=reshape(b,493,493);
    %Generate encryption key
    d=rand(493,493);
    % Apply encryption
    e=c*d;
    %Apply decryption
    de=e*inv(d);
    %Change the matrix to 2 channel wave file
    r1=reshape(de,493*493,1);
    for i=1:242902
        r2(i,1)=r1(i,1);
    end
        r3=reshape(r2,121451,2);
        sound(r3,fs)
        % Calculate the correlation between the original
        %wave file and the decrypted one
       corr2(a,r3)
```

The code was implemented several time using various stereo wave file and it was shown that the original wave file and the decrypted one have the same characteristics as shown in figures 1 and 2.

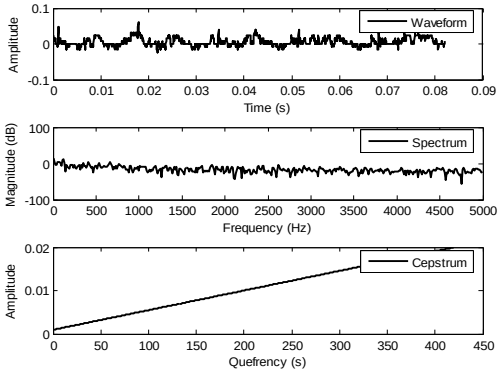

Figure 1: Characteristics of original wave file

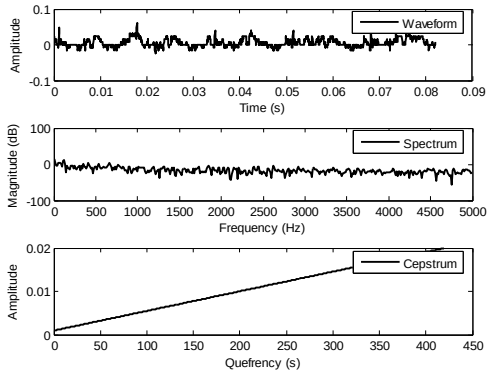

Figure 2: Characteristics of decrypted wave file

These figures were generated by implementing the following code:

```
[x,fs]=wavread('annano.wav',[24120 25930]);
ms1=fs/1000;                % maximum speech Fx at 1000Hz
ms20=fs/50;                 % minimum speech Fx at 50Hz
%
% plot waveform
t=(0:length(x)-1)/fs;       % times of sampling instants
subplot(3,1,1);
plot(t,x);
legend('Waveform');
xlabel('Time (s)');
ylabel('Amplitude');
%
% do fourier transform of windowed signal
Y=fft(x.*hamming(length(x)));
%
% plot spectrum of bottom 5000Hz
hz5000=5000*length(Y)/fs;
f=(0:hz5000)*fs/length(Y);
subplot(3,1,2);
plot(f,20*log10(abs(Y(1:length(f)))+eps));
legend('Spectrum');
xlabel('Frequency (Hz)');
ylabel('Magnitude (dB)');
%
% cepstrum is DFT of log spectrum
C=fft(log(abs(Y)+eps));
%
% plot between 1ms (=1000Hz) and 20ms (=50Hz)
q=(ms1:ms20)/fs;
subplot(3,1,3);
%plot(q,abs(C(ms1:ms20)));
plot(q);
legend('Cepstrum');
xlabel('Quefrency (s)');
ylabel('Amplitude');
```

Here are some numerical results:

- Sample of original wave file:

```
>> a(1:15,:)

ans =

   -0.0156    0.0078
   -0.0156   -0.0313
   -0.0156   -0.0781
   -0.0313   -0.1016
   -0.0313   -0.0781
   -0.0313   -0.0625
   -0.0313   -0.0781
   -0.0313   -0.0859
   -0.0313   -0.0547
   -0.0313   -0.0391
   -0.0313   -0.0547
   -0.0313   -0.0625
   -0.0391   -0.0625
   -0.0313   -0.0625
   -0.0313   -0.0781
```

- Sample of converting wave file into one channel (mono) file:

```
>> b(1:15)

ans =

   -0.0156
   -0.0156
   -0.0156
   -0.0313
   -0.0313
   -0.0313
   -0.0313
   -0.0313
   -0.0313
   -0.0313
   -0.0313
   -0.0313
   -0.0391
   -0.0313
   -0.0313
```

- Sample of converting mono wave file into 2 dimensional matrix

