Hindawi
Advances in Multimedia
Volume 2017, Article ID 5879257, 10 pages
https://doi.org/10.1155/2017/5879257

Research Article

Commutative Watermarking-Encryption of Audio Data with Minimum Knowledge Verification

Roland Schmitz and Jan Gruber

Stuttgart Media University, Nobelstrasse 10, 70569 Stuttgart, Germany

Correspondence should be addressed to Roland Schmitz; schmitz@hdm-stuttgart.de

Received 30 September 2016; Accepted 20 February 2017; Published 20 March 2017

Academic Editor: Akram M. Z. M. Khedher

Copyright © 2017 Roland Schmitz and Jan Gruber. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

We present a scheme for commutative watermarking-encryption (CWE) of audio data and demonstrate its robustness against an important class of attacks, Time-Scale Modifications (TSM). In addition, we show how the proposed CWE scheme can be integrated into a cryptographic protocol enabling public verification of the embedded mark without disclosing the mark or the watermarking key used for embedding.

1. Introduction

Commutative watermarking-encryption (CWE) means the combination of watermarking and encryption in such a way that the encryption and watermarking functions commute; that is,

$$\mathcal{M}_{W_K}\left(\mathcal{E}_K(O), m\right) = \mathcal{E}_K\left(\mathcal{M}_{W_K}(O, m)\right), \tag{1}$$

where $\mathcal{E}$ is the encryption function, K is the encryption key, W_K is the watermarking key, O is the cleartext media data, and m is the mark to be embedded.

If encryption and watermarking do commute, their combination can serve as an important building block within a Digital Rights Management (DRM) System, as detailed further in Section 2. In the present paper, an existing CWE concept for still images [1] is extended to audio files. To the best of our knowledge, this is the first CWE scheme for audio files to appear in the literature. In addition, we show that the presented CWE scheme can be integrated into a modified version of a protocol due to Craver and Katzenbeisser [2], enabling zero-knowledge verification of the watermark, meaning a verifier can verify the presence of a watermark without disclosure of the mark $\mathcal{M}$ or the watermarking key W_K. The rest of the paper is organized as follows: in Section 2, we motivate the need for CWE schemes and identify some basic requirements. In Section 3, we shortly review existing CWE schemes for still images and encryption/watermarking techniques for audio files, with a special emphasis on those algorithms using similar techniques as in our approach. In Section 4, we present our CWE scheme in detail. Section 5 provides experimental results on the robustness and fidelity of the watermarking part. Section 6 presents the integration of the CWE scheme into a zero-knowledge protocol for verifying the mark, and Section 7 concludes the paper.

2. Motivation for CWE

The concept of commutative watermarking-encryption (CWE) was first discussed in [3] with a special emphasis on watermarking in the encrypted domain. From the left-hand side of (1) it is clear that the watermarking function $\mathcal{M}$ must be able to act in the encrypted domain, which means that only a limited set of audiovisual features (if any) is available to the embedder and can be used to embed the mark.

2.1. Dispute Resolve Protocols. The prime motivation to look at CWE schemes originates from the need to implement so-called Dispute Resolve Protocols, where a rights owner R provides a digital media object O to a distributor D, who in turn sells O to some customer C. In this scenario, a number of attacks are possible, most importantly the case where C sells a copy of O in his own right. In particular, if such a

copy is detected, the Dispute Resolve Protocol must be able to identify R as the rightful owner of O and to identify C as the offending party.

An obvious solution is that R embeds a watermark identifying R as the rightful owner into O and provides the marked object $\overline{O}$ to D. The distributor D in turn marks $\overline{O}$ for each customer C with an additional watermark uniquely identifying C. Unfortunately, in this scenario the distributor D is able to generate n identical copies of $\overline{O}$ and sell them to n customers C_i. If these copies are marked with the identifier of some specific customer C, the distributor D can repudiate having generated the copies and the customer C could be held responsible for the offence of D.

The basic problem here is that D has access to the marked object $\overline{O}$ in plaintext. If a CWE scheme is available, however, the following protocol between a generic seller S and a generic buyer B becomes possible, as proposed in [4]:

(1) S encrypts O with her symmetric key K_S. The result is $C = \mathscr{E}_{K_S}(O)$.

(2) S sends C to B, together with an individual mark m_B that B is to embed into C.

(3) B embeds m_B into C and encrypts the result with his own key K_B. The final result $U = \mathscr{E}_{K_B}(\mathscr{M}(C, m_B))$ is sent to S.

(4) S verifies that U contains m_B as watermark. If the verification is successful, S removes her own encryption and sends the result $V = \mathscr{E}_{K_B}(\mathscr{M}(O, m_B))$ back to the buyer.

(5) B removes his encryption from V and is in possession of the individually marked object $\overline{O} = (\mathscr{M}(O, m_B))$.

If the distributor D takes the role of the seller in this protocol and the rightsholder R performs the en- and decryption operations in steps (1) and (4), respectively, the problem mentioned above can be solved, if a CWE scheme for the media object O is available. The need for a CWE scheme becomes obvious in steps (3) and (4), where an encrypted media object is watermarked and the presence of a watermark is verified in an encrypted object, respectively. Moreover, steps (3) and (4) call for a public key watermarking scheme, where there is a private embedding key and a public detection key, or an asymmetric scheme, where it is possible to verify the existence of a watermark without fully disclosing the embedding key or the watermark itself.

2.2. DRM Systems. In Digital Rights Management (DRM) Systems [5], encryption and watermarking are often combined in a natural way: the media data are transferred to a compliant media player in encrypted form, so that access to the plaintext data happens only under control of the compliant player. In addition, watermarks are embedded into the media data to have an additional layer of protection which is present even after the data have been decrypted. These watermarks can be used to claim copyright, enforce copying restrictions, or track illegal copies offered on the Internet. If a CWE scheme is used, compliant media players have the opportunity to detect and insert watermarks even in encrypted data. More generally, it should be possible to protect multimedia data throughout the distribution chain in a flexible way by allowing the encryption and watermarking operations to commute [6].

2.3. Searching in Encrypted Databases. With the advent of cloud computing, new security challenges have arisen. For example, cloud computing clients need to secure their data, not only to protect their data from public attacks, but also to protect their data from their cloud service provider [7]. Thus, clients need to encrypt their data in the cloud. On the other hand, a cloud service provider or a client often has the need to search through the client data according to certain metadata or tags. It is therefore highly desirable to provide techniques which can protect the clients' privacy and offer a large amount of accessibility at the same time. CWE schemes can provide such a solution, if metadata are used as watermarks and embedded into the encrypted data.

3. Related Work

3.1. CWE Schemes for Image Data. To the best of our knowledge, no CWE schemes for audio data have been proposed so far. However, there have been a number of attempts aimed at still images, of which we only review the so-called *invariant encryption* approach, as it is also used in our audio CWE scheme. For a more comprehensive review of existing CWE schemes for still images, see [8].

