Zero-Knowledge Watermark Detection Resistant to Ambiguity Attacks

Qiming Li *
Dept. of Computer and Information Science
Polytechnic University
qiming.li@ieee.org

Ee-Chien Chang
Dept. of Computer Science
National University of Singapore
changec@comp.nus.edu.sg

ABSTRACT

A zero-knowledge watermark detector allows an owner to prove to a verifier that an image in question indeed contains the owner's watermark without revealing much information about the actual watermark. In such a scenario, the owner publishes a committed watermark before watermark detection so as to show that she knows the watermark before the detection. However, this does not imply that the owner can prove that she knows the watermark before the work appeared in the public. One well known counter example is the invertibility/ambiguity attacks where an adversary can create an ambiguous situation by deriving a forged watermark from a published work, and commits the forged watermark. Furthermore, the adversary may derive a watermark from existing non-watermarked images in the public domain and later claim ownership of them. One solution is to enforce certain constraints on the *valid* watermarks. In this paper we propose a zero-knowledge watermark detector that prevents the owner from cheating by ambiguity attacks. In addition, it allows the owner to publish a large number of works with different watermarks, while committing only one secret.

Categories and Subject Descriptors

K.4.4 [**Electronic Commerce**]: Security and Intellectual Property; K.6.m [**Miscellaneous**]: Security

General Terms

Security

Keywords

Zero-knowledge proof, ambiguity attack, watermark detection

*Part of the work was done when the author was in the Department of Computer Science, National University of Singapore.

MM&Sec'06, September 26–27, 2006, Geneva, Switzerland.

1. INTRODUCTION

Digital watermarking schemes have been proposed as a tool to achieve desired goals in Digital Rights Management. One possible application of digital watermarking schemes is the proof of ownership of digital works. Let us consider a typical scenario. Alice, who is an artist, created a number of works (such as digital images), and wishes to release them to the public. For each work, Alice chooses a watermark and embeds it into the work. To convince a *verifier*, Bob, that a digital work I belongs to Alice (the *prover*), they can jointly perform a watermark detection and Alice will claim that I belongs to her if her watermark is detectable in I.

There are a number of security concerns in this scenario. First of all, during watermark detection, Alice needs to be sure that Bob would not be able to acquire sufficient information that leads to successful removal of the hidden watermark. Secondly, Bob needs to be sure that Alice did not cheat by falsely claiming that her watermark is detectable while it is not. For instance, Alice cannot just tell Bob the result of the watermark detection without providing any proof.

The third, and the most subtle concern is that, even if Alice can prove that she knows a watermark W_a that is detectable in I, it is still possible that she is cheating if the watermarking scheme is vulnerable to *invertibility attacks* [10] (or more generally, *ambiguity attacks* [1]). In such attacks, for a work I that does not belong to Alice, she would find a fake watermark W_a detectable in I, and claim the ownership of I using the fake watermark W_a. Such an ambiguous situation can be avoided if Alice can prove that she knows the watermark W_a before she knows the work I.

There are several different approaches to address the first two security concerns in the literature, including the use of a Trusted Third Party (TTP), asymmetric watermarking schemes, watermark detection using a group of proxies, and zero-knowledge watermark detection protocols (Section 2 provides the details).

However, these approaches are not sufficient to address the third security concern. That is, they cannot prevent Alice from using invertibility/ambiguity attacks, if the underlying watermarking scheme is vulnerable. Li and Chang proposed a non-invertible watermarking scheme based on a variant of spread-spectrum watermarking schemes [17], and showed that it is secure if the false alarm is low. Their main idea is to require the watermarks to be *valid* in the sense that they must be the output of a one-way function, and Alice proves that the watermarks are not derived from the

work by proving that she knows how to invert the one-way function on the watermarks. However, their scheme does not directly address the first two security concerns.

In addition, many watermarking schemes are vulnerable to a form of removal attacks where an adversary removes the watermark by observing many different works embedded with the same watermark. One possible solution is to use a different watermark for each work. However, it is inflexible to require the owner to commit every watermark. Hence, it is desirable that zero-knowledge proofs can be done for a large number of works with only a small constant number of committed secrets.

In this paper, we study a more general problem: How to design a watermarking scheme that allows Alice to publish a possibly large number of works, provides mechanisms for the watermark detection in zero-knowledge, and prevents Alice from cheating by invertibility/ambiguity attacks.

