Journal of Computer Applications
计算机应用,2013,33(9):2525 -2528
ISSN 1001-9081
CODEN JYIIDU
2013-09-01
http://www.joca.cn

文章编号:1001-9081(2013)09-2525-04
doi:10.11772/j.issn.1001-9081.2013.09.2525

基于分存策略的软件保护博弈模型

王　蕊*,杨秋翔,陈够喜,马巧梅
(中北大学 电子与计算机科学技术学院,太原 030051)
(*通信作者电子邮箱 wangruinuc@126.com)

摘　要:软件保护技术普遍是通过完善代码和应用加密方案来达到保护软件的目的。针对软件代码的静态授权抗攻击能力以及软件加密的加密强度是否足够抵抗攻击的问题,提出一种基于分存策略的软件保护博弈模型。该模型采用分存策略对密钥进行分段,得到多个检验与抵抗软件破解者攻击的验证函数,把它们隐藏在程序中,使得软件运行时有多个不同的验证函数对程序进行保护。从博弈论的角度分析论证该模型,并将其应用于软件注册码验证的实例中,提高了软件代码的安全性。实验结果和分析表明了该模型的正确性和有效性。

关键词:软件保护;博弈模型;分存;游击战术;注册码验证
中图分类号:TP309.2;TP301.5　　**文献标志码**:A

Software protection game model based on divided-storage strategy

WANG Rui*, YANG Qiuxiang, CHEN Gouxi, MA Qiaomei
(*School of Electronics and Computer Science & Technology, North University of China, Taiyuan Shanxi* 030051, *China*)

Abstract: Current software protection technologies generally achieve the software protection through improving the code and applying encryption scheme. To address the problem of whether the static authorized anti-attack capability of software code and the strength of the software encryption can sufficiently resist attack, the authors proposed a software protection game model based on divided-storage strategy. The strategy of divided-storage was used by the model to divide secret key into many segments, so multiple verified functions that were used to inspect and resist the cracker's attack were received. After being hidden in the program, the program was protected by multiple different verified functions during the running of the software. The model was analyzed and demonstrated from the perspective of game theory, also applied to the instances of software registration code verification. The security of the software code had been improved. The experimental results and analysis show that the proposed model is correct and effective.

Key words: software protection; game model; divided-storage; guerrilla tactics; registration code verification

0　引言

随着计算机软件的广泛应用,软件安全性问题日益突出,软件保护受到了越来越多的重视。为防止软件破解者对软件代码的非法执行、非法篡改、静态分析、动态跟踪等,必须采取有效的软件保护技术。目前提出的软件保护方案,基于代码混淆和软件水印的保护方案[1-2],从重构程序结构并隐藏关键代码的角度增加破解者理解软件的难度,抗攻击性较强,文献[3]提出的基于对象的软件行为模型,能够有效地检测破解者的攻击,但是增加了软件的额外开销,基于软件加密的硬保护方案[4-5],将软件与硬件配套使用,适用于带有相应硬件设备的机器,防止了非法扩散,文献[6]提出的基于中间件的可信软件保护模型,利用可信平台模块保护软件,但是存在成本过高以及硬件环境转移的问题,基于软件加密的软保护方案[7-9],通过适当的加密算法实现软件加密,保证了算法的隐蔽性,文献[10]提出的基于动态证书副本的软件版权保护模型,通过电子证书保证用户的合法性,但是无法确定其加密强度是否足够抵抗攻击,同时,由于这种软件保护方案为预先静态授权,所以也无法确定用户获得授权后的非法操作。

针对加密强度和静态授权的不足,本文提出一种基于分存策略的软件保护博弈模型。该博弈模型采用分存的策略,使用游击战术来保护软件代码,能够对软件破解者的攻击进行动态的验证与防范。此模型在用户静态授权后,可随时判断用户后续操作是否合法,还可同时使用各种加密算法保证验证函数的安全性,不依赖于任何特殊的硬件设备,成本较低并且易于实现。

1　软件保护博弈模型

如果说软件保护是软件开发者与软件破解者之间进行的一场全面战争,那么可以应用具有竞争或对抗性质的博弈行为[11]在一定程度上反映破解者对软件的攻击效果与开发者对软件的保护效果之间的关系,建立软件保护博弈模型(Software Protection Game Model,SPGM)。

