A.J. Tallón-Ballesteros (Ed.)

doi:10.3233/FAIA210284

Chained Digital Signature for the Improved Video Integrity Verification

Linju LAWRENCE [a,1] and R SHREELEKSHMI [b]
[a] *Department of Computer Science and Engineering, College of Engineering, Trivandrum (Affiliated to APJ Abdul Kalam Technological University), Thiruvananthapuram - 695016, Kerala, India*
[b] *Department of Information Technology, Government Engineering College, Barton Hill (Affiliated to APJ Abdul Kalam Technological University), Thiruvananthapuram - 695035, Kerala, India*

Abstract. The recorded videos from the surveillance cameras can be used as potential evidence in forensic applications. These videos can be easily manipulated or tampered with video editing tools without leaving visible clues. Hence integrity verification is essential before using the videos as evidence. Existing methods mostly depend on the analysis of video data stream and video container for tampering detection. This paper discusses an active video integrity verification method using Elliptic Curve Cryptography and blockchain. The method uses Elliptic Curve Digital Signature Algorithm for calculating digital signature for video content and previous block. The digital signature of the encoded video segment (video content with predetermined size) and that of previous block are kept in each block to form an unbreakable chain. Our method does not consider any coding or compression artifacts of the video file and can be used on any video type and is tested on public-available standard videos with varying sizes and types. The proposed integrity verification scheme has better detection capabilities towards different types of alterations like insertion, copy-paste and deletion and can detect any type of forgery. This method is faster and more resistant to brute force and collision attacks in comparison to existing recent blockchain method.

Keywords. Blockchain, Elliptic curve digital signature, Hash function, Video integrity.

1. Introduction

A video record can be used as a primary source of evidence in digital forensics. However, the problem is that it can undergo various tampering attacks. In such a scenario, identifying authentic video from the forged one is a challenging task. So, the verification of integrity of the video is crucial before it can be used as evidence. Forgery detection in digital videos is classified into intra-frame tampering detection and inter-frame tampering detection. Inter-frame tampering detection includes frame- duplication/frame-deletion/frame-insertion detection and temporal interpolation deletion. Frame-insertion/duplication/deletion detection uses sensor, recompression, motion, brightness and pixel level features. Intra-frame tampering detection is categorized as upscale crop

[1] Corresponding Author: Department of Computer Science and Engineering, College of Engineering Trivandrum, Thiruvananthapuram – 695016, Kerala, India; E-mail: linjulawrence680@gmail.com.

detection and copy-move detection. This copy-move detection method considers pixel-similarity/object/motion features. Different features used for forgery detection in digital videos are sensor/camera artifacts and coding/motion/object features [1]. Coding artifacts include features related to the detection of double compression. If an attacker wants to modify a genuine video stored in compressed form, he should decode, edit and then recompress it. Frame deletion in HEVC (High Efficiency Video Coding) coded videos can be detected from the picture type changes [2]. Deletion of the frame results in change in type of frames. This change in type leads to irregularity in coding features of Coding Units/Prediction Units/Transform Units, which are considered as the processing units in HEVC type videos. These features are extracted and then by applying machine learning techniques such as Linear Discriminant Analysis and Multilayer perceptron method, we can classify video as forged or genuine. In [3], generalized variation in prediction footprint is used for detection of recompression and estimation of size of group of pictures. However, the method applies only to MPEG-4/MPEG-2/H.264 video coding standards.

Integrity of video file is verified by analyzing the structure and behavior of video containers generated by mobile devices shared through instant messaging applications, social network and editing software [4]. Using atom extraction method specific features are extracted and classification is done using machine learning approaches such as t-Distributed Stochastic Neighbor Embedding, Pearson correlation coefficient and Principal Component Analysis. In [5], tampering detection is done using unsupervised analysis making use of dissimilarity between original and processed containers. The disadvantage is that the method is applicable only to MP4 file format. In [6], video integrity verification based on blockchain framework is introduced. This method uses Elliptic Curve Integrity Encryption Scheme (ECIES) and Hash-based Message Authentication Code (HMAC) for verification of integrity of videos. Video segments are key-hashed and stored in a chain in chronological order. For verification hash value of the video segment is computed and compared with the hash in the blockchain. In [6], if the verifier himself is an attacker, then he can easily decrypt the key value leading to a collision attack.

