IEICE TRANS. INF. & SYST., VOL.E104–D, NO.1 JANUARY 2021

LETTER

Practical Video Authentication Scheme to Analyze Software Characteristics*

Wan Yeon LEE[†a)], ***Member***

SUMMARY We propose a video authentication scheme to verify whether a given video file is recorded by a camera device or touched by a video editing tool. The proposed scheme prepares software characteristics of camera devices and video editing tools in advance, and compares them with the metadata of the given video file. Through practical implementation, we show that the proposed scheme has benefits of fast analysis time, high accuracy and full automation.

key words: *video integrity, authentication, software characteristics, video editing tool, surveillance camera*

1. Introduction

As video editing tools such as Movi Maker and Sony Vegas are widely used, it is easy to modify the video contents recorded by surveillance digital cameras. Thus the procedure to analyze the integrity of video contents becomes essential when adopting the video contents as evidences. Most of related video digital forensic studies [1], [2] focused on detecting abnormal patterns of tampered multimedia contents when there is no built-in integrity protection scheme such as invisible watermark embedding. They analyze sensor noise artifacts, coding artifacts or material object features in multimedia contents. However, they suffer from long analysis time, imprecise analysis results, and non-full automation. These approaches may require a lot of time due to analysis of huge still images extracted from a video with long running time [3], detect only frame deletion/insertion forgery but not minute image modification forgery [4], or depend on manual postprocessing of human experts after digital filter preprocessing of multimedia contents [5].

The proposed scheme analyzes the characteristics of softwares used to generate the video file, instead of analyzing the multimedia contents. The scheme prepares the software characteristics of as many as surveillance camera devices and video editing tools as possible in advance. Next the scheme extracts the metadata of the given video file and compares them with the prepared software characteristics. If the extracted metadata are matched with software characteristics of some camera device, the scheme determines that the given video file has *integrity-guaranteed* multimedia contents. On the other hand, if the extracted metadata are matched with software characteristics of some video editing tool, the scheme determines that the given video file has *integrity-unguaranteed* multimedia contents. We implement the proposed scheme into a practical software tool and confirm that the scheme has benefits of fast analysis time, high accuracy and full automation. The scheme takes a few seconds regardless of video running time, detects all forged videos on 100% accuracy, and does not require any manual burden of human experts.

Some recent studies [6], [7] considered software characteristics of video editing tools and/or mobile phones. They introduced the potential capability of the video authentication approach utilizing the software characteristics, but did not handle operational constraints for its feasibility and the maximization of its analyzable coverage. Our study introduces and solves inevitable issues for *realistic* feasibility of this approach: how to extract software characteristics of video editing tools against almost infinite number of different trial cases, how to efficiently extend the software characteristics for new editing tools or new camera devices, and presence of conflicting analysis between software characteristics of video editing tools and camera devices. We also evaluate the realistic feasibility of this approach through implementation with prevalently used commercial editing tools and surveillance cameras. Furthermore, we figure out the inherent limitations of this approach and introduce their solution approaches.

2. Analysis of Software Characteristics

Video editing tools first decompress the compressed multimedia data of an input video file with a codec software. Next after modifying the decompressed multimedia data, they recompress the multimedia data with its codec software and store them into a video file. On the contrary, surveillance camera devices directly compress input multimedia data with a codec software and store them into a video file. When storing the compressed multimedia data into a video file, the compression software and the employed codec parameters determine the metadata of the output video file [8].

We examines the metadata of output video files generated by the most prevalent 25 video editing softwares ** in

Manuscript received June 11, 2020.
Manuscript revised August 15, 2020.
Manuscript publicized September 30, 2020.

[†]The author is with Dongduk Women's University, S. Korea.

