

Keyless Public Watermarking for Intellectual Property Authentication

Gang Qu

Electrical and Computer Engineering Department, University of Maryland
College Park, MD 20742
gangqu@eng.umd.edu

Abstract. A constraint-based watermarking technique has been introduced for the protection of intellectual properties such as hardware, software, algorithms, and solutions to hard problems. It provides desirable proof of authorship without rendering the intellectual properties useless. However, it makes the watermark detection, which is as important as watermarking, a NP-hard problem. We propose a keyless public watermarking method that enables the watermark to be publicly detectable. The basic idea is to create a cryptographically strong pseudo-random watermark, embed it into the original problem as a special (mutual exclusive) constraint, and make it public. No key is required to detect such watermark. We combine data integrity technique and the unique characteristics in the design of intellectual property such that adversaries can get almost no advantage for forgery from the public watermarking. This new technique is also compatible with the existing constraint-based watermarking techniques to enhance the strength of the watermark. We build the mathematical framework for the this approach based on the concept of mutual exclusive constraints. We use several well-known problems to explain our approach, demonstrate its robustness against attacks, and show that there is little degradation in performance.

1 Introduction

1.1 The Problem

While designing the next generation high performance processor, Alice and her design team find that they have to solve a hard combinatorial problem to place all the components on the given chip. Fortunately, she finds the same problem instance and a list of solutions at the official website of *The Combinatorist Association*. After verifying correctness and quality of these solutions, Alice obtains several solutions that meet her requirements. But she also discovers surprisingly amount of similarities among such solutions. Apparently someone copies other's solution and claims it as his own with little or no modification. To avoid possible legal problems, Alice has to identify the original author of the solution.

1.2 Constraint-Based Watermarking: Current Solution

The best two solutions are claimed by *Bob* and *Don* despite the fact that they look almost identical. Knowing that nowadays people embeds their digital signatures when they create their intellectual properties, Alice asks *Bob* and *Don* to reveal their secret watermark from the solution.

Bob gives Alice a set of properties that his solution possesses but are not necessary for all valid solutions to the original problem instance. He further explains the constraint-based watermarking technique that he applied to enforce the existence of these special properties. In short, *Bob* hashed his company's address with MD5, a one-way hash function; then encoded the hash result into extra constraints to the problem instance using a simple encoding scheme; he finally solved the new problem instance and obtained the current solution.

Don is able to give similar explanation, however, his encoding scheme is complicated, questionable, and anecdotal. Since it is well known that watermarking approach can be challenged by an argument based on the complexity of the watermarking process and it is important to keep the watermarking process straightforward, Alice can conclude that *Bob* is most likely the author.

1.3 Publicly Detectable Watermarking: New Approach

What Alice needs is actually an easy detection mechanism that allows people without much expertise on combinatorics to accumulate sufficient information from a solution for its authorship.

In response, the *Combinatorist Association* publishes its first standard for publicly detectable watermarking: for every problem instance, *Combinatorist Association* identifies a set of additional constraints to force certain optional properties; any registered identity who wants to post his solution to this problem will pick one such constraint based on his information that is made public on registration, and his solution will automatically satisfy a unique subset of these optional properties; users of the solution can verify these properties, recognize the unique pattern, and be able to tell the source of the solution with a relatively high confidence. Notice that in this detection process, a detector does not need the author's key, or any knowledge on how to solve the problem.

1.4 Graph Partitioning: An Example

Given a graph $G = (V, E)$ on a set of vertices V and a set of edges E , the graph partitioning problem is to partition V into disjoint nonempty subsets. For example, the dashed line in Figure 1(a) partitions the 24-vertex graph into two subsets, the one on its left contains 11 vertices and the other subset has 13. The dashed line cuts through 6 edges. (Suppose that we want to cut as few number of edges as possible, and keep the difference of the two subsets' cardinality within two.).

We now show the basic idea of the keyless public watermarking. We identify eight pairs of vertices and make them public along with the original graph as

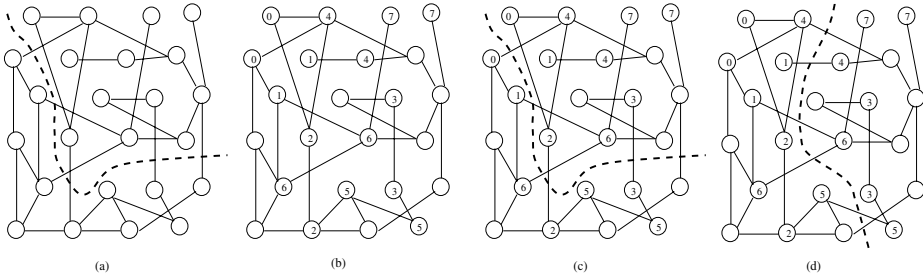


Fig. 1. (a) The original graph partitioning instance; (b) the same graph with 8 marked pairs that enables an 8-bit keyless public watermark; (c) *Bob*'s solution with his public information “01001111”; and (d) *Don*'s solution with his public information “01110000”

in Figure 1(b). We challenge each registered party who has an 8-bit public information $p_7p_6 \cdots p_1p_0$ the following problem: partition graph in Figure 1(b) into two subsets, such that the i th pair are in the same subset if $p_i = 0$ and in different subsets otherwise. The last requirement converts the public information into additional constraints and we call it public watermark embedding scheme.

