AUTHENTICATION OF MP4 FILE BY PERCEPTUAL HASH AND DATA HIDING

April Pyone Maung Maung[1], Yiqi Tew[2], and KokSheik Wong[3]

[1]Faculty of Computer Science & Information Technology, University of Malaya, Malaysia
[2]Faculty of Computing and Information Technology, Tunku Abdul Rahman University College, Malaysia
[3]School of Information Technology, Monash University Malaysia, Malaysia

Email: imdad@siswa.um.edu.my[1*] (corresponding author), yiqi@tarc.edu.my[2*] (corresponding author), wong.koksheik@monash.edu.my[3]

DOI: https://doi.org/10.22452/mjcs.vol32no4.4

ABSTRACT

With advances in multimedia signal processing and abundance of data, multimedia authentication becomes more and more challenging and demanding. Although many prior techniques were put forward to authenticate individual video or audio stream, the cross-layer authentication of audio-visual content as a whole remains unexplored. In this work, an authentication system of the dominant audio-visual content file format (MP4) is presented. Specifically, the perceptual hashes of I-frames from video stream are encrypted and embedded into the audio stream, and the audio hash vector is encrypted and embedded in the synchronization information, i.e., stts *box, in the MP4 file. The proposed system is implemented and tested on various videos downloaded from YouTube. Results show that the proposed system can authenticate MP4 files correctly without degrading the video quality while being able to provide indication of tampering.*

Keywords**:** MP4, Perceptual Hash, Data Hiding, Authentication

1.0 INTRODUCTION

Multimedia authentication is the process of confirming the genuineness of a content and its alleged source. It is vital and critical for many applications, especially those involving legal issues. The tampered contents can keep us away from the true information, and we may perceive information wrongly, which can lead to unwanted situations. With various free editing tools available, there is a high probability that multimedia data are manipulated for malicious uses. Researchers around the world have been working on multimedia authentication since several decades ago by using different techniques such as *fragile watermark* where authentication information is inserted into the content [1, 2], robust authentication against compression where information derived from the content is stored in a database [3], as well as the combined use of data hiding and cryptography [4, 5].

Thanks to the recent advances of smart device and ubiquitous network environment, audio-visual content has become a major part of our digital life style. According to the report by Lua [6], everyday >500 million hours of videos are watched on YouTube itself. New features offered by social media (e.g., live video broadcast), video-on-demand service, and home surveillance video system further encourage the use of video. However, the availability of smart devices with high computational power at affordable price and user-friendly video editing softwares have resulted in some serious issues, where videos can be easily tampered with or edited for various purposes, including those with political motive and defamation in nature. Therefore, the research on authentication of video content is more important than ever to verify the genuineness of a video due to the impact and level of penetration of today's technologies.

The work by Thies et al. [7] is able to reenact a person in video real-time. In addition, there is also user-friendly voice generator such as Adobe Voco (i.e., "Photoshop-for-Voice" by Adobe), which allows user to generate the voice of a specific person, in addition to the traditional tasks such as copy-paste, crop, etc. Furthermore, we have massive volume of data with smartphone infiltration. With powerful tools and enormous data, verification of the genuineness of audio-visual content (MP4 file) naturally becomes more demanding and challenging.

In this work, we propose an authentication scheme by utilizing the audio data and the synchronization information of a MP4 file. Specifically, I-frames of the video streams are extracted, and their corresponding perceptual hashes

are generated. Each hash vector is encrypted using the user key, and the chipertext is inserted in the audio data. Next, we compute perceptual hash vector of the audio data. The hash vector of the audio data are also encrypted using the user key and then hidden into the synchronization information of MP4 file. The proposed joint authentication method makes the following contributions: (a) video-quality is preserved since no data are hidden into the video stream, and; (b) degree of tampering can be estimated because the hamming distance between the extracted perceptual hash and the generated one can give some indication of the amount of tampering.

The rest of this paper is organized as follows: Section 2 presents a brief overview of MP4 container format, AAC, perceptual hash and data hiding in general. The proposed method is put forward in Section 4. Experimental results are presented in Section 5 and analyzed in Section 6. Finally, Section 8 concludes this paper.

2.0 PRELIMINARIES

2.1 Overview of MP4

MP4 (MPEG-4 Part 14) [8] is one of the most widely used container format for web and mobile videos. It is defined by an atomic hierarchical structure and composed of boxes (viz., atoms). Each box comprises of three parts: *size* (i.e., the entire length of the box in terms of bytes), *type* (i.e., 4 characters code identifier) and *data* (i.e., the data of this box or other boxes). Fig. 1 shows the structure of a box in MP4 container format.

