第33卷 第22期 计 算 机 工 程 2007年11月
Vol.33 No.22 Computer Engineering November 2007
·开发研究与设计技术· 文章编号：1000—3428(2007)22—0274—03 文献标识码：A 中图分类号：TP333

一种真实主引导扇区的备份方法

李全忠，张军本
(山东农业大学信息学院，泰安 271000)

摘 要：针对硬盘故障对程序和数据的严重危害，该文分析了受保护卡或还原软件保护的硬盘其主引导扇区不能被常规方法读出的原因，给出了一种新的切实可行的读取办法和 Delphi 实现程序，解决了主引导扇区的备份问题。
关键词：中断(异常)；硬盘保护卡；还原软件

Method to Backup Real MBR

LI Quan-zhong, ZHANG Jun-ben
(College of Information, Shandong Agricultural University, Taian 271000)

【Abstract】 Aiming at severe harm of hard disk for programs and data, this paper analyzes the causes that the MBR of the hard disk protected by hard disk guard or recovery program can not be read by the usual methods, and gives a new viable reading method and its Delphi realizing program. The problem to backup the MBR is solved.
【Key words】 interrupt (exception); hard disk guard; recovery program

1 概述

众所周知，硬盘的主引导扇区(又称主引导记录区 MBR)位于硬盘的 0 柱 0 头 1 扇区，是硬盘的一个重要而又特殊的扇区，它除含有系统启动时的引导程序外，还含有描述整个硬盘划分布局的分区表等重要信息。也就是说，若这个扇区遭遇到攻击或破坏，就有可能导致整个硬盘系统瘫痪，造成其中的程序或数据无法挽救的结局。因此，出于安全方面的考虑一些用户总是适时地对其制作一个备份，以便不测时再对其进行恢复，从而有效避免一些灾难性事件的发生。

在 Win9X环境下，进行主引导扇区的备份工作并非难事，比如可利用WinHex、KV系列杀毒软件的相关功能进行操作，也可以在Win9X的虚拟DOS环境下通过DEBUG调用INT13h进行读操作[1]，等等。

然而，当系统安装硬盘保护卡或安装某种还原软件后，再利用诸如上述的常规方法去备份主引导扇区时，奇怪的现象便发生了(一般用户往往难以觉察到)：此时备份出来的不是真实的主引导扇区，即不是 0 柱 0 头 1 扇区，而是某一特定的其他扇区，称它为“虚拟主引导扇区”。而对主引导扇区进行恢复操作时，出现完全类似的情况。

分析表明，当硬盘保护卡或某种还原软件进行安装和设置时，它们一般要改写 0 柱 0 头 1 扇区的代码部分，以便在系统启动时抢夺控制权，再根据用户的设置参数改写分区表信息，之后便将其保护起来。而之前的那个主引导扇区被复制后其分区表信息也作如上同样的改写(注意：仅改写分区表信息)，然后被保存在 0 柱 0 头 1 扇区之后的某个保留扇区中，它就是虚拟主引导扇区[2]。

显然，在系统处于正常状态时，用户对主引导扇区的备份与恢复，被更改为对虚拟主引导扇区的相应操作；而当真实的主引导扇区遭到破坏(相应的控制权丧失)后进行恢复操作时，所进行的操作是以虚拟主引导扇区(的备份)替代真实的主引导扇区，可以想像，其结果不是人们所希望的。那么，如何备份出真实的主引导扇区？下面以 Win9X 环境为例进行分析探讨。

2 读取真实主引导扇区的基本思路

分析表明，硬盘保护卡与还原软件的工作原理基本类似，只是介质不同。其还原原理大致上可以概括为：(1)将原中断 INT13H 的入口地址保存，并将入口地址指向自己的 INT13H 处理程序，同时通过自身的监控措施，确保该入口地址始终指向自己的 INT13H；(2)拦截所有 INT13H 中对硬盘 0 柱 0 头 1 扇区的操作，把所有对 0 柱 0 头 1 扇区的读写操作改为对虚拟主引导扇区的操作；(3)拦截所有 INT13H 中的写硬盘操作，把所进行的写操作在虚拟内存中进行，并做好完整的记录，等系统重新启动后还原这些记录。

