第3章

Windows操作系统取证实训

电子数据主要存储在计算机中，Windows 目前是在普通用户的计算机上使用最广泛的操作系统，占据了中国绝大多数个人计算机用户的操作系统市场，因此针对 Windows 系统的电子数据取证始终是电子取证研究方向的主流。基于 Windows 系统的计算机中可能包含着与犯罪事件相关的重要数据。这些数据隐藏在计算机的深处，如用户身份、犯罪记录、不良数据等信息。本章通过相关的实验，读者能够掌握 Windows 操作系统中重要数据的存放位置和主要的取证方法。

3.1 易失性数据提取

3.1.1 预备知识：易失性数据

一般来说，从计算机证据的时态性分类，可将计算机证据分为两种：持久性数据和易失性数据。持久性数据是储存在本地硬盘上的数据，当计算机关闭时会保存下来。易失性数据是当电脑断电或关闭时，会丢失的数据。传统的取证调查方法是在存储介质上进行事后检查，即将可疑的计算机接上电源以后再进行数字证据的搜索与获取。但是，随着硬盘加密技术，反取证工具与技术的不断发展，以及内存容量的不断扩展，传统的事后取证方法会导致丢失包含在易失性数据中的有价值的证据，而这些数据对于确定计算机犯罪活动往往是十分重要的。易失性数据驻留在注册表的缓存和随机访问内存(RAM)中，对易失性数据的调查称为“实时取证”。

易失性数据主要包含：

(1) 描述计算机基本配置信息的系统概要文件。如：计算机操作系统的版本、型号、安装时间、系统目录、系统注册用户、物理内存、安装的硬件及其配置和安装的应用软件等。

(2) 网络连接状况及路由信息。

(3) 当前系统的日期、时间等记录。

(4) 计算机从上一次启动到现在一共运行的时间，用于确定收集的易失性数据是否具有一定的价值。

(5) 当前系统运行的进程列表，可能会发现一些恶意进程、未授权的软件及已终止的合法进程。

(6) 登录用户最近的活动记录。

(7) 启动文件和剪贴板中的数据等。

在涉网案件的现场勘验过程中,应该首先处理会很快消失的电子数据,即易失性数据。

3.1.2 实验目的与条件

1. 实验目的

通过本实验,读者在了解了电子数据取证的基本流程及规范的基础上,熟悉计算机中易失性数据的种类,掌握使用常用软件工具,进行涉网案件现场易失性数据提取的常用方法和注意点。

2. 实验条件

本实验所需要的软硬件清单如表 3-1 所示。

表 3-1 易失性数据提取实验清单

序 号	设 备	数 量	参 数
1	取证工作站	1 台	Windows XP 以上
2	工具 U 盘(内含一些软件)	1 个	包含绿色版工具
3	屏幕录像机(oCam). exe	1 个	绿色版
4	MD5. exe	1 个	绿色版
5	clipbrd. exe	1 个	绿色版
6	DumpIt. exe	1 个	绿色版
7	systeminfo. exe	1 个	绿色版

3.1.3 实验过程

在取证过程中,首先要准备一个专用的取证工具 U 盘,如 E 盘,里面包含常用的取证工具,如 cmd. exe、MD5Checker. exe、systeminfo. exe、DumpIt. exe 等。然后按下面的方法进行数据的收集,最后将所有数据都保存到工具 U 盘中。

步骤 1:将手表或手机时间界面置于计算机前方,对照计算机右下角时间信息进行拍照,完成计算机时间信息提取。

步骤 2:插入 U 盘,运行 U 盘中的绿色版屏幕录像软件,打开软件设置,将后续快照、录像文件保存路径修改为该 U 盘,如图 3-1 所示。

图 3-1 修改生成文件的保存路径

步骤 3：单击录制按钮对后续电脑操作进行全程录像和截屏保存。

注意：操作过程中不能在硬盘上进行写入或修改操作，不得将生成、提取的数据存储在原始存储媒介中，不得在目标系统中安装新的应用程序。

步骤 4：在 U 盘中新建文件夹，分别存放内存提取文件、屏幕信息提取文件、硬盘及操作系统信息文件、正在运行有密码保护的文件、录屏及摄像文件等，如图 3-2 所示。

图 3-2　在 U 盘中新建文件夹

步骤 5：打开 U 盘中的内存提取工具，将获取的内存保存在 U 盘相应文件夹中，如图 3-3 所示。

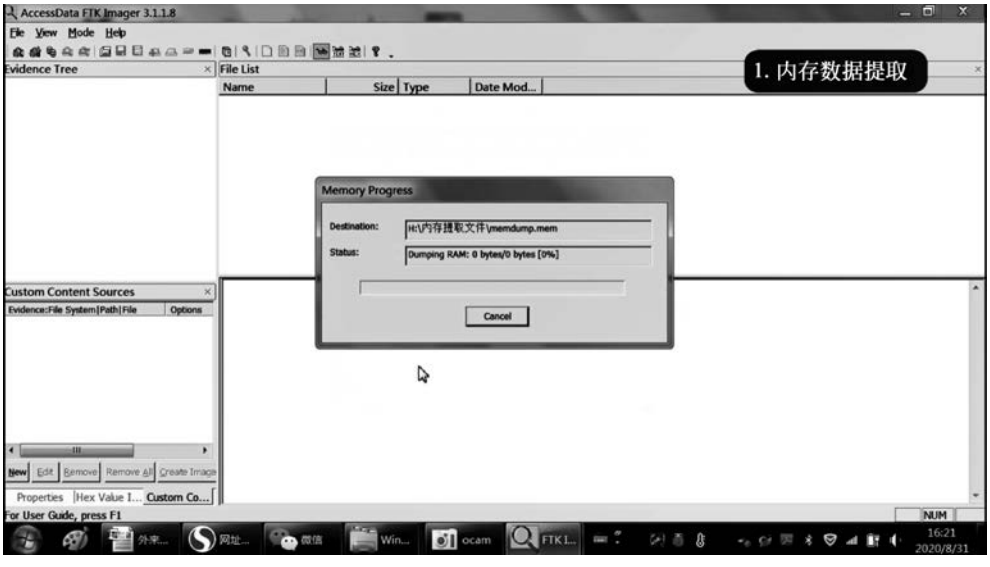


图 3-3　获取计算机运行内存

步骤 6：查看硬盘分区状况、文件显示属性、网络连接信息等，并进行录屏和截屏，保存在 U 盘相应文件夹中，部分如图 3-4、图 3-5 所示。

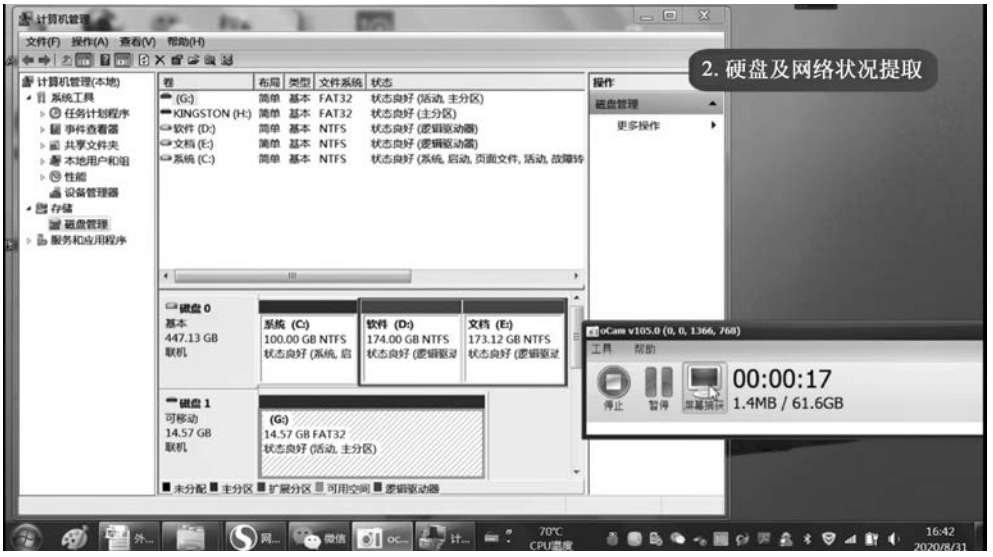


图 3-4　获取硬盘分区状况

图 3-5　获取网络连接状态信息

步骤 7：用录像和截图软件进行屏幕信息提取，提取打开的文件信息，并保存在 U 盘相应文件夹中，如图 3-6 所示。

步骤 8：运行 U 盘中的粘贴板查看器，提取粘贴板信息，并保存在 U 盘相应文件夹中，如图 3-7 所示。

步骤 9：提取浏览器、网页相关信息，并保存在 U 盘相应文件夹中。

步骤 10：对正在运行的有密码保护的数据，如微信、QQ、邮箱、网银、支付宝等，进行相关备份文件提取，并将其保存到 U 盘相应文件夹中，如图 3-8 所示。

步骤 11：将录屏文件、摄像录屏文件保存在 U 盘相应文件夹中，如图 3-9 所示。

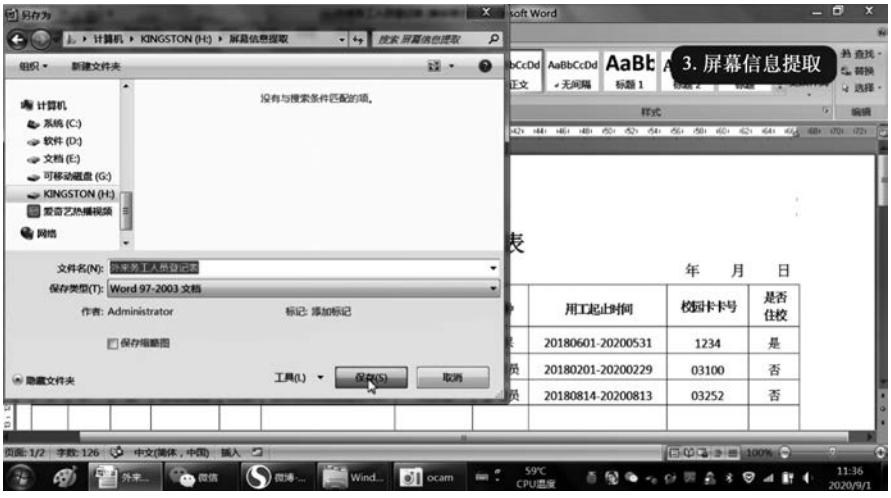


图 3-6　提取打开的文件信息

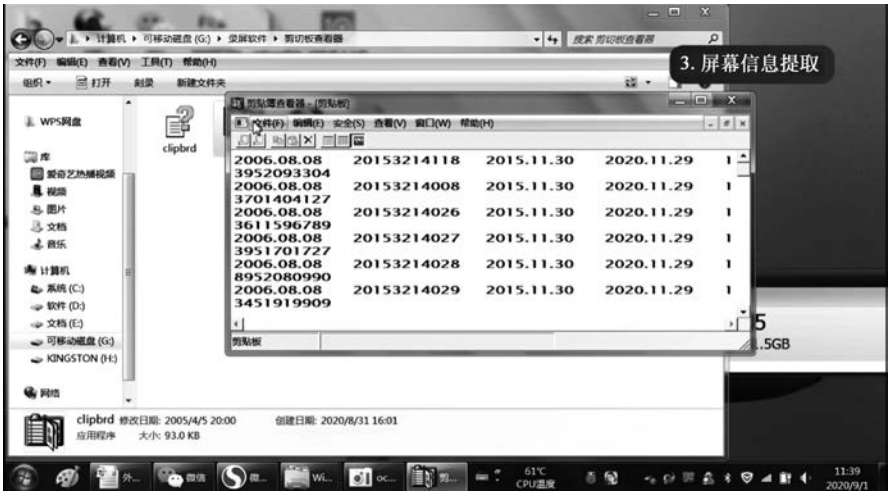


图 3-7　提取粘贴板信息

图 3-8　提取微信备份文件

图 3-9　保存录屏及摄像文件

步骤 12：使用 U 盘中的 MD5 工具，对录屏文件及所有提取的电子数据进行哈希值计算，记录文件类型、用户名、密码、路径来源及哈希值，如图 3-10 所示。

图 3-10　哈希值计算

注意：易失性数据提取完成后，对于保存数据信息的专用存储介质和现场录像等，进行唯一性编号并封存。

3.1.4　实验小结

传统的在计算机犯罪中所使用的取证流程大多数为关闭涉案计算机后，使用即插即用设备按字节流完全复制计算机的磁盘数据建立磁盘镜像，然后在实验室中对镜像数据进行事后分析。然而，随着计算机硬件水平的不断发展，大容量的内存广泛被使用，同时各种加

密与反取证技术的出现，导致在这样传统的取证过程中损失了大量有价值的信息。

计算机的易失性数据中可能包含关于犯罪行为的关键性信息，如用来加密信息所使用的密码，系统在犯罪行为发生过程中的状态，使用反取证工具的痕迹以及一些调查者在分析硬盘数据过程中容易忽略的至关重要的恶意软件或系统级后门程序等相关信息。所以近年来针对计算机易失性数据的取证分析工作越来越受到司法界和计算机安全专家的重视。

传统上一般将获取易失性数据的方法分为两类：一类基于硬件设备实现；另一类基于软件方法实现。硬件设备获取内存镜像被业界广泛认为具有更高的安全性和可靠性，而软件运行时必然会导致内存中部分数据发生变化，可能会影响到获得的内存镜像的完整性。虽然相较而言，硬件获取方式比软件更可靠，但由于软件方式使用方便、成本低，故仍然被广泛使用。本实验考虑通用性和实践操作性，选用软件方式为例进行易失性数据提取。

