SECURITY OF ROBUST AUDIO HASHES

Stefan Thiemert, Stefan Nürnberger, Martin Steinebach, Sascha Zmudzinski

Fraunhofer SIT, Darmstadt, Germany

ABSTRACT

Robust hash algorithms can be used as tools for identifying or discriminating content, e.g. for automated tracking systems and filters for file sharing networks. When they are used in security relevant applications, such as in content-fragile watermarks or for recognizing illegal content, the security of the hash value generation becomes an important issue. In this paper we discuss possible attacks on robust hash algorithms. As an example we describe a possible attack on the audio fingerprint of Haitsma et al., resulting in a different hash value while keeping the audio files perceptually similar.

***Index Terms*—** Robust hash functions, Audio fingerprints

1. MOTIVATION

Robust hash algorithms today are seen as a robust, efficient and reliable tool for identifying or discriminating content. Known examples are automated tracking systems creating playlists of radio stations, filters for file sharing networks or user generated content websites preventing the illegal distribution of copyright protected material. There are numerous advantages for such robust hash functions. They do not modify the content as the embedding process of digital watermarks does. The hash values are usually easy to compute, i.e. complexity and costs are low. And they have proven their robustness against the typical attacks successfully circumventing DRM systems, e.g. analog copies. This makes them suited tools for applications like music recommendation or playlist generation, where attacks are not to be expected as robust hashing is an enabler of an additional service.

The issue of security arises when robust hashes are used for preventing illegal copies. Here they face the same challenges as DRM systems, digital watermarking and copy protection systems. This means that a mechanism originally not designed for security is now used in applications where attacks on protocols and algorithms are common. Obviously, this requires an analysis of their suitability for these domains. So far the main focus on robust hash security was securely generating robust hashes, like the audio RMACs discussed in [1]. In this paper we address the scenario where an attacker knows the basic hashing algorithm and wants to modify an audio file in such a way that the hash value changes, but the perceived quality at the same time stays high. If this can be achieved, attackers can circumvent e.g. filters based on hashes and distribute illegal copies via channels supposed to be blocked against this.

2. ROBUST HASHING BASICS

Common techniques for the purpose of data comparison are hash functions. A hash function is a mathematical function that maps a variable length input message to an output message digest of fixed length [2]. Hash functions are used for the identification and authentication of data and to improve table look ups in databases.

Usually, hash algorithms - by design - are extremely sensitive to changes in the input data, i.e. even a one bit change in the data results in a totally different hash value. This is especially true for cryptographic hash functions (e.g. SHA-1, RIPEMD) used in authentication security protocols like digital signatures.

For multimedia data this high sensitivity is inappropriate in many scenarios. For example, a song being perceived similarly by human observers differs strongly in binary representation depending on the storage format, e.g. Audio CD or mp3 file. Thus, for multimedia applications a number of specially designed robust hash algorithms have been introduced that are tolerant against imperceptible or moderate transformations of multimedia data. These algorithms extract features from the data that are relevant to the human perception. Robust hash algorithms are also referred to as digital fingerprinting algorithms as they allow identifying similar content by the extracted features, similar to biometric dactylogram-fingerprints for the identification of people.

In order for an algorithm to serve as a robust hash it must meet a number of requirements. The most important are [3]:

- *Distinction:* Perceptually different pieces of media data shall have different hash values
- *Robustness:* The robust hash values shall show a certain degree of perceptual invariance, i.e. two pieces of media data that are perceptually similar for an average listener shall be similar, too.

This work was supported by the Center for Advanced Security Research Darmstadt (CASED), a new interdisciplinary research center funded by the German State of Hesse through its Initiative for the Development of Scientific and Economic Excellence (LOEWE).

- *Security:* The features must survive attacks that are directly aimed at the feature extraction and consecutive processing steps.

2.1. Robust Audio Hashing

Robust audio hash algorithms have also been called audio ID or audio fingerprinting in the literature. The concept here is to derive a robust content-dependent description from audio data to be able to identify the audio data by comparing the stored and a newly calculated description. This description aims to be robust to modifications of the audio data, like e.g. mp3 compression. Known concepts [4] include the inherent periodicity of audio signals, the time-frequency landscape given by the frame-by-frame Mel-frequency cepstral coefficients, principal component analysis, adaptive quantization and channel decoding.