显然，在硬盘保护卡或还原软件有效运行的条件下，那些最终通过 INT13H 来读取真实主引导扇区的想法是不可行的，这就是常规办法(或软件)失效的根本原因。

于是，自然会想到通过对硬盘控制器编程这种办法，来读取真实的主引导扇区。但实践表明，对硬盘控制器编程的I/O指令在应用程序级别上是不可以使用的，它们被限制在更高的特权级别上使用。这是因为，从 80386 以后，CPU分为 4 个特权级别(即Ring0～Ring3)，供操作系统使用，其中Ring0的级别最高，Ring3 的级别最低。Win9X在设计时只使用了CPU的 2 个特权级别，即Ring0 和Ring3。像Win9X的虚拟机管理、各种驱动程序(VxD)和中断(异常)处理程序等运行在Ring0 级，其他应用程序，甚至包括KERNEL、GDI、USER 3个主要模块在内都运行在Ring3 级[3]。

进一步的分析表明，Win9X 的运行管理机制允许 Ring3 级上的应用程序，通过 CPU 的中断(异常)获取在 Ring0 级的运行权限。因此，在 Win9X 平台上和硬盘保护卡或还原软件

作者简介：李全忠(1961－)，男，副教授，主研方向：系统分析与网络数据库开发；张军本，博士、副教授
收稿日期：2006-11-30 **E-mail：**lqz@sdau.edu.cn

有效运行环境下，读取真实主引导扇区的基本思想是：(1)通过中断(异常)的办法进入 Ring0 级；(2)通过对硬盘控制器编程读取该扇区。

3 硬盘控制器编程

对硬盘控制器编程，就是直接向 IDE 寄存器(即端口)设置参数，从而达到读写硬盘的目的。对主硬盘进行操作的常用端口是 1F0h-1F7h 号端口，各端口含义如表 1 所示。

表 1 各端口含义

端口	操作	含义
1F0H	读/写	用来传送读/写的数据(其内容是一个字的数据)
1F1H	读	用来读取错误码
1F2H	读/写	用来放入要读写的扇区数量
1F3H	读/写	用来放入要读写的扇区号码
1F4H	读/写	用来存放读写柱面的低 8 位字节
1F5H	读/写	用来存放读写柱面的高 2 位字节(其高 6 位恒为 0)
1F6H	读/写	用来存放要读/写的磁盘号及磁头号
		第 7 位　恒为 1
		第 6 位　恒为 0
		第 5 位　恒为 1
		第 4 位　为 0 代表第 1 块硬盘、为 1 代表第 2 块硬盘
		第 3-0 位　用来存放要读/写的磁头号
1F7H	读	用来存放读操作后的状态
		第 7 位　控制器忙碌
		第 6 位　磁盘驱动器准备好了
		第 5 位　写入错误
		第 4 位　搜索完成
		第 3 位　为 1 时扇区缓冲区没有准备好
		第 2 位　是否正确读取磁盘数据
		第 1 位　磁盘每转一周将此位设为 1
		第 0 位　之前的命令因发生错误而结束
	写	该位端口为命令端口，用来发出指定命令
		为 50h　格式化磁道
		为 20h　尝试读取扇区
		为 21h　无须验证扇区是否准备好而直接读扇区
		为 22h　尝试读取长扇区(用于早期的硬盘)
		为 23h　无须验证扇区是否准备好而直接读长扇区
		为 30h　尝试写扇区
		为 31h　无须验证扇区是否准备好而直接写扇区
		为 32h　尝试写长扇区
		为 33h　无须验证扇区是否准备好而直接写长扇区

需要指出的是，当系统中有两块硬盘时，通常第一 IDE 接口上的硬盘(主硬盘)端口为 1F0h～1F7h，而第二 IDE 接口上的硬盘端口为 170h～177h，其端口的含义同前者。

