Watermarking Schemes Provably Secure Against Copy and Ambiguity Attacks

André Adelsbach
Department of Computer Science
Universität des Saarlandes, Saarbrücken
adelsbach@cs.uni-sb.de

Stefan Katzenbeisser
Institute for Information Systems
Vienna University of Technology
skatzenbeisser@acm.org

Helmut Veith
Institute for Information Systems
Vienna University of Technology
veith@dbai.tuwien.ac.at

ABSTRACT

Protocol attacks against watermarking schemes pose a threat to modern digital rights management systems; for example, a successful attack may allow to copy a watermark between two digital objects or to forge a valid watermark. Such attacks enable a traitor to hinder a dispute resolving process or accuse an innocent party of a copyright infringement. Secure DRM systems based on watermarks must therefore prevent such protocol attacks. In this paper we introduce a formal framework that enables us to assert rigorously the security of watermarks against protocol attacks. Furthermore, we show how watermarking schemes can be secured against some protocol attacks by using a cryptographic signature of a trusted third party.

Categories and Subject Descriptors

E.3 [**Data Encryption**]: Public key cryptosystems; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection

General Terms

Security, Theory

Keywords

watermarking, protocol attacks, multimedia security

1. INTRODUCTION

Robust watermarking schemes were proposed as primitives in modern digital rights management systems or copyright protection protocols. Such schemes allow to insert information (e.g., the name of the copyright holder) directly as a watermark within the multimedia object in such a way that it is infeasible to remove the watermark without destroying the copyrighted content (robustness property).

In this paper, we mainly focus on attacks against dispute resolving schemes; typically, in such schemes two or more parties dispute over the copyright of *one* given object. A dispute resolving process should reveal the true author of the disputed work by checking the presence of the disputant's watermarks in the object.

Clearly, the intention of the copyright holder can be subverted if it is possible to remove a watermark from a multimedia object without rendering the object useless. Thus, modern watermarking schemes are designed to provide a considerable level of robustness. In such schemes, common intentional or unintentional modifications of the multimedia object do not destroy a contained watermark. However, this is not sufficient to provide a "secure" watermarking scheme. It was noted early during the development of watermarking algorithms that the intention of resolving the copyright situation might be subverted entirely without removing any watermark contained in multimedia objects. Indeed, the idea of *protocol attacks* is to enforce an unresolvable ambiguity during the copyright resolution process.

Although considerable progress in constructing robust watermarking systems has been made over the past few years, more effort is needed to secure watermarking schemes against protocol attacks, in particular ambiguity and inversion [6] as well as copy attacks [13]; see Section 2.4.

Several authors proposed watermarking systems that were believed to be unsusceptible to protocol attacks. For example, Craver et al. [6] proposed two systems that use the hash of the original, unmarked object during the watermark insertion process; however, both systems have been broken by attacks that are more efficient than a naive brute-force attack [16]. Qiao and Nahrstedt [15, 14] described audio and video watermarking systems claimed to be non-invertible, but their proof is flawed. A different approach to guard against copy attacks, which is based on using a robust watermarking scheme together with a fragile one, was proposed in [7]. Finally, [5, pp. 295-296] note that copy attacks might be prevented by embedding a digital signature; however, they advocate to sign only perceptually significant parts of the multimedia objects, which opens the possibility of attacks.

In this paper, we show for the first time how to build a formal framework that allows to assess the security of watermarking schemes. In the literature, the security of watermarks against protocol attacks was only analyzed with adhoc methods without a well-defined formal framework. In most cases, the security claims essentially amount to heuris-

DRM'03, October 27, 2003, Washington, DC, USA.
Copyright 2003 ACM 1-58113-786-9/03/0010 ...$5.00.

tics not justified by a rigorous analysis. The situation is totally different in cryptography, where the security of most cryptographic primitives can be established formally, assuming the intractability of certain number-theoretic problems.

The aim of our work is to combine the functionality of watermarking schemes with the advantages of classical cryptography. To this end, we suggest to imprint cryptographic signatures upon watermarks. We investigate three different cryptographic imprint patterns $\mathbb{P}_A$, $\mathbb{P}_B$ and $\mathbb{P}_C$ containing signatures of a trusted party. Our main theorem shows that they are secure against certain protocol attacks (where the attacker does not have access to the trusted party) unless common cryptographic assumptions are violated (see Theorem 3). Some of the basic ideas of this paper have been presented informally in [11].

Section 2 introduces watermarking schemes and protocol attacks, Section 3 reviews cryptographic signatures and Section 4 shows how to construct watermarking schemes provably secure against protocol attacks, assuming the security of cryptographic signatures. Finally, Section 5 discusses future research directions.

2. WATERMARKING SCHEMES AND PROTOCOL ATTACKS

2.1 Notation

A probabilistic polynomial-time algorithm for a problem S is an algorithm A that has access to a source of random bits. The algorithm A either computes the correct result of an instance of S or outputs the special symbol FAIL, indicating that the algorithm was unable to compute a solution. We say that A succeeds with probability ε, if it computes the correct result with probability ε; the probability is taken over all coin tosses of A and all problem instances in S.

The operator $\|$ stands for concatenation of two strings and $|\cdot|$ denotes the length of an object in bits. The notation $\{0,1\}^*$ denotes the set of all finite strings composed of 0 and 1 (including the empty string), whereas $\{0,1\}^n$ stands for the set of bit-strings of length n. Typically, O will denote an object without a watermark, whereas O' will refer to some object containing a watermark (or being believed to contain a mark).

2.2 Watermarking schemes

Formally, we model a watermarking scheme as a triple $\mathbb{W} = \langle G, E, D \rangle$ of a probabilistic polynomial-time algorithms G, E and D. For convenience, we assume that G, E and D never fail.