定义 1　软件保护博弈模型 SPGM。表示为四元组$(S_j, Q_i, L(\cdot), p_i(\cdot))$,满足下列条件:

1)$S_j = \{s_i \mid i = 1,2,\cdots,n\}$,$j \in \mathbf{N}$,描述战术空间纯策略组合的集合,$s_i$ 表示博弈者 i 选择的战术。

2)$Q_i = \{q_i \mid i = 1,2,\cdots,n\}$,描述博弈者自身的私人类型

收稿日期:2013-03-13;**修回日期**:2013-05-21。
基金项目:山西省科技攻关项目(20090322004);中北大学科学研究基金资助项目(2012)。
作者简介:王蕊(1989-),女,北京人,硕士研究生,CCF 会员,主要研究方向:信息安全、软件保护;　杨秋翔(1969-),男,山西临汾人,教授,主要研究方向:信息安全、网络拓扑;　陈够喜(1966-),男,山西太谷人,副教授,博士,主要研究方向:信息隐藏、图像处理;　马巧梅(1969-),女,山西灵石人,副教授,博士,主要研究方向:网络信息安全、图形图像处理。

的集合，$\boldsymbol{q}_i$ 包含了博弈者 i 所有的不确定信息，i 的纯策略就是一个从 Q_i 到可选策略集合 F_i（局势）的函数，即 $\boldsymbol{s}_i: Q_i \to F_i$。

3）$L(\cdot): Q \to [0,1]$，$Q = Q_1 \times Q_2 \times \cdots \times Q_n$，描述所有类型出现的联合累积分布。

4）$p_i(\cdot) = \{p_i(s_1(q_1), s_2(q_2), \cdots, s_n(q_n)), i = 1,2,\cdots,n\}$，描述在特定的战术组合下博弈者 i 的效用函数，是自己与对手的确定行动以及自身类型的函数。

定义 2 二人零和博弈控制函数。设 $W \subseteq \mathbf{R}^P$，$V \subseteq \mathbf{R}^Q$，其中 W、V 是有界非空闭集，并设 $w = w(t): [0, T_0] \to W$ 是可测函数，称为博弈者甲的控制函数；$v = v(t): [0, T_0] \to V$ 也是可测函数，称为博弈者乙的控制函数。集合 W，V 分别称为甲、乙的控制集。

定义 3 偏好关系。设非空集合 $C = \{\boldsymbol{c}_i \mid i = 1,2,\cdots,n\}$ 是定义于 $\mathbf{R}_+^N$ 上的博弈者所有可选结果的集合，如果对于 C 中的任意两种结果存在关系：$c_1 \succ c_2$，就可认为博弈者对结果 c_1 的评估偏好于 c_2。

由于博弈者效用函数 $p: C \to \mathbf{R}$ 的作用是给不同结果赋效用值，即 $\boldsymbol{c}_i$ 的效用值为 $p(\boldsymbol{c}_i)$，所以 $p(\cdot)$ 就能够反映出偏好关系 $\succ$。于是博弈者更偏好的结果总能被 $p(\cdot)$ 赋一个更高的值，即 $c_1 \succ c_2 \Leftrightarrow p(c_1) \geqslant p(c_2)$。

2 基于分存策略的软件保护博弈模型

2.1 分存策略

博弈的结果决定了每个博弈者的得与失，而取胜博弈者所采用的策略是决定该博弈结果的关键。分存是将一个单载体分解出的若干个多载体进行传输或存储的技术，为了增强软件代码的安全性，SPGM 采取分存策略。

设 $f(x) \in [0,1]$，$n \in \mathbf{Z}^+$，Bernstein 多项式函数 $B_n(x)$[12] 定义为：

$$B_n(x) = B_n(f(x)) = \sum_{k=0}^{n} \binom{n}{k} f\left(\frac{k}{n}\right) x^k (1-x)^{n-k} \quad (1)$$

其中：$B_n(x)$ 为 Bernstein 多项式第 n 次算子，$\binom{n}{k} = \frac{n!}{k!(n-k)!}$。

于是，可推导为：

$$B_n(x) = B_n(f(x)) = \sum_{k=0}^{n} f\left(\frac{k}{n}\right) B_k^n(x) \quad (2)$$

