ENSURING MEDIA INTEGRITY ON THIRD-PARTY INFRASTRUCTURES

Jana Dittmann[1], Stefan Katzenbeisser[2], Christian Schallhart[2], Helmut Veith[2]
[1]*Otto-von-Guericke Universität Magdeburg, Germany,* jana.dittmann@iti.cs.uni-magdeburg.de [2] *Technische Universität München, Germany* {katzenbe,schallha,veith}@in.tum.de

Abstract: In many heterogeneous networked applications the integrity of multimedia data plays an essential role, but is not directly supported by the application. In this paper, we propose a method which enables an individual user to detect tampering with a multimedia file without changing the software application provided by the third party. Our method is based on a combination of cryptographic signatures and fragile watermarks, i.e., watermarks that are destroyed by illegitimate tampering. We show that the proposed system is provably secure under standard cryptographic assumptions.

Key words: Digital Signatures, Media Integrity

1. INTRODUCTION

Commercial transactions as well as private communication are increasingly performed using electronic infrastructures—most importantly the WWW. These applications are typically built using standard software and existing third-party network infrastructures (such as standard Web browsers, standardized network protocols or off-the-shelf video streaming tools). This infrastructure is commonly used to transmit multimedia information (like video or audio files or facsimiles). The integrity of these data can be of crucial importance and may have legal and commercial consequences. Unfortunately, the level of security a user can achieve depends on the services provided, as the users themselves typically cannot change the architecture of the overall system. In this paper, we describe a method for ensuring the integrity and authenticity of multimedia data which overcomes this limitation. We use a combination of cryptographic signatures and digital watermarks to encode authentication information in the media objects themselves, which allows us to completely decouple the authentication mechanism from the rest of the application. In

essence, the security information is "tunneled" through the third-party system. No modification of the infrastructure is necessary, which makes our approach feasible for a wide range of applications; the approach is also targeted towards applications built on large portions of legacy code, which can only be changed with considerable effort or cost.

Consider the following example scenarios:

- During an online auction of a precious item, a potential bidder wants to make sure that an image of an item displayed by the auction software adequately displays the item for sale.
- In a video conferencing tool that is used to facilitate online business, both parties may (for documentation purposes) require a proof of authenticity and integrity of the video stream.
- Digital images (or digitally recorded video streams) cannot readily be used as evidence in legal cases, as they can easily be forged or altered. In particular, portions of the images can be blurred or two images can be pasted together to remove suspicious parts.
- Some government agencies store documents (such as drivers licenses, birth certificates, etc.) only in digitized form. Typically, these documents are scanned and the resulting images are retained in an image database. Given the availability of sophisticated image processing software, it is obvious that the integrity of such electronic documents must be protected.

In all cases, both the integrity and authenticity of a multimedia object is of central importance for the security of the overall software system. Integrity and authenticity can be guaranteed by cryptographically signing the multimedia objects in question; in a naive solution, a media object is always stored together with its corresponding signature (Friedman, 1993). Tightly integrating a multimedia authentication mechanism into such applications can be a serious obstacle in practice, as this may require considerable changes to existing software architectures or the adoption of nonstandard application software. From a software-engineering point of view, modular media authentication mechanisms would be desirable that can be added as plug-ins or front-ends to existing software systems and third-party network infrastructures at ease.

In this paper, we give a construction for a modular—but yet provably secure—media authentication mechanism that can readily be plugged into existing software systems. Technically, we use *invertible fragile watermarks* to store cryptographic signatures directly in the media objects themselves. Using this system, the authentication mechanism can be *completely separated from the application software*, as all authentication information is readily encoded (in a format-compliant way) in the media file itself.

In contrast to classical digital watermarks which are designed to resist signal processing attacks, *fragile watermark* encode additional data imperceptibly in some media file *without providing any robustness*; that is, signal processing attacks will destroy the embedded watermark. For an overview of digital watermarking technology see (Katzenbeisser and Petitcolas, 2000). Fragile watermarks can be seen as a kind of alerter that reports whether a media object was tampered. In our approach, we use *invertible fragile watermarks*, which allow to insert a fragile watermark into an object as usual, but facilitate the lossless removal of the watermark from an untampered watermarked object. More precisely, if a watermark is successfully detected, the information contained in the recovered watermark, together with the watermark key, suffices to remove the watermark completely from the object.

Fragile watermarks are natural candidates for assuring the integrity of image files and were proposed by various authors (e.g., Schneider and Chang, 1996; Xie and Arce, 1998). In these approaches, image-dependant patterns are encoded as fragile watermarks in digital images. An image is considered authentic if and only if it is possible to correctly recover these patterns. If a file with such a watermark is modified, then either the watermark cannot be detected any more or the recovered patterns do not match the image. In both cases, the image is considered to be tampered. Unfortunately, this approach has the apparent drawback that it is not possible to formally prove its security in a cryptographically precise way, as properties of the watermark embedder or detector become security-critical.

In this paper we provide the first construction for a *provably secure watermark-based authentication scheme for digital media files*. In contrast to previous schemes, we use watermarks merely as a communication channel for transmitting a cryptographic signature; the scheme therefore draws its security entirely from this signature. In our approach, the signing algorithm produces an authenticated object $\overline{O}$ out of a media file O in such a way that the integrity and authenticity of $\overline{O}$ can readily be asserted using only a public key and $\overline{O}$. As $\overline{O}$ will be perceptually similar to O, $\overline{O}$ can readily be used as a replacement for O in existing application software, thereby adding a layer of security.