One audio fingerprinting scheme was introduced by Haitsma et al. [5] [6] This robust feature extraction uses a time-frequency analysis of the Fourier magnitude coefficients of the audio signal. Here, in the pre-processing step the audio signal is digitally represented by PCM samples and it is divided into overlapping frames. The frame data is then Hanning windowed and transferred to the spectral domain using the Discrete Fourier Transform. In order to consider the perceptual properties of the human auditory system the Fourier magnitude coefficients are first mapped to a logarithmically scaled frequency axis, obtaining the energy coefficients $E(n,m)$ where n denotes the frame index and m the band.

Then, in the original algorithm, the energy differences of adjacent energy bands m and $m+1$ in a given frame n are compared to those with the same band indices in the previous frame $n-1$ as follows:

$$\begin{aligned} d(n,m) = E(n,m) - E(n,m+1) - \\ (E(n-1,m) - E(n-1,m+1)) \end{aligned} \tag{1}$$

In a short form (1) can be seen as the difference between the inner energy differences of current frame D_C and previous frame D_P.

$$d(n,m) = D_C - D_P \tag{2}$$

The decision about the fingerprint bits $H(n,m)$ is then made according to

$$H(n,m) = \begin{cases} 1 & if \quad d(n,m) > 0 \\ 0 & if \quad d(n,m) \leq 0 \end{cases} \tag{3}$$

It should be noted that the value of each hash bit is dependent on two pairs of energy coefficients in consecutive frames. By design, both spectral properties and temporal properties of the signal are considered.

In order to increase the robustness in the post-processing step, a number of 256 so-called sub-fingerprints (32 bit each) are extracted from consecutive audio frames with a strong overlap factor (31/32 = 97%) and combined to a hash block that represents approximately 3 seconds of playing time. Haitsma et al. decided in [6] to use a Bit Error Rate (BER) of 25% as recognition threshold, effectively meaning that a difference of a song to test and a song in the database that is greater than 25% would be considered to be a different song. A BER less or equal to 25% means that it is most likely the same song, even though at least one of them is in really bad quality. The song with the lowest BER, and if it is below the 25% threshold, is most certainly the intended match of the song one was looking for. A newer version of the paper states that even a 35% threshold is still sufficient to distinguish different songs [7]. This on the other hand increases the robustness in contrast to a 25% threshold.

Although quite simple in design, this extracted audio feature provides a high level of robustness against encoding to lossy compression, re-sampling, filtering, dynamics compression, noise addition and analog tape recording [5] [6].

3. ATTACKS ON ROBUST HASHES

In this section we want to discuss possible attacks on robust hashes and show an example of a more complex attack on the before mentioned audio hash algorithm of Haitsma et al. In general there are two kinds of attacks:

- Attacks producing a different hash value without a perceptual difference
- Attacks producing the same hash value with perceptually different content

The first family of attacks is especially focused on systems preventing the upload of copyright protected content. Providers of user generated content (e.g. YouTube) are forced to remove detected copyright protected content. In many cases the fingerprint of the copyright protected material is stored in a database. If a user tries to upload the content once again the system detects the similarity with the copyright protected content and rejects the upload. If a user is able to create a different hash value even with the perceptually same material the copyright protection system is circumvented.

The second family of attacks is focused on systems using the hash values as authentication method, i.e. verifying the authenticity of the content. Such applications are e.g. content-fragile watermarks [8]. If an attacker is able to create a forged content with the same hash value non-authentic content can be verified as authentic. This is especially critical in the case of lawsuits.

Following we will discuss an attack on the algorithm described in section 2.1, which is one of the best known audio fingerprints in literature. The goal is to modify the content in such a way that the quality of the modified content remains the same but results in a different hash value, i.e. it belongs to the first family of attacks. Since we modify several bits of the hash value directly we call this attack the "Direct Hash Bit Attack".

3.1. Direct Hash Bit Attack - Main Idea

Equation (3) states, that the hash produces a 1 regardless of the amount that D_C is greater than D_P. In turn this means, that the internal double precision floating point values can by definition have a relative distance as little as $\epsilon = 1.1 \cdot 10^{-16}$. Those very small differences are very likely to flip when quality changes and are impossible to hear. D_C represents the energy of one of the 32 sub-bands, thus the square of the amplitudes. Therefore the amplitude would have to be changed by a factor of $\sqrt{1+\epsilon}$, what is equal to $4.8 \cdot 10^{-16}$ dB. This is far beyond the loudness resolution of standard PCM wave files ($2.7 \cdot 10^{-4}$ dB for 16 bits), neither is it perceptible by the human ear.