The invariant encryption approach to CWE as introduced in [1] is to encrypt the media data completely (as opposed to the partial encryption approach, which leaves part of the data unencrypted to host the watermark), but to use a weaker cipher that leaves a feature space of the media data invariant. This invariant feature space can be used to embed a watermark. For example, a permutation cipher can be used for encryption, leaving the global first-order statistics of the image untouched. The invariant feature space is therefore represented by the image histogram and a histogram-based algorithm can be used to embed the mark. The advantage of the invariant encryption approach is that all media data are encrypted (and not just a subset as in partial encryption schemes). The disadvantage, besides using a weaker cipher, is an inherent lack of robustness of the watermark.

3.2. Audio Ciphering Methods. Let the audio signal consist of a set of N sample values $F = \{f(i) \mid 0 \leq i \leq N - 1\}$. Most existing audio ciphering methods like, for example, [9] or [10], substitute the audio sample values and change (i.e., flatten) the global histogram of the amplitudes of the sample values. The flattening of the histogram makes it impossible to use the histogram for embedding a watermark. In [11], however, a permutation cipher is used to permute the sample values in the time domain, thereby keeping the histogram invariant. This shows that it is possible to transfer the invariant encryption approach to audio data.

3.3. Audio Watermarking Methods. From the host of existing audio watermarking methods (see [12] for an overview), the

method by Xiang et al. [13] is the most important for our work, as it uses (a part of) the amplitude histogram for embedding the mark. The range of the audio sample values F is splitted into equal-sized bins. The amplitude histogram H is an L-dimensional vector

$$H = \{h(i) \mid 0 \leq i \leq L-1\}, \tag{2}$$

where $h(i)$ denotes the number of samples falling into the ith bin. The relevant part of the amplitude histogram consists of the bins covering the interval $B = [-\lambda\overline{A}, \lambda\overline{A}]$, where $\overline{A}$ is the mean value of the absolute amplitude values and λ is some fixed parameter. This condition makes sure that the bins in the relevant part of the histogram are "well filled," that is, $h(i) \gg L \forall i$. To embed a watermarking bit w_i, a triple of consecutive histogram bins with heights (a, b, c) is used. If $w_i = 1$, the relation $2b/(a+c) < T$ should hold, where T is a predefined threshold value. If the relation is not satisfied by the three bins, a certain number of samples is shifted from the first and third bin of the triple into the second bin by adding and subtracting, respectively, a bin width M to the samples. An analogous process is carried out if $w_i = 0$.

As embedding the mark has altered the mean value of the amplitude values, for extraction, the correct mean value has to be searched within a search space S. For each mean value in S, the corresponding histogram is formed and the distance between the first n extracted bits and a known synchronization sequence sync is computed. The mean value associated with the minimum distance is used to extract the remaining watermark bits.

The described synchronization process helps to make the watermark robust against TSM attacks (cf. Section 5.2). Although the watermarking scheme is based on the histogram, it cannot be used in conjunction with a permutation cipher to form CWE scheme, because only a certain number of sample values in a histogram bin are modified. Therefore, after application of the permutation cipher, different sample values than before are modified, which destroys the commutativity property. Moreover, the scheme by Xiang et al. does not use a secret watermarking key W_K.

4. The Proposed CWE Scheme

The proposed scheme is based on the earlier ideas [1, 13] described in Section 3. In order to apply them in the audio domain and in order to make the overall scheme more robust to TSM attacks, some modifications were necessary, which are described in the following paragraphs.

4.1. Ciphering Algorithm. An analogue audio signal is transferred into the digital domain by sampling the time-continuous signal at a certain discrete sampling rate. At the same time, the obtained samples are quantized according to the bit depth available, the result being a set of N sample values $F = \{f(i) \mid 0 \leq i \leq N-1\}$, where i can be seen as a discrete time coordinate. Common bit depths for representing audio are 16, 20, or 24 bit. The general idea is to permute the discrete points in time, while leaving the sample values untouched. In order to generate the permutations, the discrete version of Arnolds Cat Map [14] was used, because it is a well-known chaotic map used by many authors for generating permutations in image ciphering (see e.g., [15]). The discrete Cat Map is a two-dimensional map defined on a $H \times H$ square grid by

$$\begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} 1 & a \\ b & ab+1 \end{pmatrix} \cdot \begin{pmatrix} x_k \\ y_k \end{pmatrix} \bmod H, \tag{3}$$

where a and b are parameters that can serve as the secret key if the function is used for encryption purposes. Two-dimensional permutations of the square grid can be quickly generated by repeated application of the Cat Map. Note, however, that there are only H^2 different keys. Therefore, it has been proposed in [16] to change the secret parameters in every iteration of the Cat Map. In order to apply the Cat Map on a discrete audio signal of length N, the audio signal is rearranged into a square grid of size $\sqrt{N}$. If N is not a square number, the signal is padded with random sample values having the same probability distribution (i.e., the same histogram) as the original signal. This makes sure that the padded values cannot be distinguished from the original values by an attacker. Moreover, the original histogram is largely unchanged by the padding (cf. Figure 1). Figure 2 shows the effect of the Cat Map after five iterations on the waveform of an example signal. The resulting PSNR between original and enciphered signal is 16.47.

4.2. Watermarking Algorithm

4.2.1. Basic Principles. The design goals for the watermarking algorithm to be used within our proposed CWE scheme were as follows:

(i) The watermarking algorithm should commute with the permutation cipher in the sense of (1).

(ii) It should be robust against Time-Scale Modification (TSM) attacks (see Section 5.2).

(iii) It should be able to use a long watermarking key in order to prevent an attacker to insert her own watermark.

These goals call for a combination of the watermarking concepts described in [1, 13]: in order to have full commutativity with the permutation cipher, it is necessary to swap entire histogram bins. These swaps can be randomized using a secret watermarking key as described in [1]. However, this procedure can imply a substantial change of the histogram mean. In order to deploy a synchronization procedure for robustness against TSM attacks as described in [13], the original mean $\overline{A}$ needs to be transferred separately to the verifier. Moreover, as a TSM attack may change the height relation of two histogram bins $h(x), h(y)$ if only bins which are sufficiently different are selected for swapping, that is, $|h(x) - h(y)| > T$ should hold, where T is a strength parameter governing the robustness of the scheme. Note that this relation is unchanged if the two histogram bins are