设有参数曲面：$\Lambda: f = f(u,r)$，并对式(2) $B_n(x)$ 进行扩展，则二维乘积型 Bernstein 多项式函数[13] 又可表示为：

$$B_{n_1,n_2}(u,r) = \sum_{i=0}^{n_1} \sum_{j=0}^{n_2} f\left(\frac{i}{n_1}, \frac{j}{n_2}\right) B_i^{n_1}(u) B_j^{n_2}(r);$$
$$0 \leqslant u, r \leqslant 1 \quad (3)$$

定理 1 设给定 $D = [0,1] \times [0,1]$，$f(u,r) = \{f_{i,j} \mid i = 0,\cdots,n_1; j = 0,\cdots,n_2\}$，两组 Bernstein 多项式基函数可构成 Bézier 曲面，则在 D 上有 $n_1 \times n_2$ 次 Bernstein-Bézier 曲面 $B(u,r)$ 定义为：

$$B_{n_1,n_2}(u,r) = \sum_{i=0}^{n_1} \sum_{j=0}^{n_2} f_{i,j} B_i^{n_1}(u) B_j^{n_2}(r); 0 \leqslant u, r \leqslant 1 \quad (4)$$

其中：$B_k^n(x) = \binom{n}{k} x^k (1-x)^{n-k}$ 为 n 次第 k 项 Bernstein 多项式基函数。

利用 Bernstein 多项式对载体进行分存[14]，把密钥按式(1)分解为加密算法不同的若干密钥段，进而推导出式(2)、式(3)，然后根据用户与这些密钥段的映射关系，运用式(4)构造出不同的空间曲面参数方程，并且把这些参数方程看作是验证函数 F，进而在程序中实现分存。隐秘密钥分解和提取算法均为可逆算法[15]。

由于构造出多个互不相同且个数不确定的 F_i，所以通过任意一个 F_i 的验证都只是用户合法的必要条件，而非充分条件，真正合法的密钥必须通过所有 F_i 的验证。即使破解者找到 F 中的任意几个，也无法确定 F 总共有多少个，也无法求出密钥中的任何一段内容。即使破解者获得了所有的 F，也必须要有精深的数学水平才能求出密钥。为防止破解者通过 F 获取密钥，可对 F_i 使用不同的加密算法，增强加密的强度，尽可能地保证验证函数的安全性。

2.2 模型的建立

根据分存的特点，将分存形象化地应用于 Lanchester 数学战争模型[16] 中，使博弈者采用游击战术来保护软件代码。基于分存策略的 SPGM 建立如下：

假设 $S_1 = \{s_1, s_2\} = \{$游击战术，常规战术$\}$。软件开发者为甲方，采用的是游击战术；乙方作为软件破解者，采用的是常规战术。甲方的任务是保护用户提取密钥的合法过程，防止乙方对隐秘密钥进行破解，主要的防护手段就是利用分存策略来保护软件代码，不让乙方确定验证函数的个数和用户与密钥之间的对应关系，Q 即为隐秘密钥分存后的集合（不确定信息）。乙方的任务就是破解密钥，不断攻击甲方的验证函数。具体细节假设：

1）甲方的战斗损伤率是 $\boldsymbol{F}_i$ 的被攻击率，取决于 $\boldsymbol{F}_i$ 的个数和抗攻击率（抵抗每个破解者单位时间攻击的次数），用 $h(s_1, s_2)$ 表示；乙方的战斗损伤率可看作是甲方的防守率（每个 F 单位时间被保护的次数），用 $g(s_1, s_2)$ 表示。

2）每一方的非战斗损伤率只与本方的战斗力成正比。

3）甲乙双方的增援率是可依赖于有效的加密或解密技术，分别用 $m(t)$ 和 $n(t)$ 表示。

由此可以写出甲乙双方在 t 时刻的战斗力 $s_1(t)$、$s_2(t)$ 的关系模型：

$$\begin{cases} s_1'(t) = -h(s_1, s_2) - \alpha s_1 + m(t), \alpha > 0 \\ s_2'(t) = -g(s_1, s_2) - \gamma s_2 + n(t), \gamma > 0 \end{cases} \quad (5)$$