```
>> c(1:10,1:3)

ans =

   -0.0156   -0.0156   -0.2109
   -0.0156   -0.0156   -0.1719
   -0.0156   -0.0156   -0.1406
   -0.0313   -0.0156   -0.0781
   -0.0313   -0.0156    0.0313
   -0.0313   -0.0156    0.1328
   -0.0313   -0.0156    0.1875
   -0.0313   -0.0156    0.2188
   -0.0313   -0.0156    0.2188
   -0.0313         0    0.2031
```

- Sample of the private key:

```
>> d(1:10,1:3)

ans =

    0.9855    0.4907    0.3756
    0.2397    0.4552    0.7257
    0.7412    0.4942    0.5974
    0.1268    0.6464    0.1908
    0.7459    0.2084    0.8844
    0.9077    0.0636    0.6519
    0.0477    0.7211    0.5175
    0.3363    0.8836    0.8846
    0.8889    0.7054    0.1283
    0.0034    0.7129    0.4659
```

- Samples from the encrypted file

```
>> e(1:10,1:3)

ans =

    1.1241   -2.2642    2.1357
    0.3076   -3.2237    2.2132
   -1.3577   -4.5420    0.5204
   -2.8383   -5.1170   -1.2769
   -3.6187   -5.6302   -2.6917
   -4.4305   -6.4201   -4.4138
   -3.6509   -5.8094   -4.6207
   -2.7097   -4.4060   -3.5900
   -1.4676   -3.0444   -2.1747
    0.2025   -0.8359    0.0852
```

- Samples from the decrypted file

```
>> de(1:10,1:3)

ans =

   -0.0156   -0.0156   -0.2109
   -0.0156   -0.0156   -0.1719
   -0.0156   -0.0156   -0.1406
   -0.0313   -0.0156   -0.0781
   -0.0313   -0.0156    0.0313
   -0.0313   -0.0156    0.1328
   -0.0313   -0.0156    0.1875
   -0.0313   -0.0156    0.2188
   -0.0313   -0.0156    0.2188
   -0.0312   -0.0000    0.2031
```

- Samples from the reshaped decrypted file

```
>> r3(1:15,:)