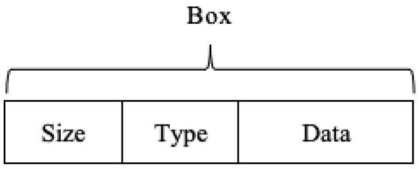

Fig. 1. The structure of a box.

Fig. 2 illustrates the structure of a typical MP4 file. At the top-most level, there are three boxes: *ftyp*, *moov* and *mdat*. The *ftyp* box contains the identification information of the MP4 file. The *moov* box stores the necessary information (viz., metadata) of the audio and video streams for decoding. The *mdat* box includes the actual audio and video data of the container. A box (e.g., *moov* and *trak*) that contains other boxes is called a parent box, and it carries related children boxes. On the other hand, a child box does not contain any boxes and it stores data such as *ftyp*, *mvhd*, *stts*, *stco*, etc. Detailed information about the MP4 structure can be found in the documentation of MP4 [8].

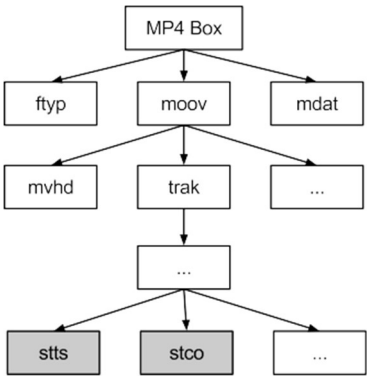

Fig. 2. The structure of MP4 container.

In this work, the *stts* box is exploited for data embedding. Therefore, we elaborate on *stts*, which is shaded in Fig. 2. The *stts* box stores sample to time information, which is utilized for audio-video synchronization in MP4 with respect to timescale. In MP4 time coordinate system, every media track has its own timescale (i.e., the number of units per second). For example, the timescale of 30 in a video stream refers to the timing of 30 frames per second. Time to sample information in the *stts* box is basically one or more entries of sample count (hereinafter referred to as S) and duration (hereinafter referred to as δ). Here, S defines the number of media samples and δ indicates

the length of a media sample (viz., video frame or audio sample) in the unit of its corresponding timescale. For example, consider a video with 500 frames given the setting of timescale 30 and $\delta = 1$. Assume the *stts* box carries only one entry of (S = 500, δ = 1). This means that each of the S = 500 frames has 1 unit of timescale, which translates to approximately 1/30 = 0.033 second.

Besides the structure described above, MP4 is the container format for several video and audio formats. The most common ones are H.264/Advanced Video Coding (AVC) [9] and Advanced Audio Coding (AAC) [10]. Therefore, in this work, the proposed system is designed to authenticate MP4 containing H.264/AVC video and AAC audio.

2.2 Overview of AAC

AAC is a perceptual audio codec and it consists of 4 basic building blocks, namely, filter bank, perceptual model, quantization and coding, and encoding of bitstream as shown in Fig. 3 [11]. The ISO/IEC 13818-7 MPEG-2 AAC [12] is non-backward compatible, but it improves on coding efficiency and achieves better quality at low bit rates when compared to MPEG-1 [11]. The standardization started in 1994 and was finalized in 1997[13].

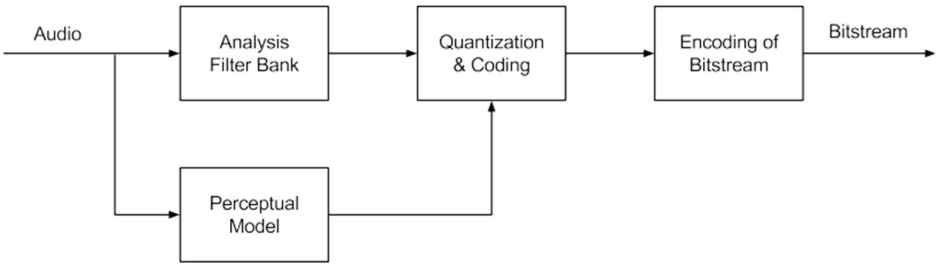

Fig. 3. Basic block diagram of AAC encoder [11].

AAC employs Modified Discrete Transform (MDCT) based filter bank to transform the audio samples into frequency coefficients. This MDCT-based filter bank takes windows of audio samples, either 2048 or 256 samples at a time, and produces 1024 or 128 spectral coefficients, respectively. There are 4 types of window, namely, *long window*, *long start window*, *long stop window*, and *short window*. Every *long window* produces 1024 spectral coefficients, while *short window* produces 128 spectral coefficients. *Short windows* always appear in group of 8, and hence they also produce 1024 coefficients (i.e., 8×128) in total. All windows are arranged to overlap 50% with its preceding neighbor for time domain aliasing cancellation (TDAC). The coefficients are then placed into intervals of 49 frequency bands.

