February 2020

Master's Degree Thesis

A Study on Blockchain-based Video Integrity Verification Method

Graduate School of Chosun University

Department of Information and Communication Engineering

Sarala Ghimire

A Study on Blockchain-based Video Integrity Verification Method

블록 체인 기반 비디오 무결성 검증 방법에 관한 연구

February 25, 2020

Graduate School of Chosun University

Department of Information and Communication Engineering

Sarala Ghimire

A Study on Blockchain-based Video Integrity Verification Method

Advisor: Prof. Bumshik Lee

A thesis submitted in partial fulfillment of the requirements for master's degree

October 2019

Graduate School of Chosun University

Department of Information and Communication Engineering

Sarala Ghimire

This is to certify that the master's thesis of

Sarala Ghimire

has been approved by examining committee for the thesis requirement for the master's degree in Engineering.

Committee Chairperson	**Prof. Jae-Young Pyun**
Committee Member	**Prof. Tai-Won Um**
Committee Member	**Prof. Bumshik Lee**

November 2019

Graduate School of Chosun University

초 록

블록 체인 기반 비디오 무결성 검증 방법에 관한 연구

사랄라 기미레
지도교수: 이범식 교수
조선대학교대학원, 정보통신공학과

비디오 영상 기록물은 범죄 현장 또는 자동차 도로 사고에 대한 증거를 제공하는 데 중요한 역할을 한다. 그러나 비디오 영상의 주요 문제점은 비디오 복사, 위조, 변조, 삭제 및 변경 등 다양한 보안 공격에 취약하다는 것입니다. 비디오에 대한 무결성 검증을 수행하기 위해서는 시각적 증거가 필요하지만, 인간의 눈으로 위와 같은 위·변조 및 삭제 등을 감지하기가 쉽지 않다. 본 학위 논문에서는 블록 체인 프레임 워크를 활용하는 새로운 비디오 무결성 검증 방법 (IVM)을 제안한다. 제안된 방법은 비디오의 무결성을 검증하기 위해 해시 기반 메시지 인증 코드 (HMAC)와 타원 곡선 암호화 (ECC)를 결합하여 중앙 집중식 비디오 데이터에 효과적인 블록 체인 모델을 사용한다. 제안 방법에서는 미리 결정된 크기 (세그먼트)를 갖는 비디오 기록물이 실시간 방식으로 키 해시되고 시간 순으로 체인 방식으로 저장되어, 보안이 매우 우수한 변경 불가능한 데이터베이스를 구축한다. 검증 과정은 해당 비디오 세그먼트에 동일한 절차를 적용하고 블록 체인의 해시와 비교할 수 있는 해시 값을 생성한다. 제안하는 IVM은 PC 환경 뿐 만 아니라 자동차 사고 데이터 레코더 (ADR)와 같은 임베디드 시스템에서 검증하였다. 실험 결과는 제안된 방법이 다른 기존 최신 방법에 비해 복사 이동, 삽입 및 삭제와 같은 다양한 종류의 탬퍼링에 대해 더 우수한 무결성 검증기능과 견고성을 가지고 있음을 보여준다. 또한 실행 시간에 따른 복잡도 분석은 블록 체인 내 블록 수의 증가가 제안된 방법에서 최소한의 오버 헤드를 만을 증가시킴을 실험적으로 검증하였다.

ABSTRACT

A Study on Blockchain-based Video Integrity Verification Method

Sarala Ghimire
Advisor: Prof. Bumshik Lee
Department of Information Communication Engineering
Graduate School of Chosun University

A video record plays a crucial role in providing evidence for crime scenes or road accidents. However, the main problem with the video record is that it is often vulnerable to various video tampering attacks. Although visual evidence is required to conduct an integrity verification before investigations, it is still difficult for human vision to detect a forgery. In this thesis, a novel video integrity verification method (IVM) is proposed that takes advantage of a blockchain framework. The proposed method employs an effective blockchain model in centralized video data, by combining a hash-based message authentication code (HMAC) and elliptic curve cryptography (ECC) to verify the integrity of a video. In the proposed method, video content with a predetermined size (segments) is key-hashed in a real-time manner and stored in a chronologically-chained fashion, thus establishing an irrefutable database. The verification process applies the same procedure to the video segment and generates a hash value that can be compared with the hash in the blockchain. The proposed IVM is implemented on a PC environment, as well as on an accident data recorder (ADR)-embedded system for verification. The experimental results show that the proposed method has better detection capabilities and robustness towards

various kinds of tampering such as copy-move, insert, and delete, as compared to other state-of-the-art methods. An analysis based on execution time along with an increase in the number of blocks within the blockchain shows a minimal overhead in the proposed method.

Index Terms: Affinity propagation clustering, Bluetooth low energy, Fingerprinting localization, Gaussian process regression, Indoor positioning system, Location based services, Weighted centroid localization

Acronyms

ADR	Accident Data Recorder
AES	Advanced Encryption Standard
bK	Clustering Block Key
CCTV	Closed-Circuit Television
DCT	Discrete Cosine Transform
dK	Data Key
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie-Hellman
ECDLP	Elliptic Curve Discrete Logarithm Problem
GF	Galois Field
GOP	Group of Picture
HMAC	Hash-based Message Authentication Code
IVM	Integrity Verification Method
KDF	Key Derivation Function
MAC	Message Authentication Code
RSA	Rivest–Shamir–Adleman
SHA-256	Secure Hash Algorithm-256
VEDR	Video Event Data Recorder
VIC	Video Integrity Code

Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Motivation

In recent years, the development of digital technology has tremendously increased. Owing to the significance that the digital/physical witness and evidence have, the use of video applications or surveillance systems, such as Closed-Circuit Television (CCTV) systems or accident data recorder (ADR) systems, i.e., the so-called vehicle "black box", has been increased. These systems are typically used in various sensitive areas like monitoring of the activity in the bank, monitoring residential areas, and monitoring the road and highways. The vehicle black box is a device that records the video images in and surrounding of the vehicle in a highway. As the video recorded by the surveillance cameras or the video applications captures critical visual information, which acts as a witness, it plays a crucial role in criminal investigations or dispute examinations [1]. For instance, to investigate the road accident and to determine the cause and victim of the accident, it requires physical evidence such as status of vehicles, victims, or the real culprit. The traces of such evidence that make easy identification of the cause is obtained by the surveillance cameras mounted publicly on the road or inside the vehicles (ADR). However, the likelihood of alteration of the video content is significantly very high if the offender aims to conceal the decisive information. The openness of the networks [2] that eases the accessibility of the video data is the main reason for the subtle manipulation. The shared and open videos may incur duplication of videos that may infringe the copyright [3] or illegal distribution [4]. In addition, the publicly-available media data can easily be manipulated or tampered with video editing tools utilized with malicious intentions without leaving any visible clues [5]. It is easy to insert or delete certain objects, activity or vehicle information from the current video or can insert frames from other videos. Unlike the surveillance cameras mounted publicly, the probable intruder

in ADR is typically the owner or the driver himself. In such a scenario, video data is always exposed to the attacker with a high possibility of video forgery. Thus, the adequacy in performing tampering operation threatens the integrity and authenticity of the video data, particularly if the video is considered as the evidence in criminal or dispute examination in court of law or any other areas like surveillance systems, advertisement and movie industry. It may potentially cause the situation that the wrong person is convicted or punished. Thus, it is important to guarantee that the video content is not forged but authentic before using it as an evidence, i.e. verify the video as authentic and unaltered.

Fig. 1.1 shows an example of how the ADR system inside the vehicle or publicly mounted surveillance cameras is important in car accident investigation. As shown in Fig. 1.1, the car-A collides with the car-B and the surveillance camera in the road or the ADR system in the car records visual information that captures every single moment of the accident that occurred during the collision. However, with the intention to conceal the real cause and to misguide the investigation proceeding, the critical information having evidence is likely to be modified by the intruder without leaving any clue of alteration. The investigation with the same modified video without any verification facility will let the real culprit to be freed while the innocent will likely to be punished. Thus, to avoid such a misleading investigation, the integrity of the video content must be verified (if the video is altered or not) before using it as evidence.

1.2 Objectives

The major objective of this research is to develop a video integrity verification method (IVM) for any surveillance system or video applications. The problems in the state-of-the-art methods is overcome by using an active video forgery detection scheme based on blockchain [6] that ensures the integrity of the video data against forgery attacks. The proposed blockchain based video IVM is employed on a centralized database system that is more robust in tampering detection and provides a high level of security.

1.3 Contributions

Typically, the surveillance videos are recorded every few minutes intervals generating small segments of videos, such as one or five minutes intervals. Thus, considering the property that the generated video segments in the surveillance systems are encoded by the video codec, which are used as an input to the proposed IVM, the method is independent of frame types and its dependencies such as I, P and B frames. Furthermore, since the 'already compressed video segments' compressed by any types of video codecs are utilized, the complicated parsing process and decoding for getting coding parameters such as motion vectors, quantized coefficients, and quantization parameters are not required. Instead, the hash values of the compressed video segments are generated that have a higher accuracy of forgery detection without consideration of parsing and dependencies. Such encoded video segments are hashed and chained with the hash of previous segments creating an unbreakable chain of hash. Thus, the integrity values (hash values) of each video segment are associated with each other in such a way that, alteration of any video segment is traced. Moreover, the standalone hash function is replaced with an Hash-based Message Authentication Code (HMAC) [7] for hashing the video segment in a blockchain, which increases the level of security with a secret unique key. The HMAC is a shared-key cryptographic algorithm that is used to verify the data integrity and authenticity using an underlying hash function [8]. In addition, the efficiency of the proposed method is increased in terms of memory consumption owing to the segment-wise hashing process. The contributions of the proposed method are summarized as follows:

1. A blockchain concept is introduced in the context of centralized video data. To the best of our knowledge, this is the first time that blockchain has been used for video integrity in centralized video data.

2. The segment-wise integrity check can achieve faster processing and less memory consumption for video integrity verification.

3. A number of security issues and attacks are systematically analyzed, by

which the IVM is verified with a high level of security.

4. The efficacy of the method is well-tested on a real-time ADR system as well as on a computer simulation, ensuring the applicability in real-world applications.

1.4 Thesis Layout

The structure of this thesis is organized as follows. In Chapter 2, the basic concept of blockchain, ECC, and randomized hashing is introduced. Previous works relevant to the video integrity verification methods are discussed in Chapter 3. Chapter 4 provides the overview of the proposed method. The experimental setup and methodology are discussed in Chapter 5. Chapter 6 demonstrates the experimental results and analysis and the security analysis is presented in Chapter 7. Finally, the thesis work is concluded with its future works in Chapter 8.

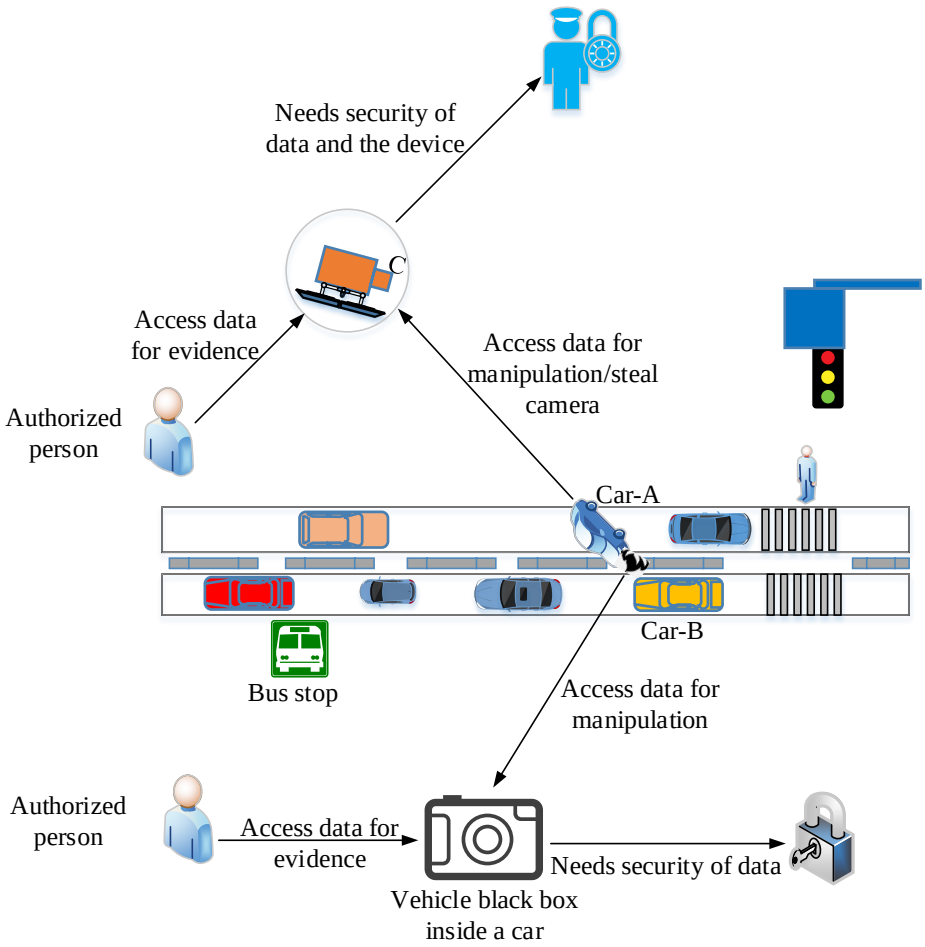

Figure 1.1. The accident data recorder (ADR) system and the need for the data security for the evidence of the accident.