在实验的过程中，要注意强调操作的规范性，如录屏、哈希校验等，否则即使提取到相关重要涉案数据，也会面临证据失效的风险。

3.2　内存的获取与分析

3.2.1　预备知识：内存取证、DumpIt 工具、Volatility 工具

1. 内存取证

网络攻击内存化和网络犯罪隐遁化，使得部分关键数字证据只存在于物理内存或暂存于页面交换文件中，这使得传统的基于文件系统的计算机取证不能有效应对。内存取证作为传统文件系统取证的重要补充，是计算机取证科学的重要组成部分，通过全面获取内存数据、详尽分析内存数据，并在此基础上提取与网络攻击或网络犯罪相关的数字证据。近年来，内存取证已赢得相关领域的持续关注，获得了长足的发展与广泛应用，在网络应急响应和网络犯罪调查中发挥着不可替代的作用。

内存取证（有时称为内存分析）是指对计算机内存转储中易失性数据进行的一种分析。信息安全专业人员可以通过内存取证，来调查和识别那些不会在硬盘驱动器数据中留下痕迹的攻击或恶意行为。通过内存取证，安全专业人员可以了解运行时的各种系统活动，例如开放的网络连接或最近执行的命令和进程等。程序在计算机上运行之前，首先需要被加载到内存中，这使得内存取证变得非常重要——这意味着所有被创建、检查或删除的程序或数据都将被保存到 RAM 中。其中包括图像、所有 Web 浏览活动、加密密钥、网络连接或注入的代码片段。在许多情况下，某些证据只能在 RAM 中找到，例如在崩溃期间存在的开放网络连接。由于攻击者可以开发只驻留在内存中而不在硬盘落地的恶意软件，从而使标准的计算机取证方法几乎看不到该恶意软件，这使得内存取证变得愈发重要。

内存取证研究的首要问题是如何完整地获取内存数据。目前，获取物理内存数据的方法很多，一般利用操作系统的相关机制和特性，通过不同方法获取物理内存数据。这些方法可概括为两大类：基于硬件的内存获取和基于软件的内存获取。在获取了内存数据之后，就需要对其进行深度分析，解析、重建出内存数据中所蕴含的网络攻击和网络犯罪证据信息。传统的内存数据分析主要采用字符串搜索方法，通过搜索内存中用户名、口令、IP 地址等文本字符串，获取部分取证辅助信息。尽管该方法操作简单、使用方便，能够提取部分内存信息，但却不能有效分析与网络攻击和网络犯罪相关的进程、注册表、解密密钥、网络连

接、可执行文件、系统状态等信息。为了全面地进行内存数据分析，需依据操作系统内核数据结构和相关机制去解析与重建内存数据所蕴含的信息，进而提取相关网络攻击和网络犯罪证据。目前的内存分析内容大致可以分为 6 种：①进程信息分析；②注册表信息分析；③密钥恢复分析；④网络连接分析；⑤可执行文件分析；⑥系统状态信息分析。

内存取证作为计算机取证科学的一个重要分支，在预防网络攻击、调查网络犯罪等方面有重要且不可替代的作用和应用前景，已成为信息安全研究者所关注的热点研究领域。

2. DumpIt 工具

不同的操作系统需要用到不同的物理内存获取工具，此外在获取物理内存数据时还需尽量减少对原有内存数据的覆盖，最大程度提取出内存数据。MoonSols DumpIt 是一款同时支持 Windows32dd 和 Windows64dd 的内存副本获取工具。用户只需双击 DumpIt.exe 即可执行程序，在提示问题后面输入 y，等待几分钟时间即可在当前目录下生成主机物理内存的副本，该副本文件是以 *.raw 为后缀的镜像文件。raw 是未经处理的意思，使用该工具对物理内存进行复制是逐位进行深度复制，即按原样进行复制，这样可以避免丢失一些重要数据。

3. Volatility 工具

在最初研究内存取证的阶段使用的工具的主要功能是打开二进制（十六进制）文件，查看具体地址及内容，比如 WinHex 工具。WinHex 可以打开内存 dump 文件并查看相应的地址和内容，一般采用字符串搜索的方式。除此之外微软公司还提供了用于 Windows 操作系统的 debug 工具（Microsoft Debugging Tools for Windows）。Windows debug 工具是一个包含了一系列功能的工具集，其中某些小工具能十分简便地帮助进行取证工作，比如打开 Windows 崩溃的 dump 文件。

2008 年，内存取证领域有了一定的发展并出现了一个集合了其他各种内存取证工具的取证框架工具 Volatility。Volatility 是一款基于 GNU 协议的开源框架，使用 Python 语言编写而成，可以分析内存中的各种数据。Volatility 各项功能都是由插件实现的，各地的取证研究者可以根据自己的需要开发 Volatility 的插件来拓展其功能。Volatility 支持对 32 位或 64 位 Windows、Linux、macOS、安卓操作系统的 RAM（随机存储器）数据进行提取与分析。

Volatility 是以命令提示符方式使用的，所以同 DOS 下面的命令一样，Volatility 的开发人员也同样为我们提供了使用该工具的帮助命令。即通过-h 或-help 选项可以显示该工具的帮助列表信息。该命令显示了可用的命令选项（Options）以及支持当前操作系统版本的插件命令（Supported Plugins Command）。

例如-f FILENAME 选项的功能是说明打开一个镜像文件所使用的文件名，这个命令选项几乎在所有命令中都会用到。

Volatility 的命令格式如下：

```
volatility -f <文件名> --profile=<配置文件> <插件> [插件参数]
```

Volatility 常用插件如下：

① imageinfo：显示目标镜像的摘要信息；

② pslist：列举出系统进程，但它不能检测到隐藏或者解链的进程，psscan 可以；

③ psscan：可以找到先前已终止（不活动）的进程以及被 rootkit 隐藏或解链的进程；

④ pstree：以树的形式查看进程列表；

⑤ mendump：提取出指定进程；

⑥ filescan：扫描所有的文件列表；

⑦ hashdump：查看当前操作系统中的 password hash，例如 Windows 的 SAM 文件内容；

⑧ svcscan：扫描 Windows 的服务；

⑨ connscan：查看网络连接。

3.2.2　实验目的与条件

1. 实验目的

通过本实验，读者可以掌握以下内容：

(1) 了解内存的基本概念和内存中常见的有价值的数据；

(2) 掌握物理内存的获取方法；

(3) 掌握使用 Volatility 工具进行内存分析的方法。

2. 实验条件

本实验所需要的软硬件清单如表 3-2 所示。

表 3-2　内存取证实验清单

序　　号	设　　备	数　　量	参　　数
1	取证工作站	1 台	Windows XP 以上
2	DumpIt 软件	1 个	无
3	Volatility 软件	1 个	绿色版

3.2.3　实验过程

步骤 1：双击 DumpIt.exe 可执行程序，在提示问题后面输入 y(如图 3-11 所示)，等待几分钟时间即可在当前目录下生成主机物理内存的副本，该文件是以 *.raw 为后缀的镜像文件。

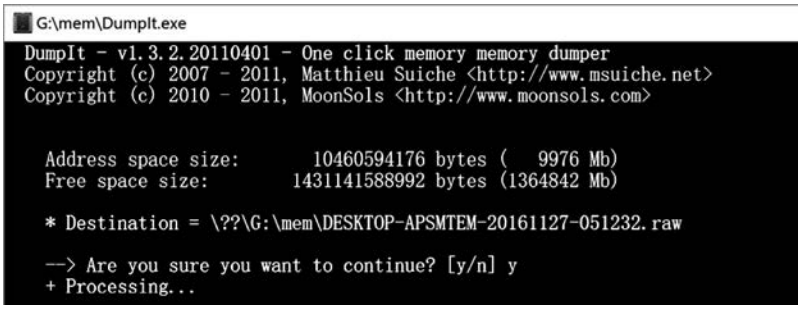


图 3-11　内存获取

注意：此步骤一般在 3.1 节现场勘验易失性数据提取环节完成。

步骤 2：将步骤 1 中获取的内存镜像文件与 Volatility 工具放于同一级目录下。打开 cmd 命令行工具，进入该级目录下。

步骤 3：使用 imageinfo 命令查看正在分析的内存样本的摘要信息，命令格式如下：

```
volatility -f victor_PC_memdump.dmp imageinfo
```

该命令可以显示主机所使用的操作系统版本、服务包以及硬件结构(32 位或 64 位)、页目录表的起始地址和获取该内存镜像的时间等基本信息。该命令的输出结果如图 3-12 所示。

```
D:\work\电子取证\内存取证>volatility -f victor_PC_memdump.dmp imageinfo
Volatility Foundation Volatility Framework 2.6
INFO    : volatility.debug    : Determining profile based on KDBG search...
          Suggested Profile(s) : Win7SP1x64, Win7SP0x64, Win2008R2SP0x64, Win2008R2SP1x64_2400
0, Win2008R2SP1x64_23418, Win2008R2SP1x64, Win7SP1x64_24000, Win7SP1x64_23418
                     AS Layer1 : WindowsAMD64PagedMemory (Kernel AS)
                     AS Layer2 : FileAddressSpace (D:\work\电子取证\内存取证\victor_PC_memdump
.dmp)
                      PAE type : No PAE
                           DTB : 0x187000L
                          KDBG : 0xf80003ffc0a0L
          Number of Processors : 2
     Image Type (Service Pack) : 1
                KPCR for CPU 0 : 0xfffff80003ffdd00L
                KPCR for CPU 1 : 0xfffff880009ef000L
             KUSER_SHARED_DATA : 0xfffff78000000000L
           Image date and time : 2018-11-02 10:31:12 UTC+0000
     Image local date and time : 2018-11-02 18:31:12 +0800
```

图 3-12　imageinfo 命令

Win7SP1x64 表明操作系统版本为 Windows 7、服务包为 SP1、硬件结构是 x64(即 64 位)。其他信息和取证关系不大。

步骤 4：使用 pslist 命令和 psscan 命令查看进程信息，查看是否有明显可疑的进程在运行。

注意：pslist 命令不能检测到内存中的隐藏进程以及由于系统受攻击导致未在链表中出现的进程信息，但 psscan 命令能够解决这个问题。

所以，本实验使用 psscan 命令查看内存进程信息，其运行结果如图 3-13 所示。由于该命令输出结果较多，我们通过> psscan. txt 将其输出结果重定向到 psscan. txt 文件中。

```
D:\work\电子取证\内存取证>volatility -f victor_PC_memdump.dmp --profile=Win7SP1x64 psscan >pss
can.txt
Volatility Foundation Volatility Framework 2.6
```

psscan.txt	2019/11/26 15:49	文本文档	11 KB
victor_PC_memdump.dmp	2018/11/2 18:43	DMP 文件	2,097,152 KB
volatility.exe	2018/6/6 9:39	应用程序	17,771 KB
volatility_2.6_win64_standalone.zip	2019/11/26 14:20	WinRAR ZIP 压缩文件	15,201 KB
内存取证工具Volatility_Framework.doc	2015/9/9 20:04	DOC 文档	978 KB

图 3-13　psscan 命令

步骤 5：使用 dlllist 命令显示一个进程装载的动态链接库的信息，使用-p PID 选项过滤输出结果，比如想要查看 PID 为 5204 的 firefox. exe 进程在运行过程中加载了哪些动态链接库，就可以通过在 dlllist 后面加上选项-p 5204，即可显示其详细信息，显示列表主要包括加载的动态链接库文件的基地址、文件大小以及文件所在路径，如图 3-14 所示。

步骤 6：使用 netscan 命令来列出所有进程连接的网络，如图 3-15 所示。

步骤 7：使用 hivelist 命令定位与硬盘上对应的注册表文件在内存中的虚拟地址和物理地址。hivelist 命令运行结果如图 3-16 所示。

```
D:\work\电子取证\内存取证>volatility -f victor_PC_memdump.dmp --profile=Win7SP1x64 dlllist -p
5204
Volatility Foundation Volatility Framework 2.6
************************************************************************
firefox.exe pid:    5204
Command line : "C:\Program Files (x86)\Mozilla Firefox\firefox.exe" -contentproc --channel="50
24.24.1484987846\1709745999" -childID 5 -isForBrowser -prefsHandle 3904 -prefMapHandle 3336 -p
refsLen 7748 -prefMapSize 190024 -schedulerPrefs 0001,2 -parentBuildID 20181030165643 -greomni
 "C:\Program Files (x86)\Mozilla Firefox\omni.ja" -appomni "C:\Program Files (x86)\Mozilla Fir
efox\browser\omni.ja" -appdir "C:\Program Files (x86)\Mozilla Firefox\browser" - 5024 "\\.\pip
e\gecko-crash-server-pipe.5024" 7776 tab


Base                             Size          LoadCount LoadTime                       Path
-------------------- ---------- ---------- ------------------------------ ----
0x0000000000120000           0x72000             0xffff 1970-01-01 00:00:00 UTC+0000   C:\Pro
gram Files (x86)\Mozilla Firefox\firefox.exe
0x0000000077ad0000           0x1a9000            0xffff 1970-01-01 00:00:00 UTC+0000   C:\Win
dows\SYSTEM32\ntdll.dll
0x0000000742b0000            0x3f000               0x3 2018-11-02 10:27:46 UTC+0000   C:\Win
```