The rest of the paper is organized as follows. After reviewing the necessary watermarking technology in Section 2, we introduce our framework for media authentication in Section 3. Finally, we present two provably secure constructions for media authentication schemes in Sections 4 and 5; the second construction can be used for large media files or in streaming applications.

2. INVERTIBLE WATERMARKS

While almost all previous watermarking schemes introduced some small amount of irreversible distortion in the data during the embedding process,

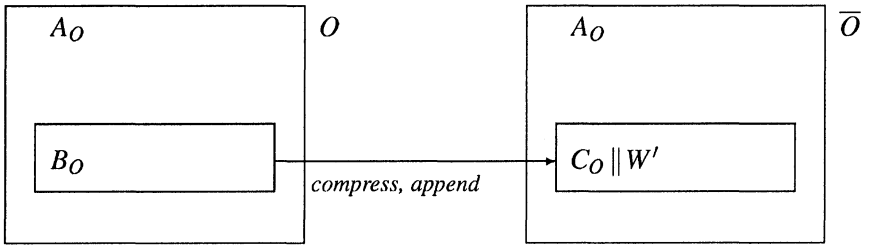

Figure 1. Invertible watermarking. An object O is divided into two parts A_O and B_O. The watermark consists of the compressed part B_O, denoted by C_O, and the watermark payload W'.

invertible watermarks (Honsinger et al., 1999; Fridrich et al., 2001; Fridrich et al., 2002a; Steinebach and Dittmann, 2003a; Maas et al., 2002; Dittmann and Benedens, 2003) can be removed completely from a watermarked object, thereby recovering the original.

Fridrich et al., 2001 introduced a general framework that allows to construct an invertible fragile watermarking scheme out of a fragile one. The general idea is to divide the object O, dependent on a public key K_W, into two (not necessarily consecutive) parts A_O and B_O. The latter part contains perceptually insignificant portions of the object that can be overwritten by a watermark without lowering the object quality, whereas A_O contains perceptually visible parts that must be preserved. To provide invertibility, the original part B_O is compressed and stored in the watermark; denote the compressed part B_O with C_O. The watermark W consists of the watermark payload W' and C_O, thus $W = C_O \| W'$. W replaces the part B_O in the watermarked object $\overline{O}$. This general framework is depicted in Figure 1.

The distortion of the watermark can easily be removed by separating the marked object $\overline{O}$ into the two parts $A_{\overline{O}}$ and $B_{\overline{O}}$. During the watermark insertion process, only $B_{\overline{O}}$ was modified, so $A_{\overline{O}} = A_O$. Now, $B_{\overline{O}}$ has the form $W = C_O \| W'$; decompressing C_O yields to the part B_O of the original object O. By overwriting $B_{\overline{O}}$ with B_O in the object $\overline{O}$, O can be completely recovered. This procedure works only if $\overline{O}$ was not altered; it is therefore a fragile watermarking scheme.

In the rest of the paper, we denote an invertible watermarking scheme as a tuple of two probabilistic polynomial algorithms $\langle \text{SEPARATE}, \text{JOIN} \rangle$. On input O and K_W, SEPARATE produces the tuple $\langle A_O, B_O \rangle$. JOIN inverts the algorithm SEPARATE, i.e., on input $\langle A_{\overline{O}}, B_{\overline{O}} \rangle$ and K_W it outputs $\overline{O}$. Except with negligible probability, we require that

$$\text{JOIN}(K_W, \text{SEPARATE}(K_W, O)) = O,$$

for all objects O and keys K_W with $\text{SEPARATE}(K_W, O) \neq \text{FAIL}$.

From the previous description it is obvious that it is not possible to embed an invertible watermark in every object. In case the part B_O cannot be sufficiently compressed, there is not enough room to store both the watermark payload and the compressed part C_O. However, typical multimedia files (such as images or audio files) contain enough redundant, compressible information so that the watermarking operation *works for virtually all relevant objects.*

This general construction principle can be instantiated for different types of media objects: for example (Fridrich et al., 2002b) gave two different constructions that operate on JPEG images; (Steinebach and Dittmann, 2003b) showed how audio files can be watermarked in an invertible manner. Other implementations can be found in (Fridrich et al., 2001; Fridrich et al., 2002a; Dittmann et al., 2002).

It is important to note that the construction of media authentication schemes given in this paper is general and *works for any invertible watermarking scheme. We will show that the security of the scheme only depends on the cryptographic primitives involved, while the watermark serves merely as insecure communication channel and thus does not affect the security of the authentication scheme*; a formal proof of this claim will be given in Sections 4 and 5.2. For this reason, we will treat an invertible watermarking scheme as black box in the remaining parts of the paper.

3. MEDIA AUTHENTICATION SCHEMES

Media authentication schemes based on invertible watermarks can be described by a tuple of four probabilistic polynomial time algorithms $\langle$GENKEY, PROTECT, VERIFY, RECONSTRUCT$\rangle$. GENKEY denotes the key-generation process; by using a private key, PROTECT authenticates an object O and outputs its signed version $\overline{O}$. Signed objects can be verified by the algorithm VERIFY and a public key; VERIFY either outputs TRUE or FALSE. In the first case, the object is deemed authentic; in the latter case, the object is considered modified. RECONSTRUCT reverses the protection mechanism and losslessly reconstructs O out of $\overline{O}$.

