Progressive Audio Scrambling in Compressed Domain

Wei-Qi Yan, Wei-Gang Fu, Mohan S. Kankanhalli
{e-mail:mohan@comp.nus.edu.sg}

***Abstract*— Audio scrambling can be employed to ensure confidentiality in audio distribution. We first describe scrambling for raw audio using the Discrete Wavelet Transform (DWT) first and then focus on MP3 audio scrambling. We perform scrambling based on a set of keys which allows for a set of audio outputs having different qualities. During descrambling, the number of keys provided and the number of rounds of descrambling performed will decide the audio output quality. We also perform scrambling by using multiple keys on the MP3 audio format. With a subset of keys, we can descramble to obtain a low quality audio. However, we can obtain the original quality audio by using all of the keys. Our experiments show that the proposed algorithms are effective, fast, simple to implement while providing flexible control over the progressive quality of the audio output. The security level provided by the scheme is sufficient for protecting MP3 music content.**

***Index Terms*— Audio Scrambling / Descrambling, Progressive Scrambling, MP3, Compressed Domain, Discrete Wavelet Transform.**

I. Introduction

Audio scrambling is an efficient approach towards audio copyright protection during data transfer. It can be used for ensuring confidentiality during distribution. Scrambling techniques have been employed in copyright protection of cable TV broadcast, secure image transfer from satellites to ground stations, and for military communication. Digital audio scrambling can also be employed for ensuring privacy in civilian communication over commercial and mobile telephones.

Audio scrambling aims to minimize the residual intelligibility of the original audio and limit the access to only authorized users. It is similar to but is not a direct application of normal cryptographic techniques. The primary reason is that audio media is an instance of a plain-text which has specific patterns of coherence that can be judiciously exploited. Audio scrambling works by breaking the coherence among the audio samples while adhering to the standard format. The main advantage of scrambling over encryption is that the scrambled audio can still be played by media players with the content being garbled. The challenges in audio scrambling are doing it progressively and that too in an efficient manner due to the high volume of data involved.

Theoretically, given an audio clip $C = \{c_1, c_2, \cdots, c_n\}$, there exists a scrambling function $S(\cdot)$, namely $\exists C' = S(C)$, where $C' = \{c'_1, c'_2, \cdots, c'_{n'}\}$, $c_i, c'_i \in \mathcal{Z}$, n', $n \in \mathcal{Z}^+$, the distance between two clips at any window length $0 < l \leq L = \min(n, n') < \infty$ is given by:

$$D_l(C, C') = \frac{1}{n-l} \sum_{i=1}^{n-l} (\min_{j=1}^{n'-l} (\frac{1}{l} \sum_{k=1}^{l} (|c_{i+k} - c'_{j+k}|))) \quad (1)$$

where L is the allowed maximum length of a window. The above equation illustrates the concept of scrambling. Since audio scrambling introduces confusion and diffusion at all resolutions, we sum the minimum distances together to obtain the degree of scrambling. Thus, if the original and the scrambled clips are different at every resolution, the target audio is considered to be completely scrambled. Audio scrambling only changes the sequence of the content data without altering the play duration of the original clip. Moreover, no additional information is injected into the clip.

Electronic commerce of audio products would be facilitated by the development of solutions which can ensure security/privacy, preferably operating in the compressed domain to reduce the bandwidth requirements. An additional flexibility of allowing for progressive audio quality control would be useful. Moreover, it should be computationally inexpensive to be used in real-time systems. Nowadays, digital goods can be easily copied and distributed, especially via the Internet. MP3 compressed audio (layer three of MPEG-1 and MPEG-2), which has a high compression ratio, is a typical example of such goods transacted over the Internet. MP3 is a popular Hi-Fi media format which carries

Wei-Qi Yan si with Queen's University Belfast; Wei-Gang Fu and Mohan S. Kankanhalli are with National University of Singapore. This paper was submitted to IEEE Transactions on Multimedia on Feb. 25th, 2006; revised on 1st Jul., 2007 and 1st Feb, 2008, respectively; accepted on 15 Mar. 2008. Corresponding author: Mohan S. Kankanhalli. {e-mail: mohan@comp.nus.edu.sg}

most of the popular and classical music of significant commercial value. Voice and speech can also be encoded in MP3. Normally, MP3 files uploaded onto a web site are protected by secret passwords. The legitimate users have to provide the passwords for data access. The potential problem is the music fans can share the password to download the products. Moreover, the full audio is available to the user once the password is provided. This makes the audio protection fragile.