图 3-14　dlllist 命令

```
D:\work\电子取证\内存取证>volatility -f victor_PC_memdump.dmp --profile=Win7SP1x64 netscan
Volatility Foundation Volatility Framework 2.6
Offset(P)          Proto    Local Address                  Foreign Address      State            Pid      Owner
 Created
0x7d4001e0         UDPv4    0.0.0.0:1900                   *:*                                   1680     PKit.exe
 2018-11-02 10:29:49 UTC+0000
0x7d404240         UDPv4    0.0.0.0:50745                  *:*                                   3232     BaofengPlatfor
 2018-11-02 10:29:49 UTC+0000
0x7d406bb0         UDPv4    127.0.0.1:49340                *:*                                   1364     svchost.exe
 2018-11-02 10:24:58 UTC+0000
0x7d40f670         UDPv4    127.0.0.1:58194                *:*                                   4940     BFPush.exe
 2018-11-02 10:29:59 UTC+0000
0x7d421ec0         UDPv4    0.0.0.0:5005                   *:*                                   3880     wmpnetwk.exe
 2018-11-02 10:24:58 UTC+0000
0x7d424910         UDPv4    0.0.0.0:5004                   *:*                                   3880     wmpnetwk.exe
 2018-11-02 10:24:58 UTC+0000
0x7d4292a0         UDPv4    0.0.0.0:63714                  *:*                                   3232     BaofengPlatfor
 2018-11-02 10:29:56 UTC+0000
0x7d456a90         UDPv4    0.0.0.0:5004                   *:*                                   3880     wmpnetwk.exe
 2018-11-02 10:24:58 UTC+0000
0x7d456a90         UDPv6    :::5004                        *:*                                   3880     wmpnetwk.exe
 2018-11-02 10:24:58 UTC+0000
0x7d46f6d0         UDPv6    ::1:1900                       *:*                                   1364     svchost.exe
 2018-11-02 10:24:58 UTC+0000
0x7d475010         UDPv4    0.0.0.0:15585                  *:*                                   3204     QyPlayer.exe
```

图 3-15　netscant 命令

```
D:\work\电子取证\内存取证>volatility -f victor_PC_memdump.dmp --profile=Win7SP1x64 hivelist
Volatility Foundation Volatility Framework 2.6
Virtual            Physical           Name
------------------ ------------------ ----
0xfffff8a00643341O 0x000000001ee81410 \??\C:\System Volume Information\Syscache.hve
0xfffff8a00000d010 0x000000002c317010 [no name]
0xfffff8a000024010 0x000000002c274010 \REGISTRY\MACHINE\SYSTEM
0xfffff8a000054010 0x000000002c264010 \REGISTRY\MACHINE\HARDWARE
0xfffff8a000110010 0x0000000029f1d010 \SystemRoot\System32\Config\SECURITY
0xfffff8a000300010 0x000000001e644010 \SystemRoot\System32\Config\SAM
0xfffff8a0008df410 0x0000000021c63410 \Device\HarddiskVolume1\Boot\BCD
0xfffff8a000a55010 0x000000002a597010 \SystemRoot\System32\Config\SOFTWARE
0xfffff8a001252010 0x000000002865b010 \??\C:\Windows\ServiceProfiles\NetworkService\NTUSER.DAT

0xfffff8a0012ea010 0x0000000022725010 \??\C:\Windows\ServiceProfiles\LocalService\NTUSER.DAT
0xfffff8a0019d3010 0x000000001436d010 \??\C:\Users\victor\ntuser.dat
0xfffff8a001ad5010 0x00000000096aa010 \??\C:\Users\victor\AppData\Local\Microsoft\Windows\UsrC
lass.dat
0xfffff8a0063bb010 0x0000000029a63010 \SystemRoot\System32\Config\DEFAULT
```

图 3-16　hivelist 命令

从运行结果可以发现，SECURITY 注册表文件在内存中的虚拟地址是 0xfffff8a000110010，SYSTEM 注册表文件在内存中的虚拟地址是 0xfffff8a000024010，这两个值我们稍后马上会用到。

步骤 8：使用 lsadump 命令读取注册表中与本地授权相关的数据。使用该命令的时

候，需要把步骤 7 中得到的 SYSTEM 注册表的虚拟地址作为-y 选项的参数，SECURITY 注册表文件的虚拟地址作为-s 选项的参数，lsa 命令及其运行结果如图 3-17、图 3-18 所示。

```
D:\work\电子取证\内存取证>volatility -f victor_PC_memdump.dmp --profile=Win7SP1x64 lsadump -y
0xfffff8a000024010 -s 0xfffff8a000110010 >lsadump.txt
Volatility Foundation Volatility Framework 2.6
```

图 3-17　lsa 命令

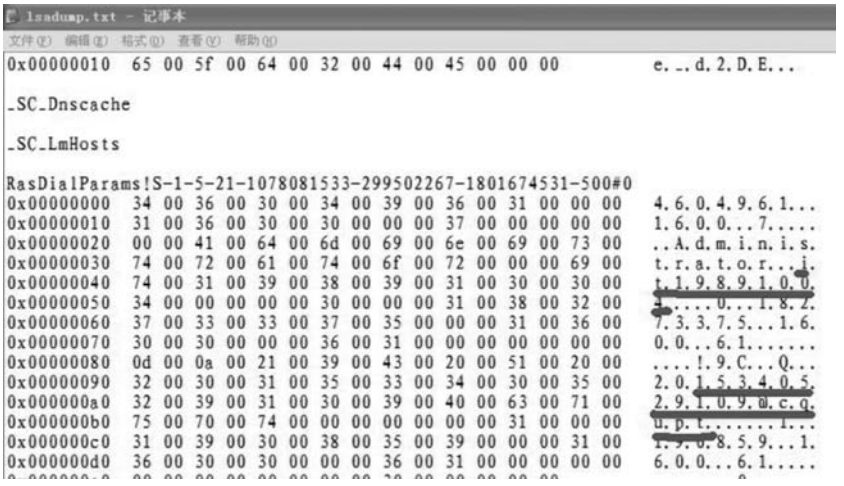


图 3-18　lsa 命令运行结果

从部分运行结果可以看见两条有用信息：第一处下画线的地方显示的是本地主机 Administrator 用户的密码：it19891004(这也确实是电脑的开机密码)；第二处下画线的地方显示的 15340529109@cqupt 正好是登录 NetKeeper 连接互联网所使用的账号名称。当然，这只是部分与 lsa 相关的账户和密码信息，更多的信息还有待进一步挖掘。

步骤 9：获取 SAM 表中所有的用户，输出结果如图 3-19 所示。

命令格式如下：volatility -f victor_PC_memdump. dmp --profile=Win7SP1x64 printkey -K "SAM\Domains\Account\Users\Names"

```
D:\work\电子取证\内存取证>volatility -f victor_PC_memdump.dmp --profile=Win7SP1x64 printkey -K "SAM\Domains\Account\User
s\Names"
Volatility Foundation Volatility Framework 2.6
Legend: (S) = Stable   (V) = Volatile

----------------------------
Registry: \SystemRoot\System32\Config\SAM
Key name: Names (S)
Last updated: 2018-10-30 04:21:24 UTC+0000

Subkeys:
  (S) Administrator
  (S) Guest
  (S) HomeGroupUser$
  (S) Lily
  (S) simon
  (S) victor
```

图 3-19　获取用户列表

步骤 10：使用 hashdump 命令获取注册表中用户密码的哈希值，具体运行结果如图 3-20 所示，下一步可使用哈希密码破解工具尝试破解密码。

命令格式如下：hashdump -y（system 的 virtual 地址）-s（sam 的 virtual 地址）

步骤 11：使用 filescan 命令获取当前所有的文件列表，如图 3-21 所示。

```
D:\work\电子取证\内存取证>volatility -f victor_PC_memdump.dmp --profile=Win7SP1x64 hashdump -y 0xfffff8a000024010 -s 0xf
ffff8a000300010
Volatility Foundation Volatility Framework 2.6
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
victor:1001:aad3b435b51404eeaad3b435b51404ee:ec042512edb5cc251bc9904d2e55fa25:::
HomeGroupUser$:1002:aad3b435b51404eeaad3b435b51404ee:0e147dd3a9f4d2a31e55f3e1fcb788c7:::
Lily:1003:aad3b435b51404eeaad3b435b51404ee:ec042512edb5cc251bc9904d2e55fa25:::
simon:1004:aad3b435b51404eeaad3b435b51404ee:ec042512edb5cc251bc9904d2e55fa25:::
```

图 3-20　用户密码哈希值

```
D:\work\电子取证\内存取证>volatility -f victor_PC_memdump.dmp --profile=Win7SP1x64 filescan
Volatility Foundation Volatility Framework 2.6
Offset(P)          #Ptr   #Hnd Access Name
------------------ ------ ------ ------ ----
0x000000007d400b30      3      0 RW-rwd \Device\HarddiskVolume3\$Directory
0x000000007d402240      2      1 R--rwd \Device\HarddiskVolume4\
0x000000007d4048d0      5      0 R--r-d \Device\HarddiskVolume3\Windows\SysWOW64\msvbvm60.dll
0x000000007d4093c0      2      0 RW-rwd \Device\HarddiskVolume3\$Directory
0x000000007d409f20     17      1 R--r-d \Device\HarddiskVolume3\Windows\System32\zh-CN\WinSATAPI.dll.mui
0x000000007d40ade0      6      0 R--rwd \Device\HarddiskVolume3\Windows\Fonts\malgun.ttf
0x000000007d40b650      1      1 R--r-d \Device\HarddiskVolume3\Windows\SysWOW64\zh-CN\kernel32.dll.mui
0x000000007d40c970     19      0 RW-rwd \Device\HarddiskVolume3\$Directory
0x000000007d40ddd0      1      1 R--r-d \Device\HarddiskVolume3\Windows\SysWOW64\zh-CN\kernel32.dll.mui
0x000000007d40f370     11      0 R--r-d \Device\HarddiskVolume3\Program Files (x86)\Baofeng\StormPlayer\diag.dll
0x000000007d4113c0      2      0 RW-rwd \Device\HarddiskVolume3\$Directory
0x000000007d411630      1      1 R--r-- \Device\HarddiskVolume3\Windows\Registration\R000000000006.clb
0x000000007d411c80     17      1 RW-rw- \Device\HarddiskVolume3\Windows\ServiceProfiles\NetworkService\AppData\Local\Mic
rosoft\Windows\Temporary Internet Files\Content.IE5\index.dat
0x000000007d413ae0     16      0 R--r-- \Device\HarddiskVolume3\Windows\inf\acpi.PNF
0x000000007d418a70     17      1 RW-rw- \Device\HarddiskVolume3\Windows\ServiceProfiles\NetworkService\AppData\Local\Mic
```

图 3-21　文件列表

3.2.4　实验小结

内存取证作为计算机取证科学的一个重要分支，在网络攻击、网络犯罪调查等方面有重要且不可替代的作用和应用前景，已成为信息安全研究者所关注的热点研究领域。本实验旨在让读者掌握基本的内存数据获取与分析方法。除了掌握常见的 Volatility 命令外，更重要的是具备一定的侦查思维，读者可多选取几个内存样本进行分析，尤其是开源的问题样本，分析的方式和思路参照但不局限于本实验。

3.3　注册表分析取证

3.3.1　预备知识：注册表

1. 注册表基础知识

注册表是 Windows 系统存储关于计算机配置信息的中央数据库，在系统中起着核心作用，存放有计算机硬件和软件的配置信息、应用软件和文档文件的关联关系以及各种网络状态信息和其他数据，可以说计算机上所有针对硬件、软件、网络的操作都是源于注册表的。同时它也是一个信息丰富的证据库，所以对电子数据取证非常重要。电子数据取证中，很多证据都是直接来源于注册表的，例如用户账号、访问记录、软件的运行历史等都是取证工作中非常重要的信息。因此，正确提取注册表中的有效数据将对取证工作大有帮助。

早期的注册表是以 ini 为扩展名的文本文件的配置文件，从 Windows 95 操作系统开始，才逐渐形成了注册表，Windows NT 是第一个从系统级别广泛使用注册表的操作系统，并在其后的操作系统中继续沿用至今。

要了解注册表，首先打开注册表编辑器，用户可以通过“Win＋R”键打开命令提示界面，输入 regedit 进入注册表编辑器，如图 3-22、图 3-23 所示。

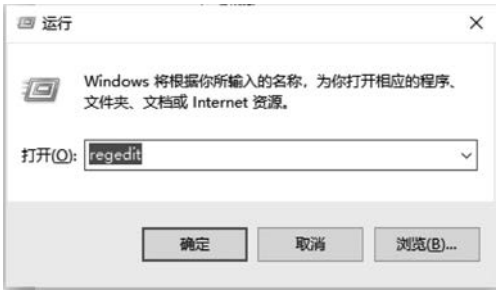


图 3-22　运行窗口打开注册表编辑器

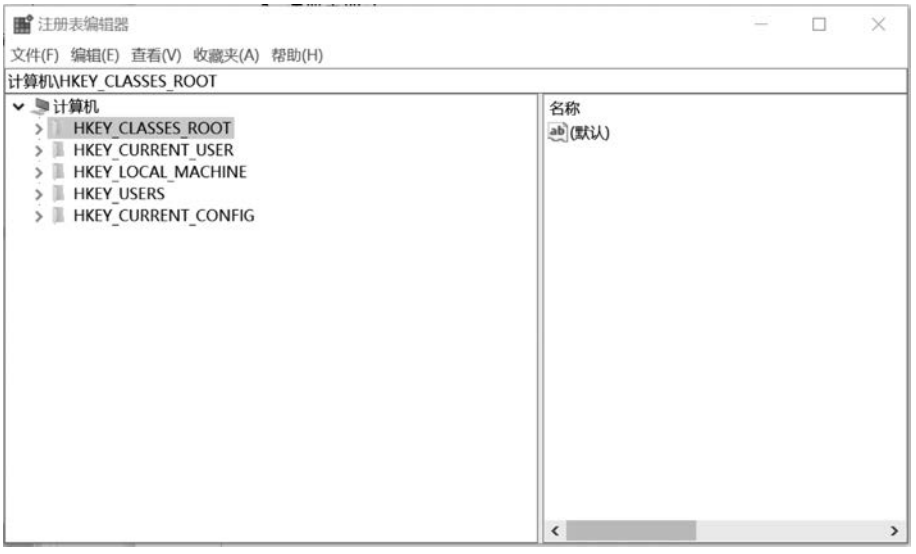


图 3-23　注册表编辑器

在注册表编辑器左侧窗格的定位区域，每一个文件夹表示注册表中的项，项又包括子项和值项，项和子项的关系就像目录可以包含子目录一样。值项也称为键值，类似硬盘上的树状目录的末端文件，项和子项可以包括一个和多个值项。值项由名称、数据类型和数值三部分组成，其格式为："名称：类型：数据"。

