Broadview
www.broadview.com.cn

安全技术大系

强有力的程序保护技术！

Hacker Disassembling Uncovered

黑客反汇编揭秘

[美] Kris Kaspersky 著
谭明金 译
罗云彬 审校

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

安全技术大系

黑客反汇编揭秘

Hacker Disassembling Uncovered

[美] Kris Kaspersky 著

谭明金 译

罗云彬 审校

電子工業出版社

Publishing House of Electronics Industry

北京 · BEIJING

内 容 简 介

本书分为两大部分。第 1 部分结合精心挑选的实例，系统地讨论了黑客代码分析技术，包括调试器与反汇编器等典型分析工具的使用、代码分析的基本过程以及相关疑难问题的处理等。第 2 部分介绍了程序保护所面临的各种挑战及其相关的反调试、反跟踪、防反汇编以及代码加密解密技术等内容，这实际上是代码分析方面的高级专题。该书在内容上将针对性、实践性与综合性有机地结合在一起，很好地满足了学习代码分析技术的需要。

该书主要是为致力于计算机安全维护而阻止黑客侵袭或者从事安全保护程序开发的人员写的。同时，本书对于深入学习程序和操作系统等计算机内核知识，也有很好的参考价值。

Hacker Disassembling Uncovered, ISBN: 1-931769-22-2

版权贸易合同登记号　图字：01-2003-8835

图书在版编目（CIP）数据

黑客反汇编揭秘 /（美）卡巴斯基（Kaspersky, K.）著；谭明金译. —北京：电子工业出版社，2004. 10

（安全技术大系）

书名原文：Hacker Disassembling Uncovered

ISBN 7-121-00206-X

I. 黑… II. ①卡… ②谭… III. 计算机网络-安全技术 IV. TP393.08

中国版本图书馆 CIP 数据核字（2004）第 079160 号

责任编辑：宋沐纪

印　　刷：北京智力达印刷公司印刷

出版发行：电子工业出版社

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：787×980　1/16　印张：34.5　字数：684 千字

印　　次：2004 年 10 月第 1 次印刷

定　　价：59.00 元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。联系电话：（010）68279077。质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

译 者 序

说起“黑客”，人们会立即想到非法侵入计算机并窃取与破坏资源的计算机高手。这种在英文中称为“cracker”的黑客，满肚子“恶肝疠胆”，说他“黑”名副其实。然而，还有一种英文叫做“hacker”的黑客，却有一副算得上“红”的“侠肝义胆”。他想方设法突破安全保护机制的目的不在于窃取与破坏，而在于寻找和建立防止窃取与破坏的各种有效措施，被冠以黑客的恶名似乎有点儿冤。

不过，撇开人性因素而仅仅从纯粹的技术角度看，两种黑客的行为方式却没有本质上的差别，他们都以代码分析作为技术手段。这或许是在中文中将他们统统称为黑客的根本原因。代码分析技术用在黑客手中就成了所谓的黑客技术。概括地说，黑客技术指的是利用调试器与反汇编器等代码分析工具，跟踪、调试、分析与反汇编程序的机器指令，进而重建程序的算法初始源代码，并据此破除程序保护机制的综合性代码分析技术。

目前，黑客分析技术正方兴未艾。这不仅是因为计算机安全方面存在迫切的现实需要，而且因为代码分析技术为人们深入地学习程序设计技术提供了一条高效率的途径，同时，它还是研究开发高质量与高性能的系统软件（如编译器与操作系统）的有效手段。须知，诸如调试器与微处理器调试功能之类的技术，最初都是为开发高质量的驱动程序这样的软件而提供的。

显然，黑客技术是一门实践性强、覆盖面广的综合性技术。然而，它并不像人们想像的那样神秘莫测而遥不可及。市场上介绍黑客技术的书籍虽然很多，但能够将针对性、实践性与综合性很好地结合在一起的著作却不多见，而这对于学习黑客技术却是至关重要的。

与其他类似书籍相比，本书在内容上类比了几种常用编译器生成的代码形式，不失为“博”；其规模几乎涵盖了高级语言的所有语言要素，算得上“大”；所列举的实例代码短小典型，无愧于“精”；书中对每个知识点知无不言、言无不尽，可谓是“深”。它所具有的“博大精深”的特点，在很大程度上满足了人们学习黑客技术的需要。您不妨去阅读它，理解它，应用它，定知“吾言不虚也”。感谢电子工业出版社为我们引进了一本好著作。然而，我不得不加上一句：For your information only，believe it or not!