- Algorithm G models the key generation process: on input 1^{n_w} (a string consisting of n_w consecutive ones), G outputs a watermarking key $K \in \{0,1\}^{n_w}$ of length n_w. Here, n_w denotes the security parameter of the watermarking scheme. Since G is probabilistic, it will output a possibly different key on each invocation.

- Algorithm E models the watermark embedding process; on input of a digital object O, a watermark $W \in \{0,1\}^n$ and a key K, it outputs a watermarked object O'. We will assume that O and O' remain "perceptually similar"[1].

- Algorithm D denotes the watermark detection process. D decides whether a given watermark W is present in an object O' under key K with respect to the original, unmarked object O. In addition, D may also use an auxiliary input Aux that does not depend on the object O (e.g., a cryptographic key). Algorithm D either outputs TRUE or FALSE: $D(O', O, W, K, Aux) \in \{\text{TRUE}, \text{FALSE}\}$. The output TRUE indicates the presence of W in O'. We require (with overwhelming probability) that $D(E(O, W, K), O, W, K, Aux) = \text{TRUE}$ for all objects O, watermarks W and keys K.

We will only consider non-blind watermarking systems in this paper, i.e., systems in which the unmarked original is needed in the detection phase. Let $\mathcal{C}$ be the set of all multimedia objects to be watermarked; without loss of generality $\mathcal{C} = \{0,1\}^n, n > 0$ and n polynomial in n_w. We assume that there exists a probabilistic polynomial-time algorithm SAMPLE that outputs, on input k, uniformly and randomly objects $O \in \mathcal{C}$ with $|O| = k$. For convenience we furthermore assume that SAMPLE never fails; let $t_S(k)$ be the runtime of SAMPLE.

2.3 Dispute resolving

An authorship dispute between two disputants A and B about a disputed work O' is a scenario where A and B claim to be the exclusive rightful authors of O'. Informally, the goal of a dispute resolving scheme is to allow a third party, the *dispute resolver* D, to resolve authorship disputes in a "fair" way, by comparing the "ownership proofs" presented by the disputants. These proofs typically contain information that allows to check the presence of a specific watermark in O'.

Most dispute resolving protocols operate in the following manner. D first checks the presence of the watermarks presented by A and B in O'. If only one watermark is detectable, the dispute is resolved in favor of the disputant whose watermark is detectable. If both marks are present in O', D tries to establish whether B derived his original object from A's object or vice versa. One way to do this is to look at the objects both A and B claim to the the true unmarked originals. If B engineered his "original" object from O', A's watermark should be present in the "original" of B. Furthermore, the mark should not appear in A's "original". If this is the case, the dispute is resolved in favor of A; otherwise, the symmetric condition is checked (i.e. whether A's original contains B's mark) to verify B's authorship claims.

Unfortunately, this approach does not yield to a secure dispute resolving process. Craver et al. [6] showed that such dispute resolving can be defeated by protocol attacks (especially the inversion attack) as described below.

2.4 Protocol attacks

Protocol attacks against watermarking schemes in dispute resolving applications are aimed at introducing some sort of ambiguity in the copyright resolution processes based on the watermarking algorithm. Instead of attacking the robustness of the watermark itself, a protocol attack utilizes fake watermarks that are either inherently present in a multimedia object or added by the attacker. Well known protocol

[1] Note that perceptual similarity depends on the object type (image, video, audio, etc.) and the application environment in which marked objects are being used (see [10] for a discussion of the importance of the latter).

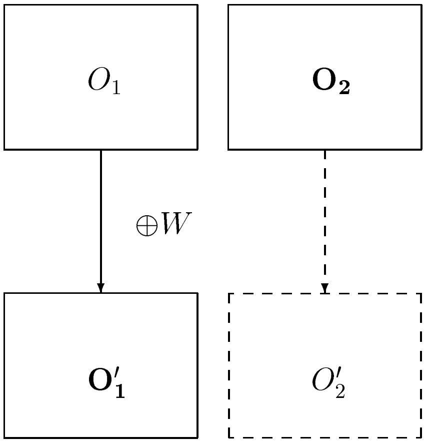

Figure 1: In a copy attack, an unknown watermark W is copied from a marked object O'_1 onto another object O_2.

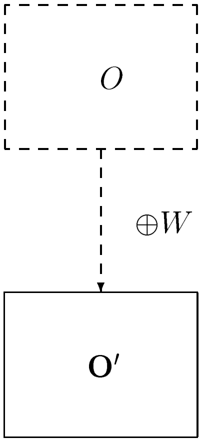

Figure 2: In an ambiguity attack, an attacker computes a watermark W (together with an alleged "original" object O and a key K) that is detectable in a given object O'.

attacks include inversion as well as copy and ambiguity attacks [6, 13]:

- A **copy attack** attempts to copy a watermark from a marked object O'_1 onto a different object O_2 *without* knowledge of the watermark W or the key K that was used to insert W in the object O_1. The attack produces an object O'_2 in which W is also detectable under key K. Figure 1 illustrates a copy attack (objects printed in bold are the input to the attack; objects computed by the attacker are enclosed in dashed boxes). A concrete example can be found in [13].

- An **ambiguity attack** amounts to computing a watermark that was never inserted in an object O', but still can be detected there. Given O', the attack enables any party to compute a watermark W, a key K and an object O such that W seems to be present in O' (using key K and O as the "original" object), without modifying O'. This attack is shown in Figure 2; an attack is described in [6].