Chapter 2
Background

2.1 Blockchain

A blockchain is typically a chain of blocks and blocks are the transactions, thus it is known as a distributed open ledger containing a block of transactions executed in a network, and is maintained by a node itself [9]. Fig. 2.1 shows a general concept of blockchain. A block of the transaction is added to the chain, and the hash of that block is contained within the next block. Thus, the chronological chain or the sequential nested blocks of data are generated, which guarantees that the data could not be changed without changing its block and the following blocks [10]. Every block contains the hash of its parent block and links the sequence of blocks to create a chain [11]. Thus, the cryptographic hash of the parent block contained within the header of the block is used to recognize the individual block. The body part of the block contains batches of valid transactions that are hashed and encoded into a Merkle tree, as shown in Fig. 2.1, by Tx_Root. In addition, a nonce value and a timestamp are added to the block. The nonce is a random integer value that is generated repeatedly until the run of leading zeros is contained in the hash of the block which qualifies the block to be added to the blockchain. As this process is iterative that requires time and resources, the correct calculation of nonce constitutes the proof of work in blockchain [12]. Hence, the integrity of blockchain is based on chained cryptography that makes it quite difficult to break.

The original concept of blockchain technology designed for financial ledger and in a decentralization system can be extended and utilized in other contexts or to other frameworks, such as permission management and medical data access system [13]. However, the implementation of blockchain including all properties requires, proof of work algorithm to be applied and the system should be decentralized. Integrating a full-principle operations in cases like lack of infrastructure,

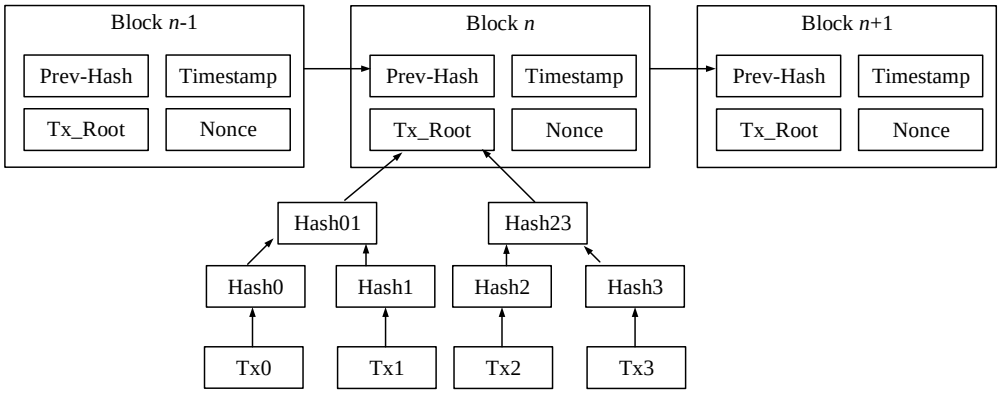

Figure 2.1. The general concept of blockchain.

low power devices, highly sensitive data can thus be an overwhelming problem.

Thus, in this thesis, a novel method that incorporates the properties of the blockchain for IVM into a centralized database systems is developed. Moreover, the likelihood of an intruder to change the entire database system by recalculating the hash value is resolved by applying a keyed hash algorithm (HMAC) with a unique private key to individual blocks.

2.1.1 Cryptographic Hash Algorithm

The general concept of a hash algorithm is the one-way function i.e. no input can be determined with the given output. Moreover, the arbitrary length input data is mapped to a small fixed-length output, which is designed to provide the integrity of the data. The algorithm is deterministic in nature, that is the same input will always give the same output, while the small change in input sequence must always produce a different output. Thus, it is typically used to verify the integrity of the data. Blockchain technology utilizes the hash algorithm to hash the transaction data that generate the integrity value of the transaction, as shown in Fig. 2.1 by Tx_Root. Generally the Secure Hash Algorithm-2 (SHA-2) is used in blockchain, which is a second-generation hash algorithm of SHA family. In addition to the transaction hash generation in the blockchain, the previous blocks

in the chain are also hashed and included in the next blocks that act as the header of the block (denoted by $Prev-Hash$ in Fig. 2.1). The hash value of the previous block is thus used to verify the integrity of that block (to check whether the block values are altered or not).

2.1.2 HMAC Algorithm

In the proposed method, the hash algorithm in the conventional blockchain is replaced by HMAC, which requires a key to hash the message, providing additional security to the data. A message authentication code (MAC) is an algorithm that produces an authentication code for the message using a secret key. It is impossible to produce a Mac of the message without knowing the secret key. Different Mac is produced for every change in the key for the same message. An HMAC is a MAC algorithm, which is based on the standard hash function. The basic idea is to hash the concatenation of key and the message together.

HMAC has two passes of hash computation. Firstly, two subkeys (inner key and outer key) are derived from a single secret key. The first pass of the hash computation produces an internal hash from the message and the inner key. The final HMAC code is then derived in a second pass by using the result of first pass and the outer key. Thus, the length extension attack with the construction of iterative hash is mitigated with this algorithm owing to the use of two passes of hashing. The message is not encrypted with HMAC, instead, the HMAC hash is sent alongside the message (encrypted or not). The message on the receiving side is then again hashed by the receiver having a secret key, and if the computed hash and the received hash matches then it is considered as authentic.

The algorithm is used to verify (authenticate) that whether the data has not been altered or replaced. That means the authenticity and integrity of the message are verified by using this algorithm. The HMAC value sent with the message is used for the verification of the integrity, recomputing the HMAC of the message and comparing it with the received HMAC. This is possible only by using the secret key shared between two trusted parties that serve the purpose of authenticity. Since its precomputation is impossible, given a cryptographic hash, to find

out what it is the hash of, knowing the hash does not make it possible to find the key. Moreover, the cryptographic strength of the HMAC depends upon the size of the secret key that is used. The brute force attack to discover the secret key is the most common attack on HMACs. However, HMACs are substantially less affected by collisions than their underlying hashing algorithms alone.

2.2 Elliptic Curve Cryptography

Cryptography is the key for securing data from an intruder, and plays a major role in storing data securely. ECC [14] is a public key cryptography based on solving the elliptic curve discrete logarithm problem (ECDLP) [14]. It provides a high level of security with a smaller key size as compared to other cryptographic schemes [14]. An elliptic curve E over the finite field (or Galois Field) GF is defined by (2.1), known as the Weierstrass equation for elliptic curves in non-homogeneous form [15]:

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6, \tag{2.1}$$

where x and y are variables in affine coordinates of the elliptic curves and a_1, ..., a_6 are the set of elements in Galois field (GF) [16], which is a field that contains set of finite number of elements and needs to have well defined operations of multiplication, addition, subtraction and divisions with certain basic rules [16]. The discriminant Δ of E is not equal to zero ($\Delta \neq 0$), and is calculated as in (2.2).

$$\Delta = d_2^2d_8 - 8d_4^3 - 27d_6^2 + 9d_2d_4d_6, \tag{2.2}$$

where d_2, d_4, d_6, and d_8 are calculated using the elements $a_1, \ldots, a_6$ from finite field GF as in (2.3) and (2.4)

$$d_2 = a_1^2 + 4a_2, d_4 = 2a_4 + a_1a_3, d_6 = a_3^2 + 4a_6 \tag{2.3}$$

and (2.4) can finally be obtained as:

$$d_8 = a_1^2a_6 + 4a_2a_6 - a_1a_3a_4 + a_2a_3^2 - a_4^2. \tag{2.4}$$

The condition $\Delta \neq 0$ assures that the curve is non-singular, and that there are no curve points with two or more different tangent lines. The projective form of (2.1) of an elliptic curve E defined over GF is obtained by replacing x by X/Z^c , and y by Y/Z^d , and clearing denominators. Here X, Y, and Z are projective coordinates of elliptic curve and c and d are positive integer values. Thus, the homogeneous form of (2.1) can be written as:

$$Y^2Z + a_1XYZ + a_3YZ^2 = X^3 + a_2X^2Z + a_4XZ^2 + a_6Z^3, \tag{2.5}$$

where $X = xZ^c$, $Y = yZ^d$, c = 1, and d = 1.This entails the existence of a special point, known as the identity element, that characterizes the elliptic curve. When the same point is added several times to itself in an elliptic curve, the addition operator is transformed into the scalar multiplication, which in practice allows to multiply an elliptic curve point P by a positive integer n in order to produce another elliptic curve point, $S = n \cdot P$.

Moreover, for the ECC computation, two types of finite fields $GF(q)$, prime and binary finite fields, with $q = p^m$ elements are used, where p is a prime number called the characteristic of finite field, and m is a positive integer. The prime finite field is denoted by GF (p) where p is any odd prime number with m=1. The binary finite field which is also known as characteristic-two finite field GF (2^m) is a finite field with p=2 and m having any integer value greater than 1. By changing the variables for finite fields, the Weierstrass equation (2.1) can be simplified [17]. Thus, if the characteristics of the finite field is neither two nor three, the equation (2.1) can be reduced to the form (2.6):

$$y^2 = x^3 + ax + b. \tag{2.6}$$

This curve equation is used in ECC, where the discriminant is defined as:

$$\Delta = -16(4a^3 + 27b^2). \tag{2.7}$$

Similarly, a and b are constants with the constraint of $4a^3 + 27b^2 = 0$. The coordinate points of an elliptic curve are used for a cryptographic operation [18]. The equation for the elliptic curve over a finite field is written as (2.8):

$$y^2 = (x^3 + ax + b) mod(p), \tag{2.8}$$

where mod is a modular operator. The set of parameters that defines the curve used in the ECC implementation depends on the underlying finite field, i.e. if the finite field is GF(p) then, the set of parameters are (p, a, b, G, n, h). Here a and b are field elements that specify the equation of EC, p is a prime number that characterizes the finite field GF(p), n is the prime number whose value represents the order of the point G (i.e. $n \cdot G$=O; the point in infinity). Similarly, h is co-factor, and G is a point on the curve that is used as a generator of the points representing public keys. Some of the mathematical operations that are used for implementation of ECC are shown in Fig. 2.2.

If two parties A and B require secure communication between them, then they should agree upon a common elliptic curve equation and a generator G. Assuming that the private keys of A and B are denoted as K_A and K_B, respectively, then their public keys are derived from $P_A = K_A G$ and $P_B = K_B G$. These public keys are distributed publicly. If A wants to communicate with B and wants to send a message, then A encrypts the message with the public key of $B(P_B)$ and generate the cipher text P_C as (2.9)

$$P_C = kG, P_m + kP_B, \tag{2.9}$$

where k is any random number, P_m is a message to send and P_C is the cipher text. For the decryption of the message, B decrypts the message as (2.10).

$$P_m = P_m + kP_B - K_B kG. \tag{2.10}$$

As k is a random number that generates a unique value every time, a different cipher text is generated for the same message each time. This makes it difficult for anyone who tries to illegally decrypt the message. Thus, the message can only be decrypted with the agreed-upon secret key. Furthermore, the computational complexity required to break the encryption of the ECC algorithm is very high, making it more secure than other asymmetric encryption algorithms [19]. Hence, to achieve a higher level of security, the ECC encryption algorithm is employed to encrypt the key in the proposed method. Moreover, a performance comparison between ECC and Rivest–Shamir–Adleman (RSA) is described in Section 5.2.

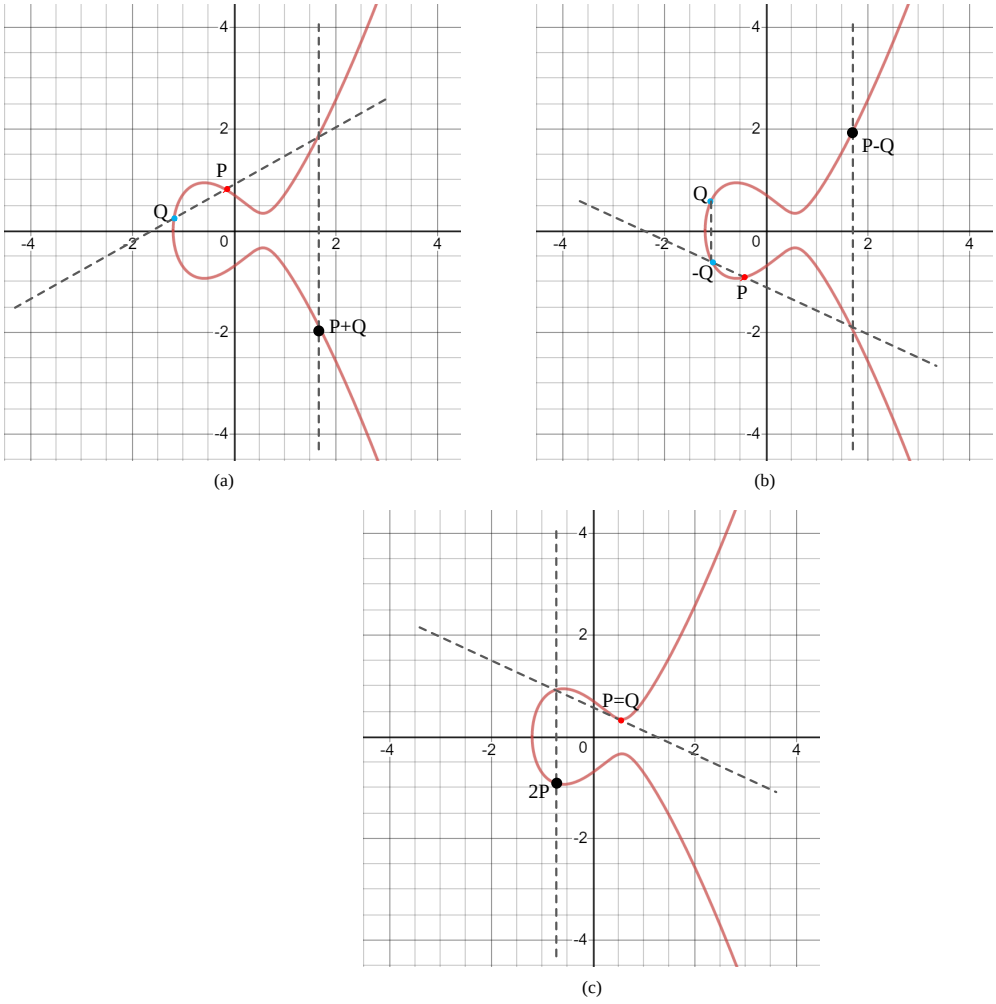

Figure 2.2. (a) Point addition, (b) Point subtraction, and (c) Point doubling.

