基于混沌优化的分存软件水印方案*

李 斌, 周清雷
(郑州大学 信息工程学院, 郑州 450001)

摘 要: 针对软件水印鲁棒性差、水印分存算法执行效率低的问题,提出了一种基于混沌优化的分存软件水印方案。该方案通过引入混沌系统,将水印信息矩阵分割、混沌置乱,形成分存水印;水印嵌入时,将分存水印一一编码为 DPPCT 拓扑图,并将 hash 处理后的水印信息分别填充于各个 DPPCT 的 info 域;水印嵌入后,利用混沌加密,保护全部代码,防止逆向工程等手段对软件水印的破坏。理论分析和实验表明,该方案可有效地抵抗各种语义保持变换攻击,减少程序负载,提高水印的鲁棒性及执行效率。

关键词: 软件水印; 混沌理论; 水印分存; DPPCT 拓扑图; 鲁棒性
中图分类号: TP309 **文献标志码**: A **文章编号**: 1001-3695(2013)11-3418-03
doi:10.3969/j.issn.1001-3695.2013.11.055

Sharing software watermarking scheme based on chaotic optimization

LI Bin, ZHOU Qing-lei
(*School of Information Engineering, Zhengzhou University, Zhengzhou* 450001, *China*)

Abstract: In order to solve the poor robustness of software watermarking and the low execution efficiency of watermarking sharing algorithm, this paper proposed a sharing software watermarking scheme based on chaotic optimization. The scheme took advantage of chaos system, matrix partition and chaotic scrambling the watermarking information to form sharing watermarking. When watermarking was embedded, using DPPCT topology graph encode the sharing watermarking, then putting the hash value of the watermarking information into the info field of each DPPCT. After the watermarking embedded, using chaotic encryption to protect all code and prevent reverse engineering and other methods to attack the software watermarking. Theoretical analysis and experimental results show that the scheme can effectively resist various semantics preserving transformation attacks, decrease the program load and improve robustness and execution efficiency of the watermarking.

Key words: software watermarking; chaos theory; sharing watermarking; DPPCT topology graph; robustness

人们在享受互联网下载便利的同时,软件盗版和复用也日益猖獗,给个人和企业带来了重大的经济损失,软件版权的保护越来越受到人们的重视。为保护软件版权,软件水印应运而生,它是数字水印的一个分支,是信息安全、密码学、图论、算法设计、软件工程的交叉研究领域。基于图论的软件水印,由于隐蔽性强、安全性高,成为了目前研究的热点。文献[1]提出了首个动态图水印(dynamic graph watermarking, DGW)的 CT(Collberg-Thomborson)算法,其主要思想是用两个大素数 P、Q 的乘积 $W(W=P\times Q)$来表示水印信息,再将 W 编码为某种拓扑图嵌入到程序代码中。由于 W 是个大数,水印嵌入过程中,往往需要将其分割成多个子水印,以提高水印的隐蔽性、鲁棒性并减小构造水印拓扑图的复杂度[2]。文献[3]提出的基于中国剩余定理的水印分存算法,其隐秘性好,但水印恢复过程复杂、计算量大。文献[4]提出了一种基于密钥分存理论的 Asmuth-Bloom(AB)门限算法,增强了水印的鲁棒性,但在实现过程容易导致水印数据的大幅扩张。文献[5,6]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[7]提出了基于混沌优化的动态水印算法,其鲁棒性强,能有效地抵抗逆向工程的攻击,但混沌加密过程复杂,对程序性能影响较大。

针对以上方案的缺点,本文提出了一种基于混沌优化的分存软件水印方案,通过对水印信息的矩阵分割、混沌置乱(chaotic scrambling, CS),有效地控制水印分存粒度,提高水印隐蔽性,再将分存的水印一一编码为 DPPCT(double circular linked planted plane cubic tree)拓扑图,并将 hash 处理后的水印信息分别填充于各个 DPPCT 的 info 域,形成多水印信息与水印分支共存的结构。水印嵌入后,利用混沌加密(chaotic encryption, CE)系统,将程序代码划分为敏感代码段(code sensitive block, CSB)和非敏感代码段(code insensitive block, CIB),使用 CIB 的 hash 值,加密 CSB,形成交错保护机制,保护全部代码,提高水印的抗攻击能力及鲁棒性。

