2009 Third International Conference on Genetic and Evolutionary Computing

A Software Watermarking Method Based on Public-Key Cryptography and Graph Coloring

Zetao Jiang Rubing Zhong Bina Zheng
School of Computer
Nanchang Hangkong University
Nanchang, P.R.China
zetaojiang@yahoo.com.cn pmz0215@yahoo.com.cn

***Abstract*—A new software watermarking method based on public-key cryptography and graph coloring problem (GC) is proposed in this paper. The method with good stealth and security can be applied to protect and verify software copyright. Firstly, the copyright information of ownership should be encrypted for software by RSA public-key cryptography algorithm to ensure that the information is confidentiality, integrity and certainty and prevent the information from tampering and forging. Secondly, the message encrypted is embedded into software by the watermarking technique for the GC problem which has the advantage of being very stealthy due to the fact that it doesn't add any code. Experiments show that a high level of stealth and security can be achieved with at most 1-color-overhead for proposed watermarking method.**

Keywords-Software Watermark; Graph Coloring; RSA Cryptosystem; Interference Graph

I. INTRODUCTION

With the rapid development of software industries, the protection of intellectual property of software from piracy takes on greater importance both in computer business and academia. Software watermarking [1, 2, 3] is an approach that embeds a message into software to claim its ownership. It is an effective mechanism to protect the intellectual property of software developers.

Data rate, stealth, and resilience are the main criteria of software watermarking. The data rate expresses the quantity of hidden data that can be embedded within the cover message. The stealth expresses how imperceptible the embedded data is to an observer, and the resilience expresses the hidden message's degree of immunity to attack by an adversary. At present, some outstanding software watermarking algorithms appear in international community as follows: 1) Moskowitz [4] describes a data watermarking method in which the watermark is embedded in an image (or other digital media such as audio or video) using one of the many media watermarking algorithm. This image is then stored in the static data section of the program. Unfortunately, once the behavior of generating and implementing code without foundation is unusual, it is easy to arouse the attention of attackers, which can lead to extract or destroy the watermark, so the stealth of the method is not ideal. 2) Nagra and Thomborson [5] proposed a threading software watermarking algorithm based on the intrinsic randomness for a thread to run in a multithreaded program in 2004. It is characterized as better stealth and anti-attack capability, but the quantity of hidden data is less.

In this paper, a novel software watermarking method based on public-key cryptography and graph coloring is implemented, which makes use of public-key cryptography and software watermarking technique in protecting and verifying software copyright. The advantages of them are: 1) the watermarking algorithm for GC problem [6, 7] has the advantage of being very stealthy due to the fact that it doesn't add any code. The only changes that can be seen between the original version and the watermarked version are the local variable numbers. Therefore, this algorithm has good robustness against Additive Attacking and Distortive Attacking. And a large amount of information can be embedded into the graph with at most 1-color-overhead. 2) The RSA algorithm has good security, whose strength lies in the tremendous difficulty in factorization of large numbers, so it is difficult for an adversary to decrypt the message encrypted to receive the copyright information of ownership, even if he extracted the message embedded.

This paper is structured as follows. After the introduction is Section Ⅱ and Section Ⅲ which detail necessary background describing Graph coloring problem and RSA Cryptosystem. Section Ⅳ describes a new software watermarking method based on public-key cryptography and graph coloring problem. We report the technique analysis and experimental results in Section Ⅴ and summarize our work in Section Ⅵ.

II. GRAPH COLORING REGISTER ALLOCATION

An important function of any register allocator [8] is to target registers so as to eliminate copy instruction. Graph coloring register allocation is an elegant approach to this problem. If the source and destination of a move instruction don't interfere, then their nodes can be coalesced in the interference graph. Nodes in this graph represent values or temporaries. An edge between two nodes exists if and only if they are simultaneously live at some point in the code. Thus, an edge between two nodes implies that they can't occupy the same register. The graph coloring problem can be stated as follows: given a program's interference graph G and a positive integer K, assign a color to each vertex of G, using at most K coloring, such that no two adjacent vertices receive the same color.

