第42卷第1期 中国科学技术大学学报 Vol. 42, No. 1
2012年1月 JOURNAL OF UNIVERSITY OF SCIENCE AND TECHNOLOGY OF CHINA Jan. 2012

文章编号:0253-2778(2012)01-0077-08

基于混沌优化的动态水印算法研究

罗养霞[1,2],房鼎益[2]

(1.西安财经学院信息学院,西安 710127;2.西北大学信息科学与技术学院,西安 710069)

摘要:针对现有软件水印隐蔽差、鲁棒性低的问题,将混沌理论应用于软件水印算法中,基于混沌替换、混沌加密算法对动态图CT算法进行优化和改进,提高水印的隐蔽性和鲁棒性.在VC6.0环境下对C++软件实现基于混沌的动态水印算法(CBDW)原型系统,并以该实现为例进行分析.实验表明,该方法在保留原水印高数据率优点的同时,增强了其隐蔽性和鲁棒性.

关键词:软件水印;混沌理论;动态图水印;隐蔽性;鲁棒性

中图分类号:TP309 **文献标识码**:A doi:10.3969/j.issn.0253-2778.2012.01.012

引用格式: Luo Yangxia, Fang Dingyi. Dynamic watermarking algorithm based on chaotic optimization[J]. Journal of University of Science and Technology of China, 2012,42(1):77-84.
罗养霞,房鼎益. 基于混沌优化的动态水印算法研究[J]. 中国科学技术大学学报,2012,42(1):77-84.

Dynamic watermarking algorithm based on chaotic optimization

LUO Yangxia[1,2], FANG Dingyi[2]

(1. *Department of Information, University of Finance & Economics, Xi'an* 710127, *China*;
2. *School of Information and Technology, Northwest University, Xi'an* 710069, *China*)

Abstract: In order to improve the stealth and robustness of software watermarking, the chaos theory was applied to the software watermarking algorithm, mainly by optimizing and improving the dynamic graph CT algorithm based on chaos-replacement and chaos-encryption. Chaos-based dynamic watermarking algorithm (CBDW) was achieved under the VC6.0 C++ environment, and some analyses were conducted. Experimental results demonstrate that, while maintaining its advantage of high data rate, the proposed approach is effective in enhancing stealth and robustness of the original watermark.

Key words: software watermarking; chaos theory; dynamic map watermarking; stealth; robustness

0 引言

软件水印是数字水印的一种,通过一定的嵌入算法,将版权信息隐藏在软件中,当发生版权纠纷时,从软件中提取水印信息,以达到保护载体的目的[1].该技术是密码学、软件工程、程序设计、算法设计、图论等学科的交叉研究领域.软件水印是否具有鲁棒性,也直接关系着版权认证的实效性.

目前,用于软件版权保护的水印研究主要集中在提高水印的鲁棒性、隐蔽性和数据率几个方面[2].对于水印的鲁棒性,文献[2]给出证明,动态水印算法比静态算法在抗去除攻击方面鲁棒性更高.动态水印算法主要可分为以下4类:

(Ⅰ) Easter Egg水印[3]

收稿日期:2011-04-28;**修回日期**:2011-09-06
基金项目:陕西省教育厅科研计划项目(11JK0987,2010JK554),陕西省科技厅自然科学基金研究项目(2011JM8016)资助.
作者简介:罗养霞(通讯作者),女,1974年生,讲师.研究方向:数字水印,软件及信息安全.E-mail:Yxluo8836@163.com

该水印算法是通过一个输入产生一个输出，水印无须检测，而是封装在应用程序当中的，但这种封装，使攻击者容易找到水印在程序中的位置，进行去除或剪切攻击，而且数据率较低.

（Ⅱ）动态执行轨迹水印[4]

通过对程序中指令的执行顺序或内存地址走向进行编码生成水印，水印检测则通过控制地址和操作码顺序来进行，并给出机器码及 Java 的实现过程. 这种算法过度地依赖顺序性，而且构造较复杂.

（Ⅲ）基于线程的水印[5]

通过线程竞争编码水印，改变线程数量，应用多线程程序难调试的特点提高鲁棒性，但由于引入大量线程，会降低程序的执行效率.