Suppose Alice gets *Bob*'s solution in Figure 1(c) and *Don*'s solution in Figure 1(d). She wants to verify their authorships from the following public information:

- the graph partitioning instance, with the 8 pairs of vertices for embedding public watermarks, in Figure 1(b);
- the public watermark embedding scheme being used;
- *Bob*'s solution in Figure 1(c) with his public information “01001111”, letter ‘O’ in ASCII code;
- *Don*'s solution in Figure 1(d) with his public information “01110000”, letter ‘p’ in ASCII code.

All Alice needs to do is a simple check of the eight pairs of vertices. In Figure 1(c), the two vertices in pairs 0, 1, 2, 3, and 6 are separated, which implies 1's at the corresponding bit positions from the known public watermark embedding scheme. This observation reveals an 8-bit message “01001111”, which coincides with *Bob*'s public information and provides a proof to *Bob*'s authorship. Similarly, *Don*'s authorship could also be established from Figure 1(d).

1.5 Paper Organization

Our work is motivated by the proliferation of intellectual property (IP) and the potential of it being illegally redistributed and misused. We will discuss, in the next section, why IP authentication cannot be achieved by the well-studied digital watermarking techniques for digital content. In Section 3, we review the constraint-based watermarking technique and explain the reason that it fails

to provide easy detection schemes. We introduce the generic keyless public watermarking technique in Section 4 and discuss how such watermark is created, embedded, and detected. We demonstrate its robustness and show its impact to IP's quality by simulations on two well-known problems: the Boolean satisfiability and graph vertex coloring. We conclude by summarizing the approach and its advantages over previous techniques.

2 Motivation and Previous Work

2.1 Intellectual Property Protection in System Design

The advances in VLSI semiconductor technology and system-on-a-chip design paradigm, coupled with the shrinking time-to-market window, have changed the traditional system design methodology. Design reuse and intellectual property (also known as virtual component) based design become more and more important. Design challenge nowadays is to find IPs and make necessary modification, as little as possible, to meet customer's requirements in a timely fashion.

The Virtual Socket Interface Alliance (VSIA), an international organization that includes representatives from system houses, semiconductor vendors, electronic design automation companies, and IP providers, specifies open standards to facilitate the mix and match of virtual components from multiple sources in order to accelerate system-chip development. According to VSIA, virtual component (VC) is a block that meets the virtual socket interface specification and is used as a component in the design environment[8].

Virtual components trading plays a central role in the design-for-reuse methodology and the potential of infringement is growing fast. In the area of electronic design, there are an estimated 100 reverse engineering shops in the US; approximately 70% funded by government, and many of the techniques developed are leaked, or even published, to the industry. The American Society for Industrial Secrets estimates that in the US alone, trade secret theft is in excess of \$2 billion per month[8]. However, the global awareness of IP protection remains low. The newly released IP protection white paper by VSIA provides guidelines for IP providers to protect their IPs against unauthorized use, to detect and to trace the use of their IPs.

2.2 Information Hiding in Digital Contents

To provide evidence of authorship and to trace the usage of an object, hiding data (information, message, signature, or watermark) into the content is the method that has been used for thousands of years and is still one of the most powerful and popular techniques today. The proliferation of digitized data (text, image, audio, video, and multimedia) is pushing hard for copyright enforcement to protect authorship. Figure 2 depicts the basic steganography system from the First International Information Hiding Workshop [6].

The author of the cover-data wishes to hide his digitized signature (embedded-data) into the original cover-data. He embeds his signature using

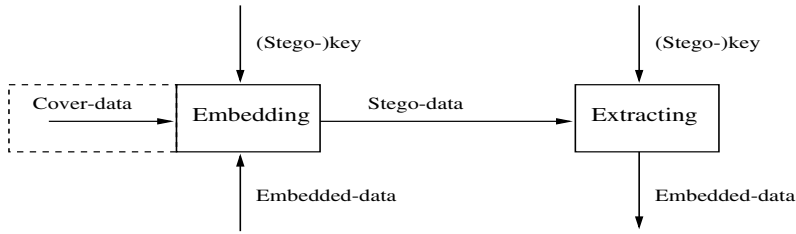


Fig. 2. The basic steganography system (redrawn from [6])

his secret key and obtains the stego-data. This process is referred as embedding. For the purpose of authentication, he uses the same key (or a related one) to extract the embedded-data from the stego-data.