预定义项，是代表注册表中的主要部分的项，是指在注册表中以 HKEY 作为前缀的文件夹，位于注册表树状结构的最顶层。预定义项类似于硬盘上的根目录，Windows 2000/XP 及以上的注册表编辑器中有五大预定义项，分别为：

① HKEY_CLASSES_ROOT。

包含了启动应用程序所需的全部信息，包括文件扩展关联信息及 OLE 数据库，存储在这里的信息可确保使用 Windows 资源管理器打开文件时能打开正确的程序。

② HKEY_USERS。

包含了计算机上的所有以活动方式加载的用户信息（如用户在该系统中设置的口令、标识等）和默认配置文件，默认配置文件决定了没有人登录时，计算机如何响应。

③ HKEY_LOCAL_MACHINE。

包含了本地计算机的配置信息（用于任何用户），如软件、硬件及安全。

④ HKEY_CURRENT_USER。

包含了当前登录用户的配置信息。用户的文件夹、屏幕颜色和“控制面板”设置都存储在这里。

⑤ HKEY_CURRENT_CONFIG。

在启动过程中动态创建，包含系统启动时的硬件相关的配置信息。

配置单元是作为文件出现在系统注册表的一部分，位于 HKEY_LOCAL_MACHINE 和 HKEY_USERS 两个预定义项下，是项、子项和值的离散体，它位于注册表层的顶部。配置单元是一个文件，可以通过注册表编辑器中“加载配置单元”和“卸载配置单元”选项，从一个系统移动到另一个系统。

注册表配置单元是注册表中的一组项、子项和值，它有一组包含其数据备份的支持文件，如表 3-3 所示。配置单元（HKEY_CURRENT_USER 除外）的支持文件都位于 Windows\System32\Config 文件夹中，包括 SAM、Security、System、Software、Default，称为系统注册表文件，包含 Hardware（硬件）、User Settings（用户配置）、Software（软件）、System configuration（系统配置等信息）；HKEY_CURRENT_USER 的支持文件位于 Windows\Users\用户名文件夹中，包含 NTUSER. DAT，称为用户注册表文件，每一个用户都有一个注册内容，能够记录用户活动的相关细节，是取证应该检查的关键内容。

表 3-3　配置单元文件

注册表配置单元	相 关 文 件
HKEY_LOCAL_MACHINE\SAM	SAM、SAM. log、SAM. sav
HKEY_LOCAL_MACHINE\Security	SECURITY、Security. log、Security. sav
HKEY_LOCAL_MACHINE\Software	SOFTWARE、Software. log、Software. sav
HKEY_LOCAL_MACHINE\System	SYSTEM、System. alt、System. log、System. sav
HKEY_USERS\DEFAULT	Default、Default. log、Default. sav
HKEY_CURRENT_CONFIG	System、System. alt、System. log、System. sav、Ntuser. dat、Ntuser. dat. log

2. 注册表取证

注册表给取证人员提供了大量的系统配置信息和用户使用信息。通过分析注册表，可以提供一份详尽的嫌疑人计算机设备的简要报告，包括硬件配置、系统配置、使用者信息、用户账号、外置设备等。

常见的注册表取证分析项有：

① 用户账户及安全设置（SAM/SECURITY）。

- 用户账号/SID；
- 登录时间、登录次数；
- 最后登录时间等。

② 系统及软件信息（SYSTEM/SOFTWARE）。

- 系统信息（OS 版本、安装日期、最后关机时间等）；
- 时区信息（Time Zone）；
- 硬件信息/服务列表；
- 网络配置信息/共享文件夹信息；

- 应用程序运行痕迹记录；
- USB 设备使用记录等。

③ 用户相关信息(NTUSER.DAT)。

最近打开的文件记录(MRU,RecentDocs)。

3.3.2 实验目的与条件

1. 实验目的

通过本实验,让读者在了解注册表基础知识和常见的注册表分析项的基础上,学会使用常见的注册表分析工具,掌握常用的注册表键值的取证方法。

2. 实验条件

本实验所需要的软硬件清单如表 3-4 所示。

表 3-4 注册表分析取证实验清单

序 号	设 备	数 量	参 数
1	取证工作站	1 台	Windows XP 以上
2	U 盘	1 个	无
3	WRR.exe	1 个	无

3.3.3 实验过程

读者在做此实验内容前,需要在自己的计算机上进行如下操作,作为实验素材。

① 在 IE 地址栏分别输入任意 2 个网址,并浏览查看。

② 单击“开始”→“运行”,分别执行 regedit、msconfig、eventvwr 命令。

③ 依次单击“开始”→“搜索”→“文件或文件夹”,并执行 2 次搜索任务(搜索内容自己定,例如可以搜索 C 盘所有 bmp 图片或者包含有 *** 内容的文件)。

④ 打开并简单查看“ **.pdf”。

⑤ 将 U 盘插入到自己的主机上。

1. 打开注册表编辑器查看相应注册表项内容

步骤 1：打开注册表编辑器中如下项,查看通过标准的文件“打开/保存”对话框所操作文件的历史记录(MRU 为 most recently used 缩写),如图 3-24 所示。

HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\OpenSaveMRU

步骤 2：打开注册表编辑器中如下项,查看通过 Windows 资源管理器打开或者运行的最近的文件,如图 3-25 所示。

HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs

步骤 3：打开注册表编辑器中如下项,查看通过“开始”→“运行”方式执行的命令,如图 3-26 所示。

HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU

步骤 4：打开注册表编辑器中如下项,查看系统中安装的程序信息,如图 3-27 所示。

HKLM\Software\Microsoft\Windows\CurrentVersion\Uninstall

值得注意的是,运行于 64 位系统下的 32 位应用程序默认操作 32 位注册表项(即被重定向到 WOW6432Node 下的子项),而 64 位应用程序才是上述操作的直观子项。

