An Audio Encryption Using Transposition Method

Ahmad Jawahir[a,1], Haviluddin[b,2]

[a] *Migent Software Researcher, Jakarta - Indonesia*
[b] *Dept. of Computer Science, Faculty of Mathematics and Natural Science, Mulawarman University - Indonesia*
[1] *ahmadjawahirabd@gmail.com;* [2] *haviluddin@unmul.ac.id*

ARTICLE INFO

Article history:
Received
Revised
Accepted

Keywords:
Encryption
Decryption
Audio
Transposition
WAV
MSE

ABSTRACT

Data security techniques including sound from attackers is vital and continue to be required in which the voice data randomization technique can be done by using encryption. In this study, transposition technique which corresponds to a WAV file extension is used. Then, the performance of the transposition technique is measured by using the mean square error (MSE). In the test, the value of MSE of original audio files and encrypted audio files were compared; the original audio files and decrypted audio files with the correct password is 'SEMBILAN' and the incorrect password is 'DELAPAN'. The experimental results showed that the original audio files and audio files encrypted, the original audio files and audio files decrypted with the correct password has a value of MSE = 0, and with the incorrect password has a value of MSE 0.00000428 or ≠ 0. In other words, the transposition technique is able to ensure the security of audio data well.

I. Introduction

Currently, the exchange data between users is 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. This security system is widely used in the database area such as internet banking and audio communication channels. Therefore, the system security is one important aspect in an information system that many researchers continue to develop, especially in the field of audio security.

Today, most security systems used are encryption techniques. Security encryption technique is a technique that uses specific algorithms to maintain data security. In principle, the process in the encryption technique is to change the data with specific algorithms so that the data become unintelligible. Therefore, if the encryption technique is applied to the audio data security, it would be difficult for the attackers to know and to hear audio contents because the sound is scrambled [1-4].

In this paper, one of the encryption techniques applied is using the WAV file format which is herein, the format has a header and a data structure. The purpose of this research is to create a system that can encrypt audio files without damaging the structures of the bit file. Unlike other encryption methods that use substitution and shift, this system needs to obtain the original structure of the voice data, so there is no reduction or increase of the structure of the bit file [2, 3, 5, 6]. The working principle of this system is the exchange position of the bit file structure in which the system uses the transposition technique, to exchange audio chunks of data. If the audio data is played back, then the user can still hear the sounds scrambled. Therefore, to encrypt the data audio file, the user needs to enter a key that serves to influence the outcome file encryption. Meanwhile, the key length depending on the size of the audio file. The larger the file size the larger the key length will be.

Based on the problems, this paper will apply one method of encryption is the transposition technique. This method will be applied to the voice data that has WAV file extension. Section 2 of this paper presents the construction of an encryption technique using the transposition algorithm. Analysis of the proposed transposition technique is in section 3. Section 4 is the results of our experiment while section 5 presents the conclusion.

II. Methodology

In general, the transposition technique consists of encryption and description. The encryption process begins with preprocessing segmentation of voice data. Each audio chunks of data are separated by intervals of

bits in which the form of chunks of audio data array will be used as an input in the transposition. The next procedure is the process of transposition. This process will swap the index array for audio encrypt the data chunks. The key is going to affect the outcome of the exchange index. After the index interchangeable, audio chunks of data that will be redeveloped into a new audio file in which the audio file is the result of the system's encryption. The next process is decrypted audio file encryption results with the goal is to get back to the original sound. This process requires the correct key to be used to encrypt the audio file. However, if the key is different, then the decryption result is not going to sound like the original sound.

A. WAV Document

The WAV stands for Waveform Audio Format. The WAV format is part of Microsoft's RIFF (Resource Interchange File Format) specification as a storage of multimedia files into chunks. The WAV file consists of three parts, namely main chunk, chunk format, and data chunk [5, 7, 8]. A WAV file structure can be seen in **Table 1**.