978-0-7695-3899-0/09 $29.00 © 2009 IEEE
DOI 10.1109/WGEC.2009.76

An example program is shown in Instance 1 and its interference graph in Fig. 1. The nodes are labeled with the temporaries they represent, and there is an edge between two nodes if they are simultaneously live. For example, nodes j, a and b are all connected since they are live simultaneously at the end of the block. Assuming that there are four registers available on the machine, then the simplify phase can start with the nodes c, d, i and e in its working set, since they have less than four neighbors each. A color can always be found for them if the remaining graph can be successfully colored. If the algorithm starts by removing d and c, and all their edges, then node a becomes a candidate for removal and can be added to the work list. A possible order in which nodes are removed is represented by the stack shown in Fig. 2(a), where the stack grows upwards. The nodes are now popped off the stack and the original graph reconstructed and colored simultaneously. Starting with g, a color is chosen arbitrarily since the graph at this point consists of a singleton node. The next node to be put into the graph is i. The only constraint is that it be given a color different from g, since there is an edge from g to i. When the original graph has been fully reconstructed, a possible assignment for the colors is shown in Fig. 2(b).

```
Input: a b
        c=mem[b];
        d=a-1;
        e=c*d;
        f=mem[b+4];
        g=mem[b+8];
        h=mem[e];
        i=f+8;
        j=i;
       a=g+8;
       b=h;
Output: j a b
```

Instance1. example program

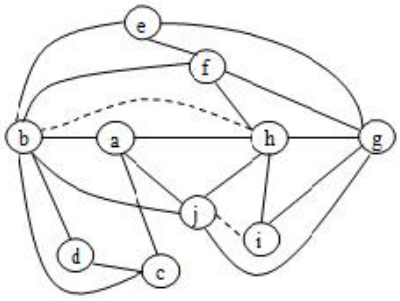

Figure 1. interference graph

Dotted lines are not interference edges but indicate move instructions.

Stack	Assignment
g	2
i	1
h	4
e	3
f	1
b	4
j	1
a	2
d	2
c	1
(a)	(b)

Figure 2. (a) shows the stack after all nodes have been removed; and (b) is a possible assignment of colors.

III. RSA CRYPTOGRAPHY

The RSA public-key cryptosystem [9, 10] was invented by Rivest, Shamir, and Adleman. Since then, the RSA system has been the best known and most widely accepted public key cryptosystem. Usually, the RSA system is deployed in application systems for providing privacy and/or ensuring authenticity of digital data. Its strength lies in the tremendous difficulty in factorization of large numbers. However, the factorization of large numbers is a famous difficult problem in the mathematics and we have no approach to solve it to this day. The security of RSA algorithm can be guaranteed.

The process of RSA algorithm as follows:

Step 1: Find two primes p and q randomly, a few decimal digits apart.

Step 2: Compute n=p*q.

Step 3: Now choose a random integer e, ($0<e<\Phi(n)$), such that $e*d==1 \bmod \Phi(n)$ and $\gcd(e, \Phi(n))=1$. We find $d=e^{-1} \bmod \Phi(n)$ using the extended Euclidean algorithm. Since the inverse is unique, i.e. $\gcd(e, \Phi(n))=1$, we are certain that there is exactly one solution between 0 and $\Phi(n)$ that satisfies the above equation.

The public key is: (e, n).

The private key is: (d, n).

Step 4: Let M be the plain text and C be the cipher text.

Encryption

$f(M) = C = Me \bmod n$

Decryption

$f^{-1}(C) = Cd \bmod n = Med \bmod n = M.$

An example of the RSA algorithm:

We now look at an over simplified example for illustrating the algorithm.

Let p=7 and q=17, be two randomly selected primes.

$n=7*17=119$

$\Phi(n) = (7-1)*(17-1) = 96$

We choose randomly, e such that gcd(e,96)=1. Let e=5, gcd(96,5)=1. Thus there exists an integer d such that $5*d==1 \bmod 96$ or $d=5^{-1} \bmod 96$, so d=77.