In this paper, we have developed progressive audio scrambling and descrambling techniques on wavelet transform data and MP3 data [6]. Given an audio clip, we use the wavelet transform to decompose the audio signals to multiple levels, and progressively scramble/descramble the audio. Given a MP3 audio, we directly manipulate the coefficients in the compressed domain so as to progressively scramble/descramble the audio. In both instance, our motivation comes from the progressive mode of JPEG images on a web browser. If a subset of the data is descrambled, the user can obtain access to only a low-quality version of the original audio. Perfect reconstruction is possible only if the user has been fully authorized to utilize all of the keys. This provides tremendous flexibility to a music distributor. A low quality version can be sent to a potential buyer for preview which can then be progressively enhanced based on mutual agreement of commercial terms.

II. Related Work

A lot of research has been conducted in this area. Most of the existing algorithms are based on permutation of temporal segmentations or frequency sub-bands [2]. All of them aim at minimizing the perceptibility of the original audio and at controlling the access to only authorized users.

The Hadamard transform approaches are widely employed in audio scrambling [16] [18]. Milosevic et al [14] present an algorithm that makes use of Hadamard matrices to perform scrambling. It not only changes the sequence of the data elements, but also the values. Hence, it enhances security, but at the cost of increased complexity.

The wavelet transform tends to create low-value coefficients at the finer scale. Ma et al. [11] use wavelets to scramble analog signals in 2D space and it is highly secure in temporal and spatial domain. Kadambe [8] uses adaptive wavelets for speech scrambling. However, there has been no work on compressed domain for audio scrambling.

Borujeni [13] presents a fast Fourier transform (FFT) related algorithm to deal with the permutation of FFT coefficients in a speech encryption system, which is able to provide a high level of security. Matsunaga et al [2] present another FFT based algorithm which is accompanied by adaptive dummy spectrum insertion and commanding operation, and prove that the dummy spectrum insertion will further enhance the security.

Farkash et al. [5] present an algorithm that modifies the Gabor representation of the input speech signal. The proposed scheme can perform any invertible modifications, and it does not need the commonly used permutation operation.

In this paper, we present our work of audio scrambling in compressed domain. Our work is different from others in two aspects - it supports scrambling in the compressed domain and directly allows for progressive scrambling. With different keys, we can obtain the decoded audio clips having differing quality.

III. Our Contributions

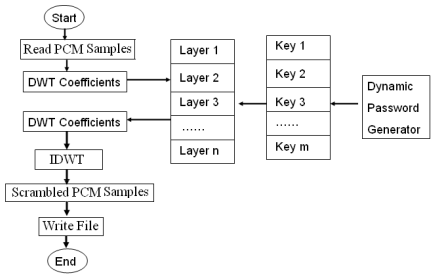
In this section, we describe our contributions regarding scrambling an audio clip by using Digital Wavelet Transform (DWT) first. We subsequently introduce the approach for MP3 audio scrambling. The details of DWT can be found in [4][12].

A. Scrambling

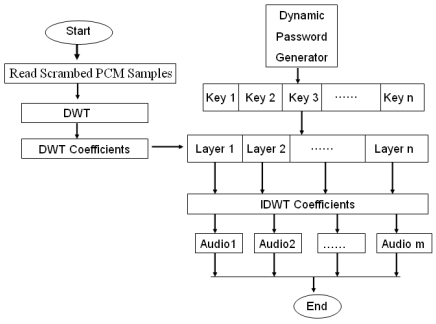
Figure 1 is the flowchart of audio scrambling and descrambling process based on the wavelet transform – Fig. 1(a) illustrates the scrambling operation while Fig. 1(b) is the descrambling operation. In Fig. 1(a), raw PCM data is read from an audio file, and DWT is applied to obtain the wavelet coefficients. We XOR the DWT coefficients with the random numbers (keys) generated by the dynamic password generator. The wavelet coefficients scrambled with these keys are employed to construct the scrambled audio clip by using IDWT (Inverse Discrete Wavelet Transform). A scrambled audio needs to satisfy the access requirement, therefore it can be played like a regular audio file. The scrambling of this audio is independent of any media player. Thus, the descrambling is also independent of any specific audio player. If no key is provided, the scrambled audio is totally jumbled up and will sound like noise. However, once the right keys are used, the audio will be reconstructed at the appropriate quality level which will be the highest when all of the keys are available.

These four aspects need to be considered in progressive audio scrambling:

- *Keys*. In progressive audio scrambling and descrambling, the various keys play a vital role. These keys are generated in an integrated manner by using the *dynamic password scheme* due to the need for



(a) Scrambling



(b) Descrambling

Fig. 1. Scrambling and Descrambling based on Discrete Wavelet Transform.

multiple decryptions at various layers. For each decryption, we generate a different key. These keys are managed by the authorized owners who have the audio copyright at the allowed quality. The various keys have different importance – those used in low frequencies after DWT transform will play a major role while others used in high frequency bands will have a lower impact on the quality of the reconstructed audio.