1 相关理论

1.1 动态图软件水印基本模型

软件水印系统 S 可用一个十元组$(P_o, W, K, E_m, E_x, P_w, D_r, D_e, A_t, P'_w)$表示。动态图软件水印系统[8]由于加入了水印分存算法和水印拓扑图编码,可以表示为 $DGS=(P_o, \{I_i\}, W, f, f^{-1}, g, R, E_m, E_x, P_w, D_r, D_e, A_t, P'_w)$。其中:$P_o$ 为原始程序;

收稿日期: 2013-02-28; **修回日期**: 2013-04-16 **基金项目**: 国家"863"计划基金资助项目(2009AA012201)

作者简介:李斌(1986-),男,河南郑州人,硕士,主要研究方向为信息安全、软件水印(cctvlibin@163.com);周清雷(1962-),男,河南新乡人,教授,博士,主要研究方向为形式语言自动机、形式化方法、信息安全.

W 为水印信息;P_w 为嵌入水印后的程序;$\{I_i\}$ 为用户输入密钥;水印分存映射函数 f 满足 $f(W)\rightarrow\{W_i\}$;逆映射函数 f^{-1} 满足 $f^{-1}(\{W_i\})\rightarrow W$;拓扑图编码函数 g 满足 $g(W_i)\rightarrow G_i$,G_i 为水印拓扑图结构;R 为识别函数;E_m 为嵌入算法 $E_m(P_o\times W\times I_m)\rightarrow P_w$;$E_x$ 为提取算法 $E_x(P_w\times I_m)\rightarrow P_o$;$D_r$ 为数据率;D_e 为隐蔽性;抗攻击性 $A_t(P_w)\rightarrow P'_w$。

在保证 P_o 和 P_w 功能一致的前提下,动态图软件水印系统的衡量标准主要有:a)数据率,水印拓扑图表示水印信息大小的程度;b)隐蔽性,水印拓扑图的不可察觉程度;c)抗攻击性,水印拓扑图对各种攻击的抵抗程度;d)性能过载,嵌入水印拓扑图后对程序执行性能的影响程度。四个特性之间是相互制约、相互矛盾的,需要对其进行权衡以取得最佳的性能效果。

1.2　混沌理论

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[9]。Logistic 映射来源于著名的统计学模型,是目前广泛应用的一种混沌动力系统,其动态的数学模型可表示为

$$x_{k+1}=\mu x_k(1-x_k)$$

其中:混沌域为(0,1),$0\leqslant\mu\leqslant4$ 称为分支参数,$x_k\in(0,1)$。混沌动力系统的研究工作指出,当 $3.5699456\cdots<\mu\leqslant4$ 时,Logistic 映射工作于混沌态。即由初始条件 x_0 在 Logistic 映射的作用下所产生的序列 $\{x_k;k=0,1,2,3,\cdots\}$ 是非周期的、不收敛的并对初始值非常敏感的。转换为二进制函数为

$$S_n(k)=R_n(x_k)=\begin{cases}0 & x_k\in\bigcup_{d=0}^{2^{n-1}-1}I_{2d}^n\\ 1 & x_k\in\bigcup_{d=0}^{2^{n-1}-1}I_{2d+1}^n\end{cases}$$

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。通过调用 $S_n(k)$ 将混沌序列 $\{x_k;k=0,1,2,3,\cdots\}$ 转换为二进制输出序列。

2　基于混沌优化的软件水印分存方案

基于混沌优化的软件水印分存方案,在原有 CT 算法的基础上,通过对水印信息的矩阵分割处理,形成分存水印,并利用混沌置乱 CS,对分存水印进行加密处理;然后利用 DPPCT 表示加密后的分存水印,并将 hash 处理后的水印信息填充于 DPPCT 的 info 域;最后利用混动加密 CE,对程序的敏感代码段 CSB 进行加密处理。方案整体构架如图 1 所示。