2. Proposed Method

The video integrity verification method proposed uses the benefits of blockchain framework. This method does not consider the type of frames or coding parameters such as motion vectors and quantization parameters. Our method is applicable to any type of video and can detect any type of forgeries. Video clips recorded every few minutes are termed as video segments. Signature of a video segment is stored in the corresponding block and signature of the whole block is stored in next block. The blocks are chained in order in which videos are captured using Closed Circuit Television, Accident Data Recorder, etc. The overall structure of our method is shown in Figure 1. The private keys for both block signature and video segment signature are randomly generated. For generation of signatures Elliptic Curve Digital Signature Algorithm (ECDSA) [7] is used. ECDSA is faster than other digital signature algorithms based on integer factorization and discrete logarithm and the key length is shorter for providing the same level of security [8]. Section 2.1 details key generation, signature generation and integrity verification using ECDSA.

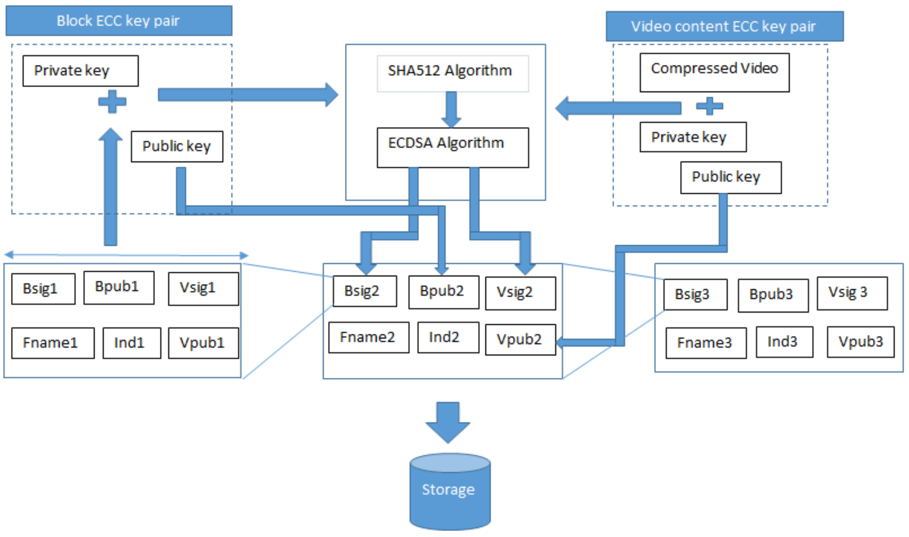

Figure 1. A schematic block diagram of proposed mechanism.

2.1. Digital Signature Calculation Using ECDSA

Consider an elliptic curve C having the equation $y^2 = x^3 + Ax + B$ defined over a finite field F_q such that $A,B \in F_q$. G generates the curve C and n is the prime modulus, such that the prime power $q = n^r$ where r is a large positive integer.

- **Key generation**: Let a be the private key in the interval [$1, n$-1]. Public key Q is computed by

$$Q = aG \tag{1}$$

- **Generation of signature**: Select a random value k in the interval $[1, n\text{-}1]$. Point P can be calculated using

$$P = kG \tag{2}$$

  P's x-coordinate represents R. Calculate S by

$$S = k^{-1}(Z + aR) mod n \tag{3}$$

  Z represents the hash value. Signature created is (R, S). If R or S is zero, repeat the generation of signature with different random number k.

- **Verification of signature:** The signature is invalid if R and S are values not in the interval [$1, n$-1]. The point P' is computed by

$$P' = S^{-1}ZG + S^{-1}RQ \tag{4}$$

If R and x-coordinate of point P' are equal, signature is valid. Otherwise signature is invalid. If $P' = 0$, signature is rejected.

2.2. Proposed Blockchain Generation

Each block includes signature of video segment, signature of previous block, public key of video segment signature, public key of previous block signature, path of the video file, and index of the block. The following process describes blockchain generation.

1. Compute the public keys Q_V and Q_B using Eq. (1) from corresponding private keys a_V and a_B respectively, where Q_V and a_V are the keys of the video content and Q_B and a_B are the keys of previous block.
2. Generate the random values k_V and k_B for the video segment and previous block respectively.
3. Generate the hash values Z_V and Z_B for video and previous block respectively using SHA-512 algorithm.
4. P_V and P_B are the points for the video segment and previous block respectively, which can be calculated by Eq. (2). R_V and R_B represent the x-coordinates of P_V and P_B respectively.
5. Calculate S_V and S_B corresponding to video content and previous block using Eq. (3). (R_V, S_V) is the signature of video segment, (R_B, S_B) is previous block's signature.
6. Add block B_V containing fields $((R_V, S_V), (R_B, S_B), Q_V, Q_B, Index, Vname)$ into the blockchain.