Now that we know that very similar energies are very likely to cause bit flips, the reasonable question for quantitative measurement raises. Figure 1 shows the distribution of the difference $D_C - D_P$. Due to the very steep slope of the curve toward zero, the bigger part of the area is accumulated at the centre, around zero. That in turn means, that the desired 25% of bits we only need to touch are very close to zero. Independent of the genre used to create the distribution (Pop, Rock, Classic and Blues) the boundaries of the 25% area around the Y-axis were approximately from -12 to 11. That means, that the energies of the less weak bits at the boundary have to be changed by at most 12 units, what depending on the amplitude can be as high as 11.1 dB for very quiet signals.

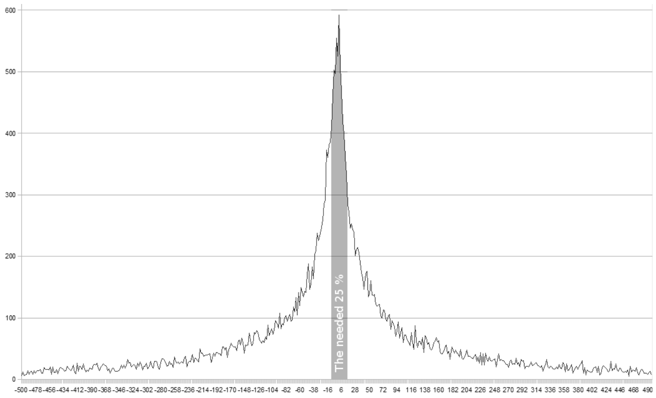

Fig. 1. Distribution of difference $D_C - D_P$

In a later paper of Haitsma [7], he was obviously aware of the fact that there are weak bits, i.e. energies close to each other. So he introduced the additional reliability bit. For each hash-block additional 32 bits are saved stating whether the according index in the hash bit is deemed to be reliable. These reliable bits are calculated as a threshold distance of energies. If the energies are too close to each other, the bit is marked as unreliable, whereas a distance greater than that threshold is claimed to be reliable. Unfortunately the paper does not contain a statement about the value of the threshold. However, a very low threshold would deem almost every hash bit as reliable.

The general idea of the Direct Hash Bit Attack is to deliberately modify the input energies that make up the 32 hash bits per frame in order to selectively flip certain bits. That means it would also be possible to 'impersonate' the hash of a provided different file. To achieve this, after computing the current hash, the bits are flipped accordingly and the creating process is reversed to change the input energies and in turn the raw input data. As it is possible to selectively flip the bits according to a provided second hash, we do not differ between impersonating another hash and attacking a hash, which means to flip more than 25% of the original hash.

In order to know which hash bits have to be changed and which can be left untouched according to the provided bit pattern to impersonate, the robust hash of the original audio file is generated first. This process is not different from the normal analysis stage of the program. After each of the 32 bits for every single frame was calculated, the attacking process is initiated.

3.2. Attack Order

Statistical analysis of different music genres has shown that a hash-block usually incorporates weak bits (small energy distances) as well as strong bits (big energy differences). Since modification of strong bits would be perceptually noticeable only a subset containing the smallest differences of the 32 hash bits is selected for an attack.

In addition the modification of stronger bits would influence other frames and therefore reduce the overall amount of flipped bits. Experiments that attacked only 20% of the weakest bits have shown that in average an amount of 14 bit-flip attempts per hash block (14/32 44%) leads to a maximal ratio of overall flipped bits about 17-18%. To achieve the necessary 25% of overall flipped bits, 35% of the weakest bits have to be taken into consideration and the maximum shifted to 17 bit-flip attempts per hash-block. While analyzing the file in order to compute the hash-block (stage a) the hash value for each frame is weighted with a confidence value, which is the sum of the 14 smallest distances. Hence a small total weight of a hash-block implies an overall small distance between D_C and D_P. With the weighted hash value the position of the frame is stored. Afterwards, the hashes are sorted by their confidence value in ascending order, starting with the weakest hash blocks. A loop iterates through the sorted list and stops whenever the desired threshold of attacked hash bits is exceeded. This ensures that the needed threshold of 25% flipped bits is achieved with least possible degradation.