AAC uses Perceptual (Psycho-acoustics) Model 2. The psychoacoustic model calculates the masking threshold for each band based on the input signals, which is then compared to the coefficients. If the coefficients are smaller than the thresholds, they are ready for the quantization. Specifically, AAC uses non-uniform quantizer, which needs to satisfy the requirements as stipulated by the pyscho-acoustic model. The coefficients in each scale factor band share the same scale factor. At the same time, the encoder needs to take care of the number of allowed bits for user specified bitrate. In addition, AAC is equipped with a bit reservoir to allow extra saved bits to be shared in difficult-to-encode frames. To assemble format-compliant bitstream, AAC uses bitstream formatter to multiplex Huffman codes and all the information needed by the decoder.

There are other features such as gain control, temporal noise shaping (TNS), prediction, etc. They can be selectively deployed according to different AAC profiles. For detailed information about AAC features, we refer the interested readers to Ref. [14]. In this work, we exploit the quantized spectral coefficients to realize the proposed authentication scheme.

2.3 Perceptual Hash

A perceptual hash is a vector derived from various features from the content. It is also known as fingerprint of a multimedia file, which commonly utilized for content based identification systems such as songs identification (e.g., Shazam [15]). Perceptual hashes are robust against compression, color adjustments, rotations, etc. In contrast

to cryptographic hashes which are highly sensitive to changes, perceptual hashes are correlated if the features from the content are similar.

2.4 Data Insertion

Data insertion (viz., data hiding) is a process of inserting data into a host content such as image, audio, and video, where the data are either external, internal (e.g., derived from the content), or the combination of both. Data hiding can be further fine-tuned to achieve a wide range of applications, including such as watermarking, authentication, and steganography.

There are various ways to insert data into a content. One of the most common approaches is the least significant bit (LSB) substitution using selected integers such as pixel values, quantized coefficients, etc. As for this work, we revisit some data insertion methods which are designed for H.264/AVC, AAC and MP4 metadata. Tew et al. surveys various data hiding methods in H.264/AVC including bit plane replacement, spread spectrum embedding, histogram manipulation, mapping rules, divisibility and matrix encoding that exploits prediction process, transform process, quantization process and entropy encoding [16]. Similarly, for AAC, there are modulo watermarking [17], spread spectrum method [18], matrix encoding [19], sign bit manipulation in Huffman code words [20], Huffman coding section based steganography [21], etc. Apart from methods in the compression layer, there are data insertion methods designed to operate in the system layer (i.e., MP4 container) such as MP4 flags manipulation [22] and TrueCrypt container insertion [23]. Additionally, in our previous work, we proposed a data insertion method by exploiting the audio-video synchronization information in MP4 container [24].

In this work, we employ a simple LSB manipulation to the AAC quantized coefficients and the modified version of data insertion method [24] without causing any synchronization error.

3.0 RELATED WORK

Although numerous works have been carried out for authentication of audio [25], image [26, 27], and video [28], they are designed to operate only on one type of data. The literature of authentication is still limited for audio-visual content (hereinafter referred to as movie) in any container format that holds multiple tracks as a whole, especially for MP4, which is the most widely used container format. One of the earliest work is by Sakar et al. [29], where an authentication method is proposed for XML based audio-visual content, i.e., MPEG-4 Part 11 (i.e., XMT format). However, Sakar et al.'s method is only applicable for MPEG-4 XMT structure. Jake et al. then proposed an authentication scheme [30] using the MP4 structure and metadata. However, the functionality of the proposed authentication method depends on the unique MP4 structure of specific devices, although MP4 itself is a wrapper where the real data are the audio and video streams within the MP4 container. In addition, using metadata alone is insufficient to verify the authenticity of audio and video inside MP4.

Previously, we proposed an authentication system [5] by using joint data embedding in audio-video tracks. To the best of our knowledge, it is the first cross-track authentication technique in the literature. Specifically, the authentication tags are generated based on the statistics of quantized coefficients and the cryptographic hash function is utilized. However, as multimedia data are often transmitted over the network nowadays, [5] is impractical because a single flipped bit (e.g., due to transmission error) in MP4 would cause the authentication to fail. In addition, data insertion in the video stream also degrade the visual quality. Therefore, in this work, we improve our previous work [5] by making it more robust (i.e., noise-resistance) while preserving the video quality.

4.0 AUTHENTICATION SCHEME

The proposed authentication scheme relies upon perceptual hash and data hiding techniques. Specifically, the perceptual hashes of I-frames from the video stream are embedded into audio stream (using data insertion), and the perceptual hash vector of resulting audio stream is embedded into the MP4 synchronization information (i.e., *stts* box). Fig. 4 illustrates the overall architecture of the proposed scheme, where each step is detailed as follows.

Step 1 The input MP4 file is de-multiplexed to separate the audio and video streams.