*This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2018R1D1A1B07045806).

a) E-mail: wanlee@dongduk.ac.kr

DOI: 10.1587/transinf.2020EDL8081

**Premiere, Final Cut, Vegas, iMovie, Avid Media Composer, Edius, Powerdirector, Lightworks, Pinnacle Studio, Movie Maker, Movavi Video Editor, Gilisoft Video Editor, Adobe Photoshop, After Effect, Avidemux, Bandicut, VapMix, GomMix, VirtualDub, Free Video Editor, YAMB, FFmpeg, VivaVideo App., Quick App., and Kakao Talk.

Copyright © 2021 The Institute of Electronics, Information and Communication Engineers

the worldwide. In this examination, only AVI, MP4 (same to 3GP and K3G) and MOV formats are considered because the other formats are very rarely used in surveillance cameras. Oridinary metadata of video files cannot be used for the software signatures due to their common usage. Thus we first check which factor affects the discriminative metadata of output video files because there are almost infinite number of trial cases for combinations of many media modification functions, codec types, codec parameters, various file formats of input/output files, different sizes of input/output files, software release versions, etc. There are possibly millions of combinations just for the change of media modification functions. With experiments of major media modification functions, we check that the change of media modification functions except the audio removal function does not affect the discriminative metadata of output video files. Also the input file format, the input file size, the output file size in the MP4 and MOV format, and the minor version of software release do not affect the discriminative metadata of output video files. On the other hand, the codec type, the codec parameters, the output file format, the output file size in the AVI format and the major version of software release affect the discriminative metadata of output video files.

We also examines the metadata of output video files recorded by 58 dashboard cameras, one CCTV, and 24 smartphones. Only the audio on/off function and hardware release models affect the metadata of output video files. The resolution setting and the firmware update do not affect the metadata of output video files. The metadata of compression software of camera devices are relatively simpler than those of recompression software of video editing software. This is because the software of video editing tools requires compatibility with various codec applied to input video compression for the decompression operation while the software of camera devices requires compactness.

When extracting the software characteristics for another (or a newly emerging) video editing tool and camera device, our analysis study of software characteristic helps to reduce significantly the burden of finding its signature. For video editing tools, it does not need to artificially prepare and examine video samples with the change of media modification functions except the audio removal function, input file formats, input file sizes, output file sizes in MP4 and MOV format, and minor versions of software release. For camera devices, it does not need to artificially prepare and examine video samples with the change of resolution settings and firmware updates.

3. Proposed Scheme

The proposed scheme prepares in advance two signature databases for the video editing tool and the surveillance camera device respectively, as described in Sect. 2. Given a target video file, the scheme extracts the metadata and compares them with signatures in the two databases. If any signature in the database of cameras devices is matched, the scheme determines that the video integrity of the target file is guaranteed. If any signature in the database of video editing tools is matched, the scheme determines that the video integrity of the target file is not guaranteed and possibly violated. When there is no matched signature in the two databases, the scheme does not provide any analysis output. Figure 1 shows the overall operation procedure of the proposed scheme.

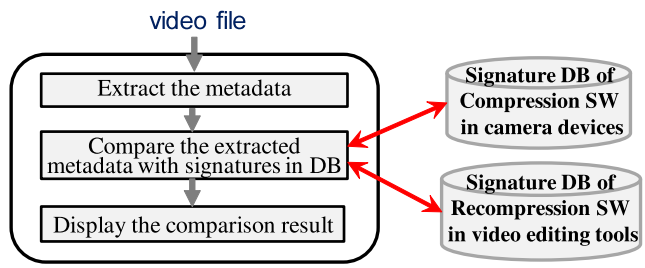


Fig. 1 Overall procedure of the proposed scheme

The integrity analysis accuracy is dominated by the signatures in the two databases prepared in advance. Among the metadata of video files, the field name sequence of their file structure and the values in some selected fields are recorded as their signature. Figure 2 shows a signature example of Movavi Video Editor in the database of editing tools, which is a simplified version for the sake of readability. The signature values are structuredly recorded according to the JSON format, which is arbitrarily employed because of its structural scalability and easy implementation. In this figure, the area "item_name" stores the name of the examined editing tool (or the device name in the database of camera devices). The area "id_seq" stores the field name sequence where each field name is separated. The area "check_value" stores the values of selected fields, where the "meta" field is selected among the fields in the area "id_seq" and the word "mdtaencoder" is picked out among the values in the "meta" field. We select empirically the field of "check_value", for example "meta" in MP4 format and "ISFT" in AVI format because these fields contain commonly the used codec name or software/device title. We pick out manually the codec name or software/device title among all values in the selected "check_value" field, which is our unsolved obstacle against the full automation of the signature generation procedure.

