Applied Computing and Informatics xxx (2017) xxx–xxx

Contents lists available at ScienceDirect

Applied Computing and Informatics

journal homepage: www.sciencedirect.com

A method for verifying integrity & authenticating digital media

Martin Harran [a], William Farrelly [a], Kevin Curran [b,*]

[a] Letterkenny Institute of Technology, Donegal, Ireland
[b] Ulster University, Derry, United Kingdom

ARTICLE INFO

Article history:
Received 6 March 2017
Revised 22 May 2017
Accepted 24 May 2017
Available online xxxx

Keywords:
Secure authentication
Integrity
Digital certificates
Security
Network security

ABSTRACT

Due to their massive popularity, image files, especially JPEG, offer high potential as carriers of other information. Much of the work to date on this has focused on stenographic ways of hiding information using least significant bit techniques but we believe that the findings in this project have exposed other ways of doing this. We demonstrate that a digital certificate relating to an image file can be inserted inside that image file along with accompanying metadata containing references to the issuing company. Notwithstanding variations between devices and across operating systems and applications, a JPEG file holds its structure very well. Where changes do take place, this is generally in the metadata area and does not affect the encoded image data which is the heart of the file and the part that needs to be verifiable. References to the issuing company can be inserted into the metadata for the file. There is an advantage of having the digital certificate as an integral part of the file to which it applies and consequently travelling with the file. We ultimately prove that the metadata within a file offers the potential to include data that can be used to prove integrity, authenticity and provenance of the digital content within the file.

1. Introduction

More of today's communications are moving to exclusively digital format with no hard copy backups being kept. This has implications regarding integrity, authentication and provenance in various areas such as litigation where it is necessary that both sides are satisfied with the integrity of digital evidence or in the insurance industry where claims can succeed or fail on very detailed terms and conditions; it may be necessary to know exactly what the terms and conditions were at the time the policy was created [1]. There is also a need to verify the terms of a contract that applied when the contract was agreed and establish dates of original creation when copyright issues arise regarding digital content [2]. Likewise, real world requirements exist to establish prior knowledge before signing Non-Disclosure Agreements. An important aspect of this is that the raw data content of a file can be verified even when the metadata is altered due to the file moving across operating systems or devices.

Widespread and easily available tools have become common for video synthesis. It is important to ensure the authenticity of video uses such as in courts of law, surveillance systems, advertisements and the movie industry. There is therefore a large body of research in the ares of video authentication and tampering detection techniques [3–5]. There is also research using Darwin Information Typing Architecture in protecting the integrity of digital publishing applications [6].

Although there are many tools available to create, encrypt and extract data, there is relatively little available in the area of establishing integrity, authentication and provenance. For instance, [7] creates and issues a certificate containing a SHA 256 hash value of the submitted media file along with the user details and a timestamp. That certificate is in turn verified by a digital certificate issued by leading certification authority Comodo. The problem here however is that the original file and the certification are separate entities and could easily become separated during distribution or circulation of the file. In general, wrapping the file and its certification inside an outer envelope is no real guarantee that the field and its certification will remain together. Inserting the certification into the file is the only apparent reliable method and is the approach used by leading companies such as Microsoft and Adobe for digital signing of Office and PDF documents [8]. A limitation with inserting the certificate inside the file is that it

* Corresponding author.
E-mail address: kj.curran@ulster.ac.uk (K. Curran).

Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.aci.2017.05.006
2210-8327/© 2017 The Authors. Production and hosting by Elsevier B.V. on behalf of King Saud University.

must follow the metadata specification set out for the particular file type which means that distinct software applications or components have to be developed for every file type.

This research outlines a method in which integrity, authentication and provenance can be established for raw data within a file even when the metadata attached to it has changed. We also achieve this in a manner in which a digital proof stamp can be made visual to help achieve virality for the end users.

2. Digital fingerprinting

Digital fingerprinting is based on the use of a mathematical function to produce a numerical value where the function takes an arbitrary length of data as its input and outputs a numerical value of a specified length; 128 bits (16 bytes), 256 bits (32 bytes) and 512 bits (64 bytes) are typical lengths. The output value of such a function is generally referred to as a *hash value* or simply *hash*. In order to be useful, the function used to produce a digital fingerprint must be quick to use but in order to be trustworthy for authentication purposes, it must also meet certain security requirements [9]. For this reason, digital fingerprinting uses *cryptographic hash functions*; the authors describe the general security properties of these functions as preimage resistance, second preimage resistance (or weak collision resistance) and collision resistance (or strong collision resistance) where:

- Preimage resistance means that for a hash function H and an output value of z, it is *computationally infeasible*[1] to find an input value m such that $z = H(m)$. This property is also known as *onewayness*.
- Second preimage resistance means that given an input m_1, it is computationally infeasible to find a different message m_2 where $H(m_1) = H(m_2)$. This is also known as *weak collision resistance*.
- Collision resistance means that it is computationally infeasible to find two different inputs where m_1 *and* m_2 where $H(m_1) = H(m_2)$. This is also known as *strong collision resistance*.