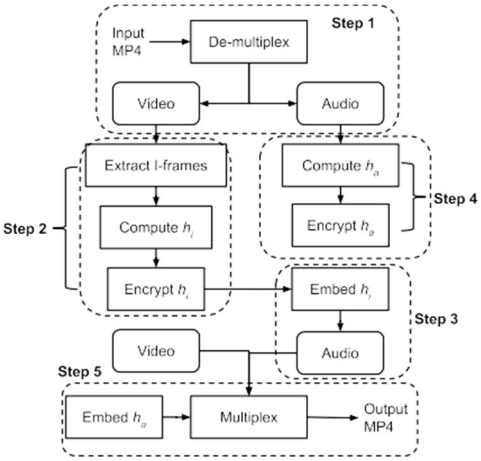

Fig. 4. Architecture overview of the proposed system.

Step 2 The I-frames from the video stream are then extracted. Let I_0 denote the first I-frame, I_1 be the second I-frame, and so forth. The hash of the i-th I-frame (denoted by I_i) is computed by using Type-II DCT based hashing algorithm (denoted by H), which is inspired by [31]. The length of each hash is 64 bits / 8 bytes. The resulting hash value is encrypted by using a strong encryption algorithm E (e.g., Triple DES [32], AES [33]) with a user key K. Then, the encrypted hashes are concatenated to form a single hash string h_v as follows:

$$h_v = E(H_{image}(I_i), k)|| \cdots ||E(H_{image}(I_N), k), \quad (1)$$

where H_{image} is the perceptual hash function for image.

Step 3 The video hash, h_v is embedded into the audio stream by implementing LSB substitution on the quantized coefficients of AAC stream. Recall that MP4 holds H.264 video stream and AAC audio stream. There are1024 quantized spectral coefficients in a AAC audio frame. For $j \in \{1, \ldots, 1024\}$, let $c[j]$ denote the j-th quantized coefficient, and let $c'[j]$ be the modified $c[j]$ after LSB substitution. We utilize the quantized spectral coefficients $|c[j]| > 2$ as the host coefficients to embed h_v, where $|X|$ denotes the magnitude of the value X. For the boundary cases of $c[j] = \pm 2$, we set $c'[j] \leftarrow c[j] \pm 2$.

Step 4 The audio hash h_a is computed from the audio stream. There are various audio fingerprinting algorithms such as [34, 35]. For experiment purposes, Evan et al.'s work [35], which is based on a feature vector from bark scale frequency spectrum, is utilized. It is also recommended that the output audio hash vector (denoted by h_a) to be encrypted, i.e.,

$$h_a = E(H_{audio}(A), k), \quad (2)$$

where A is decoded audio data.

Step 5 The audio and video streams are multiplexed to form the MP4 file. The hash value h_a is embedded into the synchronization information of the resulting MP4 file. Recall that the sample timing information resides in the *stts* box of MP4 for every media track, irregardless of audio or video (see Section 2.1). The *stts* data determines the number of consecutive samples (denoted by S) to be displayed/played for duration (denoted by δ) in *timescale* units, where δ is also determined in *stts*. S and δ each takes 4 bytes of storage space. We exploit the space of δ to embed h_a by specifying $S = 0$ explicitly. Note that zero sample count (i.e., $S = 0$) does not contribute to any sample for the actual audio-video synchronization. Specifically, we inject the neutral entries of $(S, \delta) = (0, h_a^x)$ for $x \in \{0, \ldots, \text{length}(h_a)\}$.

For experiment purpose, h_a is embedded in the *stts* of the first track (i.e., video track). Fig. 5 illustrates the process of embedding h_a into the *stts* data. Note that since the *stts* box is enlarged, the *stco* box needs to be updated accordingly to ensure correct positioning of the media data. This is crucial for maintaining format-compliant with respect to the MP4 format.

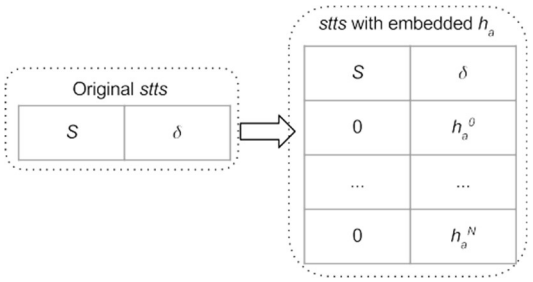

Fig. 5. Hiding h_a in the *stts* box.