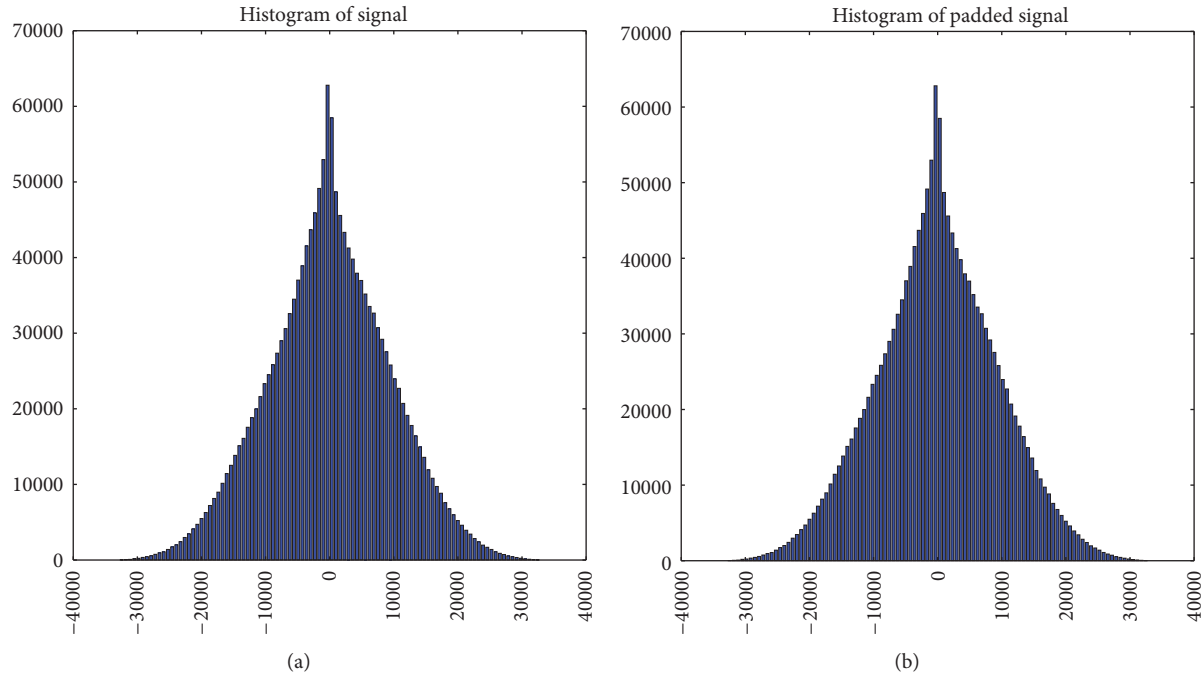


Figure 1: Comparison of signal histograms. (a) Before padding; (b) after padding.

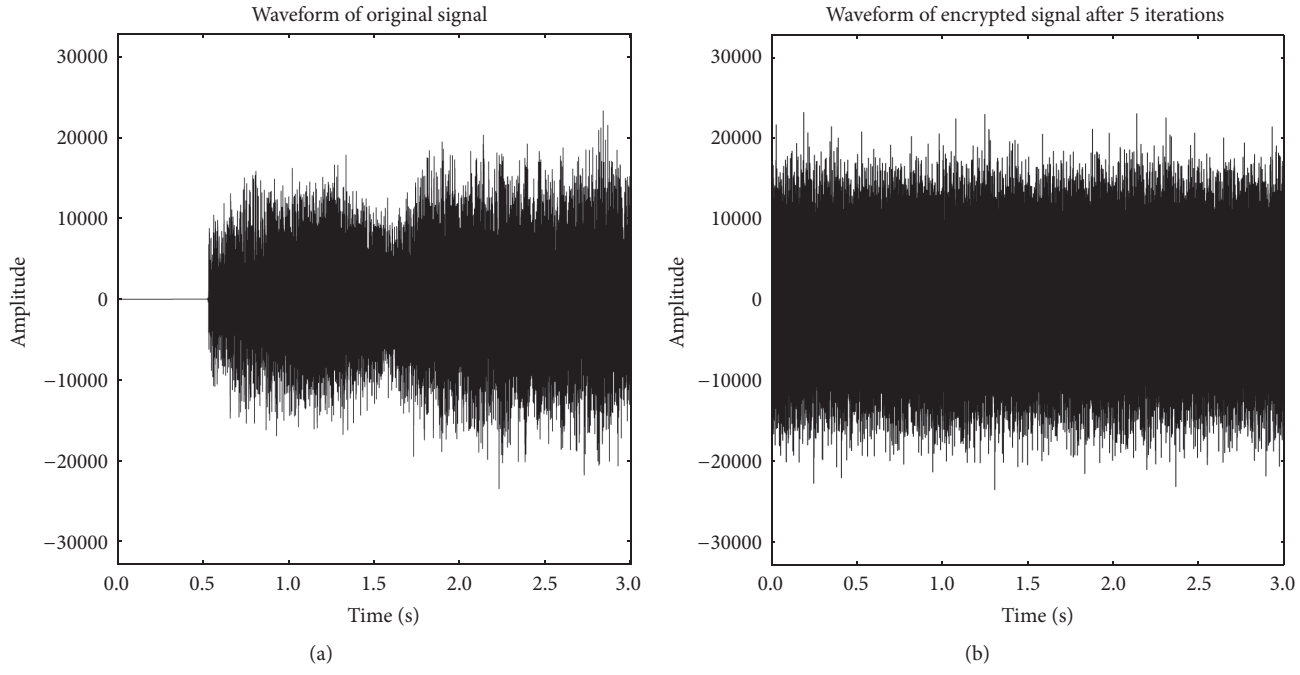


Figure 2: Comparison of the first three seconds of signal waveforms. (a) Original signal; (b) encrypted signal.

swapped. However, it might be changed by a TSM attack, which could lead to the detector choosing the wrong bin pairs for extracting the mark. Therefore, the bin pairs used to embed the mark now need to serve as watermarking key, as opposed to an initial seed for a pseudorandom number generator as in [1].

4.2.2. Algorithm Details

Embedding. Let the watermark W consist of L_W bits w_i. The first s bits of the watermark are used as synchronization sequence sync and should be known to the detector. As a first step, the embedder generates the amplitude histogram H of

the audio signal and forms its relevant part $B = [-\lambda\overline{A}, \lambda\overline{A}]$ as described in Section 3.3. For each $1 \le i \le L_W$ the embedder computes a histogram bin pair in the following way:

(i) Generate a random number $1 \le x_i \le L_B$, where L_B is the number of bins within B.

(ii) Find the x_ith unused bin within B. Generate another random number step, such that $1 \le \text{step} \le \text{step}_{\max}$ and $0 \le y_i = x_i + \text{step} \le L_B$.

(iii) If y_i has not been used before and $|h(x_i) - h(y_i)| > T$, save the pair (x_i, y_i). The watermarking bit w_i is now embedded in the following way:

(a) For $w_i = 1$, the relation $h(x) > h(y)$ must hold. If this is not the case, swap the bins by assigning new values to all samples in the bins.

(b) For $w_i = 0$, the relation $h(x) < h(y)$ must hold. If this is not the case, swap the bins by assigning new values to all samples in the bins.

(iv) If y_i has been used before or $|h(x_i) - h(y_i)| \le T$ generate a new random number x_i.