（Ⅳ）动态数据结构水印[6]

输入特定信息激发程序把水印信息隐藏在内存堆栈或者全局变量域等程序状态中. 当所有信息都输完之后，通过在内存中检测程序变量的当前值来进行水印提取. 后经过多种改进研究[7-9]，使得该算法在数据率和抵抗保持语义的各种攻击方面鲁棒性较高，但研究仍存在以下几个方面的不足：

①水印是通过标记 Mark( )位置，指定嵌入源码中，嵌入水印的代码与周围的原代码存在差异(weak cut)，隐蔽性较低；

②水印被分割成子水印后存在一定的相关性，容易被按次序破解；

③所使用的加密算法是以传统的加密方式，每个子水印加密方法相同，攻击方式相同，容易遭受到设置断点、输出明文的攻击.

本文针对以上的不足，拟研究采用非线性的混沌理论对动态图 CT 算法进行优化，在保留其动态水印、数据率高等优点的同时，隐藏其嵌入位置和加密方式，提高其鲁棒性.

1 相关工作

1.1 混沌及混沌系统

混沌是描述非线性动力学系统中出现的一种类似随机不确定输出，无序中又包含有序，其具有不可分解、有规律性以及不可预测性的特征[10].

本文在研究中应用的混沌系统是改进的帐篷映射[11]. 一是由于其迭代轨道有很好的伪随机特性和自相关性；二是由于该算法本身不复杂，易于计算和实现，有利于提高水印提取效率. 混沌映射函数如下：

$$X(t+1)=F_p(X(t))=\begin{cases}X(t)/p, & x\in[0,p);\\ (X(t)-p)/(0.5-p), & x\in[p,0.5);\\ (1-X(t)-p)/(0.5-p), & x\in[0.5,1-p);\\ (1-X(t))/p, & x\in[1-p,1].\end{cases}$$

转换成二进制函数如下：

$$S_n(t)=R_n(x(t))=\begin{cases}0, & x(t)\in\bigcup_{d=0}^{2^{n-1}-1}I_{2d}^n;\\ 1, & x(t)\in\bigcup_{d=0}^{2^{n-1}-1}I_{2d+1}^n.\end{cases}$$

式中，n 为任意正整数；$I_0^n, I_1^n, I_2^n, \cdots$ 是区间[0,1]的 2^n 个连续的等分区间. 为了提高序列的混杂性，引入参数 P 的随机变化和 m 序列[12]对该混沌序列进行改进，步骤如下：①利用混沌系统生成混沌序列 c；②将混沌序列与扰动 m 相加，$r=c+m$；③对 r 进行二进制转换得到输出序列.

1.2 混沌理论在水印中的应用

混沌理论在图像水印中的应用较多，归纳为以下几个方面[13-15]：①将混沌序列直接作为水印信息进行嵌入；②应用混沌系统对水印进行预处理；③应用混沌系统对水印的嵌入过程进行控制，如混沌置乱图像块、混沌加密特殊像素. 但遗憾的是，混沌理论在软件水印方面的应用较少.

文献[16-17]首次将混沌理论应用在软件水印系统中，通过混沌预处理和混沌散列编码来改进传统水印算法 Easter Egg 水印. 但由于该算法在嵌入水印后，改变了各模块代码及在内存中的加载位置，需要通过自定义代码标记来指定位置，降低了水印的隐蔽性.

文献[18]研究将混沌散列应用于动态图水印算法，以使动态图子水印分布到整个程序中，以增强鲁棒性，但并没有给出具体的实现过程及在目标代码中的加载方法.

受以上算法的启发，本文应用混沌理论对动态图 CT 算法[8]进行混沌替换和混沌置换，以增强水印的隐蔽性(解决节 1 中的不足①和②)；对大数分解后的子水印，以及软件的嵌入函数等敏感信息进行混沌加密，并结合嵌入位置的 SHA 值实现水印系统的防篡改功能和抵抗逆向攻击能力(解决引言中的不足③)，以提高水印的隐蔽性和鲁棒性.

2 CBDW 算法框架

针对动态图水印 CT 算法的不足，提出基于混

沌替换和混沌加密的软件水印算法——CBDW 算法(chaotic-based dynamic watermark). 该算法在原 CT 算法的基础上增加两个模块:混沌替换和混沌加密,主要包括两个过程:水印嵌入和水印提取,如图 1 所示. 这里主要介绍和混沌有关的算法,关于拆分、图拓扑编码等不作详细介绍.

2.1　水印的嵌入过程

①对水印信息 W 使用中国剩余定理(GCRT)[8]分割成子水印 $W_1, W_2, \cdots, W_n$.

②使用混沌替换算法 CR 将子水印混沌替换成 $W_1', W_2', \cdots, W_n'$.

③对程序源码进行预处理:标记、追踪,确定水印嵌入位置及函数调用关系;确定被加密的函数集合;以及插入校验和函数、hash 函数、加解密函数.

