doi: 10.7690/bgzdh.2018.07.010

软件保护技术

糜　旗，宗俊珺，朱　杰，范涵冰
（中国航天科技集团第八研究院上海航天动力技术研究所，上海 201109）

摘要：针对越来越多软件遭到盗版和破解的问题，详细阐述了软件保护技术的发展历程和技术难点，并具有针对性地研究了常见软件保护技术。结果表明：这些技术可以在特定的时间段提高软件破解的难度和成本，从而使软件变得相对安全，同时也为软件保护后续研究提供参考。

关键词：破解；软件保护技术；安全
中图分类号：TP311.5　　**文献标志码：**A

Software Protection Technology

Mi Qi, Zong Junjun, Zhu Jie, Fan Hanbing
(*Shanghai Space Propulsion Technology Research Institute*, *No*. 8 *Academy*, *China Aerospace Science & Technology Corporation*, *Shanghai* 201109, *China*)

Abstract: For more and more software has been pirated and cracked, the paper describe the software protection technology development and technology difficulties in detail, and research on the ordinary software protection technology specially. The results show that these technologies can improve the software cracking difficulties in special time and cost, so as to make the software relatively safe and provide reference for the follow-up research of software protection.

Keywords: crack; software protection; safe

0　引言

随着互联网的快速发展，通过网络进行交易和传播的软件越来越多。与实体商品不同，软件容易被复制、修改和传播；因此，在互联网上传播的软件存在着大量的盗版和侵权问题。而最终用户有意或无意地对软件进行任意拷贝，尤其是无所顾忌的二次传播[1]，会以几何级数的速度消耗潜在的用户资源，给软件开发者造成巨大的经济损失，挫伤他们使用互联网扩展业务的积极性[2]。大力发展软件保护技术就变得尤为迫切和重要[3-4]。

从广义的角度来说，软件保护技术包括计算机系统和软件的安全。这些技术包括防止合法用户和其数据被恶意客户端程序攻击，合理设计管理计算机来实现一个严密的安全系统，是目前大多数计算机安全研究的重点。从狭义的角度来说，软件保护技术是在恶意环境下如何保护软件的自身数据和计算结果不受到破坏和剽窃，软件能够在授权范围内正确使用的相关技术。因此，软件保护技术是一种综合的计算机技术[5]，可以涉及到底层硬件、驱动程序、操作系统、网络协议数据库和软件编程等各方面的内容。

1　软件保护技术发展历程

笔者认为，软件保护技术从开始兴起到迄今为止，大致有 40 a 的发展历史，伴随着 PC 个人计算机操作系统的发展，大体上经历了 6 个时期，分为前 DOS 时期、DOS 时期、前 Windows98 时期、Windows98 时期、WindowsXP 时期和后 WindowsXP 时期。

1.1　前 DOS 时期

软件保护技术的最早开始时间已经难以考证，但大体上，20 世纪 80 年代初期在 Commond64，Amiga 游戏机上首先出现了软件保护技术的雏形。那时游戏机软件并没有得到与其他商业软件相同的认可，大部分人虽然觉得好玩，却不愿意购买，大家都去借或者互换游戏机软件。为了避免这种现象，游戏机软件开发商开始使用各种各样保护技术[6]。

1.2　DOS 时期

随着个人计算机和 DOS 操作系统的兴起，出现了大量的游戏、商业和办公软件，而软件保护的传统也被继承下来。DOS 时期，软盘是软件传播唯一的载体，因此 DOS 时期的软件保护技术绝大部分都

收稿日期：2018-04-10；**修回日期：**2018-05-13
作者简介：糜　旗(1979—)，男，上海人，硕士，高级工程师，从事网络安全、软件保护研究。

是基于软盘实现的。这种保护技术的原理是在软盘的特定位置制作一些特定数据，软件在运行过程中会校验这些特定数据，最为关键的是这些特定数据使用常规方法根本无法复制。这种保护技术简单、高效、廉价，在很长一段时间内是软件开发者的优先选择。

同期还出现过密码本加密和卡加密 2 种软件保护技术，但是由于这 2 种保护技术存在着先天缺陷，如使用不便，成本过高，因此这 2 种软件保护技术并没有大规模流行。

1.3　前 Windows98 时期

Windows3.0 的出现，让基于软盘的软件保护技术很难在保护模式下运行，这让很多软件开发者不适应这个全新的操作系统。当 Windows95 问世时，很多软件开发者意识到 DOS 时期的软件保护技术已经被淘汰，于是民用类、娱乐类软件开始使用简单的序列号保护技术，但这种全新的保护技术并不完善，很容易被破解。而商用软件则采用加密锁保护技术，加密锁插在计算机并口或串口上，里面有逻辑电路或存储单元，能够与计算机进行通信。由于加密锁安装在计算机外部，正版用户可以自行安装，操作简单，但用户的软件成本会相对增加。总的来看，这个时期的软件保护技术亮点很少。