Our main idea is the following. To combat invertibility attacks, we employ a method that is slightly different from a known provably secure non-invertible scheme [17]. We use the same cryptographically secure pseudo-random number generator as in [17], and require that the watermarks are all generated from the generator with the same seed but with different indices associated with different works. In other words, to prove ownership, Alice has to show that the image contains a valid watermark generated from her committed secret and an index. The scheme in [17] can be considered as a special case when the index is always fixed. We also observe that there are existing zero-knowledge proofs that can be used to prove the correctness of computation in a field, and inequalities of secrets. Based on that, we propose a zero-knowledge proof for the non-invertible watermarking scheme such that after the protocol is followed, Bob will be convinced that the watermark detection is done correctly, and Alice did not cheat.

In Section 2 we discuss related works on the problem. In Section 3 we define our problem more formally, and introduce some background on the secure pseudo-random number generator and zero-knowledge proofs. We analyze the security requirements in Section 4, and propose our zero-knowledge proof protocols in Section 5. The analysis of security follows in Section 6. We conclude our paper in Section 7.

2. RELATED WORKS

In an ideal situation, all the three security concerns mentioned in Section 1 can be addressed if a *trusted third party* (TTP) is available. In particular, Alice can register all her watermarks with the TTP before she publishes her works, and the TTP performs the watermark detection on behalf of Alice. However, it has been well understood that the designer of a watermarking scheme should make as little trust assumption as possible. For example, putting too much trust in the sellers leads to the "customers rights problem" [20, 18]. There are several ways to deal with the first two security concerns without using a TTP. One way is to employ asymmetric watermarking schemes such as [22, 15, 13], in which the key to embed a watermark into a work is different from the key required during detection. Unfortunately, known methods are not satisfactory and there are a number of successful attacks [12]. An alternative is to replace the TTP with a group of proxies [16] and employ a secret sharing scheme, where the security is achieved if the majority of the parties are honest. The scheme also allows the owner to delegate watermark detection to third-parties without revealing her secret watermark.

A different approach uses zero-knowledge proofs, which are well studied primitives in cryptography. Recently, Adelsbach et al. [3] and Craver [9] proposed the use of zero-knowledge proofs in watermarking. Yu and Lu proposed a similar method based on a specially designed robust watermarking scheme [23]. In most cases, we can consider the watermark as the secret, and the property to be proved is the existence of the watermark. Hence, zero-knowledge proofs allows Alice to prove the existence of a certain (hidden) watermark in a given image without revealing anything else.

Craver et al. [10] first studied the *invertibility attacks*, where an attacker derives a watermark that is detectable in a given work, and reverse the watermark embedding process to obtain a fake original. In that case, one cannot distinguish an honest owner and an attacker without any additional evidence. A generalized notion, called *ambiguity attacks*, is given by Adelsbach et al. [1]. Craver et al. proposed a "non-invertible" scheme where the watermarks are computed as the hash values of the original [10]. The idea is that the attacker would have to invert the hash function to obtain a faked original.

This method was later proved insecure by Ramkumar et al., who also gave an improved scheme [21]. It was pointed out in [2, 1] that a watermarking scheme cannot be non-invertible if the false alarm is high. Yu and Lu claimed that their proposed zero-knowledge watermark detector [23] is resistant to ambiguity attacks by incorporating a one-way function based on the difficulty in finding Hamiltonian cycles in graphs. However, it seems that their security proof is not sufficient, since the difficulty of inverting the one-way function does not imply the resistance against ambiguity attacks. When the false alarm is very low, Li and Chang [17] showed that a provably secure non-invertible watermarking scheme is possible by requiring the watermarks to be generated from a cryptographically secure pseudo-random number generator.

3. NOTATIONS AND MODELS

3.1 The Watermarking Problem

Assume that there is a prover, Alice, who owns a database of n images $I_1, ..., I_n$. Each image I_i is associated with a unique key K_i. For example, K_i could be the primary key of the i-th tuple in the database. We will call K_i's the *tuple keys*. In fact, the use of the terms "database" and "image" is just for convenience, since we can easily apply the same results to any watermarking system where a unique key is somehow associated with each of the digital work to be watermarked[1].