2.3 Randomized Hashing

The hash function reduces the arbitrary length data to a fixed size that ensures the integrity and originality of the data. It generates so-called 'fingerprint' of the data, which is always unique and equal. That is, the small changes in the input result in large variations in output whereas the same input always gives the same output. Thus, the hash can be used to verify the integrity of the video by comparing its fingerprint with the one generated on the validation. Similarly, another property of the hash function is collision resistant where attacker cannot find two messages M_1, and M_2 such that, $H(M_1) = H(M_2)$ which plays a vital role in integrity checking, where $H(\ )$ is a hash function. However, it is reported in [20] that using a hash function alone is less resilient to the collision in spite of a number of benefits. The randomization on the input data before hashing is used to overcome the shortcoming of the hash function, where the hash function is not modified instead the input data is randomized with the key before applying a standard hash algorithm [21]. Fig. 2.3 shows the basic concept of the randomized hashing for the comparison with the conventional hashing.

As shown in Fig. 2.3, two sub-keys, r_0 and r' are generated from the main key r, and these keys are used to randomize the input data. Sub-key r_0 is prepended to the input while r' is processed with a XOR operator to all the blocks of the parsed input data $M = (m_1, m_2, \ldots, m_l)$. Subsequently, the output from the randomization is input to the hash function. For the input message $M = (m_1, m_2, \ldots, m_l)$, output of a conventional hash algorithm is computed as

$$H_M = Hash(M). \tag{2.11}$$

However, randomized hash is computed as (2.12)

$$H'_M = Hash(M'), \tag{2.12}$$

where M' is a randomized message with a key r as obtained as (2.13)

$$M' = r_0(m_1 \oplus r', m_2 \oplus r', \ldots, m_l \oplus r'), \tag{2.13}$$

where $r_0 = r \parallel$ 0×00, r'= $r \parallel r$, $\oplus$ and $\parallel$ are the XOR and OR operators, respectively

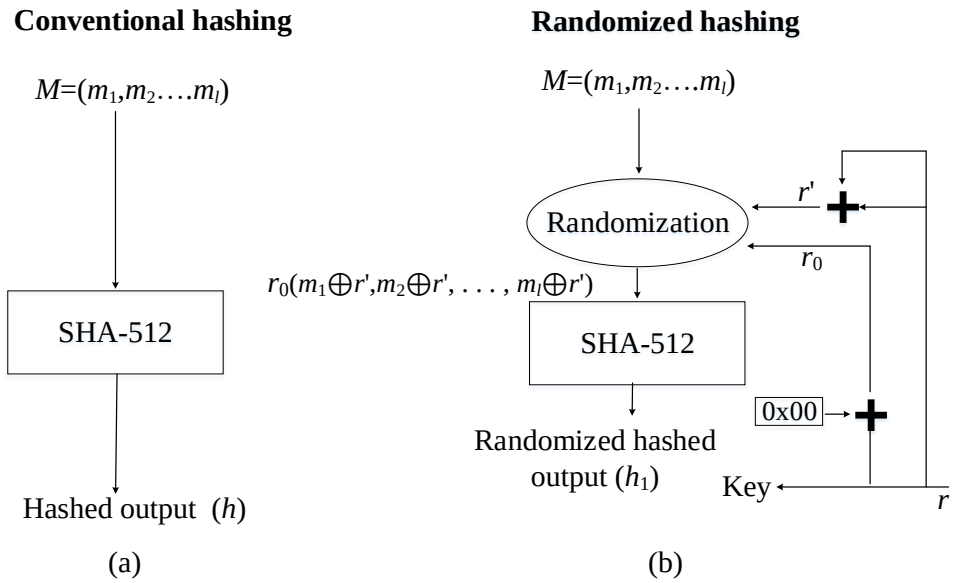

Figure 2.3. The basic concept of (a) conventional hashing and (b) randomized hashing.

Chapter 3
Related Works

3.1 Compression-based Approaches

A number of passive video tampering detection schemes based on compression artifacts were proposed in [22],[22–28]. For the detection of double compression artifacts, [23] exploits the relations between video frame tampering (delete, copy, and move) and coding-type changes, e.g., in intra- and inter- coding, and uni- and bi-directional coding. In addition, the different influences of quantization on different frame coding types are qualitatively analyzed. However, if the number of deleted frames is equal to the group of picture (GOP) size, it is difficult to detect the tampering, as the detection of the tampering is highly dependent on the changes in the coding types. In [24], Markov modeling of a difference of discrete cosine transform (DCT) coefficients is used to detect double compression artifacts. The method has the limitation of scaling factors (quantization parameters) for first and second compressions. It is reported in [24] that the system fails to detect tampering if the scaling factor in the first compression is an odd multiple of the scaling factor of the second compression. The block-level correlation of noise residues is explored in [25]. The correlations between the noise residues of temporally neighboring blocks (i.e., blocks in the same position belonging to two adjacent frames) are evaluated. However, the method is likely to miss the detection of a forgery if a calculation error of noise residues occurs. In [26] artifacts from the distribution of DCT coefficients of I-frames and prediction errors of P-frames are utilized to classify a video as a double-compressed video, and accordingly to detect the occurrence of forgery in the video. The principle of estimation theory is used in [27] to detect double compression artifacts by double quantization. Each pixel of a given frame is estimated from a spatially co-located pixel of all the other frames in a GOP. The error between original and estimated pixel values is subjected to a threshold to identify the double-compressed frames

in a GOP. In [28], a segmentation of background regions in each frame is obtained based on the motion vector field which is used to calculate the prediction residuals in each frame with adaptive parameters. The artifacts of double compression are detected by temporal periodic analysis of the feature sequence generated from the post-processed prediction residual. The detection method in [29] is based on a variation of macroblock prediction types in a re-encoded P-frame. Similarly, the compression noise characteristics in a frame are analyzed in [22], where the method fails to detect tampering if a quantization step size in the second compression is the same as the one used in the first compression. The aforementioned previous methods [22],[21–28] are based on artifacts generated by the double compression. The main drawback of these methods is that the forgery detection rate is significantly reduced if parameters such as GOP size, quantization step size, or scaling factors are not identical between the first and second compressions.

3.2 Content-based Approaches

Content-based video hashing methods in [30] and [31] ensures the authenticity of the video feature. However, they are more focused on video retrieval and identification. The work in [30] is used for near-duplicate video retrieval and video feature representation, whereas the method in [31] uses normalized vector representation of the video for identification of the video that includes the spatial resizing and temporal sampling of the content of the video. Moreover, such methods require sensitivity towards the content changing manipulations, but they must be robust against content-preserving manipulations, ignoring the non-malicious changes.

3.3 Filesystem-based Approaches

File system-based integrity verification methods (IVMs) are proposed in [1] and [32]. The method in [32] exploits video frames remaining in the slack space of

media storage, whereas [1] utilizes the structure of the video content in a video event data recorder (VEDR), and identifies a difference in a frame index field between forged and original files. However, these methods may not work for all video file systems with different file structures, and is not promising for detection of integrity violations such as frame replacement, insertion, and image editing. Song et al. [33] proposed a video IVM based on the file structure of manipulated video content. Because the file structure generated by a video editing tool is stored as a signature in a database, it can only detect tampered files whose structures are stored in the database. Thus, a video manipulated by using a proprietary method instead of using a standard editing tool is not detected by the system.

3.4 Hash-based Approaches

In [34], chained hash and symmetric-key encryption are used to check data forgery. Similarly, storage integrity guaranteeing mechanism against tampering attempts (SIGMATA) using cryptographic algorithms is proposed in [35], where integrity assurance values of every frame are generated by the hashing algorithms MD5 [36], RIPEMD-128 [37], and SHA-1 [38], and are used for verifying the integrity. However, as the hash algorithms are applied for every individual frame, the encoding and verification processes of these methods consume large amounts of time and memory.

Robert et al. [39] proposed a signature-based image authentication method for the JPEG image using blockchain in the decentralized network. The scheme utilizes the quantized coefficients of each 8×8 block of the image as a signature, which is encrypted with Advanced Encryption Standard (AES) [40] encryption and stored in the blockchain. However, the video consisting of large number of image frames requires significantly higher computational complexity for signature generation, compared to the case of using still images for this method. Moreover, it is required to parse the bitstream to obtain the quantized coefficients for signature generation, which also results in high complexity. Furthermore, [41] and [42] developed an architecture of integrity verification method for ADR and

analyzed the method with different configurations. The best method with HMAC and elliptic curve cryptography (ECC) is recommended for the implementation.

Chapter 4
Proposed Method

A video IVM based on a principle of a blockchain with HMAC and ECC is proposed in this thesis that stores the hash of each video segment in chronological order throughout the videos captured using CCTV or ADR, etc. It is to be noted that, the videos are recorded in the surveillance system every few minutes interval. For example, CCTV usually records video of every few minutes interval based on the user's setting [43], whereas the cameras of the ADR system capture the scene of every one-minute interval. In light of this fact, video clips recorded every one-minute intervals are defined as video segments in the proposed method.

The overall framework of the proposed video IVM is shown in Fig. 4.1. As shown in Fig. 4.1, the architecture of the proposed method is divided into four sections: a data hash section, key encryption section, block key generation section, and blockchain storage section. The hash and message authentication code (MAC) with the HMAC algorithm [44] for each video segment is generated in the data hash section. Each video segment is hashed by a secure hash algorithm (SHA-256) [45], after which an HMAC algorithm with a data HMAC key (denoted as dK) is applied to the hash value, which ensures the authenticity and confidentiality of the video data. Moreover, for each video segment, a random and unique key dK is generated. However, the key has the most important role in keeping the data safe. If the key is easily accessible to an attacker without any huge effort, one could tamper the video and generate hash value and replace it, leaving no clue of alteration. Nevertheless, owing to the lack of evidence of alteration, the integrity of the video can still be valid. Thus, the second section includes the key encryption, where the ECC encryption algorithm is used to encrypt the key. To increase the level of security and to increase the difficulty in forgery attack, the key dK, as well as the MAC value of the video, is encrypted. The output value from this encryption is defined as the video integrity code (VIC) in

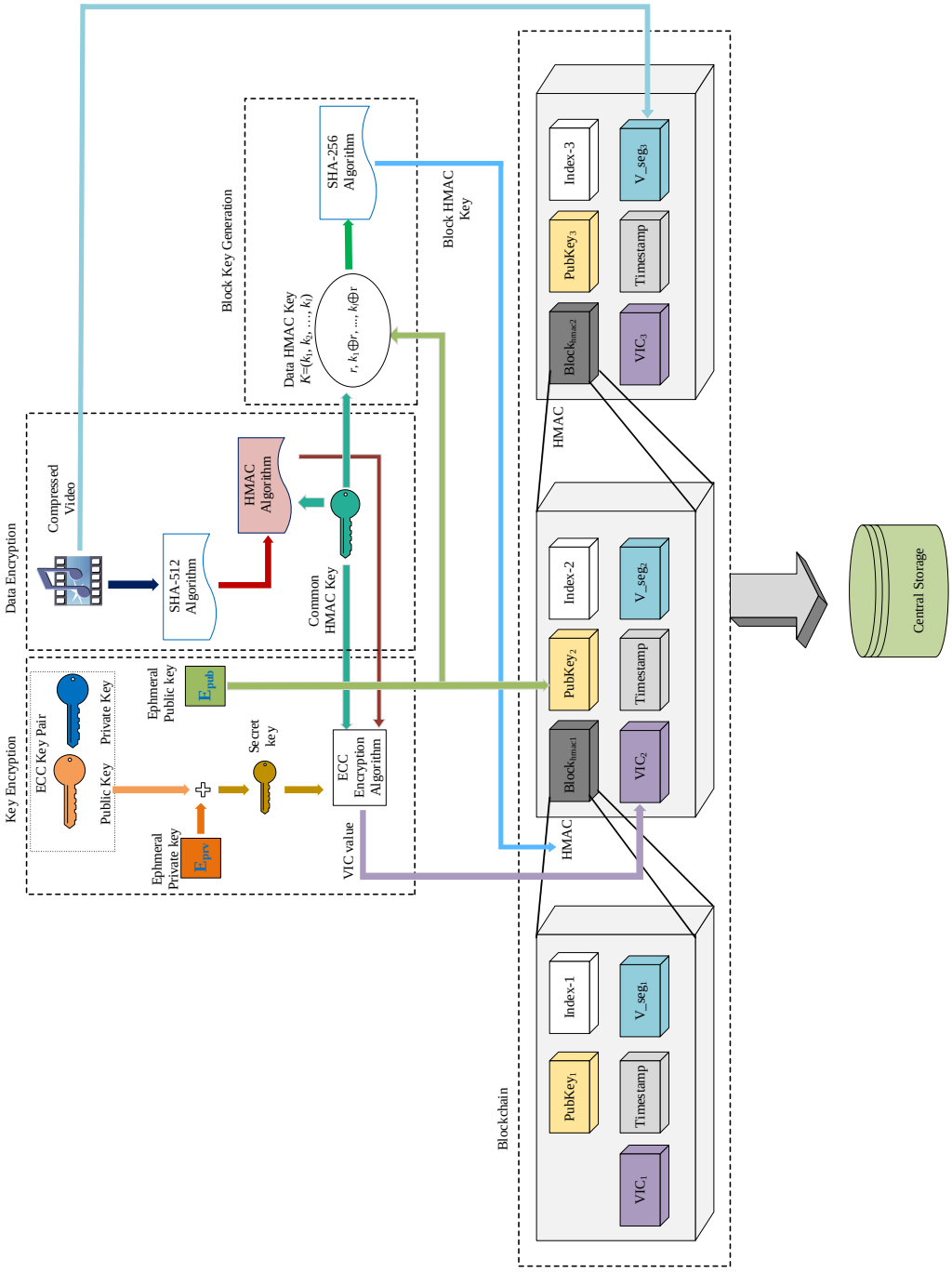

Figure 4.1. A schematic block diagram of the proposed method.