1.4　Windows98 时期

在 Windows98 时期，计算机开始进入互联网时代，一种新的软件销售模式——共享软件出现，这是一种先用后买的销售模式，从互联网上下载后可以直接使用，但软件为未注册版，存在附加限制，如时间限制或功能限制，使用一段时间后，觉得软件不错再花钱购买。

销售模式的变化直接导致了软件保护技术的变化，软件注册授权就是这个时期最主流的保护技术。软件注册授权的基本流程是软件在运行过程中搜集当前计算机的特定信息，有时也可将其称之为机器码，当用户通过网络支付或汇款购买软件后，用户通过网络将计算机的特定信息发给软件开发者，软件开发者再根据特定信息计算出相应授权信息，用户使用授权信息后，软件由未注册版变成了正式版[4]。但这种软件保护技术给正版用户也造成了一定的麻烦，因为软件被限制在特定的一台计算机上使用，如果用户更换计算机或者升级计算机硬件，就需要重新验证授权信息，即使不需要额外购买授权信息，也会耗费掉重新验证授权信息的时间。

1.5　WindowsXP 时期

进入 WindowsXP 时期，软件注册授权依然是最主流的保护技术，而且越来越多的软件开发者在编写软件的验证授权信息代码时，采用密码学中不可逆加密算法。这类算法往往强度很高，如果使用方法得当，可以提高软件的保护质量。

同时期还出现了各种各样专业的加壳软件，这类加壳软件采用大量与操作系统应用程序编程接口（application programming interface，API）函数紧密相连的软件保护技术，比如可移植可执行的（portable executable，PE）文件技术、反跟踪技术、反 Dump 技术、API 重定向、变形代码、驱动技术和虚拟机技术等。如果给软件加上一个好的外壳，无疑会大幅增加软件的破解难度，延长软件被破解的周期，这意味着可以给软件开发者带来更多经济利益[7]。

1.6　后 WindowsXP 时期

在后 WindowsXP 时期，随着互联网的高速发展，网络速度与接入用户都呈现出爆发式增长，于是软件开发者开始尝试使用网络激活的保护技术。这种保护技术最大特点就是把原本软件中验证授权信息的代码移植到远程服务端上，软件把计算机特定信息通过网络发送到远程服务端进行授权验证，远程服务端验证完毕后返回验证结果，软件根据验证结果判断是否激活软件全部功能，提高了软件的安全质量[8-9]。

而在商业软件方面，中国加密锁厂商率先将应用于安全领域的智能卡芯片引入到软件保护领域。这种智能卡内置了中央处理器（central processing unit，CPU）、只读内存（read-only memory，ROM）和随机存储内存（random access memory，RAM），允许加密锁厂商进行内置操作系统开发，从功能角度讲近似于一个迷你计算机。目前国内及国际相关智能卡加密锁的大部分发明专利都为中国人所有。和传统的单片机加密锁相比，它具有更高的安全性能、更大的存储空间和远程升级功能[10-11]。

2　软件保护技术难点

在软件保护的研究过程中，不仅需要学习计算机安全方面的技术，还需要了解掌握计算机科学其他方向的技术，如逆向分析、数字水印和编译器优化等。

事实上，我们很难在软件开发过程中熟练合理地使用软件保护技术，原因主要有 2 个方面：1) 很多公司对自己拥有的软件保护技术专利讳莫如深；2) 缺乏软件保护技术专门的刊物、学术会议[12]。从而导致了国内外大多数软件开发者在设计和实现自己的软件保护过程中，或多或少都会存在以下 3 种误区。

1) 仅使用软件保护产品的默认解决方案。

从事软件保护行业的公司在销售软件保护产品的同时，也可能会提供一些默认解决方案或者经典案例，但由于这类产品在市场上能够被轻易获取，且方案会产生相对固定的软件保护模式。

如果软件开发者仅使用软件保护产品的默认解决方案，而未将软件与软件保护产品进行二次开发，那么破解者在分析破解时就变得相对容易轻松[13]。

2) 仅在软件开发完成后实施软件保护。

一个优秀的软件保护方案应该贯穿于软件开发的始终，从一开始就已经充分地融合到软件设计方案中，而不是等到软件开发接近尾声，再临时实施软件保护。因为临近软件发布，时间紧急，很难设计出比较完善的软件保护方案。而大多数破解者都身经百战，经验丰富，匆忙设计出的软件保护根本无法有效抵抗分析破解。