Detection. The detector needs to know the original mean value $\overline{A}$ of the samples of the unmarked cover work O, along with the synchronization sequence sync of length s and the sequence of bin pairs $W_K = \{(x_i, y_i) \mid 1 \le i \le L_W\}$ used for embedding the mark W, which serves as a watermarking key.

As described in [13], for finding the correct mean value after a potential TSM attack, the detector first computes a search space $S = \{\overline{A}(1-\Delta), \overline{A}(1-\Delta)+1/\lambda, \ldots, \overline{A}(1+\Delta)\}$, where λ is a parameter governing the size of the search space. Now, for each member $\overline{A_k}$ of the search space, the corresponding histogram part B_k is formed and a synchronization sequence sync_k is extracted from $\overline{O}$ by comparing the first s histogram bin pairs given in W_K. For each histogram part B_k, sync_k is extracted and the Rogers-Tanimoto [17] dissimilarity

$$d_{\text{RT}} = \frac{2\,(c_{01} + c_{10})}{2\,(c_{01} + c_{10}) + c_{11} + c_{00}} \tag{4}$$

from sync is computed, where c_{ij} is the number of occurrences where a sync bit is i and a corresponding sync_k bit is j. The histogram part B_k leading to the minimum dissimilarity is used to extract the remaining $L_W - s$ bits from $\overline{O}$.

4.3. Security Analysis

4.3.1. Watermarking. By construction, the proposed watermarking algorithm is highly sensitive to histogram-changing modifications of the cover audio files, like random exchange of histogram bins or histogram flattening operations. This kind of operation is able to remove the watermark partly or even completely, as our experiments with StirMark (cf. Section 5.2) have shown.

Unauthorised embedding and detection of the watermark, on the other hand, is difficult without knowledge of the watermarking key

$$W_K = \{(x_i, y_i) \mid 1 \le i \le L_W\}. \tag{5}$$

In what follows, we provide a lower bound of the number of possible keys.

W_K consists of L_W bin pairs chosen from the relevant part B of the amplitude histogram containing L_B bins. We divide the bins into L_W equal parts of size L_B/L_W. Assuming that, in each pair, the first bin comes from a different part and second bin is chosen in a distance $\le \text{step}_{\max}$ from the first, there are $(L_B/L_W) \cdot \text{step}_{\max}$ possibilities to choose a single bin pair. Because there are L_W bin pairs and their order is important, we arrive at a bound

$$M = L_W! \times \left(\frac{L_B}{L_W} \cdot \text{step}_{\max}\right)^{L_W} \tag{6}$$

for the number of keys. Typical parameter choices like $L_B = 128$, $L_W = 32$ and $\text{step}_{\max} = 5$ lead to a bound $M \approx 2^{232}$. Note that if the histogram pairs in W_K are revealed, but not their order, as in the protocol described in Section 6.2, a watermark length of $L_W = 48$ is still sufficient to provide a key length of about 200 bit.

4.3.2. Permutation Cipher. As mentioned above in Section 4.1, the Cat Map suffers from a low number of possible keys, which can be remedied only if the key parameters are changed from iteration to iteration. Because in principle the required permutations can be generated in a different way, it is more interesting in this context to look at the security of permutation ciphers for audio files in general. In [18] the authors investigated the security of permutation ciphers as applied to $H \times H$ images with L greyvalues and found that if $p = \log_L(H^2)$ plaintexts are known, attacks with a complexity $O(p \cdot H^4)$ are possible, requiring frequent key updates. Applying these results to an audio file of length N means that $p = \log_L(N)$ known plaintexts are sufficient to break the cipher, where L is the number of possible sample values. Because the bit depth of an audio file is usually higher than that of an image file (16 Bit as opposed to 8 Bit per sample, resp., pixel), the key for the permutation needs to be updated twice as often as for an image file of comparable size.

5. Experimental Results

The following experiments were carried out with a collection of audio files provided by the European Broadcast Union (EBU) for sound quality assessment. The audio files include artificially generated signals as well as speech, single instruments, and pop music (https://tech.ebu.ch/publications/tech3253).

5.1. Perceptibility. In order to measure the perceptibility of an embedded mark, the Peak Signal-to-Noise Ratio (PSNR) between the marked work and the original work was computed, as is common in the literature. Figure 3 shows the PSNR between original file and marked file for increasing length of the watermark and seven example soundfiles. The parameter set used for embedding was $L_B = 1500$, $\text{step}_{\max} = 9$, $\lambda = 2.5$, $T = 2$. According to [19], noise becomes

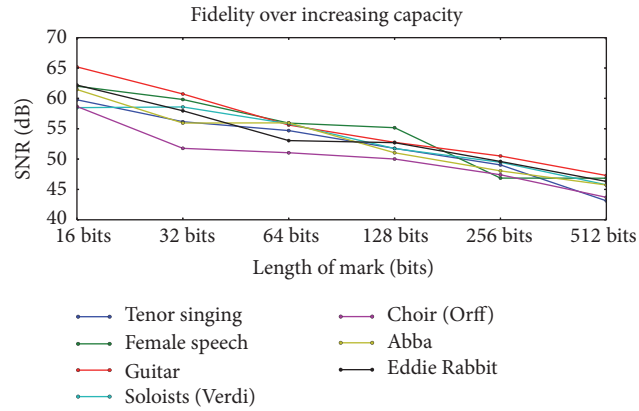


FIGURE 3: Perceptibility of watermarks with increasing lengths, where $L = 1500$.

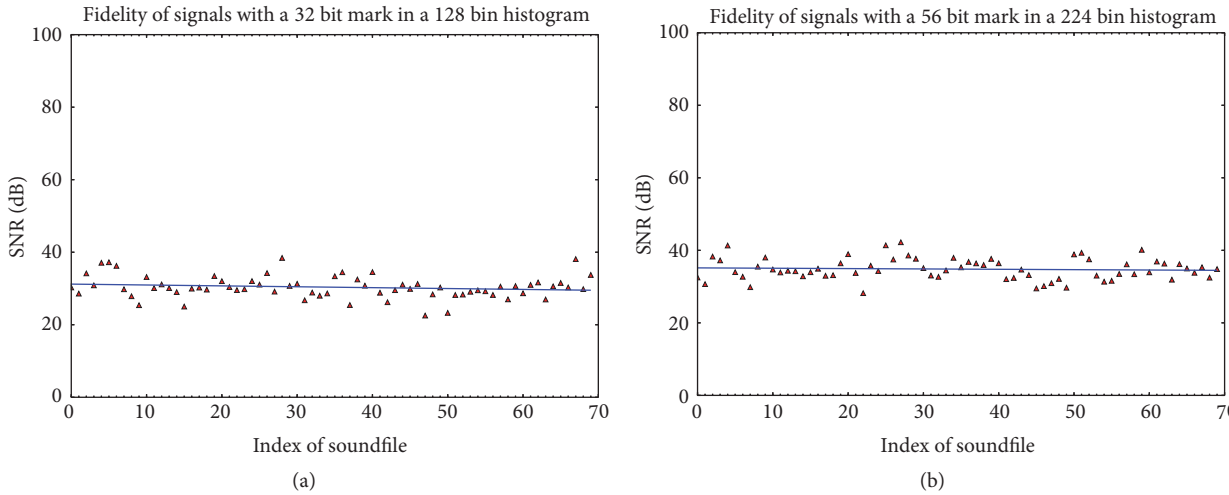


FIGURE 4: Impact of different parameter choices on perceptibility. (a) 32 bit watermark, $L_B = 128$. Average PSNR is 32 dB; (b) 56 bit watermark, $L_B = 224$. Average PSNR is 38 dB.

perceptible at a PSNR < 35 dB. Figure 3 therefore shows that using these parameters it is possible to embed up to 512 bit without problems.