图 3-24　最近使用文件列表 1

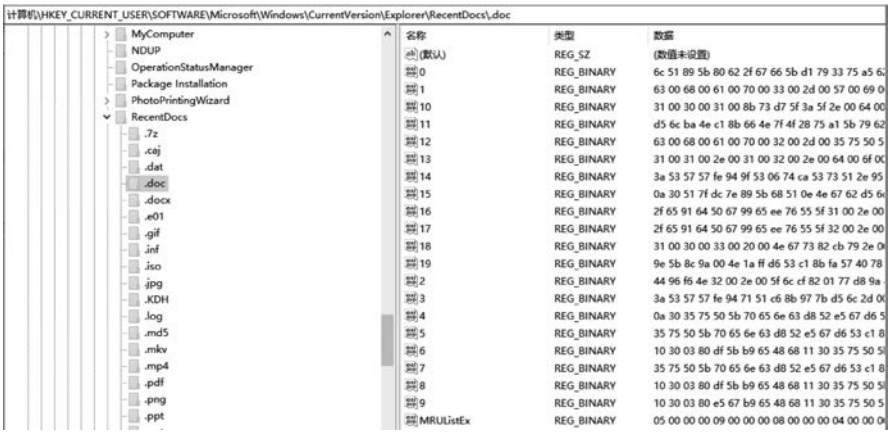


图 3-25　最近使用文件列表 2

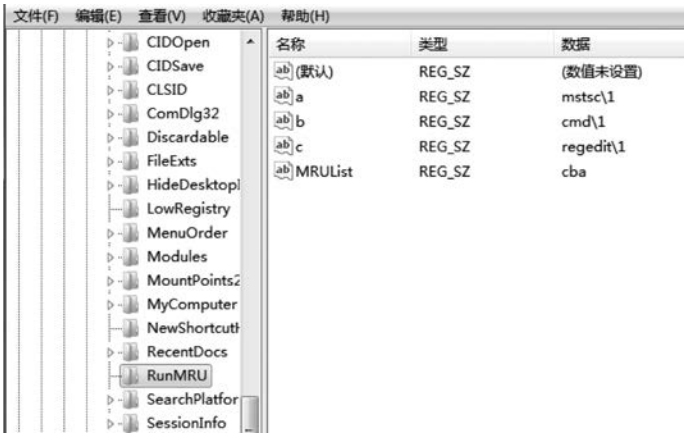


图 3-26　cmd 中运行命令的历史记录

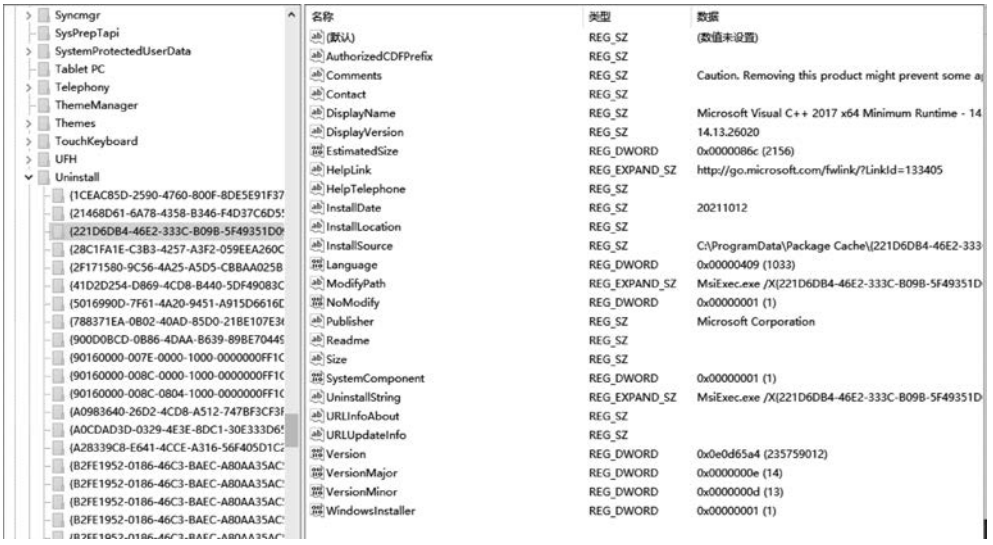


图 3-27　安装的应用程序信息

步骤 5：打开注册表编辑器中如下项，查看 32 位应用程序（如 WeChat）的相关信息，如图 3-28 所示。

HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall

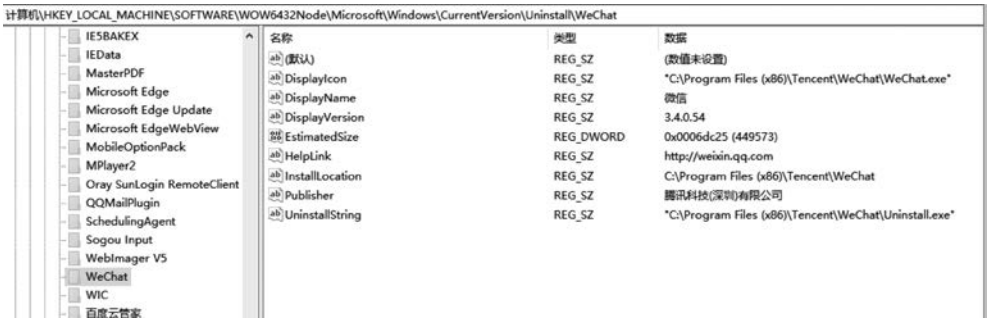


图 3-28　安装的 32 位应用程序信息

步骤 6：打开注册表编辑器中如下项，查看自动运行的程序，如图 3-29 所示。

HKLM\Software\Microsoft\Windows\CurrentVersion\Run

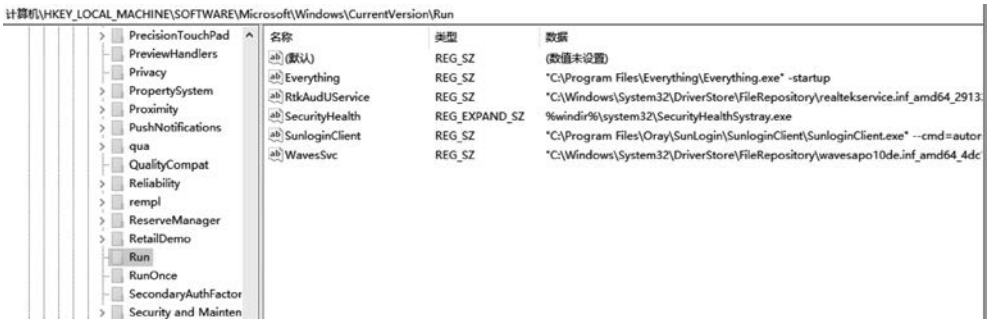


图 3-29　自动运行的程序

步骤 7：打开注册表编辑器中如下项，查看浏览器地址栏中键入的 URL 地址和文件路径，如图 3-30 所示。

HKCU\Software\Microsoft\Internet Explorer\TypedURLs

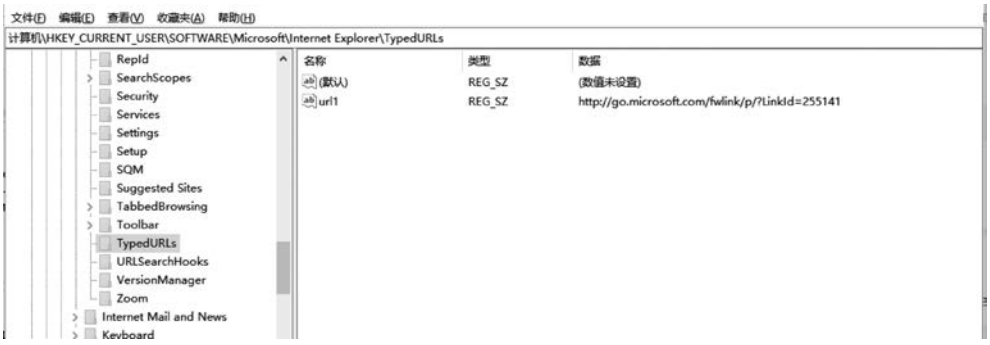


图 3-30　浏览器地址栏中键入的 URL 地址和文件路径

步骤 8：打开注册表编辑器中如下项，查看计算机上曾经使用过的所有 USB 设备，如图 3-31 所示。

HKLM\SYSTEM\ControlSet001\Enum\USBSTOR

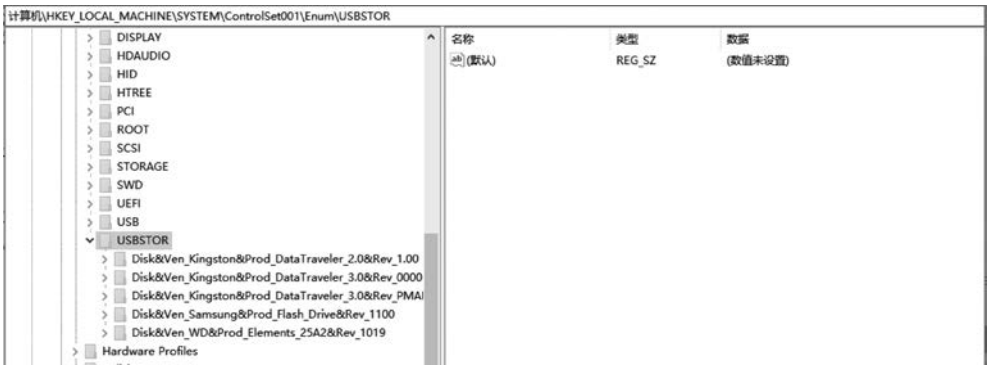


图 3-31　计算机上使用过的所有 USB 设备

步骤 9：打开注册表编辑器中如下项，查看计算机连接过的无线接入点的 GUID 列表，如图 3-32 所示。

HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\NetworkList\Profiles

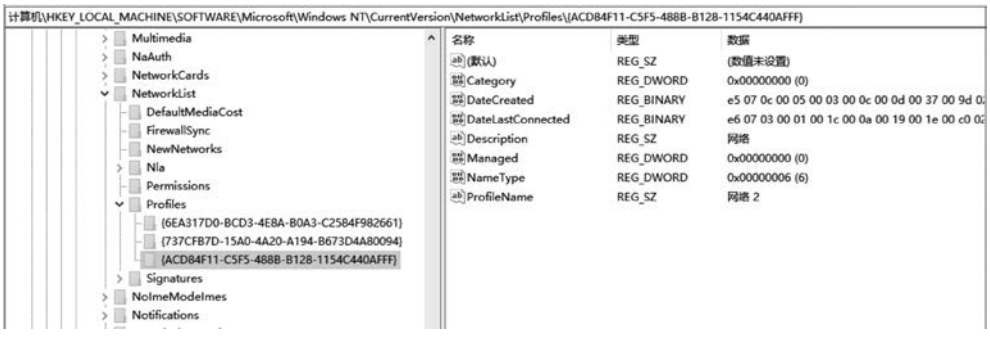


图 3-32　计算机连接过的无线接入点

步骤 10：打开注册表编辑器中如下项，查看每个接口的 IP 地址及相关信息，如图 3-33 所示。

HKLM\System\CurrentControlSet\Services\Tcpip\Parameters\Interfaces

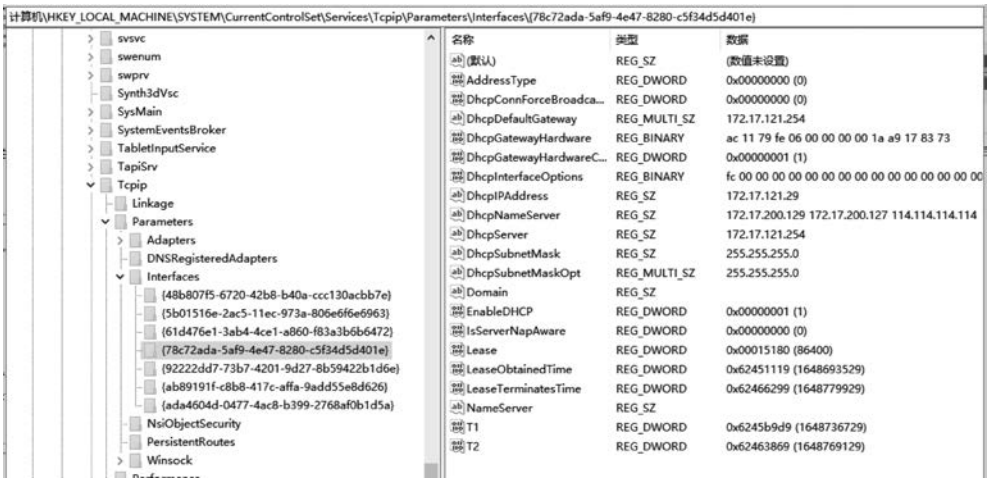


图 3-33　每个接口的 IP 地址及相关信息

步骤 11：打开注册表编辑器中如下项，查看计算机连接的默认打印机信息，如图 3-34 所示。

HKU\S-1-5-21-2019780057-1784683038-**********-1001\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows

图 3-34　默认打印机

2. 使用 WRR(Windows Registry Recovery)工具查看注册表文件

步骤 1：导出计算机中的注册表文件，包括 SAM、Software、System 等。

步骤 2：使用 WRR 工具打开注册表文件 SAM，查看计算机中所有用户信息，如图 3-35 所示。

步骤 3：使用 WRR 工具打开注册表文件 Software，查看计算机中安装的软件信息，包括操作系统信息，如图 3-36、图 3-37 所示。

步骤 4：使用 WRR 工具打开注册表文件 System，查看系统配置相关信息，如服务及驱动、网络配置、防火墙设置等，如图 3-38、图 3-39、图 3-40 所示。