ans =

   -0.0156    0.0078
   -0.0156   -0.0313
   -0.0156   -0.0781
   -0.0313   -0.1016
   -0.0313   -0.0781
   -0.0313   -0.0625
   -0.0313   -0.0781
   -0.0313   -0.0859
   -0.0313   -0.0547
   -0.0312   -0.0391
   -0.0312   -0.0547
   -0.0313   -0.0625
   -0.0391   -0.0625
   -0.0313   -0.0625
   -0.0313   -0.0781
```

All the results showed that the original file and the decrypted one are the same and the correlation coefficient between them was equal 1, which means that there is no any damage in the information.

IV. EXPERIMENTAL RESULTS

The proposed technique was implemented using Intel Core i3-3120M CPU of 2.50 GHz CPU speed with 4 GB RAM. Some of the experimental results are shown in table 1.

Table 1: Experimental results

Wave file	Size (kb)	Encryption time(s)	Decryption time(s)
Bird	307	0.005432	0.071008
Cow	120	0.001771	0.021080
Dog	312	0.003197	0.032283
Dolphin	157	0.002652	0.035743
Donkey	464	0.009416	0.098309
Duck	802	0.018260	0.152848
Elephant	395	0.005723	0.072021
Horse	60.7	0.001081	0.020075
Lion	530	0.010260	0.150148
Rooster	314	0.005532	0.071108
Sum	3461.7	0.0633	0.7246
Average time for each kb		18.2858 microsecond	209.3191 microsecond

From table 1 we can see that a one Kbyte of wave file needs 18.2858 micro seconds to be encrypted and 209.3191 micro seconds to be decrypted (in average), and the difference between the encryption and decryption time is due inverse matrix calculation during the decryption phase.

From these results we can see the efficiency of the proposed technique, also this technique provides a high degree of security because the private key used for encryption-decryption is vary huge and it is very difficult to hack it, table 2 shows the sizes of private keys for some selected wave files.

Table 2: Private key size

Wave files size(double elements)	Private key size(double elements)
78588*2	397*397
242881*2	697*697
30964*2	249*249
46069*2	304*304
39294*2	281*281
118944*2	488*488
205536*2	642*642

V. RESULTS DISCUSSIONS

The throughput for encryption as well as decryption is calculated one by one. Encryption time is used to calculate the throughput of an encryption technique .The throughput of the encryption technique is calculated by dividing the size of wave file in MB by total encryption time in second. If the throughput value is increased, the power consumption of this encryption technique is decreased .Similar procedure has been followed to calculate the throughput of decryption technique. For my experiment. The performance metrics are analyzed by :
(a) Encryption/decryption time. (b) CPU process time – in the form of throughput.
Throughput = Plain Text (MB) / Encryption or decryption time (Sec.)

In [11] a comparative analysis of encryption-decryption techniques was proposed and i will add the results obtained by the proposed techniques for comparisons.

Table 3 shows these results:

Table 3: Comparative analysis

Wave file size(MB)	DES encryption /decryption time(sec)	Blowfish encryption /decryption time(sec)	Proposed technique encryption time(sec)	Proposed technique decryption time(sec)
10	12	52	0.189	2.153
20	12	59	0.431	3.274
30	16	122	0.645	6.852
40	16	152	0.782	8.901
50	18	155	0.957	11.437
Average time	14.8	108	0.6008	6.5234
Throughput	2.07	0.27	49.9334	4.5988

From table 3 we can see that for deferent techniques of wave file encryption-decryption encryption/decryption time varies proportionally according to the size of data. For all block cipher techniques that are analyzed, with increase in key size, encryption time also increases, and in the worst case(decryption phase) the throughput is around 2 times of the DES throughput.

VI. CONCLUSIONS

A proposed technique for wave file encryption/decryption was proposed, tested and implemented. It was shown that this technique is highly efficient comparing with other available techniques, the experimental results showed that this method of encryption/decryption is minimum 2 times faster than DES method. The proposed method apply encryption decryption without error and without any

damage of information, also it is very secure because it is very hard to hack the huge size private key.

REFERENCES

[1]:Shital C. Patil, R. R. Keole, "Cryptography, Steganography & Network Securities", "International Journal of Pure and Applied Research in Engineering and Technology", 2012; Volume 1(8): pp. 9-15.

[2]: Animesh Kr Trivedi, Rishi Kapoor, Rajan Arora, Sudip Sanyal and Sugata Sanyal, RISM - Reputation Based Intrusion Detection System for Mobile Ad hoc Networks, Third International Conference on Computers and Devices for Communications, CODEC-06, pp. 234-237. Institute of Radio Physics and Electronics, University of Calcutta, December 18- 20, 2006, Kolkata, India.

[3]: Nosrati Masoud, Ronak Karimi, Hamed Nosrati, Ali Nosrati, "Taking a Brief look at steganography: Methods and Approaches", Journal of American Science, vol.7, Issue 6, 2011, pp. 84-88.

[4]: Dey, Sandipan, Ajith Abraham, and Sugata Sanyal. "An LSB Data Hiding Technique Using Natural Number Decomposition", IEEE Third International Conference on Intelligent Information Hiding and Multimedia Signal Processing, 2007. IIHMSP 2007, vol.2. 2007, pp. 473-476.

[5]: Cui, Zhihua, Chunxia Yang, and Sugata Sanyal. "Training artificial neural networks using APPM", International Journal of Wireless and Mobile Computing 5.2 (2012): 168-174.

[6]: R. A. Vasudevan, A. Abraham, Sugata Sanyal, D.P. Agarwal, "Jigsaw-based secure data transfer over computer networks", Int. Conference on Information Technology: Coding and Computing, pp. 2-6, vol.1, April, 2004.

[7]: M. Kaur and S. Kaur, "Survey of Various Encryption Techniques for Audio Data," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 4, pp. 1314-1317, 2014.

[8]: R. Munir, Kriptografi. 2006. Bandung: Penerbit INFORMATIKA. Bandung, Indonesia.: Informatika, 2006.

[9]: A. A. Tamimi and A. M. Abdalla, "An Audio Shuffle - Encryption Algorithm," in The World Congress on Engineering and Computer Science 2014 WCECS 2014, 22-24 October, 2014, San Francisco, USA, 2014.

[10]: S. M. Seyedzadeh, B. Norouzi, and S. Mirzakuchakib, "RGB color image encryption based on Choquet fuzzy integral," The Journal of Systems and Software, vol. 97, pp. 128–139, 2014.

[11]: Srinivas B.L, Anish Shanbhag, Austin Solomon D'Souza, A Comparative Performance Analysis of DES and BLOWFISH Symmetric Algorithm, International Journal of Innovative Research in Computer and Communication Engineering *(An ISO 3297: 2007 Certified Organization)* Vol.2, Special Issue 5, October 2014.