计算机专业人员书库

加密与解密——软件保护技术及完全解决方案

看雪 编著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
URL: http://www.phei.com.cn

计算机专业人员书库

加密与解密
——软件保护技术及完全解决方案

看　雪　编著

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

内 容 简 介

软件保护工作是维护软件开发人员利益的必要手段，是软件开发过程中的重要环节。本书讲述 Windows 环境下的软件保护技术及相关解决方案。全书共分为 3 个部分。第一部分介绍与软件保护技术相关的基础知识。第二部分讲述各种软件保护方式，如序列号、时间限制、CRC、SMC 及各种加密算法，并且还介绍了注册表、VB 保护程序的特点。第三部分主要介绍脱壳的基本技巧，以及一些大型商业软件保护的实例。

本书是国内密界一流好手的呕心之作，通过解析大量的实例来展示软件调试的最深处，是软件开发人员不可多得的读物。

图书在版编目(CIP)数据

加密与解密——软件保护技术及完全解决方案/看雪编著. - 北京：电子工业出版社，2001.9

(计算机专业人员书库)

ISBN 7-5053-6923-7

Ⅰ.加… Ⅱ.看… Ⅲ.软件开发 - 保护 - 技术 Ⅳ.TP311.56

中国版本图书馆 CIP 数据核字(2001)第 058725 号

丛 书 名：计算机专业人员书库

书　　名：**加密与解密——软件保护技术及完全解决方案**

编 著 者：看　雪

责任编辑：郭　立　毛兆余

特约编辑：吴明卒

排版制作：电子工业出版社计算机排版室

印 刷 者：北京大中印刷厂

装 订 者：三河市双峰装订厂

出版发行：电子工业出版社　URL: http://www.phei.com.cn

北京市海淀区万寿路 173 信箱　邮编 100036

经　　销：各地新华书店

开　　本：787 × 1 092　1/16　印张：26　字数：666 千字

版　　次：2001 年 9 月第 1 版　2001 年 9 月第 1 次印刷

书　　号：ISBN 7-5053-6923-7 / TP·3942

印　　数：8 000 册　　定价：40.00 元(含光盘)

出 版 说 明

信息产业的飞速发展迫切要求人们快速掌握新技术，并加以应用。选择适合自身的好书，少走弯路，从而节约时间，赢得主动就显得尤为重要。作为出版者，提供明确读者定位的作品，让读者对在什么情况下选择什么样的书有一个理性的而不是情绪化的选择，是我们的责任和义务。

秉着“专”和“精”的原则，我们推出这套《计算机专业人员书库》。

- ✓ 专——内容专，丛书面向的是有一定专业基础的计算机应用开发人士。
- ✓ 精——内容和作者精。在精心的前期选题内容组织安排下，挑选来自专业领域的资深技术专家和大学教授作为本套丛书的作者，保证丛书的质量。

出版高品位、高品质的图书是电子工业出版社计算机图书事业部的努力目标，您的意见是我们创造精品的动力源泉！如果您对本丛书有任何意见和想法，或有意撰写面向计算机专业人员的书籍，欢迎指点或垂询！

我们的 E-mail：jsj@phei.com.cn

电话：010-68216158。

电子工业出版社

计算机图书事业部

2001 年 09 月

前　言

选择本书的理由

自计算机诞生之日起，其技术的发展可谓日新月异，各种新技术、新思路不断涌现。个人计算机操作系统也经历了 DOS、Windows 3.x、Windows 9x、Windows 2000 的历程，如今 Windows XP 正一步步地向我们逼来。各种应用软件从最初的几个、几十个字节发展到现在的动辄几张光盘，成千上万的共享软件和商业软件越来越庞大，技术内涵也日趋复杂。一款优秀的软件，其技术秘密往往成为他人窃取的重点。为了保护辛辛苦苦开发的软件不轻易被他人“借鉴”，作为软件开发人员，有必要对软件保护（加密）和破解（解密）方面进行研究。但目前软件保护和破解方面的资料比较匮乏，许多软件开发人员不得不自行摸索，导致在重复劳动中走了不少弯路，耗费了大量的时间和精力。

在本书中，我们试图从软件保护（加密）和破解（解密）两方面对当今流行的软件保护技术进行分析。希望读者在看过本书后，能够对当今流行的各种软件保护技术及破解技术有所了解。保护与破解的技术在相辅相成中不断发展，既没有无坚不摧的矛，也不会有坚不可摧的盾。只有了解了各种保护技术及破解技术，才能根据各自的特点开发出更加先进的保护技术。