Notice that the key is required to detect the hidden information and this prevents the embedded-data to be extracted publicly. On one hand, it provides security and robustness. On the other hand, it makes authentication hard and (almost) impossible without the key. In the rest of this section, we survey the previous efforts on relaxing this assumption. Further discussion on the general embedding process and its application can be found in the proceedings of the previous three information hiding workshop as well as in a special issue of *Proceedings of the IEEE (Vol. 87, No. 7, July 1999)*.

Most of the reported literatures on detection focus on how to extract and recover the embedded-data with the secret key from the stego-data, even after it has been attacked. There are only two existing approaches to make watermark publicly detectable. One is based on the so-called public-key watermarking, the other relies on zero-knowledge protocols.

Hartung and Girod [3] present an extension of spread-spectrum watermarking that enables public decoding and verification of the watermark. However, an attacker can also discover the original watermark from the author's public key and remove it easily. Although the author can still retrieve the private part of the watermark by his private key, the property of public authentication is no longer there. To make it even worse, the attacker may further embed his own public watermark and claim his authorship.

Craver [2] uses zero-knowledge protocols to make the watermarks public enough to be detected yet private enough not to be removable. In such schemes, interaction between the detector and the author is required and the detector will challenge the author similar problems, possibly many times, to establish a proof of the authorship. More discussion on proving authorship of digital content can be found in [1] where a general model for proof of ownership is proposed.

2.3 Challenges for IP Authentication

IP authentication is different from extracting the embedded watermark from digital contents. For such contents, signatures are encrypted using the secret

key and embedded into the cover-data as minute errors in such a way that they are perceptually invisible (*unobtrusive*) and difficult to remove (*robust*). The transparency of the signature relies on human's insensitiveness to such subtle changes. Here the intellectual property refers to objects like software, hardware, algorithms, or solutions to hard problems, which perform certain tasks or satisfy given constraints. The IP's value depends on the fact that it performs the correct task or finds the right answer. When the author of the IP attempts to embed his signature into the IP, he has to maintain this correct functionality. Otherwise the IP may malfunction and become useless. Embedding is the first challenge for IP authentication.

According to the basic steganography system in Figure 2, in IP authentication, the IP itself is the cover-data in which we want to hide the embedded-data. One may still be able to find ways to put watermark into the original IP and create a stego-IP. For example, changing variable names in software, making small variations in algorithms, and exploiting locality to modify solutions to a problem. However, it is challenging to make the watermark unobtrusive and robust.

Finally the extracting process is difficult. Suppose that we successfully create a stego-IP which contains invisible and robust watermark. The extracting process, which normally is the reverse of embedding process, is as hard as embedding. This makes public IP authentication almost impossible.

3 The Constraint-Based IP Watermarking

A conceptually new method [4], *constraint-based watermarking technique*, translates the to-be-embedded signature into a set of additional constraints during the design and implementation of IP in order to uniquely encode the signature into the IP. IP authentication is conducted by showing the existence of these additional constraints and the small probability for a random solution to satisfy all these constraints. The effectiveness of this generic scheme has been demonstrated at all stages of hardware design process [4,5]. Qu and Potkonjak, in last year's information hiding workshop, present a theoretical framework for analyzing such watermarking techniques. They build the necessary mathematical foundation by proving that strong proof of authorship is achievable without significant degradation of solution's quality [7].

In contrast to the steganography system in Figure 2, we illustrate in Figure 3 the basic idea of the constraint-based watermarking technique and show its key concept of cutting solution space. We view the design and implementation of IP as solving a hard problem where the requirements for the IP serve as constraints to the problem. The problem solving process corresponds to the invention of IP which is a solution to the problem.