4 获取 Ring0 级运行权限

Win9X 自己提供了中断(异常)处理程序，且当中断(异常)发生时，便转到 Ring0 级去执行相应的处理程序。Win9X 的这种运行管理机制，正好为用户提供了一种从 Ring3 级向 Ring0 级转化的手段。

具体地说，在应用程序中，先将那些包含了被限制在 Ring0 级上使用的指令系列，设计成一个中断(异常)处理程序，如 INT 3，接着由应用程序为该中断(异常)设置好入口地址，然后调用它。这样，在应用程序中执行 INT 3 指令时，会产生 CPU 异常，于是系统就自动到 Ring0 级执行 INT 3 处理程序。从而达到了在 Ring3 级上执行 Ring0 级指令的目的。

其中，设置入口、调用中断(异常)的逻辑过程是：(1)获取中断描述表的基地址；(2)计算原 INT *n* 的中断门的基地址；(3)读出并保存原 INT *n* 的入口地址；(4)获取新 INT *n* 的入口地址；(5)将新 INT *n* 的入口地址写入原 INT *n* 的中断门的入口地址中；(6)调用新 INT *n*；(7)恢复原 INT *n* 的入口地址到相应的中断门中。

5 读取真实主引导扇区的实现程序

下面是用 Delphi 编写的实现程序，它首先读取真实主引导扇区的内容并暂存入数组中，然后再将数组中的内容保存为数据文件，其路径和文件名为“C:\BootReal.dat”。

其中，数组变量 Sector_001 为模块变量，用于存放主引导扇区的内容，过程 Backup001 是主程序，过程 Read001 被定义为 3 号中断(异常)处理程序，供主程序产生中断(异常)时调用。

这两个过程完全是按照前面分析的逻辑框架编写的，而且代码比较简炼，实现的具体细节都有详细的注释，所以比较易于理解和使用。

```
var
Sector_001:array[0…255] of WORD; //暂存 0 柱 0 头 1 扇区内容
procedure TForm1.Backup001;
var
OInt3_ADD:dword; //保存原先的 3 号中断入口地址
Idtr_Save:array[0…5] of byte; //保存中断描述符表寄存器
FileHandle:Integer; //文件句柄
begin
    asm
    push    eax
    push    ebx
    push    ecx
    //获取中断描述符表基地址
    sidt Idtr_Save
    //获取原先的 3 号中断门描述基地址
    mov    eax,dword ptr Idtr_Save+02h
    add    eax,3*08h
    //保存原先的 3 号中断入口地址
    cli
    mov    ecx,dword ptr[eax+04h]
    mov    cx,word ptr[eax]
    mov    dword ptr OInt3_ADD,ecx
    //设置新的 3 号中断入口地址
    lea    ebx,Read001
    mov    word ptr[eax],bx
    shr    ebx,10h
    mov    word ptr[eax+06h],bx
    //执行新的 3 号中断，即在 Ring0 级
    //执行 Read001 过程
    int    3
    //恢复原先的 3 号中断入口地址
    mov    ecx,dword ptr OInt3_ADD
    mov    word ptr[eax],cx
    shr    ecx,10h
    mov    word ptr[eax+06h],cx
    sti
    pop    ecx
    pop    ebx
    pop    eax
    end;
    //将扇区数据形成“C:\BootReal.dat”数据文件
    FileHandle:=filecreate('C:\BootReal.dat');
    filewrite(FileHandle,Sector_001,512);
    fileclose(FileHandle);
end;
procedure TForm1.Read001;assembler;
begin
    asm
    push    eax
    push    ebx
```

```
push    ecx
push    edx
push    edi
mov    dx,1f6h       //要读入的磁盘号及磁头号
mov    al,0a0h       //磁盘 0,磁头 0
out    dx,al
mov    dx,1f2h       //要读入的扇区数量
mov    al,1          //读一个扇区
out    dx,al
mov    dx,1f3h       //要读的扇区号
mov    al,1          //扇区号为 1
out    dx,al
mov    dx,1f4h       //要读的柱面的低 8 位
mov    al,0          //柱面低 8 位为 0
out    dx,al
mov    dx,1f5h       //柱面高 2 位
mov    al,0          //柱面高 2 位为 0
out    dx,al
mov    dx,1f7h       //命令端口
mov    al,20h        //尝试读取扇区
out    dx,al
@@L1:
in     al,dx
test al,8            //扇区缓冲是否准备好?
jz     @@L1          //直到扇区缓冲准备好才向下执行
mov    ecx,512/2     //设置循环次数(512/2 次)
lea    edi,Sector_001 //数组变量首地址
mov    dx,1f0h       //16 位数据寄存器
    @@L2:
    in     ax,dx          //从扇区缓冲读取 16 位数据
    stosw                 //保存 16 位数据到数组变量
    loop   @@L2
    pop    edi
    pop    edx
    pop    ecx
    pop    ebx
    pop    eax
    iretd
    end;
end;
```