本书的编写力求以实用为目的。对于每项技术，首先进行详细而透彻的讲解，然后通过若干应用实例进行具体的分析，使得一般的读者朋友在了解 8086/8088 汇编基础上，能尽快地掌握这些技术，进入软件开发和应用的高级阶段。

本书的主要内容

本书所讲述的内容是基于 Windows 9x 与 Windows 2000 操作系统，所有程序均已调试通过。第一部分介绍基础知识，讲述软件保护的种类，软件解密调试工具的安装与使用。第二部分讲述各种软件保护方式，如序列号、时间限制、CRC、SMC 及各种加密算法以及注册表、VB 程序保护的特点等。第三部分主要介绍脱壳的基本技巧（这是汉化软件的必修课）以及一些大型商业软件的保护方法。

作者及参编人员简介

本书作者段钢（网上 ID：看雪），1994 年毕业于同济大学，2000 年创建“看雪学院”。

参与本书编写的 CCG 成员：Sun Bird、Blowfish、Ajj、Hying、DDXia、Jojo、小楼、洋白菜、似曾相识、Garfield、CoolBob、小牧童、ICEbIRD、PeterS、Gfh 等。

热心参与的网友：吴朝相、TiANWEi、D.boy、Fisheep、WinDos2K、Passion、Arbiter、Ljtt、Henryw、Mr.wei、DREAMtHEATER、冰毒、Arbiter、真 H.P、八爪鱼、Zjs、Leafred、Iceworld、Yyjiang、Sbing、Liutong、Wind、梦醒时分、程式猎人、Zxem、Angela、Denver 等。

另外还有众多网友将自己的经验无私地放在网上供大家学习，他们对本书的帮助也是非常大的。

由于软件保护与破解必然涉及很多微软操作系统的内核技术，而这些技术往往是微软公司不公开的，加上我们的水平有限，所以本书中难免会有种种不足，我们真诚欢迎读者给予批评与指正。

看　雪

2001.8

目　录

第1章 基本常识

1.1 软件保护与解密

软件的破解技术与保护技术是矛与盾的关系，它们是在互相斗争中发展进步的。两者在技术上的较量归根到底是一种利益的冲突。软件开发者为了维护自身的商业利益，不断地寻找各种有效的技术来保护自身的软件版权，推迟软件被破解的时间；而破解者则受盗版所带来的高额利润的驱使或出于纯粹的个人兴趣，而不断开发新的破解工具，针对新出现的保护方式进行跟踪分析以找到相应的破解方法。从理论上说，几乎没有破解不了的保护。对软件的保护仅仅靠技术是不够的，最终要靠人们的知识产权意识和法制观念的进步以及生活水平的提高。如果一种保护技术的强度强到足以让破解者在软件的生命周期内无法将其完全破解，这种保护技术就可以说是非常成功的。软件保护方式的设计应在一开始就作为软件开发的一部分来考虑，列入开发计划和开发成本中，并在保护强度、成本、易用性之间进行折衷考虑，选择一个合适的平衡点。

在桌面操作系统中，微软的产品自然是独霸天下，一般个人用户接触得最多，研究得自然也更多一些。在 DOS 时代就有不少比较好的软件保护技术，而在 DOS 中使用得最多的恐怕要算软盘指纹防拷贝技术了。由于 DOS 操作系统的脆弱性，在其中运行的普通应用程序几乎可以访问系统中的任何资源，如直接访问任何物理内存、直接读写任何磁盘扇区、直接读写任何 I/O 端口等，这给软件保护者提供了极大的自由度，使其可以设计一些至今仍为人们称道的保护技术；自 Windows 95 开始（特别是 Windows NT 和 Windows 2000 这样严格意义上的多用户操作系统），操作系统利用硬件特性增强了对自身的保护，将自己运行在 Ring 0 特权级中，而普通应用程序则运行在最低的特权级 Ring 3 中，限制了应用程序所能访问的资源，使得软件保护技术在一定程度上受到一些限制。开发者要想突破 Ring 3 的限制，一般需要编写驱动程序，如读写并口上的软件狗的驱动程序等，这增加了开发难度和周期，自然也增加了成本。由于 Win32 程序内存寻址使用的是相对简单的平坦寻址模式（其采用的 PE 文件格式也相应地比以前的 16 位 EXE 程序的格式要容易处理一些），并且 Win32 程序大量调用系统提供的 API，而 Win32 平台上的调试器如 SoftICE 等恰好有针对 API 设置断点的强大功能，因而这些特点都给跟踪破解带来极大的方便。