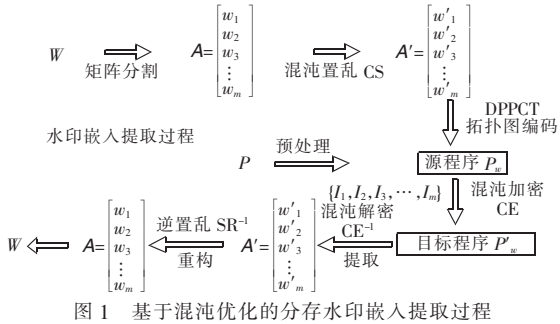

图 1　基于混沌优化的分存水印嵌入提取过程

水印嵌入过程如下:

a)对水印信息 W 进行矩阵分割,形成分存水印 $(w_1,w_2,w_3,\cdots,w_m)$,并将其转换为矩阵 A;

b)利用混沌置乱,对矩阵 A 进行置乱加密,形成矩阵 A';

c)对程序源码进行预处理,确定水印嵌入位置及程序敏感代码段 CSB,添加 hash 函数、加/解密函数;

d)对矩阵 A' 每行的分存水印信息 $w'_i(1\leqslant i\leqslant m)$ 进行 DPPCT 拓扑图编码,并嵌入到程序指定位置,在程序运行状态下通过用户输入的序列 $\{I_1,I_2,I_3,\cdots,I_m\}$,使其在堆栈中生成;

e)编译生成目标程序 P'_w;

f)应用混沌加密对 P'_w 进行加密处理。

水印提取过程为水印嵌入的逆过程。

2.1　矩阵分割算法

该算法通过对水印信息 W 预处理,使其转换为含有水印信息的 0、1 矩阵 A。首先将水印信息 W 转换为二进制信息,根据二进制信息的长度 L_w 将水印信息 W 进行分割,每段长度为 $\lfloor\sqrt{L_w}\rfloor+1$,最后一段长度不足 $\lfloor\sqrt{L_w}\rfloor+1$ 的,判断该段最后一位,是 0 则补 1,是 1 则补 0。处理后的水印信息记为 $W=(w_1,w_2,w_3,\cdots,w_m)$。然后构造一个 $A_m\times A_n$ 的矩阵 A,其中 $A_m=m$,即水印分支数,$A_n=\lfloor\sqrt{L_w}\rfloor+1$,即每段的长度。将分割好的水印信息按序填放在矩阵的每一行。

例如:水印信息 $W=29$,转换为二进制信息为 11101,根据上述算法 $A_m=2,A_n=3$,则 $A=\begin{bmatrix}111\\010\end{bmatrix}$。

2.2　混沌置乱

该算法产生 $2m\times n$ 个混沌序列值,可对任意 $m\times n$ 的 0、1 矩阵进行置乱加密。置乱加密过程如下:

a)由混沌系统产生含有 $2m\times n$ 个混沌序列值的序列 C;

b)将混沌序列 C 进行矩阵分割,得到两个 $m\times n$ 的矩阵 C_1、C_2;

c)将矩阵 A 与 C_1 按行进行异或操作,形成新矩阵 T_1,即 $T_1=A\oplus C_1$;

d)对 T_1 进行转置,形成新矩阵 T_2;

e)将矩阵 T_2 与 C_2 按行进行异或操作,形成新矩阵 T_3,即 $T_3=T_2\oplus C_2$;

f)对 T_3 进行转置,形成新矩阵 T_4。

以上是对水印信息矩阵 A 的一个迭代置乱过程,最后得到矩阵 T_4,即为混沌置乱后的矩阵 A'。若要得到更好的置乱加密结果,可将上述过程进行多次迭代。

其逆过程为:首先将矩阵 T_4 进行转置,得到矩阵 T_5,再将 T_5 与 C_2 进行异或操作,得到矩阵 T_6;然后对 T_6 进行转置操作,得到矩阵 T_7;最后将矩阵 T_7 与 C_1 进行异或操作,得到矩阵 T_8。矩阵 T_8 即原水印信息矩阵 A。