If the histogram bins are broadened (i.e., if L_B is decreased), the capacity goes down, as less bins are available for embedding, while more samples are affected by the embedding, making the watermark more perceptible, but also more robust. Figure 4 shows the PSNR values for 70 test files for $L_B = 128$ and a 32 Bit mark and for $L_B = 224$ and a 56 Bit mark, respectively. The solid lines indicate the average PSNR values. The watermarking parameters are chosen in such a way that a good robustness against TSM attacks is achieved (cf. Section 5.2); however, the noise introduced by the watermark is at the border of being noticeable.

5.2. Robustness

5.2.1. TSM Attacks. TSM attacks basically try to desynchronize embedder and detector by compressing or extending the time axis of the audio file. A common requirement is that an audio watermark should be able to survive a rescaling of about 10% [20]. Moreover, the human auditory system is relatively insensitive to TSM attacks, which makes even higher percentages seem realistic. In resample mode, certain audio samples are repeated or removed in order to stretch or extend the time axis. In pitch-invariant mode, the speed of the audio file is modified without changing the samples. In order to implement these attacks in practice, the popular open-source tool Audacity V2.1.1 (http:/www.audacityteam.org) was used.

Figure 5 shows the Bit Error Rates (BER) when retrieving watermarks of 32 bit and 56 bit length after TSM attacks. For these parameter choices, the proposed watermarking algorithm is extremely robust against TSM attacks in resample mode, while the robustness against pitch-invariant mode is slightly worse, but still very good. In particular, the required robustness against 10% rescaling is fulfilled. In general, the robustness is quite sensitive to the choice of parameters. In particular, if the number L_B of histogram bins is further increased, the robustness decreases accordingly.

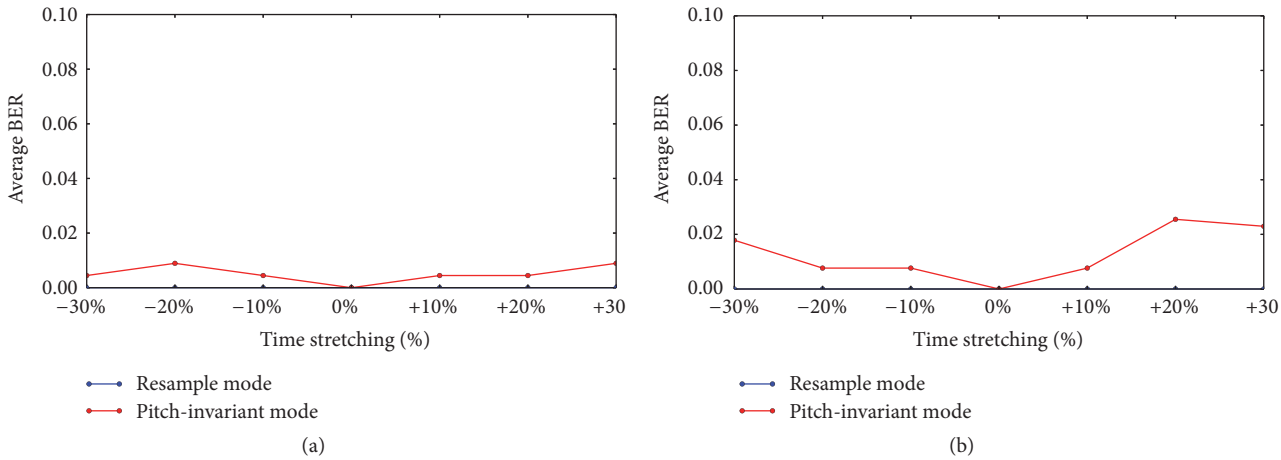


Figure 5: Robustness against TSM attacks with $L_B = 128$ and sync = W. (a) 32 bit watermark; (b) 56 bit watermark.

5.2.2. StirMark. In order to evaluate the robustness against general signal manipulations, the well-known benchmarking tool StirMark for Audio, V1.3.2 (https:/sourceforge.net/projects/stirMark) was used to simulate common attacks. Although a low number L_B of histogram bins were chosen ($L_B = 128$ or $L_B = 224$) to achieve a higher robustness, the algorithm performed very differently, depending on the type of attack. For example, low-pass filtering or inserting a sine signal into the audio file is able to destroy the mark completely. On the other hand, sample manipulations like inserting Zero-Samples, periodic deletion of samples (cropping, or CutSamples in StirMark), or manipulating the least significant bits (LSBs) of the samples do not affect the watermark. Table 1 (for $L_B = 128$) and Table 2 (for $L_B = 224$) give the detailed results of our experiments with StirMark. In general, it is hard to devise a CWE algorithm which is robust against a wide class of attacks, because in a CWE scheme the embedder must be able to operate in the encrypted domain and therefore cannot rely upon important perceptual features of the cover work. It is possible, however, to achieve certain robustness against a certain well-defined class of attacks, as the proposed algorithm shows.

5.3. Commutativity with Encryption. The presented CWE scheme is not fully commutative in theory because of the padding needed in the encryption step: as the embedding step changes the original histogram, the padding samples introduced by a mark-then-encrypt operation can be slightly different from the padding in an encrypt-then-mark operation. We tested the influence of this issue by verifying a mark which was embedded into seven test files in the following three scenarios:

(i) $V(\mathcal{M}_{W_K}(\mathcal{E}_K(O), m))$: the mark m is embedded into the encrypted cover work, and then extracted.

(ii) $V(\mathcal{E}_K(\mathcal{M}_{W_K}(O, m)))$: the mark is embedded into the plaintext cover work. The marked cover work is encrypted and the mark is extracted.

(iii) $V(E_K^{-1}(\mathcal{M}_{W_K}(\mathcal{E}_K(O), m)))$: the mark is embedded into the encrypted cover work, then the work is decrypted and the mark extracted.

In all scenarios the mark could be extracted without any errors from all test files. In practice, the number of padding samples is very small compared to the overall number of samples and they are rarely influenced by the watermarking.