加密与解密的发展有以下 4 个历程。

1. DOS 时期

这个时代的软件主要是正式版和功能不全的 Demo 版，很少有所谓的共享软件（Shareware）。所以，DOS 时代所谓的解密，通常是去掉软件中的某些限制或是跳过原版磁盘检查，然后通过广大业余的 BBS 提供下载。

2．Windows 95 早期

当 Windows 95 出现的时候，很多人不适应这个平台，在它上面的加解密资料奇缺，对许多人来说就像做了一场恶梦。这段时期共享软件渐渐地盛行起来，采用序列号保护的共享软件越来越多。由于当时许多程序员对刚刚出现的 Windows 95 不了解，感觉有些手足无措，编制的软件在加密部分都比较脆弱，所以那时候的序列号加密方案特别脆弱。

3．Windows 95 末期

其实这段时期应该是 Windows 95 和 Windows 98 共存的时期。在这个时期，程序员已经对 Windows 9x 这个系统了如指掌，像 Vopt 等等很需要编程技巧、与系统核心等底层联系紧密的软件纷纷出笼。这时期共享软件大多还是采用序列号加密方案，但其序列号通常也经过复杂的计算，很难像早期的软件一样，随随便便就可以解密了。

4．Windows 2000 时期

这段时期就是 Windows 9x 和 Windows 2000 共存时期，各种软件的加密外壳泛滥，特别是各种专门的加密软件出现，大大地提高了软件的保护质量；此时解密技术也上了一层楼，各种新式的解密工具随即出现。此时的序列号加密越来越多地采用密码学中不可逆的加密算法，使得解密的过程越来越像高等数学的研究。软件解密者要想得到正确的序列号就必须对加密算法有很深入的了解，找到软件加密算法的漏洞（像 WinRAR、CloneCD 等软件的 KeyGen 就是利用 ECC 加密算法的漏洞编制出来的）。

1.2　软件分析技术

在进行软件汉化、软件解密的过程中，一个首要问题是对被汉化的软件、解密的软件进行分析。对于它们的分析可以使用动态调试程序 SoftICE、TRW2000 等进行，还有一种方式就是静态分析，两者各有利弊。为了更有效地调试软件，有必要对软件分析的一般方法进行研究，以总结出对软件进行分析的一般途径和策略。

1.2.1　从软件使用说明和操作格式中分析软件

若要分析一个软件，首先应该学会使用该软件，最好认真阅读软件的使用手册，特别是自己所关心的关键部分的使用说明，并学会操作使用，以从外部了解软件所完成的功能。一个有经验的程序分析员往往通过软件的使用说明和操作，推测到软件的设计思想和编程思路。

1.2.2　静态分析技术

所谓静态分析即从反汇编出来的程序清单上分析，最常用的方法是从提示信息入手。目前，大多数软件在设计时，都采用人机对话方式。所谓人机对话，即在软件运行过程中，需要由用户选择的地方，软件即显示相应的提示信息，并等待用户的按键选择。而在执行

完某段程序之后，便显示一串提示信息，以反映该段程序运行后的状态，是正常运行，还是出现错误，或者提示用户进行下一步工作的帮助信息。为此，如果对静态反汇编出来的程序清单进行阅读，通过显示提示信息的程序段，就可知道在显示提示信息前后的程序片段所完成的功能，从而宏观了解软件的编程思路。常用的静态分析工具是 W32Dasm、IDA 和 HIEW 等。

1.2.3 动态分析技术

虽然从静态上可以了解各个模块的功能以及整个软件的编程思路，但是，并不可能真正地了解软件中各个模块的技术细节。对于被分析的软件来说，静态分析只是工作的第一步，动态跟踪是分析软件的关键所在。所谓动态跟踪主要是指利用 SoftICE 或 TRW2000 等调试工具，一步一步跟踪分析。为什么要对软件进行动态分析呢？