Let the plain text M=19.

Then C=195 mod 119=66.
And M=6677mod 119=19, as desired.

IV. A SOFTWARE WATERMARKING Method BASED ON PUBLIC- KEY CRYPTOGRAPHY AND GRAPH COLORING

The method proposed is based on public key encryption algorithm and graph coloring problem. In this algorithm, it requires the vertices of the graph to be indexed, that is, each vertex must be labeled with a unique integer in the range 0 to |V (G)|-1. Our algorithm relies heavily on the ordering of node indices. The followings are some concepts used in the algorithm.

Definition1. K-colorable we say a graph G = (V (G), E (G)) is k-colorable if it has an ancillary coloring function F: V= (v_0, v_1,…, v_{n-1}) with the following properties.

$(v_i,v_j) \in E(G) \Rightarrow C(v_i) \neq C(v_j)$

Definition2. Cyclic mod n ordering: We use "<" to denote the cyclic mod n ordering, such that $0 < 1 < \ldots < n-1$.

Definition3. Two candidate vertices: for a vertex v_i of a graph G with |V | = n and a coloring of G, we say v_i has two candidate vertices $v_{i1} \in V$ and $v_{i2} \in V$ if $i < i_1 < i_2 < n$ and vertices v_i, v_{i1}, and v_{i2} have a same color and $(v_i, v_{i2}) \notin E$; furthermore, $j: i < j < i_1$ and $j: i_1 < j < i_2 < n$, vertices v_i and v_j have different color.

A. Embedding Algorithm

Given an interference graph G (V, E) and a message W to be embedded in G. Let V= (v_0, v_1,…, v_{n-1}) and we use RSA Cryptosystem to encrypt the message W into cipher text M. Then convert the cipher text M to binary $M=m_0m_1\ldots$, i.e. the cipher text M is watermark bits and M is embedded into the graph G.

The principal steps of Embedding algorithm are given as follows:

- Firstly, use RSA Cryptosystem to encrypt the copyright information W into cipher text M and change the message M into a binary string $M=m_0m_1\ldots$.The public key and private key ,which are (e, n) and (d, n) respectively, are affected on W;
- Secondly, find two candidate vertices with respect to v_i（$0 \leqslant i \leqslant n-1$） in the given graph G. If there exit two candidate vertices with respect to v_i, then carries out the third step, otherwise do the second step.
- Thirdly, the watermark bits to be embedded determine the choice between these candidate vertices. If the watermark bit is 0 then connect v_i to v_{i1} , otherwise connect v_i to v_{i2};
- Finally, change the color of the current connected vertices different one from the adjacent vertices' colors.

Instance 2 and Instance 3 show RSA Encryption Algorithm and Embedding Algorithm.

```
//input the plain text
Public void getText () throws Exception
 {
System.out.println ("please input the plain text :");
BufferedReader stdin=new BufferedReader(new Input-
StreamReader(System. in));
Stringbr=stdin.readLine ();
this.text=Long.parseLong(br);
}
/*Encryption calculation, use the formula
secretWord=text^Public_key (mod n) to receive cipher
text secretword*/
Public long Colum (long W, long n, long e)
{
//w:the plain text, n: p*q, e: public_key
Long M;
if (e==1)
M=W%n;
else
M=W*this.colum(W,n,e-1)%n;
return M;
}
```

Instance 2. RSA encryption algorithm

```
Input:    an original graph G (V, E)
          a message M= M=m0m1…mk to be embedded
          into the G(V,E)
Output: a watermarked graph G' with message M
          embedded in it
Algorithm:
N=|V|;
G'=G;
s=0;
if k>n return G // not all bits of M inserted in G
for ( i=0; i<n; i++)
  if s>k return G′ //all bits of M already inserted in G
  if  vi has two candidate vertices vi1 and vi2
      if  ms=0
          connect vi to vi1 in G'
          change the color of vi1 to different one from the
          current colors used in G'
      else
          connect vi to vi2 in G'
          change the color of vi2 to different one from the
          current colors used in G'
   s++;
if k>s //not all bits of M inserted in G
  return G
else
  return G'
```