④对 $W_1', W_2', \cdots, W_n'$进行 PPCT 图拓扑编码,并插入到指定的位置.

⑤编译生成目标代码 $P'w$.

⑥应用混沌加密算法 CE 对图拓扑构造函数及嵌入的敏感代码段进行迭代加密.

2.2　水印的提取过程

①按照特定的输入来触发程序执行.

②对水印及关键代码段进行解密.

③当输入完毕后,在堆中会生成表示水印子数据的 PPCT 结构,由每个 PPCT 结构的根节点得到 PPCT 完整的图结构,然后将 PPCT 结构解码为相应的水印子数据 w_i'.

④根据混沌替换逆算法,将 w_i'还原为 w_i.

⑤根据中国剩余定理分割原理,将一系列的子水印 w_i 还原为最初的大水印数据 w.

2.3　混沌替换 CR 算法

用混沌序列对子水印进行隐藏,替代后的序列是混沌序列与水印序列求异或运算而生成. 首先将混沌序列和水印序列分割成 8 bit 长度的子序列,再规范序列的二进制位数,使混沌序列和水印序列等长,最后进行异或运算. 伪代码表示如下:

```
C;      //chaotic sequence;
W;      //watermarks;
ChaoticReplace {
Generate (C);      //From chaotic system;
Split C→c1 ,c2 ,…,cm ;      //ci 8-bit sequence;
Split W→w1 ,w2 ,…,wn ;      //wi 8-bit sequence;
If (m<n)
  C={c1 ,c2 ,…,cm ,0,0,…,0,};      //add 0 at the end of
                                    sequence;
If (m>n)
  C={c1 ,c2 ,…,cn }      //delete sequence from cn+1 to cm ;
Then For (i=1;i<=n;i++) {
  si =ci⊕wi
  S={s1 ,s2 ,…,sn }={c1⊕w1 ,c2⊕w2 ,…,cn⊕wn }
  }
}
```

其逆过程为

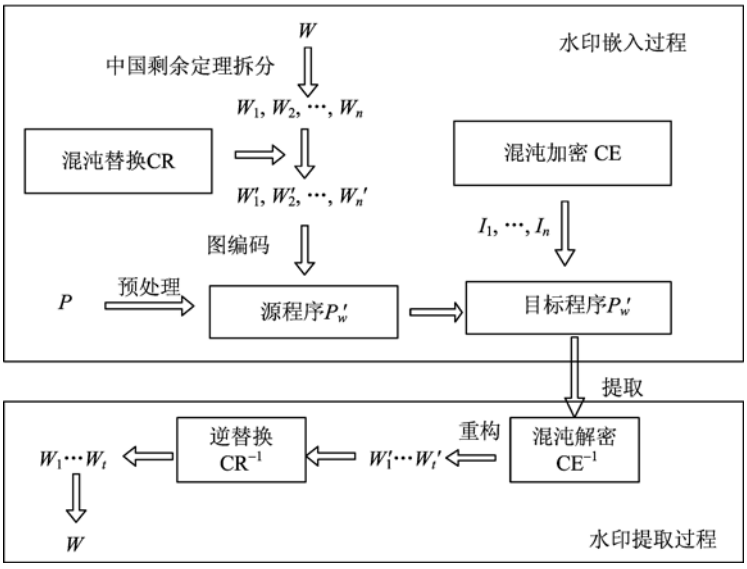


图 1　基于混沌的软件水印嵌入/提取过程

Fig. 1　The embedding and extracting process of CBDW

$$W = CR^{-1}(S) = C \oplus S = \{c_1 \oplus s_1, c_2 \oplus s_2, \cdots, c_n \oplus s_n\} = \{w_1, w_2, \cdots, w_n\}.$$

2.4 混沌加密 CE 算法

混沌在密码学中的应用，主要是两种模式：①直接利用混沌本身构建密码，但是它的情况比较复杂，安全性及稳定性还有待进一步研究；②混沌与传统的特性优异的密码算法相结合构建新的密码算法，目前这种方法是较实用的混沌密码算法.

我们设计一种 AES 算法与混沌序列相结合的加密方案，抵御已知明文的差分和线性攻击. 可变长密钥和迭代保护机制是该体制的设计要点，对水印及敏感代码实施保护，并结合敏感代码的哈希杂凑值，实现水印系统的防篡改功能，加密过程如图 2 所示.