Table 1. Format of WAV document

Endian	Offset Document (byte)	Attribute Name	Attribute Size (byte)	*RIFF Chunk Descriptor*
Big	0	*Chunk ID*	4	"RIFF" chunk format, the format of concern here is the "WAVE" in which requires two sub-chunk, namely: "*fmt*" and "*data*"
Small	4	*Chunk Size*	4	
Big	8			
Big	12			
Small	16	*Format*	4	
Small	20			
Small	22			
Small	24			
Small	28	*Sub-Chunk1 ID*	4	*Sub-chunk* "*fmt*" in which describes the voice information in the *data sub-chunk*.
Small	32	*Sub-Chunk1 Size*	4	
Small	34	*Audio Format*	4	
Big	36	*Num Channels*	2	
Small	40	*Sample Rate*	2	
	44	*Byte Rate*	4	
		Block Align	2	
		Bitspersample	2	
		Sub-Chunk2 ID	4	*Sub-chunk* "data"
		Sub-Chunk2 Size	4	
		Data	*Sub-Chunk2 Size*	Indicates the size of the sound information and contains raw sound data

B. Transposition Encryption

In the transposition cipher, the letters in the plaintext remains the same, only the order changed. In other words, this algorithm will perform transpose the entire range of characters in the text. Other name for this method is a permutation or scrambling, because of transposition every character in the same text with these character permutations. In this study, a system that will be created by performing the transposition twice, namely transposition by columns and by rows [2, 3, 5].

B.1. Encryption Process

The encryption process begins with the reading of the original WAV file by using WAV reader. Further, to separate the audio data to be encrypted and residual unused chunk then the audio preprocessing of data is done by inserting the key. The audio of data encrypted by the column, and next continued with randomization by line. Results of randomization is then recombined with the residual chunk and subsequently written into a WAV file. The encryption process in this study can be seen in **Fig. 1**.

B.2. Preprocessing Audio Data and Residual Chunk Separation

In this section, WAV reader is used to read data bit audio files. Data bits starts from 0 until the end of the bit data length audio file. Then, the interval is used to make the piece sound. In this study, a sample audio file has a data length of 288,000 bytes with the value interval is 1,024 bytes so that the file has 288,000 / 1,024 = 281 chunks. When the result is written back to WAV files, leftover bits are inserted into the data output audio. In this experiment, the remaining bits are 288 000 - (281 x 1,024) = 256 Byte. Furthermore, audio sets of data preprocessing, and then audio preprocessing of data dismembered and put into an array, sample pieces can be seen in **Table 2**.

B.3. Transposition key column-table encryption

The process begins with inserting the indices of audio chunks of data into the table as a horizontal row in which sorted key is arranged and followed by exchanging the position of table column. After that, the indices of audio chunks of data is read vertically and formed into a new order indices which can be seen in **Fig. 2**.