本书的作者 Kris Kaspersky 先生是黑客破译、反汇编与代码优化技术的专栏作家，更是一位涉猎颇多的“杂家”。其研究内容涵盖编译器开发、优化技术、安全机制研究、实时操作系统内核的创建以及反病毒程序的设计等诸多领域。可以认为，正是因为他虽“杂”却“博”、虽“博”却“深”，才能用诙谐而轻松的话语，把严密的科技知识在谈笑间透彻

地加以剖析，让读者在轻松愉快之中学习和体验科技的奥妙，这是一种特色、一种方式、一种态度，更是一种境界。

该书的翻译得到王璐、沈鑫剡、胡勇强、伍红兵、龙瑞、魏涛、汪东等同志的大力支持与协助，在此一并致谢。

由于黑客技术涉及的知识非常丰富，本书的实例含有很多的细节内容，加之译者水平有限，译文中必定存在疏漏和不当之处，请读者不吝赐教。

译　者

2004 年 6 月

前　言

本书通过展示如何创建与绕过保护环节等方面的内容，向读者开启了一扇通往安全机制这一神秘世界的大门。该书是为喜欢破译难解之谜的人们，以及将业余时间或者上班时间都花在对程序与操作系统进行刨根问底的人们写的，同时，它也是为经常或者偶尔从事保护程序开发的人员，以及想知道如何出色而可靠地阻止无处不在的黑客的人员写的。

黑客底层分析技术，即用调试器与反汇编器开展工作所需要的技能，是本书介绍的主要内容。书中详细介绍了识别与重建源代码的关键结构——函数（包括虚函数）、局部与全局变量、分支、循环、对象及其层次、数学运算符等方面的内容。

选取阅读本书时所需要的工具软件，完全由读者个人的喜好决定。每个人的爱好是很不一样的，因此，请不要把我在下面提到的那些东西看成板上钉钉的事，只管将它们当做一些建议好了。使用本书需要的工具软件包括：

- 调试器——**SoftIce**，版本在 3.25 以上
- 反汇编器——**IDA**，版本为 3.7*x*（推荐使用 3.8 版本，4.*x* 版本更佳）
- 十六进制编辑器——**HIEW**，任何版本都行
- 开发包——**SDK** 与 **DDK**（后者不是必需的，但拥有它的确很好）
- 操作系统——任何版本的 Windows 操作系统都行，但强烈推荐使用 **Windows 2000** 或者更高的版本
- 编译器——读者最喜欢的任何 C/C++或者 Pascal 编译器（书中虽然对 Microsoft Visual C++、Borland C++、Watcom C、GNU C 以及 Free Pascal 等编译器各自具有的特点都进行了详细描述，但用得最多的只是 Microsoft Visual C++ 6.0 编译器）

现在，我们来比较详细地介绍一下这些内容：

- SoftIce。SoftIce 调试器是黑客们使用的主要武器。虽然有一些免费的调试器（比如 Microsoft 的 WINDEB 与刘涛涛的 TRW）可供使用，但 SoftIce 比所有这些免费程序提供的功效加在一起还要有效与方便。几乎所有版本的 SoftIce 都能满足用户的需要，作者使用的是 3.26 版本。这是一个经受了时间考验的版本，它具有很好的稳定性，并且在 Windows 2000 下运行良好。该软件的当前版本 4.*x* 与我计算机上的电视卡（Matrox Millennium G450）不能在一起很好地运作，大致说来，随着运行时间的推移，它会出现死机。除此以外，在第四个版本的所有新增功能中，只有“参考点省略”（FPO，Frame Point Omission）这一部分功能（参见“局部堆栈变