To hide his signature, the author first creates another set of constraints using his secret key; he then adds these additional constraints into the original problem and produces a more constrained stego-problem; he now solves the stego-problem, instead of the original problem, and obtains a stego-solution that con-

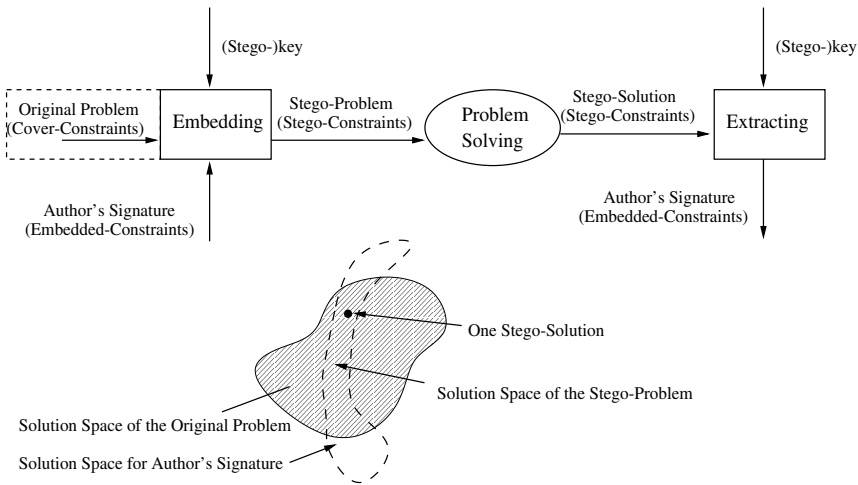


Fig. 3. The constraint-based watermarking technique

tains his signature. The stego-solution satisfies both the original and additional constraints. Using information hiding terminologies, we refer the original problem as *cover-constraints*, the signature as *embedded-constraints*, they combine to output the *stego-constraints* which the *stego-solution* must meet.

For authentication, the author has to demonstrate that the stego-solution carries the hidden information which is based on his signature. This can be done by verifying the satisfiability of the embedded-constraints. The secret key is necessary for converting such embedded-constraints back to the author's signature.

The essence of the constraint-based watermarking technique is cutting the solution space. In the bottom of Figure 3, we show the solution spaces for the cover-constraints (the shaded area) and the embedded-constraints (the area enclosed by the dashed curve). Their intersection are (stego-)solutions that satisfy both constraints. To claim the authorship of a stego-solution, the author will argue that he has much better chance to find this solution from the relatively small solution space for the stego-problem rather than the large solution space for the original problem. Previous results show that this method can provide a very strong proof of the authorship[7].

4 Keyless Public Watermarking

The addition of watermark makes it possible for IP authentication. A detection scheme normally requires either complete knowledge of the IP (in particular, the secret key) or expertise on forensic engineering (a technique that is used to identify solutions generated by strategically different algorithms). However, an IP buyer usually does not possess such knowledge or expertise. Therefore, IP authentication can only be achieved by IP author, not by IP buyers. This

greatly limits the watermarking’s primitive purpose of deterring misuse and illegal redistribution. Because attackers will be encouraged if they know that the buyers are unable to verify the authorship of the IP. In this section, we present the keyless public watermarking to solve this problem.

4.1 General Approach

Figure 4 illustrates the generic keyless public watermarking technique. We start by finding places in the original problem, which we call *public watermark holder*, to accommodate the public watermark. We then make the original problem public with the public watermark holder as cover-constraints. Author’s public signature will be embedded behind the public watermark holder to form a stego-problem. Solving this problem gives us a stego-solution that satisfies the stego-constraints. The keyless public authentication is done in the extracting box. One checks the satisfiability of the cover-constraints in the public watermark holder, and based on which constraint is satisfied, he can determine the author’s public signature from the public watermark embedding scheme.

Notice the two major differences between this approach and the conventional constraint-based watermarking described in the previous section:

- There is a step for identifying the public watermark holder. The public watermarking will be hidden only at such places, instead of being spread out all over the original problem. Thus one will know where to look for the watermark while doing authentication.
- There is no secret (stego-)key involved in the watermark embedding process. So everyone, including IP buyers, will be able to extract the author’s public signature from the public watermark holders.

Due to these reasons, the new approach is referred as *keyless public watermarking*. In the rest of this section, we explain in detail the three phases: finding the public watermark holder; watermark embedding; and watermark detection.

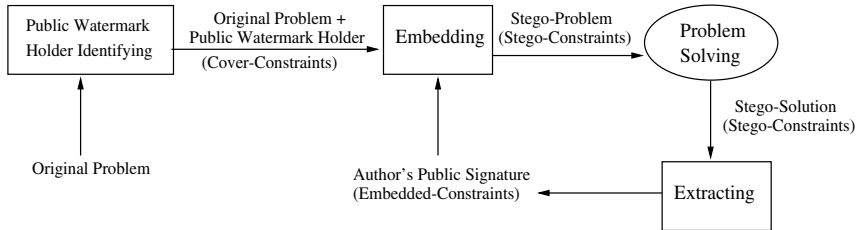


Fig. 4. The keyless public watermarking technique

4.2 Mutual Exclusive Constraints: The Public Watermark Holder

We embed the public watermark by adding a special type of constraints: mutual exclusive constraints. We introduce the necessary definitions and explain them by the example of graph partitioning problem we discussed in the first section, where we want to partition a graph with k vertices, $\{v_1, v_2, \dots, v_k\}$ into four partitions, $\mathcal{A}, \mathcal{B}, \mathcal{C}$ and \mathcal{D} .