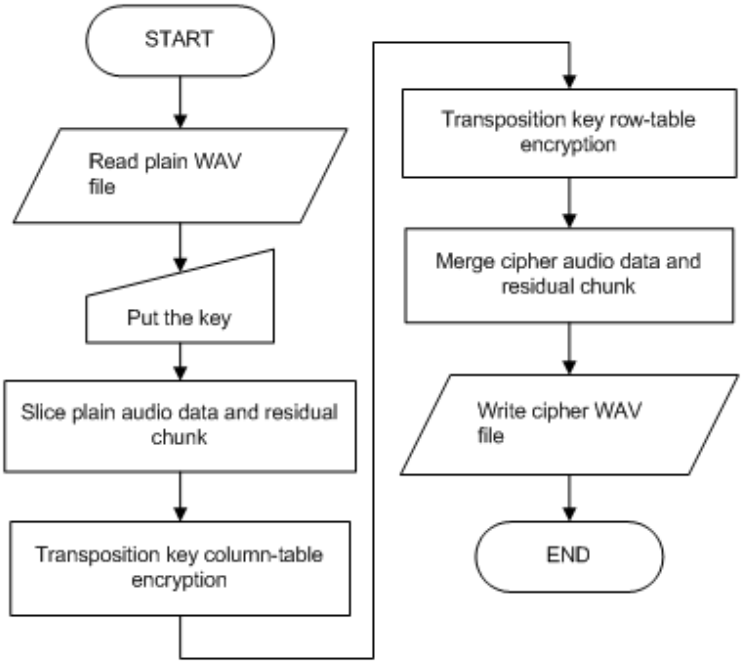


Fig. 1. Flowchart of encryption process

Table 2. Sample of audio data chunks

Array Index	First Bit	Last Bit
0	0	1,024
1	1,024	2,048
2	2,048	3,072
...	...	...
...	...	...
279	285,696	286,720
280	286,720	287,744

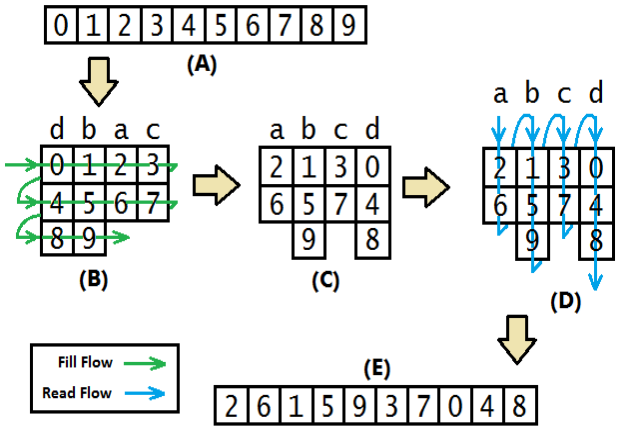


Fig. 2. Indices of audio chunks of data (A), Inserting Table Column (B), Ordering Column Based-on key character (C), Reading Table Column (D), and The results of Column Encryption (E)

B.4. Transposition key row-table encryption

The process begins with inserting the indices of audio chunks of data into the table rows vertically. Then, the keys are sorted and followed by exchanging the position of the table row. After that, the indices of audio chunks of data are read horizontally and formed into a final sequence of indices that can be seen in **Fig. 3**.

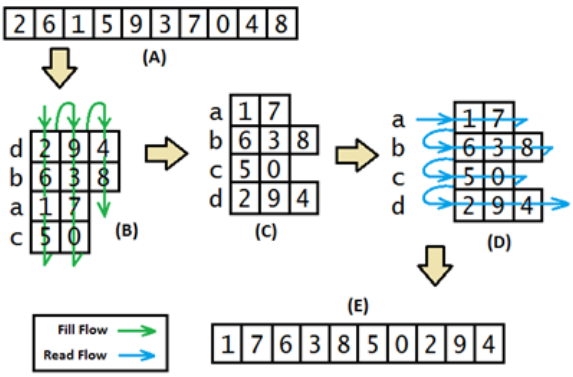


Fig. 3. Indices of Results Encryption Column (A), Inserting Table Row (B), Line Ordering Based on Key Character (C), Reading Table Line (D), and the results of the Line Encryption (E)

C. Decryption Process

Decryption process is a reversal of the encryption process in which the process of inserting the same key when the encryption process into a WAV file using a reader reading of WAV. After the preprocessing of the audio data is done with the intention to separate the audio data to be encrypted and unused residual chunk. The audio data are then encrypted based on the lines which is then continued with randomization by columns. The results of randomization is then recombined with the residual chunk and written into a WAV file. The decryption process in this study can be seen in **Fig. 4**.