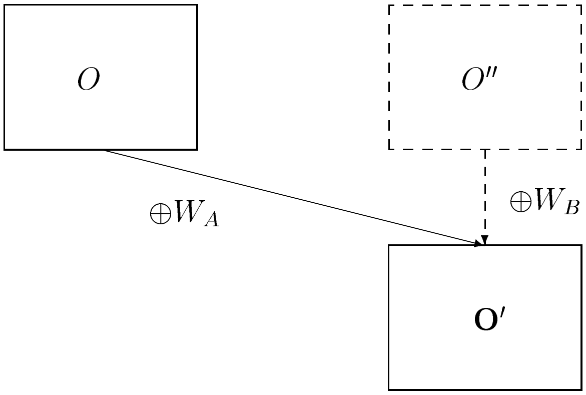

Figure 3: Inversion attack: Bob comes up with a fake watermark W_B and a fake original O'' and claims O'' to be the true original.

- A special case of ambiguity is the **inversion attack** [6]. An attacker who wants to commit a copyright infringement for an existing object O' uses an ambiguity attack to come up with a watermark W_B and an alleged original O'' such that his mark W_B is detectable in the controversial object O', although O' already contains a different mark W_A. This situation is illustrated in Figure 3. In case the watermarking system tolerates multiple insertion of watermarks (which is a consequence of the robustness), O'' will still contain the mark W_A (as O'' was derived from O'). Similarly, O seems to contain the mark W_B, as O can be thought of as a manipulated version of O'. In that case, no conclusion on the copyright status can be drawn, as no order of watermark insertion is establishable. As a consequence, the dispute resolving process, as outlined in Section 2.3, has to fail. However, as inversion is just a special case, we will concentrate on copy and ambiguity attacks in this paper.

2.5 Formal definition

Formally, copy and ambiguity attacks can be defined in the following manner:[2]

Definition 1. *Let $\mathbb{W} = \langle G, E, D \rangle$ be any watermarking scheme.*

- *Let W be a watermark, K be a watermarking key, O_1 be an arbitrary object and O'_1 its watermarked version, i.e. $D(O'_1, O_1, W, K, Aux) = \text{TRUE}$ for some auxiliary input Aux. A* copy attack *on $\mathbb{W}$ is a probabilistic algorithm*

$$\textsc{Copy}(O'_1, O_2, Aux) = \begin{cases} O'_2 & s.t.\ D(O'_2, O_2, W, K, Aux) = \text{TRUE}, \\ & \textit{with probability } \varepsilon_{\textsc{Copy}} \\ \text{FAIL} & \textit{with probability } 1 - \varepsilon_{\textsc{Copy}}. \end{cases}$$

[2] Note that [6] gave a different definition of inversion and ambiguity attacks, which involves the watermark embedder. However, we think the current definitions are more practical and realistic.

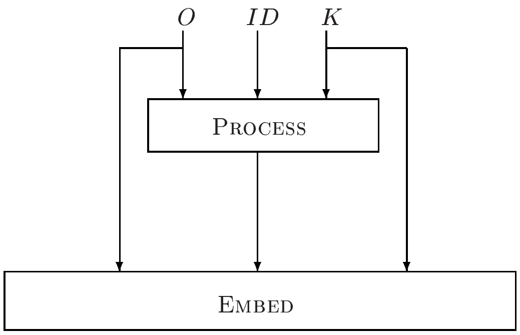

Figure 4: General structure of the embedding mechanism of a "non-invertible watermarking scheme".

- *Let O' be an arbitrary object and Aux be some auxiliary input of the detector. An* ambiguity attack *on* $\mathbb{W}$ *is a probabilistic algorithm*

$$\textsc{Ambig}(O', Aux) = \begin{cases} \langle W, K, O\rangle & s.t.\ D(O', O, W, K, Aux) = \textsc{true}, \\ & \textit{with probability } \varepsilon_{\textsc{Ambig}} \\ \textsc{fail} & \textit{with probability } 1 - \varepsilon_{\textsc{Ambig}}. \end{cases}$$

In order to assess the security of a given watermarking scheme, we need to quantify the success probability of copy and ambiguity attacks.

Definition 2. *Let $\mathbb{W} = \langle G, E, D\rangle$ be any watermarking system. A copy attack* Copy *or an ambiguity attack* Ambig *(t, ε)-breaks $\mathbb{W}$, if* Copy *(or* Ambig*) runs in time t and succeeds with probability at least ε.*

Note that both t and ε depend on the security parameter of the watermarking scheme.

Intuitively, for a watermarking scheme to be secure, every polynomial-time (copy or ambiguity) attack should have only small success probability. Note that it is not possible to require that *every* attack fails on *every* input, as it is always possible to "guess" the correct result. We say that a watermarking scheme is secure against copy or ambiguity attacks, if the success probability of any polynomial-time algorithm Copy or Ambig is negligible, i.e. if the success probability is majorized by the fraction of any polynomial.

Definition 3. *A sequence n_i of non-negative real numbers is* negligible*, if for all polynomials p there exists an integer i_0 such that $n_i < 1/p(i)$ for all $i \geq i_0$.*

If a probabilistic polynomial-time algorithm for a problem A with negligible success probability is repeated (independently) polynomially often, the resulting algorithm is still polynomial-time and has still negligible success probability.