2.3　DPPCT

现在主要的动态水印图拓扑结构有基数 K 链表、父指针树、排列图、PPCT 结构等。其中,基数 K 链表数据率最高,但抗攻击性较差,而 PPCT 结构数据率较低,但抗攻击性能却最好。结合基数 K 链表和 PPCT 结构,构造混合编码[10] DPPCT 结构。

DPPCT 节点在原有 PPCT 节点结构上,为每个节点添加一个指针域和 info 域,其结构如图 2 所示。

DPPCT 的编码结构如下:

a)将混沌置乱后的水印分支 $w'_i(1\leqslant i\leqslant m)$ 展开为基数 k_i 的形式。

b)构造具有 k_i 个叶节点的 DPPCT 结构,从最右叶节点开始改变每个叶节点的 R_1、R_2 指针,以此编码每项的系数,规则如下:

(a)若 R_1、R_2 指向自身,该叶节点不表示任何系数;

(b)若 R_1 指向自身,R_2 指向邻近节点,则系数为 0;

(c)若 R_1 指向其他叶节点,R_2 指向邻近节点,从 R_1 指向的

叶节点开始到原叶节点,经过的叶节点个数为 n,则系数为 n。

c)向 info 域添加信息,规则如下:

(a)将 w'_i 和基数 k_i 添加到 origin 节点的 info 域;

(b)对水印信息 W 进行 hash 处理,hash 函数可使用 SHA1、MD5 等算法,得到消息摘要 W_h;

(c)将 W_h 添加到其余节点的 info 域。

通过新添加的指针域回指,使 DPPCT 结构具有双向循环链表的特点,如果改变某个非叶节点的指针或添加、删除某个非叶节点,通过指针回指技术,都可以将该节点还原。对于叶节点,如果改变了某个叶节点的 L、R_2 指针,对系数的编码影响不大,如果改变了叶节点的 R_1 指针,将造成系数编码错误。为此,可构造大于 k_i 个叶节点的 DPPCT 结构来混淆攻击者。同时,通过向 info 域添加 hash 处理后的水印信息 W_h,使得除 origin 节点外每个节点都含有 W_h,即使攻击者破坏了其中一个节点,其他节点仍然会含有 W_h。

假设 w'_i 表示的水印数为 12,基数 k_i 为 3,则展开式为:$w'_i = 12 = 0 \times 3^0 + 1 \times 3^1 + 1 \times 3^2$。其 DPPCT 结构如图 3 所示。

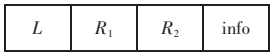

图 2 DPPCT 节点的结构

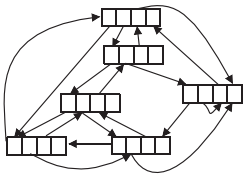

图 3 具有三个叶节点的 DPPCT 拓扑图结构

2.4 混沌加密

目前,较为实用的混沌密码算法是将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。本文将混沌序列与 AES 加密算法相结合,构造具有可变长密钥和交错保护机制的混沌加密方案,抵御已知明文的差分和线性攻击,对水印及敏感代码实施保护,并结合非敏感代码的哈希值,实现水印系统的防窜改功能[11]。混沌加密过程如图 4 所示。