In the example of Fig. 2, there are two pairs of signature values obtained from metadata examinations with various cases of codec types, codec parameters, etc. The first signature values are denoted with a solid-line box and the second signature values are denoted with a broken-line box. If any pair of these two signature values is matched, the proposed scheme determines that the target video is made by the editing tool of Movavi Video Editor. In most cases, there are multiple pairs of signature values for each editing tool or camera device. The proposed scheme compares all pairs of signature values for each editing tool and camera device, and displays all matched editing tools and camera devices.

We implemented the proposed scheme into a software

```
{"item_name": "Movavi Video Editor",
 "id_seq":
    [ ["ftyp", "free", "mdat", "moov", "mvhd",
       "trak", "tkhd", "edts", "elst", "mdia", "mdhd", "hdlr", "minf", "vmhd",
       "dinf", "dref", "stbl", "stsd", "stts", "stss", "stsc", "stsz", "stco",
       "trak", "tkhd", "edts", "elst", "mdia", "mdhd", "hdlr", "minf", "smhd",
       "dinf", "dref", "stbl", "stsd", "stts", "stsc", "stsz", "stco",
       "sgpd", "sbgp", "udta", "meta" ],
      ["ftyp", "free", "mdat", "moov", "mvhd",
       "trak", "tkhd", "edts", "elst", "mdia", "mdhd", "hdlr", "minf", "vmhd",
       "dinf", "dref", "stbl", "stsd", "stts", "stss", "stsc", "stsz", "stco",
       "sgpd", "sbgp", "udta", "meta" ]
    ]
 "check_value":
    [ { "id": "meta", "value": "mdtaencoder" },
      { "id": "meta", "value": "mdtaencoder" }
    ]
}
```

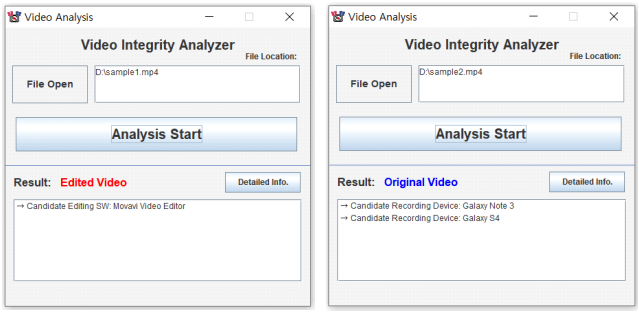
Fig. 2 Signature example for video editing tools

tool with JAVA programming on Eclipse IDE †. Figure 3 shows the main graphic user interface of the implemented tool. The button "File Open" activates a function to read the target video file. The button "Analysis Start" activates a function to analyze its video integrity. If any signature in the database of video editing tools is matched, it displays the result message "Edited Video" as shown in Fig. 3(a). Specific name of the matched video editing tool is also displayed below the result message. If any signature in the database of camera devices is matched, it displays the result message "Original Video" as shown in Fig. 3(b), where there are two matched camera devices. When there is no matched signature in the two databases, it displays the result message "No Decision". If there are overlapped signature matches in both databases of video editing tools and camera devices, it displays the result message "Analysis Error", which never occurs in our experiments. This tool operates automatically just with two button clicks.