2.3. Video Integrity Check

Integrity verification of the particular block B_V as follows:

1. Extract the block $B_V \rightarrow ((R_V, S_V), (R_B, S_B), Q_V, Q_B, Index, Vname)$.
2. Compute Z_B', the hash of previous block of B_V and Z_V' is the hash of the video segment $Vname$ from B_V.
3. Compute P_B' and P_V', points corresponding to the verification of previous block and video content using Eq. (4).
4. If x-coordinate of P_B' is equal to R_B, then previous block signature is verified. Likewise x-coordinate of P_V' is equal to R_V, then video content signature is verified. If both the signatures are valid the video is genuine otherwise tampered with.

3. Experimental Results

Five video segments [9], [10], [11], [12], [13] , which are publicly available are used in the experiments. These five videos are of resolution 1280x720 pixels and frame rate is 30 frames per second. Forged videos are created by using AVS video editor [14]. Tampered videos are created by deletion/copy-paste/insertion of frames in the video segment. Experiments were conducted on a PC having Intel Core i7-45U CPU@1.8GHz×4 and 12 GB RAM. The test videos were encoded using H.264/AVC video codec by FFmpeg [15]. For using cryptographic functions, cryptographic library called OpenSSL is used. Some of the videos from benchmark datasets such as VIRAT [16], SULFA [17], Derf's collection [18] are also used for testing.

3.1. Performance Evaluation

Table 1 shows the comparison of performance of our method with best existing method on different test videos with varying sizes. Time to create single block is defined as encoding time and that of verifying single block is verification time. From Table 1 it is obvious that our method is about 35 percent faster than the compared work.

Table 2 compares our method with other video integrity verification/forgery detection methods, where both our method and [6] possess the same capabilities. However, if the case arises that the verifier himself is an attacker, he can easily decrypt the key and hash from the verification integrity code stored in each block leading to a collision attack. In our method, even if the verifier is the authorized person, he can only verify the signature is valid or not.

Table 1. Encoding and verification time of proposed method in comparison to existing blockchain method

Video	**Encoding time (Seconds)**		**Verification time (Seconds)**	
	ECIES with HMAC	ECDSA	ECIES with HMAC	ECDSA
Video_1 [9]	0.0122	0.0080	0.0120	0.0077
Video_2 [10]	0.0184	0.0119	0.0184	0.0117
Video_3 [11]	0.0860	0.0567	0.0860	0.0565
Video_4 [12]	0.1650	0.1100	0.1630	0.1090
Video_5 [13]	0.0428	0.0270	0.0420	0.0269
VIRAT_S_010005_02_ 000177_000203 [16]	0.0138	0.0090	0.0136	0.0088
VIRAT_S_010106_03_ 000730_000782 [16]	0.0480	0.0310	0.0475	0.0305
VIRAT_S_000200_01_ 000226_000268 [16]	0.0188	0.0123	0.0185	0.0121
08_original [17]	0.0167	0.0108	0.0161	0.0105
05_original [17]	0.0324	0.0210	0.0316	0.0206
harbor_cif [18]	0.0493	0.0323	0.0488	0.0320
football_422_ntsc [18]	0.1826	0.1201	0.1807	0.1186

Table 2. Comparison of integrity verification capabilities of different methods.

Method	**Dependency on file type**	**Inter-frame tampering detection**	**Intra-frame tampering detection**
[2]	Yes (HEVC only)	Yes (Deletion only)	No
[3]	Yes (MPEG-2, MPEG-4, H.264 only)	Yes	No
[4]	Yes (MP4, MOV, 3GP only)	Yes	Yes
[5]	Yes (MP4 only)	Yes	Yes
[6]	No (Applicable to all types)	Yes	Yes
Proposed	No (Applicable to all types)	Yes	Yes

3.2. Security Analysis

By using blockchain, modification in the chained hash leads to integrity violation [19]. In our method, modification of the hash value is prevented by using the digital signatures for the video content and entire block. Two unique private keys are used for each block and these keys are randomly generated. ECDSA is based on infeasibility of solving elliptic curve discrete logarithm problem. From the pubic key in ECDSA it is difficult to calculate the private key [8]. This difficulty is doubled because of the use of two random keys. For a point G on C over Fq and integer a, the point aG can be computed in O ((log a) (log q)^3) bit operations. Thus time complexity of digital signature calculation is O ((log a) (log q)^3). Given G and aG, fastest known algorithms can compute a in O ($\sqrt{q}$) [20], which is greater than the time complexity for solving integer factorization problem. Furthermore, a digital signature only verifies the signature is valid or not, and does not reveal the content.

- **Brute-Force Attack**: One has to guess both block signature and video content signature for each block. Computational complexity is exponential in terms of key length [21] to find the key for each signature in each block. Unique pair of keys is used to generate signature for each video segment and each block which makes brute force attack even more difficult [8].
- **Collision Attack**: One tries to determine two messages having same value of hash. To attack the hash function SHA-512, the number of computations is of the order of $2^{m/2}$, m is output size in bits [21]. If initialization vector and algorithm are known, hash code for the messages can be generated to find the collision [21]. However, for ECDSA, the private key also should be known to create a message and code pair. For a successful attack, he should know both private keys for block and video content signature which is infeasible.