（1）许多软件在整体上完成的功能，一般要分解成若干模块来完成；后一模块在执行时，往往需要使用其前一模块处理的结果，这一结果叫做中间结果。如果只对软件本身进行静态分析，一般是很难分析出这些中间结果的。而只有通过跟踪执行前一模块，才能看到这些结果。在程序的执行过程中，往往会在某一地方出现许多分支和转移，不同的分支和转移往往需要不同的条件，而这些条件一般是由运行该分支之前的程序产生的。若想知道程序运行到该分支时，到底走向哪一分支，不进行动态跟踪和分析是不得而知的。

（2）有许多软件在运行时，其最初执行的一段程序往往需要对该软件的后面各个模块进行一些初始化工作，而没有依赖系统的重定位。

（3）有许多加密程序为了阻止非法跟踪和阅读，对执行代码的大部分内容进行了加密变换，而只有很短的一段程序是明文的。加密程序运行时，采用逐块解密，逐块执行的方法，首先运行最初的一段明文程序。该程序在运行过程中，不仅完成阻止跟踪的任务，还要负责对下一块密码进行解密。显然，仅对软件的密码部分进行反汇编，不对该软件动态跟踪分析，是根本不可能进行解密的。

由于上述原因，在对软件静态分析困难的条件下，就要进行动态分析了。那么如何有效地进行动态跟踪分析呢？一般来说有如下两点。

1．对软件进行粗跟踪

所谓粗跟踪，即在跟踪时要大块大块地跟踪。也就是说，每次遇到调用指令 CALL、重复操作指令 REP、循环操作指令 LOOP 以及中断调用指令 INT 等，一般不要跟踪进去，而是根据执行结果分析该段程序的功能。

为什么要大块大块地跟踪呢？其原因是，一个软件一般划分为若干模块，当要分析一个软件时，主要是分析软件中我们所关心的那一部分模块，而最初执行的模块往往不可能是我们所关心的模块；如果一开始就一条一条地跟踪程序，往往会进行一些不必要的跟踪，浪费精力。这里必须注意一个问题，就是如何合理地设置断点？这就需要了解 Win32 API 函数了，根据当时的情况选择合适的断点，如拦截对话框，一般对话框是调用 MessageBoxa 函数来实现的，就可用此函数设断点，程序一调用此函数将被中断。

2．对关键部分进行细跟踪

对软件进行了一定程度的粗跟踪之后，便可获取软件中我们所关心的模块或程序段。这样，就可针对性地对该模块进行具体而详细的跟踪分析。一般情况下，对关键代码的跟踪可能要反复进行若干次才能读懂该程序，每次要把比较关键的中间结果或指令地址记录下来，这样会对下一次分析有很大的帮助。

软件分析是一种比较复杂和艰苦的工作，上面的几点分析方法，只是提供一种基本的分析思路。若要积累软件分析的经验，需要在实践中不断地探索和总结。

1.3　CPU、操作系统与编程语言

现在所使用的语言无非是两种，一种是解释执行的语言，另一种就是编译后才能够执行的语言。解释执行的语言因为解释器不需要直接同机器码打交道，实现起来较为简单，而且便于在不同的平台上面移植，如 Visual Basic、Visual FoxPro 等。编译执行的语言因为要直接同 CPU 的指令集打交道，具有很强的指令依赖性和系统依赖性，但编译后的程序执行效率要比解释语言高得多，像 Visual C/C++等。

现有开发语言所实现的加密性能实际上是比较差的，但现今世界上的开发语言大多是以执行的效率、易学性和可移植性为标准的，没有哪种语言是以加密性为标准的。如果希望编制加密性特别强的软件，就必须从现有的环境出发，采用一些特别的方法来提高软件的安全性能。

任何软件编制都离不开特定的 CPU、操作系统以及编程语言。虽然现在有 Java 这样号称不依赖任何 CPU 和操作系统的编程语言，但实际上只是 Java 虚拟机完成了这些工作。而且由于这些语言都不是真正的编译性语言，所以加密性都很差。