定理 2 在一个软件保护博弈 $(S_j, Q_i, L(\cdot), p_i(\cdot))$ 下，对于所有的博弈者 $i (i = 1,2,\cdots,n)$，$\exists\ \boldsymbol{q}_i, \boldsymbol{q}_i \in Q_i$，$\forall\ \boldsymbol{s}'_i$，$\boldsymbol{s}'_i \in S_j$，都有：

$$p_i(\boldsymbol{s}_i(\boldsymbol{q}_i)) \geqslant p_i(\boldsymbol{s}'_i(\boldsymbol{q}_i)).$$

证明 在甲和乙的软件保护博弈中，甲采用游击战术，乙采用常规战术，当这种纯策略的结果 c_1 偏好于其他策略的结果 $\boldsymbol{c}_i$ 时，应满足下式：

$$p(c_1) \geqslant p(\boldsymbol{c}_i) \Leftrightarrow c_1 \succ \boldsymbol{c}_i;\ i = 2,3,\cdots,n$$

由式(5)可以看出，软件开发者和破解者其中任一方所得即为另一方所失，$s_1(t)$ 和 $s_2(t)$ 都是时间 t 的函数，二人零和博弈进行的过程是一个连续的过程，于是采用离散序列法将连续问题离散化。由定义 2 给出的控制函数，可知二人零和微分博弈的一般形式：

对任意正整数 n，令 $d = T_0/n$，于是，区间 $[0, T_0]$ 被分成了 n 段长度为 d 的小区间 $I_1 = [0, d]$，$I_i = [(i-1)d, id]$ $(i = 2, 3, \cdots, n)$。记 w_i、v_i 为 w、v 在 I_i 上的限制，W_i、V_i 为 W、V 在 I_i 上的限制。

若甲在区间 $I_i(i=1,2,\cdots,n)$ 上采取行动,

$\Pi^{d_i}: V_1 \times W_1 \times V_2 \times W_2 \times \cdots \times V_{i-1} \times W_{i-1} \times V_i \to W_i$

则称 $\Pi^d = (\Pi^{d_1}, \Pi^{d_2}, \cdots, \Pi^{d_n})$ 为甲的游击战术;若乙在区间 $I_i(i=1,2,\cdots,n)$ 上采取行动,

$\Sigma^{d_i}: W_1 \times V_1 \times W_2 \times V_2 \times \cdots \times W_{i-1} \times V_{i-1} \times W_i \to V_i$

则称 $\Sigma^d = (\Sigma^{d_1}, \Sigma^{d_2}, \cdots, \Sigma^{d_n})$ 为乙的游击战术;若甲在 I_i 上采取行动,

$\Pi_{d_i}: W_1 \times V_1 \times W_2 \times V_2 \times \cdots \times W_{i-1} \times V_{i-1} \to W_i$

则称 $\Pi_d = (\Pi_{d_1}, \Pi_{d_2}, \cdots, \Pi_{d_n})$ 为甲的常规战术;若乙在 I_i 上采取行动,

$\Sigma_{d_i}: V_1 \times W_1 \times V_2 \times W_2 \times \cdots \times V_{i-1} \times W_{i-1} \to V_i$

则称 $\Sigma_d = (\Sigma_{d_1}, \Sigma_{d_2}, \cdots, \Sigma_{d_n})$ 为乙的常规战术。

显然,在甲乙二人零和微分博弈过程中,有以下三种可能策略:甲采用游击战术,乙采用常规战术,结果为 c_1;甲乙均采用常规战术,结果为 c_2;某阶段甲占优势,某阶段乙占优势,剩下阶段双方均不占优势。由于不可能在同一时刻双方均占优势,所以双方不可能同时采用游击战术,只需判断前两种策略的结果关系 $c_1 > c_2$ 是否成立。

策略 1　甲方采用游击战术,将 F_i 隐藏在程序代码 s_x 中,乙方向这个代码区域进行攻击,并且不知道攻击情况。这时甲方战斗损伤率不仅与乙方攻击率有关,还与甲方 F_i 有关。简单地设 $h = as_1s_2$,a 表示乙方平均每次攻击对甲方 F_i 的破解率,称有效战斗系数。可表示为 $a = r_y p_y = r_y s_{ry}/s_x$,其中 r_y 是乙方的攻击率,p_y 是每次攻击的命中率,等于乙方一次攻击的有效代码行数 s_{ry} 与甲方程序代码总行数 s_x 之比。