- *Audio data format*. The audio data to be scrambled normally are encoded in only one physical audio file. The data is organized according to the specific format. The scrambled and unscrambled data must adhere to the format so that it can be played.
- *Media player*. Both the scrambled and descrambled audio file should be playable by a suitable media player. Descrambling should be completed before playing and so the descrambling component must be separate from the media player.
- *Audio duration*. After scrambling, the duration of the scrambled data should not differ from the original audio length. This is because inverse transforms have a rigid length requirement. Suitable padding may occasionally be required.

B. Descrambling

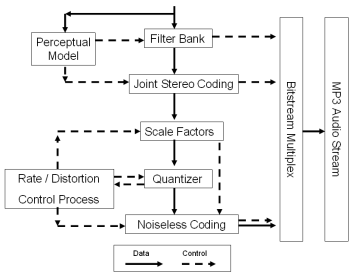
If we need to legitimately play a scrambled audio, we have to decrypt the audio data with an authorized key first. Consequently, we first decompose the scrambled audio data as wavelet coefficients at different layers by using DWT. Using the keys (used earlier for scrambling), we descramble (using the XOR operation) the wavelet coefficients of each layer. Therefore, we obtain a subset of the original unscrambled wavelet coefficients at a certain layer. Then, these wavelet coefficients at this layer and the scrambled coefficients at other scrambled layers are transformed by using IDWT to obtain the partially unscrambled audio. The more keys we have, the more original coefficients are restored and the better is the quality of the audio that we can reconstruct. If a user has all the keys, (s)he can restore the highest quality audio. This essentially allows for *progressive descrambling*. We illustrate this idea in Fig. 1(b). If we have n keys, we can reconstruct $\sum_{i=0}^{n} C_n^i = 2^n$ new descrambled audio clips with progressive qualities.

In the descrambling procedure, the user *ID* and the *keys* which a user has are input into the decryption module. From the keys and audio scrambling layer, we can find the layer number and the corresponding password. The true passwords are used to decrypt the scrambled audio [7]. On the other hand, if a user cannot provide a correct key for the decryption of a layer, that layer will not be decoded. Consequently, the scrambled wavelet coefficients at this layer will participate in the audio reconstruction. This will introduce noise into the reconstructed audio, which will affect the final audio quality.

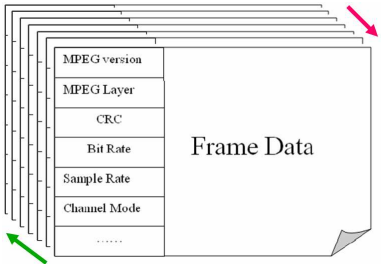
C. MP3 Audio Scrambling

For audio scrambling in compressed domain, we introduce MP3 scrambling in this section. MP3 generation is shown in Fig. 2(a) and its file structure depicted in Fig. 2(b). The MP3 file structure can linearly be expressed in this scheme (TAGs are optional, details are omitted):

[TAG v2] Frame1 Frame2 Frame3 $\cdots$**[TAG v1]**. A MP3 file is divided into frames and each frame has a constant time-length of 0.026 second. But the size of one frame (in bytes) varies according to the $bitrate$. The first four bytes of each frame are the frame header and the rest are audio data. The frame header consists of information about the frame, such as the $bitrate$, $SampleRate$ etc, and each of them can have its own



(a) Flowchart for MP3 Generation



(b) MP3 File Structure

Fig. 2. Basic information of MP3 Coding

characteristics. The MP3 specification (ISO -11172-3) does not specify on how the MP3 encoding shall be done. It only specifies the resultant output format. Developers are supposed to design their own algorithms to meet the requirements [17]. There are many different MP3 encoders available, with LAME being a popular one. These MP3 encoders perform lossy compression by discarding the information that does not significantly contribute to signal perception.

The wide-spread use of the MP3 format has necessitated compressed domain scrambling. Past research has been based on permutation in either the frequency or temporal domain [2]. However, not much work has been done directly in the compressed domain. MP3 is mainly used for online music distribution. There is a need for allowing the music owners to have flexible control over their music. This implies a need for an algorithm that allows them to provide music of various quality to different users and which is able to protect the copyright for the MP3 files simultaneously.

MP3 audio scrambling has the following typical application scenario for purchase of music CDs. Currently the purchasing process is most likely to be: A customer walks in; Listens to the sample CD; Picks up the CDs which (s)he likes; Approaches the cashier and makes the payment. While with the Internet business model, there is no physical sample CD available, the purchase process will become: A customer downloads sample audio clips; Performs an online transaction. The question is: if the customer can download the original audio, will he still actually purchase it?