Ref. [10] stated that collision resistance implies second preimage resistance but does not guarantee preimage resistance. Ref. [11] pointed out that that the validity of those claims is limited to specific definitions of the various terms. The subtleties involved, however, have no impact on this project as only second preimage resistance applies in regard to the use of digital fingerprints for file authentication - a potential attacker has access to both the original input (the file) and the output (the hash value for the file).

A wide range of algorithms have been developed to meet the requisite security requirements as well as having the underlying performance requirement of computational speed. Two of the most commonly encountered functions are MD5 and the SHA family of functions. MD5 (Message Digest algorithm 5) was developed by Ref. [12] who had previously developed earlier versions MD2 – MD4. It became extremely popular following its release but beginning in 1996, vulnerabilities were increasingly identified in regard to collision resistance. Ref. [13] stated that although practical applications were not yet threatened, it "comes rather close" and advised users to move to other algorithms. Eight years later, Ref. [14] announced collisions based on the full hash. Four years after that, Ref. [15] based MD5 collision vulnerabilities to create a rogue digital certificate which would be accepted by all common browsers. The credibility of MD5 was effectively terminated in December that year by the release of a Vulnerability Notice by US-CERT [16] which stated that "it [MD5] should be considered cryptographically broken and unsuitable for further use". Although MD5 is still adequate for checking file consistency and other non-secure applications, the known weaknesses should have effectively ended its use for digital signatures. It is still in widespread use, however, as late as 2012 when it was used to fake a Microsoft digital signature in the *FLAME* malware attack [17]. Rivest has produced an MD6 version but after submitting it to the NIST open competition for SHA-3, he withdrew it stating that it was not ready for use [18]. Following the decline of MD5, the predominant algorithms in use today are members of the SHA family.

[1] An expression used in cryptology to signify that, given enough time and resources, it may be theoretically possible to decrypt the result of a cryptographic function but the time and resources actually available make it impractical in any meaningful way.

2.1. Secure hash algorithm

Secure Hash Algorithm (SHA) is a family of cryptographic hash functions developed by the US National Security agency (NSA) and published as standards by the US National Institute of Science and Technology (NIST). It is the required algorithm for secure applications used by US government agencies. An important feature of the SHA algorithms is that they implement an *avalanche effect* which means that a minor change to the input cascades into a major change in the output value.

The first version, SHA-1 produces a 160-bit value and was released in 1993; it was, however, withdrawn shortly after publication due to an undisclosed vulnerability and a modified version was released two years later [19]. In 2002, NIST published the SHA-2 family of functions. Unlike SHA-1 with hash size fixed at 160 bits, SHA-2 is offered in six versions producing a range of output sizes from 224 to 512 bits of which the most widely used are SHA-256 and SHA-512. Like MD5 and SHA-1, the SHA-2 functions are based on the Merkle–Damgård construction. The algorithm used for SHA-256 is:

1. A 256-bit buffer is created, made up of 8×32-bit words which are initialised with the first 32 bits of the fractional parts of the square roots of the first 8 primes.
2. A 64 element table of constants is prepared using the first 32 bits of the fractional parts of the cube roots of the first 64 primes

The input is padded with an initial bit '1' and the length of the original input expressed as a 64-bit integer, separated by the required number of zeros required to make the message length, including the padding, a multiple of 512 bits.

3. Each 512-bit block is processed through 64 rounds where each round involves a series of operations comprised of bitwise operations and modular addition.
4. The value of the buffer on completion of each block is the initial value for the following block; at the end of the final block, the buffer contains the hash value.

Despite offering better security and faster performance, SHA-2 was only adopted slowly over the next three years with SHA-1 remaining in extensive use. The main contributory reasons were that SHA-2 was not supported on systems using Windows XP (SP2) or older, there was no perceived urgency as no collisions had yet been found in SHA-1 and SHA-256 is about 2.2 times slower than SHA-1 though some of that drop can be reduced by a move to SHA-512 on 64-bit hardware and operating systems [20].

Ref. [21] revealed collision attacks on the full SHA-1 function. This work did not completely undermine the practical reliability of the function – it was a *strong collision attack* which did not impact on the preimage – but the following year, NIST instructed