一个软件的加密性是否良好，同具体的 CPU、操作系统与编程语言是有很大关系的。若要破解 SGI 工作站上的程序，一定比 PC 上的程序要难得多。实际上并不是 SGI 工作站上的程序编得特别好，而是人们对 SGI 工作站的了解要比对 PC 的了解少得多，而且也很难找到相关的系统资料。这里有个加密方面的摩尔定律，某种技术的普及化程度越高，那么它的加密性就越差。例如用 VB 写一个程序，其中调用了大量的 ActiveX、COM 等控件，而且针对 MMX、3DNow、SSE…做了具体的优化。这样的程序要比完全用 VC 写的程序的安全性好得多，因为这完全是一种纯技术上的比拼，如果某个黑客对 VB、ActiveX、COM 的技术及内部工作原理十分清楚，可能破解这种程序要比 VC 编写的程序容易许多。但一个人的精力是有限的，不可能去了解所有的技术。一个人学习如何使用 ActiveX 来编程序可能只要 2～3 天的时间，但如果要完全搞懂 ActiveX 的工作原理，恐怕没有十天半个月是不行的。而且上面这些方法都不需要加密方面的专家就能实现，一个好的程序员只要多花些心思就能完成。

对 PC 而言，经历了 DOS、Windows 3.1、Windows 95/98、Windows NT、Windows 2000 等一系列操作系统的变迁。在 DOS 下面的程序基本都是 16 位段模式、单任务、线性指令流式编程。而在 Windows 3.1 下面已经转变为保护模式以消息传递为基础的编程，到了 Windows 95/98/NT/2000 又转向了 32 位保护模式、多任务，以消息传递为基础的编程。不同

平台下的编程方式也有很大的区别，相对于传统的单任务、线性指令流的编程方式，Win32下面的消息传递和多任务方式的编程大大增加了软件的复杂性，如果好好利用这一点，软件的加密性能够提升到一个相当的高度。

通常我们认为软件的加密性与软件的执行效率是成反比的，一个软件如果在加密上面考虑得过多，那么执行效率必然要下降。这个问题是因为大多数的软件开发者都在软件完成后才开始考虑加密性的问题，这样的加密不但斧凿的痕迹特别明显，而且带来软件效率的下降。如果在软件编制的初期就能考虑到加密性的问题，这个问题很容易解决。就拿常说的面向对象编程（OOP）来说，大多数人只在被动地使用 OOP 编程，而在具体的函数功能实现上往往还是传统的模块化思想。如果能把每个函数每个功能都细化成 OOP，软件的加密强度自然大大提高，而且软件的可维护性也会很好。针对核心的计算函数进行特定 CPU 的优化（FPU、MMX、3DNow、SSE…），不但软件的加密性提高，而且运算速度也会大大加快。

如果说软件的加密性同软件的执行效率不冲突，那么软件的加密性同软件的可移植性就很难共存了，因为不可能希望一把锁住所有门的锁有很好的安全性。不同的平台、不同的 CPU 都会产生不同的加密方案，但软件必然是在特定的 CPU 和操作系统平台上执行的。针对特定平台的优化，不但是安全性的要求，更多的是执行效率上的考虑。

1.4　保护模式简介

如果你想系统地学习解密与加密技术，保护模式的概念一定要掌握好，具体的内容请参考专门的技术资料[1]。当然初学者可先掌握一些基本概念，等入门后再回过头来深入研究，这也是一种比较好的学习方法。

1.4.1　Windows 的保护模式（Protected Mode）

一般来说，80x86（80386 及其以后的各代 CPU）可在实模式、保护模式和 V86 模式三种模式下运转。实模式就是古老的 MS-DOS 的运行环境。Windows 9x 只利用了保护模式和 V86 模式两种模式。在保护模式下程序可以利用更多的内存，而运行在实模式下的 16 位程序最多只能存取 1MB 的内存。如果你有一台 128MB 内存的机器运行在实模式下，那么只能利用 1MB 内存，其余的全部浪费了。在这种情况下，你的 386/486/586/P II/PIII只相当于一个跑得快的 8086。

理论上，在保护模式下，CPU 可以寻址 4096MB（即 4GB）内存。这就是说，只需要把程序编译成 32 位的可执行程序（当然得借助 32 位编译器），你就可在程序中充分利用内存了。

从硬件结构上说，386 由三个寄存器 CR0、CR1、CR2 控制着 CPU 的运转。比如说，CR0 的第 0 位就是用来判断当前 CPU 是工作在保护模式还是实模式下。学过 8088/8086 汇编语言的人，一定熟悉 AX、BX、CX、DX、SI、DI、SP、BP 这些 16 位的寄存器；在 80386 中，

[1] 推荐书籍：周明德. 保护方式下的 80386 及其编程. 北京：清华大学出版社
杨季文. 80x86 汇编语言程序设计教程. 北京：清华大学出版社