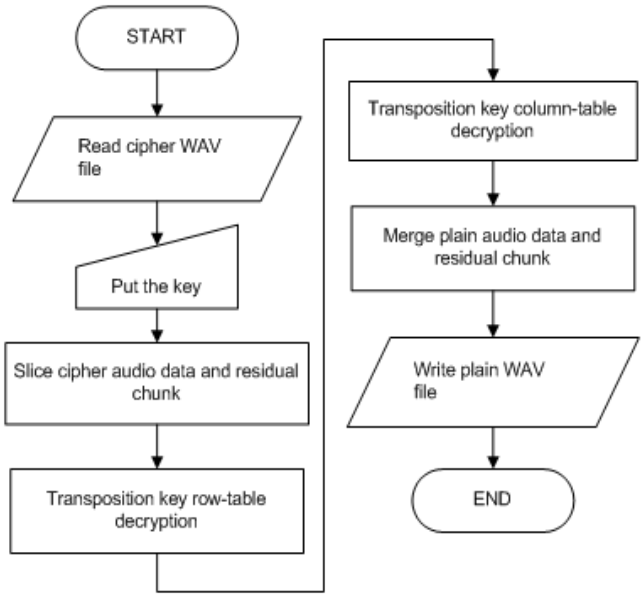


Fig. 4. Flowchart of decryption process

C.1. Transposition key row-table decryption

The first step before starting to form transposition tables is to make an array of length of each key. In this transposition method, there is a table cell is bypassed, so there are multiple rows or columns may not intact. Therefore, the value of *m* and *n* should be determined in advance before forming the array in which *m* values found using the following algorithm.

```
m = 1
While(True)
    If m x Length(Key) < Length(AudioData) Then
        Break
    End If
Loop
```

After finding the value of *m*, then the value of *n* can be found by using the following calculation.

```
n = Length(AudioData) - ((m - 1) x Length(Key)
```

Thus, the length of the array which is the same key, will be valued sequentially. This array is labelled using a key character. Then, the value of *m* is inserted as the value of *n* and the remainder is filled by *m - 1*. After that, a key is sorted, so the position of the array elements will also be exchanged in accordance with the character key. Examples of this array formation as can be seen in **Fig. 5**.

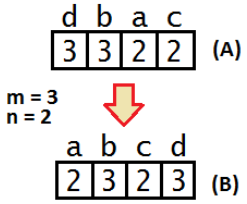


Fig. 5. Array of length of each key by unsorted key (A), Array of length of each key by sorted key (B)

After getting an array of length of each used key, then the process is continued by incorporating indices of audio chunks of data into the table rows vertically. The row table follows the key positions that have been sequenced. Furthermore, the length of lines for each character following the key value in the array of length of each key. Then, the original key is returned in accordance sequence and followed by exchange position table row. After that, all the audio index chunks of data read are horizontally and then formed into a sequence of data chunks new audio as seen in **Fig. 6**.

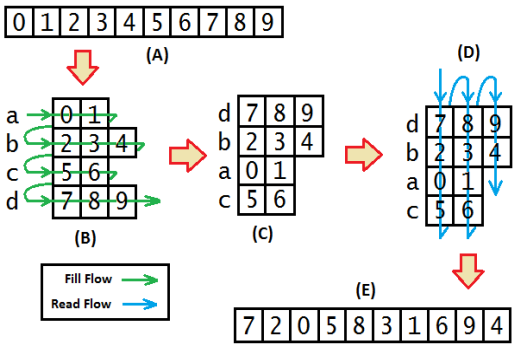


Fig. 6. Indices of audio data chunks (A), An inserting table key row (B), Line ordering based-on key character (C), Reading table lock line (D), and Decryption results classified (E)

C.2. Transposition key column-table decryption

From the array of length of each formed key, the process is continued by the insertion of audio data chunks indices which has been formed from the transposition of the key table rows. These indices of audio chunks of data is inserted into the table rows vertically. The row table follows the key positions that have been sequenced. Meanwhile, the column length for each character follows the key value in the array of length of each key. Next, the original key is returned in accordance to the sequence and followed by the exchange position of table row. After that, the indices of audio chunks of data are read horizontally and then set up into a sequence of audio indices of the last chunks of data. This can be seen in **Fig. 7**.

D. Merger of Audio Data Chunks and Residual Bit

The last process is the union of indices of audio chunks and leftover bits of data. The rest of the bits that have been previously separated are then recombined. The results of this merger is written into WAV files by using the WAV *writer*.