4. Conclusion

We proposed a blockchain based method for integrity verification of video data. Each block in blockchain includes the digital signature of the video content and previous block. Blocks are chained in chronological order. In verification, validity of both the signatures is checked. From the experimental result, it is evident that our method applies to any video type and detects any forgery type. Also, the analysis of the computational complexity confirms that the proposed method is faster. Use of ECDSA reduces memory consumption as the key size is smaller and provides higher level of security in comparison to other digital signature algorithms. Security analysis demonstrates that integrity verification method proposed is more resistant to brute force attacks and collision attacks. The proposed method can be used for video integrity verification in applications which demand faster integrity verification and higher robustness against tampering attempts.

References

[1] Singh RD, Aggarwal N. Video content authentication techniques: a comprehensive survey. Multimedia Systems. 2018; 24: p. 211–240.

[2] Hong JH, Yang Y, Oh BT. Detection of frame deletion in HEVC-coded video in the compressed domain. Digital Investigation. 2019; 30: p. 23–31.

[3] Vázquez-Padín D, Fontani M, Shullani D, Pérez-González F, Piva A, Barni M. Video integrity verification and GOP size estimation via generalized variation of prediction footprint. IEEE Transactions on Information Forensics and Security. 2019; 15: p. 1815–1830.

[4] Huamán CQ, Orozco ALS, Villalba LJG. Authentication and integrity of smartphone videos through multimedia container structure analysis. Future Generation Computer Systems. 2020; 108: p. 15–33.

[5] Iuliani M, Shullani D, Fontani M, Meucci S, Piva A. A video forensic framework for the unsupervised analysis of MP4-like file container. IEEE Transactions on Information Forensics and Security. 2018; 14: p. 635–645.

[6] Ghimire S, Choi JY, Lee B. Using blockchain for improved video integrity verification. IEEE Transactions on Multimedia. 2019; 22: p. 108–121.

[7] Al-Zubaidie M, Zhang Z, Zhang J. Efficient and secure ECDSA algorithm and its applications: a survey. arXiv preprint arXiv:1902.10313. 2019.

[8] Jana B, Poray J. A performance analysis on elliptic curve cryptography in network security. In 2016 International Conference on Computer, Electrical & Communication Engineering (ICCECE); 2016. p. 1–7.

[9] Real yellow car-YouTube. [Online]. [Accessed on April 2021]. Available from: https://www.youtube.com/watch?v=o2YNaYcwdbA/.

[10] Nature in 30 seconds-YouTube. [Online]. [Accessed on April 2021]. Available from: https://www.youtube.com/watch?v=MHna8CzxPLk/.

[11] 1 min of nature footage—4K (Ultra HD)-YouTube. [Online]. [Accessed on April 2021]. Available from: https://www.youtube.com/watch?v=WLKJnHu0GC4/.

[12] 4K video Ultra HD—Epic footage-YouTube. [Online]. [Accessed on April 2021]. Available from: https://www.youtube.com/watch?v=od5nla42Jvc/.

[13] 029-Realistic beautiful flower painting timelapse by artistbrownlion— Satisfying video—2.5 Min-YouTube. [Online]. [Accessed on April 2021]. Available from: https://www.youtube.com/watch?v=2g8bS-_nNYE&t=7s/.

[14] AVS video editor. [Online]. [Accessed on Dec. 5, 2020]. Available from: https://www.avs4you.com/avs-video-editor.aspx/.

[15] FFmpeg. [Online]. [Accessed on Nov. 20, 2020]. Available from: http://www.ffmpeg.org/.

[16] Oh S, Hoogs A, Perera A, Cuntoor N, Chen CC, Lee JT, et al. A large-scale benchmark dataset for event recognition in surveillance video. In CVPR 2011; 2011. p. 3153–3160.

[17] Qadir G, Yahaya S, Ho ATS. Surrey university library for forensic analysis (SULFA) of video content. 2012.

[18] Derf's Collection. [Online]. [Accessed on August. 6, 2021]. Available from: https://media.xiph.org/video/derf/.

[19] Aitzhan NZ, Svetinovic D. Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams. IEEE Transactions on Dependable and Secure Computing. 2016; 15: p. 840–852.

[20] Calabresi M. An introduction to elliptic curve cryptography. Ohio State Univ. 2016;: p. 10.

[21] Stallings W. Cryptography and network security, 4/E: Pearson Education India; 2006.