Definition 4. *Let $\mathbb{W} = \langle G, E, D\rangle$ be any watermarking system. $\mathbb{W}$ is secure against copy or ambiguity attacks, if for all probabilistic polynomial-time algorithms* Copy *(or* Ambig*), $\varepsilon_{\textsc{Copy}}$ (or $\varepsilon_{\textsc{Ambig}}$) is negligible in the security parameter n_w.*

2.6 Previous work

In order to remove the vulnerability of dispute resolving schemes against protocol attacks, several authors proposed to use watermarking schemes that are not susceptible to ambiguity attacks as primitives; these proposals were called "non-invertible watermarking schemes". Most of them follow the general design principle depicted in Figure 4. A watermark is said to be valid, if it is detectable in an object and if it has been generated from the original object O, an identity string ID, identifying the copyright holder in case of dispute resolving applications, and the watermarking key K in a standard one-way manner; formally, a valid watermark W will be the output of an algorithm Process, constructed out of symmetric cryptographic primitives, such as hash functions or symmetric ciphers. During the watermark detection process, W has to be disclosed together with any information that allows the watermark detector to verify the validity of the watermark (basically, he has to disclose ID, O and K which allows to verify the correct generation of W). A watermark detection will only be accepted, if W is valid.

During watermark embedding, a valid mark W is produced from the "evidence" O, ID and K by running Process. For example, some authors advocated to use a random sequence, seeded by a one-way hash of the original object, as watermark [6]; alternatively, there were also attempts to produce the watermark bits of W by encrypting the original image with a block cipher like DES [14]. It was believed that these constructions make it more difficult for an attacker to perform an ambiguity attack, as he is not only faced with constructing a detectable mark W, but also with computing some fake evidence, allowing to verify the validity of W. Basically, this amounts to computing an original image O, an identity string ID and key K such that $\textsc{Process}(O, ID, K) = W$. One way to do this is to attack the watermarking scheme to obtain a detectable mark W and then to invert Process in order to obtain evidence. Since this process is computationally hard, some authors were misled to believe that their construction is "secure".

However, these constructions are not provably secure. In fact, the security depends heavily on the properties of the underlying watermarking scheme. Since watermarking schemes depend on statistical methods, the detection process has a small probability of "false alarms". Thus, with a small probability p the watermark detector will report a watermark to be present, even though it was never inserted into the object. If the false alarm probability is high, an attacker can use this fact to break the scheme. Instead of inverting Process, he chooses a tuple $\langle O, ID, K\rangle$, computes the corresponding $W = \textsc{Process}(I, ID, K)$ and checks whether this mark is detectable in O. If this is the case, he has successfully performed an ambiguity attack; if not, he discards W and chooses another tuple $\langle O, ID, K\rangle$. Obviously, the process depends heavily on the probability p and the distribution of the outputs of Process. If p is non-negligible, a polynomial-time attack against the scheme exists.

This is a very general observation that applies to constructions where Process can be evaluated by an attacker in an unlimited manner (see [3] for details). To avoid this problem, we construct the watermark in such a way that the computation of one single valid mark is already hard by incorporating digital signatures of a trusted party. This trusted party is required to generate signatures of digital objects, identity strings or watermarks, as shown in Section 4. As long as the attacker cannot query the trusted party, this construction can be proven secure.

3. CRYPTOGRAPHIC SIGNATURES

A cryptographic signature scheme (see e.g. [9]) is a triple $\mathbb{S} = \langle G_S, S_S, V_S \rangle$ of probabilistic polynomial-time algorithms:

- G_S denotes the key generation; on input 1^{n_s}, G_S outputs a pair (P, S) of keys (P is the public key, whereas S is the secret key); n_s is a security parameter determining the length of the constructed public key.

- S_S is the signing algorithm; it takes a message $m \in \mathcal{M}$, where $\mathcal{M}$ denotes the set of all possible messages (called message space), and a secret key S in the range of $G_S(1^{n_s})$. S_S outputs a string s called the signature of m.

- V_S models the signature verification process; on input s, m and P, the algorithm V_S determines whether s is a valid signature of m under public key P and outputs TRUE or FALSE accordingly.

We say that a signature scheme is secure, if it is secure against existential forgery of signatures under a chosen-message attack [8]. Technically, security is defined via a game between an attacker (Eve) and the signer (Alice); let P be the public key of Alice:

- Eve picks a message $m_1 \in \mathcal{M}$ and asks Alice to provide a corresponding signature s_1 such that $V_S(m_1, s_1, P) =$ TRUE. Alice complies.

- Eve does polynomial-time computations and, given m_1 and s_1, comes up with a message m_2, whose signature she wants to see. Alice again provides a signature s_2 such that $V_S(m_2, s_2, P) =$ TRUE.

- Eve and Alice continue this game until Eve either fails or outputs (after a certain number q of iterations) a pair $\langle m, s \rangle$, such that $V_S(m, s, P) =$ TRUE and $m \notin \{m_1, \ldots, m_q\}$, i.e. m was not sent to the oracle "Alice" previously.

In other words, Eve's goal is to compute a signature s on a *new* message m without asking Alice about the concrete signature. Alice plays the role of an oracle, which computes valid signatures. Formally, Eve's strategy can be described by a probabilistic polynomial-time algorithm with access to a signing oracle. We say that a signature scheme is secure, if the success probability for each such probabilistic algorithm is negligible:

DEFINITION 5. *A signature scheme $\mathbb{S}$ is* secure *over message space $\mathcal{M}$, if any probabilistic polynomial-time attacker following the above game has negligible success probability (negligible in the security parameter n_s). The probability is taken over all random coin flips during the game.*

To allow a quantitative analysis of security, we use the following standard definition [9]:

DEFINITION 6. *A signature scheme $\mathbb{S}$ is (t, q, ε)-secure over message space $\mathcal{M}$, if for any adversary who runs in time t and makes $\leq q$ oracle queries, the success probability is at most ε. Consequently, an adversary (t, q, ε)-breaks the scheme, if there exists a probabilistic polynomial-time algorithm that runs in time t, makes at most q oracle queries and succeeds in the above game with probability at least ε.*

The development of provably secure signature schemes is an active area of research, as not every cryptographic signature satisfies this security property. In order to provide an adequate level of security, long key sizes must be used in cryptographic signatures, yielding rather long signature sizes in the range of several thousand bits. This may be a limitating factor if we want to embed cryptographic signatures as watermarks, as some watermarking schemes allow only to embed a few bits in a robust manner. In this respect, cryptographic signature schemes that yield to short signatures are of central importance; an interesting attempt is described in [4], although the security of the scheme is not well-understood.