混沌加密器是对目标代码的加密过程. 加密器每次只加密一个函数，可以重复使用加密器的加密功能，直到所有函数都被加密为止. 算法如下：

输入：输入待加密的文件名、混沌序列 q_i，密钥 k_i、待加密段的起始地址 beginAddr，以及待加密段的结束地址 endAddr. 这两个地址是待加密段代码相对 PE 文件头的偏移地址，可以通过 FlexHEX 工具根据代码段里插入的标记获得. k_i 的值是在水印嵌入过程中修改位移表 T 时已计算并记录下来的，此时可以直接使用.

处理：计算加密密钥，使用加密算法对 beginAddr 与 endAddr 间的代码进行加密.

①由水印分支函数从水印子数据 w_i 对应的 PPCT 结构中提取 w_i 的值，并根据公式 $k_i = \mathrm{SHA1}[k_{i-1} \oplus w_i]$，计算下一个密钥 k_i；

②根据 k_i 和完美 hash 函数 h，计算 $h(k_i)$ 的值后，查找构造表 T，得出待计算校验和代码的起止地址 address1 和 address2，其中 address1 $= T[h(k_i)][1]$，address2 $= T[h(k_i)][2]$；

③计算程序加载到内存中的基地址 pMZheader；

④计算 address1 和 address2 之间的代码段在内存中的绝对地址 absoluteAddr1 和 absoluteAddr2；

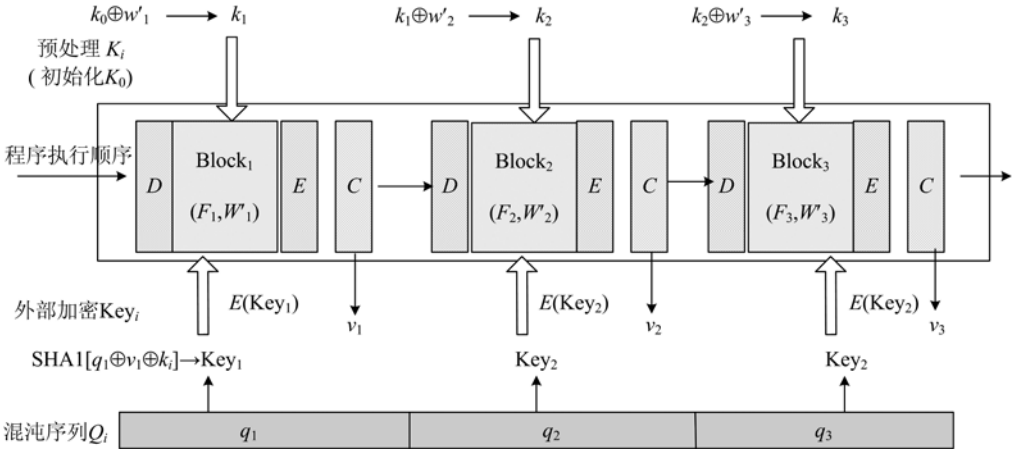
⑤使用 SHA1 算法，计算 address1 和 address2 之间代码段的校验和 v_i = Check (address1, address2)；

⑥根据混沌系统计算 q_i，并保存当前状态 C_i，以用于解密；

⑦根据公式 $\mathrm{Key}_1 = \mathrm{SHA1}[q_1 \oplus v_1 \oplus k_i]_1$，计算加密密钥 Key_1；

⑧查询 beginAddr 和 endAddr 之间的代码在内存中的页，并将该页的保护属性设置为可读写；

⑨调用 AESEncryptFile (szFileName, key, beginAddr, Size) 加密算法，将 Key_1 作为密钥，对



Block(f, w)：嵌入水印后的待保护程序块；C：计算校验和的函数；
V_i：程序中的敏感性代码段，由校验和函数计算它们的短消息摘要；
E：加密算法，用来加密待保护块；D：解密算法，用来解密待保护块；
W'_i：嵌入在程序里的经过混沌替换后生成的 PPCT 结构的水印代码

图 2 混沌加密水印系统示意图

Fig. 2 Chaotic encryption schema of watermarking system

文件中的 beginAddr 与 endAddr 之间的代码进行加密，并将密文写回文件.

⑩将页的保护属性重新设置为只读.

输出：将密文写回文件.

3　系统实现

3.1　系统功能模块

在 Windows XP 的平台上，采用 VC6.0 作为开发工具，开发语言为 C++，实现基于混沌的水印嵌入系统(CBDW WMark)，系统模块设计如图 3 所示. 水印数生成模块分割子水印，然后混沌替换模块构造表示这些子数据的 PPCT 结构代码；源代码处理模块对需要嵌入水印信息的源代码进行处理后，再将其编译成目标代码；目标代码处理模块根据一定的策略，修改生成的目标代码，最后使用混沌加密器对目标代码中的部分代码进行加密.