乙方采用常规战术,处于甲方的防范范围之内,一旦乙方的攻击被甲方防守住,甲方的防守力还可集中在其余攻击上,所以乙方的战斗损伤率只与甲方的防守有关。简单地设 $g = bs_1$,b 表示甲方的有效战斗系数,可表示为 $b = r_x p_x$,其中 r_x 是甲方的防守率,p_x 是每次被攻击后防守的命中率。于是,根据式(5)列出本策略的战争模型方程:

$$\begin{cases} s_1' = -as_1s_2 - \alpha s_1 + m(t) \\ s_2' = -bs_1 - \gamma s_2 + n(t) \end{cases} \tag{6}$$

由于 $\alpha s_1 \ll h(s_1, s_2)$,$\gamma s_2 \ll g(s_1, s_2)$,所以分析时忽略非战斗损伤率一项,并且假设双方都没有增援。记双方的初始战斗力分别是 $s_1(0) = s_{10}$ 和 $s_2(0) = s_{20}$,于是,式(6)的解为:

$$as_2^2 - 2bs_1 = n \tag{7}$$

乙方取胜的条件(策略 1 的结果 c_1)可推导为:

$$(s_{20}/s_{10})^2 > 2b/as_{10} = 2r_x p_x s_x / r_y s_{ry} s_{10} \tag{8}$$

假定破解者乙方攻击力较强,以游击战术作战的甲方虽防守率较弱,但 F_i 活动的区域较大。利用式(8)可以估算出乙方为了取胜需投入多大的初始战斗力。为确定起见,不妨设甲方 F_i 的个数 $s_{10} = 10$,防守命中率 $p_x = 0.1$,防守率 r_x 是乙方攻击率 r_y 的一半,程序代码的总行数为 $s_x = 10^6$,乙方每次攻击的程序有效代码的行数为 $s_{ry} = 100$,那么由式(8),乙方的攻击必须 10 倍于甲方的防守才可取胜。

策略 2　甲乙双方都采用常规战术作战,只需分析一方的战斗损伤率即可。于是,可将式(5)轻易化为:

$$\begin{cases} s_1' = -ds_2 - \alpha s_1 + m(t) \\ s_2' = -bs_1 - \gamma s_2 + n(t) \end{cases} \tag{9}$$

忽略非战斗损伤率并设 $m = n = 0$,在初始条件下,式(9)的解为:

$$ds_2^2 - bs_1^2 = k \tag{10}$$

进一步分析乙方取胜的条件(策略 2 的结果 c_2)为:

$$(s_{20}/s_{10})^2 > b/d = r_x p_x / r_y p_y \tag{11}$$

双方初始战斗力之比以平方关系影响着战争的结局。若甲方的战斗力比如防守率 r_x 增加到原来的 4 倍(p_x、r_y、p_y 均不变),那么为了与此相抗衡,乙方只需将初始战斗力 s_{20} 增加到原来的 2 倍。

分析上述两种策略,通过对式(8)和式(11)表示的策略结果进行举例比较,得到 $c_1 > c_2$。因此,在软件开发者与软件破解者之间进行的软件保护博弈中,开发者采用游击战术对抗攻击,可以增加破解者的攻击难度,提升了软件的安全性。

证毕。

3　实验验证与分析

软件保护的目的是向合法用户提供完整的功能,所以软件保护必然包括验证用户合法性的环节,而这一环节通常采用注册码验证的方式实现。注册码验证过程如下:首先用户向软件开发者申请注册并提交用户码 U;然后软件开发者计算出注册码 $R = f(U)$ 返回给用户;接着用户便可在注册界面输入 U 和 R;最后软件通过验证函数 $F(U,R) = f^{-1}(R) - U$ 的值是否合法来判定用户的合法性。为了保护注册码验证的安全性,防止破解者通过验证函数 F 推导出注册机 f,就必须提出有效的方案来防范算法求逆。具体方案的一般步骤为:

Step1　将 R 分存成多段 R_i。

Step2　构造不同的 f 映射,使得:$R_i = f_i(U)$。

Step3　构造空间曲面参数方程 $F_i(U, R_i)$,使得:$F_i = f_i^{-1}$。

Step4　将 F_i 使用不同的加密算法,使软件达到足够的加密强度。例如,使 F_1 使用 AES 算法,F_2 使用 RSA 算法,F_3 使用 DSA 算法,F_4 使用 ECC 算法等。