Definition 1 (mutual exclusive): Given a problem \mathcal{P} , a set of $n \geq 2$ constraints $\{C_1, C_2, \dots, C_n\}$ are *mutual exclusive* if any solution \mathcal{S} satisfies at most one constraint C_i , ($1 \leq i \leq n$).

For vertex v_1 , the following four constraints are mutual exclusive:

- $C_1 : v_1$ must be in partition \mathcal{A} ;
- $C_2 : v_1$ must be in partition \mathcal{B} ;
- $C_3 : v_1$ must be in partition \mathcal{C} ;
- $C_4 : v_1$ must be in partition \mathcal{D} .

However, adding another constraint $\{C'_1 : v_2 \text{ must be in partition } \mathcal{A}\}$ makes this set not mutual exclusive because a solution which places both v_1 and v_2 in \mathcal{A} will satisfy both C_1 and C'_1 .

Definition 2 (complete mutual exclusive set): A mutual exclusive set of constraint $\{C_1, C_2, \dots, C_n\}$ is *complete* if any solution \mathcal{S} satisfies exactly one constraint.

The set $\{C_1, C_2, C_3\}$ is mutual exclusive, but not complete because solution that has vertex v_1 in partition \mathcal{D} does not satisfy any of these three constraints. Adding constraint C_4 makes it complete.

Definition 3 (strongly mutual exclusive set): A mutual exclusive set is *strongly mutual exclusive* if for any constraint C_i , there exists a solution \mathcal{S} that satisfies C_i and violates C_j ($j \neq i$).

The set $\{C_1, C_2, C_3, C_4\}$ is strongly mutual exclusive since we can always find a solution by putting v_1 in one of the partitions and then partition the other vertices. However, it is clear that any set that contains the constraint $\{v_1 \text{ must be in both } \mathcal{A} \text{ and } \mathcal{B}\}$ can not be strongly mutual exclusive.

Definition 4 (independent constraints): Two constraints are *independent* if any solution's satisfiability to one constraint has no impact on its satisfiability to the other one. Two sets of constraints, $\{C_1, C_2, \dots, C_n\}$ and $\{C'_1, C'_2, \dots, C'_m\}$, are *independent* if C_i and C'_j are independent for any $1 \leq i \leq n$, and $1 \leq j \leq m$.

Obviously, constraints $\{C_1 : v_1 \text{ must be in partition } \mathcal{A}\}$ and $\{C'_1 : v_2 \text{ must be in partition } \mathcal{A}\}$ are independent if there is no further constraints to enforce v_1 and v_2 to be in/out of the same partition.

Definition 5 (join): The *join* of two sets of constraints, $\{C_1, \dots, C_n\}$ and $\{C'_1, \dots, C'_m\}$, is defined as the set $\{C_1 \wedge C'_1, \dots, C_1 \wedge C'_m, C_2 \wedge C'_1, \dots, C_n \wedge C'_m\}$, where constraint $C_i \wedge C'_j$ is satisfied if and only if both constraints C_i and C'_j are satisfied.

Of our particular interest is the set of complete strongly mutual exclusive constraints. We list the following properties without proof due to space limit:

Theorem 1 (Cutting Space Theorem): A set of complete strongly mutual exclusive constraints partitions the solution space as the union of nonempty disjoint subsets.

Theorem 2 (Existence Theorem): Complete strongly mutual exclusive set exists for all problems with more than two different solutions.

Lemma 1 (Subset Lemma): A subset of a mutual exclusive set is mutual exclusive; a subset of a strongly mutual exclusive set is strongly mutual exclusive; a subset of a complete mutual exclusive set is not complete.

Lemma 2 (Join Lemma): The join of two disjoint complete mutual exclusive sets is complete. Furthermore, if they have n_1 and n_2 constraints respectively, the join will have $n_1 \cdot n_2$ mutual exclusive constraints.

Theorem 3 (Join Theorem): If two complete mutual exclusive sets have n_1 and n_2 constraints respectively and they have l constraints in common, then their join is complete and has $(n_1 - l) \cdot (n_2 - l) + l$ constraints.

Lemma 3 (Data Hiding Lemma): n different information (of any length) can be hidden with a (complete) strongly mutual exclusive set of n constraints.

4.3 Public Watermark Embedding

We now explain the embedding box in Figure 4 that creates the stego-problem with author’s public signature embedded. It consists of three phases: constructing mutual exclusive set of constraints, creating public watermarks, and defining public watermark embedding schemes.