We tested about 100 video files recorded by 83 different surveillance cameras and about 900 video files generated artificially by 25 video editing tools. The tested video files are separated and different from the training video files examined in the signature generation. In this evaluation, the implemented tool shows 100% analysis accuracy, and there is no overlapped match in both signature databases of video editing tools and camera devices. In some cases of video files recorded by camera devices, there are two or more names of matched camera devices. This tool takes less than one second in most cases and about four seconds in the worst case for the single video analysis.

The button "Detailed Info." in Fig. 3 activates a function to show detailed metadata (i.e., file structure) of the target video file, such as the video format, the list of field names, and the values in each field. Figure 4 shows the detailed metadata information of the example video file applied to Fig. 3(a). The upper red box shows the field name

†This tool can be found at https://drive.google.com/drive/folders/1RoOfpb0ad7zfhV2qS8xkDrEEetQvlwu2?usp=sharing



(a) (b)

Fig. 3 Working examples of the implemented tool: (a) analysis result of edited video and (b) analysis result of original video

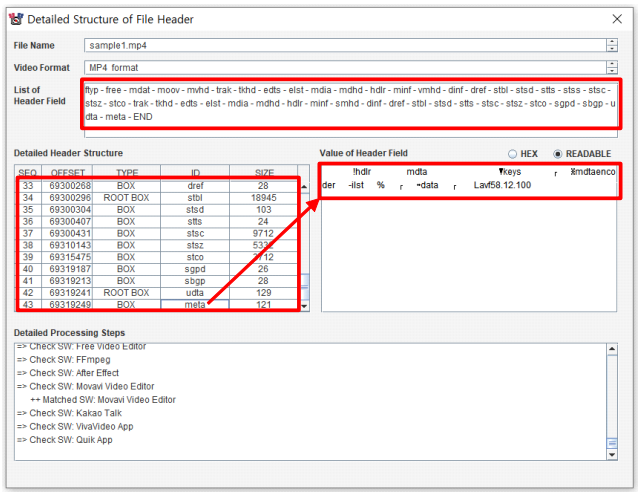


Fig. 4 Extraction of detailed metadata including field name sequence and values of each field

sequence used in our signature. The lower left box shows the detailed information of each field and the lower right box shows the stored values in each field. Whenever a field in the lower left box is clicked, its corresponding values stored in that field are displayed in the lower right box, where the "meta" field is clicked in Fig. 4. With the automatic metadata extraction of this tool, we prepare the signatures of video editing tools and video camera devices by copying the list of all field names (in the upper red box) and discriminative values in selected fields (in the lower right box) into our JSON format explained in Fig. 2. It is very complex to find the discriminative fields containing the model name or the used encoder name and pick out a meaningful substring in those fields especially for video editing tools with a lot of different trial cases, which is left for our future work. Note that the related previous studies [6], [7] did not support the automatic extraction of all field values, whereas our tool supports it. Also they did not address how to efficiently extract software characteristics of video editing tools against almost infinite number of different trial cases.

The limitation of the proposed scheme is unable to detect the integrity of video files generated by non-registered editing tools or camera devices, where the result message

"No Decision" is displayed. But the analysis capability of the proposed scheme can be enhanced progressively by expanding the signature database of more video editing tools or camera devices. It is worthy to mention that expanding the signature database of camera devices is much more efficient than that of video editing tools because the signature analysis of camera devices requires the metadata examinations of at most two video samples while that of video editing tools requires the metadata examinations of a lot of video samples, as explained in Sect. 2. If the signature of some camera device is already registered, all output videos made by the same model of the camera device can be analyzed even when there is no registered signature exactly matched. In that case, "No Decision" means "Edited Video" derived from some non-registered editing tool. All registered video editing tools and camera devices are displayed in the bottom text box in Fig. 4. With the support of our implemented tools, it is not difficult to find the signature of a new camera device. For two input sample videos made with each mode of audio-on and audio-off, we extract the field name sequence and copy it into the "id_seq" area shown in Fig. 2. Next we test the analysis accuracy of the current signature with support of our tool shown in Fig. 3. If the current signature does not result in enough accuracy, we need to find the discriminative field value with support of our tool shown in Fig. 4. We check the values of each field and search for the device model name or the encoder name. We choose an appropriate field value and copy it into the "check_value" area shown in Fig. 2. We repeat this procedure until the modified signature shows enough analysis accuracy. For MP4 or MOV video files, the signature made only with the field name sequence shows enough analysis accuracy but the discriminative field values reduce the number of candidate devices, shown in the bottom text box of Fig. 3(b). In contrast, for AVI video files, the discriminative field values as well as the field name sequence are essential for enough analysis accuracy due to short length of their field name sequence.