该基于分存策略的软件保护博弈模型,使验证函数 F_i 的个数不确定,并且使破解者无法了解用户码 U 和注册码 R 的对应关系。在用户输入 R 后仅仅使用 F_1 进行验证,并将 R 以密文形式写入自定义格式的数据文件中,如果验证通过就显示注册成功。另外的 F 隐藏在程序中,只有用户后续执行特定的操作时才被调用。一旦任何一个 F 发现 R 非法,就清除 R 并将软件恢复为未注册状态。

为了验证基于分存策略的软件保护博弈模型在软件注册码验证实例中的正确性和有效性,对该模型中的两种策略分别进行分析。模型仿真在 Intel Xeon W3503 2.4 GHz CPU, 6 GB RAM, Windows Server2008, Matlab 8.0 实验环境中完成。策略 1 由式(7)确定的相轨线是抛物线族,如图 1 所示。

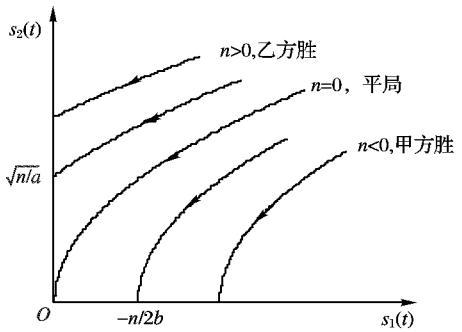

图 1　策略 1 模型的相轨线

图 1 中的箭头表示随时间 t 的增加，$s_1(t)$、$s_2(t)$ 的变化趋势。当 $n>0$ 时，$\exists\ t_1, s_2(t_1)_{\min} = \sqrt{n/a}$，$s_1(t_1)=0$，说明当乙方的战斗力减少到 $\sqrt{n/a}$ 时，甲方战斗力为零，则乙方取胜；当 $n<0$ 时，$\exists\ t_2, s_1(t_2)_{\min} = -n/2b, s_2(t_2)=0$，说明当甲方的战斗力减少到 $-n/2b$ 时，乙方战斗力为零，则甲方取胜；当 $n=0$ 时双方战平。

策略 2 由式(10)确定的相轨线是双曲线族，如图 2 所示。

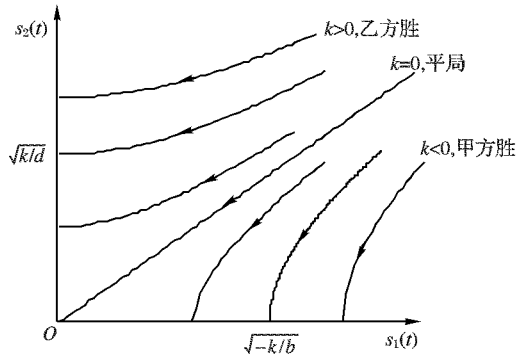

图 2 策略 2 模型的相轨线

图 2 中的箭头表示随时间 t 的增加，$s_1(t)$、$s_2(t)$ 的变化趋势。当 $k>0$ 时，$\exists\ t_1, s_2(t_1)=\sqrt{k/d}, s_1(t_1)=0$，说明乙方获胜；同理，当 $k<0$ 时，说明甲方获胜；当 $k=0$ 时双方战平。

通过分析两种相轨线的趋势，并对两种策略的结果做数值实验，对比的实验数据如表 1～2 所示。

表 1 两种策略攻击复杂度对比实验数据(s_x 变化)

程序代码千行数	攻击所需次数	攻击所需代码千行数	攻击复杂度(倍数) 策略一	策略二
10	100	10	1.0	0.7
40	800	80	2.0	
100	3 200	320	3.2	
400	25 200	2 520	6.3	
10^3	10^5	10^4	10.0	

表 2 两种策略攻击复杂度对比实验数据(s_{ry} 变化)

每次攻击有效代码千行数	所需总有效代码千行数	攻击复杂度(倍数) 策略一	策略二
0.1	10^3	10.0	0.7
1	10^4	3.2	
5	5×10^4	1.4	
10	10^5	1.0	