3.1 Definition

Formally, an *invertible media authentication scheme* is defined as follows:

- Algorithm GENKEY generates keys for the application. On input 1^n, GENKEY produces a triple of strings $\langle K_P, K_V, K_R\rangle$ with $|K_P \,\|\, K_V \,\|\, K_R| = n$; the operation $\|$ denotes string concatenation. The key K_P will be used in the protection step, whereas K_V and K_R are used for verification and recovery. The verification key K_V is a public key, whereas K_P and K_R are private keys.

- Algorithm PROTECT takes K_P, K_R and an object O. The output of the algorithm consists of an authenticated object $\overline{O}$.
- Algorithm VERIFY takes the verification key K_V and an object $\overline{O}$ and outputs a boolean variable.
- Algorithm RECONSTRUCT takes the keys K_R and K_V and an object $\overline{O}$ and restores the original object O.

Note that we have defined all algorithms as probabilistic, which implies that they can fail on certain instances (as noted above it may not be possible to embed a watermark in an invertible manner); in this case, the algorithms output a special symbol FAIL. We require that the media authentication scheme "works" for almost all objects that can be watermarked. In particular, $\text{VERIFY}(\text{PROTECT}(O,K_P,K_R),K_V) = \text{TRUE}$ and $\text{RECONSTRUCT}(\text{PROTECT}(O,K_P,K_R),K_R,K_V) = O$ must hold except for a negligible fraction of all objects O with $\text{PROTECT}(O,K_P,K_R) \neq \text{FAIL}$.

As usual, we will denote a cryptographic *signature scheme* as triple of probabilistic polynomial algorithms $\mathbb{S} = \langle \text{GENSIGN}, \text{SIGN}, \text{SIGVERIFY} \rangle$; GENSIGN denotes the key generation, SIGN the signing and SIGVERIFY the signature verification algorithm. A signature scheme is said to be secure, if it is secure against existential forgery of signatures under a chosen-message attack (Goldwasser et al., 1988); that is, if the attacker is unable—even with access to a signing oracle—to forge a valid pair of a message and a corresponding signature.

3.2 Attacker Model

Sticking to Kerckhoffs' principle, we assume that an attacker possesses complete knowledge of the system. The general goal of an attacker will be to *forge an authentic object* relative to a public key K_V, whose corresponding secret key K_P is unknown to him.

Similar to attacks against cryptographic signature schemes, we can distinguish several types of attacks against media authentication schemes according to the possibilities for an attacker to interfere with the system. It seems natural to assume that an attacker will know several protected media files under one verification key K_V, as such objects might be freely available on the Internet. A more powerful attacker may even launch a *chosen message attack.* In this setup, an attacker is able to obtain protected objects of his own choice. That is, he can obtain an authenticated version $\overline{O}$ of object O chosen during the attack. In imaging applications, such an attack is particularly realistic, as long as the attacker has physical access to the imaging device and can take pictures of his own choice.

For this reason, we adopt the notion of *existential forgery under chosen message attacks* for the present scenario. For a given public key K_V, the attacker

attempts to output an object O so that $\text{VERIFY}(O, K_V) = \text{TRUE}$. During his attack, he is free to use an oracle that delivers him arbitrary authenticated objects on request. We say that an attack is successful, if the attacker manages to output (with non-negligible probability) an object $\overline{O}$ together with an alleged original O such that $\text{VERIFY}(\overline{O}, K_V) = \text{TRUE}$, where the original object O was *not* presented to the oracle previously. It is generally agreed that this is the *most general attacker model* one can define for the signature scenario.

DEFINITION 1 *Let* $\langle \text{GENKEY}, \text{PROTECT}, \text{VERIFY}, \text{RECONSTRUCT} \rangle$ *be a media authentication scheme and* QUERY_{K_P} *be an oracle that computes* $\overline{O} \leftarrow \text{PROTECT}(O, K_P)$ *on input* O. *Furthermore, let* $\langle K_P, K_V, K_R \rangle \in [\text{GENKEY}(1^{n_K})]$. *An attack is a probabilistic algorithm* ATTACK *with oracle access to* QUERY_{K_P} *and success probability* $\varepsilon_{\text{ATTACK}}$ *such that*

$$\text{ATTACK}(1^n, K_V) = \begin{cases} \langle O, \overline{O} \rangle & \text{such that } \text{VERIFY}(\overline{O}, K_V) = \text{TRUE}, \\ & |O| = n,\ \overline{O} \in [\text{PROTECT}(O, K_P)] \\ & \text{and } O \neq O_i \text{ for all } 1 \leq i \leq l, \\ & \text{with probability } \varepsilon_{\text{ATTACK}} \\ \text{FAIL} & \text{with probability } 1 - \varepsilon_{\text{ATTACK}}, \end{cases}$$

where O_i *denotes the input to the i-th oracle query* QUERY_{K_P}. *The probability is taken over all coin tosses of* ATTACK *and all keys* $\langle K_P, K_V, K_R \rangle$.