Note that the noncommutativity is not intrinsic to the overall scheme, but results from the special way the permutations are generated, namely, by deploying the two-dimensional discrete Cat Map. If the permutation is generated by some alternative mechanism like, for example, the one proposed in [21], which does not require padding, the scheme is fully commutative by construction.

6. Minimum Knowledge Verification of the Mark

The discussion in Section 2 has shown that in order to be useful in a generic buyer-seller protocol, there ought to be a way to verify the watermark without fully disclosing either the mark or the watermarking key. In [2], Katzenbeisser and Craver propose a probabilistic protocol which is in principle able to integrate any symmetric watermarking algorithm. Here, we make a few modifications to this protocol to work with the proposed audio watermarking algorithm. These modifications strive to make full use of the special properties of the proposed CWE scheme and are able to eliminate a certain weakness of the scheme by Craver and Katzenbeisser.

6.1. The Protocol by Craver and Katzenbeisser. In this protocol, a prover Alice wants to prove the presence of a watermark W to some verifier Bob without disclosing W or the watermarking key W_K. The cover work O is viewed as an array of n samples. The watermark W, which has also length n, is embedded by the prover using some symmetric watermarking algorithm. The result is the marked work $\overline{O}$. Alice now generates some secret permutation τ and publishes

Table 1: Bit error rates of extracted 32 bit watermarks after various StirMark attacks on seven example audio files. Parameters used are $L_B = 128$, $\lambda = 2.5$, $T = 50$, $\text{step}_{\max} = 5$, $\Delta = 0.05$, and syn = wmk.

Index	Attack + parameter	Tenor singing	Female speech	Guitar	Soloists (Verdi)	Choir (Orff)	Abba	Eddie Rabbit
0	AddDynNoise-20	0.406	0.312	0.469	0.438	0.406	0.438	0.344
1	AddNoise-10	0.031	0.0	0.0	0.0	0.0	0.0	0.0
2	AddNoise-1000	0.438	0.375	0.438	0.469	0.406	0.562	0.438
3	AddSinus-120-3000	0.406	0.344	0.469	0.438	0.344	0.5	0.375
4	Amplify-50	0.375	0.406	0.469	0.438	0.438	0.5	0.406
5	BassBoost-150-6.123	0.344	0.312	0.5	0.469	0.406	0.469	0.406
6	BitChanger-1-99.99	0.0	0.0	0.0	0.0	0.0	0.0	0.0
7	Compressor-6.123-2.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8	CopySample-10000-2000-6000	0.0	0.0	0.0	0.0	0.0	0.0	0.0
9	CutSamples-100-7	0.0	0.0	0.0	0.0	0.0	0.0	0.0
10	Echo-50	0.406	0.406	0.531	0.469	0.469	0.469	0.344
11	Exchange	0.0	0.0	0.0	0.0	0.0	0.0	0.0
12	FFT_HLPassQuick-1024-150-15000	0.375	0.188	0.375	0.25	0.281	0.25	0.406
13	FFT_Invert-1024	0.375	0.375	0.469	0.375	0.375	0.375	0.281
14	FFT_RealReverse-1024	0.438	0.406	0.5	0.5	0.438	0.469	0.406
15	FlippSample-10000-2000-6000	0.0	0.0	0.0	0.0	0.0	0.0	0.0
16	Invert	0.375	0.375	0.469	0.375	0.375	0.375	0.281
17	LSBZero	0.0	0.0	0.0	0.0	0.0	0.0	0.0
18	Normalizer1-2048-28000	0.281	0.312	0.344	0.219	0.281	0.281	0.375
19	RC_LowPass-9000	0.25	0.219	0.219	0.406	0.406	0.375	0.094
20	ReplaceSamples-525-1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0
21	Smooth	0.344	0.25	0.406	0.438	0.406	0.469	0.219
22	Smooth2	0.062	0.125	0.219	0.125	0.062	0.0	0.094
23	Stat1	0.281	0.219	0.281	0.438	0.406	0.406	0.031
24	ZeroCross-1000	0.312	0.125	0.375	0.219	0.094	0.281	0.25
25	ZeroLength-10	0.0	0.0	0.0	0.0	0.0	0.0	0.0
26	ZeroRemove	0.0	0.0	0.0	0.0	0.0	0.0	0.0

$\overline{O}, \tau(\overline{O}), \tau(W)$ along with a random graph G having n nodes and $\tau(G)$.

In order to prove the presence of W within $\overline{O}$ to Bob, Alice and Bob engage in a multistep protocol. Each step i consists of the following substeps:

(1) Alice generates two permutations ρ_i and σ_i with the property $\sigma_i \circ \rho_i = \tau$. Then she computes $G_i = \rho_i(G)$ and $\overline{O_i} = \rho_i(\overline{O})$.

(2) Alice generates a so-called *ownership ticket*

$$\text{OT} = \left(C_1\left(\sigma_i\right), C_2\left(\rho_i\right), H\left(\overline{O_i}\right), H\left(G_i\right)\right), \tag{7}$$

where H is a secure hashfunction and C_1 and C_2 are encrypted versions of σ_i and ρ_i, respectively. Alice sends OT to Bob.

(3) Bob flips a coin and, depending on the outcome, asks Alice either to decrypt C_1 or C_2 for him.

(4) If C_1 is opened, Bob can compute $\overline{O_i} = \sigma_i^{-1}(\tau(\overline{O}))$ and $G_i = \sigma_i^{-1}(\tau(G))$ and verify the hashvalues contained in OT. Having thus verified to be in possession of the correct σ_i, Bob goes on to verify that $W_i = \sigma_i^{-1}(\tau(W))$ is present within $\overline{O_i}$.

(5) If C_2 is opened, Bob computes $\overline{O_i} = \rho_i(\overline{O})$ and $G_i = \rho_i(G)$ and verifies the hashvalues contained in OT. In this case, Alice's knowledge of τ is verified.

Craver and Katzenbeisser go on to show that if these steps are repeated k times, Alice has probability 2^{-k} to fool Bob into believing that her watermark is contained in $\overline{O}$.

In our opinion, this protocol, while being very ingenious, has two drawbacks: first, the verifier gets to know the marked work $\overline{O}$ together with $\tau(\overline{O})$ and can therefore get some information about the secret permutation τ (the same is true for G and $\tau(G)$, but in this case getting information about τ means being able to solve an instance of the graph isomorphism problem [22]).

Second, and more importantly, it is not clear how exactly in step (4) the presence of W_i in the scrambled work $\overline{O_i}$ should be verified without disclosure of the watermarking key.

6.2. The Modified Version. In our modified version of the protocol, we take advantage of the special structure of our watermarking algorithm and strive to eliminate the two drawbacks mentioned above. As in Craver and Katzenbeissers original protocol, the prover Alice generates a secret

Table 2: Bit error rates of extracted 56 bit watermarks after various StirMark attacks on seven example audio files. Parameters used are $L_B = 224$, $\lambda = 2.5$, $T = 50$, $\text{step}_{\max} = 5$, $\Delta = 0.05$, and syn = wmk.