Our progressive MP3 scrambling scheme provides a solution for this scenario. In order to make the customer purchase the audio, we can provide a poor quality version of the MP3 sample first; after the customer makes the payment, we issue him a password and allow him to reconstruct a better quality song. In fact, progressive scrambling allows for a tiered payment model for varying quality of reconstruction.

We propose a progressive algorithm that does multiple rounds of scrambling of MP3 audio. We can reconstruct the MP3 outputs of different qualities based on the number of keys provided and rounds of scrambling performed. This work is a part of our research program on media security. We had earlier developed a compressed domain video scrambling technique [9].

The key idea behind our contribution is to scramble the Huffman code words in the MP3 file. However, instead of directly shuffling the Huffman code word table, given an MP3 audio, we ignore the frame headers and only operate on the audio data. This is to ensure the elimination of repeats of values, i.e. one value in the original audio will be shuffled to different output values, which will break the statistical coherence between the source audio and the output. This is critical for thwarting statistical cryptanalysis which can defeat the scrambling scheme.

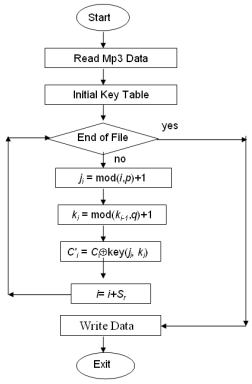


Fig. 3. Algorithm for MP3 Audio Scrambling

The procedure of MP3 audio scrambling/descrambling is explained in Fig. 3. Given a MP3 audio, we parse the frame headers and the compressed audio data from the MP3 file and scramble all of the audio data $C = \{c_i, i = 1, \cdots, n\}$, n is the length of the audio stream including the file header information. The scrambled data

are then written back as a MP3 file. A sampling rate s_r [3] needs to be predefined before each round of scrambling. In order to make the quality degradation linear, the sampling rate $s_r(l)$ needs to be set at an exponentially increasing rate for different rounds.

$$s_r(l) = a^l, a \in \mathcal{Z} \tag{2}$$

where l is the level, $l = 1, \cdots, L$; L is the total number of rounds of scrambling to be performed, and a is a constant. Also a key table K_T will be initialized; the format of K_T is shown as follows:

$$K_T = (key(i,j))_{p \times q} \tag{3}$$

The key table K_T generation can make use of special techniques to ensure the uniqueness and randomness of the key tables generated. For the keys used in this scheme, we employ the Arnold matrix [15] to generate the $p \times q$ matrix. Here we assign $p = q$. The Arnold matrix is shown as follows:

$$A = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 2 & \cdots & 2 \\ \cdots & \cdots & \cdots & \cdots \\ 1 & 2 & \cdots & p \end{pmatrix} \tag{4}$$

We first generate a random matrix $R = (r_{i,j})_{p \times q}$

$$r_{i,j} = rand(range), i = 1, 2, \cdots, p; j = 1, 2, \cdots, q \tag{5}$$

where $rand(\cdot)$ is a function to generate a random number sequence from a seed $range$. Then K_T is generated by doing a matrix product with the Arnold matrix [15] via:

$$K_T = A \cdot R \ mod \ p \tag{6}$$

We use the Arnold matrix to transform the keys in the key table to provide a different choice of keys for different users. The reason why we use the Arnold transform is that it enlarges the key space. The matrix after Arnold transform ensures the validity of the keys since they are bounded. Assuming $det(K_T) \neq 0$, the keys fully span the whole space.

For the i-th data item c_i in the original audio, we do a modulo division of the current position i with the row length p of the key table and the remainder decides the row number j of the key.

$$j_i = mod(i, p) + 1 \tag{7}$$

The column number k_i will cyclically range from the first column to the maximum column number q.

$$k_i = mod(k_{i-1}, q) + 1 \tag{8}$$

After we select the key $key(j_i, k_i)$, we perform an XOR with the MP3 audio datum c_i. XOR is chosen due to the speed and simplicity of implementation, since it is its own inverse. The changed datum c_i' is written into the corresponding position of the output stream:

$$c_i' = c_i \oplus key(j_i, k_i) \tag{9}$$

where $key(j_i, k_i)$ is the element in matrix K_T, $\oplus$ is the bit-wise XOR. This procedure is repeated till the end of the data stream and the output data stream is also an MP3 stream. The keys $K_l = \cup_{i=1}^{p=q} key(l, i)$ will be stored and distributed to the authorized consumers for descrambling.

The descrambling process is exactly similar to the scrambling procedure. For descrambling level l, an MP3 file generated from descrambling level $l-1$ will be used as an input. We apply the same process and the same K_T that has been used in the scrambling round l.