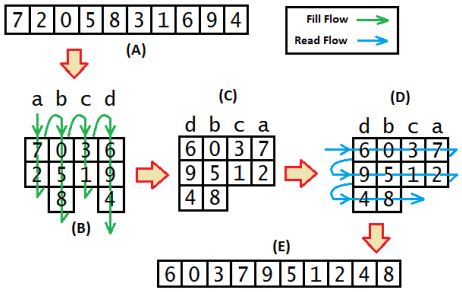


Fig. 7. Indices results line encryption (A), An Inserting table key column (B), Ordering table column based-on key character (C), Reading table key column (D), and Decryption results column (E)

III. Experimental

In this experiment, to run the transposition algorithm, the encryption and decryption program created by using C # programming. Then, to analyze amplitude sound original file has been used MATLAB R2013b. The original WAV data with names *cartoon008.wav* (http://static1.grsites.com/archive/sounds/cartoon/cartoon008.wav) has been used. In this experiment, the input encryption and decryption program is simply made, with two buttons "*Encrypt*" and "*Decrypt*" and a "*bar*" property WAV data. The process of the "*Encrypt*" is used to encrypt the data WAV, while the "*Decrypt*" is used to perform the decryption process of the WAV data that has been encrypted. The GUI of input program can be seen in **Fig. 8**. Then, the results of the original amplitude of the sound file is displayed on MATLAB which can be seen in **Fig. 9**.

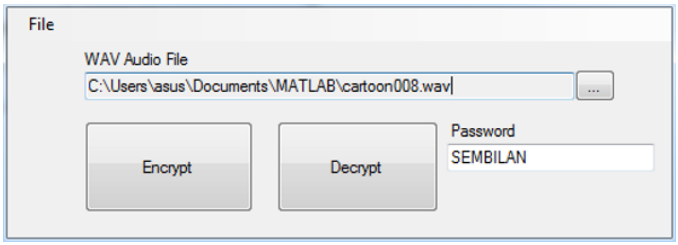


Fig. 8. GUI of input program

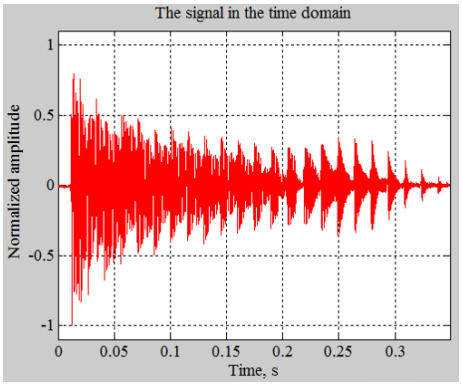
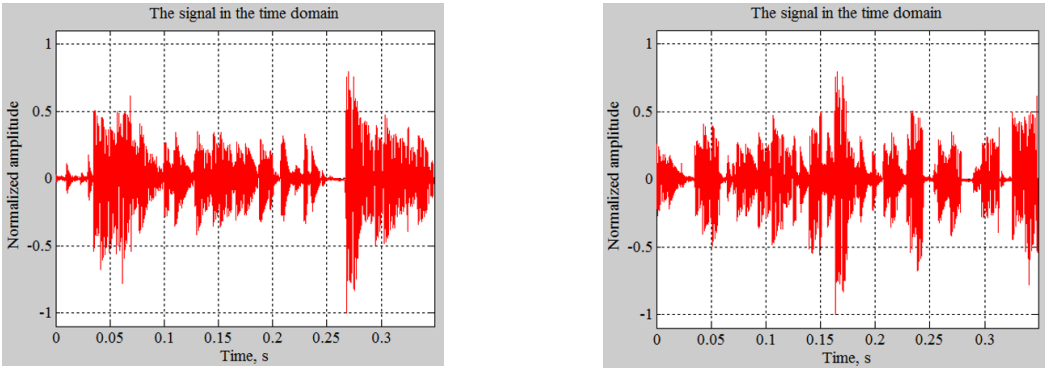


Fig. 9. The original sound of amplitude

In the first experiment, the encryption and decryption use a correct password of "SEMBILAN". The results show that the sound return to its original. The amplitude results of sounds of encryption and decryption can be seen in **Fig. 9** and **Fig. 10.(a)**. The next experiment, using the incorrect password "DELAPAN" the results show that the sound does not return to its original, even more randomized. The amplitude result can be seen in **Fig. 10. (b)**.