量”一节）才对通过 ESP 寄存器直接寻址局部变量的程序员特别有用，这无疑是一个很实用的特性，然而，如果一定要实现类似的操作，没有它同样可以做到。去购买 SoftIce 调试器吧，读者是不会遗憾的。（破译程序的工作与盗版行为可不是一回事，直到今天，从事这类工作的人们都还算得上是诚实的。）

- IDA Pro。目前世界上功能最强的反汇编器无疑是 IDA。没有 IDA 固然可以生存，但有了它肯定可以生活得更好。IDA 为浏览所剖析的内容提供了一个便利的工具，它自动识别库函数和局部变量（包括通过 ESP 直接寻址的那些变量），并支持众多的处理器与文件格式。一句话，黑客要是不配备它就不能称为黑客了。但是在我看来，为它大做广告其实是不必要的。惟一的问题是，读者如何才能得到这个 IDA 工具程序。加进该软件程序的盗版是十分罕见的（我看到过的最新版本是 IDA 3.74，不过它显得不够稳定），提供它的 Internet 网站通常更少。IDA 开发人员会迅速阻止任何非授权性质的产品分发企图。获取该软件的惟一可靠途径是从开发人员（**http//www.idapro.com**）或者正式发行商那里购买。不幸的是，该软件并不提供配套的文档资料（非常简短且不系统的即时帮助信息除外）。
- HIEW。HIEW 不仅仅是一个十六进制的编辑器，它还将反汇编器、汇编器与编码器的功能集于一身。HIEW 虽然不能取代 IDA 而使用户不必去购买 IDA，但在某些情况下，它会带给用户很多的回报。（IDA 运行起来很慢，因此，假如用户仅仅是想快速地看一眼待用文件的话，则浪费大量的时间确实是件令人头疼的事情。）不过，HIEW 的主要用途不在于反汇编，而在于进行一些细小的修剪——针对二进制文件进行一些小小的外科手术。这样做的目的通常是去除部分保护机制，从而使该工具能够正常地运行。
- SDK（软件开发套件——应用程序开发包）。需要从 SDK 包获得的主要资源是关于 Win32 API 函数以及用于 PE 文件的 DUMPBIN 实用工具等方面的文档资料。没有这些资料，无论是黑客还是开发人员什么都做不了。至少，用户需要通过文档资料来了解主要系统函数的原型和用途。这类信息虽然可以从浩如烟海的编程书籍中收集而来，但是没有哪本书能够宣称它所展示的内容达到了必要的深度和广度，所以说，用户迟早要去使用 SDK。怎样去获取 SDK 呢？SDK 是 MSDN 的一部分，MSDN 按季度以光盘的形式进行发布，同时也可以通过征订来分发。（用户可以通过官方网站 http//msdn.microsoft.com 来了解关于订阅条件方面的内容。）此外，MSDN 也随 Microsoft Visual C++ 6.0 编译器一起提供（这个版本虽然不是特别新，但它对于阅读本书已经足够了）。
- DDK（驱动程序开发套件——驱动程序开发包）。黑客使用 DDK 开发包做什么？它有助于弄清楚，驱动程序是如何开发、工作，以及如何被攻击的吗？除了基本的文

档和大量的样例之外，DDK 包还包括一个非常有价值的文件 NTDDK.h，该文件含有绝大部分非公开结构的定义，它同揭示系统某些非一般性操作细节的注释内容一起加载。随 DDK 一起提供的一些实用程序也是有用的，其中，WINDEB 调试器就是 DDK 所包括的实用工具之一，这是一个相当好的调试器，但无论从什么角度看，它都难及 SoftIce 的项背，因此，本书不考虑它的使用。（如果读者找不到 SoftIce，WINDEB 也凑合能用。）用于编写驱动程序的 MASM 汇编工具显得十分有用，另外，DDK 还有一些使黑客的生活变得稍微容易点儿的小程序。最新的 DDK 版本可以从 Microsoft 站点免费下载。不过，要记住一点，用于 NT 的整个 DDK 的大小超过了 40MB（压缩的），它甚至要求磁盘上存在更多的空间。