其他信息操作类似，不再一一列出。

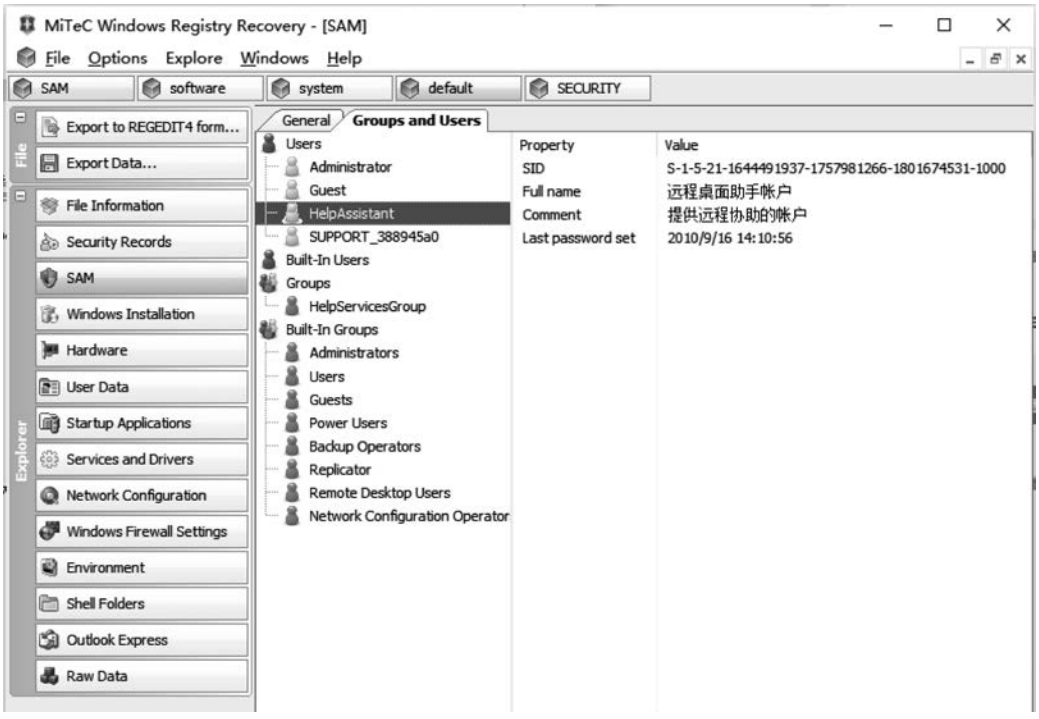


图 3-35　注册表文件 SAM 中用户信息

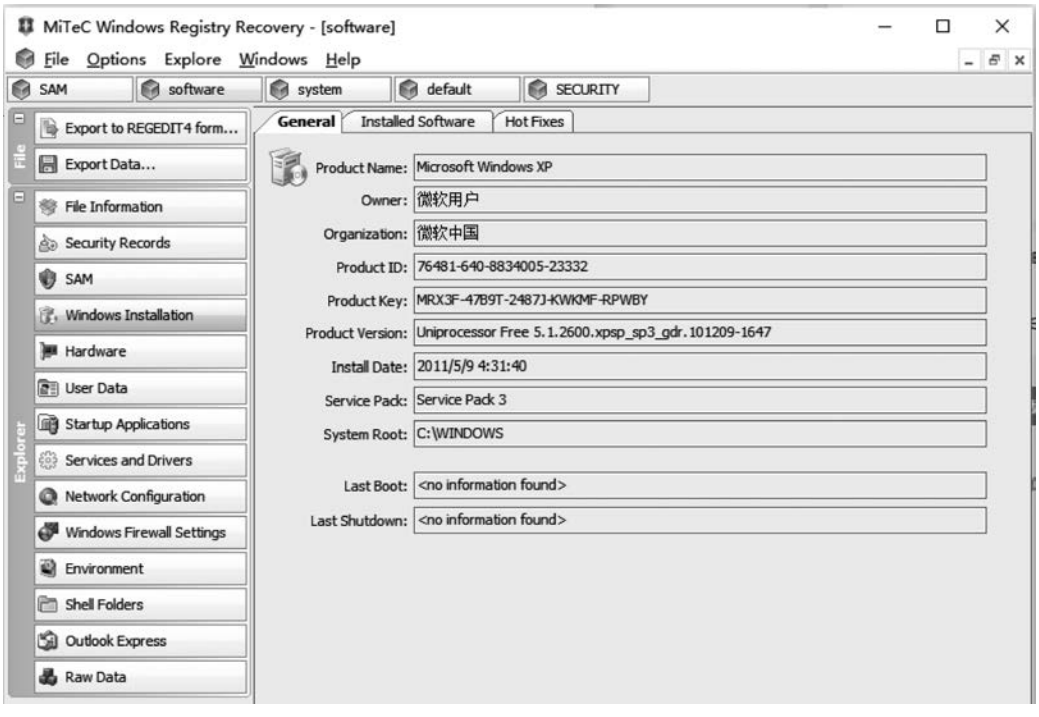


图 3-36　Software 注册表文件中操作系统信息

图 3-37 Software 注册表文件中安装软件信息

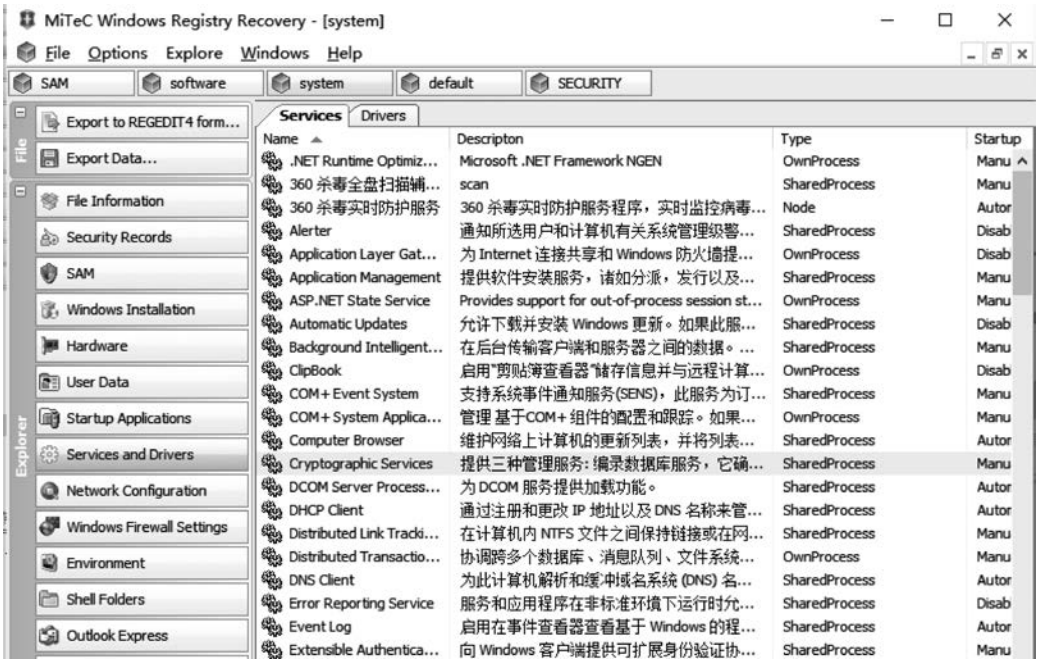


图 3-38 System 注册表文件中服务及驱动信息

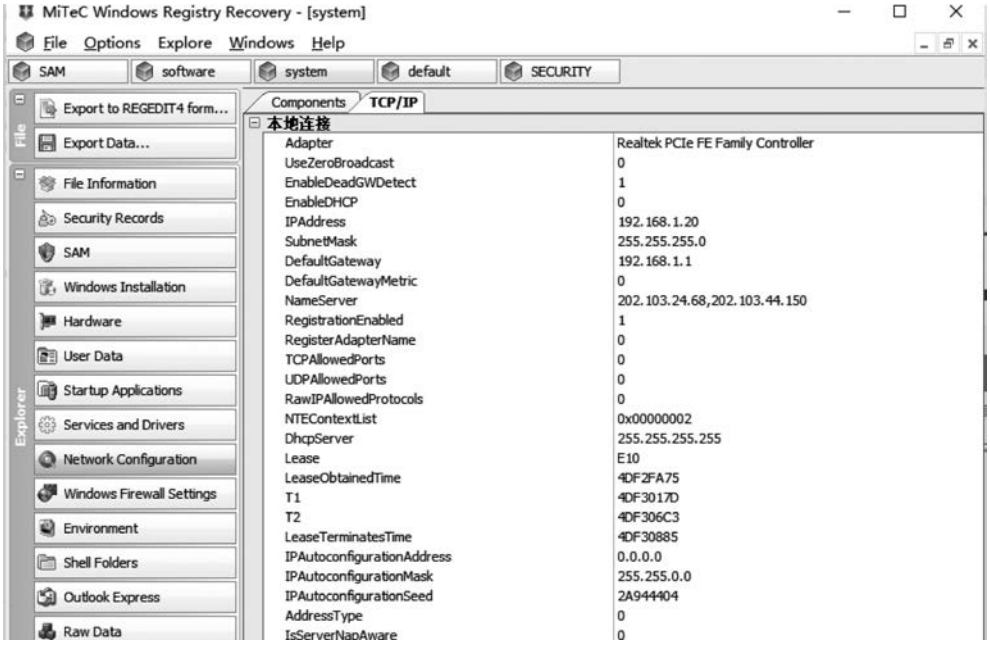


图 3-39　System 注册表文件中网络配置信息

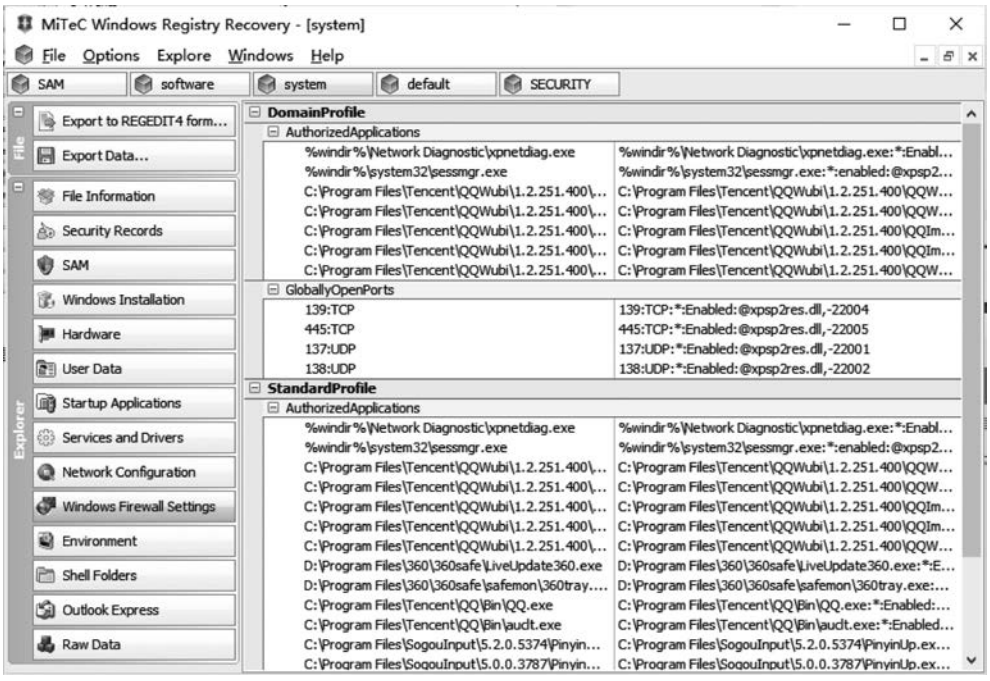


图 3-40　System 注册表文件中防火墙设置信息

3.3.4　实验小结

通过本次实验的学习，读者了解注册表不仅仅是一个用于存储 Windows 系统用户、硬件和软件的存储配置信息的数据库，更是计算机犯罪取证中的一个宝库。虽然常见的综合

性取证软件大都集成了注册表分析功能，但仍存在很多信息需要取证人员结合案情手工分析。想要成为一名合格的计算机取证人员，必须要熟练运用和掌握注册表的相关知识。

3.4 Windows 事件日志取证

3.4.1 预备知识：Windows 事件日志

1. Windows 事件日志

从 Microsoft Windows NT 3.5 操作系统起，日志服务就一直存在于微软公司开发的 Windows 操作系统中。但是从 Microsoft Windows NT 6.0 操作系统（Windows Vista 与 Windows Server 2008）开始，微软公司采用了一种全新的日志（EVTX 日志）服务，EVTX 由 Windows 事件查看器创建，包含 Windows 记录的事件列表，以专有的二进制 XML 格式保存。

Windows 事件日志文件保存在 C:\Windows\System32\winevt\Logs 路径中，如图 3-41 所示。

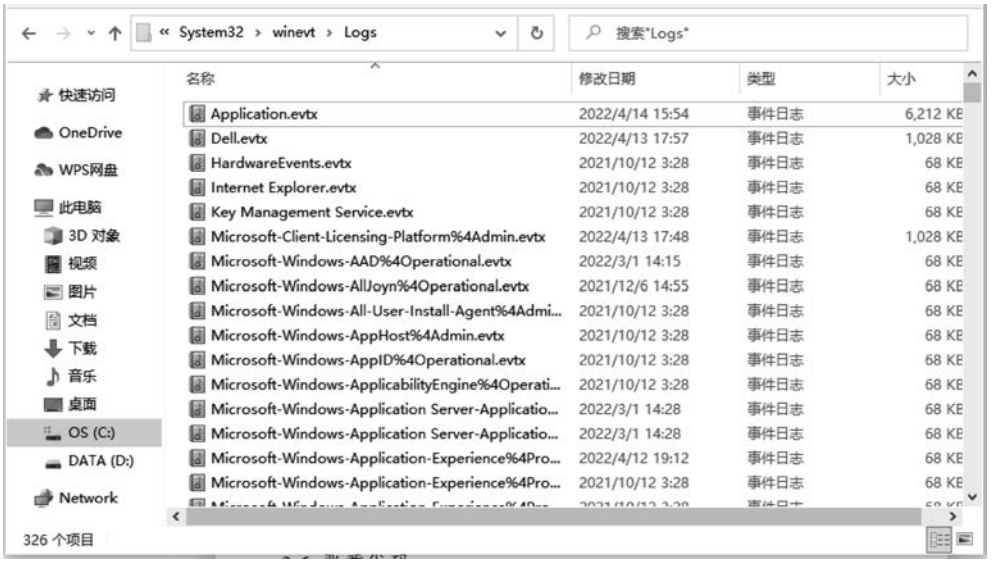


图 3-41 Windows 事件日志文件

核心的日志文件主要有三个，分别是：System.evtx、Application.evtx 和 Security.evtx，它们分别是系统日志、应用程序日志和安全日志。

（1）系统日志：系统日志记录系统进程和设备的驱动程序的活动。它输出的记录包括设备驱动程序是否启动失败，硬件是否自检出错，以及系统服务的开启、关闭、暂停。

（2）应用程序日志：记录普通的用户程序和一些商用程序在运行过程中出现的事件，它会输出自己记录的所有报错和需要用户知晓的信息。

（3）安全日志：记录系统的安全审计日志事件，比如登录事件、对象访问、进程追踪、特权调用、账号管理、策略变更等。安全日志也是取证中最常用到的，是处理入侵事件的重要武器，分析者需要查看和筛选这些文件中的信息从而发现蛛丝马迹。

三个文件的默认大小均为 20480KB(20MB),当记录事件数据超过 20MB 时,系统将优先覆盖过期的日志记录,从头开始写入新的记录,也就是相当于一个循环记录的缓存文件。

2. Windows 事件日志的查看方式

查看 Windows 事件日志的标准方法是使用 Windows 系统自带的“事件查看器”(可以通过运行“eventvwr”来启动),如图 3-42 所示。

图 3-42　事件查看器

在事件查看器中,系统日志被分为 Windows 日志,应用程序和服务日志两大类,其中 Windows 日志包括了应用程序、安全、Setup、系统和 Forwarded Events(转发事件)。

事件查看器可以将日志文件导出为 evt、evtx、xml、txt 和 csv 等格式,并导入其他系统的事件查看器进行浏览。因为日志文件格式在各个 Windows NT 版本中通用,所以调查人员也可以利用本地计算机的事件查看器远程连接其他计算机,以管理员权限查看浏览日志文件。调查人员可以利用事件查看器的“筛选”功能,显示特定时间类型和时间段的相关内容。

通过对 Windows 系统日志的取证分析,取证人员可以对操作系统、应用程序、服务、设备等操作行为记录通过关键的时间点进行回溯。

3. 常见的 Windows 事件日志分析

(1) 系统日志

系统日志可以捕获由系统自身产生的事件。任何自动执行的操作,或直接利用 OS 功能的用户驱动操作都会记入日志,包括软硬件安装、打印作业和网络层事件等。取证人员关注的系统事件常与案件的性质和被调查者的抗辩有关,常见的有:

① 事件日志启动和停止。事件 ID6005 和 ID6006 代表日志服务的启动和停止,主动关闭日志服务的行为往往值得深入追查。

② 系统关机和重启。事件 ID6008 表示系统的一次意外关闭,ID6009 则和系统重启相

关。当发现 ID6006 后不久紧跟 ID6009 事件，通常可以认为是系统原因。事件 ID1074 显示引起系统关闭的进程，ID1076 显示系统关闭的原因。

③ 登录失败。事件 ID100 表示一个已知账户的验证失败，调查中发现的这类事件，有可能是特定用户通过猜测密码或使用穷举等破解工具的线索。

④ 机器信息改变。事件 ID6011 表示系统名称改变，如果发现名称与现存信息不匹配，就要重点查找这个事件 ID。

⑤ 打印。ID10 显示的是打印作业和来源，以打印请求者用户名的方式显示。

(2) 应用程序日志

应用程序日志由应用程序使用产生，Windows 允许第三方软件通过 API 记录应用程序事件，防病毒软件和安装程序通常会使用这样的功能，在调查中经常使用到的有：

① 确认软件安装。使用微软安装程序的情况下，通过事件 ID11707(成功)、事件 ID11708(失败)和事件 ID11724(卸载)来记录软件包的运行，查看这些 ID 可以发现特定软件的安装、试图安装和卸载的时间。

② 确认和排除病毒感染。大多数防病毒软件在检测到病毒时，会产生一个 ID5 事件。案件调查中，涉案人员有时会辩称系统问题是病毒引起的，通过查看这个事件，可以显示和排除他声称的时间内是否有病毒发作。

③ 启动和关闭防火墙。记录了用户主动打开或关闭系统防火墙的行为。

④ 检查黑客攻击企图。ID 为 1000～1004 的事件记录有错误的应用程序，可以提供应用程序漏洞被利用的线索，事件 ID4097 也有可能代表类似活动。

应用程序日志事件常依赖于特定系统中安装的具体应用程序，以及是否独立使用事件日志服务，或者利用本地私有日志对系统日志进行补充，所以调查人员在检查应用程序日志之外，通常还必须检查应用程序是否使用了本地私有日志记录。

(3) 安全日志

安全日志是所有日志的基础，登录、注销、尝试连接和改变系统策略等关键事件，都会在安全日志中反映出来。企业为了支持安全事件调查和溯源，通常会在本地或组策略下的审核策略中要求计算机系统激活如审核账户登录事件、账户管理、登录、策略改变、特权使用等。其中，登录和注销对于证实什么人在什么时间执行了什么操作尤为重要，而其他安全事件则根据案件不同，会对某些特定的调查有帮助。

① 成功登录和注销事件。交互式的登录事件通过事件 ID4624 来描述，是登录类型的一个子类，调查人员比较关注的登录类型有 ID2(本地)、ID3(网络)、ID7(Ctrl＋Alt＋Del 或屏幕解锁)、ID10(远程桌面)、ID11(缓存的用户凭证登录)。

另外，注销事件显示了某用户连接的时间段，以 ID4647 为用户启动注销的开始，ID4634 为结束。

② 登录失败事件。登录失败是判定是否有人进行密码猜测或暴力攻击的有力证据之一，登录失败事件通过 ID4625 来描述。可以输入事件 ID：4625 进行日志筛选，若用户登录失败次数明显偏多，那么这台服务器管理员账号可能遭遇了暴力猜解。

③ 对象访问。在一个特定对象属性的“安全”选项卡上单击“高级”按钮，可对待定的 NTFS 文件和文件夹进行审核。激活对象审核可以记录从试图读取对象到成功删除对象的任何操作。如果系统开启这个级别的审核，就能显示某个特定实体在何时被访问、被谁访

问、特定文件和目录的改变和删除，或者突出显示对关键对象的非法访问企图。

④ 用户账户相关事件。事件 ID4720、ID4722、ID4723、ID4724、ID4725、ID4726、ID4738、ID4740，表示当用户账号发生创建、删除、改变密码时的事件记录。

⑤ 日志清除。事件 ID1102 表示安全事件日志被清除。在没有合理原因将旧文件存储到一个新文件之前，安全日志几乎是不会被清除的，一旦有该事件发生，很有可能表明使用者或入侵者在故意掩盖痕迹。