We say that a media authentication scheme is secure, if the success probability of *every* probabilistic polynomial time attack is negligible:

DEFINITION 2 *A media authentication scheme is secure against existential forgery of authenticated objects, if every probabilistic polynomial time attack* ATTACK *has negligible success probability.*

4. OFFLINE AUTHENTICATION

In this section, we describe a media authentication scheme that needs access to the whole media file at once during the protection algorithm (such schemes will be called offline media authentication schemes). This construction is therefore primarily suitable for image files or short video sequences. A separate construction for streaming media will be given in Section 5.

Let $\mathbb{S} = \langle \text{GENSIGN}, \text{SIGN}, \text{SIGVERIFY} \rangle$ be a cryptographic signature scheme producing signatures of length k, ENCRYPT and DECRYPT be the encryption and decryption function of a symmetric cipher and COMPRESS be the compression algorithm of a lossless compression scheme. Furthermore, we fix an invertible watermarking scheme $\langle \text{SEPARATE}, \text{JOIN} \rangle$ that can embed watermark strings of length k.

Loosely speaking, the media authentication scheme stores a cryptographic signature of the unmodified portion of the object (the part A_O) and the encrypted, compressed part B_O as an invertible watermark. During the verification process, we check whether we can extract a valid cryptographic signature out of the watermark. The construction is as follows:

- GENKEY runs GENSIGN to obtain a key pair $\langle K_{SS}, K_{VS} \rangle$; furthermore, it computes a key K_E for the symmetric cipher and a random string K_W. Let $K_P = K_{SS} \,\|\, K_W$, $K_V = K_{VS} \,\|\, K_W$ and $K_R = K_E \,\|\, K_W$.
- PROTECT, on input O, $K_P = K_{SS} \,\|\, K_W$ and $K_R = K_E \,\|\, K_W$, separates O, using algorithm SEPARATE and key K_W, into two parts A_O and B_O. The latter part is compressed to obtain C_O. Denote with W' the string $W' = X \,\|\, s$, where

  $$X \leftarrow \text{ENCRYPT}(K_E,\, C_O \,\|\, H(O)),$$

  H is a hash function and

  $$s \leftarrow \text{SIGN}(K_{SS},\, A_O \,\|\, X).$$

  Note that C_O is stored encrypted so that the reconstruction of the original is only possible by knowing the key K_E. PROTECT runs JOIN on K_W and $\langle A_O, W' \rangle$ to obtain the authenticated object $\overline{O}$ or FAIL. If JOIN fails, PROTECT outputs FAIL, otherwise $\overline{O}$.
- VERIFY, on input $\overline{O}$ and $K_V = K_{VS} \,\|\, K_W$, runs SEPARATE on K_W and $\overline{O}$ to obtain the two parts $A_{\overline{O}}$ and $B_{\overline{O}}$ of $\overline{O}$. The latter part has the form $B_{\overline{O}} = X \,\|\, s$, where X is an arbitrary string and s is a cryptographic signature. VERIFY outputs the Boolean value $\text{SIGVERIFY}(K_{VS},\, A_{\overline{O}} \,\|\, X,\, s)$.
- RECONSTRUCT, on input $\overline{O}$, $K_R = K_E \,\|\, K_W$ and $K_V = K_{VS} \,\|\, K_W$, first runs VERIFY to assure the integrity of $\overline{O}$; in case VERIFY outputs FALSE, RECONSTRUCT exits with FAIL. Otherwise, it separates $\overline{O}$ (using SEPARATE and key K_W) into the two parts $A_{\overline{O}}$ and $B_{\overline{O}}$. The latter part has the form $B_{\overline{O}} = X \,\|\, s$. By using K_E, RECONSTRUCT decrypts X to obtain $C_O \,\|\, h$, where h denotes a hash; the part C_O is decompressed to obtain B_O. Finally, the part $B_{\overline{O}}$ of $\overline{O}$ is overwritten with B_O to obtain an object O. If $H(O) = h$, RECONSTRUCT outputs O, otherwise FAIL.

Intuitively, the scheme is secure because of the following argument: in case an attacker modified the part $A_{\overline{O}}$ of $\overline{O}$, the embedded cryptographic signature s is matched against a modified media file. On the other hand, if any bit in $B_{\overline{O}}$ is modified, then at least one bit of the embedded fragile watermark (containing either the signature s or the compressed part B_O) is destroyed. In all cases, the tampering will be detected during the signature verification. Formally, we can state this result as a theorem:

THEOREM 3 *If $\mathbb{S}$ is a cryptographic signature scheme secure against existential forgery of messages under a chosen message attack, then the above scheme is a secure media authentication scheme.*

Proof. Suppose, for the sake of contradiction, that there exists an attack ATTACK (with access to the media authentication oracle QUERY_{K_P}) against the scheme, which succeeds with non-negligible probability. We show that in this case there exists also an attack FORGE (with access to a signing oracle $\text{SIGNQUERY}_{K_{SS}}$) against $\mathbb{S}$, which contradicts the assumption.