4. SECURE WATERMARKING SCHEMES

In this section, we show how watermarking schemes can be secured against some protocol attacks. The main idea is to partition the set of all possible watermarks $\{0, 1\}^*$ into two disjoint subsets, namely the set of all "valid" watermarks $\mathcal{V}$ and the set of "invalid" watermarks $\mathcal{I}$. For a successful watermark verification, we require that the watermark is both *valid* and *detectable* in a multimedia object. The set $\mathcal{V}$ will be constructed in such a way that it is infeasible for an attacker to compute any element $W \in \mathcal{V}$ in polynomial-time. This can be achieved by using cryptographic signatures of a trusted third party as a part of the watermarks $W \in \mathcal{V}$.

4.1 Cryptographic watermarks

Given a signature scheme $\mathbb{S} = \langle G_S, S_S, V_S \rangle$ that is secure against existential forgery of messages, we will construct a "secure" watermarking system $\mathbb{P} = \langle G_P, E_P, D_P \rangle$ on top of a traditional watermarking scheme $\mathbb{W} = \langle G, E, D \rangle$. The only requirement is that the watermarking scheme $\mathbb{W}$ allows to insert sufficiently large watermarks into digital objects.

In order to establish the security of the new scheme, we show that a successful copy or ambiguity attack amounts to breaking the underlying signature scheme. More precisely, we show that any copy or ambiguity attack that (t, ε)-breaks $\mathbb{P}$ can be converted into an attack that breaks the underlying signature scheme $\mathbb{S}$ with little overhead and similar probability.

We investigate three different constructions in this paper, where the watermark consists of some sort of identity string ID chosen by the copyright holder, concatenated with one or two cryptographic signatures of a trusted third party TTP (with secret key S and public key P). The trusted third party is used to generate signatures of the original, unmarked multimedia object O, an identity string ID and a watermarking key K (or of some string derived from it):

Pattern A:	ID	$S_S(O, S)$	$S_S(ID \| K, S)$
Pattern B:	ID	$S_S(O \| ID \| K, S)$	
Pattern C:	ID	$S_S(O \otimes (ID \| K), S)$	

The operator $s_1 \otimes s_2$ is a special XOR operation; if $|s_1| = |s_2|$, $\otimes$ denotes the ordinary XOR. If $|s_1| < |s_2|$ or $|s_1| > |s_2|$ the smaller string is repeated in a cyclic manner (and cut off at the appropriate position) before computing the XOR operation. We will assume in this section that the concatenation is invertible, i.e. that the length of the signatures is constant and known in advance.

For a fixed pattern, the set of all watermark strings containing valid signatures will form the set of valid watermarks $\mathcal{V}$. During this section we make the assumption that the attacker *cannot* query the trusted third party (we call these attacks *passive*); Section 4.3 discusses possible extensions that remove this assumption partly. Hence, an attacker who wants to compute elements of $\mathcal{V}$ is faced to "forge" signatures of the trusted third party. It is evident from the construction that the use of the trusted third party is a limiting factor for the usability of the scheme; unfortunately, the queries to the TTP cannot be avoided.

Here, the string ID denotes the payload of the watermark (i.e., bits that must be stored in the digital objects in addition to the digital signature). For example, depending on the application, ID might be some description of the copyright status of the object or an identity string of the copyright holder.

In the detection phase, it will be checked whether the watermark is *present* and whether it is *valid* by verifying its structure and the contained cryptographic signature(s). If both tests passed, the watermark is said to be present.

The watermarking scheme $\mathbb{P}_x = \langle G_{P,x}, E_{P,x}, D_{P,x} \rangle$, $x \in \{A, B, C\}$ will be constructed in the following manner:

Key generation $G_{P,x}$: equivalent to G.

Embedding $E_{P,x}$: Given an original object O, an identity string ID and a key K, compute a watermark string W according to pattern x defined above:

- for $x = A$, set $W = ID \| S_S(O, S) \| S_S(ID \| K, S)$,
- for $x = B$, set $W = ID \| S_S(O \| ID \| K, S)$ and
- for $x = C$, set $W = ID \| S_S(O \otimes (ID \| K), S)$.

To obtain the appropriate signature, the trusted party is queried during the embedding process. Embed W in O using the embedding function E: $O' = E(O, W, K)$. Take O' as output of $E_{P,x}$.

Detection $D_{P,x}$: The detection process is detailed in Figure 5; O' denotes an allegedly watermarked object, O the alleged original, W a watermark, K a watermark key and P denotes the public key of the trusted third party (auxiliary input of the detector).

According to Kerckhoffs' principle [12], we will assume that an attacker has complete knowledge of the watermarking scheme $\mathbb{P}$. We will show that if an attacker (Eve) successfully performs a *passive* ambiguity attack or a copy attack, she can also forge signatures of the trusted third party, which is computationally infeasible.