Construct Mutual Exclusive Set of Constraints For a given problem, we first construct a set of (complete) strongly mutual exclusive constraints. From the Data Hiding Lemma, this set must have sufficient number of constraints to accommodate different public watermark. The Existence Theorem guarantees that there always exist complete strongly mutual exclusive sets. The Join Lemma and Join Theorem give a constructive method of building large set from small sets.

Example:

Suppose we want to partition a graph with n vertices, $\{v_1, \dots, v_n\}$, into two partitions. We select $2k$ distinct vertices (for example randomly): $v_{i_1}, v_{i'_1}; v_{i_2}, v_{i'_2}; \dots; v_{i_k}, v_{i'_k}$. Define k sets of constraints: $\{C_{i_1}, C'_{i_1}\}, \dots, \{C_{i_k}, C'_{i_k}\}$ as

C_{i_j} : vertices v_{i_j} and $v_{i'_j}$ are in the same partition.

C'_{i_j} : vertices v_{i_j} and $v_{i'_j}$ are in different partitions.

It is easy to verify that every set is complete strongly mutual exclusive and

these k disjoint sets are independent. The join of these k sets gives us a complete (actually strongly in this case) mutual exclusive set with 2^k constraints:

$$\{\bar{C}_{i_1} \wedge \bar{C}_{i_2} \wedge \dots \wedge \bar{C}_{i_k} : \bar{C}_{i_j} = C_{i_j} \text{ or } C'_{i_j}\} \quad (*)$$

where the join constraint $\bar{C}_{i_1} \wedge \dots \wedge \bar{C}_{i_k}$ is satisfied if and only if all \bar{C}_{i_j} 's are satisfied. For example, $C_{i_1} \wedge C_{i_2} \wedge \dots \wedge C_{i_k}$ is the constraint that requires vertices v_{i_j} and $v_{i'_j}$ be in the same partition for all $1 \leq j \leq k$.

Create Public Watermarks Figure 5 shows step-by-step how to create the keyless public watermark from author's public signature. The public watermark is a bitstream with a header and a body. The header is just the author's plain text public signature (with a fixed length) in ASCII code. This ASCII code is hashed by a one-way hash function (e.g. MD5); the hash is put into a stream cipher (e.g. RC4) with the ASCII code as key and the produced pseudo random bitstream makes the body of the public watermark. The simplicity of watermark header facilitates public authentication and the pseudo-random watermark body provides robustness against attacks which we will discuss in the section of authentication.

Define Embedding Schemes Now we have a set of mutual exclusive constraints and a set of public watermarks. A *watermark embedding scheme* is a one-to-one function from the set of watermarks to the set of constraints such that different public watermarks are mapped to different constraints. We intend to keep the embedding scheme as simple as possible for the purpose of public authentication.

As a continuation of the previous graph partitioning example, we can define the watermark embedding scheme as follows:

for public watermark $p_1p_2 \dots p_k$, we choose the constraint $\bar{C}_{i_1} \wedge \dots \wedge \bar{C}_{i_k}$ from the complete strongly mutual exclusive set () we constructed earlier, where $\bar{C}_{i_j} = C_{i_j}$ if $p_j = 0$ and $\bar{C}_{i_j} = C'_{i_j}$ if $p_j = 1$.*

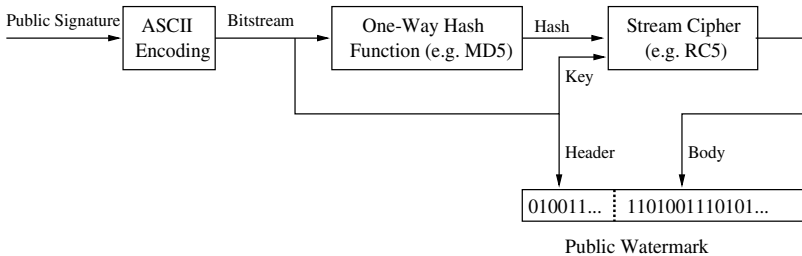


Fig. 5. The creation of keyless public watermark from public signature

The stego-problem is obtained by adding *THE* constraint that corresponds to the public watermark under the embedding scheme. Different watermarks are mapped to different constraints from a strongly mutual exclusive set. Therefore all stego-problems will be different and the property of mutual exclusiveness guarantees their solutions will be distinct. In sum, we have

Theorem 4 (Correctness of the Approach):

If the constraints are strongly mutual exclusive, there always exist (stego-) solutions for the stego-problem. Furthermore, different stego-problems will have different (stego-)solutions which are all solutions to the original problem.