Fig. 6 depicts the verification process. Specifically, there are two main parts, namely: (i) extracting h^l_a and h^l_v, and; (ii) computing h_a and h_v. Given a MP4 file, first, we extract and decrypt h^l_a from the *stts* box and de-multiplex MP4 to separate the video and audio streams. Next, we extract and decrypt h^l_v from the audio stream. Then, we compute h_a and h_v as described in section 4. If $h^l_a = h_a$ and $h^l_v = h_v$, the authentication is 100% successful. However, it is more practical to consider the hamming distance [36] to compare the hashes so that we can estimate how much tampering has occurred, as well as to localize the tampered region. We can set an acceptance threshold (e.g., 0.3) and say if the hamming distance is below the threshold, the authentication is successful, or failed otherwise.

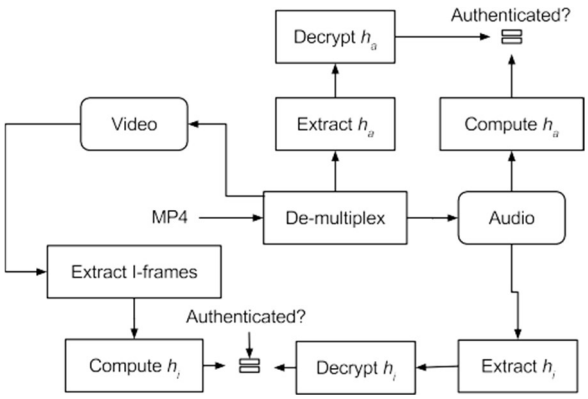

Fig. 6. Block diagram for verification process.

5.0 RESULTS

To verify the feasibility of the proposed system, we implemented the proposed system by utilizing pHash [35] (the opensource perceptual hash library) and Farunhofer FDK AAC Codec Library (ver. 0.1.3, FDK AAC extracted from Android) [37]. In addition, we also utilized tools: MP4Box (i.e., a tool from GPAC [38]) to multiplex audio and video streams in MP4 files and FFMPEG [39] to extract I-frames. The class structure and functions (including *AnalyseFile, parseAtoms, findAtom* and *adjustSampleOffsets* from [40]) are adopted to modify the *stts* and *stco* boxes. The proposed system was implemented in Python (ver. 2.7.10).

We downloaded various video clips from YouTube for experiment purposes. The videos are trimmed into short clips with duration of approximately 10 seconds each. Fig. 7 shows the representative samples of extracted I-frames. On the other hand, the original audio has stereo channel and it is sampled at 44100 kHz. However, we down sampled it to 8000 kHz followed by the conversion to mono channel due to the limitation of pHash [35].

Experiment results show that the proposed system is robust against the transcoding of video. Specifically, authentication can be performed even after the video has been processed, as long as the content of the video is preserved. However, audio stream is fragile to any type of signal processing attacks. It is due to the utilization of fragile hiding algorithm (i.e., LSB substitution). Nevertheless, if the audio hash h^l_a is not removed from the system layer, we still can authenticate the audio stream by comparing h^l_a and h_a, although audio stream is being attacked (i.e., transcoded and processed in some form). It is verified that if no attack is performed, the MP4 file is successfully authenticated. Therefore, we conclude that the proposed authentication is robust to noise to some extent and hence it can be deployed for actual application.

(a) [41] (b) [42] (c) [43] (d) [44] (e) [45]

(f) [46] (g) [47] (h) [48] (i) [49] (j) [50]

Fig. 7. Representative I-frames from the test movies.

Furthermore, the proposed system can also determine how much of the audio and video streams have been tampered by calculating the hamming distances. In addition, since the perceptual hashes are computed and embedded sequentially, the proposed system can also localize the tampered region based on the position of the perceptual hash. For example, suppose there are 5 hashes for a very short video with 5 I-frames. Assume that the distance between the third hash value and the corresponding hash value extracted from the audio stream is greater than the accepted threshold, then we can say the region of the third I-frame is modified. Similarly, the audio hashes are computed sequentially, and hence the tampered audio samples can also be localized accordingly.

6.0 ANALYSIS

6.1 Various Perceptual Hashing Algorithms

There are various algorithms for perceptual hashing. For image, there are DCT based hash [31], Marr-Hildreth operator based hash [51], radial variance based hash [52], block mean value based hash [53], etc. On the other hand, for audio, there are bark scale frequency based hash [35], audio hash [34] based on Fourier Transform, etc. We did not perform any benchmarking on hashing algorithms, but for experiment purposes, we utilize the publicly available opensource hashing library [35]. It is advised that suitable hashing algorithms should wisely be chosen for computational efficiency in the context of the given applications.

6.2 Audio Quality

We observed the objective measure of the audio quality before and after processing the original MP4 file. The summary of the results is presented in Table 1. Signal-to-Noise Ratio (SNR) is considered to compute the distortion caused by processing MP4 files (i.e., insertion of authentication values). Each test audio quality is quantified by calculating SNR before and after inserting the authentication values. The maximum change in SNR (less than 0.098 dB (i.e., 3.392%)) was observed suggesting the distortion caused by data insertion is insignificant.