We assume that there exists a secret master key M that is large enough so that it is not easily guessed (e.g., $|M| = 1024$). The tuple keys, however, are made public. Each image I_i is watermarked by a secret watermark W_i, which is computed by $W_i = f(M, K_i)$ for some known function $f(\cdot, \cdot)$. We also assume that good embedding and detection

[1]This key could be associated with the digital work in many ways, e.g., it can be in the header of a digital image. It may be easy for Bob to break this association, but he seem to lack the incentive to do so.

algorithms are available such that general watermarking requirements, such as robustness, false alarm and distortion can be met (e.g., the scheme presented in Section 3.2). In particular, we require that the false alarm of the scheme should be negligible.

Note that the master key and the tuple keys are necessary to facilitate the use of the scheme in [17]. These keys are only used during the watermark generation process and are not related directly to the embedding process of the watermarks. Similarly, the requirement on false alarm is the prerequisite for the scheme in [17] to be non-invertible.

We assume that there is a verifier, Bob, who can submit any image J with key K_j to the prover for watermark detection. Our goal is to find a suitable watermark generation function $f(\cdot,\cdot)$, and derive a zero-knowledge protocol for watermark detection such that both the prover and the verifier cannot cheat, and the verifier cannot learn much information about the secret watermarks, except the existence of W_j in J. The detailed security requirements will be given in Section 4.

3.2 Watermarking Model

To illustrate our idea, we employ a relatively simple variant of the well-known spread spectrum method [8] in our discussion. Other types of watermarking schemes could be adapted with minor modifications.

We assume that the images and watermarks are "discretized". An image I can be represented by an integer vector $\langle x_1, \ldots, x_l \rangle$, and each element of the vector $x_i \in \mathbb{Z}_e$, where e is some sufficiently large integer determined by media representation. Let $W = \langle w_1, \ldots, w_l \rangle$ be the watermark to be embedded, where $w_i \in \mathbb{Z}_d$, and $d < e$ is a constant determined by the desired energy of the watermark.

During embedding, given an image I and watermark W, the watermarked image $\widetilde{I}$ is computed (by the owner) as

$$\widetilde{I} = I + W \mod e. \tag{1}$$

During detection, given an image J, the correlation

$$c = J \cdot W \tag{2}$$

is computed, where $\cdot$ is the vector inner product. If the correlation exceeds certain threshold T, then the image is declared as watermarked, otherwise it is declared as non-watermarked.

3.3 A Secure PRNG

Similar to the scheme in [17], it is essential in our solution to have a secure pseudo-random number generator (PRNG), whose output cannot be predicted but can be verified by some zero-knowledge proof protocols [2].

The generator we employ is a variant of the simple Blum-Blum-Shub generator [5] based on the difficulty of factoring a Blum number which is the product of two large primes, each congruent to 3 (mod 4). More specifically, let N be a Blum number, and $s \in \mathbb{Z}_N^*$ be a secret seed to the pseudo-random number generator $G_s(\cdot)$. We require that s is a quadratic residue modulo N, i.e., $s = s_0^2 \mod N$ for some s_0. The i-th number given by the generator is computed as

$$G_s(i) = (s^{2^i} \mod N) \mod d \tag{3}$$

where $d = 2^\delta$, and $\delta = O(\log(\log N))$ is a small integer. In other words, we square the number each time to get the next number, and take a few least significant bits from each number as the output of the generator. When N is sufficiently large, the scheme is secure[3].

The security of the generator means that given any sequence of some outputs, it is infeasible to predict the output either before or after this sequence. This also implies that given any set of outputs, it is infeasible to compute the seed, or the indices of the outputs (where it is in the entire sequence).

3.4 Commitment Schemes

Commitment schemes are essential building blocks of zero-knowledge proofs. Generally speaking, a commitment scheme allows one party (say, Alice) to seal a piece of secret data into an envelope to show that she would not change her mind during some protocol.

More specifically, the commitment $com(x)$ is a one-way function on x, such that given $com(x)$ it is difficult to guess the value of x, and it is also difficult to find an y such that $com(x) = com(y)$. In this paper we do not restrict the type of commitment schemes used for the proof, as long as they can be used in the zero-knowledge proofs. For example, we can use the commitment schemes in [19, 14, 11], and the zero-knowledge proofs in [7, 6].

Note that the security of these commitment schemes and zero-knowledge proofs reply on certain assumptions on the difficulties of some computational problems (such as integer factorization and discrete logarithm), which we will follow implicitly for our schemes.

3.5 Zero-Knowledge Proofs

In our solution, we employ the interactive proof protocols proposed in [7] and [6], the former provides generic methods to prove arithmetic properties of committed values, and the latter proposed an efficient protocol to prove that a committed number lies in a certain interval.