3) 采用高强度国际标准算法加密很难被破解。

一部分软件开发者认为，高强度国际标准算法等同于高强度的软件保护，但实际上这完全隶属于 2 个不同的概念。因为算法保护的对象是数据，而软件正确无误地运行在内存中，所有的加解密、判断、循环对于破解者都是完全公开的，即使破解者看不懂加密算法，但若是某些敏感字符串以明文形式出现在内存中，同样也可以轻松分析破解。

3 软件保护技术研究

软件保护技术的作用是为了防止软件被破解盗版，但使用软件保护技术同时又将提高软件开发成本，降低软件性能，使软件开发周期变得更复杂。很多软件开发者对软件保护技术有一个错误的认知，认为只有无法被破解的软件保护技术才是成功的软件保护技术。实际上，几乎不存在无法被破解的软件。因为软件最后是让最终用户去使用，无论如何保护软件，都必须在最终用户的计算机上正常运行。从理论上而言，软件在实现保护技术前后功能应当完全相同，那么必须有某种方法可以在保证软件功能不发生变化的前提下，把保护技术从软件中消除，仅需要时间而已。因此笔者认为，如果某种软件保护技术能够使软件在生命周期内不被破解，那么这种软件保护技术就是成功的。

3.1 加壳

和大自然中植物的壳用来保护种子一样，计算机软件的壳是一段专门用来保护软件不被非法篡改或反编译的代码或程序[14]。加壳是软件保护的常用手段，就是利用特殊的算法，对 EXE、DLL 或 OCX 程序里的资源进行压缩和加密，类似压缩软件 Winzip 的效果，只不过被压缩后的程序，仍然可以独立运行[14]，在内存中完成解压和释放，完全隐蔽。加壳程序的原始代码在磁盘中一般是以加密的形式存在，无法查看源代码。直接运行后在内存中还原，这样就可以比较有效地防止程序被静态反编译，同时也可以防止原始程序被非法修改[15]。

当加壳程序运行后，壳会附加在原始程序上通过 Windows 加载器载入内存，先于原始程序执行得到控制权[16]，关于这点，壳和病毒比较类似。壳在执行过程中对原始程序进行解密、还原，还原完成后再把控制权交还给原始程序，执行原来的代码部分。

壳一般分为压缩壳和加密壳 2 种。压缩壳仅对原始程序里的资源进行压缩，如 Aspack、UPX 和 PeCompact 等，通常使用压缩壳保护，被保护程序大小都小于原始程序。而加密壳不仅对原始程序里的资源进行压缩，而且还具备函数输入表加密、反调试、双进程保护和虚拟机等功能，如 ASProtect、Armadillo 和 Themida 等，通常使用加密壳保护，被保护程序大小会略小于甚至大于原始程序[17]。

有些壳甚至还集成反跟踪技术、序列号识别、使用次数及时间限制功能，如果给软件加一个优秀的壳，无疑会大大增加软件的破解难度，延长软件的生命周期，这意味着可以给软件开发者带来更多的经济效益。

3.2 花指令加密

花指令技术源于多态病毒技术，在不修改原代码功能的基础上，通过某种代码变换方式，使原代码的自身结构变化出不同的代码，但原代码与新代码的逻辑功能完全相同[18]。

花指令使用的是多态病毒技术中的代码模糊变换方法，主要用于对抗静态反汇编。在 80x86 系列计算机中代码的反汇编分析和真正执行过程并不完

全相同。代码的真正执行过程是 CPU 每执行完一条指令后，再去取下一条指令。而反汇编分析是一种 Dump 式自上向下的处理过程。花指令就是利用了这两者的差异性，成为一种有效欺骗静态分析的重要手段。当原代码经过模糊变换后，新代码难以进行逆向工程分析，使反汇编结果出现异常错误，从而实现软件保护。目前花指令技术采用较多的一种形式是直接在程序代码中 JMP 指令后添加花指令。该方法具有简单、易实现的优点。

3.3　反调试器

调试器工作于 CPU 和操作系统之间，运行在 Ring0 或 Ring3 级[19]，能够调试操作系统内核或应用软件。调试器能将已经是机器码的 EXE 文件加载入内存，具有下断点、单步运行等功能[20]，软件如果不具备反调试器的保护技术，破解者就可以使用调试器分析软件。反调试技术中主要使用 4 种技术：句柄检测、断点检测、反加载和反监视[21]。