6.3 Audio-video Synchronization

The proposed system inserts authentication values to the *stts* box by modifying synchronization information (i.e., sample timing information) in the MP4 file. However, the data inserted by the proposed system does not affect the audio-video synchronization because sample count, S does not carry any value (i.e., $S = 0$). Therefore, the proposed system does not produce any synchronization error.

6.4 File Size Increment

File size increment may occur due to the LSB substitution method in use. Specifically, when handling the border cases (i.e., coefficient with value +2 or -2), in order to encode the value '0', the magnitude is increased to 4, because during decoding, only coefficients with magnitude greater 2 are considered. Therefore, this process will affect the selection process of optimized Huffman code word. However, if we limit the choices of the host coefficients (e.g., consider the biggest 256 coefficients), there will be no increase in file size. On the other hand, file size increment is inevitable when embedding audio hash h_a into the system layer (i.e., *stts* box).

Table 1. Audio Quality Analysis

Test Audio	Original SNR [dB]	Processed SNR [dB]	Changes [%]
test0	-2.8599	-2.9275	2.364
test1	-2.9460	-2.9889	1.456
test2	-2.9875	-3.0103	0.763
test3	-2.8804	-2.9781	3.392
test4	-3.6002	-3.6111	0.303
test5	-2.7285	-2.7523	0.872
test6	-3.2078	-3.2685	1.892
test7	-3.0068	-3.0255	0.622
test8	-2.9029	-2.9457	1.474
test9	-3.0530	-3.0728	0.649

Table 2 presents the summary the file size of the test movie clips before and after inserting the perceptual hashes. Note that sample count S and δ takes 4 bytes each and thus, each *stts* entry requires 8 bytes. Therefore, there is an increment of 8 bytes for every *stts* entry inserted. Throughout the experiments, file size increment occurred in the test movie "test4" is the largest (i.e., 1.136%). However, the movie clips are usually large in size and the observed increment for each clip is negligible.

Table 2. File Size Analysis

Test Clip	*stts* entries	Original Size [byte]	New Size [byte]	Changes [%]
test0	145	122459	123619	0.947
test1	594	620381	625133	0.766
test2	594	784630	789382	0.606
test3	594	552829	557581	0.860
test4	594	418259	423011	1.136
test5	594	463499	468251	1.025
test6	594	944675	949427	0.503
test7	594	513342	518094	0.926
test8	594	621692	626444	0.764
test9	594	896961	901713	0.530

Table 3. Comparison of Authentication Methods

Authentication Scheme	Quality Degradation		File Size Increment			Applied to	Authentication Methods
	Audio	Video	System	Audio	Video	System		
Jake [30]							MP4 Structure	Metadata Identification
Previous [5]	✓	✓		✓	✓	✓	Audio, Video, Synchronization	Data Insertion, Cryptography
Proposed	✓			✓		✓	Audio, Video, Synchronization	Data Insertion, Cryptography, Perceptual Hash

7.0 COMPARISON WITH CONVENTIONAL METHODS

To the best of our knowledge, there are only two authentication systems specifically designed for MP4. The first one [30] exploits the MP4 structure and metadata. It does not directly authenticate audio and video streams, but it is solely meant for determining whether the MP4 is generated by genuine recording device. Since Jake's method does not manipulate any part of the media stream, there is neither quality degradation nor file size increment. On the other hand, the second method is our previous work [5], where data insertion and cryptographic hash function are utilized jointly. [5] causes quality degradation and a slight increment in file size. In the current work, the proposed method maintains the quality of the video stream, and the quality of the audio stream is slightly degraded (< 0.98 dB).

Furthermore, Jake's method is very fragile to any sort of re-multiplexing, and our pervious work is also sensitive to any changes in bit due to the utilization of one way cryptographic hash function. In the proposed authentication scheme, the deployment of perceptual hash allows us to focus on the authentication perceived information instead of a perfect match, and hence the proposed authentication scheme is robust against compression. In other words, the proposed system can authenticate a movie as long as we can compute the perceptual hashes from the video stream. To sum up, the proposed authentication system overcomes video quality degradation and is robust against re-compression or video transcoding. Table 3 summarizes the differences between the proposed and the existing authentication methods for MP4.

8.0 CONCLUSION

In this work, an authentication scheme for MP4 is proposed by utilizing perceptual hash and data insertion. Specifically, the perceptual hash of an I-frame in the video stream is encrypted and inserted into the quantized spectral coefficients of the audio stream. Then, the audio hash vector is encrypted and inserted into the synchronization information in the MP4 container. Experiment results showed that our proposed system is able to authenticate audio-visual content (i.e., MP4 file). The proposed system leads to insignificant quality degradation, i.e., $< 2.3\%$ for audio, and negligible file size increment, i.e., $\leq 3.4\%$ for *stts* data. As future work, we shall design robust movie (video + audio) hash derived from video and audio streams.