6 结束语

以上程序，分别在装有硬盘保护卡和还原软件的微机上运行通过。它为及时备份主引导扇区和分析其中的内容提供了方便。另外，利用本文所给出的思想方法可以实现对其他硬件的编程与操作，特别是对该程序稍作修改，便可实现恢复(或改写)真实主引导扇区的功能。

参考文献

1 熊桂喜. Windows95 技术内幕[M]. 北京: 清华大学出版社, 1996.

2 李全忠, 刘长文. 实模式下获取磁盘文件存储地址的一种捷径[J]. 小型微型计算机系统, 2004, 25(4): 774-776.

3 李全忠, 王希超. 直接读取PCI网卡的MAC地址的原理与方法[J]. 计算机工程, 2005, 31(18): 213-215.

(上接第 265 页)

参考文献

1 鞠晓东. 千道脉冲幅度分析器设计与应用[J]. 核电子学与探测技术, 2006, 26(1): 5-8.

2 王南萍, 裴少英, 黄　英, 等. 环境 γ 能谱测量方法研究及应用[J]. 辐射防护, 2005, 25(6): 347-356.

3 贾文懿, 方　方, 唐　红, 等. 核地球物理仪器[M]. 北京: 原子能出版社, 1998.

4 Pico Envirotec Inc. Instrumentation/Gamma Ray Spectrometer[EB/OL]. (2006-08). http://www.picoenvirotec.com/html/ gamma_spectrometers.html.

5 Terraplus Inc. PRODUCTS/GR 320 Programmable Spectrometer[EB/OL]. (2006-08). http://www.terraplus.ca/products/radiat/g320. htm.

6 肖无云, 魏义祥, 艾宪芸, 等. 数字化多道脉冲幅度分析技术研究[J]. 核技术, 2005, 28(10): 787-790.

7 Measurement Computing Corp. PCI-DAS4020/12 User's Guide[Z]. 2002.

(上接第 273 页)

输带宽，更好地保证了 Soft Phone 的通话质量和效率。LZW 压缩算法也具有一定的局限性，有待进一步研究，比如当输入流内容冗余较少或者输入流尺寸太大，将导致内部维护的字节串表(String Table)很快充满，会使它很快丧失压缩能力，但这种缺陷并不是完全无法避免的，有些 LZW 变种压缩算法(如 ARC)通过适当清空字节串表，来避免字节串表被充满，从而能够有效提高压缩率。未来网络趋向使用简洁有效的协议，具有组建灵活、扩展方便、移动性支持良好等优点的 SIP 必将在音频视频网络通信中得到广泛的应用。

参考文献

1 SIP: Session Initiation Protocol[S]. RFC 3261, 2002.

2 萨洛蒙. 数据压缩原理与应用[M]. 2 版. 北京: 电子工业出版社, 1999.

3 袁占亭, 张秋亲, 冯　涛, 等. 数据通信中文本文件无损压缩算法的实现[J]. 计算机工程与应用, 2001, 37(9).

4 黄贤武, 王加俊, 李家华. 数据图像处理与压缩编码技术[M]. 成都: 成都电子科技大学出版社, 2000.

5 王　平. LZW 无损压缩算法的实现与研究[J]. 计算机工程, 2002, 28(7): 98-99.