With the implementation of the scrambling process as described above, the quality of the output MP3 file depends on the number of levels of descrambling performed. For example, during scrambling if an MP3 audio has been scrambled 5 times using 5 different keys, a user needs to have all the 5 keys and perform 5 rounds of descrambling before they can perfectly restore the original audio.

The advantage of this technique for MP3 scrambling is that the descrambling is performed on the previous output. Given an MP3 audio with quality level n, we can get the audio file with a better quality level $n-1$ by descrambling this audio further with the key for level $n-1$. The desired paradigm of *one scrambling yet multiple descramblings* is achieved with this design.

D. Dynamic Passwords

In this section, we provide the scheme of generating dynamic passwords. The reason for advocating the use of dynamic passwords is because it is not easily replicable and is robust against transmission attacks. We have previously used dynamic passwords for image watermarking [1]. The dynamic password scheme was described by Harn [7], which integrates the concept of public key cryptography with the original dynamic password scheme [10]. We use a similar scheme.

Suppose we have two large primes p_1 and p_2, we can get $n = p_1 \cdot p_2$, calculate m-th password PW_m by using the following equation:

$$PW_m^{2^m} = ID_i \ mod \ n \tag{10}$$

where ID_i is related to the ID of i-th user. For this user, in the r-th logging on, the m-th dynamic password will be:

$$PW_r = (PW_m)^{2^{(m-r)}} \ mod \ n \tag{11}$$

The security of dynamic password scheme depends on the difficulty of factoring n into two large prime numbers p_1 and p_2. Since the user's password is changed dynamically, it is impossible for an eavesdropper to tamper with the communication and to impersonate the user. Now we employ the dynamic passwords as the keys for the audio scrambling and descrambling.

When the dynamic password scheme is employed in audio scrambling, passwords are used as keys for encryption as well as for decryption. For a same audio clip to be encrypted multiple number of times, the advantage of dynamic passwords will be apparent, since dynamic passwords are perfectly secure for repeated encryptions.

In audio scrambling, the keys are put into one string, as shown in Table I. Each password occupies 16 bits (2 bytes). All the passwords for different wavelet layers are located in $16 \times L$ bits or $2 \times L$ bytes string. If a user has not authorized some keys, the positions of these passwords can be filled up with special code such as all zeros. The layer corresponding to this key will not participate in the decryption process.

TABLE I

THE LAYOUT OF KEYS IN THE KEY STRING

KEY_1	KEY_2	0	0	0	$\cdots$	$\cdots$	KEY_n

E. Performance Evaluation

Normally, the audio quality is measured by PSNR (Peak Signal Noise Ratio) which computes the square root of mean value of corresponding elements between two audio streams:

$$PSNR(C, C') = 20 \cdot log_{10} \frac{\max_x(|c(x) - c'(x)|)}{\sqrt{\frac{1}{m}\sum_{x=1}^{m}(c(x) - c'(x))^2}} \quad (12)$$

where $c(x)$ and $c'(x)$, $0 < x \leq m \in \mathcal{Z}^+$ are the original audio data and the audio data with noise respectively, m is the length of the two audio clips, $\max(\cdot)$ is the extreme value in the domain. The disadvantage of this metric is that it is very sensitive to positional changes. Furthermore, we would like to measure the difference between two audio clips at multiple resolutions by using wavelet coefficients. The scrambling degree also reflects the varying differences among the audio clips. We sum all the differences in the audio layers and provide the multi-resolution comparison.

$$MRC(C, C') = MRC_a(C, C') + MRC_d(C, C') \quad (13)$$

$$MRC_a(C, C') = \sum_{j=0}^{Len \times 2^{(-l)}} (a_l(j) - a'_l(j))^2 \quad (14)$$

$$MRC_d(C, C') = \sum_{i=0}^{l} \sum_{j=0}^{Len \times 2^{(-i)}} (d_i(j) - d'_i(j))^2 \quad (15)$$

where the audio clips after wavelet decomposition are $C = (d_1, d_2, \cdots, d_l, a_l)C$, $C' = (d'_1, d'_2, \cdots, d'_l, a'_l)C'$, l is the layer number, d_i, d'_i, a_i, and a'_i are the coarse and fine wavelet coefficients at level i respectively, $i = 1, 2, \cdots, l$; Len is the length of audio clip. The advantage of wavelet-based multiresolution comparison is that a minor difference can also be captured [12].

IV. EXPERIMENTAL RESULTS

In our experiments, we take voice and music data as our test sets. We scramble the audio data by using the wavelet transform first, then we scramble MP3 audio clips. All the results are implemented using the MATLAB 6.0 platform.