3.4　CRC 自校验

作为一种有效的差错检测手段，循环冗余校验(cyclic redundancy check，CRC)在数据存储和网络通信领域被广泛使用。利用 CRC 校验原理，软件在编译时，在代码内写入自身的合法 CRC 校验值[22]，软件运行过程中，通过 CRC 校验值对自身的完整性进行检查。如果修改了软件中的任何一个字节，软件通过校验计算可以发现当前 CRC 校验值与原合法 CRC 校验值不同，由此判断软件已经被修改[23]。

3.5　授权码验证

授权码也可以称为注册码、序列号等。授权码验证的过程，就是验证特定信息和授权码之间的数学映射关系，如果数学映射关系不正确，由此可以判断软件未被合法授权使用。这个映射关系是由软件开发者设定，所以不同软件生成授权码的算法可能完全不同。这个映射关系越复杂，授权码越不容易被破解。根据映射关系的不同[24]，授权码验证通常有以下 4 种方法：

1) 以特定信息作为自变量，一般特定信息为用户名或者机器码，通过函数 F 变换后得到授权码，公式表示为：

$$\text{授权码}=F(\text{特定信息})。\tag{1}$$

将这个授权码和用户输入的授权码进行字符串比较或者数值比较，以确定用户是否为合法用户。

2) 通过授权码验证特定信息的正确性。软件开发者生成授权码的时候，仍然使用函数 F，公式为

$$\text{授权码}=F(\text{特定信息})。\tag{2}$$

但这个函数 F 是可逆运算，软件在检查授权码时是利用函数 F 的逆运算：

$$\text{特定信息}=F^{-1}(\text{授权码})。\tag{3}$$

对输入的授权码进行变换。如果变换的结果和特定信息相同，则说明是正确的授权码。使用这种方法用来生成授权码的函数 F 没有直接出现在代码中，而且正确授权码的明文也未出现在内存中，所以这种方法比第 1 种方法要安全。

3) 通过对等函数检查授权码。如果特定信息和授权码满足式(4)，则认为是正确的授权码，采用这种方法授权码明文同样不会在出现在内存中。这种方法其实是第 2 种方法的延伸变形。

$$F1(\text{特定信息})=F2(\text{授权码})。\tag{4}$$

4) 使用二元函数，同时以特定信息和序列号为自变量。这种验证方法采用下式的判断规则：

$$\text{特定值}=F(\text{特定信息，授权码})。\tag{5}$$

当对特定信息和授权码进行变换时，如果得出的结果和某个特定值相等[24]，则认为是合法的一对特定信息和授权码。这种方法比前几种方法更为安全，特定信息与授权码之间的关系更为模糊。根据式(5)的思路可以把特定信息和授权码分拆为几部分来构造多元函数公式(6)。

$$\text{特定值}=F(\text{特定信息1, 特定信息2……}\\ \text{授权码1, 授权码2……})。\tag{6}$$

3.6　硬件保护技术

常见的加密锁主要有以色列阿拉丁公司的 HASP 系列、彩虹公司的 SuperPro 系列、北京飞天诚信公司的 ROCKEY 系列和深思公司的深思洛克系列等。最初的加密锁，即第一代加密锁被称为逻辑电路加密锁。它完全用硬件实现内部算法，通过并口通信与主机通信的软件交互数据[25]。它比加密卡成本低，容易使用，所以迅速被普及。许多重要的行业软件都有过采用加密锁的经验。第一代逻辑电路锁中的算法虽然软件开发商可以选择，但是毕竟只有很少的几种。

随着技术的发展，第二代加密锁可以让开发商在锁中定义自己的独特信息，比如软件密钥、用户信息，甚至是软件运行中产生的中间变量，所以被称为存储器加密锁。第三代充分吸取了前两代的优点，内置了存储器和硬件逻辑电路，既能记录软件开发者的授权信息，又能为其提供高强度的硬件加

密算法[26]，因此被称为逻辑电路加存储器加密锁。第四代为可编程的加密锁，可编程加密锁最初设计目的是让软件开发者能够将软件中重要的代码或模块写入到加密锁中运行，使软件与加密锁实现真正无缝链接。但受限于成本，公开销售的几款加密锁都采用了低成本的单片机，给代码写入造成了很大的困难，主要表现在 3 个方面：缺少复杂度高的算法变换、程序区和指令编码的空间太小。这些问题使得软件开发者很难使用第四代加密锁实现高强度的软件保护方案[27]。于是第五代加密锁采用智能卡技术，全面改进了第四代加密锁的缺点。