Index	Angriff + parameter	Tenor singing	Female speech	Guitar	Soloists (Verdi)	Choir (Orff)	Abba	Eddie Rabbit
0	AddDynNoise-20	0.304	0.339	0.357	0.411	0.375	0.411	0.411
1	AddNoise-10	0.018	0.0	0.054	0.0	0.0	0.036	0.018
2	AddNoise-1000	0.339	0.357	0.464	0.357	0.375	0.5	0.429
3	AddSinus-120-3000	0.286	0.357	0.411	0.429	0.357	0.411	0.357
4	Amplify-50	0.304	0.339	0.411	0.411	0.411	0.429	0.446
5	BassBoost-150-6.123	0.286	0.375	0.375	0.446	0.357	0.429	0.446
6	BitChanger-1-99.99	0.0	0.0	0.0	0.0	0.0	0.0	0.0
7	Compressor-6.123-2.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8	CopySample-10000-2000-6000	0.0	0.0	0.0	0.0	0.0	0.0	0.0
9	CutSamples-100-7	0.0	0.0	0.0	0.0	0.0	0.0	0.0
10	Echo-50	0.321	0.375	0.411	0.446	0.411	0.446	0.446
11	Exchange	0.0	0.0	0.0	0.0	0.0	0.0	0.0
12	FFT_HLPassQuick-1024-150-15000	0.286	0.339	0.357	0.393	0.375	0.375	0.429
13	FFT_Invert-1024	0.286	0.357	0.357	0.411	0.375	0.429	0.393
14	FFT_RealReverse-1024	0.321	0.375	0.393	0.446	0.411	0.446	0.464
15	FlippSample-10000-2000-6000	0.0	0.0	0.0	0.0	0.0	0.0	0.0
16	Invert	0.268	0.339	0.375	0.429	0.375	0.393	0.411
17	LSBZero	0.0	0.036	0.161	0.143	0.0	0.161	0.143
18	Normalizer1-2048-28000	0.25	0.393	0.339	0.304	0.339	0.339	0.304
19	RC_LowPass-9000	0.214	0.393	0.339	0.375	0.411	0.393	0.214
20	ReplaceSamples-525-1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0
21	Smooth	0.232	0.321	0.321	0.429	0.393	0.429	0.25
22	Smooth2	0.143	0.107	0.161	0.089	0.036	0.125	0.089
23	Stat1	0.25	0.357	0.357	0.393	0.393	0.429	0.214
24	ZeroCross-1000	0.304	0.125	0.375	0.321	0.107	0.286	0.161
25	ZeroLength-10	0.0	0.0	0.0	0.0	0.0	0.0	0.0
26	ZeroRemove	0.0	0.0	0.0	0.0	0.0	0.0	0.0

permutation τ and a graph G. She marks the cover work O with the watermark W using the algorithm described in Section 4.2 to get the marked work $\overline{O}$. She then publishes $\overline{O}, G, \tau(G)$ and the permuted watermarking key $\tau(W_K)$, but *not* $\tau(\overline{O})$ or $\tau(W)$. Note that for the watermarking algorithm described here, the watermarking key W_K consists of a list of bin pairs. If this list is used for extracting the watermark in permuted form, the result will be the permuted watermark. The modified protocol now proceeds in k steps. Each step i consists of the following substeps:

(1) Alice generates two permutations ρ_i, σ_i with the property $\sigma_i \circ \rho_i = \tau$. Then she computes $G_i = \rho_i(G)$ and $W_i = \rho_i(W)$.

(2) Alice generates an ownership ticket

$$\text{OT} = \left(C_1\left(\sigma_i\right), C_2\left(\rho_i\right), H\left(W_i\right), H\left(G_i\right)\right), \tag{8}$$

where H is a secure hashfunction and C_1, C_2 are encrypted versions of σ_i and ρ_i, respectively. Alice sends OT to Bob.

(3) Bob flips a coin and, depending on the outcome, asks Alice either to decrypt C_1 or C_2 for him.

(4) If C_1 is opened, Bob can compute $G_i = \sigma_i^{-1}(\tau(G))$ and verify the hashvalue $H(G_i)$ contained in OT. Having thus verified to be in possession of the correct σ_i, Bob goes on to compute $W_{Ki} = \sigma_i^{-1}(\tau(W_K))$ and uses W_{Ki} to extract $W_i = \sigma_i^{-1}(\tau(W))$ from $\overline{O}$. He can check the correctness of W_i by verifying the hashvalue $H(W_i)$ in OT.

(5) If C_2 is opened, Bob computes $G_i = \rho_i(G)$ and verifies the hashvalue contained in OT. In this case, Alice's knowledge of τ is verified.

In this modified version of the protocol, it is not necessary to publish $\tau(\overline{O})$ because the permuted mark W_i can be extracted directly from $\overline{O}$. Moreover, it is clear how to do this without knowledge of the watermarking key, as the permuted key W_{k_i} will yield the permuted mark W_i. Another interesting aspect of the modified protocol in connection with the watermarking algorithm described here is that it can be applied to a permuted marked work $\pi(\overline{O})$ in exactly the same way, where π is some permutation independent of the secret permutation τ used in the interactive verification protocol.

The proposed protocol has been implemented and extensively tested in Python. For a 1024 bit watermark, the amount of data to be transferred between prover and verifier is about 4 Kilobytes per step.

A serious drawback of the protocol is the fact that publication of the permuted watermarking key reveals the bin pairs where the watermarking bits are embedded. It is therefore an easy task for an attacker to modify those bins accordingly in order to remove the watermark. By using $\tau(W_K)$, the attacker could also embed a permuted mark $\tau(W)$. The attacker would not, however, be able to prove knowledge of τ or W itself.

7. Conclusion

We have shown that existing approaches for commutative watermarking-encryption for images can be transferred to audio files. While it is hard in this context to define a watermarking algorithm which is generally robust, we could achieve a relatively strong robustness against Time-Scale Modification (TSM) attacks. By introducing some modifications into an earlier interactive verification protocol, it was possible to fit the watermarking algorithm defined here into an interactive verification protocol with minimum knowledge verification, that is, without disclosure of the watermark or the watermarking key. Our further research efforts will focus on the question of how the proposed watermarking algorithm can be made more robust in general.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] R. Schmitz, S. Li, C. Grecos, and X. Zhang, "A new approach to commutative watermarking-encryption," in *Communications and Multimedia Security*, pp. 117–130, Springer, Berlin, Germany, 2012.

[2] S. Craver and S. Katzenbeisser, "Copyright protection protocols based on asymmetric watermarking: the ticket concept," in *Communications and Multimedia Security Issues of the New Century*, pp. 159–170, Kluwer, 2001.