We construct the signature forging algorithm FORGE (for the public signature key K_{VS}) in the following manner. On input K_{VS}, FORGE chooses random keys K_E and K_W and simulates ATTACK. Whenever ATTACK makes an oracle query $\text{QUERY}_{K_P}(O_i)$, this query is replaced by the following probabilistic algorithm, which utilizes the signing oracle $\text{SIGNQUERY}_{K_{SS}}$; here, K_{SS} denotes the corresponding secret signature key:

$\langle A_{O_i}, B_{O_i}\rangle \leftarrow \text{SEPARATE}(K_W, O_i)$
compress B_{O_i} to obtain C_{O_i}
$X_i \leftarrow \text{ENCRYPT}(K_E,\ C_{O_i} \,\|\, H(O_i))$
query $\text{SIGNQUERY}_{K_{SS}}(A_{O_i} \,\|\, X_i)$ for signature s
$W_i' = X_i \,\|\, s$
output $\text{JOIN}(K_W, \langle A_{O_i}, W'\rangle)$

Note that JOIN either outputs FAIL or the object $\overline{O}_i$.

When the simulation of ATTACK is finished, ATTACK either outputs FAIL or obtains a tuple $\langle O, \overline{O}\rangle$. In the first case, FORGE exits with FAIL. Otherwise, FORGE runs SEPARATE on $\overline{O}$ and K_W, resulting in the tuple $\langle A_{\overline{O}}, B_{\overline{O}}\rangle$; $B_{\overline{O}}$ has the form $B_{\overline{O}} = X \,\|\, s$. Finally, FORGE outputs the pair $\langle A_{\overline{O}} \,\|\, X, s\rangle$. It is easy to see that FORGE perfectly simulates ATTACK so that a valid pair of a message and a signature is produced if and only if ATTACK succeeded.

It remains to show that the message $A_{\overline{O}} \,\|\, X$ was not presented to the signature oracle previously. For this, assume the contrary, i.e., that there exists an index i such that $A_{\overline{O}} \,\|\, X = A_{O_i} \,\|\, X_i$. This can only be the case if $A_O = A_{\overline{O}} = A_{O_i}$ and $X = X_i$, i.e., $\text{ENCRYPT}(K_E,\ C_O \,\|\, H(O)) = \text{ENCRYPT}(K_E,\ C_{O_i} \,\|\, H(O_i))$. This requires that both O and O_i agree on part A; furthermore, by ENCRYPT being uniquely decipherable, we have $C_O \,\|\, H(O) = C_{O_i} \,\|\, H(O_i)$. This can only be the case if both O and O_i agree on part C and thus also on part B. We conclude that $O = O_i$, but this contradicts the definition of a successful attack against the media authentication scheme. This completes the proof. □

5. ONLINE AUTHENTICATION

The authentication method of the previous section assumes that the full media O is present when the media file is authenticated. However, for many multimedia applications such a solution is unacceptable, e.g., in audio or video

streaming. In this section we present an online authentication scheme that operates only on fixed-length chunks of media at a time, but nevertheless allows the full media object to be authenticated. For this purpose, an object O is considered to consist of n chunks of equal length $O_1, \ldots, O_n$; in abuse of notation, we write $O = O_1 \| \cdots \| O_n$.

The online media authentication scheme presented in this paper is targeted towards applications where it must be possible to produce authenticated *excerpts*, i.e., small consecutive portions of the media stream. For example, consider the evidence produced by eavesdropping a telephone, which might be automatically authenticated by a recording device; in a court hearing only a small and relevant part of the overall evidence is presented to the public. In order to prevent tampering, this excerpt should be produced *without* access to the secrets of the eavesdropping system (i.e., the protection key K_P). Nevertheless the integrity and authenticity of the excerpt should be publicly verifiable.

Given an object O, we call an object O' an *excerpt* of O, if O' may be obtained from O by removing some chunks from the beginning and the end of O. Formally, $O' = O'_1 \| \cdots \| O'_m$ is an excerpt of $O = O_1 \| \cdots \| O_n$, written as $O' \preceq O$, if $m \leq n$ and there exists an index $1 \leq i \leq n - m$ so that $O'_1 = O_i, \ldots, O'_m = O_{i+m}$.

Given an original object O, it is possible with the proposed system to generate a signed object $\overline{O}$ such that *each* excerpt of the signed object $\overline{O}' \preceq \overline{O}$ can be checked for its integrity and authenticity without further preprocessing. More precisely, the verification algorithm described below will detect any modifications in an excerpt of $\overline{O}$ and will report the presence of non-consecutive chunks.

Formally, the attacker model we use for online authentication schemes is similar to the one presented in Section 3.2, with the sole exception that the production of excerpts is not considered an attack. Again, an attacker is forced to perform a selective forgery under a chosen message attack. However, the media object obtained at the end of the attack must not be an excerpt of an object submitted to the signing oracle previously.

DEFINITION 4 *Let* $\langle \text{GENKEY}, \text{PROTECT}, \text{VERIFY}, \text{RECONSTRUCT} \rangle$ *be an online authentication scheme and* QUERY_{K_P} *be an oracle that, on input* O*, computes* $\overline{O} \leftarrow \text{PROTECT}'(O, K_P)$*. Furthermore, let* $\langle K_P, K_V, K_R \rangle \in [\text{GENKEY}'(1^{n\kappa})]$*. An attack is a probabilistic algorithm* SATTACK *with oracle access to* QUERY_{K_P} *and success probability* $\varepsilon_{\text{SATTACK}}$ *such that*

$$\text{SATTACK}(1^n, K_V) = \begin{cases} \langle O, \overline{O} \rangle & \text{such that } \text{VERIFY}'(\overline{O}, K_V) = \text{TRUE}, \\ & |O| = n,\ \overline{O} \in [\text{PROTECT}'(O, K_P)] \\ & \text{and } O \not\preceq O^{(i)} \text{ for all } 1 \leq i \leq l, \\ & \text{with probability } \varepsilon_{\text{SATTACK}} \\ \text{FAIL} & \text{with probability } 1 - \varepsilon_{\text{SATTACK}}, \end{cases}$$

where $O^{(i)}$ denotes the input to the i-th oracle query QUERY_{K_P}. *The probability is taken over all coin tosses and all keys* $\langle K_P, K_V, K_R \rangle$.