市面上的第五代加密锁主要有北京飞天诚信公司的 ROCKEY5、深思公司的 SenseLock IV 型锁和挪威 Sospita 公司的 Sospita 加密锁。这些加密锁大多功能强大，可将部分程序直接放入加密锁中运行。

4 结束语

在软件保护技术不断发展的过程中，保护技术和破解技术一直处于不断的博弈中，不存在一种绝对安全、没有漏洞的保护技术。软件的安全都是相对而言的，只能是在特定的时间段提高软件破解的难度和成本，从而变得相对安全。一旦难度和成本降低，软件安全性可能无从保证，因此软件保护技术依然存在着巨大的发展潜力和空间。

参考文献：

[1] 周国祥, 陆文海. 基于 BHO 技术的数字版权保护系统的研究与设计[J]. 计算机研究与发展, 2010, 47(6): 316-320.

[2] 刘芳, 金松根, 卢国强. 数字版权管理与数字图书馆建设[J]. 图书馆学刊, 2011(4): 105-108.

[3] 国务院. 计算机软件保护条例[J]. 新疆新闻出版, 2013(2): 82-84.

[4] 刘军. 基于 CUDA 的软件保护技术研究与实现[D]. 长沙: 湖南大学, 2011: 5-25.

[5] KENNETH C L. 编译原理与实践[M]. 冯博琴, 等, 译. 北京: 机械工业出版社, 2004: 30-60.

[6] SEANBARNUM, G MG. Knowledge for software security[J]. Security & Privaey May-azine IEEE, 2005, 3(2): 74-78.

[7] 段钢. 加密与解密[M]. 2 版. 北京: 电子工业出版社, 2004: 40-48.

[8] 吕杨, 李超. 逆向工程之软件破解与注册机编写[J]. 计算机安全, 2011(1): 66-69.

[9] 罗云彬. Windows 环境下 32 位汇编语言程序设计[M]. 3 版. 北京: 电子工业出版社, 2013: 15-19.

[10] ZHAO X W, ZHAO F G, TIAN H B. Dynamic asymmetric group key agreement for ad hoc network[J]. Ad hoc Network, 2011, 9(5): 928-939.

[11] SHARMA B K, AGARWAL R P, RAGHURAJ S H. Copyright Protection of Online Application using Watermarking[J]. International Journal of Computer Applications, 2011, 18(4): 47-51.

[12] 武新华, 安向东, 苏雅. 加密与解密全方位学习[M]. 北京: 中国铁道出版社, 2006: 28-90.

[13] 黄剑军. 基于带权欧氏距离的壳检测与脱壳技术的研究[D]. 杭州: 杭州电子科技大学, 2009: 11-33.

[14] 高艳军. 数据安全管理系统加壳技术研究与实现[D]. 长沙: 国防科学技术大学, 2008: 15-30.

[15] 马珂. 基于虚拟机的内核模块行为分析技术研究[D]. 湘潭: 湘潭大学, 2014: 20-28.

[16] 郝景超. Windows 下软件实名机制的设计与实现[D]. 郑州: 解放军信息工程大学, 2008: 24-36.

[17] 李勇. 基于 Windows 平台的目标代码混淆[D]. 成都: 电子科技大学, 2007: 37-45.

[18] 孙国梓, 陈丹伟, 蔡强. 子程序花指令模糊变换逻辑一致性研究[J]. 计算机科学, 2009, 36(8): 89-91.

[19] 郑大公. 软件保护中的反跟踪技术[J]. 现代企业教育, 2007, 24(12): 100-101.

[20] 高兵, 林果园, 王莹. 基于代码自修改的软件反跟踪技术结构[J]. 信息网络安全, 2014(5): 46-51.

[21] 刘晓冬. 软件加壳技术的研究与实现[D]. 沈阳: 沈阳工业大学, 2006: 8-24.

[22] 卢晓雄. 程序 CRC 自校验方法与实现[J]. 岳阳职业技术学院学报, 2008, 69(1): 87-89.

[23] 刘建中. 软件系统加密研究[D]. 郑州: 解放军信息工程大学, 2004: 19-38.

[24] 杨彩霖. 常用软件加密技术及优缺点分析[J]. 商丘职业技术学院学报, 2011, 10(5): 28-30.

[25] 李洁. 提花编织机工艺图信息提取算法和软件的研究[D]. 哈尔滨: 哈尔滨工业大学, 2007: 18-36.

[26] 徐亮. 嵌入式加密卡设计[D]. 大连: 大连海事大学, 2009: 26-43.

[27] 徐智穹. 基于 ASIC 的软件加密方法研究及实现[D]. 武汉: 武汉理工大学, 2004: 15-41.