the proposed method. With the increasing number of video segments, the number of keys increases, resulting the memory consumption to increase. Thus, an ECC that provides a high level of security with smaller key sizes [17] is employed in the proposed method. An elliptic curve integrated encryption scheme (ECIES) is applied for the encryption, which is an extended form of ECC. The ECIES uses the Elliptic Curve Diffie-Hellman (ECDH) algorithm for the key agreement function and symmetric encryption algorithm for encryption of the key. The next section generates the key for the HMAC algorithm, which is used to hash a block in a blockchain, thus the section named key generation section. A block key (bK) is generated from data key dK with randomized hashing applied to it. Finally, the last section of the block diagram in Fig. 4.1 is a "Blockchain" section, which stores the outputs from data and the key encryption sections in the block. Every block is linked with previous and subsequent blocks forming a blockchain. A single block in the blockchain comprises the path of the video segment V_seg, an ephemeral public key of ECC, the VIC value of the corresponding video segment, the HMAC value of the previous block and a timestamp. Moreover, storing a timestamp to any file or document to verify the integrity is renowned [46], which provides evidence and authentic-time of the file.

4.1 Key Encryption Using ECIES

An overall pipeline of the ECIES is shown in Fig. 4.2. As described in Section 2.2, the ECIES is a public-key encryption scheme, which depends on the elliptic curve equation, and its domain parameters over a finite field. The randomly chosen private key is used to calculate the public key on the basis of the same field and parameters as:

$$Kb = KvG, \tag{4.1}$$

where Kv is a private key, Kb is a public key, and G is the generator point on the curve. The private key is not disclosed while the public key is distributed

publicly. The ephemeral public-private key pair is similarly generated using the same agreed curve parameters, which is used to derive an ECC key, as shown in Fig. 4.2. The private key is generated randomly and is used to calculate the public key using (4.2):

$$k_{ue} = K_e G, \tag{4.2}$$

where k_e denotes ephemeral private key and k_{ue} is an ephemeral public key. The main public key and the ephemeral private key is then used to derive secret key as defined in (4.3). The ephemeral private key is destroyed after the secret key is derived.

$$SK_E = k_e Kb. \tag{4.3}$$

A key derivation function (KDF) is applied to the secret key to generate the ECC encryption key, for which the SHA-256 hash algorithm is used for KDF as in (4.4).

$$E_k = Hash(SK_E, info, i). \tag{4.4}$$

Here, i is the number of iterations by which the hash function is applied, $info$ is an optional extra common value, and E_k is the ECC encryption key. The key E_k is used in ECIES (AES encryption algorithm) to encrypt the HMAC value of the video H_m and the key dK, as:

$$VIC = Encrypt(E_k, H_m||dK), \tag{4.5}$$

where $||$ is the concatenation operator, and the encryption output is given by VIC. The secret key in the decryption side is calculated by using the main private key and the ephemeral public key, as (4.6):

$$SK_D = k_{ue} Kv. \tag{4.6}$$

Similar to the encryption key generation, the decryption key is then obtained by using a key derivation function.

$$D_k = Hash(SK_D, info, i) \tag{4.7}$$

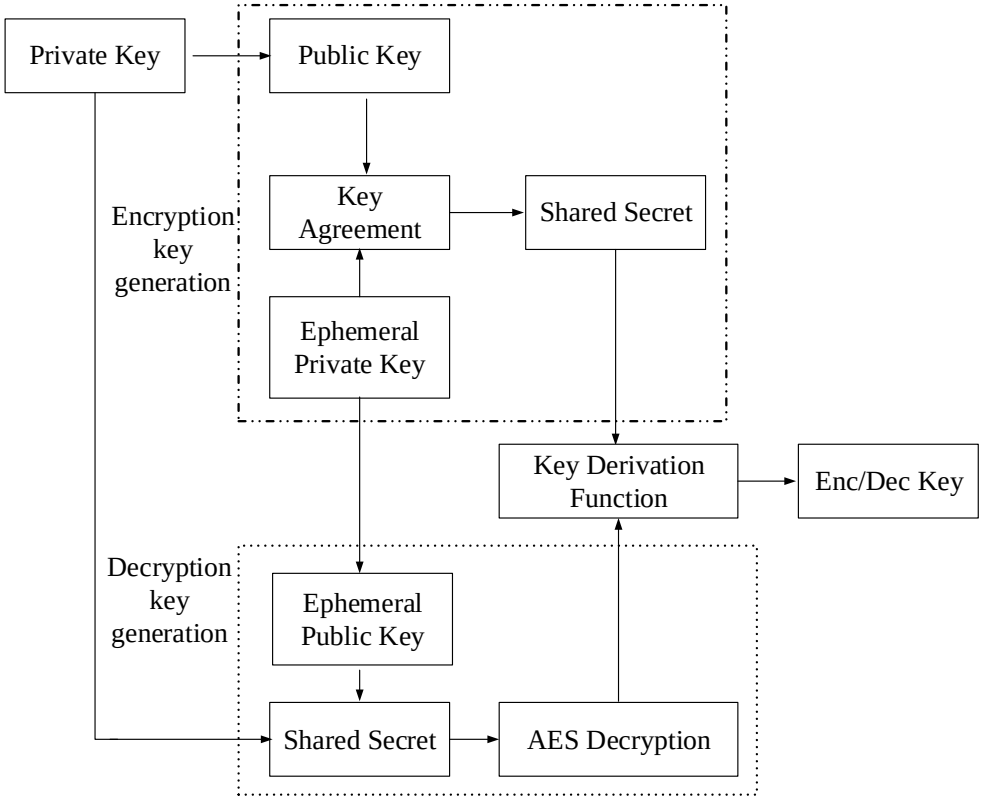

Figure 4.2. Elliptic curve cryptography (ECC) functional diagram.

Overall, the public key is visible to everyone and has access to encrypt, whereas the decrypt privileges are only with the authorized person who has the (static) private key. Furthermore, the randomly generated ephemeral private key to generate ECC key is subsequently destroyed immediately. Thus, for more than one encryption, a single public key can be adequately used. Hence, for the multiple encryptions, the single public key can be is used keeping it open to the public. This shows that the computational cost is equal irrespective of the number of different encryptions, which is why it is more beneficial than conventional schemes that require the management of multiple keys on both encryption and decryption side.

4.2 Proposed Blockchain Generation

Typically, the blockchain can be defined as the chain of blocks and each block contains transaction information, reference to the previous block, proof of work, Merkle root, and timestamp, which is publicly decentralized. The reference to the previous block is the hash value of the previous block, and the transaction integrity value which is used to verify the transaction is a Merkle root. Similarly, the proof of work defines the difficulty in finding a qualified hash value and the timestamp denotes the time at which the block was added to the chain. With this concept, an unbreakable chain of video segments is generated in a real-time recording of centralized surveillance video data in the proposed blockchain-based scheme. The physical memory occupied by one block of the proposed method is shown in Table 4.1. The total physical memory occupied by a block is thus 114 bytes.

Table 4.1. Block structure of the proposed blockchain in bytes.

ID	Prev-HMAC	VIC	Time-stamp	Pub-Key	v-fname
4	32	32	4	32	16

In Table 4.1, ID is the block index in the blockchain, HMAC value of the previous block is denoted by $Prev{-}HMAC$, and HMAC value of the corresponding video segment is concatenated with the key dK and the concatenation output is encrypted that gives VIC value. Similarly, timestamp is the time at which the block was added, and the ephemeral public key is $Pub{-}Key$. $v{-}fname$ gives the file name of the video segment. The captured video segments from the surveillance camera are compressed and then hashed by an SHA-512 algorithm, and HMAC algorithm is applied to the output that produces the MAC value. The key for HMAC algorithm is dK. To protect the data and the key from the attacker and to verify the integrity, the concatenation of the key and the HMAC value are encrypted using ECC encryption algorithm that gives the VIC value.

To show the efficacy of the blockchain, two block diagrams with and without blockchain in the proposed method are compared in Fig. 4.3 and Fig. 4.4. Note

that only three video segments are considered in the elaboration for brevity. Three video segments are hashed and securely stored in the first step. With the assumption that the second segment is suspicious, the attacker discovered the key Key2 in the second step, which is followed by the data modification and replacement in the next step. Finally the integrity validation is done in the final step.

Fig. 4.3 shows the process of forgery and validation for the video integrity verification method without blockchain. If any frame or images contain suspicious activity in a video segment, a suspect involved in such activity may try to determine a key for that segment. After getting the key, he/she manipulates the video and hash it with HMAC algorithm, after which he/she replaces the original video and its HMAC value with the modified one without leaving any clue of modification, as shown in Fig. 4.3. In contrast, a hash of a block contained in the next block of blockchain in a chained fashion can easily identify and control the violation. Fig. 4.4 shows the overall process of forgery and validation of video segment with blockchain in the proposed method.

As shown in Fig. 4.4, in a similar sort of way the method without blockchain works, determining a single key and changing a single segment is an easy task, however, to get the single segment modified, the entire key should be discovered and the corresponding sequence in the chain should be modified, which is significantly a complex task. Conversely, the single modification can be traced, and lead to an invalid result. Furthermore, since the proposed method is considered to work in a centralized database system, if the key is discovered for any suspicious block, the probability of changing the entire system by calculating the HMAC value for the modified video and replacing all the previous hash values within the blocks in a loop is very high. This violation is resolved by calculating the HMAC value of the block instead of applying hash algorithm alone.

Thus, the block in a blockchain of the proposed method contains HMAC value of the previous block instead of hash value. The key for the HMAC algorithm to hash the block is obtained from the output of randomized hashing applied to the data key dK. The ephemeral public key of ECC is used as a random seed value of randomized hashing. The direct use of dK for block HMAC is not a secure

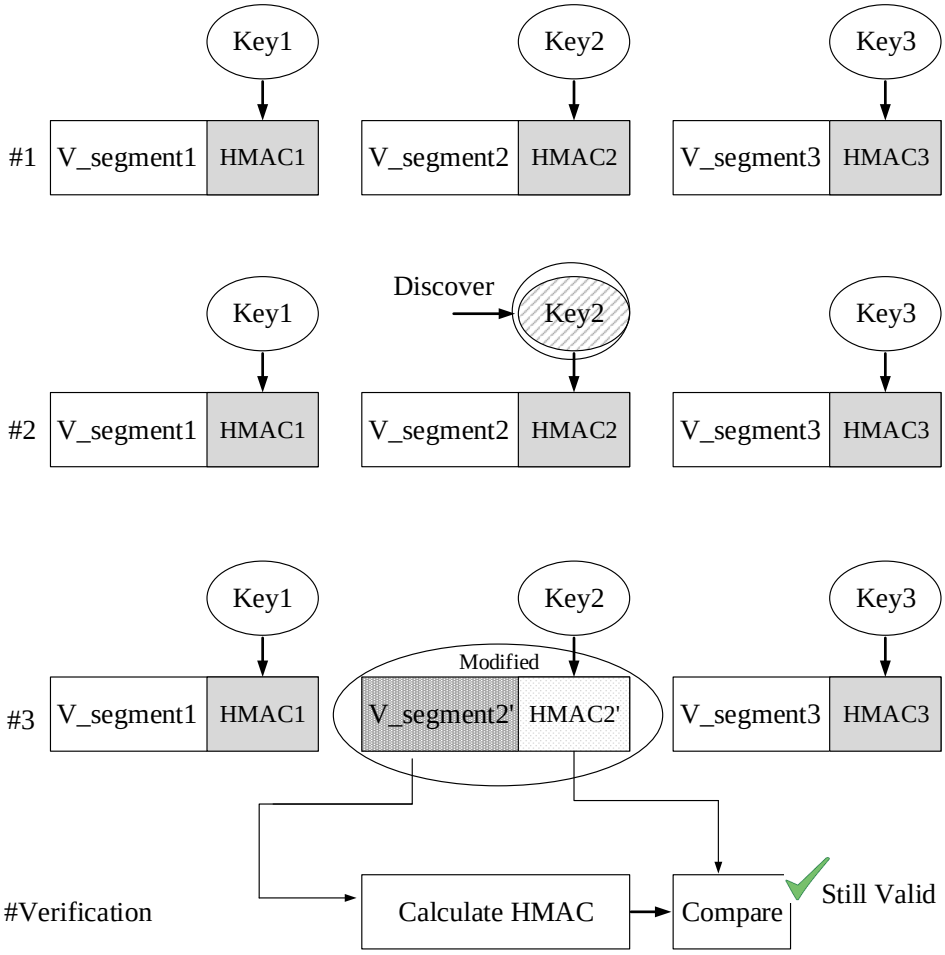

Figure 4.3. A forgery and validation process in video segments without using blockchain in four steps. $Key1$, $Key2$, and $Key3$ are the keys for the hash-based message authentication code (HMAC) algorithm applied to the corresponding video segments $V\_segment1$, $V\_segment2$, and $V\_segment3$ to generate $HMAC1$, $HMAC2$, and $HMAC3$ as output, respectively. $Key2$ in step 2 (diagonal hatching) indicates the determination of the key by the attacker. $V\_segment2'$ and $HMAC2'$ in step 3 indicate the modification and replacement of the corresponding value with the original one.