Again, we say that an online media authentication scheme is secure, if *every* probabilistic polynomial attack has only negligible success probability.

5.1 Construction

In this section, we provide the construction of an online media authentication scheme that operates block by block on the media content. Essentially, we apply the authentication scheme described in the previous section on each chunk O_i, with the exception that the there is some linkage between the chunks, computed by a hash function. Technically, we rely on the concept of hash chains introduced by (Gennaro and Rohatgi, 1997).

Fix any collection of hash functions

$$\mathbb{H} = \left\langle H_h : \{0,1\}^* \to \{0,1\}^{\ell(|h|)} \mid h \in \{0,1\}^* \right\rangle$$

for any super-logarithmically growing function $\ell : \mathbb{N} \mapsto \mathbb{N}$. Denote with k_h an index to $\mathbb{H}$; furthermore, let k be the length of the cryptographic signatures. We assume that both k_h and k are polynomial in the security parameter. For the construction we use an invertible watermarking scheme that is capable of storing $k + \ell(k_h)$ bits. The construction is as follows:

- GENKEY runs GENSIGN to obtain a tuple of keys $\langle K_{SS}, K_{VS} \rangle$; furthermore it computes a key K_E for a symmetric cipher and a random string K_W. GENKEY′ outputs the keys $K_P = K_{SS} \| K_W$, $K_V = K_{VS} \| K_W$ and $K_R = K_E \| K_W$.
- PROTECT, on input $O = O_1 \| \cdots \| O_n$, K_P and K_R, performs the following steps:

  $h_0 \leftarrow \text{RANDOM}(\ell(k_h))$
  for $i = 1, \ldots, n$ **do**
  $\langle A_{O_i}, B_{O_i} \rangle \leftarrow \text{SEPARATE}(K_W, O_i)$
  compress B_{O_i} to obtain C_{O_i}
  $X_i \leftarrow \text{ENCRYPT}(K_E, C_{O_i} \| H_h(O_i))$
  $s_i \leftarrow \text{SIGN}(K_{SS}, A_{O_i} \| X_i \| h_{i-1})$
  $h_i \leftarrow H(A_{O_i} \| X_i \| h_{i-1})$
  let $W_i = X_i \| h_{i-1} \| s_i$
  $\overline{O}_i \leftarrow \text{JOIN}(K_W, \langle A_{O_i}, W_i \rangle)$
  if $\overline{O}_i = \text{FAIL}$, **exit with** FAIL
  end for
  output $\overline{O} = \overline{O}_1 \| \cdots \| \overline{O}_n$

- VERIFY, on input $\overline{O} = \overline{O}_1 \| \cdots \| \overline{O}_n$ and K_V, performs the following steps:

for $i = 1, \ldots, n$ **do**
  $\left\langle A_{\overline{O}_i}, B_{\overline{O}_i} \right\rangle \leftarrow$ SEPARATE$(K_W, \overline{O}_i)$
  $B_{\overline{O}_i}$ has the form $X_i \,\|\, h_{i-1} \,\|\, s_i$
  if $i > 1$ and $h_{i-1} \neq \tilde{h}$ **exit with** FAIL
  let $\tilde{h} = H_h(A_{\overline{O}_i} \,\|\, X_i \,\|\, h_{i-1})$
  $b_i \leftarrow$ SIGVERIFY$(K_{VS}, A_{\overline{O}_i} \,\|\, X_i \,\|\, h_{i-1}, s_i)$
  if $b_i =$ FALSE, **exit with** FALSE
end for
exit with TRUE

- RECONSTRUCT applies the reconstruction algorithm of Section 4 on the chunks of $\overline{O}$.

5.2 Security Against Forgeries

In a similar way as in Theorem 3, the security of the above scheme can be established:

THEOREM 5 *If $\mathbb{S}$ is a cryptographic signature scheme secure against existential forgery of messages under a chosen message attack and if $\mathbb{H}$ is a collection of preimage- and collision-resistant hash functions, then the above scheme is a secure online media authentication scheme.*

Proof. Suppose, for the sake of contradiction, that there exists an attack SATTACK against the above scheme, which succeeds with a non-negligible probability. We show that in this case there exists also an attack FORGE against $\mathbb{S}$, which contradicts the assumption.

We construct the signature forging algorithm FORGE (for the public signature verification key K_{VS}) in the following manner. On input K_{VS}, FORGE first chooses random keys K_E and K_W. Finally, FORGE invokes SATTACK. In the rest of the proof, denote with $O^{(i)}$ the input to the i-th query to the oracle QUERY$_{K_P}$, whereas $O^{(i)}_j$ denotes the j-th chunk of $O^{(i)}$; the number of chunks in $O^{(i)}$ is given by n_i.