REFERENCES

[1] M. Wu and B. Liu, "Watermarking for image authentication," in *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*, vol. 2, pp. 437–441, IEEE, 1998.

[2] M. M. Yeung and F. Mintzer, "An invisible watermarking technique for image verification," in *Image Pro cessing, 1997. Proceedings., International Conference on*, vol. 2, pp. 680–683, IEEE, 1997.

[3] Y. Yoshitomi, T. Asada, Y. Kinugawa, M. Tabuse, *et al.*, "An authentication method for digital audio using a discrete wavelet transform," *Journal of Information Security*, vol. 2, no. 02, p. 59, 2011.

[4] Y. Tew, K. Wong, and R. C.-W. Phan, "Hevc video authentication using data embedding technique," in *Image Processing (ICIP), 2015 IEEE International Conference on*, pp. 1265–1269, IEEE, 2015.

[5] I. MaungMaung, Y. Tew, and K. Wong, "Authentication of mp4 file by joint data embedding in audio and video tracks," in *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2016 Asia-Pacific*, pp. 1–6, IEEE, 2016.

[6] L. Alfred, "50 video marketing stats to help you create a winning social media strategy in 2017." https://blog.bufferapp.com/social-media-video-marketing-statistics, 2017. [Online; accessed 24-May-2018].

[7] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2face: Real-time face capture and reenactment of rgb videos," in *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2016.

[8] ISO, "Information technology - coding of audio-visual objects - part 14: Mp4 file format," ISO 14496-14, International Organization for Standardization, Geneva, Switzerland, 2003.

[9] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the h. 264/avc video coding standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, no. 7, pp. 560–576, 2003.

[10] ISO, "Generic coding of moving pictures and associated audio information"-part 7: Advanced audio coding (aac)," ISO 13818-7, International Organization for Standardization, Geneva, Switzerland, 2006.

[11] K. Brandenburg, "Mp3 and aac explained," in *Audio Engineering Society Conference: 17th International Conference: High-Quality Audio Coding*, Audio Engineering Society, 1999.

[12] M. Bosi, K. Brandenburg, S. Quackenbush, L. Fielder, K. Akagiri, H. Fuchs, and M. Dietz, "Iso/iec mpeg-2 advanced audio coding," *Journal of the Audio engineering society*, vol. 45, no. 10, pp. 789–814, 1997.

[13] M. Bosi and R. E. Goldberg, *Introduction to digital audio coding and standards*, vol. 721. Springer Science & Business Media, 2012.

[14] M.-. Committee *et al.*, "Generic coding of moving pictures and associated audio information: Video," *ISO/IEC*, 2000.

[15] W. contributors, "Shazam (company) — wikipedia, the free encyclopedia," 2017. [Online; accessed 29- December-2017].

[16] Y. Tew and K. Wong, "An overview of information hiding in h. 264/avc compressed video," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 24, no. 2, pp. 305–319, 2014.

[17] S. Cheng, H. Yu, and Z. Xiong, "Error concealment of mpeg-2 aac audio using modulo watermarks," in *Circuits and Systems, 2002. ISCAS 2002. IEEE International Symposium on*, vol. 2, pp. II–261, IEEE, 2002.

[18] S. Cheng, H. Yu, and Z. Xiong, "Enhanced spread spectrum watermarking of mpeg-2 aac audio," in *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, vol. 4, pp. IV–3728, IEEE, 2002.

[19] Y. Wang, L. Guo, Y. Wei, and C. Wang, "A steganography method for aac audio based on escape sequences," in *Multimedia Information Networking and Security (MINES), 2010 International Conference on*, pp. 841–845, IEEE, 2010.

[20] J. Zhu, R. Wang, and D. Yan, "The sign bits of huffman codeword-based steganography for aac audio," in *2010 International Conference on Multimedia Technology*, 2010.

[21] J. Zhu, R.-D. Wang, J. Li, and D.-Q. Yan, "A huffman coding section-based steganography for aac audio," *Information Technology Journal*, vol. 10, no. 10, pp. 1983–1988, 2011.

[22] C. Oliboni, "Openpuff v4.00 steganography and watermarking," Jul 2012.

[23] T. Foundation, "Truecrypt."

[24] I. MaungMaung, K. Wong, and K. Tanaka, "Reversible data hiding methods based on audio and video synchronization in mp4 container," in *Intelligent Signal Processing and Communication Systems (ISPACS), 2016 International Symposium on*, pp. 1–6, IEEE, 2016.