假定程序在运行时每次读取 100 行。从表 1 中的数据可以看出：随着程序代码的行数增加(s_x 增大)，意味着可隐藏验证函数 F_i 的空间增大，能够促使开发者生成更多个不确定的 F_i 隐藏在程序内部，进而导致破解者攻击程序所需的次数不断增多，攻击的范围逐渐扩大，攻击 F_i 的难度持续增大，使用策略一的攻击复杂度大于使用策略二。而表 2 表示，通过破解者攻击有效代码行数的增加(s_{ry} 增大)，导致攻击复杂度逐渐下降，但是使用策略一同样保持着优于策略二的效果。

实验数据表明：软件破解者欲想成功，其所需的攻击必须多倍于软件开发者的防守。软件开发者运用游击战术抵抗攻击，导致软件破解者所需的攻击复杂度的倍数比其运用常规战术策略的大，效果更优。

由于对软件保护而言，没有绝对安全的保护方法，当破解软件的代价大于通过正当途径获得合法版权软件的代价时，就可认为该方法是安全的。于是采用了游击战术之后的软件保护方法可以提升软件的安全性，有效地防止软件被攻击。基于分存策略的软件保护博弈模型可以在许多领域的基本验证系统中实现。

4 结语

现今的软件保护技术日趋成熟，需对其不断加入新的理解。本文用抽象博弈模型作为统一的框架来表示软件保护方案，并模型化软件开发者的保护与破解者的攻击两者之间的关系，在基于一定的假设条件下提高了软件的保护强度，并进行了相应的实验与分析。

本文应用分存技术对软件密钥实施多层保护，加大破解者生成非法密钥的难度，增加软件的抗攻击能力，运算量相对不大，同时动态产生的验证函数采用成熟的加密算法，不依附于硬件设施。不过本模型有一个约束条件，不适于小型软件代码的保护，并且仍旧缺乏对博弈模型从形式语义的角度进行有效性证明，还需要在今后作进一步的研究。

参考文献：

[1] 王建民，余志伟，王朝坤，等. Java 程序混淆技术综述[J]. 计算机学报，2011，34(9)：1578－1588.

[2] 许金超，曾国荪. 一种基于线程关系的软件水印算法[J]. 电子学报，2012，40(5)：891－896.

[3] 傅建明，陶芬，王丹，等. 基于对象的软件行为模型[J]. 软件学报，2011，22(11)：2716－2728.

[4] 朱勤，陆志明. 基于信息隐藏的外包数据库版权保护系统[J]. 计算机科学，2010，37(1)：163－166.

[5] 郝耀辉，刘洪波，郑礼，等. 基于 USB 加密锁的软件防盗版方法[J]. 计算机工程，2010，36(23)：119－123.

[6] 蔡增玉，甘勇，刘书如，等. 一种基于中间件的可信软件保护模型[J]. 计算机应用与软件，2010，27(2)：71－73.

[7] LIU W T. Software protection with encryption and verification[J]. Advances in Intelligent and Soft Computing, 2012, 115(2): 131－138.

[8] MIAO Q X. Research and analysis on encryption principle of True-Crypt software system[C]// Proceedings of the 2nd International Conference on Information Science and Engineering. Piscataway, NJ: IEEE Press, 2010: 1409－1412.

[9] GREDIAGA A, BROTONS F, LEDESMA B, *et al*. Analysis and implementation hardware-software of Rijndael encryption[J]. IEEE Latin America Transactions, 2010, 8(1): 82－87.

[10] 李章兵，李曙红，冯建湘. 一种动态证书副本的软件版权保护模型研究[J]. 小型微型计算机系统，2011，32(8)：1633－1638.

[11] 涂志勇. 博弈论[M]. 北京：北京大学出版社，2009.

[12] PHILLIPS G M. Interpolation and approximation by polynomials[M]. Berlin: Springer, 2003.

[13] 宋瑞霞，李国富，邹建成，等. Bernstein 多项式及其幻曲面[J]. 计算机辅助设计与图形学报，2003，15(5)：532－536.

[14] 陈够喜，伍玉良，陈俊杰，等. 面向图像分存的批量隐写算法研究[J]. 中北大学学报：自然科学版，2011，32(6)：738－745.

[15] 付东来，陈够喜，杨秋翔. 非完备多载体隐写算法研究[J]. 小型微型计算机系统，2012，33(2)：388－391.

[16] 姜启源，谢金星，叶俊. 数学模型[M]. 北京：高等教育出版社，2011.