Whenever SATTACK makes an oracle query QUERY$_{K_P}(O^{(i)})$ in order to obtain a signed stream $\overline{O}^{(i)}$, given $O^{(i)} = O^{(i)}_1 \,\|\, \ldots \,\|\, O^{(i)}_{n_i}$, this query is simulated by the following probabilistic computation that uses a signature oracle SIGNQUERY$_{K_{SS}}$ (essentially, this code is equivalent to that of PROTECT):

$s_{i,0} \leftarrow$ RANDOM$(\ell(h_k))$
for $j = 1, \ldots, n_i$ **do**
  $\left\langle A_{O^{(i)}_j}, B_{O^{(i)}_j} \right\rangle \leftarrow$ SEPARATE$(K_W, O^{(i)}_j)$
  compress $B_{O^{(i)}_j}$ to obtain $C_{O^{(i)}_j}$
  $X^{(i)}_j \leftarrow$ ENCRYPT$(K_E, C_{O^{(i)}_j} \,\|\, H_h(O^{(i)}_j))$

$s_j^{(i)} \leftarrow \text{SIGNQUERY}_{K_{SS}}(A_{O_j^{(i)}} \| X_j^{(i)} \| h_{j-1}^{(i)})$
$h_j^{(i)} \leftarrow H_h(A_{O_j^{(i)}} \| X_j^{(i)} \| h_{j-1}^{(i)})$
let $W_j^{(i)} = X_j^{(i)} \| h_{j-1}^{(i)} \| s_j^{(i)}$
$\overline{O}_j^{(i)} \leftarrow \text{JOIN}\left(K_W, \left\langle A_{O_j^{(i)}}, W_j^{(i)} \right\rangle\right)$
if $\overline{O}_j^{(i)} = \text{FAIL}$, **exit with** FAIL
end for
output $\overline{O}^{(i)} = \overline{O}_1^{(i)} \| \cdots \| \overline{O}_{n_i}^{(i)}$

Up to here, ATTACK perfectly simulates SATTACK. When the simulation of SATTACK is finished it obtains (with non-negligible probability) a tuple $\langle O, \overline{O} \rangle$, where $\overline{O}$ is a signed media stream with n chunks and $O \not\preceq O^{(i)}$ for all $1 \leq i \leq l$. If SATTACK fails, ATTACK fails as well.

Denote with

$$\mathbf{Q} = \{A_{O_j^{(i)}} \| X_j^{(i)} \| h_{j-1}^{(i)} \parallel 1 \leq i \leq l, 1 \leq j \leq n_i\}$$

the set of oracle queries. For all $1 \leq k \leq n$, ATTACK runs SEPARATE on $\overline{O}_k$ and K_W to obtain $A_{\overline{O}_k} = A_{O_k}$ and $B_{\overline{O}_k}$; the latter string has the form $B_{\overline{O}_k} = X_k \| h_{k-1} \| s_k$. Consider two cases:

- Case 1: there exists an index $1 \leq k \leq n$ such that $A_{O_k} \| X_k \| h_{k-1} \notin \mathbf{Q}$. Then, ATTACK outputs the tuple

  $$\langle A_{\overline{O}_k} \| X_k \| h_{k-1},\ s_k \rangle$$

  as signature forgery. By assumption, this tuple is a valid forgery.
- Case 2: for all indices $1 \leq k \leq n$ we have $A_{\overline{O}_k} \| X_k \| h_{k-1} \in \mathbf{Q}$. In this case, ATTACK fails. We argue later that this case can happen only with negligible probability.

ATTACK can distinguish the two cases in polynomial time; furthermore, the success probability of ATTACK equals the success probability of SATTACK, up to a negligible quantity (resulting out of case 2). This contradicts the assumption.

It remains to show that case 2 happens only with negligible probability. Note that, by assumption, O (and thus also $\overline{O}$) contains at least two chunks, as otherwise trivially $O \preceq O^{(i)}$ for some index $1 \leq i \leq l$. Consider the last chunk $\overline{O}_n$; its decomposition according to SEPARATE is given by $\langle A_{\overline{O}_n}, X_n \| h_{n-1} \| s_n \rangle$. By assumption, there exist indices $1 \leq i \leq l$ and $1 \leq j \leq n_i$ such that

$$A_{O_j^{(i)}} \| X_j^{(i)} \| h_{j-1}^{(i)} = A_{O_n} \| X_n \| h_{n-1}.$$

In particular, also $h_{j-1}^{(i)} = h_{n-1}$. Distinguish two cases:

- Case (a): We have $j = 1$. Now, as both $\overline{O}$ and $\overline{O}^{(i)}$ are valid,

$$h_{n-1} = H_h(A_{O_{n-1}} \| X_{n-1} \| h_{n-2}).$$

  By assumption, $h_{n-1} = h_0^{(i)}$, showing that $A_{O_{n-1}} \| X_{n-1} \| h_{n-2}$ is a pre-image of the random string $h_0^{(i)}$.