4. Conclusions and Discussion

To verify the integrity of video contents, the proposed scheme exploits software characteristics of both video editing tools and camera devices, instead of analyzing their multimedia contents. The scheme has benefits of fast analysis time, high accuracy and full automation, compared to related previous methods analyzing multimedia contents of video files. We investigate the issues how to extract signatures of video editing tools against almost infinite number of different trial cases, and how to efficiently maximize the detectable coverage with expanded signatures. We also show non-existence of conflicting analysis between software characteristics of editing tools and camera devices through the implementation with numerous commercial editing tools and surveillance camera.

Due to emerging brand-new camera devices and video editing tools, it is intrinsically impossible to prepare the signatures of all camera devices and editing tools in advance. As a solution to this limitation, the nearest signature among the registered signatures can be used instead of the exactly matched signature. In this case, the analysis result may have a form of probability calculated with the proximity to the nearest signature, which is left for our future work.

If the registered signatures of the proposed scheme are disclosed, the scheme becomes vulnerable to signature forgery of camera devices, similar to fingerprint forgery of human authentication. The signature forgery attack requires a new software implementation which enforces the metadata of its output file to be matched with any registered signature in our databases. To defend this signature forgery attack, the scheme needs to be combined with existing video authentication mechanisms [3]–[5]. When the scheme determines the target video as "Original Video", the scheme triggers an additional analysis procedure performed by the existing video authentication mechanism that analyze multimedia contents instead of software characteristics. On the other hand, when the scheme determines the target video file as "Edited Video", it is the final analysis result because there is no motivation for signature forgery of video editing tools.

References

[1] K. Sitara and B.M. Mehtre, "Digital video tampering detection: An overview of passive techniques," Digital Investigation, vol.18, pp.8–22, Sept. 2016.

[2] R.D. Singh and N. Aggarwal, "Video content authentication techniques: A comprehensive survey," Multimedia Systems, vol.24, no.2, pp.211–240, March 2018.

[3] L. Su, C. Li, Y. Lai, and J. Yang, "A fast forgery detection algorithm based on exponential-fourier moments for video region duplication," IEEE Trans. Multimedia, vol.20, no.4, pp.825–840, April 2018.

[4] Z. Zhang, J. Hou, Q. Ma, and Z. Li, "Efficient video frame insertion and deletion detection based on inconsistency of correlations between local binary pattern coded frames," Security and Communication Networks, vol.8, no.2, pp.311–320, Jan. 2015.

[5] M. Zampoglour, F. Markatopoulou, G. Mercier, D. Touska, E. Apostolidis, S. Papadopoulos, R. Cozien, I. Patras, V. Mezaris, and I. Kompatsiaris, "Detecting tampered videos with multimedia forensics and deep learning," Int. Conf. Multimedia Modeling (MMM), pp.374–386, Springer, Jan. 2019.

[6] T. Gloe, A. Fisher, and M. Kirchner, "Forensic analysis of video file formats," Digital Investigation, vol.11, no.1, pp.S68–S76, May 2014.

[7] J. Song, K. Lee, W.Y. Lee, and H. Lee, "Integrity verification of the ordered data structures in manipulated video content," Digital Investigation, vol.18, no.C, pp.1–7, Sept. 2016.

[8] B. Carrier, File System Forensic Analysis, Addison-Wesley, 2005.