- 操作系统。我无意将自己的品味和嗜好强加于读者，不过，我还是强烈地建议读者安装 **Windows 2000** 或者更高的版本。这样做的原因在于它是一个性能非常稳定且运行牢靠的操作系统，能够抵御严重的应用程序错误的破坏。与黑客的工作紧密相关的一件事情是，这种针对程序的核心层所施加的外部干扰使程序非常容易陷入崩溃而出现难以预料的行为。Windows 9*x* 操作系统和崩溃程序的频繁“罢工”表现出“高度一致”的步调，有时候，计算机请求启动高达几十次之多！如果启动频繁发生而读者又不用去修复因为故障而遭到破坏的磁盘的话，那只能说用户运气不错。（这样的事情虽然很少发生，但还是存在发生的必然。）冻结 Windows 2000 要困难得多，在我睡眠不好或者疏忽大意的时候，我“成功”实施冻结的次数一个月也不会多于两次。而且，Windows 2000 允许用户在任何时候加载 SoftIce 而不用重新启动系统，这实在是太方便了！最后要说明的一点是，本书的所有内容都是基于 Windows 2000 或者更高版本而展开叙述的，作者也很少提及其他系统所表现出的不同之处。

本书的读者需要事先对汇编语言很熟悉。即使没有用汇编语言编写程序的经历，至少也应该知道诸如寄存器、段、机器指令等概念指的是什么，否则，阅读本书很可能感到太复杂而难于理解。作者的建议是，读者去找一本汇编方面的指南书籍，然后进行系统的学习。

除了汇编语言，读者至少还应该具备操作系统方面的一般概念。

此外，从 Intel 和 AMD 站点上下载所有可以获得的关于处理器的文档资料，也是很有用的。

至此，已经准备足够素材，该是起步向前的时候了！

目　录

第1部分　精通黑客基本技术

第 2 部分 提高软件分析难度的技术途径

第1部分
精通黑客基本技术

CHAPTER

1

第1章 概 述

1.1 保护方式分类

身份鉴定是具有压倒性多数的保护机制所要处理的中心内容。无论在什么情况下，都必须保证使用程序的人就是他或者她宣称的那个人，并确保此人具有使用程序的授权。“人”这个字眼并不仅仅指用户，它还表示用户的计算机或者存储程序合法拷贝的介质等。据此，可以将所有的保护机制分为以下两个主要的类型：

- 基于知识（密码、序列号等）的保护
- 基于财产占有关系（钥匙盘、文档资料等）的保护

基于知识的保护在合法物主对保守秘密失去兴趣的情况下是无用的。物主可能将密码（或者序列号）交给他喜欢的任何人，从而使拥有这些知识的任何人都可以在他的计算机上使用带有这类保护的程序。

因此，基于知识的保护对于防止非法拷贝是没有效果的，但是，为什么实际上所有的知名软件开发商都使用序列号呢？答案很简单——使他们的知识产权避免受到强力的侵袭（不管这种威胁是多么的不可能）。该想法大致上基于这样的考虑：某公司宁静的办公场所突然闯进一伙经过化装的代理人，把 Windows 的许可证号（Microsoft Office、Microsoft Visual Studio）与特许协议进行比较，如果他们发现哪怕只有一份非法拷贝，某个官员就会像是从外面地缝里冒出来的一样突然出现在面前，并对一直希望出现的意外收获大喜过望。最好的情况是，他们只强迫公司买下所有的非法拷贝。最坏的情况就……

很自然，不会有人去闯用户的家，甚至连这样的想法也不会有（至少目前没有）——您的家终究是您的地盘。再说，能向家庭用户收取什么呢？对于开发商来说，产品的广泛

分发是一件好事，而谁又能比盗版人分发得更好呢？即使在这种情况下，序列号也不会显得多余——非注册用户不能得到技术支持，这可能会促使他们购买合法的软件版本。

这种类型的保护虽然对于大型公司而言是很理想的，但它不能适用于小型程序员团队或者个体开发人员。特别是，如果他们靠为有限的市场（比如说，星谱分析或者核反应建模）编写高度专业化的程序来谋生的话，情况更是如此。由于没有强制性手段可用，他们自己去要求用户出示使用许可授权是不切实际的，要从非法用户那里“榨出”一些费用也几乎是不可能的。他们所能够做的一切，只是借助施压与雄辩来讨个“公道”。