- Case (b): We have $j > 1$. Again, as both $\overline{O}$ and $\overline{O}^{(i)}$ are valid, $h_{n-1} = H_h(A_{O_{n-1}} \| X_{n-1} \| h_{n-2})$ and $h_{j-1}^{(i)} = H_h(A_{O_{j-1}^{(i)}} \| X_{j-1}^{(i)} \| h_{j-2}^{(i)})$. By assumption, $h_{n-1} = h_{j-1}^{(i)}$. If

$$A_{O_{n-1}} \| X_{n-1} \| h_{n-2} \neq A_{O_{j-1}^{(i)}} \| X_{j-1}^{(i)} \| h_{j-2}^{(i)},$$

  we have found a collision of H_h. Otherwise, $A_{O_{n-1}} = A_{O_{j-1}^{(i)}}$, $h_{n-2} = h_{j-2}^{(i)}$ and $X_{n-1} = X_{j-1}^{(i)}$. The latter equation implies

$$\underbrace{\text{ENCRYPT}(K_E, C_{O_{n-1}} \| H(O_{n-1}))}_{X_{n-1}} = \underbrace{\text{ENCRYPT}(K_E, C_{O_{j-1}^{(i)}} \| H(O_{j-1}^{(i)}))}_{X_{j-1}^{(i)}}.$$

  Since ENCRYPT is uniquely decipherable, $C_{O_{n-1}} = C_{O_{j-1}^{(i)}}$, implying that $B_{O_{n-1}} = B_{O_{j-1}^{(i)}}$. This shows that now O and $O^{(i)}$ also agree on their second-last chunk. By assumption, O must therefore have at least one more chunk (as otherwise trivially $O \preceq O^{(i)}$). Applying this argument inductively, we either find a collision or have $n > j$. In the latter case, as in case (a), $A_{O_{n-j-1}} \| X_{n-j-1} \| h_{n-j-2}$ is a pre-image of $h_0^{(i)}$.

In summary, if case 2 happens, then we can either find a pre-image of a random string with respect to H_h or a collision of H_h (a formal proof of this claim uses again a reducibility argument). By the assumptions on $\mathbb{H}$, this can happen only with negligible probability. This completes the proof. □

6. CONCLUSIONS

In this paper, we provided two constructions which solve the data integrity problem for multimedia applications by combining methods from cryptography and watermarking. Technically, we used digital watermarks to encode cryptographic signatures directly in multimedia files. One construction is suitable for images and short media files, whereas the other one is targeted towards streaming applications. Both schemes were shown to be secure under standard

cryptographic assumptions and can easily be incorporated into existing software systems or used as front-end programs to applications whose code is not under control of the user.

Acknowledgement. The work described in this paper has been supported in part by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT. The information in this document reflects only the author's views, is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

References

Dittmann, J. and Benedens, O. (2003). Invertible authentication for 3d-meshes. In *Proceedings of the SPIE vol. 5020, Security and Watermarking of Multimedia Contents V*, pages 653–664.

Dittmann, J., Steinebach, M., and Ferri, L. (2002). Watermarking protocols for authentication and ownership protection based on timestamps and holograms. In *Proceedings of the SPIE vol. 4675, Security and Watermarking of Multimedia Contents IV*, pages 240–251.

Fridrich, J., Goljan, M., and Du, R. (2001). Invertible authentication. In *Proceedings of the SPIE vol. 3971, Security and Watermarking of Multimedia Contents III*, pages 197–208.

Fridrich, J., Goljan, M., and Du, R. (2002a). Lossless data embedding—new paradigm in digital watermarking. *EURASIP Journal on Applied Signal Processing*, (2):185–196.

Fridrich, J., Goljan, M., and Du, R. (2002b). Lossless data embedding for all image formats. In *Proceedings of the SPIE vol. 4675, Security and Watermarking of Multimedia Contents IV*, pages 572–583.

Friedman, G. L. (1993). The trustworthy digital camera. *IEEE Transactions on Consumer Electronics*, 39(4):905–910.

Gennaro, R. and Rohatgi, P. (1997). How to sign digital streams. In *Advances in Cryptology (CRYPTO'97)*, volume 1294 of *Lecture Notes in Computer Science*, pages 180–197. Springer.

Goldwasser, S., Micali, S., and Rivest, R. (1988). A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–302.

Honsinger, C. W., Jones, P., Rabbani, M., and Stoffel, J. C. (1999). Lossless recovery of an original image containing embedded data. US patent application, Docket No: 77102/E/D.

Katzenbeisser, S. and Petitcolas, F. A. P., editors (2000). *Information Hiding Techniques for Steganography and Digital Watermarking*. Artech House.

Maas, D., Kalker, T., and Willems, F. M. (2002). A code construction for recursive reversible data-hiding. In *Proceedings of the ACM Workshop on Multimedia*, pages 15–18.

Schneider, M. and Chang, S.-F. (1996). A robust content based digital signature for image authentication. In *IEEE International Conference on Image Processing, Proceedings*, Lausanne.

Steinebach, M. and Dittmann, J. (2003a). Watermarking-based digital audio data authentication. *EURASIP Journal on Applied Signal Processing*, (10):1001–1015.

Steinebach, M. and Dittmann, J. (2003b). Watermarking-based digital audio data authentication. *EURASIP Journal on Applied signal processing*, 10:1001–1015.

Xie, L. and Arce, G. R. (1998). A blind wavelet based digital signature for image authentication. In *European Signal Processing Conference, Proceedings*, Rhodes, Greece.