4.4 Public Watermark Authentication

In this part, we explain the extracting box in Figure 4 whose function is to detect the public watermark from a given stego-solution and retrieve the author's public signature. The followings are available to the public: **(i)** the original problem; **(ii)** the set of mutual exclusive constraints which is the public watermark holder; **(iii)** the public watermark embedding scheme; **(iv)** the fixed length of all author's public signature; and **(v)** a stego-solution needs for authentication.

A detector checks which constraint from the set of mutual exclusive constraints **(ii)** does the given stego-solution **(v)** satisfy. Then he obtains the embedded public watermark from the known embedding scheme **(iii)**. He now takes the watermark header of fixed length **(iv)**, this gives the author's public watermark in ASCII format and suggests the possible author. The detector may further hash this watermark header and use the stream cipher to re-produce the watermark body. A strong proof to the authorship is established if the re-produce the watermark body coincides with the one extracted from the stego-solution.

4.5 Remarks

We make the following remarks on more features about the public watermark:

Robustness: the stego-solution is obtained by solving the stego-problem which contains a unique public watermark. A successful forgery is a different solution obtained by modifying the given solution and has the attacker's public watermark embedded. A different solution may not be difficult to get. However, it is hard in general to hide another information unless the attacker is able to solve the problem by himself in which case he has little incentive for forgery.

Role of the public watermark header: the key that enables the keyless public watermarking authentication. It is important to keep it in plain text.

Role of the public watermark body: for many problem, one may be able to find a new solution from a given solution by study the locality of the problem. The public watermark body provides the public watermark integrity and makes it hard to forge a watermark (theoretically, even one bit change in watermark header results in half of the bits being flipped in the watermark body).

The use of join: provides an efficient way to produce large set of mutual exclusive constraints. It also enables a logarithmic time IP authentication instead of linear.

Impact to the quality of the solution: similar to the conventional constraint-based watermarking techniques, adding extra constraints may introduce degradation of the solution's quality. One of the criteria for building mutual exclusive constraints is to keep this overhead at the minimum level.

Public-private watermarking technique: the public watermarking technique is compatible with all the existing watermarking techniques. This suggests a public-private watermarking approach where authors can embed more information based on their secret keys after the public watermarks are enforced. It is used to enhance the watermark's credibility.

5 Validation and Experimentation

We report the implementation of the public watermarking technique and the experimental results on two combinatorial problems to show the watermark's robustness and its impact to solution's quality.

5.1 Boolean Satisfiability

The Boolean satisfiability problem (SAT) seeks to decide, for a given formula, whether there is a truth assignment for its variables that makes the formula true.

Given a formula \mathcal{F} on a set of boolean variables V , we select a subset of variables $\{v_1, v_2, \dots, v_k\}$. We embed a public watermark $m_k \cdots m_2 m_1$ by forcing $v_i = m_i$ in the solution. This can be done by adding to the formula \mathcal{F} single-literal clause v_i (if $m_i = 1$) or v'_i (if $m_i = 0$).

We create watermarks with 32-bit headers and 128-bit (64-bit for small instances) bodies and embed them into DIMACS SAT benchmarks ([http:// dimacs.rutgers.edu/](http://dimacs.rutgers.edu/)). We solve the stego-problems to get a solution from which an attacker tries to make a successful forgery. Our goal is to show that there will be no correlation between the original solution and adversary's new solution (the forgery). Consequently, the attacker has no advantage from the original solution to create the forgery.

We consider the attacker retrieves the original public watermark and changes randomly 4 bits, 8 bits, 16 bits, and 24 bits in the 32-bit watermark header. He then computes his own watermark body, and tries to modify the solution to create a forgery. We solve the formula with the faked watermark and calculate the Hamming distance between the new solution and the original solution (i.e., the number of variables that receive different values).

Table 1 shows our experimental results on repeating each trial 10 times. The second column gives the number of variables N in the problem instances; columns

Table 1. Robustness demonstrated by watermarking SAT benchmarks

		4 bits in header		8 bits in header		16 bits in header		24 bits in header	
\mathcal{F}	N	body	sol.	body	sol.	body	sol.	body	sol.
ii8b1	336	31.2	148	32.8	150	31.8	168	32.6	170
ii8b2	576	33.6	260	30.6	258	32.4	265	32.0	272
ii8b3	816	62.2	363	64.0	376	67.4	358	61.6	387
ii8b4	1068	65.8	489	66.2	472	63.4	492	62.6	513
Ave. Dist. (%)		40.2%	-	43.5%	-	50.5%	-	56.2%	-
Ave. Dist. (%)		-	44.9%	-	44.9%	-	46.5%	-	48.3%

labelled “body” show the average number of bits changed in the forged watermark comparing to the original watermark; The average Hamming distances (rounded to the nearest integer) are reported in columns with label “sol.”.