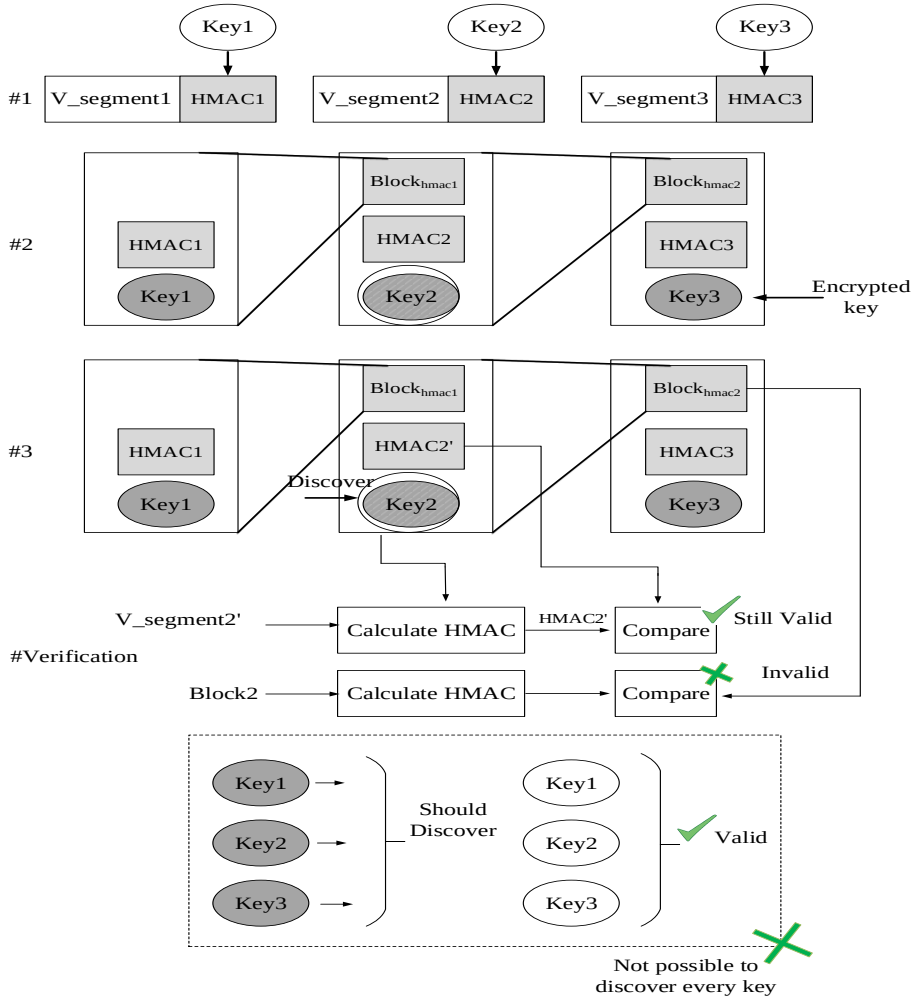

Figure 4.4. A forgery and validation process in video segments using blockchain in four steps. $Key1$, $Key2$, and $Key3$ are the key for HMAC algorithm applied to the corresponding video segments $V\_segment1$, $V\_segment2$, and $V\_segment3$ that generate $HMAC1$, $HMAC2$ and $HMAC3$ as output, respectively. Every segment and their corresponding values are stored in the block, which is connected with each other by applying HMAC to the previous block, denoted as $Block_{hmac1}$ and $Block_{hmac2}$. $Key2$ in step 2 (diagonal hatching) indicates the determination of the key by the attacker. $HMAC2'$ and $V\_segment2'$ in step 3 indicate the modification and replacement of the corresponding value with the original one.

approach and hashing dK to obtain bK also has the security threat of collision attack on bK. Thus, the randomized hashing of dK that prevents collision attack is employed in the proposed method [20]. The use of two random and unique values (dK, and ephemeral public key) for dual purpose in the method expresses the feasibility in memory-limited applications. Furthermore, the hash algorithm followed by the HMAC and encryption increases the level of security as well as the difficulty for the suspect to alter the data in the proposed method.

The overall process of blockchain generation is described as in the following steps:

Hash $(V_{seg}) \rightarrow H_v$. compute hash algorithm to the video segment.

H_v = Hash (V_{seg})

HMAC $(dK, H_v) \rightarrow H_m$. Compute HMAC to the hash value H_v as,

H_m = HMAC (dK, H_v)

ECC Key pair $\rightarrow (Kb, Kv)$. Given an elliptic curve E over Z_n with a number of points that is divisible by the large prime n, choose a base point G on the curve and a random integer $d \in [1,n\text{-}1]$, where d is stored secretly and is known as the private key, i.e. $Kv = d$. The public key $Kb = KvG$ is a point on the curve that is published publicly with other domain parameters E, G, and n.

Encryption secret key $(Kb, k_e) \rightarrow SK_E$. Select a random integer value $k_e \in Z_n$ as an ephemeral private key, and compute $k_{ue} = k_eG$ as an ephemeral public key.

Encryption secret key is $SK_E = k_eKb$

Encryption key $(SK_E) \rightarrow E_k$. Apply a key derivation function to the secret key to generate encryption key, E_k= KDF (SK_E).

Encrypt $(E_k, H_m \parallel dK) \rightarrow VIC$. The VIC value of the video segment can be generated as:

VIC = E $(E_k, (H_m \parallel dK))$

Generate Block Key $(k_{ue}, dK) \rightarrow bK$.

The HMAC key for block is, $bK = Hash_r(k_{ue}, dK)$, where $Hash_r$ is a

randomized hash function.

Block HMAC (bK, ($VIC_{i-1} \parallel timestamp_{i-1} \parallel Index_{i-1} \parallel k_{ue,(i-1)} \parallel V\_seg_{i-1}$) $\rightarrow Block_{hmac}$.

The previous block of the blockchain is hashed with HMAC using the block key bK. Here i-1 indicates the index of the previous block.

$Block_{hmac}$ = HMAC (bK, ($VIC_{i-1} \parallel timestamp_{i-1} \parallel Index_{i-1} \parallel k_{ue,(i-1)} \parallel V\_seg_{i-1}$).

Store on Blockchain ($Block_{hmac}$, VIC, $timestamp$, $Index$, k_{ue}, V_seg) $\rightarrow B_v$ $\rightarrow$ blockchain.

Note that VIC, $timestamp$, $Index$, k_{ue}, and V_seg are the values of the current block B_v in the blockchain.

4.3 Video Integrity Check

For the video integrity validation, firstly the corresponding values from the block of the blockchain are extracted and the VIC value is decrypted with ECC decryption algorithm. The output from the decryption gives the key and the MAC value of the corresponding video. The decrypted key (dK) is used in HMAC algorithm to hash the video segment submitted for integrity verification. The new HMAC value thus generated is compared with the one decrypted from the blockchain. The video segment is considered tampered if the comparison gives negative result otherwise, the state of the modification with replacement of hash value is examined again. For this, the current block of the corresponding video segment is hashed by using the HMAC and compared with the previous HMAC value of the next block in a chain. A positive comparison result specifies the video as untampered, whereas a forgery with modification and replacement of the original hash value is indicated by negative result. The overall process of video integrity validation process is described as follow:

Extract the corresponding block of the video segment submitted for integrity

verification

$$B_v \rightarrow (Block_{hmac}, VIC, timestamp, Index, k_{ue}, V\_seg).$$

Decryption secret key $(Kv, k_{ue}) \rightarrow SK_D$. The secretly stored private key Kv is extracted, and is used to generate the decryption secret key as:

$$SK_D = k_{ue}Kv$$

Decryption key $(SK_D) \rightarrow D_k$. Apply a key derivation function to the secret key to generate decryption key, D_k= KDF (SK_D).

Decrypt $(D_k, VIC) \rightarrow (H_m \parallel dK)$. Decrypt VIC value of the video segment, $(H_m \parallel dK)$= D $(D_k$, VIC)

Split H_m and dK from the concatenated output.

Extract video segment $(V\_seg) \rightarrow V_{seg}$.

The video segment V_{seg} is extracted from the path V_seg of the storage.

Hash $(V_{seg}) \rightarrow H'_v$, compute hash algorithm to the video segment, H'_v= Hash (V_{seg})

HMAC $(dK, H'_v) \rightarrow H'_m$, compute HMAC to the hash value H'_v, H'_m= HMAC (dK, H'_v)

Verify $(H_m, H'_m) \rightarrow ev$.

If $(H_m = H'_m) \rightarrow ev$ = 1 else ev = 0

Case 1: If $(ev$ = 1$) \rightarrow$ **Verify** $(Block_{hmac,j}, Block'_{hmac,j})$

Case 2: Else, the video segment is tampered.

Here j represents the index of next block in blockchain, i.e. j= $Index$ + 1, where $Index$ is the index value of the current block.

Verify $(Block_{hmac,j}, Block'_{hmac,j}) \rightarrow eb_j$,

- **Decryption secret key** $(Kv, k_{ue,j}) \rightarrow SK_{D,j}$: The secretly stored private key Kv is extracted, and is used to generate the decryption secret key of

the $j-th$ block as: $SK_{D,j}$ = $k_{ue,j}Kv$, where $k_{ue,j}$ is an ephemeral public key of the $j-th$ block.

- **Decryption key** $(SK_{D,j}) \rightarrow D_{k,j}$. Decryption key of the $j-th$ block, $D_{k,j}$ = KDF $(SK_{D,j})$

- **Decrypt** $(D_{k,j}, VIC_j) \rightarrow (H_{m,j} \parallel dK_j)$. Decrypt VIC of the $j-th$ block, $(H_{m,j} \parallel dK_j)$ = D $(D_{k,j}, VIC_j)$.

  Split $H_{m,j}$ and dK_j from the concatenated output.

- **Block key** $(k_{ue,j}, dK_j) \rightarrow bK_j$. The HMAC key for hashing previous block from $j-th$ block $bK_j = Hash_r(k_{ue,j}, dK_j)$

- **Block HMAC** (bK_j, (VIC$\parallel$ $timestamp$ $\parallel$ $index$ $\parallel$ k_{ue} $\parallel$ V_seg)) $\rightarrow$ $Block'_{hmac,j}$.

  The current block of the blockchain is hashed with HMAC using the block key bK_j.

  $Block'_{hmac,j}$ = HMAC (bK_j, (VIC$\parallel$ $timestamp$ $\parallel$ $index$ $\parallel$ k_{ue} $\parallel$ V_seg))

- **Verify** If ($Block_{hmac,j}$= $Block'_{hmac,j}$) $\rightarrow$ eb_j = 1 else eb_j =0

  Case 1: If eb_j = 0 $\rightarrow$ Forgery with replacement of the HMAC value with modified one.

  Case 2: Else, the video segment is untampered.

Chapter 5
Experimental Procedure

5.1 Setup and Methodology

To evaluate the proposed method, five publicly available video clips are used from [47], [48], [49], [50], and [51] with 1280×720 pixels. The frame rate of all the video segments is 30 frames per second (fps). The test videos are forged by adding forgery with the commercial video editor, the AVS video editor [52]. Tampering is computed by randomly inserting, copying and pasting, and deleting frames in the video segments. The five test videos, with a number of frames, type of tampering and their tampered sequences utilized in the experiments are listed in Table 5.1. Furthermore, the secure hash algorithm-512 (SHA-512) is used for hashing, and to calculate the HMAC value SHA-256 is used. ECIES is used for key encryption.

The experiments performed on five test videos, tampered by different forgery types, are listed in Table 5.1. The experimental results with successful detection of all forgeries on all test videos indicate that the proposed method is robust irrespective of any forgery type.