4.2 Formal security proof

The following theorem establishes the security of $\mathbb{P}_A$:

```
$\mathrm{D}_{P,A}(O', O, W, K, P)$
/* Detection process for pattern A */
  $W = W_1 \| W_2 \| W_3$
  if $D(O', O, W, K) = $ FALSE then
    return FALSE
  fi
  if $V_S(O, W_2, P) = $ TRUE and
    $V_S(W_1 \| K, W_3, P) = $ TRUE then
    return TRUE else
    return FALSE
  fi
```

```
$\mathrm{D}_{P,B}(O', O, W, K, P)$
/* Detection process for pattern B */
  $W = W_1 \| W_2$
  if $D(O', O, W, K) = $ FALSE then
    return FALSE
  fi
  if $V_S(O \| W_1 \| K, W_2, P) = $ TRUE then
    return TRUE else
    return FALSE
  fi
```

```
$\mathrm{D}_{P,C}(O', O, W, K, P)$
/* Detection process for pattern C */
  $W = W_1 \| W_2$
  if $D(O', O, W, K) = $ FALSE then
    return FALSE
  fi
  if $V_S(O \otimes (W_1 \| K), W_2, P) = $ TRUE then
    return TRUE else
    return FALSE
  fi
```

Figure 5: Watermark detection process.

Theorem 1. *For each passive ambiguity attack that (t, ε)-breaks $\mathbb{P}_A$ for objects of length n, one can construct a forging algorithm that $(t + t_S + O(n), 0, \varepsilon)$-breaks the underlying signature scheme $\mathbb{S}$. For each copy attack that (t, ε)-breaks $\mathbb{P}_A$, one can construct a forging algorithm that $(t + 2t_S + t_G + p(n), 2, \varepsilon(1 - 2^{-n}))$-breaks $\mathbb{S}$ for some polynomial p.*

Proof. We first show the security against ambiguity attacks. Suppose Ambig is a passive ambiguity attack that (t, ε)-breaks $\mathbb{P}_A$ for objects of length n. We construct a signature forging algorithm Forge$_n$ in the following manner; let P be any public signature key:

```
                                          | runtime  | success
Forge$_n(P)$                              |          |
  $O' \leftarrow$ Sample$(n)$             | $t_S$    | $1$
  $\langle W, K, O \rangle \leftarrow$ Ambig$(O', P)$ | $t$ | $\varepsilon$
  if attack is successful                 |          |
    $W = W_1 \| W_2 \| W_3$               | $O(1)$   | $1$
    output $\langle W_1 \| K, W_3 \rangle$ | $O(n)$  | $1$
  else                                    |          |
    output FAIL                           | $O(1)$   | $1$
```

Forge$_n$ runs in time $t + t_S + O(1)$ and produces a signature forgery (for public key P). This signature is valid, since a successful ambiguity attack implies $V_S(W_1 \| K, W_3, P) =$

TRUE, as shown in Figure 5. Thus, Forge_n $(t + t_S + O(1), 0, \varepsilon)$-breaks the signature scheme.

Suppose now that COPY is a copy attack that (t, ε)-breaks $\mathbb{P}_A$ for objects of length n and watermarking keys of length n_w. We construct a signature forging algorithm Forge'_{n,n_w} in the following manner:

```
                                          | runtime     | success
$\text{Forge}'_{n,n_w}(P)$                |             |
  $K \leftarrow G(1^{n_w})$               | $t_G$       | $1$
  $O_1 \leftarrow \text{Sample}(n)$       | $t_S$       | $1$
  $O_2 \leftarrow \text{Sample}(n)$       | $t_S$       | $1$
  if $O_1 = O_2$ output FAIL              | poly($n$)   | $1 - 2^{-n}$
  $ID \leftarrow \text{Random}()$         | $O(1)$      | $1$
  $S_1 \leftarrow \text{Query}_P(ID\|K)$  | $1$         | $1$
  $S_2 \leftarrow \text{Query}_P(O_1)$    | $1$         | $1$
  $O'_1 \leftarrow E(O_1, ID\|S_1\|S_2, K)$ | poly($n$) | $1$
  $O'_2 \leftarrow \text{Copy}(O'_1, O_2, P)$ | $t$     | $\varepsilon$
  if attack is successful                 |             |
    output $\langle O_2, S_2 \rangle$     | poly($n$)   | $1$
  else                                    |             |
    output FAIL                           | $O(1)$      | $1$
```

Here, "poly" stands for specific, but unspecified, polynomials and t_G for the runtime of G. Query_P denotes a signature oracle query. Intuitively, the attack generates a valid watermarked object O'_1 and uses the copy attack to copy the contained watermark onto a different object O_2. If COPY succeeds, the output of Forge'_{n,n_w} is a valid forgery (again, a successful copy attack implies, by Figure 5, $V_S(O_2, S_2, P) = \text{TRUE}$). Obviously Forge'_{n,n_w} runs in time $t_G + 2t_S + t + \text{poly}(n)$, and makes two oracle queries. As the forging algorithm can fail independently at two different steps, its success probability is given by $\varepsilon(1 - 2^{-n})$. Thus, we have constructed an algorithm that $(t + 2t_S + t_G + \text{poly}(n), 2, \varepsilon(1 - 2^{-n}))$-breaks the underlying signature scheme. □

A similar security property holds for $\mathbb{P}_B$ and $\mathbb{P}_C$:

THEOREM 2. *Any passive ambiguity attack that (t, ε)-breaks $\mathbb{P}_B$ or $\mathbb{P}_C$ for objects of length n can be extended to an attack that $(t + t_S + O(n), 0, \varepsilon)$-breaks the underlying signature scheme $\mathbb{S}$. Similarly, every copy attack that (t, ε)-breaks $\mathbb{P}_B$ or $\mathbb{P}_C$ can be extended to an attack that $(t + 2t_S + t_G + p(n), 1, \varepsilon(1 - 2^{-n}))$-breaks $\mathbb{S}$ for some polynomial p.*