Instance 3. embedding algorithm

In Figure. 3, vertices 2 and 3 are the nearest two vertices that are not connected to vertex 0.The essence of this technique is to add an extra edge between two vertices, These two vertices have to be colored by different colors which may not be necessary in the original graph G. Fig. 3 shows a graph of 11 nodes with solid lines for original edges. The message $249_{10}=11111001_2$ has been embedded by 11dotted edges, each represents on bit marked on the edge.

A4-color scheme:

$\{\{v_0,v_2,v_6\}.\{v_1,v_7,v_{10}\},\{v_3,v_8,v_9\}\{v_4,v_5\}\}$.

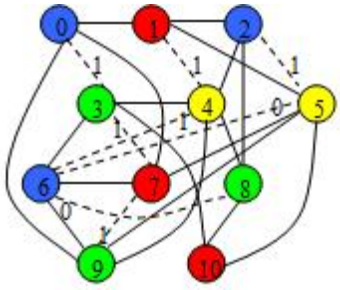

Figure 3. an example

B. Extraction Algorithm

Input: an unwatermarked graph G (V, E) with n=|V|
a watermarked graph G'(V,E')
Output: the message M embedded in the watermarked graph G' (V, E')
Algorithm:
s=0;
for (i=0; i<n; i++)
if v_i has two candidate vertices v_{i1} and v_{i2}
if v_i and v_{i1} have different colors in G'
m_s=0;
connect v_i and vi_1 in G
change the color of v_{i1} to different one from the current colors used in G
else
m_s=1;
connect v_i and v_{i2} in G
change the color of v_{i2} to different one from the current colors used in G
s++;
return M=$m_0m_1\ldots m_s$

Instance 4. extraction algorithm

```
// Decrypt encrypted
Public void pascolum()throws Exception
{
this.getText();
System.out.println("input the plain text:"+this.text);
//encryption
this.secretword=this.colum(this.text,this.n,this.pub-
lic_key);
System.out.println("Ciphertext
obtained:"+this.secretword)
//decryption
this.word=this.colum(this.secretword,this.n,this
vate_key);
System.out.println("the plain text obtained:"+word);
}
Public static void main (String [] args) throws Exception
{
Rsa t=newRsa();
t.inputPQ();
t.getPublic_key();
t.getPrivate_key();
t.pascolum();}
```

Instance 5. RSA decryption algorithm

V. Technique Analysis And Experimental Results

A. Technique Analysis

The message can be anything that is capable of identifying authorship. We can transfer it into binary, encrypt it by a public key cryptography and assume the final bit stream is random. To have quantitative analysis, we assume that are required to color the graph $G_{n,p}$, the consists of G with n vertices and the edges are chosen independently with probability p(0<p<1),b=1/(1-p),G where C is given by:

$$C(G_{n,p}) = \frac{n}{2\log_b n} \quad (1)$$

It follows immediately that after adding e extra edges into the graph $G_{n,p}$ according to the message, the resulting graph remains random with the same number of vertices and a new edge probability:

$$p' = p + \frac{2e}{n(n-1)} \quad (2)$$

So formula (1) for the chromatic number still holds, we denote this number by C′. The overhead is defined to be C′-C, i.e., the number of extra colors required to color the watermarked graph. The more edges we add, the more colors we need to mark the graph. Since the number of colors is one of the most important criteria for the quality of coloring scheme, we want to keep this overhead as low as possible.

Adding $e(n)$ edges to a random graph $G_{n,p}$, for almost all $G_{n,p}$, $\lim_{n\to\infty} C' - C = \infty$, iff $e(n) \in \omega(n\log_b n)$.