These protocols can be substituted by other protocols as long as the desired properties can be proved in zero-knowledge. In fact, since the properties need to be proved in our solution are relatively simple, and some conditions are relaxed (for example, we allow users to see the tuple keys instead of committing them), it is possible to derive more efficient protocols just for our solution. However, in this paper we only present a general solution, in which we are interested in how the problem can be solved rather than building up an optimal construction in terms of computational efficiency.

In a zero-knowledge proof, a *prover* (Alice) proves some properties of a secret to a *verifier* (Bob) without revealing any other information. In particular, let $com(x)$ be the commitment value of x, and for x the proposition $p(x)$ holds, then we denote the protocol that proves this property by

$$PK\{(x) : p(x)\}.$$

We assume that the commitment $com(x)$ and public values (such as the tuple keys) are known by both parties and are used implicitly in the protocol. For security reasons, some of the parameters of the commitment scheme and the zero-knowledge proofs should be chosen by the verifier or by both

[2] In fact, any one-way function that can be verified in zero-knowledge would suffice.

[3] The security relies on the difficulty to factorize N.

parties cooperatively[4].

Following the constructions in [7], the following arithmetic properties can be proved in zero-knowledge.

Secret Modular Addition and Multiplication. Given commitment values $com(w)$, $com(x)$, $com(y)$, $com(z)$, $com(n)$, the prover can prove the following

$$\begin{aligned} &PK\{(w,x,y,n) : w + x \equiv y \mod n\} \\ &PK\{(w,x,z,n) : wx \equiv z \mod n\}. \end{aligned} \tag{4}$$

Secret Modular Exponentiation. Given commitment values $com(x)$, $com(y)$, $com(n)$ and constant α, the prover can convince the verifier that $x^\alpha \equiv y \mod n$ by

$$PK\{(x,y,n) : x^\alpha \equiv y \mod n\}. \tag{5}$$

Note that this is a special case of the algorithm in [7], which can be greatly simplified when α is known to the verifier.

Also, from [6] we have a efficient protocol to prove the following inequalities.

Inequality of a Secret. Given commitment of x, we can prove one of the following

$$\begin{aligned} &PK\{(x) : x \geq 0\} \\ &PK\{(x) : x \leq 0\}. \end{aligned} \tag{6}$$

4. SECURITY REQUIREMENTS

Following the discussions in Section 1, the security considerations of the system consists of two categories. Firstly, Alice should not be able to cheat. In particular, if Alice claims that an image from Bob is indeed watermarked, she has to show evidence that she knew the watermark before she sees the image, as well as the detection has been performed correctly.

On the other hand, it is also cheating when Alice claims that the image is not watermarked when it is. However, Alice lacks the incentive to do so in most practical applications. Therefore, it is only a secondary consideration and we only give a zero-knowledge proof for Alice to prove that a given image does not contain a certain watermark[5].

Secondly, Bob should not be able to gain much useful information that leads to successful attacks during watermark detection. In particular, we require that from the results of one detection, Bob should not be able to compute Alice's master key, or any of the watermarks. We also require that given some Alice's watermarks, Bob should not be able to compute the master key, or any other watermarks.

More formally, we consider the following security requirements.

S1. Prior Knowledge of Watermark. Given image I, it should be infeasible for Alice to compute a master key M, a tuple key K, and a watermark W such that W is generated from M and K and is detectable in I. Since we assume that the tuple key K is associated with each work I, it can be provided by the verifier together with I. In this case we only need a weaker requirement that it should be infeasible to compute M and W given I and K. However, it can be seen from Section 3.3 that these two requirements are actually equivalent under our construction.

S2. Correctness of Claims. Given master key M, tuple key K, and image I, Alice should be able to prove, in zero-knowledge, that there exists a watermark W and a tuple key K such that W is correctly computed from M and K, and that W is (or is not) detectable in image I.

S3. Detection Should Be Secure. During the detection, Bob should not obtain any information except the existence of a certain watermark in a given image.

S4. Inferences Should Be Infeasible. Given watermarks $W_1, \ldots, W_{n-1}$ from Alice, as well as the tuple keys $K_1, \ldots, K_{n-1}, K_n$, it should be infeasible for Bob to compute W_n, or the master key M. [6]

Note that the first two requirements belong to the first category and the last two requirements belong to the second category.