3.3. Attack Conditions

For every single frame to attack the corresponding raw PCM samples of that particular frame, as well as the previous frame (that overlaps $\frac{31}{32}$) are extracted from the wave file and converted to frequency domain. Afterwards the phase is removed and the energies are calculated. Then the new hash bits H'_n for the potentially already altered underlying frames are accordingly computed:

$$H'_n = \begin{cases} 1 & if \quad D_C > D_P \\ 0 & if \quad D_C \leq D_P \end{cases} \tag{4}$$

Each of the 32 hash bits of a block is then compared according to the bits in the hash to impersonate. They can only be equal or different, so there are two cases:

- The newly computed hash H'_n already equals the hash to impersonate. In this case, nothing is done and the next hash bit is inspected.
- The new computed hash H'_n is different from the hash to impersonate, which means the attack has to be initialized in order to flip the bits accordingly.

3.4. The Attack

To flip the bit, the energy difference D_C is approximated to D_P, plus a small adjustable offset to ensure the bit will flip as well when exposed to influences by overlapping frames. Let d denote the difference to the desired value including the offset, then the bit flips for the new energy difference $\hat{D}_C = D_C - d$, resulting in $\hat{D}_C \leq D_P$.

4. EVALUATION

In this section we present the evaluation results concerning the Direct Hash Bit Attack against the scheme of Haitsma et al. We mainly focus on the properties Bit Error Rate, audio quality degradation and complexity. The set of files we used for evaluation contained 164 audio files of different genres. The total duration of all audio files is 13.3 hours with an average duration of 4:53 min.

4.1. Bit Error Rate

Figure 2 shows the Bit Error Rate (BER) of all tested audio files after the attack and in addition followed by an MP3 compression. The BER is the ratio of flipped bits compared to the hash value of the original content. It ranges from 4 to 33% for uncompressed material and from 8 to 33% for compressed material. In 25 (resp. 37) of the 164 files the BER was above 25%, which is the threshold to decide that audio files are different. If we would set the required BER down to 22% the number of non-authentic files would increase to 72 (resp. 80). According to [5] the BER introduced by typical modifications (e.g. MP3 compression and dynamic compression) is usually far beyond the critical 25% threshold. In turn, the differences of the audio files to each other usually have a BER of approximately 50%, like one would expect the hamming distance of two random bit streams to be. The results show, that with the direct modification of a few bits a new audio hash can be created. Hence an attacker, trying to circumvent a system preventing him from uploading copyright protected content, would be able to upload the content once again without detection. Further studies need to show if the attack produces audio files of acceptable quality when we enforce a BER of 35% according to [7].

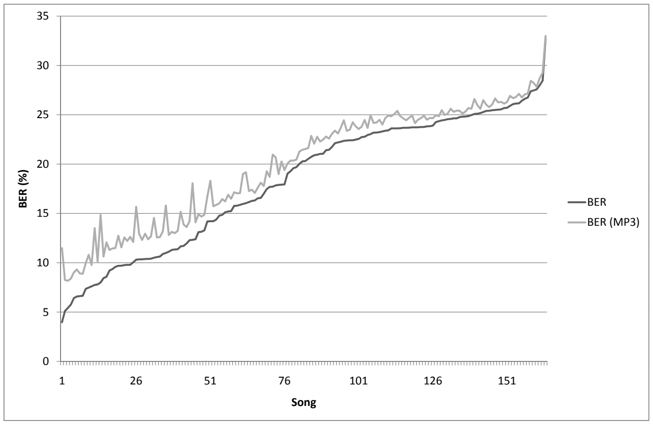

Fig. 2. Bit Error Rate after the Direct Hash Bit Attack

4.2. Audio Quality Degradation

Objective Difference Grade (ODG) is a unit of measurement of ITU-R BS.1387-1 Perceptual Evaluation of Audio Quality (PEAQ) and ranges from 0.0 ODG, representing the best quality, to -4.0 ODG, representing the lowest quality level. The values were obtained using the Opera$^{\text{TM}}$command line analyzing tool performing the PEAQ basic test.

The results in figure 3 show that compared to the BER the Direct Hash Bit Attack decreases the quality of the audio files slightly. The ODG values range from -0.08 ODG to -3.3 ODG (resp. -0.54 ODG to -2.9 ODG) with an average of -1.67 ODG (resp. -1.71 ODG). We can conclude that the Direct Hash Bit Attack changes the sound quality significantly for half of the files. It is up to the application scenario if this would be relevant to the user.