(a) (b)

Fig. 10. (a). The amplitude result of sounds encryption by using correct-password "SEMBILAN", and (b). The amplitude result of sounds encryption by using incorrect-password "DELAPAN"

IV. Results and Discussions

This system has a sound output to prove the success of the completed encryption. In this study, the MSE (Mean Square Error) is applied to prove the performance of the algorithm transposition test. The first experiment is done by comparing the value of the MSE between the original audio file and the encrypted audio file, then the original audio file and the decrypted audio file use the same password namely "SEMBILAN". The rule applied in this study is that if the encrypted file has a value [MSE $\neq$ 0], then it indicates an exchange of index position of the data audio WAV file. Meanwhile, if the file is decrypted result has the value [MSE = 0], then it indicates that the file is exactly the same as the original file, as well as voice. The results of the comparison WAV files as in **Table 3**.

The second test is comparing the value of MSE between the original file with encrypted file by using the correct password "SEMBILAN". Then, the original file with a file decrypted compared using the wrong password "DELAPAN" has been successfully carried out.

Table 3. A comparison of MSE values of audio by using correct password

File Name	File Size	Vector Size	MSE
cartoon008.wav	30,832 bytes	15,394 x 1	-
encrypt.wav	30,832 bytes	15,394 x 1	0.00000362
decrypt.wav	30,832 bytes	15,394 x 1	0

Table 4. A comparison of MSE values of audio by using incorrect password

File Name	File Size	Vector Size	MSE
cartoon008.wav	30,832 bytes	15,394 x 1	-
encrypt.wav	30,832 bytes	15,394 x 1	0.00000362
decrypt.wav	30,832 bytes	15,394 x 1	0.00000428

The results of the second test showed that the ratio of the MSE between the original file and the results of decryption does not have the same value of 0.00000428 or MSE $\neq$ 0. In other words, the arrangement of the data even more scrambled audio and sound never returned to the original.

V. Conclusions

The results of this study have shown that the transposition audio files through randomization algorithm can be used to secure the sound files. The original sound can be encrypted with various combinations by using a password, and the results of randomization sounds can be restored to the original sound by using the correct password. The sound will be randomized if the used password is incorrect. The use of passwords must be the user's attention because the results also showed that the use of different passwords but the same order will make the encryption will be easily solved. This could be weakness of the transposition method. Therefore, it is recommended that users always apply complicated passwords, such as long-character or mix-character password.

Acknowledgment

The writers would like to thank to Mulawarman University, Migent Software, and colleagues who have given support to complete this study.

References

[1] R. Gnanajeyaraman, K. Prasadh, and Ramar, "Audio encryption using higher dimensional chaotic map," *International Journal of Recent Trends in Engineering,* vol. 1, pp. 103-107, 2009.

[2] 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.

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

[4] T. Xiang, C. Yu, and F. Chen, "Secure MQ coder: An efficient way to protect JPEG 2000 images in wireless multimedia sensor networks," *Signal Processing: Image Communication,* vol. 29, pp. 1015–1027, 2014.

[5] 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.

[6] 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.
[7] A. M. S. Rahma and A. A. k. Maisaa, "To Modify the Partial Audio Cryptography for Haar Wavelet Transform by Using AES Algorithm," *Eng. & Tech. Journal*, vol. 32, pp. 170-182, 2013.
[8] A. G. Soriano-Sánchez, C. Posadas-Castillo, M. A. Platas-Garza, and D. A. Diaz-Romero, "Performance improvement of chaotic encryption via energy and frequency location criteria," *Mathematics and Computers in Simulation*, vol. 112, pp. 14–27, 2015.