PROOF. (Sketch) The proof is analogous to the proof of Theorem 1. Consider pattern B first. In an ambiguity attack, Forge_n runs AMBIG to obtain a mark $W = W_1\|W_2$ and outputs $\langle O\|W_1\|K, W_2\rangle$. In a copy attack, Forge'_{n,n_w} constructs a valid object O'_1 containing a watermark $W = W_1\|W_2$ (note that only one oracle query is required) and simulates COPY on O'_1 and O_2. This produces a valid watermarked object O'_2 if the attack succeeds. Finally, Forge'_{n,n_w} outputs $\langle O_2\|W_1\|K, W_2\rangle$. It can be seen easily that Forge_n and Forge'_{n,n_w} output valid forgeries in case the copy or ambiguity attack succeeded; furthermore, they satisfy the required time and probability bounds. The proof for pattern C is similar. □

We can thus conclude:

THEOREM 3. *Suppose that $\mathbb{S}$ is secure against existential forgery of messages (under a chosen message attack). Then, the constructed watermarking schemes $\mathbb{P}_A, \mathbb{P}_B$ and $\mathbb{P}_C$ are secure against copy and passive ambiguity attacks.*

PROOF. Assume the opposite, i.e., there is a copy or a passive ambiguity attack that (t, ε)-breaks $\mathbb{P}$ with non-negligible success probability ε and an arbitrary polynomial t. Then, by Theorems 1 and 2 there exists an attack that $(\bar{t}, q, \bar{\varepsilon})$-breaks the underlying signature scheme with non-negligible success probability $\bar{\varepsilon} = k\varepsilon$ (where $\frac{1}{2} < k \leq 1$), $q \leq 2$ and a polynomial $\bar{t}$. This contradicts the assumption. □

From a practical perspective, pattern B is preferable to pattern A for two reasons. First, the constructed watermark is shorter, increasing the practical feasibility of the approach. Second, if the same string ID and key K is used for several multimedia objects, an attacker might gain the signature $S_S(ID\|K, S)$ after a successful watermark detection. This knowledge can be used in "cut-and-paste" attacks. In fact, in pattern A, an attacker can, given two valid cryptographic watermarks, compute a third one by appropriately pasting parts of the watermarks together. This does not contradict the above security property; however, it becomes an issue in interactive ambiguity attacks, as described in Section 4.3. Therefore, we advocate to use pattern B.

It is evident that variants of the patterns A, B and C are conceivable, for which security can be established in a similar way. We leave this to future research.

4.3 Interactive ambiguity attacks

The security result of the previous section was derived under the assumption that an attacker against the watermarking scheme does *only* local computations, but cannot query the trusted party for a signature of a string chosen by him. In a dispute resolving scenario, this assumption does not hold in general. For example, an attacker might be a legitimate owner of some different object and thus must have access to the trusted party to obtain signatures for his own objects. As the trusted third party cannot distinguish between a true author and an attacker, *interactive attacks* become possible. During such an attack, he can query the trusted third party TTP for legitimate watermarks built from an arbitrary object O, key K and identity string ID.

It is easy to see that the construction is not secure against unlimited interactive attacks. The argument is similar to that presented in Section 2.6. If the attacker is able to make an unlimited (but polynomial) number of queries and use one of the watermarks received from the trusted third party as output, then he can again "guess" a false original, identity string and key and obtain a signature from the trusted party. He repeats the process until he finds a detectable mark. Again the false positives rate of the underlying watermarking scheme would be a fundamental factor limiting the security of the scheme. *Up to now it is an open research question whether it is possible to construct watermarking schemes that are not susceptible to interactive ambiguity attacks, independent of the underlying embedding mechanism.*

However, one can deal with interactive ambiguity attacks in case the output of the attacker is restricted. If one requires that the watermark obtained at the end of the interactive ambiguity attack is *new*, i.e. that it contains no signature of an object, an identity string and a key that was presented to the oracle previously, the security of the proposed scheme in Section 4 can again be established.

Definition 7. *A watermark W* belongs *to the tuple $\langle O, K, ID\rangle$, if W contains a cryptographic signature of at least one element of the set $\{O, K, ID\}$, which may be concatenated with an arbitrary string (either from the left or right).*

Definition 8. *Let O' be an arbitrary object and Aux be some auxiliary input of the detector. A* limited interactive ambiguity attack *on $\mathbb{W}$ is a probabilistic algorithm* LIntAmbig *with oracle access to* Ttp *such that*

$$\textsc{LIntAmbig}(O', Aux) = \begin{cases} \langle W, K, O\rangle & s.t.\ D(O', O, W, K, Aux) = \text{TRUE} \\ & \textit{and } W \textit{ does not belong to } \langle O_i, K_i, ID_i\rangle, \\ & \textit{with probability } \varepsilon_{\textsc{Ambig}} \\ \text{FAIL} & \textit{with probability } 1 - \varepsilon_{\textsc{Ambig}}, \end{cases}$$

where $\langle O_i, K_i, ID_i\rangle$ denote the queries made by LIntAmbig. *A limited interactive ambiguity attack* LIntAmbig *(t, q, ε)-breaks $\mathbb{W}$, if* LIntAmbig *runs in time t, makes at most q queries to the trusted party* Ttp *and succeeds with probability at least ε.*

Note that, due to the restriction to watermarks that do not belong to any oracle query, the "cut-and-paste" attacks described at the end of Section 4.2 are not considered valid attacks in this limited model. We can establish the security of $\mathbb{P}_A$ in the following manner:

Theorem 4. *For each limited interactive ambiguity attack that (t, q, ε)-breaks $\mathbb{P}_A$ for objects of length n, one can construct a forging algorithm that $(t + t_S + O(1), 2q, \varepsilon)$-breaks the underlying signature scheme $\mathbb{S}$.*

Proof. (Sketch) The proof is analogous to the proof of Theorem 1, where Ambig is replaced by LIntAmbig. Each oracle query of $\textsc{LIntAmbig}(O, K, ID)$ is transformed into two signature queries $\textsc{Query}_P(O)$ and $\textsc{Query}_P(ID \| K)$. Again, the constructed forging algorithm yields a valid signature forgery, as the output was never presented to the oracle as input according to the definition of a limited interactive ambiguity attack. □

In a similar way the security of $\mathbb{P}_B$ and $\mathbb{P}_C$ can be established.