综上所述，不同的事件 ID 代表了不同的意义，这些可以在网上很容易查到，需要根据案件的类型和基本情况，有重点地查看。

3.4.2　实验目的与条件

1. 实验目的

通过本实验，读者可以掌握以下内容：

(1) 了解 Windows 事件日志的概念及查看方式；

(2) 掌握 Windows 事件日志的种类及内容；

(3) 掌握常见的 Windows 事件日志的分析。

2. 实验条件

本实验所使用的 Windows 操作系统为 Windows 10(Windows Vista 以上)。

3.4.3　实验过程

步骤 1：学生两两分组，互相进行远程桌面访问，首先打开将要被远程访问的计算机，在计算机系统属性中，选择“远程”选项卡，在远程桌面选项中勾选“允许运行任意版本远程桌面的计算机连接(较不安全)”，单击“确定”按钮，如图 3-43 所示。

图 3-43　开启远程桌面功能

步骤 2：在另一台计算机上打开远程桌面窗口，输入被控制计算机的 IP 地址，单击“连接”按钮，如图 3-44 所示，尝试包括登录成功、断开连接等操作。

图 3-44 远程桌面访问

步骤 3：在被远程访问的计算机上，右击“我的电脑”，在弹出的快捷菜单中选择“管理”选项，打开“事件查看器”；或者按下 Win＋R 的组合键，在运行窗口中输入“eventvwr. msc”，如图 3-45 所示，直接打开“事件查看器”。

图 3-45 打开事件查看器

步骤 4：攻击者使用 RDP 远程登录受害者计算机，会在事件日志中生成相应事件。在“Windows 日志”下，选择“安全”(Security. evtx)，筛选事件 ID 为 4624，如图 3-46 所示，即为账户成功登录事件，如图 3-47 所示。其中，“LogonType＝10”的为远程桌面访问，可以看到访问主机的源 IP 地址。

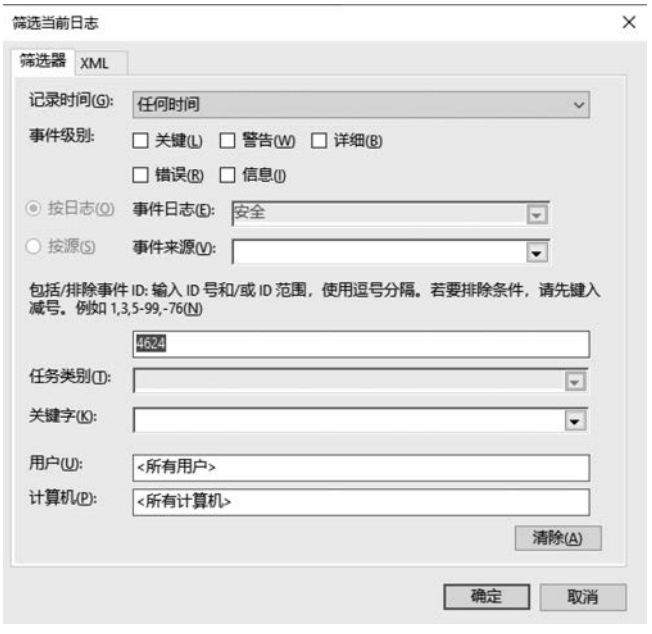


图 3-46 筛选日志

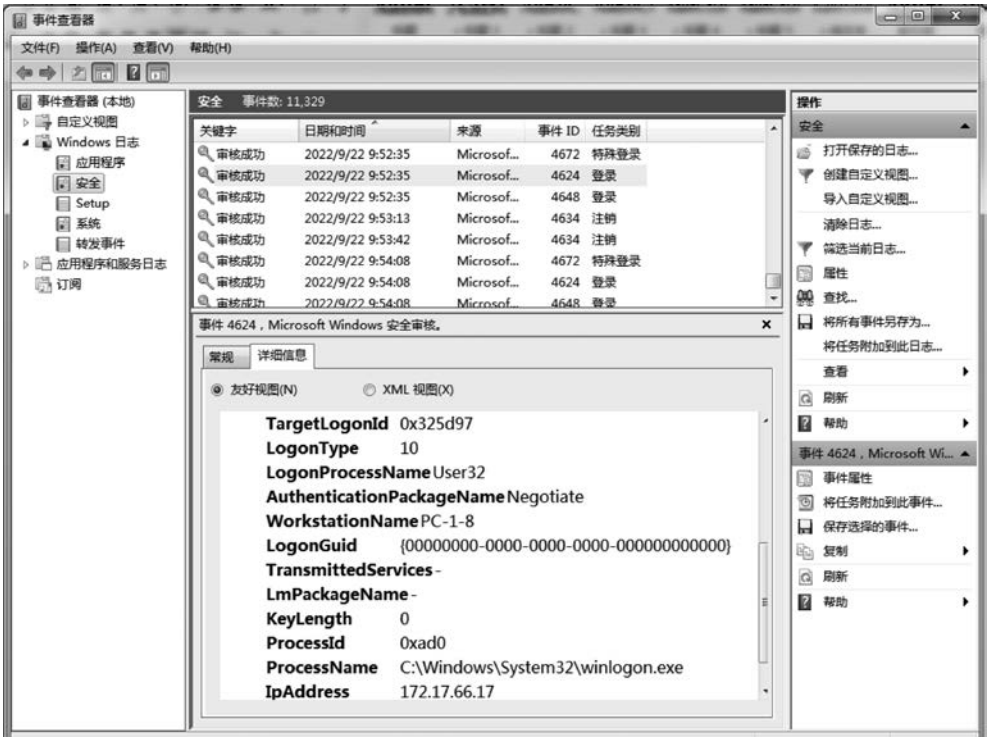


图 3-47　远程桌面登录成功

步骤 5：查看同路径下事件 ID 为 4648 的事件，为用户使用明文凭证尝试登录的事件，其中，“详细信息”记录了用户的 IP 地址，如图 3-48 所示为本机登录，图 3-49 所示为远程主机登录。

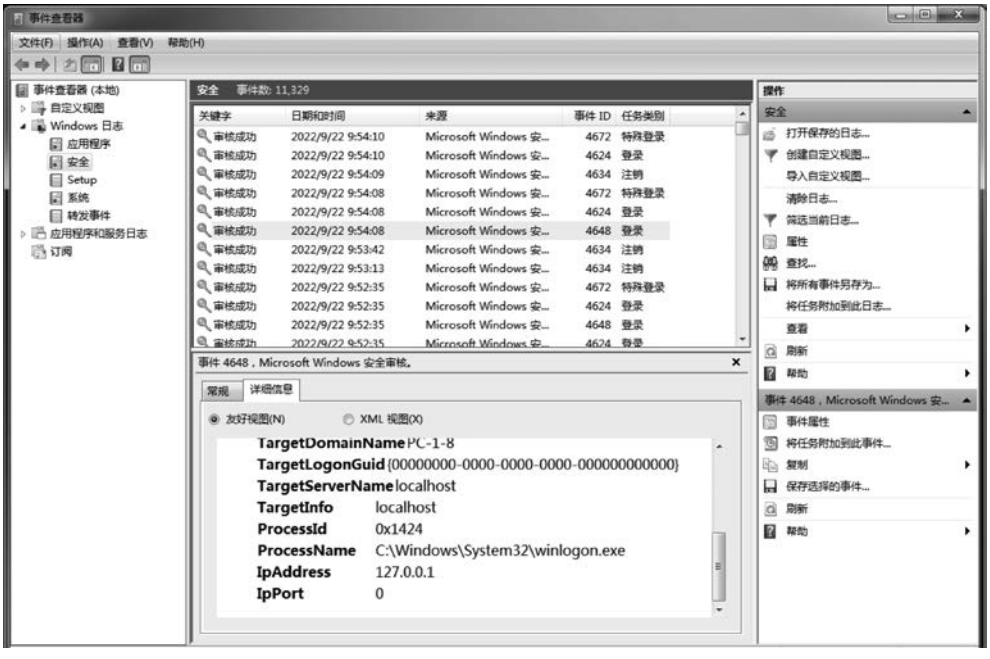


图 3-48　本机明文登录事件

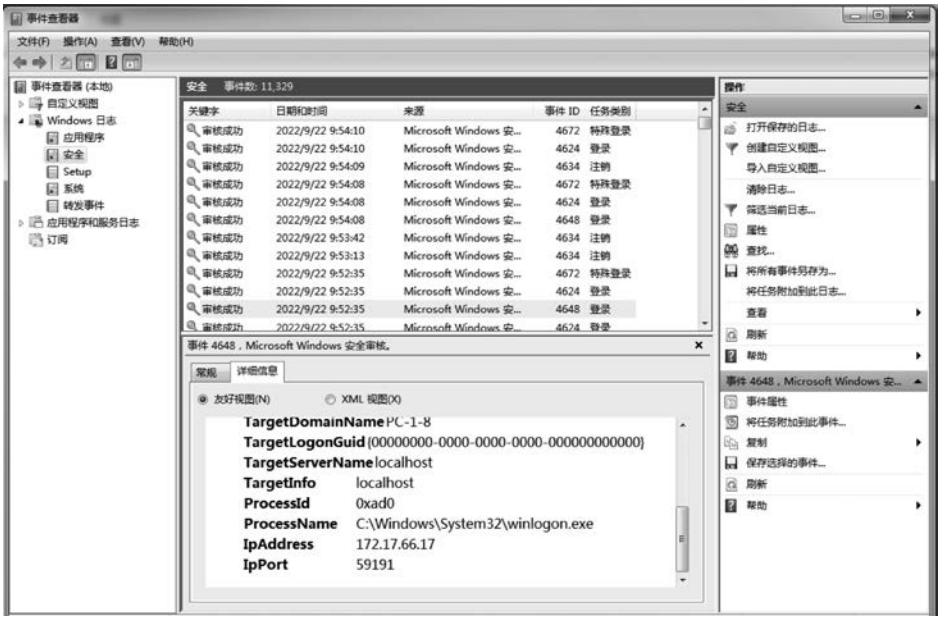


图 3-49 远程主机明文登录事件

步骤 6：同样地，ID4778 事件为重新连接到一台 Windows 主机的会话，ID4779 事件为断开到一台 Windows 主机的会话。

步骤 7：查看远程连接日志，具体路径为：应用程序和服务日志-Microsoft-Windows-TerminalServices-RemoteConnectionManager-Operational。其中，ID1149 事件为用户认证成功，如图 3-50 所示，可以清晰地看到用户名及源网络地址。

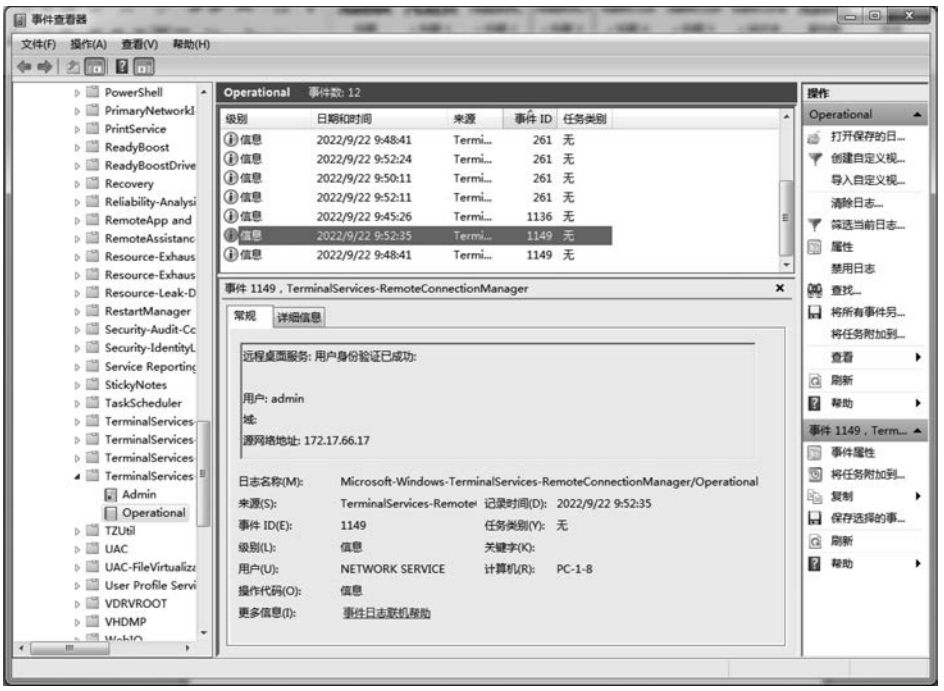


图 3-50 用户认证成功

3.4.4 实验小结

其他常见的 Windows 事件包括：用户登录或注销、远程访问审计、即插即用设备使用、系统时间修改、无线网络接入等，读者可自行查阅资料，练习、查看。实际应用中，要通过日志准确分析出恶意行为是需要大量实践经验的，同样也需要了解 Windows 日志中各种事件 ID 组合、状态码所对应的事件。

3.5 回收站取证

3.5.1 预备知识：回收站运行机制

对删除文件的恢复，一直都是电子数据取证的重要部分。在 Windows 操作系统中，用户选择删除一个文件后，这个文件并没有真正删除，而是进入了一个叫作回收站的地方，如果删错了或者后悔了都可以在回收站中进行恢复操作。所以，回收站是一个重要的信息来源，通过分析回收站可以知道被删除文件的信息，包括原始路径、删除时间、文件大小等。

回收站是 Windows 操作系统中的一个隐藏的系统文件夹，其文件名及存放路径根据 Windows 系统版本的不同而不同，具体如表 3-5 所示。

表 3-5 回收站文件夹存储位置

操作系统版本	分 区 格 式	回收站位置
Windows 95/98/ME	FAT32	\Recycled\INFO2
Windows NT/2K/XP	FAT32	\Recycled\INFO2
	NTFS	\Recycled\<USER SID>\INFO2
Windows 7/10	NTFS	$Recycle.Bin\<USER SID>\

对于 Windows XP 来说，在 FAT32 文件系统下，删除的文件在 Recycled 文件夹中的命名格式为：D[文件原始隶属盘符][索引号][原始扩展名]。同时在文件夹中会存在一个名为 INFO2 的二进制文件，用来记录所有删除文件的时间及路径信息。