A. Dynamic Passwords

First, we describe a group of dynamic passwords, which is based on Eq. (11). We assign different ID for each wavelet layer, they are (d_1, d_2, d_3, a_3), the ID we assign to these layers is $(26, 25, 24, 23)$ as shown in Table II, when we need to encrypt the wavelet coefficients at different times, we use different passwords at each instance.

TABLE II

DYNAMIC PASSWORDS FOR THREE LAYERS COEFFICIENTS OF WAVELET DECOMPOSITION

Encryption Times	a_3 ID=23	d_3 ID=24	d_2 ID=25	d_1 ID=26
1	3318	2842	5	499
2	145	1343	3904	3044
3	7009	6467	3315	5780
4	2315	9144	176	2789
5	1745	2515	4512	3464
6	8849	3371	7906	1374
7	490	625	7208	2024
8	666	25	8086	1370
9	3528	5	2342	5028
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$

B. DWT based audio scrambling

With these keys, we scramble the wavelet coefficients of each layer. Table III shows the results for each layer by calculating the multiresolution distance using eq.(13).

TABLE III

SCRAMBLING DEGREE FOR AUDIO CLIPS WITH 'DB1' WAVELET

Wave Name	$d1 \cdot C$	$d2 \cdot C$	$a2 \cdot C$
yahoo.wav*	$1.24 \cdot 10^{-33}$	0.0394	0.1186
notify.wav*	$6.72 \cdot 10^{-33}$	$7.9 \cdot 10^{-4}$	0.0128
tada.wav*	$5.5 \cdot 10^{-33}$	0.0017	0.017
ringout.wav*	$1.3 \cdot 10^{-33}$	0.0171	0.0907

The test audio clips in Table III with "$\star$" have been selected from the media gallery of Microsoft Windows XP, while the audio clip with "$*$" has been selected from Yahoo Messenger. Using the earlier described keys without repetition, we reconstruct the audio clips. In Table IV, we provide a progressive comparison at different layers with different keys.

TABLE IV

THE COMPARISON OF MULTIPLE LEVEL DESCRAMBLING

Wavelets	$d_1 \cdot C$	$d_2 \cdot C$	$a_2 \cdot C$	$d_1 \cdot C$ $\&d_2 \cdot C$
Daubechies	.0432	.0340	.0092	.0340
Harr	.0432	.0340	.0092	.0340
Simlet	.0432	.0340	.0092	.0340
Coiflets	.0451	.0380	.0071	.0380
Biorthogonal	.0432	.0340	.0092	.0340
Meyer	.046	.0405	.0055	.0405
Wavelets	$d_1 \cdot C$ $\&a_2 \cdot C$	$d_2 \cdot C$ $\&a_2 \cdot C$	$d_1 \cdot C \& d_2 \cdot C$ $\&a_2 \cdot C$	$d_i, a_i = 0$
Daubechies	.0092	1.24×10^{-33}	1.24×10^{-33}	.0432
Harr	.0092	1.24×10^{-33}	1.24×10^{-33}	.0432
Simlet	.0092	1.24×10^{-33}	1.24×10^{-33}	.0432
Coiflets	.0071	5.36×10^{-26}	5.36×10^{-26}	.0451
Biorthogonal	.0092	1.24×10^{-33}	1.24×10^{-33}	.0432
Meyer	.0055	4.33×10^{-7}	4.33×10^{-7}	.046

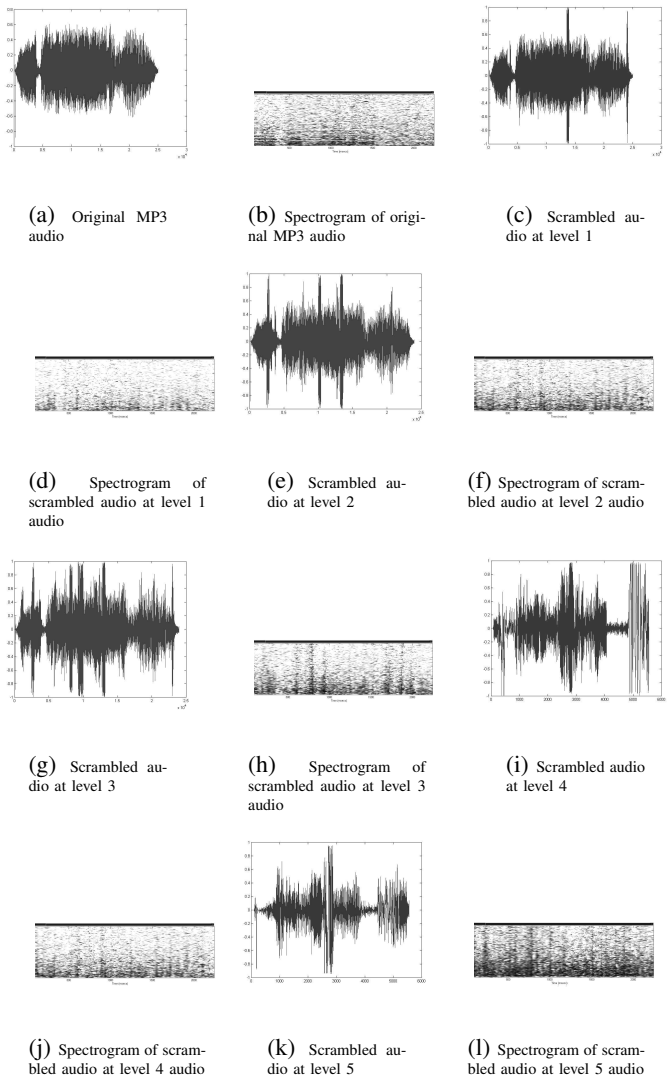
From Table IV, we observe that the d_1C layer plays a vital role in audio reconstruction. No matter which layer it combines with, the result appears to be of better quality than the others. From Table IV, we see that the scrambling is independent of the particular wavelet basis. Utilizing different wavelet bases, we obtain similar results.