5. OUR SOLUTION

In our solution, Alice first decides on a secret master key M, and for each image I_i, she derives the watermarked work as the following.

1. Choose a tuple key K_i.
2. Compute the watermark W_i using a PRNG with M as the seed and K_i as the index, (Section 5.1).
3. Embed W_i into I_i.

During detection, given image J (and optionally K), Alice proves that there is a watermark W detectable in J, and that she knows an M and a K (if it is not given) such that W is correctly computed from G with M and K as inputs. In the following, we give the detailed algorithm for the derivation of the watermarks, as well as the zero-knowledge proof protocols used in the verification. The embedding and detection of the watermark use the algorithms described in Section 3.2.

5.1 Watermark Derivation

As described in Section 3.2, our watermarks are all vectors, each of which consists of l integer elements. That is, $W_i = \langle w_{i,1}, \ldots, w_{i,l} \rangle$, where each $w_{i,j} \in \mathbb{Z}_d$. We require that the master key M also be a vector with l integer elements, such that $M = \langle m_1, \ldots, m_l \rangle$ and $m_j \in \mathbb{Z}_N$.

The coefficients of the watermarks are derived by applying the secure pseudo-random number generator (Section 3.3) with the corresponding coefficients from the master key as the seeds and the tuple keys as the indices. More formally, we compute the j-th coefficient of the i-th watermark by

$$w_{i,j} \triangleq G_{m_j}(K_i) = (m_j^{2^{K_i}} \mod N) \mod d \tag{7}$$

[4]The actual method to choose the parameters depends on the commitment scheme and zero-knowledge proofs employed.

[5]Note that by iterating the procedure for all her watermarks she can prove the non-ownership.

[6]This requirement is to prevent the attacker who happen to know a few watermarks from deducing other watermarks.

where $d \in O(\log N)$ is determined by the desirable energy of the watermark. For convenience, let $w'_{i,j} = m_j^{2^{K_i}} \mod N$, and denote $W'_i = \langle w'_{i,1}, \ldots, w'_{i,l} \rangle$.

The owner then commits the vectors M, W_i and W'_i by computing the commitments of their elements $com(m_j)$, $com(w_{i,j})$, $com(w'_{i,j})$ for all possible values of i and j. After that the commitment $com(M)$ is published such that anyone who wishes to perform a watermark detection can access and get a copy of the value beforehand. The values of N, d and tuple keys K_i are made public. In case the parameter N is chosen on-the-fly for each detection, the commitment of M only needs to be given to the verifier at the beginning of each detection, together with the commitment of the watermark.

When N and $com(M)$ are fixed (e.g., when it is required to be chosen and signed by a trusted third party), the owner only needs to show that M is properly committed, and does not need a third party to certify the validity of all the watermarks. This is an advantage compared to the straight forward solution where all the watermarks are signed by some trusted third party before the works are distributed. Note that when the false alarm of the underlying watermarking scheme is negligible, having a trusted third party to choose and sign N and M would be unnecessary.

5.2 Zero-Knowledge Proofs

The prover needs to prove the following: First, the watermarks are computed correctly according to (7). Secondly, the watermark detection is done correctly according to Section 3.2. Hence, our zero-knowledge proofs consist of two protocols. One protocol proves the correct computation of each watermark W_i, and the other proves the correct watermark detection.

Let $Correct(x)$ be the proposition "x is computed correctly". To show that watermark W_i is computed correctly, the prover can convince the verifier that each coefficient in W_i is computed correctly. That is,

$$\begin{aligned} &PK\{(M, W_i, W'_i) : Correct(W_i)\} = \\ &PK\{(m_j, w_{i,j}, w'_{i,j}) : Correct(w_{i,j})\}. \end{aligned} \tag{8}$$

Next, the correct computation of $w_{i,j}$ can be confirmed by verifying correct computation from m_j to $w'_{i,j}$

$$\begin{aligned} &PK\{(m_j, w'_{i,j}) : Correct(w'_{i,j})\} = \\ &PK\{(m_j, w'_{i,j}) : w'_{i,j} \equiv m_j^{2^{K_i}} \mod N\} \end{aligned} \tag{9}$$

and from $w'_{i,j}$ to $w_{i,j}$

$$\begin{aligned} &PK\{(w'_{i,j}, w_{i,j}) : Correct(w_{i,j})\} = \\ &PK\{(w_{i,j}, w'_{i,j}) : w_{i,j} \equiv w'_{i,j} \mod d\}. \end{aligned} \tag{10}$$