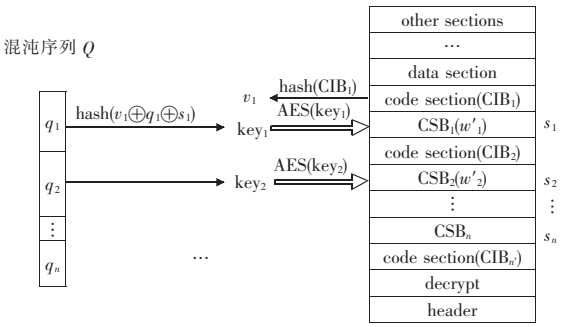

图 4 混沌加密过程

算法流程如下:

a)首先将待加密的敏感代码段标记为(CSB_1,CSB_2,…,CSB_n),非敏感代码段标记为(CIB_1,CIB_2,…,$CIB_{n'}$),其中 CSB_i 可能包含水印信息 w'_j,$n \leqslant n'$;

b)由混沌系统产生混沌序列 $Q = (q_1, q_2, q_3, \cdots, q_n)$;

c)计算 CSB_i 的起始地址 $address_1$ 和 $address_2$,CIB_i 的起始地址 $address_3$ 和 $address_4$;

d)使用 hash 算法,计算 $address_1$ 和 $address_2$ 之间代码段的校验和 $s_i = hash(address_1, address_2)$,$address_3$ 和 $address_4$ 之间代码段的校验和 $v_i = hash(address_3, address_4)$;

e)由公式 $Key_i = hash[v_i \oplus q_i \oplus s_i]$ 计算加密密钥 Key_i,并保存 q_i、s_i 以用于解密;

f)对 CSB_i 进行加密,$AES(CEB_i, Key_i)$。

在程序执行过程中,遇到密文 CSB_i 代码块时,调用 Decrypt 解密函数,对 CSB_i 进行解密,然后执行明文代码块 CSB_i。加密过程中,使用混沌序列 q_i 作为参数,具有可变长密钥,可有效增加攻击者的破解难度,使用 CIB_i 的 hash 值作为参数,可有效防止整个程序被攻击者窜改。程序运行结束后,再次调用加密模块,完成对敏感代码段的保护。

3 性能分析

3.1 鲁棒性分析

动态图软件水印的鲁棒性主要体现在拓扑图结构的抗攻击性。DPPCT 拓扑图存储在动态建立的堆栈结构中,而且可存在多种图结构,因此很难定位水印。同时由于变化的图拓扑结构,增强了水印的抗模式匹配和共谋攻击的能力。其次,通过新添加的指针域和 info 域,使 DPPCT 结构具有双向循环链表的特点,并使得除 origin 节点外每个节点都含有水印信息,进一步提高了 DPPCT 的鲁棒性和安全性。DPPCT 结构与其他编码方案的抗攻击性比较如表 1 所示。最后,通过混沌系统对水印进行混沌置乱和混沌加密,利用混沌序列良好的伪随机性和自相关特性产生密钥,抵御已知明文的差分和线性攻击。如果破解二维混沌参数 μ,需要该混沌两个关联值对,对于具有 n 个混沌序列值的混沌系统,其破解概率为 n^{-2}。同时,利用敏感代码段和非敏感代码段的 hash 值,加密敏感代码段,形成交错保护机制,保护全部代码,增强水印系统的防窜改能力。如果程序中的某代码段被修改,将造成解密失败,程序终止运行。

表 1 三种编码方案的抗攻击性分析比较

编码方案	添加、剪裁攻击	扭曲攻击	容错性
基数 K	弱	弱	一般
PPCT	强	强	强
DPPCT	强	较强	较强

3.2 数据率分析

DPPCT 结构结合了基数 K 编码和 PPCT 编码的特点,有 $2n$ 个节点的 DPPCT 可编码的范围为 $0 \sim n^{n-1} - 1$。在节点数一定时,基数 K、PPCT、DPPCT 三种编码方式的数据率对比,结果如表 2 所示。

表 2 不同编码方式数据率分析比较

节点数	基数 K	PPCT	DPPCT
n	$n^{n-1} - 1$	$2C_{n-2}^{n/2-1}/n$	$(n/2)^{n/2-1} - 1$
10	1.00×10^9	1.40×10^1	6.25×10^2
20	5.24×10^{24}	4.86×10^3	1.00×10^9
50	1.78×10^{83}	1.29×10^{12}	3.55×10^{33}