在这样的情况下，那种基于在一般意义（理想情况）上特别难于拷贝或者不可能拷贝的主体占有关系（占有关系具有惟一性）而施加的保护，会显得更加合适一些。使用这类保护机制的第一种形式是钥匙软盘，该软盘具有按某种使软盘不能复制的方式而记录的信息。准备这种软盘最简单（但不是最好）的方法是用一颗钉子（一根锥子或者一把小刀）轻轻地扎一下磁盘，然后找出具有缺陷的扇区（通过读写任何测试信息来确认——在到达“垃圾”数据存放点之前，阅读操作一直正常地进行），并在程序中把该扇区记录下来。于是，程序每次启动时，都会检查缺损是否出现在同样的地方。由于软盘的应用越来越少，这样的技术已经开始在光盘上使用，比较有钱的人用激光来破坏光盘，而一般的人仍然要用锥子或者钉子来做同样的事情。

这样一来，程序就与盘片牢固地绑定在一起，要有盘片才能运行程序。由于复制该盘片是不可能的（惟一能做的是尽力在一份拷贝上形成同样的缺损），盗版人不得不放弃拷贝行为。

另外一些基于占有关系的保护机制频繁地修改占有的主体而限制程序的启动数目或者使用期限。这类机制通常用于安装程序，如果没有对应的“钥匙”，程序就不运行。如果安装次数受到限制，那么因为一份拷贝在多台机器上进行非授权安装而给开发商造成的损害是很小的。

令人遗憾的是，所有这些机制都剥夺了合法用户的权利。谁愿意自己购买的程序在安装次数上受到限制？（有人需要每个月一次乃至一天几次地安装操作系统或者软件。）此外，加锁盘并不能在所有类型的驱动器上使用，并且它们在网络上通常都是“看不见”的设备。假如为了更加有效地阻止黑客的攻击，保护机制可能绕开驱动程序而直接访问设备，那么这样的程序必定无法在 Windows NT/2000 下正常运行，即使在 Windows 9*x* 下也可能运行失败。（当然，这只是在起先设计不合理的情况下才会发生。但是，运行正常的情况可能更糟，原因在于保护机制以最高级别的权限运行，可能对系统的性能造成严重损害。）再说，“钥匙”也可能被遗忘、失窃或者不能正常地发挥作用。（软盘很容易消磁与形成坏簇，CD 盘可能被划坏，而电子钥匙则会发生“耗尽”。）主要保护类型如图 1 所示。

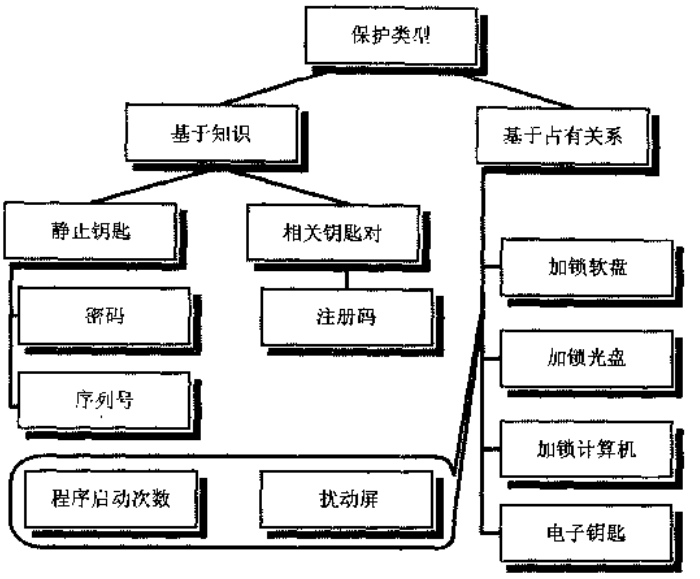

图 1　主要保护类型

很自然，这些考虑关心的是“钥匙”在防止黑客攻击方面的有效性，并不是一般意义上钥匙的概念。针对用户终端进行的保护绝不是一种很好的选择！如果因为保护引起不方便，用户可能宁愿光顾最近的盗版场所而购买非法软件，人格、道义与廉耻方面的说教已经起不到任何作用了，感到难为情的应该是你——程序开发人员！为什么要使用户的生活变得更加错综复杂呢！用户同样是人！