The last two rows report these average distances percentage-wise. The first is the distance in the public domain, which will be very close to 50% if we exclude the watermark header. This shows the robustness of cryptographic tools (MD5 and RC4) in generating pseudo-random bit streams. The last row shows that the new solutions are not close to the original solution. (Considering the fact that when we solve the original instances for multiple solution, the average distance is also about 45%). This means the attacker has little advantage from the known solution.

5.2 Graph Coloring

The NP-hard graph vertex coloring optimization seeks to color a graph with as few colors as possible, such that no two adjacent vertices receive the same color. We propose the following public watermarking technique and use it to demonstrate our approach’s impact to the quality of the solution:

Given a graph, we select pairs of vertices that are not connected directly by an edge. We hide one bit of information behind each pair as follows: adding one edge between the two vertices and thus making them to have different colors to embed 1; collapsing the two vertices and thus forcing them to receive the same color to embed 0.

To evaluate the trade-off between protection and solution degradation, we first color the original graph, then color the watermarked graph and compare the average number of colors required. We consider two classes of real life graphs (the *fpsol2* and *inithx* instances from <http://mat.gsia.cmu.edu/COLOR/instances.html>) and the DIMACS on-line challenge graph.

Table 2 shows the number of vertices in each graph (*vert.* column), the optimal solutions (*opt.* column), and the overhead introduced by public watermark messages of various length. The graph DSJC1000 is still open and the number in the *opt.* column is the average of 10 trials with 85-color solutions occur several times. For each instance, we create ten 32-bit and ten 64-bit public watermark

Table 2. Embedding public watermark to real life graph and randomized graph

original instance			32-bit watermark		64-bit watermark	
	vert.	opt.	overhead	best	overhead	best
fpsol2.i.1	496	65	0.2	65	0.7	65
fpsol2.i.2	451	30	0.1	30	0.5	30
fpsol2.i.3	425	30	0.1	30	0.5	30
inithx.i.1	864	54	0.0	54	0.2	54
inithx.i.2	645	31	0.9	31	1.8	32
inithx.i.3	621	31	1.1	31	1.9	32
DSJC1000	1000	85.8	0.5	86	2.0	87

messages randomly. We add the message to the graph and color the modified graph. The average number of colors and the best solution we find are reported. One can easily see that the proposed approach causes little overhead for real life instances, but fails to find any best solution (85-color) for the randomized DSJC1000 graph. The reason is that there exist localities in real life graphs of which we can take advantage. However, such localities do not exist or are very difficult to find in random graphs.

6 Conclusion

We propose the first intellectual property watermarking technique that facilitates easy and public detection. We achieve this by allowing part of the watermark to be public. We use cryptographic techniques, in particular techniques for data integrity, to protect the public watermark from forgery. We build a sound theoretical framework for this approach based on the concept of mutual exclusive constraints. We explain the basic approach and develop specific techniques for several well-known combinatorial problems. The new approach takes advantage of the fact that the embedded watermark (in the format of additional constraints) cannot be modified randomly while maintaining the correct functionality. With the help from organizations pushing for design standards, this method has the potential of solving eventually the intellectual property authentication problem.

References

1. A. Adelsbach, B. Pfitzmann, and A. Sadeghi. "Proving Ownership of Digital Content". *The 3rd International Information Hiding Workshop*, pp. 126-141, September 1999. 100
2. S. Craver. "Zero Knowledge Watermark Detection". *The 3rd International Information Hiding Workshop*, pp. 102-115, September 1999. 100
3. F. Hartung and B. Girod. "Fast Public-Key Watermarking of Compressed Video". *IEEE International Conference on Image Processing*, pp. 528-531, October 1997. 100

4. A. B. Kahng, et al. "Watermarking Techniques for Intellectual Property Protection". *35th Design Automation Conference Proceedings*, pp. 776-781, 1998. 101
5. J. Lach, W. H. Mangione-Smith, and M. Potkonjak. "Fingerprinting Digital Circuits on Programmable Hardware". *The 2nd International Information Hiding Workshop*, pp. 16-31, April 1998. 101
6. B. Pfitzmann. "Information Hiding Terminology", *The 1st International Information Hiding Workshop*, pp. 347-350, May 1996. 99, 100
7. G. Qu and M. Potkonjak. "Hiding Signatures in Graph Coloring Solutions". *The 3rd International Information Hiding Workshop*, pp. 391-408, September 1999. 101, 102
8. Virtual Socket Interface Alliance. "Intellectual Property Protection White Paper: Schemes, Alternatives and Discussion Version 1.0", September 2000. 99