3.2　系统实现类图

CBDW WMark 采用面向对象的设计思想，将实现不同功能的模块划分为不同的类，提高了系统的易操作性、可扩展性和实用性. 系统类图如图 4 所示，下面对每个类进行简要说明.

GeneratorWM 类：实现水印生成及混沌替换模块. 包括：调用中国剩余定理，将水印分割为一系列的子水印；用混沌系统生成的混沌序列进行替换处理；计算每个子水印的 PPCT 结构的叶节点序号及其右指针所指的节点序号.

WMController 类：实现水印嵌入与提取过程.

WMEmbedController 类：负责嵌入水印信息，对源代码的混沌加密、编译.

WMExtractController 类：实现水印提取模块.

WMTrack 类：对源代码进行标记、追踪，跟踪触发到的函数.

WMInsert 类：插入校验和计算函数、完美 hash 函数以及生成 PPCT 结构的代码.

CE Process 类：处理与混沌加密相关的功能.

Chaotic Encryptor 类：对给定位置处的代码进行混沌异或计算. 该类有 2 个方法：①产生混沌密钥的方法；②对异或计算方法.

3.3　嵌入时遇到的困难

水印嵌入时，PE 文件代码节(.text)中存储的是软件源代码经编译后产生的机器码指令. 若指令 x 长度大于跳转指令，则将此指令移入构造代码段，用指向构造代码段的跳转指令替换此指令. 在此过程中，可使用的跳转指令有多种，可以是无条件跳转指令(JMP)、子程序调用指令(CALL)，还可以是一定会发生跳转的(伪)有条件跳转指令. 加密部分为替换的指令位置(即水印位置)及构造代码段起始位置. 水印嵌入前后可执行文件(PE)的变化情况如图 5 所示.

4　性能比较与分析

4.1　混沌序列的自相关特性

混沌系统取有限精度 $L=12$，扰动位数 $n=2$，$x_0=0.125$，参数 m 序列的级数为 9，特征多项式为 $x^9+x^6+x^3+x$，扰动 m 序列的级数为 13，特征多项式为 $x^{13}+x^7+x^3+x$. 取 500 个点为例，在 Matlab 中给出混沌序列信号流分布函数和对应的二制信号流自相关函数模拟结果.

从图 6 和图 7 中看出，该混沌序列生成的密钥流在有限精度下，经过扰动，具有良好的随机性和自相关特性.