Table 5.1. Video files used in the detection test.

Video	Original frames	Tampering type	Tampered frames	Time length	Forgery detection
1	450	Copy-paste	450	15 s	Yes
2	900	Copy-paste	900	30 s	Yes
3	1797	Insert	1800	60 s	Yes
4	3593	Insert	3600	120 s	Yes
5	4867	Delete	4800	160 s	Yes

5.2 Adoption of ECC over RSA

In Fig. 4.1, the architecture of the proposed method, ECC can be changeable with RSA. As described in Section 2.2, RSA or EIGamal asymmetric encryption algorithms based on integer factorization or discrete log problems are less secure than ECC, which is based on the elliptic curve discrete log problems. To ensure the efficacy of the algorithms, the performance of ECC and RSA is analyzed in terms of execution time and security bit level for varying key sizes. The analysis of total execution time (encryption and decryption time) for ECC and RSA with different sized test videos is shown in Fig. 5.1 . The experiments are computed for ten times, and the average values are calculated. As shown in Fig. 5.1, the time consumption by ECC is slightly less for all video sizes than those by RSA.

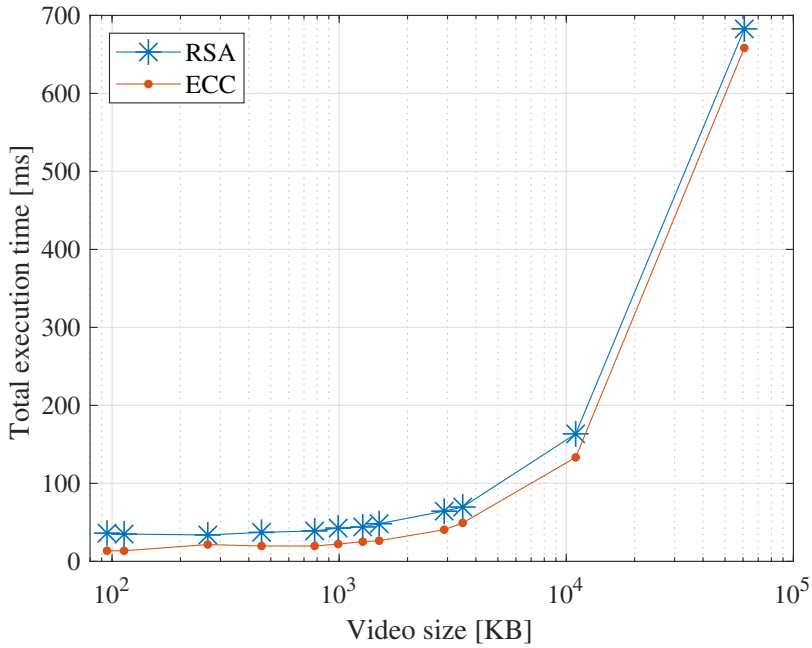

Figure 5.1. Comparison of total execution time (encryption and decryption time) for Rivest–Shamir–Adleman (RSA) and ECC with different test videos of varying frame length.

Similarly, Fig. 5.2 shows a comparison of RSA and ECC based on the security

bit levels for different cryptographic key lengths. As shown in Fig. 5.2, to achieve the same security level, RSA requires a larger key size than ECC. The difference in key sizes between RSA and ECC is significantly large over entire security bit levels. Thus, with the given security level, the computation is faster owing to the smaller key size of ECC, additionally benefited with the lower power and memory consumption [53]. The results in Fig. 5.1 and 5.2 indicate that ECC is thus more efficient and secure than RSA. Hence, the ECC is adopted instead of RSA for the encryption of the key in the proposed method.

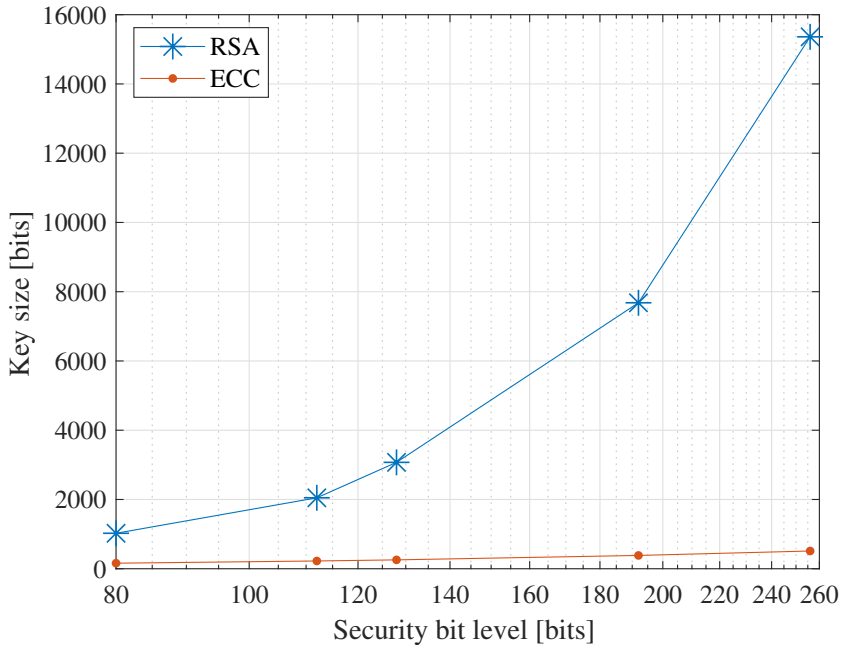

Figure 5.2. Comparable security bit level for cryptography key length.

Three different experimental analysis have been performed for verification of the proposed method: (a) comparison with the state-of-the-art methods based on verification capabilities; (b) performance evaluation based on execution time with and without the blockchain simulated on PC as well as embedded board, and (c) analysis on the increasing size of the blockchain.

Chapter 6
Experimental Results and Discussion

The test videos are firstly encoded with the H.264/AVC [54] video codec by using FFmpeg [55], which is then hashed and added to the blockchain. During video integrity validation, the hashing is performed similarly on the stored bitstream and the generated hash value is then compared with the decrypted hash value stored in a blockchain.

6.1 Comparison with State-of-the-art Methods

For the analysis, the state-of-the-art methods [1][31–34] based on verification capabilities are compared with the proposed method as shown in Table 6.1. The file system integrity-based approaches, video editing tools, cryptography-based schemes, and file structure-based methods are considered for verification capabilities. The methods in Lee et al. [1] [32] and Song et al. [33] are file type and video editing tool dependent while methods in Kwon et al. [35] and Kim et al. [34] are independent of video editing tools or video file types for forgery detection. Furthermore, the image edit forgery cannot be detected by the method in Lee et al. [1][32], however, in contrast, the proposed method and rest other methods Song et al. [33], Kim et al. [34], and Kwon et al. [35] can detect any category of image editing forgeries, such as copy-move or copy-paste. The segment-based proposed method is more efficient than other methods (where every individual frame must be processed to get the integrity value) in terms of memory consumption.

Table 6.1. Comparison of various methods based on integrity verification capabilities.

Method	File type dependent	Tool dependent	Image editing detection	Segment wise detection
Lee[1][31]	Yes	No	No	No
Song[32]	No	Yes	Yes	No
Kim[33]	No	No	Yes	No
Kwon[34]	No	No	Yes	No
Proposed	No	No	Yes	Yes

6.2 Performance Analysis based on Execution Time

6.2.1 Simulation on PC

The experimental analysis is performed on a PC with an Intel® Core™ i7-770K CPU @ 4.20 GHz, and 8 GB RAM. The analysis based on the running time for hash value generation and validation on various length videos with and without blockchain are shown in Fig. 6.1, 6.2, and 6.3, respectively.

As shown in the figures, the running time required by the method with blockchain is 8 ms more than the time required by the method without blockchain for both hashing and integrity verification, which is very nominal. The additional process of hashing for block and the key is the cause for this trivial increase in time for both processes.

6.2.2 Simulation on Ambarella Board

Similarly, to estimate the practicability in real-world scenarios, the analysis based on execution time on an Ambarella board [56] of a 792-MHz ARM® Cortex™-A9 CPU, DDR3 / DDR3L with up to 600 MHz memory, the most widely used embedded board prototypes for the ADR made by Ambarella Inc, is performed. The videos are recorded through inside the vehicle, front windshield, or rear window in the ADR system. The performance of the proposed method is thus analyzed on Ambarella board with and without blockchain based on execution time for various length videos, which is demonstrated in Fig. 6.4.

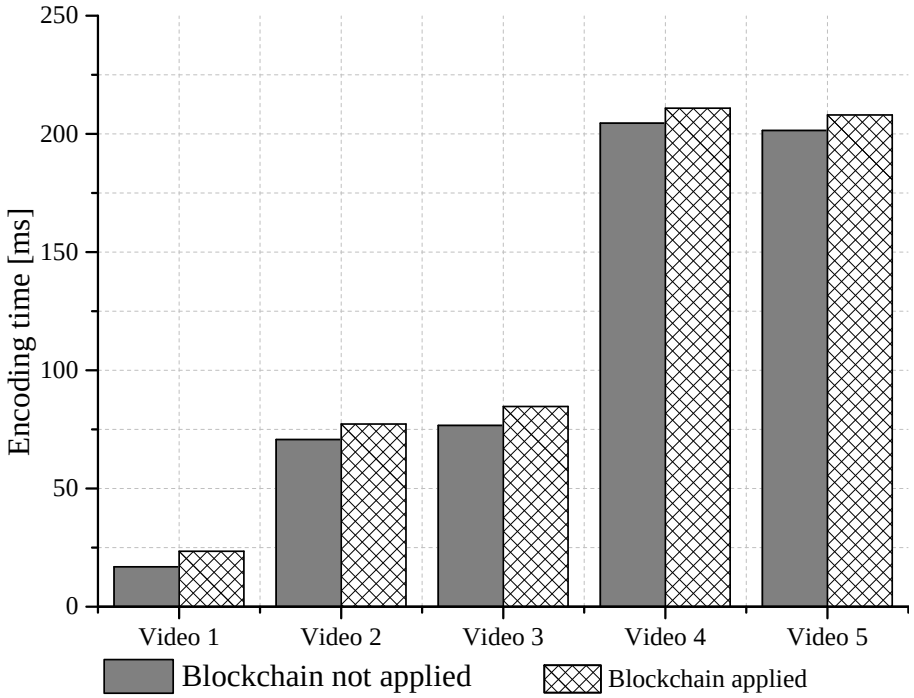

Figure 6.1. Comparison of average encoding time for different test videos of varying frame lengths with and without blockchain.

The result in Fig. 6.4 shows that the increase in size of the video increases the running time of encryption and validation processes for both methods, with and without blockchain. The method with blockchain required a slightly longer time (encryption and decryption) than by the method without blockchain. On average, the execution time increases by 0.5s for all test videos with blockchain, which states that the proposed method with additional blockchain does not produce extra burden/computation time. The execution times range from 1s to 20s, conditioned on the video sizes. With the consideration of execution time and video size, video 3 with 6s execution time, can be practicable in real-world consequences.

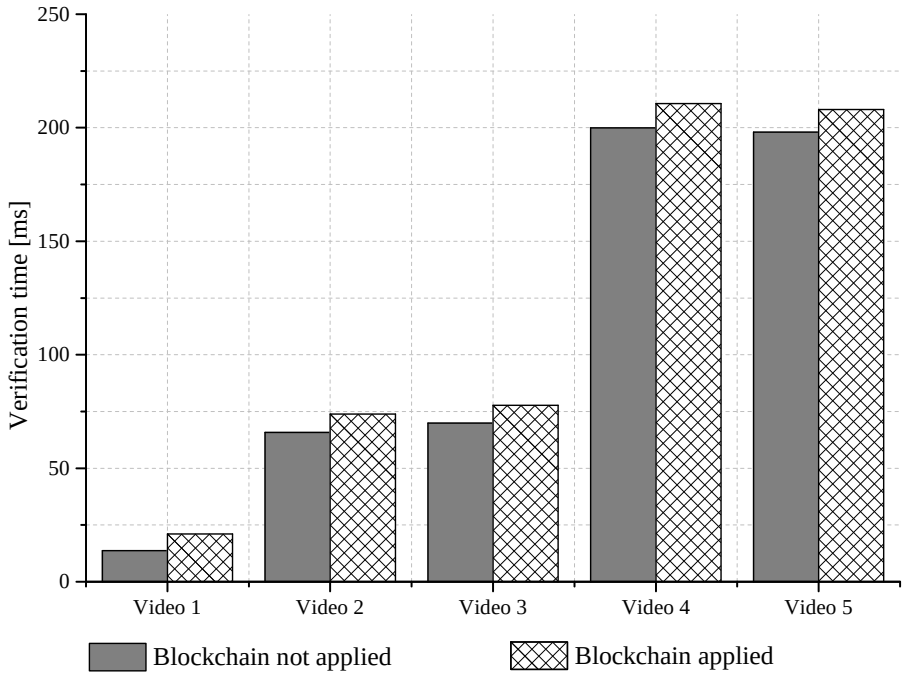

Figure 6.2. Comparison of average verification time for different test videos of varying frame length with and without blockchain.