Adding $e(n)$ edges to g*raph* $G_{n,p}$, *if* $\lim_{n\to\infty} \frac{e(n)}{n\ln n} = 1$, *then* $\lim_{n\to\infty} C' - C \le 1 + \frac{1}{1-p}$. *In particular, if* $e(n) \in o(n\log_b n)$, *for almost all* $G_{n,p}$,, *the overhead is at most 1.*

B. Experimental Results

The main goal of our experiment is compare the difficulty of coloring the original graph vs. the watermarked graph, as well as the quality of solution. As shown in table 1, each graph is colored 10 times and the average result is reported. Experiments are conducted by java.

TABLE I. Color The Watermarked $G_{N, 0.5}$

Original $G_{n, 0.5}$		Colors	Adding n Edges		
n	*Edges*		*Edges*	*Information*	*Overhead*
125	2973	19	125	125	0
250	4000	30	250	250	0.2
451	8691	30	451	451	0.2
500	11650	50	500	500	0.3
1000	21600	85	1000	1000	0.9

Table 1 shows the results on random graphs $G_{n, 0.5}$, and the corresponding watermarked graphs by add n random edges. The columns labeled colors are the average numbers of colors on 10 trials for each instance, while the columns information measure the amount of information(in bits)

being embedded in the graph and overhead are the number of extra colors required in the graph coloring problem.

VI. CONCLUSION

In this paper, a new method of software watermarking which utilizes approach of register allocation by coloring the interference graph of this software and public key encryption algorithm(RSA) is proposed, which are provably capable to provide high credibility with at most 1-color overhead for large graphs. Asymptotic formulae are given on the amount of information that can be embedded into the graph without too much overhead.

On the basis of the exiting algorithms, the next stage of research work is to improve anti-attack capability of the core algorithm for software watermarking.

ACKNOWLEDGMENT

This paper is sponsored by Science Foundation of aviation in China (2007zc56003) and Nature Science Foundation of Jiang xi province in China (2008GZS0033).

REFERENCES

[1] W. Zhu, C. Thomborson, and F.-Y. Wang, "A survey of software watermarking", In IEEE ISI 2005, volume 3495 of LNCS, 2005, pp. 454-458.

[2] W. Zhu, C. Thomborson, and F.-Y. Wang, "Application of homomorphic function to software obfuscation", In WISI 2006, volume 3917 of LNCS, 2006, pp. 152-153.

[3] W. Zhu, C. Thomborson, and F.-Y. Wang, "Obfuscate arrays by homomorphic functions", In Special Session on Data Security and Privacy in IEEE GrC 2006, pp. 770-773.

[4] Scott A.Moskowitz and Marc Cooperman, "Method for stega-cipher protection of computer code", US Patent 5,745,569, January 1996. Assignee: The Dice Company.

[5] J. Nagra and C. Thomborson, "Threading Software Watermarks", Proc. Information Hiding Workshop, 2004.

[6] G. Qu and M. Potkonjak, "Analysis of Watermarking Techniques for Graph Coloring Problem", Proceeding of 1998 IEEE/ACM International Conference on Computer Aided Design, ACM Press, 1998, pp. 190-193.

[7] William Zhu and Clark Thomborson, "Algorithms to Watermark Software through Register Allocation", R. Safavi-Naini and M. Yung (Eds.): DRMTICS 2005, LNCS 3919, 2006, pp. 180-191.

[8] Fernando Magno Quintao Pereira and Jens Palsberg, "Register Allocation via Coloring of Chordal Graphs", In Proceeding of APLAS'05, Asian Symposium on Programming Languages and Systems, 2005.

[9] Hung-min sun, Wu-Chuan Yang and Chi Sung Laih, "On the Design of RSA with Short Secret Exponent", Journal of Information Science and Engineering 18, 2002, pp. 1-18.

[10] Johannes Blömer, Er May, "A generalized Wiener attack on RSA", In Practice and Theory in Public Key Cryptography (PKC 2004), Lecture Notes in Computer Science, Springer-Verlag 2947