C. MP3 audio scrambling

Figure 4 depicts the waveform for an audio clip at different scrambling levels. Fig. 4(a) is the waveform of the source MP3 audio, we scramble the audio 5 times and obtain Fig. 4(k). We descramble Fig. 4(k) to get Fig. 4(i), which is of slightly better quality. We generate Fig. 4(g) to Fig.4(c) by performing additional rounds of descrambling. The waveforms show that the output audio clips are gradually reconstructed and eventually we get the audio that is exactly same as the original. Subjective evaluation is a reasonable way to measure the scrambling levels. In our work of MP3 audio scrambling, we conduct an evaluation of 4 audio clips belonging to different categories, namely classical, pop, jazz and rock music. Each music clip has been scrambled to 5 levels of degraded qualities. Seven subjects provided feedback and the results are in Table V by using Eq. (16).

$$AV_{i,j} = \frac{1}{N_E} \sum_{k=1}^{N_E} SV_{i,j,k} \tag{16}$$

where N_E is the number of evaluators, $SV_{i,j,k}$ is the subjective value for the k-th evaluator for the item in the i-th row and j-th column as well as $SV_{i,j,k}$ in Table V.



(a) Original MP3 audio (b) Spectrogram of original MP3 audio (c) Scrambled audio at level 1 (d) Spectrogram of scrambled audio at level 1 audio (e) Scrambled audio at level 2 (f) Spectrogram of scrambled audio at level 2 audio (g) Scrambled audio at level 3 (h) Spectrogram of scrambled audio at level 3 audio (i) Scrambled audio at level 4 (j) Spectrogram of scrambled audio at level 4 audio (k) Scrambled audio at level 5 (l) Spectrogram of scrambled audio at level 5 audio

Fig. 4. Original, scrambled MP3 audio waveforms and their spectrograms at different scrambled levels

Table V shows that most of the listeners are able to perceive the gradual decrease in quality as the amount of scrambling increases. They can quite accurately identify the correct sequence of the scrambled audio, especially in the better quality levels. Most of the evaluators are able to rank the scrambling levels correctly. We also measured the computation time required which is shown in Table VI. The result in Table VI shows that our contribution is practical for Internet applications. For an MP3 audio of 30 seconds duration, it needs only 8.7 seconds (in MATLAB) for the scrambling and descrambling processes, which is much shorter than the music duration. Thus the proposed algorithm can potentially be used in real-time systems.

In our scheme, the security of our scheme relies on the key table. Range of values allowed in the key table determines the key space. The amount of calculation for

TABLE V

AUDIENCES' SCORE FOR DIFFERENT QUALITY LEVELS (1 FOR THE BEST QUALITY - 6 FOR THE WORST QUALITY)

Quality	POP	JAZZ	ROCK	CLASSICAL
Level 6	6.00	6.00	6.00	6.00
Level 5	4.14	4.57	5.00	5.00
Level 4	3.86	3.71	3.29	3.86
Level 3	3.86	3.00	3.00	2.86
Level 2	1.71	2.29	2.14	1.71
Level 1	1.29	1.57	1.57	1.57