5. CONCLUSION

In this paper, we addressed the problem of securing watermarking schemes against protocol attacks. We introduced a formal framework that allows to assess the security of watermarking schemes against copy and ambiguity attacks, subsuming the case of inversion attacks. Our proof method demonstrates that our methodology is applicable to a large number of protocol attacks.

We have argued in Section 2.6 that many previous attempts for providing non-invertible watermarking schemes failed if used with a watermark detector that has a large number of false positives. We provided a new construction that relies on signatures of a trusted third party. This construction was shown to be secure against *passive* ambiguity attacks and certain special cases of interactive ambiguity attacks. However, it is still not secure against attackers that can make unlimited queries to the trusted party. A possible solution in case of authorship proofs and dispute resolving is to let the trusted third party maintain a database of previous queries and return valid signatures only if there has not been a query for a perceptually similar object before (see [1]).

The above considerations raise the question whether it is theoretically possible to construct "secure" watermarking schemes that do not use a trusted party or (equivalently) use a trusted party which can be queried by an attacker in an unlimited way. Current research results suggest that the security depends heavily on the underlying detection mechanism; it might be possible that a "universal" construction for secure watermarking schemes (i.e., a construction that is secure independent of the statistics of the underlying embedding process) cannot be found. We leave this to future research.

The results in this paper immediately have consequences for watermarking-based dispute resolution protocols. In case the false positives rate of the detection mechanism is unknown, current cryptographic constructions to achieve non-invertibility may fail. The security of dispute resolving protocols that use alleged originals for inferring the order of watermark creation heavily depends on the statistical properties of the watermark detector. *It is thus questionable whether such protocols can be considered secure any more.* One possible way to avoid this problem entirely is the use of alternative methods for dispute resolving that do not require the underlying watermarking method to be non-invertible; for an overview of such constructions see [2].

6. REFERENCES

[1] A. Adelsbach, B. Pfitzmann, A. Sadeghi, "Proving Ownership of Digital Content", in *Proceedings of the Third International Workshop on Information Hiding*, Springer Lecture Notes in Computer Science, vol. 1768, 2000, pp. 117–133.

[2] A. Adelsbach, A. Sadeghi, "Advanced Techniques for Dispute Resolving and Authorship Proofs on Digital Works", in *Proceedings of the SPIE vol. 5020, Security and Watermarking of Multimedia Contents V*, 2003, pp. 677–688.

[3] A. Adelsbach, S. Katzenbeisser, A. Sadeghi, "On the Insecurity of Non-Invertible Watermarking Schemes for Dispute Resolving", *International Workshop on Digital Watermarking (IWDW'03), Proceedings*, to appear.

[4] N. Courtois, L. Goubin, J. Patarin: "The Sflash signature scheme", technical report available at `http://www.minrank.org/sflash`, 2001.

[5] I. Cox, M. Miller, J. Bloom: *Digital Watermarking*, Morgan Kaufmann, 2002.

[6] S. Craver, N. Memon, B. L. Yeo, M. M. Yeung, "Resolving Rightful Ownerships with Invisible Watermarking Techniques: Limitations, Attacks and Implications", in *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 4, 1998, pp. 573–586.

[7] F. Deguillaume, S. Voloshynovskiy, T. Pun: "Hybrid Robust Watermarking Resistant Against

Copy Attacks", in *European Signal Processing Conference*, 2002.

[8] S. Goldwasser, S. Micali, R. Rivest: "A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks", in *SIAM Journal on Computing*, vol. 17, no. 2, 1988, pp. 281–308.

[9] S. Goldwasser, M. Bellare, *Lecture Notes on Cryptography*, available at `http://www-cse.ucsd.edu/users/mihir/papers/gb.html`, 1996.

[10] G. Heileman, Y. Yang, *The Effects of Invisible Watermarking on Satellite Image Classification*, ACM CCS DRM Workshop, October 27, 2003.

[11] S. Katzenbeisser, H. Veith, "Securing Symmetric Watermarking Schemes Against Protocol Attacks", in *Proceedings of the SPIE vol. 4675, Security and Watermarking of Multimedia Contents IV*, 2002, pp. 260–268.

[12] A. Kerckhoffs, "La Cryptographie Militaire", in *Journal des Sciences Militaires*, vol. 9, 1883, pp. 5–38.

[13] M. Kutter, S. Voloshynovskiy, A. Herrigel, "The Watermark Copy Attack" in *Proceedings of the SPIE vol. 3971, Security and Watermarking of Multimedia Contents II*, 2000, pp. 371–380.

[14] L. Qiao, K. Nahrstedt, "Watermarking Schemes and Protocols for Protecting Rightful Ownership and Customer's Rights", in *Journal of Visual Communication and Image Representation*, vol. 9, No. 3, 1998, pp. 194-210.

[15] L. Qiao, K. Nahrstedt, "Non-Invertible Watermarking Methods for MPEG Encoded Audio", in *Proceedings of the SPIE vol. 3675, Security and Watermarking of Multimedia Contents*, 1999, pp. 194–202.

[16] M. Ramkumar, A. N. Akansu, "Image Watermarks and Counterfeit Attacks: Some Problems and Solutions", in *Content Security and Data Hiding in Digital Media, Proceedings*, Newark (NJ), 1999.