For the second protocol, let $Detect(\cdot)$ denote the watermark detection. That is, when detection is correctly done, $Detect(W, J) = 1$ if and only if the correlation of the image and the watermark is greater than a certain threshold. Hence, for detection threshold T, the second protocol can be expressed as

$$\begin{aligned} &PK\{(W_i) : Correct(Detect(W_i, J))\} = \\ &((Detect(W_i, J) = 1) \wedge PK\{(W_i) : (J \cdot W_i \geq T + 1)\}) \vee \\ &((Detect(W_i, J) \neq 1) \wedge PK\{(W_i) : (J \cdot W_i \leq T)\}). \end{aligned} \tag{11}$$

To prove the relationship between the correlation $c = J \cdot W_i$ and T, the prover first computes the value of $c = J \cdot W_i$, then the prover performs different actions based on the result. If $c > T$, the prover replies with a 1 to indicate that the image is watermarked, together with the commitment $com(c-(T+1))$. If $c \leq T$, the prover replies with a 0 together with the commitment $com(c - T)$.

When $c > T$, the prover uses the following protocol

$$\begin{aligned} &PK\{(W_i) : (J \cdot W_i \geq T + 1)\} = \\ &PK\{(W_i, c - (T+1)) : \sum_{k=1}^{l} w_{i,k} J_k - (T+1) = c - (T+1)\} \\ &\wedge PK\{(c - (T+1)) : c - (T+1) \geq 0\} \end{aligned} \tag{12}$$

and when $c \leq T$, the prover uses another protocol

$$\begin{aligned} &PK\{(W_i) : (J \cdot W_i \leq T)\} = \\ &PK\{(W_i, c - T)) : \sum_{k=1}^{l} w_{i,k} J_k - T = c - T\} \\ &\wedge PK\{(c - T) : c - T \leq 0\}. \end{aligned} \tag{13}$$

Conversion to Non-Interactive Protocols: In these zero-knowledge interactive proofs, the interactive steps are mainly used by the verifier to send a *challenge* to the prover, which is a random number chosen from a pre-defined domain, so that the verifier can be convinced that the prover is following the protocol honestly by the fact that the prover can answer the challenge correctly.

In general, the challenge steps can be omitted by allowing both the verifier and the prover to compute the random challenge from a public one-way function [4], such as a pseudo-random number generator or a secure hash function, with a random input that is known by both parties, e.g., some input of the protocol, or the output of some previous steps of the protocol.

6. SECURITY ANALYSIS

Recall that in Section 4, we categorize the security requirements into two categories. First let us examine the security requirements in the first category, which concerns with possible cheating by the owner, Alice.

Suppose Bob sends an image I to Alice for detection. Knowing that I does not contain any of her watermarks, Alice can still try to cheat by finding a watermark W that is detectable in I, through efficient filtering on I. However, our scheme requires that W satisfies the constrain that it must be computed from the master key M, the tuple key K and the Blum secure PRNG, for which it is infeasible to compute the seed given any sequence of the output. Therefore, Alice cannot cheat in this way, and has to guess the correct M, which has a low probability of success. As a result, security requirement (*S1*) can be met. As for the requirement of correctness of claims (*S2*), it is clearly met during the process of the zero-knowledge proof protocols.

For the second category of requirements, the detection is clearly secure (*S3*) under zero-knowledge proofs, and inferences are hard (*S4*) because of the nature of the secure PRNG.

7. CONCLUSIONS

In this paper, we investigate the use of cryptographic primitives (such as secure PRNG and zero-knowledge proofs)

in digital watermarking schemes, and show that these well studied primitives can be employed to meet sophisticated security requirements.

Specifically, we study the problem of generating and managing large number of watermarks, while allowing the owner to prove that the detection is done honestly without leaking other information about the secret watermarks. We show that this can be achieved by incorporating a secure PRNG and zero-knowledge proofs, without involving any trusted third party.

We propose a zero-knowledge watermark detection protocol based on a slightly modified version of a known probably secure non-invertible watermarking scheme, so that the user does not learn anything other than the existence of the watermark, but at the same time can be convinced that the watermark detection was done honestly.

8. REFERENCES

[1] A. Adelsbach, S. Katzenbeiser, and H. Veith. Watermarking schemes provably secure against copy and ambiguity attacks. In *ACM Workshop on Digital Rights Management*, pages 111–119, Washington DC, October 2003.