4.2　混沌鲁棒性

破解一维线性分段混沌参数 p，需要该混沌同一分段的 2 个值对，1 个值对落入 1 个指定的混沌

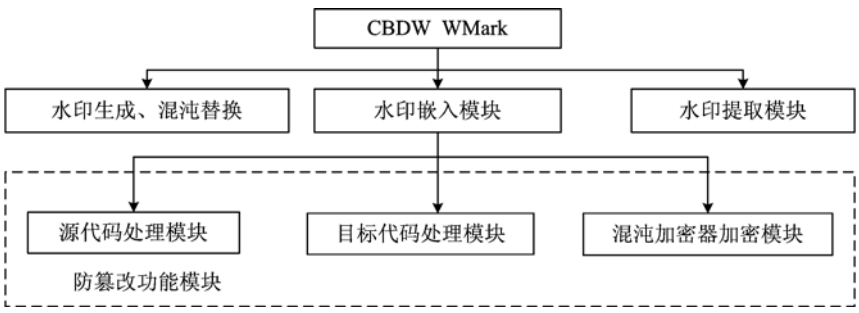


图 3　CBDW WMark 模块结构图

Fig. 3　CBDW WMark module structure diagram

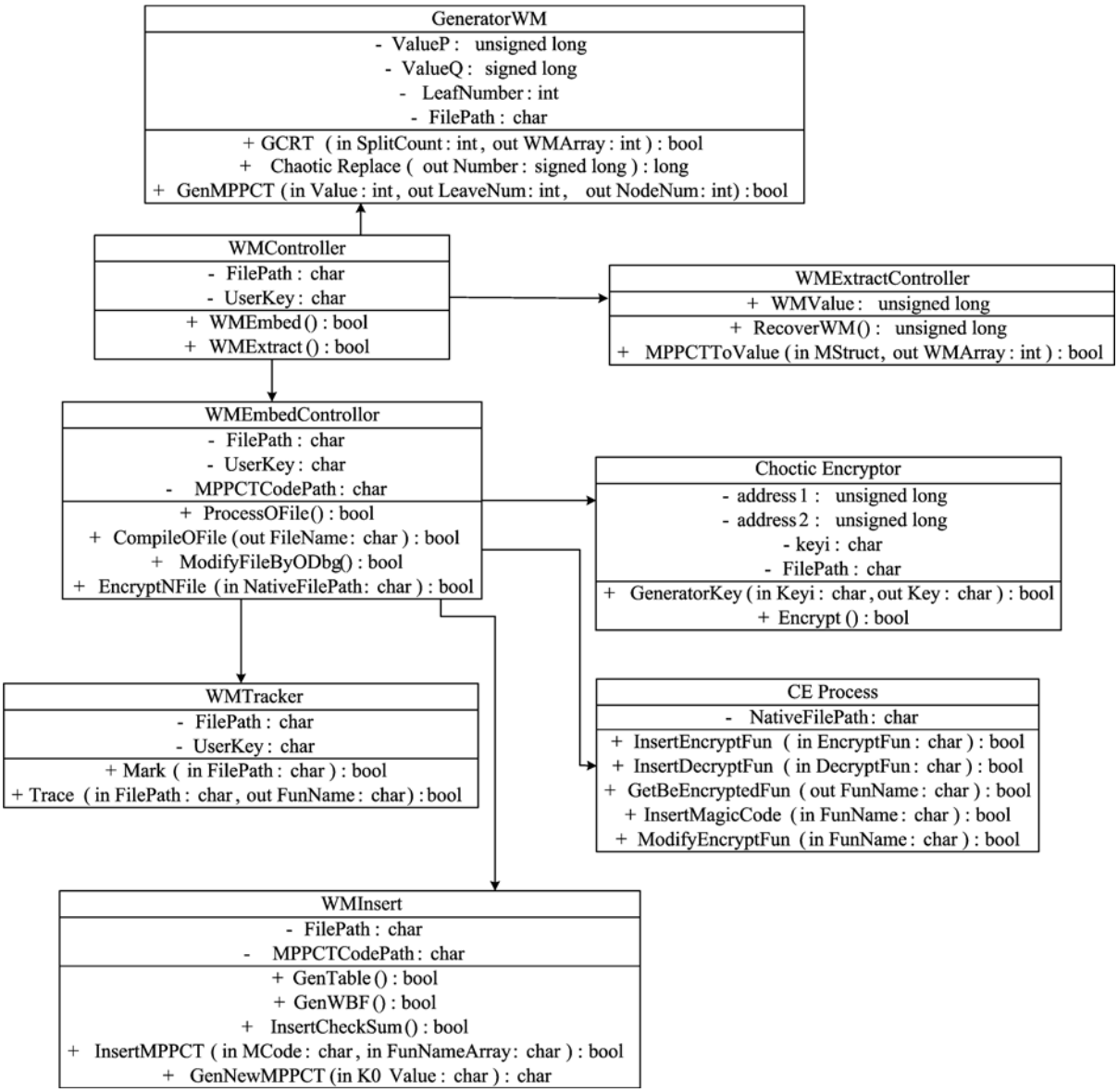


图 4 CBDW WMark 类图

Fig. 4 CBDW WMark class diagram

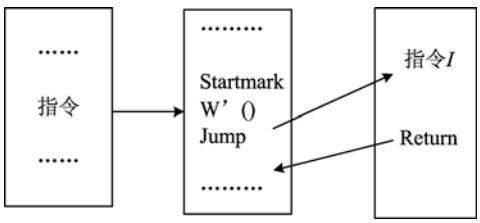


图 5 指令跳转示意图

Fig. 5 Schematic diagram of jump instructions

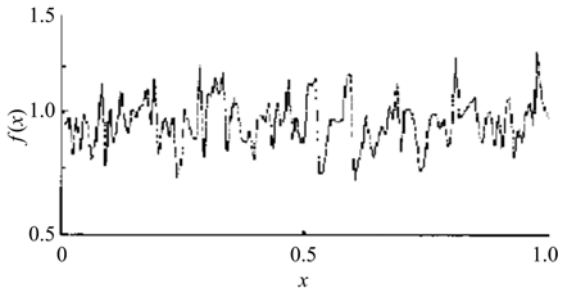


图 6 混沌信号序列分布函数

Fig. 6 Distribution function of chaotic signal sequence

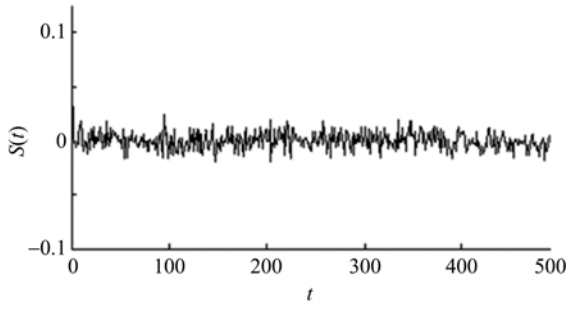


图 7　混沌序列自相关特性

Fig. 7　Self-correlation property of chaotic sequence

概率为 n^{-1}，2 个值对同时落入同一混沌的概率为 n^{-2}. 由于这里的参数 p 是动态变化的，全部破解 $p_1, p_2, \cdots, p_n$ 这 n 个参数，且在 n_p 级序列 $2^{n_p}-1$ 个状态作参数周期变化的情况下，其复杂度是一维线性分段混沌的 $(2^{n_p}-1)^3-2^{3n_p}$ 倍，复杂度指数增大. 对任意小的值 ε，总可以找到 m 序列的级数 $n_p \geqslant \log_2(\varepsilon^{-1/2}+1)$，使得指定的 2 个值对落入该混沌系统的概率为 $P_{cs}=(2^{n_p}-1)^{-2} \leqslant \varepsilon$. 由此，只要适当的选择级数 n_p，就满足 $P_{cs} \leqslant \varepsilon$，可以具有抗选择明文攻击的强度.

4.3　数据率分析

对算法中采用的 PPCT 图拓扑结构与 K 基数循环链表拓扑结构的数据率进行对比，结果如表 1 所示.

表 1　不同水印拓扑图结构数据率对比

Tab. 1　Watermark data rate comparison between 2 kinds of topologies

节点数	K 基数循环链表	PPCT 结构