TABLE VI

TIME CONSUMPTION FOR MP3 SCRAMBLING / DESCRAMBLING

Audio	Operations	POP	JAZZ	ROCK	CLASSICAL
		3KB/2s	23KB/22s	10KB/9s	32KB/32s
Level 1	Scrambling	0.82s	0.91s	0.84s	0.94s
	Descrambling	0.81s	0.90s	0.85s	0.94s
Level 2	Descrambling	0.79s	0.97s	0.84s	1.04s
Level 3	Descrambling	0.84s	1.04s	0.91s	1.15s
Level 4	Descrambling	0.87s	1.23s	0.99s	1.41s
Level 5	Descrambling	1.07s	3.07s	1.77s	4.1s

attacker to break the whole MP3 file of length L by brute force is calculated as following: given an MP3 audio of duration 30 seconds and data length of 3.2×10^4 bytes, assume that the key table allows random values within the range 1 to 16 and the key table is of size 16×16. This will take more than 10^{250} years at a rate of 10^{12} instructions per second. However, this can potentially be reduced by studying audio coherence for domain-specific cryptanalysis. Since the purpose of this work is to protect music clips, the security provided appears to be sufficient. Incidentally, this scheme has no size blow-up. The original and scrambled audio clips are of exactly the same size.

V. CONCLUSION

In this paper, we describe two schemes for progressive audio scrambling which are structurally similar. The first scheme works on raw audio while the second scheme works on MP3 audio directly on the compressed domain. We decompose the raw audio data using DWT, then the pre-generated keys from dynamic passwords are embedded into the wavelet coefficients; if all the keys among these layers can be provided, the audio would be fully restored, or else only partial descrambling of this audio can be achieved. Dynamic passwords ensure that different users at different time will have different keys. The contributions of this paper can be employed in audio authentication and copyright protection. It is fast and simple yet it can provide sufficient security for music e-commerce applications. Our future research will be on using secret sharing in audio scrambling and on enabling traitor tracing.

REFERENCES

[1] Monica Bansal, Wei-Qi Yan, and Mohan S. Kankanhalli. Dynamic watermarking for images. In *Proc. of IEEE PCM*, Singapore, Dec. 2003.

[2] S.E. Borujeni. Speech encryption based on fast fourier transform permutation. In *Proc. of The 7th IEEE International Conference on Electronics, Circuits and Systems*, volume 1, pages 290–293, Dec. 2000.

[3] William G. Cochran. *Sampling techniques(3rd Edition)*. John Wiley, New York, USA, 1977.

[4] I. Daubechies. Ten lectures on wavelets. 1992.

[5] S. Farkash, S. Raz, and D. Malah. Analog speech scrambling via the gabor representation. In *Proc. of 17th Convention of Electrical and Electronics Engineers*, pages 365–368, Israel, Mar. 1991.

[6] Wei-Gang Fu, Wei-Qi Yan, and Mohan S. Kankanhalli. Progressive scrambling for MP3 audio. In *Proc. of IEEE ISCAS'05*, pages 5525–5528, Kobe, Japan, May 2005.

[7] L. Harn. A public-key based dynamic password scheme. In *Proc. of Symposium on Applied Computing*, pages 430–435, 1991.

[8] S. Kadambe and P. Srinivasan. Application of adaptive wavelets for speech coding. In *Proc. of the IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis*, pages 632–635, Oct. 1994.

[9] M.S. Kankanhalli and T.T. Guan. Compressed domain scrambler and descrambler for digital videos. *IEEE Transactions on Consumer Electronics*, 48(2):356–365, May 2002.

[10] L. Lamport. Password authentication with insecure communication. *Communications of the ACM*, 24(11):770–774, Nov. 1981.

[11] F.L. Ma, J. Cheng, and Y.M. Wang. Wavelet transform based analogue speech scrambling scheme. *Electronics Letters*, 32(8):719–721, Apr. 1996.

[12] S.G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, Jul. 1989.

[13] A. Matsunaga, K. Koga, and M. Ohkawa. An analog speech scrambling system using the FFT technique with high level security. *IEEE Transactions on Selected Areas in Communications*, 7(4):540–547, May 1989.

[14] V. Milosevic, V. Delic, and V. Senk. Hadamard transform application in speech scrambling. In *Proc. of 13th International Conference on Digital Signal Processing*, pages 361–364, 1997.

[15] Dong-Xu Qi, Jian-Cheng Zou, and Xiao-Yu Han. A new class of scrambling transformation and its application in the image information covering. *Science in China (series E)*, 43(3):305–311, Jun. 2000.

[16] V. Senk, V.D. Delic, and V.S. Milosevic. A new speech scrambling concept based on Hadamard matrices. 4(6):161–163, Jul. 1997.

[17] Y. Wang, W.D. Huang, and K. Jari. A framework for robust and scalable audio streaming. In *Proc. of ACM Multimedia'04*, New York City,NY,USA, Oct. 2004.

[18] Y. Wu and B.P. Ng. Speech scrambling with hardamard transform in frequency domain. In *Proc. of IEEE ICSP*, pages 1560–1563, 2002.