由表 2 可以看出,采用 DPPCT 编码方式所表示的整数范围介于基数 K 和 PPCT 编码之间。

3.3 性能过载分析

水印嵌入后,肯定会对程序的性能产生影响,主要表现为空间过载和时间过载。实验利用 SandMark 平台对 TTT. jar 程序处理,嵌入不同水印数,并分析不同水印数下,对原始程序大小及运行时间的影响,结果如图 5、6 所示。由图 5 可以看出,水印分存后,程序大小增加缓慢。这是由于对水印信息的矩阵分割,使分存水印数呈线性增加,并使用数据率较高的 DPPCT 编码,减小了程序负载。由图 6 可以看出,水印嵌入后,程序运行时间并没有受到明显影响。这是由于嵌入的水印代码并不参与程序主要功能模块的运算,且只对程序的敏感代码段进行混沌加密,并不会明显增加程序的整体运行时间。 (下转第 3429 页)

相似域名具有明显的迷惑性,表明了本方法可以较准确地衡量中文域名的仿冒程度。

5　结束语

本文分析了当前国际化域名的仿冒攻击问题,并总结了中文域名可能出现的仿冒攻击的具体形式。针对最普遍也是最难检测的一种仿冒攻击方式,本文提出了一种基于中文字符点阵和向量空间的方法,对单个中文字符间的相似性进行计算,并以此为基础,提出了一种衡量中文域名整体相似性的计算方法。通过使用实际中文字库数据的实验,证明了本文提出的方法的有效性。需要指出的是,本文的方法从原理上来讲并没有依赖于任何中文字符独有的特性,因此本方法的原理是可以适用于任何国际化域名的仿冒攻击检测问题的。在未来的工作中,本文将通过大量的人工辨别实验,确定判定域名仿冒的阈值取值方法和数值范围,并且针对文中提到的其他中文域名攻击方式以及多种方式结合使用的情况,提出整体性的解决方案,以更好地解决中文域名仿冒攻击的问题,并且在整个 Unicode 字符的范围内提升和改进现有的相似性检测方法。

参考文献:

[1] Wikipedia. IDN homograph attack[EB/OL]. (2013-01-27)[2013-01-28]. http://en. wikipedia. org/wiki/IDN_homograph_attack.

[2] 维基百科. 钓鱼式攻击或网络钓鱼[EB/OL]. (2012-11-28)[2013-01-28]. http://zh. wikipedia. org/wiki/% E7% BD% 91% E7% BB% 9C% E9% 92% 93% E9% B1% BC.

[3] 中国电子商务协会. 2012 年中国网站可信验证行业发展报告[EB/OL]. http://ectrust. knet. cn/column _ 2/201207/W020120704645636974021. pdf.

[4] 中国反钓鱼网站联盟[EB/OL]. [2013-01-28]. http://apac. cn/.

[5] Anti-phishing Working Group. Global phishing survey 1H2012[EB/OL]. http://docs. apwg. org/events/sessions/ecrime2012/APWG _ GlobalPhishingSurvey1H2012. pdf.

[6] FALTSTROM P. RFC 3490, International domain names in applications(IDNA)[S]. 2003.

[7] 中国互联网络信息中心. 第 31 次中国互联网络发展状况统计报告[EB/OL]. [2013-01]. http://www. cnnic. cn/hlwfzyj/ hlwxzbg/hlwtjbg/201301/P020130122600399530412. pdf.

[8] 郭思华. 商标中的语言文字仿冒[J]. 语文教学与研究, 2007(7): 62-63.

[9] HONG Bo, WANG Wei, WANG Li-ming, *et al*. A hybrid system to find & fight phishing attacks actively[C]//Proc of the 10th IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology. Washington DC: IEEE Computer Society, 2011.

[10] FU A Y, ZHANG Wan, DENG Xiao-tie, *et al*. Safeguard against unicode attacks: generation and applications for UC-Simlist[C]//Proc of the 15th International World Wide Web Conference. New York: ACM Press, 2006: 917-918.