M	$M^{M-1}-1$	$\frac{2}{M}C_{M-2}^{\frac{M-2}{2}}$
2	1	1
10	1×10^{9}	14
100	1×10^{198}	5.09×10^{26}
200	2×10^{398}	2.27×10^{56}

在给定节点数目情况下，PPCT 结构的数据编码容量虽然低于 K 基数循环链表，但是由于自身结构的特点，它的抗攻击性要高于 K 基数循环链表. 在水印系统中，使用混沌级联加密技术增强系统的安全性，攻击者很难找到 PPCT 水印的位置，所以采用 PPCT 结构作为隐藏水印信息的拓扑图结构，是一种良好的水印编码方案.

4.4　性能过载分析

水印的嵌入肯定会给程序带来性能影响，这里从空间过载(表 2)以及时间过载(表 3)两个方面衡量水印性能影响. 下面通过实验数据来分析嵌入水印前后，宿主程序在空间和执行时间上的变化. 测试环境如下：操作系统 Windows XP；处理器 Intel (R) Core (TM)2 Duo CPU E8400 3.00 GHz；内存 3.25 G；水印数据为 3.

表 2　嵌入水印前后程序大小改变情况

Tab. 2　Changes of program sizes before and after embedding watermark

测试程序	原始大小/kB	嵌入水印后大小/kB
Notepad	80	86
转换工具	356	358
MSN	11 878	11 879

表 3　嵌入水印前后程序执行时间改变情况

Tab. 3　Changes of execution time before and after embedding watermark

测试程序	原时间/ms	嵌入水印后时间/ms	增长比例/%
Notepad	14	15	7
转换工具	16	18	12.5
MSN	45	51	13.3

【注】　执行 10 次取平均值

4.5　抗逆向攻击

基于动态图拓扑水印，在堆栈中生成动态图水印结构，可防止静态逆向攻击. 在利用迭代混沌加解密子水印后，设子水印数为 m，载体可嵌入地址为 n，混沌状态为 χ，则通过水印及敏感性迭代加密后，当载体可嵌入地址 n 不变时，要破解所有水印的时间复杂度为 $T_1=A(m-1)\chi$，其中 A 为常数，随着水印数 m 的增大，复杂度呈线性增加，但系统的性能会影响退化. 如果嵌入的水印数 m 不变，破解所有代码所需要的时间复杂度为

$$\delta=(m-1)C_n^m\chi=\frac{(m-1)n!}{m!(n-m)!}\chi,$$

当载体 n 增大时，则复杂度呈指数增加，但保护强度会减小. 但这两者都要依赖于混沌状态 χ 是可知的，如果 χ 不可知，则不能破解，而且要在保护强度和程序性能退化之间取折衷.