[25] Z. Ali, M. Imran, and M. Alsulaiman, "An automatic digital audio authentication/forensics system," *IEEE Access*, vol. 5, pp. 2994–3007, 2017.

[26] J. Ouyang, Y. Liu, and H. Shu, "Robust hashing for image authentication using sift feature and quaternion zernike moments," *Multimedia Tools and Applications*, vol. 76, no. 2, pp. 2609–2626, 2017.

[27] S. Sadeghi, H. A. Jalab, K. Wong, D. Uliyan, and S. Dadkhah, "Keypoint based authentication and localization of copy-move forgery in digital image," *Malaysian Journal of Computer Science*, vol. 30, no. 2, pp. 117–133, 2017.

[28] F. Khelifi and A. Bouridane, "Perceptual video hashing for content identification and authentication," *IEEE Transactions on Circuits and Systems for Video Technology*, 2017.

[29] Z. Sakr and N. D. Georganas, "Robust content-based mpeg-4 xmt scene structure authentication and multimedia content location," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 3, no. 3, p. 18, 2007.

[30] J. Hall, "Mp4 video authentication using file structure and metadata," in *The Digital Forensic Research Conference*, 2015.

[31] B. Coskun and B. Sankur, "Robust video hash extraction," in *Signal Processing Conference, 2004 12th European*, pp. 2295–2298, IEEE, 2004.

[32] ISO, "Information technology - Security techniques - Encryption algorithm - Part 3: Block ciphers," ISO 18033-3, International Organization for Standardization, Geneva, Switzerland, 2010.

[33] N. F. Pub, "197: Advanced encryption standard (aes)," *Federal Information Processing Standards Publication*, vol. 197, pp. 441–0311, 2001.

[34] J. Haitsma and T. Kalker, "A highly robust audio fingerprinting system.," in *ISMIR*, vol. 2002, pp. 107–115, 2002.

[35] K. Evan and S. David, "phash." http://www.phash.org/, 2010.

[36] R. W. Hamming, "Error detecting and error correcting codes," *Bell Labs Technical Journal*, vol. 29, no. 2, pp. 147–160, 1950.

[37] M. Storsjo, "A standalone library of the fraunhofer fdk aac code from android".

[38] T. ParisTech, "Gpac multimedia open source project," 2003.

[39] W. contributors, "Ffmpeg — wikipedia, the free encyclopedia," 2017. [Online; accessed 29-December-2017].

[40] M. Fiedler, "Real steganography with truecrypt." http://keyj.emphy.de/real-steganography-with-truecrypt, Feb 2011.

[41] M. Archive, "Elon musk incredible speech - motivational video by mulliganbrothers." https://www.youtube.com/watch?v=QygpaIJclm4, 2016.

[42] B. News, "Why some catalans want independence... and some don't - bbc news." https://www.youtube.com/watch?v=CGYzH8nEZr0, 2017.

[43] B. News, "Uber loses london operating licence." https://www.youtube.com/watch?v=Z4QcwsEbKyo, 2017.

[44] B. News, "World's oldest giant panda dies aged 37." https://www.youtube.com/watch?v=yiOw_gWdmeg, 2017.

[45] B. News, "Rocket launch in 360 video." https://www.youtube.com/watch?v=FArI sJkSOA, 2015.

[46] B. News, "Moment leicester city became premier league champions (360 video)." https://www.youtube.com/watch?v=GPlYgzatFBk, 2016.

[47] B. News, "Why are people paying $6 for a bag of human waste?." https://www.youtube.com/watch?v=yrAOH3GwUxg, 2017.

[48] B. News, "Children interrupt bbc news interview." https://www.youtube.com/watch?v=Mh4f9AYRCZY, 2017.

[49] B. News, "Parsons green: Ambulance crews took 18 people to hospital." https://www.youtube.com/watch?v=35yy0MrUAyg, 2017.

[50] B. News, "Meet japan's batman - chibatman a real life dark knight." https://www.youtube.com/watch?v=2CzHFhZFk3g, 2014.

[51] S. Bhattacharjee and M. Kutter, "Compression tolerant image authentication," in *Proceedings 1998 International Conference on Image Processing. ICIP98 (Cat. No.98CB36269)*, vol. 1, pp. 435–439 vol.1, Oct 1998.

[52] F. Lefebvre, B. Macq, and J.-D. Legat, "Rash: Radon soft hash algorithm," in *Signal Processing Conference, 2002 11th European*, pp. 1–4, IEEE, 2002.

[53] B. Yang, F. Gu, and X. Niu, "Block mean value based image perceptual hashing," in *Intelligent Information Hiding and Multimedia Signal Processing, 2006. IIH-MSP'06. International Conference on*, pp. 167–172, IEEE, 2006.