6.3 Analysis on Increasing Size of Blockchain

The running time for the validation process in the video IVM is an important aspect to consider. To verify the implication of the proposed method with the increase in a number of blocks in the blockchain, the time variation for the method with differently-sized blockchains is analyzed for both tampered and untampered videos, as shown in Fig. 6.5. The experimental result postulates that the proposed method yields a very least increment of time with a corresponding increase in number of blocks in the blockchain. Furthermore, the time difference between the tampered and untampered video is also very nominal (at most 8ms), untampered video draining trivially more time. However, the case is identical if the video is altered and the newly generated hash replaces the original hash; the time required to validate such video is nearly equal to the time required to validate untampered

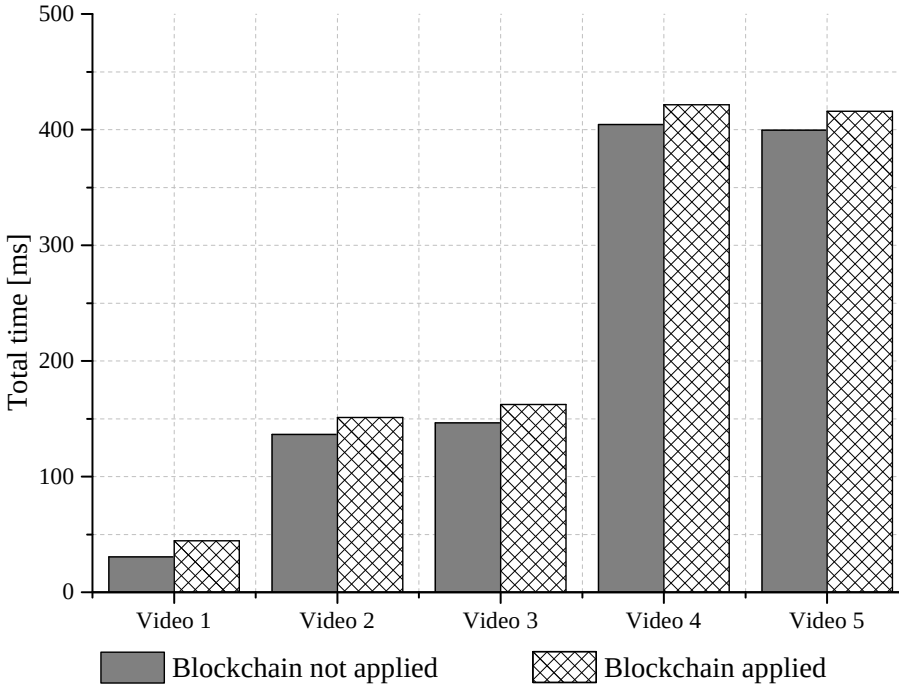

Figure 6.3. Comparison of the average time for different test videos of varying frame length with and without blockchain.

video. For instance, the verification stops if the first step of comparison/matching yields a negative result, and on the contrary, if the result gives a positive outcome, the second step of matching is carried out. Hence, the time consumption for the overall verification process tends to increase with the multiple-step matching process.

6.4 Analysis on Physical Memory Consumption

The evaluation of the physical memory consumed by both the method, with and without blockchain is illustrated in Table 6.2. The method with blockchain requires twice the memory required by the method without blockchain. Although the memory occupied by the proposed method with blockchain is double, the method is more suitable owing to the security it provides, which will be enlight-

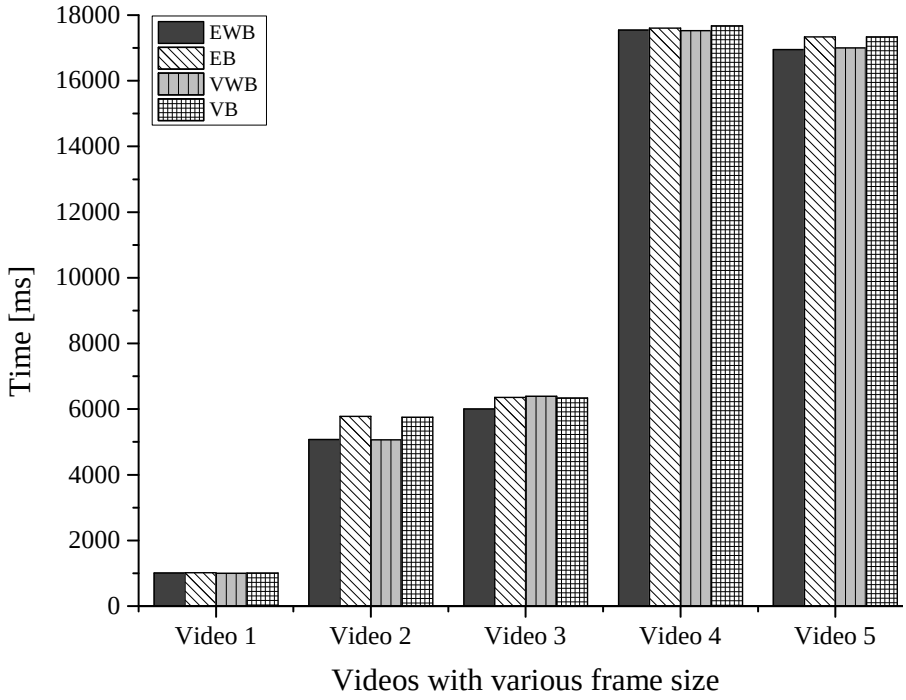

Figure 6.4. Comparison of execution time for different test videos of varying frame length with and without blockchain on the Ambarella board. EWB: Encryption without blockchain, EB: Encryption with blockchain, VWB: Validation without blockchain, VB: Validation with blockchain.

ened in the security analysis in Chapter 7. This shows the compromise of memory size with the security it provides.

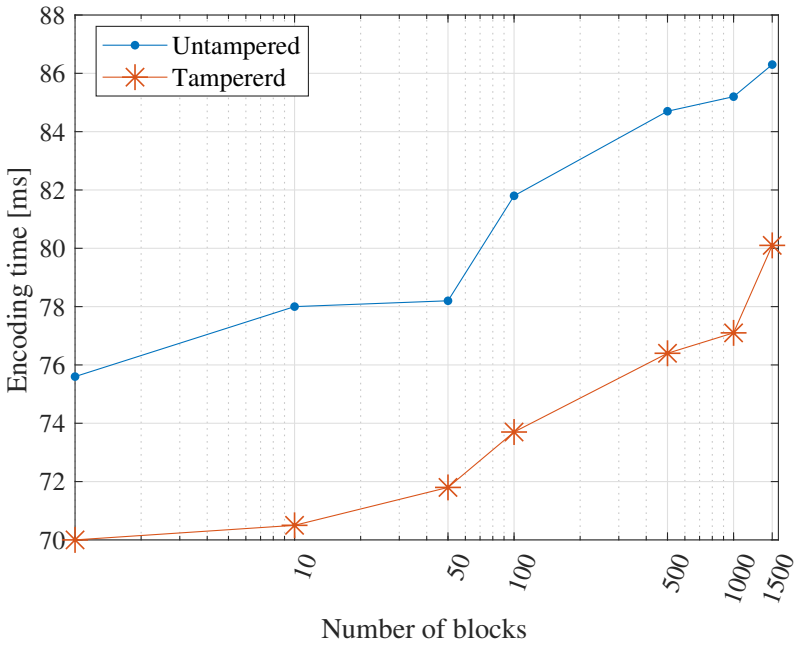

Figure 6.5. Comparison of average encoding time along with a change in the size of blockchain with tampered and untampered video segments.

Table 6.2. Comparison of physical memory consumed by the hash output of one video segment with and without blockchain [Unit: bytes].

Output	With Blockchain	Without Blockchain
ID	4	0
Previous HMAC	32	0
VIC value	32	32
Ephemeral public key	32	32
Timestamp	0	0
Filename	16	0
Total	128	64

Chapter 7
Security Analysis

As mentioned in Section 4.2, the chained security acquired by the blockchain can trace a modification in the video content even if the key is discovered by the intruder. Similarly, the destruction of the video content and removal of the traces of alteration by running the hash function in a loop is resolved by applying the HMAC algorithm with a unique key to every block. Furthermore, the proposed method has less effect of collision attacks owing to the use of the HMAC algorithm for both data and the block instead of using standalone hash algorithms (which is less collision resistive [7], [57]). However, the method is dependent on the key applied in the HMAC algorithm. The likelihood that the violation of video integrity is increased, if the key is revealed. Thus, as the HMAC key is encrypted using the ECC encryption algorithm, the attack on the ECC key is the possible security threat. Hence, the legitimacy and possibility of such an attack is estimated for the security analysis.

If the private key of the ECC is assumed to keep secret and not exposed, the exponential time required to solve elliptic curve discrete logarithmic problem and the unique ephemeral private key utilization for each video segment makes the estimation of the private key k given public key k_p very difficult [58]. Moreover, an enormous amount of time is required to solve the problem. Likewise, the computational complexity to perform encryption and decryption is considered as a metric to determine the strength of any cryptographic algorithm [19]. Computing encryption in RSA with a fixed public exponent e is given by $M^e mod N$ and has a time complexity of $O(loga(N)^2)$. Here, M denotes the input message and the product of two prime numbers is represented by N. Similarly, the time complexity for the decryption of the ciphertext C given private exponent d, $C^d mod N$, is given by $O(loga(N)^3)$, where the size of d in bits is in proportion to N. However, for a given point P on an elliptic curve E over a finite field F_q and integer x, the point xP on the same curve can be computed in $O((logx)(logq)^3)$ bits opera-

tions. Here, finite field F_q has q elements, q being very large prime power given by $q = p^r$, where p is a prime number and r is a very large positive integer value. The basic phenomenon of encryption and decryption in ECC is the point multiplication xP, thus to compute the ECC encryption or decryption, the complexity of $O((logx).(logq)^3)$-bit operations are required. Furthermore, if the cryptanalyst requires a large amount of time to break the algorithm, the cryptographic algorithm is considered as a computationally secure algorithm, i.e., the complexity in terms of difficulty. The difficulty for encryption algorithms (such as RSA) based on solving an integer factorization problem is given by $O(\sqrt{d})$, where d is a private exponent [19]. The computational difficulty for discrete logarithm problems (like EIGalam algorithm and digital signature algorithm (DSA)) is identical to the difficulty for integer factorization. However, substantially larger computational complexity is required to solve an elliptic curve discrete log problem than solving an integer factorization or discrete log problem. The reason for this computational difficulty is due to the large prime power q, which is given by $O(\sqrt{q})$ [19].Thus, it requires large amount of time to break the ECC key providing high level of security.

Since the proposed method utilizes ECC in every block of the blockchain, for n blocks of blockchain in the proposed method, the time complexity is given by $O(n\sqrt{q})$. However, the complexity equivalent to ECC of $O(\sqrt{q})$ is required for the IVM without blockchain. Hence, the increase in computational complexity with the introduction of blockchain in an IVM can be concluded as higher-level security.

Furthermore, since the HMAC of the block is stored without encryption in the blockchain, the likelihood of an attack on the HMAC is high. The security of HMAC functions is reliant on the security strength of the underlying hash function [8], e.g., SHA-256 [59]. In addition, the likelihood of attacks on HMAC is equivalent to the likelihood of attacks on the underlying hash function.

1. Brute-force attack:

   For this attack, the initialization vector (IV) of the hash function is replaced with a secret random value. The brute force attack on the key is required

for this attack. However, to obtain a required key for one block, an effort on the order of 2^n (n is the size of hash digest) is required. That means, to alter the single block of the blockchain all the key (of subsequent blocks in the blockchain) must be determined. If the number of subsequent keys is denoted with m, the brute force attack requires an effort of m times 2^n. Thus, this attack is infeasible.

2. Collision attack:

   Determination of two messages M and M' having same hash values $H(M) = H(M')$ is known as a collision attack. Thus, the attacker tries to find a message pair that produces the same hash. To attack a standalone hash function like SHA-256, the effort of a level equal to $2^{(n/2)}$ is required for the attacker, where n gives the size of the hash digest in bits (which is 2^{128} for SHA-256 of n=256). As the algorithm and the value of the IV is public and known to everyone, it is easier to calculate the hash value for several message pairs to find a collision. However, in case of HMAC, the intruder must know the key to generate a hash code, for which he/she must observe the sequence of several (message, code) pairs generated with the same key. For a code/digest length of 128 bits,($2^{(n/2)}$) 2^{64} codes generated with the same key must be learned [59]. Since, the proposed method utilizes a unique key for each video segment, learning the sequence of message/code pairs to obtain the key is very difficult even though a dedicated computing facility is provided.

Thus, the attack on the HMAC value of the block in a blockchain is infeasible.

Furthermore, with the properties of one-way function of the hashing algorithm [60], finding the key dK is difficult even if the key bK is discovered. However, the key can be vulnerable to the hash collision attack. Despite this, the randomized hashing of the key in the proposed method helps to alleviate such an attack.

Chapter 8
Conclusion and Future Work

In this thesis, a novel method for video IVM based on blockchain in centralized video data is proposed. In the proposed method, the HMAC algorithm is used in the blockchain for hashing and ECC encryption algorithm is used to encrypt the key and the hash value. Moreover, the VIC value is generated from the encryption and stored in the blockchain with chronologically-chained fashion. The stored integrity code (VIC) value is then used to validate the integrity of the video segment by comparing it with the newly-generated code of the corresponding video. The experimental results from the implementation on PC environment, as well as on the embedded system show that the proposed method is more robust against forgery detection compared to other conventional methods. Futhermore, the analysis of the computational complexity states that the proposed method produces a marginal increase in execution time, and guarantees a high level of security than other conventional methods, with the blockchain additionally used in the method. The performance analysis with the increase in size of the blockchain, i.e., the number of blocks contained in the blockchain, demonstrates that the proposed method yields very less overhead due to the increase in number of blocks that rationalizes the applicability in real-world applications. Finally, the security analysis on the proposed IVM shows that the method is more robust against several attacks, which has time complexity of $O(n\sqrt{q})$ ensuring higher level of security.