5　结论

本文中混沌优化对动态图水印算法的保护是基于源代码及目标代码格式下的保护方式，主要通过混沌替换来增加水印的隐蔽性，通过混沌加密来控制水印的嵌入过程，提出基于混沌理论的动态软件

水印改进算法.文中给出算法的详细过程和系统实现结构图,并对所实现的水印系统进行性能评价和分析.下一步的工作是将访问控制和权限发放与此水印系统结合,研究软件版权动态协议,为建立基于水印的软件版权保护系统提供理论分析和技术支撑.

参考文献(References)

[1] Zhang Lihe, Yang Yixian, Niu Xinxin, et al. A Survey on Software Watermarking[J]. Journal of Software, 2003,14(2): 268-276.
张立和,杨义先,钮心忻,等.软件水印综述[J].软件学报,2003,14(2): 268-276.

[2] Collberg C, Nagra J. Surreptitious software: Obfuscation, watermarking, and tamperproofing for software protection. Addison-Wesley[M]. Addison-Wesley Professional, 2009:539-540.

[3] Collberg C, Thomborson C. Software watermarking: Models and dynamic embeddings[C]// Proceedings of the 26th ACM Sigplan-Sigact Symposium on Principles of Programming Languages. New York: ACM, 1999: 311-324.

[4] Collberg C, Carter E, Debray S, et al. Dynamic path-based software watermarking[C]// Proceedings of the ACM Sigplan 2004 Conference on Programming Language Design and Implementation. New York: ACM,2004:107-118.

[5] Nagra J, Thomborsonm C. Threading software watermarks [J]. Lecture notes in computer science, 2004, 3200:208-223.

[6] Collberg C, Thomborson C, Townsend G. Dynamic graph-based software watermarking [R]. Tucson, USA: University of Arizona, 2004: TR04-08.

[7] 苏林. 基于代码加密的防篡改软件水印技术的研究与实现[D].西安:西北大学,2009.

[8] Luo Yangxia, Ma Jun, Zhang Zhigang, et al. Dynamic graph software watermark algorithm based on threshold scheme[J]. Computer Engineering, 2009, 35:153-155.
罗养霞,马君,张志刚,等. 基于门限方案的动态图软件水印算法[J].计算机工程, 2009,35:153-155.

[9] Zhu Jianqi, Liu Yanheng, Wang Aimin. H function based tamper-proofing software watermarking scheme [J]. Journal of Software, 2011,6(1):148-155.

[10] Rawat S, Raman B. A chaotic system based fragile watermarking scheme for image tamper detection [J]. AEU-International Journal of Electronics and Communications, 2011,65: 840-847.

[11] Zhang Xueping, Liu Yawei. A novel spatial obstructed distance by dynamic piecewise linear chaotic map and dynamic nonlinear PSO [J]. Lecture Notes in Computer Science,2010, 6146:468-475.

[12] Acharya B, Sunder S S, Thiruvenkatam M, et al. Image encryption using index based chaotic sequence, M sequence and gold sequence[C]//Proceedings of the 2011 International Conference on Communication, Computing & Security. New York: ACM, 2011: 541-544.

[13] Chang C C, Chen K N, Lee C F, et al. A secure fragile watermarking scheme based on chaos-and-hamming code [J]. Journal of Systems and Software, 2011,84:1 462-1 470.

[14] Sidiropoulos P, Nikolaidis N, Pitas I. Invertible chaotic fragile watermarking for robust image authentication [J]. Chaos, Solitons & Fractals, 2009, 42: 2 667-2 674.

[15] Cao Zaihui, Sun Jianhua. Image algorithm for watermarking relational databases based on chaos [J]. Lecture Notes in Electrical Engineering, 2010, 72: 411-418.

[16] Lu Bin, Luo Xiangyang, Liu Fenlin. A chaos-based framework and implementation for software watermarking algorithm [J]. Journal of Software, 2007,18:351-360.
芦斌,罗向阳,刘粉林.一种基于混沌的软件水印算法框架及实现[J].软件学报,2007,18:351-360.

[17] Liu Fenlin, Lu Bin, Luo Xiangyang. A chaos-based robust software watermarking, information security practice and experience [J]. Lecture Notes in Computer Science, 2006, 3903:355-366.

[18] 张少波.基于混沌的动态图软件水印算法研究[D].湖南:湖南科技大学,2009.