[2] A. Adelsbach, S. Katzenbeisser, and A. Sadeghi. On the insecurity of non-invertible watermarking schemes for dispute resolving. In *International Workshop on Digital Watermarking*, volume 2939 of *LNCS*, pages 355–369, October 2003.

[3] A. Adelsbach and A. Sadeghi. Zero-knowledge watermark detection and proof of ownership. In *Information Hiding Workshop*, volume 2137 of *LNCS*, pages 273–288, Pittsburgh, PA, April 2001.

[4] M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, Fairfax, Virginia, USA, November 1993.

[5] L. Blum, M. Blum, and M. Shub. A simple secure unpredictable pseudo-random number generator. *SIAM Journal on Computing*, 15:364–383, 1986.

[6] Fabrice Boudot. Efficient proofs that a committed number lies in an interval. In *EUROCRYPT*, volume 1807 of *LNCS*, pages 431–444, 2000.

[7] Jan Camenisch and Markus Michels. Proving in zero-knowledge that a number is the product of two safe primes. In *EUROCRYPT*, volume 1592 of *LNCS*, pages 106–121, Prague, Czech Republic, May 1999.

[8] I.J. Cox, M.L. Miller, and J.A. Bloom. *Digital Watermarking*. Morgan Kaufmann, 2002.

[9] S. Craver. Zero knowledge watermark detection. In *Information Hiding Workshop*, volume 1768 of *LNCS*, pages 101–116, 2000.

[10] S. Craver, N. Memon, B.L. Yeo, and M.M. Yeung. Resolving rightful ownerships with invisible watermarking techniques: Limitations, attacks, and implications. *IEEE JSAC*, 16(4):573–586, 1998.

[11] I. Damgard and E. Fujisaki. A statistically-hiding integer commitment scheme based on groups with hidden order. In *ASIACRYPT*, volume 2501 of *LNCS*, pages 125–142, Queenstown, New Zealand, December 2002.

[12] J.J. Eggers, J.K Su, and B. Girod. Asymmetric watermarking schemes. In *Sicherheit in Mediendaten*, pages 107–123, Berlin, Germany, September 2000.

[13] J.J. Eggers, J.K Su, and B. Girod. Public key watermarking by eigenvectors of linear transforms. In *European Signal Processing Conference*, Tampere, Finland, September 2000.

[14] E. Fujisaki and T. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *Crypto*, volume 839 of *LNCS*, pages 16–30, Santa Barbara, California, USA, August 1994.

[15] T. Furon and P. Duhamel. An asymmetric public detection watermarking technique. In *Information Hiding Workshop*, volume 1768 of *LNCS*, pages 88–100, University of Wollongong, NSW, Australia, December 2000.

[16] Q. Li and E.-C. Chang. Public watermark detection using multiple proxies and secret sharing. In *International Workshop on Digital Watermarking*, volume 2939 of *LNCS*, pages 558–569, Seoul, Korea, October 2003.

[17] Q. Li and E.-C. Chang. On the possibility of non-invertible watermarking schemes. In *International Workshop on Digital Watermarking*, volume 3200 of *LNCS*, pages 13–24, Seoul, Korea, 2004.

[18] N. Memon and P.W. Wong. Buyer-seller watermarking protocol based on amplitude modulation and the ElGamal public key crypto system. In *SPIE Security and Watermarking of Multimedia Contents*, pages 189–294, 1999.

[19] T.P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, *CRYPTO*, volume 576 of *LNCS*, pages 129–140, Santa Barbara, California, USA, August 1992.

[20] L. Qiao and K. Nahrstedt. Watermarking schemes and protocols for protecting rightful ownerships and customer's rights. *Journal of Visual Communication and Image Representation*, 9(3):194–210, 1998.

[21] M. Ramkumar and A. Akansu. Image watermarks and counterfeit attacks: Some problems and solutions. In *Symposium on Content Security and Data Hiding in Digital Media*, pages 102–112, New Jersey Institute of Technology, May 1999.

[22] R.G. van Schyndel, A.Z. Tirkel, and I.D. Svalbe. Key independent watermark detection. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, volume 1, pages 580–585, Florence, Italy, June 1999.

[23] Chia-Mu Yu and Chun-Shien Lu. Robust non-interactive zero-knowledge watermarking scheme against cheating prover. In *ACM Multimedia and Security Workshop*, New York City, NY, USA, 2005.