For future work, the investigation of the robustness of the proposed blockchain-based IVM in high-performance computing power is desirable. Such high computing power can be for example the quantum computer. In addition, it is worthwhile to investigate if the proposed method can be relevant to any other type of multimedia data such as images, audios, and documents, etc.

Bibliography

[1] C. Lee, J. Lee, Y. Pyo, and H. Lee, "Broken integrity detection of video files in video event data recorders." *KSII Transactions on Internet & Information Systems*, vol. 10, no. 8, 2016.

[2] X. Nie, Y. Yin, J. Sun, J. Liu, and C. Cui, "Comprehensive feature-based robust video fingerprinting using tensor model," *IEEE Transactions on Multimedia*, vol. 19, no. 4, pp. 785–796, 2016.

[3] C.-L. Chou, H.-T. Chen, and S.-Y. Lee, "Pattern-based near-duplicate video retrieval and localization on web-scale videos," *IEEE Transactions on Multimedia*, vol. 17, no. 3, pp. 382–395, 2015.

[4] M. Asikuzzaman, M. J. Alam, A. J. Lambert, and M. R. Pickering, "Robust dt cwt-based dibr 3d video watermarking using chrominance embedding," *IEEE Transactions on Multimedia*, vol. 18, no. 9, pp. 1733–1748, 2016.

[5] L. Yu, H. Wang, Q. Han, X. Niu, S.-M. Yiu, J. Fang, and Z. Wang, "Exposing frame deletion by detecting abrupt changes in video streams," *Neurocomputing*, vol. 205, pp. 84–91, 2016.

[6] F. X. Olleros and M. Zhegu, *Research handbook on digital transformations*. Edward Elgar Publishing, 2016.

[7] H. Krawczyk, R. Canetti, and M. Bellare, "Hmac: Keyed-hashing for message authentication," 1997.

[8] M. U. Arshad, A. Kundu, E. Bertino, A. Ghafoor, and C. Kundu, "Efficient and scalable integrity verification of data and query results for graph databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 5, pp. 866–879, 2018.

[9] M. Singh and S. Kim, "Introduce reward-based intelligent vehicles communication using blockchain," in *2017 International SoC Design Conference (ISOCC)*. IEEE, 2017, pp. 15–16.

[10] N. Z. Aitzhan and D. Svetinovic, "Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 5, pp. 840–852, 2016.

[11] S. Singh and N. Singh, "Blockchain: Future of financial and cyber security," in *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)*. IEEE, 2016, pp. 463–467.

[12] A. S. Bruyn, "Blockchain an introduction," *University Amsterdam*, pp. 1–43, 2017.

[13] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, "Medrec: Using blockchain for medical data access and permission management," in *2016 2nd International Conference on Open and Big Data (OBD)*. IEEE, 2016, pp. 25–30.

[14] L. D. Singh and K. M. Singh, "Image encryption using elliptic curve cryptography," *Procedia Computer Science*, vol. 54, pp. 472–481, 2015.

[15] D. Hankerson, A. J. Menezes, and S. Vanstone, "Guide to elliptic curve cryptography," *Computing Reviews*, vol. 46, no. 1, p. 13, 2005.

[16] L. E. Dickson, *Linear groups: With an exposition of the Galois field theory*. Courier Corporation, 2003.

[17] V. G. Martínez, L. H. Encinas, and C. S. Ávila, "A survey of the elliptic curve integrated encryption scheme," *ratio*, vol. 80, no. 1024, pp. 160–223, 2010.

[18] L. D. Singh and K. M. Singh, "Implementation of text encryption using elliptic curve cryptography," *Procedia Computer Science*, vol. 54, pp. 73–82, 2015.

[19] M. Calabresi, "An introduction to elliptic curve cryptography," 2016.

[20] S. Halevi and H. Krawczyk, "Strengthening digital signatures via randomized hashing," in *Annual International Cryptology Conference*. Springer, 2006, pp. 41–59.

[21] S. Halevi and W. Shao, "Implementing the halevi–krawczyk randomized hashing scheme, 2007."

[22] H. Ravi, A. V. Subramanyam, G. Gupta, and B. A. Kumar, "Compression noise based video forgery detection," in *2014 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2014, pp. 5352–5356.

[23] Y. Su, W. Nie, and C. Zhang, "A frame tampering detection algorithm for mpeg videos," in *2011 6th IEEE Joint International Information Technology and Artificial Intelligence Conference*, vol. 2. IEEE, 2011, pp. 461–464.

[24] X. Jiang, W. Wang, T. Sun, Y. Q. Shi, and S. Wang, "Detection of double compression in mpeg-4 videos based on markov statistics," *IEEE Signal processing letters*, vol. 20, no. 5, pp. 447–450, 2013.

[25] C.-C. Hsu, T.-Y. Hung, C.-W. Lin, and C.-T. Hsu, "Video forgery detection using correlation of noise residue," in *2008 IEEE 10th workshop on multimedia signal processing*. IEEE, 2008, pp. 170–174.

[26] W. Wang and H. Farid, "Exposing digital forgeries in video by detecting double mpeg compression," in *Proceedings of the 8th workshop on Multimedia and security*. ACM, 2006, pp. 37–47.

[27] A. V. Subramanyam and S. Emmanuel, "Pixel estimation based video forgery detection," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 3038–3042.

[28] P. He, X. Jiang, T. Sun, and S. Wang, "Double compression detection based on local motion vector field analysis in static-background videos," *Journal of Visual Communication and Image Representation*, vol. 35, pp. 55–66, 2016.

[29] D. Vazquez-Padin, M. Fontani, T. Bianchi, P. Comesaña, A. Piva, and M. Barni, "Detection of video double encoding with gop size estimation," in *2012 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2012, pp. 151–156.

[30] J. Song, Y. Yang, Z. Huang, H. T. Shen, and J. Luo, "Effective multiple feature hashing for large-scale near-duplicate video retrieval," *IEEE Transactions on Multimedia*, vol. 15, no. 8, pp. 1997–2008, 2013.

[31] V. E. Liong, J. Lu, Y.-P. Tan, and J. Zhou, "Deep video hashing," *IEEE Transactions on Multimedia*, vol. 19, no. 6, pp. 1209–1219, 2016.

[32] S. Lee, J. E. Song, W. Y. Lee, Y. W. Ko, and H. Lee, "Integrity verification scheme of video contents in surveillance cameras for digital forensic investigations," *IEICE TRANSACTIONS on Information and Systems*, vol. 98, no. 1, pp. 95–97, 2015.

[33] 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, pp. 1–7, 2016.

[34] M. Kim and K.-Y. Kim, "Data forgery detection for vehicle black box," in *2014 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2014, pp. 636–637.

[35] H. Kwon, S. Kim, and H. Lee, "Sigmata: Storage integrity guaranteeing mechanism against tampering attempts for video event data recorders," in *7th International Multi-Conference on Complexity, Informatics and Cybernetics, IMCIC 2016 and 7th International Conference on Society and Infor-*

mation Technologies, ICSIT 2016. International Institute of Informatics and Systemics, IIIS, 2016.

[36] Z. Yong-Xia and Z. Ge, "Md5 research," in *2010 Second International Conference on Multimedia and Information Technology*, vol. 2. IEEE, 2010, pp. 271–273.

[37] B. Preneel, A. Bosselaers, and H. Dobbertin, "The cryptographic hash function ripemd-160," 1997.

[38] D. Eastlake and P. Jones, "Us secure hash algorithm 1 (sha1)," 2001.

[39] R. A. Dobre, R. O. Preda, C. C. Oprea, and I. Pirnog, "Authentication of jpeg images on the blockchain," in *2018 International Conference on Control, Artificial Intelligence, Robotics & Optimization (ICCAIRO)*. IEEE, 2018, pp. 211–215.

[40] S. M. Soliman, B. Magdy, and M. A. A. El Ghany, "Efficient implementation of the aes algorithm for security applications," in *2016 29th IEEE International System-on-Chip Conference (SOCC)*. IEEE, 2016, pp. 206–210.

[41] S. Ghimire and B. Lee, "An architecture of integrity check for accident video data recording system," in *Proceedings of KIIT Summer Conference*, 2018, pp. 419–422.

[42] ——, "Data integrity verification algorithms and performance evaluation for vehicle accident data recording system," *KSAE 2018 Annual Autumn Conference and Exhibition*, pp. 788–791, 2018.

[43] "Taiwan Digital Time-lapse Interval Video recorder CCTV Security Camera-1 — FOREVER PLUS CORP." https://foreverplus.en.taiwantrade.com/product/digital-time-lapse-interval-video-recorder-cctv-security-camera-1-582444.html, accessed: 2018-07-17.

[44] M. Bellare, R. Canetti, and H. Krawczyk, "Keying hash functions for message authentication," in *Annual international cryptology conference*. Springer, 1996, pp. 1–15.

[45] M. Calabresi, "Description of sha-256, sha-384 and sha-512," 2016.

[46] B. Gipp, J. Kosti, and C. Breitinger, "Securing video integrity using decentralized trusted timestamping on the bitcoin blockchain." in *MCIS*, 2016, p. 51.

[47] "Real yellow car," https://www.youtube.com/watch?v=o2YNaYcwdbA., accessed: 2018-12-17.

[48] "Nature in 30 seconds," https://www.youtube.com/watch?v=MHna8CzxPLk., accessed: 2018-12-17.

[49] "1 min of Nature Footage - 4K (Ultra HD)," https://www.youtube.com/watch?v=WLKJnHu0GC4., accessed: 2018-12-17.

[50] "4K Video Ultra HD - Epic Footage," https://www.youtube.com/watch?v=od5nla42Jvc., accessed: 2018-12-17.

[51] "029- Realistic Beautiful Flower Painting Timelapse by Artistbrownlion - SATISFYING VIDEO - 2.5 Min," https://www.youtube.com/watch?v=2g8bSnNYE, accessed: 2018-12-17.

[52] "AVS VideoEditor 9.0, - easy video editing software for Windows," https://www.avs4you.com/avs-video-editor.aspx, accessed: 2018-12-17.

[53] Gemalto, "Benefits of elliptic curve cryptography," 2012.

[54] I. E. Richardson, *The H. 264 advanced video compression standard*. John Wiley & Sons, 2011.

[55] “FFmpeg ,” https://www.ffmpeg.org/, accessed: 2018-12-17.

[56] “Ambarella — Embedded Computer Vision SoCs.” https://www.ambarella.com/, accessed: 2018-12-17.

[57] “SHA-1 Broken - Schneier on Security.” https://www.schneier.com/blog/archives/2005/02/, accessed: 2018-12-17.

[58] B. Jana and J. Poray, “A performance analysis on elliptic curve cryptography in network security,” in *2016 International Conference on Computer, Electrical & Communication Engineering (ICCECE)*. IEEE, 2016, pp. 1–7.

[59] S. Williams, “Cryptography and network security: Principles and practices,” *ed: Pearson Education*, 2005.

[60] Q. Dang, R. M. Blank, C. F. E. Barker *et al.*, “Nist special publication 800-107 revision 1 recommendation for applications using approved hash algorithms,” 2012.

Acknowledgement

I would like to express my deep sense of gratitude to my advisor Prof. Bumshik Lee for his encouragement, invaluable support, suggestions, and precious guidance throughout my research work. I am really fortunate to have a supervisor who constantly encouraged and monitored my work and responded to my queries so promptly. I am thankful to all the Professors of Information and Communication Engineering Department for providing a suitable platform to make this research success.

I would like to extend my gratitude to the committee members, Prof. Jae-Young Pyun and Prof. Um Tai Won, for their insightful guidance and suggestions. Likewise, I am highly indebted to all anonymous reviewers at different leading journals and reputed conferences for their time and efforts in giving valuable feedbacks for the manuscripts. Their suggestions have raised the quality of the manuscripts, which are the foundation of this thesis.

I would like to show my appreciations to all my fellow lab mates of Multimedia and Image Processing Lab for their extensive professional and personal guidance and their valuable suggestions regarding thesis work.

Finally, my deepest gratitude to my caring husband, Santosh, who is always behind me for unending support and encouragement. As ever, I would like to express my profound gratitude to the role models of my life, my parents and my family, whose love and guidance are with me in whatever I pursue. I will be grateful forever for your love and support.

Publications

- Ghimire Sarala, and Bumshik Lee. ”An Architecture of Integrity Check for Accident Video Data Recording System.” Proceedings of KIIT Conference. 2018.
- Ghimire Sarala, and Bumshik Lee. ”Data Integrity Verification Algorithms and Performance Evaluation for Vehicle Accident Data Recording System.” Korean Society of Automotive Engineers Autumn Conference and Exhibition (2018): 788-791.
- Ghimire Sarala, Jae Young Choi, and Bumshik Lee. ”Using Blockchain for Improved Video Integrity Verification.” IEEE Transactions on Multimedia (2019).