这表明，基于注册码的保护方式有必要得到普遍的应用：在首次运行之后，程序将自己与该台计算机捆绑在一起，打开一个“计数器”并不时去阻止程序的某些功能性操作。为了使程序发挥全部的功能，用户需要用金钱作为补偿，从开发人员那里换取密码而输入计算机。为了阻止盗版复制，密码一般是从用户计算机的关键参数衍生出来的（或者在比较初等的情况下是从用户名衍生出来的）。

当然，对保护机制进行的这种走马观花式的介绍省去了其中的许多类型，不过，对保护类型进行详细的讨论超出了本书的写作范围，我们把它留给另外一本书去完成。

1.2　保护强度

如果保护是基于代码不会被破译或者修改这样的假设，那么它只能算是很无力的保护机制。隐藏源代码对于破解与修改应用程序来说不是什么不可逾越的障碍，现代的逆向工

程技术可以自动识别库函数、局部变量、堆栈参量、数据类型、分支和循环等，并且，在不远的将来，反汇编器能够生成与高级语言类似的代码。

然而，即使在今天，分析机器代码也不像长期阻止黑客攻击方面的工作那样复杂，不断出现且数目惊人的破解实例就是最好的说明。最理想的情况是，泄露保护算法应该不至于影响保护的强度。不过，事情并不总是这样单纯，例如，假设服务器程序的演示版对同时连接的数目存在限制（通常都是如此），那么黑客要做的全部事情就是找到处理该检测的代码并进行删除就可以了。虽然通过经常性的测试校验，确实可以检测和阻止对程序的修改，但是，计算校验和并将它与特定值进行比较的代码却是可以找到并加以删除的。

不论施加了多少个保护层（一个或者一百万个），程序总是可以攻破的！这不过是时间和精力的问题。然而，当没有保护知识产权的有效法律时，软件开发人员必须更多地依赖保护措施而不是法律实施体系来维护自己的权益。有一种普遍的看法是，如果突破保护机制的开销比进行程序合法拷贝所需要的费用还要高时，就没有人去破解它了，这是不对的！物质利益并不是黑客的惟一追求，更强烈的动机似乎在于智力较量（谁更聪明：是你保护机制开发人员还是黑客我？）、竞赛（哪个黑客能够攻破更多的程序？）、好奇（是什么驱使保护机制发挥作用的？）、提高自己的技能（在能够建立良好的保护机制之前，首先需要学会怎么去破解它），以及仅仅是作为一种消磨时间的娱乐方式。许多年轻黑客会花几个星期的时间去消除程序的保护机制，但该程序只不过需要几美元就可以买到，有时甚至是免费供应的。

保护的使用限制了它的竞争力，如果说其他方面完全一样，那么用户总是愿意去选择不加保护的产品，即使所施加的保护并不影响顾客的权利。目前，虽然程序员显得非常供不应求，但在遥远的未来，开发人员应该是要么在进行程序保护方面达成一项协议，要么干脆不加保护，这样一来，安全专家将不得不去找其他的活干了。

当然，这不意味着本书没有用途了，相反，应该尽可能快地对书中提供的知识加以应用，因为保护方面的需求目前还未见有消失的征兆。

CHAPTER

2

第2章　第一步：热身

最简单的鉴定算法含有一个将用户输入的密码与参照值逐字符进行比较的操作过程，参照值要么存放在程序中（通常情况就是这样），要么保存在程序之外——比如，放在配置文件或者注册表里（这种方式用得较少）。

这类保护措施的优点在于软件的实现特别简单，其真正的核心只不过一条代码而已。该条代码可以用C语言写成：

```
if (strcmp(password entered, reference password))
{
   /*密码不正确*/
}
else
{
   /*密码正确*/
}
```

下面，首先用提示用户输入密码和显示比较结果的过程对该代码进行补充，然后检测程序抵御攻击的脆弱性。

列表1　最简单的鉴定系统

```
//逐字符匹配密码

# include <stdio.h>
# include <string.h>
```