Experiments have shown that regardless of the fact, that the BER introduced by a deliberate attack depends on the audio characteristics itself, the resulting audio quality in ODG is approximately linear to the number of bits flipped. This in turn means, that the a priori set number of bits to be attacked cannot give a rough estimate of the resulting quality of the attacked audio material. As the proportionality constant depends on the actual audio material, no precise estimate can be

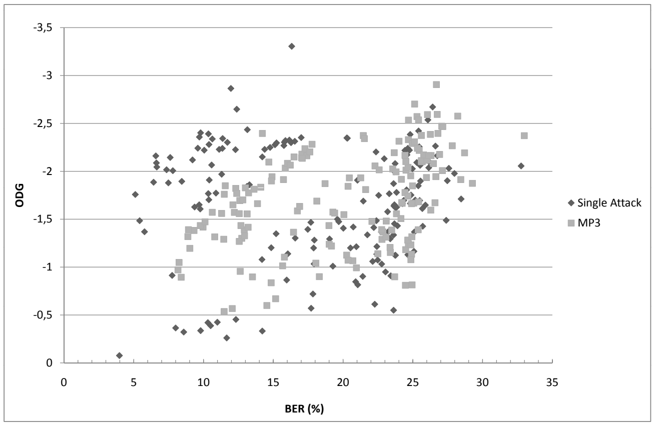

Fig. 3. ODG values of the modified audio files compared to BER

assumed about the resulting BER either. Even though, once two different quality degradations are known, a linear correlation can be estimated. Furthermore, it is noteworthy that despite that fact, our attack involves a psycho acoustic model, a subsequent MP3 encoding of the file (even at 192 kbit/s) enhanced the ODG quality of the attacked file. The positive (with respect to an intended attack) side effect is, that the resulting BER of the MP3-encoded attacked files are usually 3-5% higher. This means, that attacking the original audio material with resulting BER of 23% will usually suffice if it is subsequently encoded using the MP3 format.

4.3. Complexity

We applied the attack on an Intel Core 2 Duo 3GHz machine. The computation of the hash performs 50 times faster than real time. The attack itself took in average 1.38 times of real time, both for mono and stereo files. Since the process to create an audio file with a modified hash value is not time critical the complexity of the Direct Hash Bit Attack is acceptable.

5. CONCLUSION AND FUTURE WORK

Robust hash functions show an excellent performance concerning the properties distinction and robustness. Perceptual differences can be detected as well as perceptual similarities after compression. When such functions are used within security relevant systems, e.g. copyright protection and counterfeit detection, the security of the hash function becomes an important issue. In our work we have introduced a concept for attacking robust hashes utilizing the various strength of the individual hash bits and provided test results derived from a prototypic implementation proofing its general feasibility. We have shown that a slight modification of audio documents results in a different hash value created with a well-known audio hash method.

To provide the security of robust hash functions in such systems the feature extraction and hash generation neccessarily needs to be combined with a shared secret, i.e. a secret key. In [8] we modified the scheme of Haitsma et al. in such a way that the selection of coefficients is based on a key instead of using consecutive bands and frames.

Even though a secret key is used for generating a hash value it needs to be analyzed whether the hash function is suitable for the usage in security relevant applications. Here the evaluation needs to be focused on the points whether the content can be modified in such a way that perceptual similar content results in a different hash or different content results in the same hash value. Examples for both application scenarios were given in section 3.

6. REFERENCES

[1] J. Fridrich and M. Goljan, "Robust hash functions for digital watermarking," in *Proceedings of the International Conference on Information Technology: Coding and Computing, Las Vegas, USA*, 2000.

[2] B. Schneier, *Applied Cryptography, Second Edition*, Wiley & Sons, 1996.

[3] M. K. Mıcak and R. Venkatesan, "A perceptual audio hashing algorithm: A tool for robust audio identification and information hiding," in *4th International Workshop Information Hiding, IH 2001*, Moskowitz, Ed. 2001, vol. 2137 of *LNCS*, Springer, Heidelberg.

[4] H. Özer, B. Sankur, and N. Memon, "Robust audio hashing for audio identification," in *Proceedings of 12th European Signal Processing Conference (EUSIPCO), Vienna, Austria*, 2004.

[5] T. Kalker, J. Haitsma, and J. Oostveen, "Robust audio hashing for content identification," in *Proceedings of the Int. Workshop on Content Based Multimedia Indexing (CBMI), Brescia, Italy*, 2001.

[6] J. Haitsma and T. Kalker, "A highly robust audio fingerprinting system," in *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR), Paris, France*, 2002.

[7] J. Haitsma, "Audio fingerprinting - a new technology to identify music," Tech. Rep., TU Eindhoven, 2002.

[8] S. Zmudzinski and M. Steinebach, "Perception-based audio authentication watermarking," in *Media Forensics and Security XI, San Jose, USA*, 2009, vol. 7254 of *Proceedings of SPIE*.