[11] KRAMMER V. Phishing defense against IDN address spoofing attacks[C]//Proc of International Conference on Privacy, Security and Trust. New York: ACM Press, 2006: 275-284.

[12] MoziliaZine. Network IDN blacklist chars[EB/OL]. (2009-03-01)[2013-01-28]. http://kb. mozillazine. org/Network. IDN. blacklist_chars.

[13] 百度百科. 蔡戈尼效应[EB/OL]. (2013-01-06)[2013-01-28]. http://baike. baidu. com/view/1302263. htm? fromId = 494914.

(上接第 3420 页)

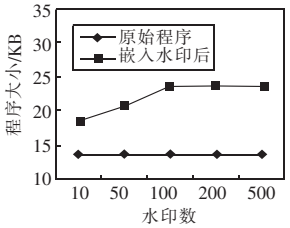

图 5　嵌入水印后程序大小改变情况

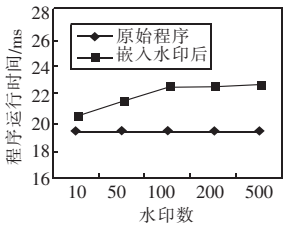

图 6　嵌入水印后程序运行时间改变情况

3.4　抗逆向工程分析

DPPCT 拓扑图结构隐藏在动态建立的堆栈结构中,可防止静态逆向工程的攻击。在利用混沌加密后,对分存水印进行混沌加密,设嵌入分存水印数为 m,可嵌入地址数为 n,混沌状态为 χ。当可嵌入地址 n 不变时,要破解所有水印的时间复杂度为 $\delta_1 = Tm\chi$,其中 T 为常数,随着水印数 m 的增大,复杂度呈线性增加,但增加了程序负载,性能影响退化。如果嵌入的分存水印数 m 不变,破解所有水印的时间复杂度为 $\delta_2 = TmC_n^m\chi$,当嵌入地址数 n 增大时,则复杂度呈指数增加,但保护强度会减小。以上两者都依赖于混沌状态 χ 是可知的,如果 χ 不可知,则不能破解。同时,要在程序性能退化和保护强度之间折中。

4　结束语

本文下一步研究如何结合软件特征,构造更加完善的水印拓扑图结构,并利用混沌系统及软件防窜改技术,构造更加实用的水印防窜改体系。

参考文献:

[1] COLLERG C, THOMBORSON J, TOWSEND G M. Dynamic graph-based software watermarking, TR04-08[R]. Pennsylvania: Pennsylvania State University, 2004.

[2] COLLBERG C, NAGRA J. Surreptitious software: obfuscation, watermarking, and tamperproofing for software protection[M]. [S. l.]: Addison-Wesley, 2009: 526-528.

[3] 白雪梅,凌捷. 基于中国剩余定理的动态水印方案[J]. 计算机工程, 2006, 32(16): 155-157.

[4] 赵彦锋. 基于 Asmuth-Bloom 体系的动态图水印实现方案[J]. 现代电子技术, 2011, 34(5): 125-128.

[5] 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(2): 351-360.

[6] LIU Fen-lin, LU Bin, LUO Xiang-yang. A chaos-based robust software watermarking, information security practice and experience[C]// Lecture Notes in Computer Science, vol 3903. 2006: 355-366.

[7] 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.

[8] 李淑芝,王显珉. 基于 m-n 变进制规则的动态图软件水印算法[J]. 计算机工程, 2012, 38(21): 17-21.

[9] 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(10): 840-847.

[10] 王慧娇,沙宗鲁,轩爱成. 基于 PPCT 和基数 K 的动态图混合编码方案[J]. 计算机工程与应用, 2010, 46(25): 109-111.

[11] TANG Zhan-yong, FANG Ding-yi. A tamper-proof software watermark using code encryption[C]//Proc of IEEE International Conference on Intelligence and Security Informatics. 2011: 156-160.