[3] J. Herrera-Joancomartí, S. Katzenbeisser, D. Megías et al., "Ecrypt European network of excellence in cryptology, first summary report on hybrid systems," Tech. Rep., European Network of Excellence in Cryptology, 2005, http://www.ecrypt.eu.org/ecrypt1/documents/D.WVL.5-1.0.pdf.

[4] J. Herrera-Joancomartí and D. Megías, "Watermarking in the encrypted domain," in *Proceedings of the Congreso Iberoamericano de Seguridad Informatica*, Montevideo, Uruguay, 2009.

[5] E. Becker, *Digital Rights Management: Technological, Economic, Legal and Political Aspects*, vol. 2770, Springer, 2003.

[6] A. Boho, G. Van Wallendael, A. Dooms et al., "End-to-end security for video distribution: the combination of encryption, watermarking, and video adaptation," *IEEE Signal Processing Magazine*, vol. 30, no. 2, pp. 97–107, 2013.

[7] M. M. Alani, "Security threats in cloud computing," in *Elements of Cloud Computing Security*, pp. 25–39, Springer, 2016.

[8] R. Schmitz, S. Li, C. Grecos, and X. Zhang, "Towards robust invariant commutative watermarking-encryption based on image histograms," *International Journal of Multimedia Data Engineering and Management*, vol. 5, no. 4, pp. 36–52, 2014.

[9] 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.

[10] Z. Su, J. Jiang, S. Lian, D. Hu, C. Liang, and G. Zhang, "Selective encryption for G.729 speech using chaotic maps," in *Proceedings of the 1st International Conference on Multimedia Information Networking and Security (MINES '09)*, vol. 1, pp. 488–492, IEEE, Hubei, China, November 2009.

[11] A. A. Tamimi and A. M. Abdalla, "An audio shuffle-encryption algorithm," in *Proceedings of the World Congress on Engineering and Computer Science*, San Francisco, Calif, USA, October 2014.

[12] M. A. Nematollahi, C. Vorakulpipat, and H. G. Rosales, *Digital Watermarking*, Springer, Berlin, Germany, 2016.

[13] S. Xiang, J. Huang, and R. Yang, "Time-scale invariant audio watermarking based on the statistical features in time domain," in *Information Hiding: 8th International Workshop, IH 2006, Alexandria, VA, USA, July 10–12, 2006. Revised Selcted Papers*, vol. 4437 of *Lecture Notes in Computer Science*, pp. 93–108, Springer, Berlin, Germany, 2006.

[14] G. Chen, Y. Mao, and C. K. Chui, "A symmetric image encryption scheme based on 3D chaotic cat maps," *Chaos, Solitons & Fractals*, vol. 21, no. 3, pp. 749–761, 2004.

[15] J. Fridrich, "Symmetric ciphers based on two-dimensional chaotic maps," *International Journal of Bifurcation and Chaos*, vol. 8, no. 6, pp. 1259–1284, 1998.

[16] S. Lian, J. Sun, and Z. Wang, "Security analysis of a chaos-based image encryption algorithm," *Physica A: Statistical Mechanics and Its Applications*, vol. 351, no. 2–4, pp. 645–661, 2005.

[17] S. S. Choi, S. H. Cha, and C. C. Tappert, "A survey of binary similarity and distance measures," *Journal of Systemics, Cybernetics and Informatics*, vol. 8, no. 1, pp. 43–48, 2010.

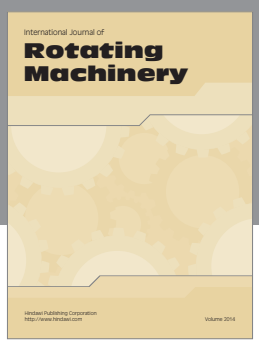
[18] S. Li, C. Li, G. Chen, N. G. Bourbakis, and K.-T. Lo, "A general quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks," *Signal Processing: Image Communication*, vol. 23, no. 3, pp. 212–223, 2008.

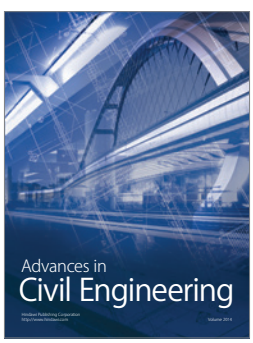
[19] A. G. Acevedo, "Audio watermarking: properties, techniques and evaluation," *Information Security and Ethics: Concepts, Methodologies, Tools, and Applications*, vol. 6, pp. 23–61, 2007.

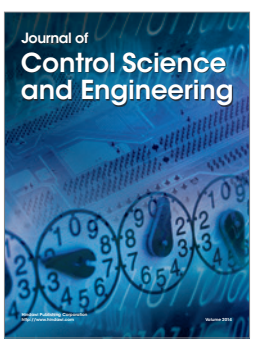
[20] Japanese Society for Rights of Authors, Composers and Publishers: Announcement of evaluation test results for step 2000, international evaluation project for digital watermark technology for music, http://www.jasrac.or.jp/watermark/ehoukoku.htm.

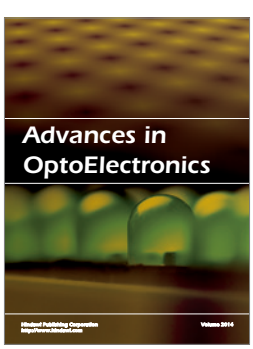
[21] G. Ye, "Image scrambling encryption algorithm of pixel bit based on chaos map," *Pattern Recognition Letters*, vol. 31, no. 5, pp. 347–354, 2010.

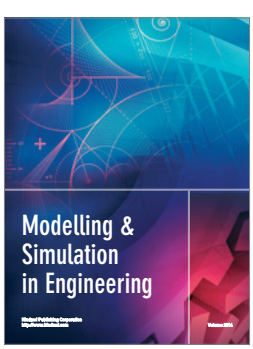
[22] J. Köbler, U. Schöning, and J. Torán, *The Graph Isomorphism Problem: Its Structural Complexity*, Springer, 2012.

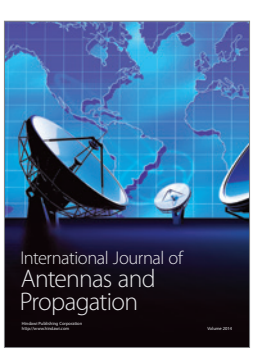


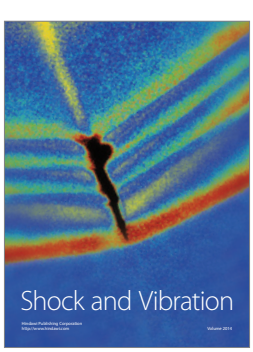


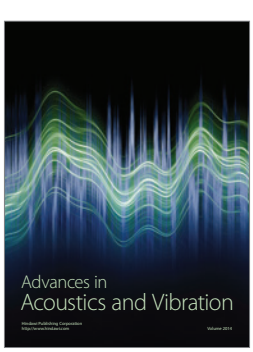












Hindawi

Submit your manuscripts at
https://www.hindawi.com