而 Windows 7 及以后的系统，回收站的机制发生了改变，抛弃了 INFO2 文件保存删除文件信息的做法，而是为每个被删除文件建立一个删除记录。通过分析每个删除记录，可以了解文件的原始信息。当一个文件被删除时，它被进行两个操作：①将删除的文件重命名为“$R”开头的文件，后面跟着随机字符串，后缀与原来文件一致。②创建一个“$I”开头的文件，后面与“$R”开头的文件相同。“$I”开头的文件为对应的“$R”开头的文件的回收站记录文件。这样每个删除文件都有自己的回收站记录文件。

“$I”开头的回收站记录文件，大小都是 544 字节，主要包括以下删除记录信息（具体结构如表 3-6 所示）：

- 被删除文件原始路径；
- 被删除文件大小；
- 被删除文件的删除信息（64 位 Windows 时间）。

表 3-6 回收站记录文件结构

数据结构	长度	偏移量
操作系统版本	8	0x00
被删除的文件大小	8	0x08～0xF
文件删除时间	8	0x10～0x17
被删除的文件名(全路径)	0～520	0x18～0x21F

目前主流的取证工具,例如 EnCase、FTK 等都支持对回收站文件的解析。但是由于操作系统的版本不同,导致回收站的结构和运行机制都有所不同。因此还需要在理解回收站的结构和机制的前提下,利用相应的工具进行取证。

3.5.2 实验目的与条件

1. 实验目的

通过本实验,读者重点掌握以下内容:

(1) 了解回收站的运行机制;

(2) 掌握回收站记录文件的解析过程;

(3) 掌握在注册表中查找文件删除者信息的过程。

2. 实验条件

本实验所需要的软硬件清单如表 3-7 所示。

表 3-7 回收站取证实验清单

序号	设备	数量	参数
1	取证工作站	1 台	Windows XP 以上
2	EnCase 软件	1 套	EnCase7
3	检材 U 盘 (包含证据文件“Malone's HDD 1A. Ex01”)	1 个	无

3.5.3 实验过程

步骤 1:打开 EnCase 软件,新建案例并添加证据文件“Malone's HDD 1A. Ex01”。

步骤 2:找到分区 C 中的 $Recycle. Bin 文件夹,如图 3-51 所示。

图 3-51 回收站文件夹

步骤 3：查看回收站文件夹中的文件，找到可疑文件（如图 3-52 所示：nuclear-explosion.jpg），查看其短名（如图 3-53 所示：$RAFR6IT.jpg），即为文件删除后，重命名为“$R”开头的文件。

图 3-52　回收站中的可疑文件

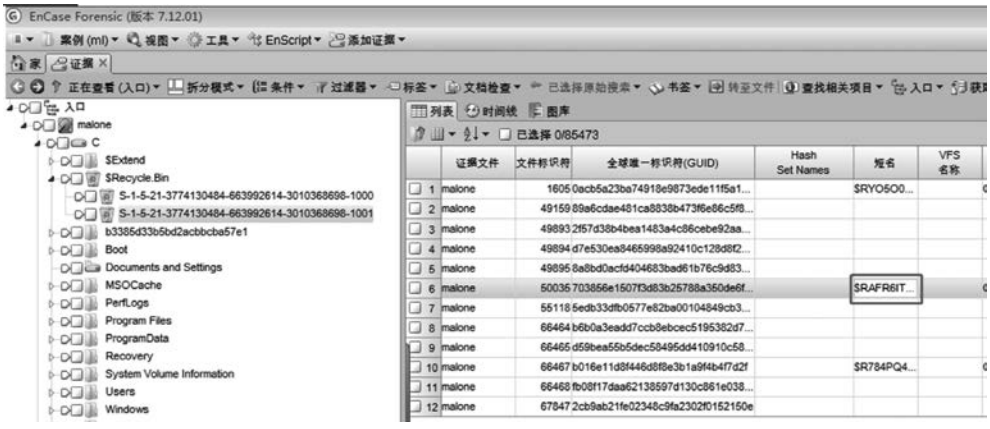


图 3-53　可疑文件的短名

步骤 4：根据该文件的短名找到其对应的以“$I”开头的回收站记录文件（如图 3-54 所示：$IAFR6IT.jpg）。

图 3-54　回收站记录文件

步骤 5：通过查看回收站记录文件中的十六进制数据，解析回收站记录文件（操作系统版本、文件大小、删除时间、原始路径等）。

其中，0x00～0x07 按小端顺序解析为 1，表示是 windows 7 操作系统，如图 3-55 所示。

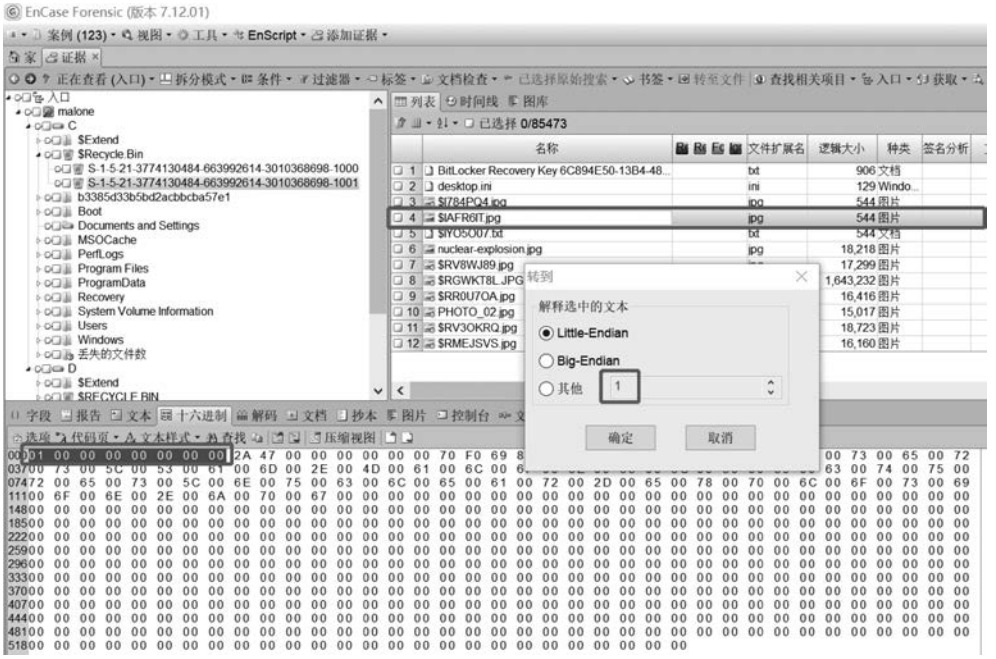


图 3-55　回收站记录文件中操作系统版本解析

0x08～0x0F 按小端顺序解析为 18218，表示被删除的文件逻辑大小是 18218 字节（与 EnCase 解析的 nuclear-explosion.jpg 文件逻辑大小一致），如图 3-56 所示。

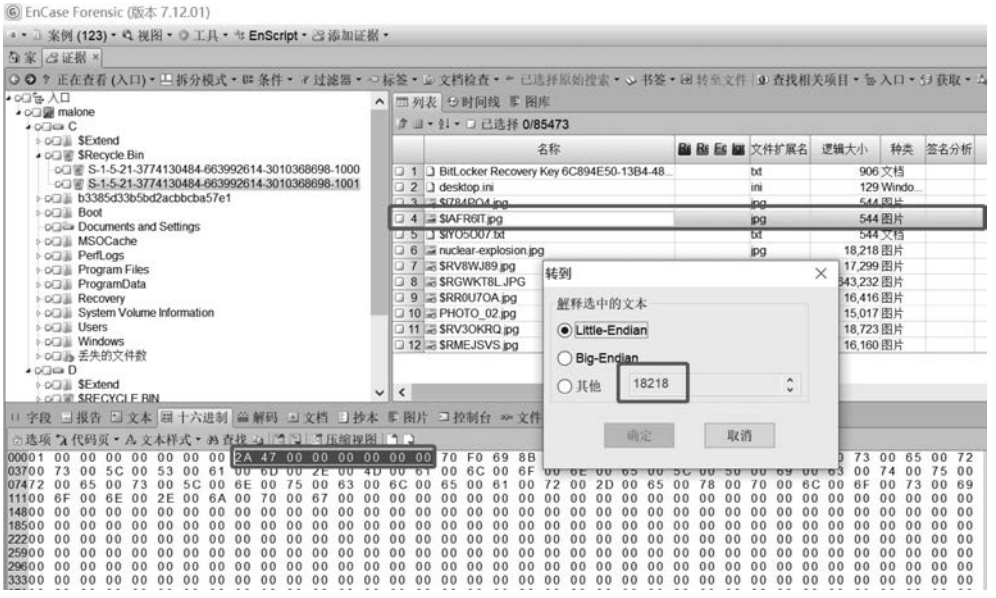


图 3-56　回收站记录文件中文件大小解析

0x10～0x17 字节（如图 3-57 所示）选中后，使用 EnCase 解码功能，按照“Windows 日期/时间”格式解析即可得到文件的删除时间，如图 3-58 所示。

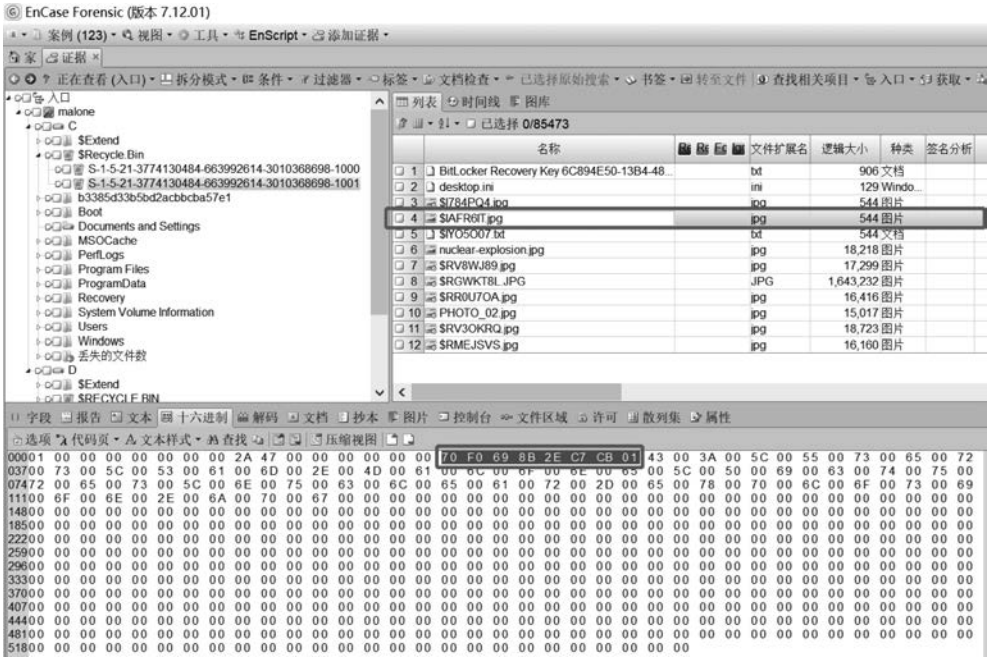


图 3-57　回收站记录文件中文件删除时间解析

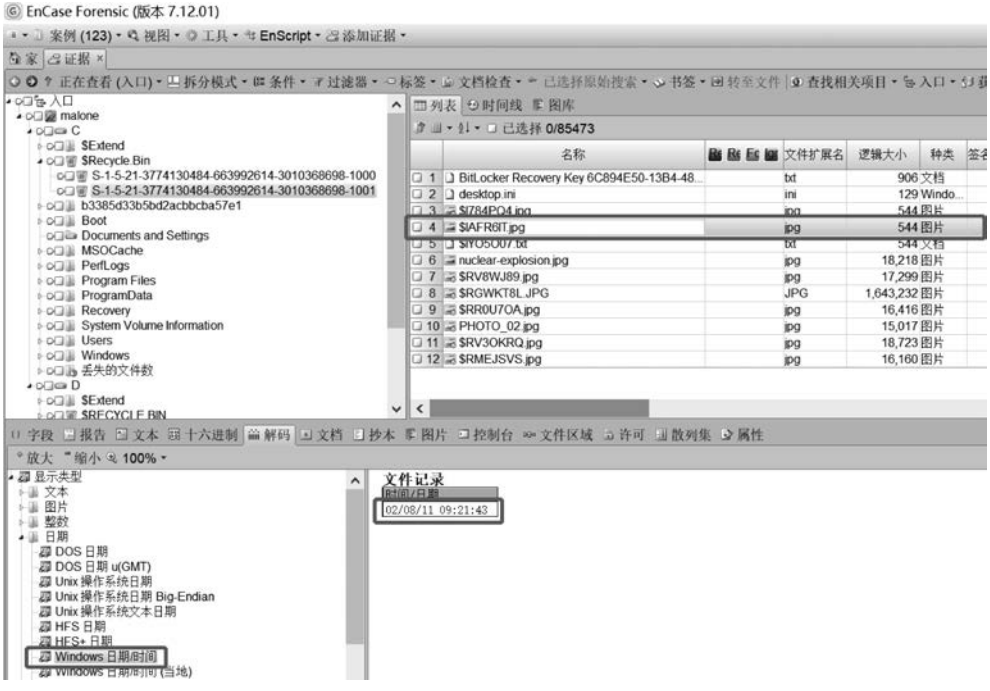


图 3-58　EnCase 解码日期时间

0x18～0x21F 字节解析为“C:\Users\Sam. Malone\Pictures\nuclear-explosion. jpg”，即为被删除文件的原始路径(全路径)，如图 3-59 所示。

步骤 6：单击工具栏中的“许可”按钮，可以看到该回收站记录文件的权限信息，其所有者为 Sam. Malone，如图 3-60 所示，说明文件的删除者即为 Sam. Malone。

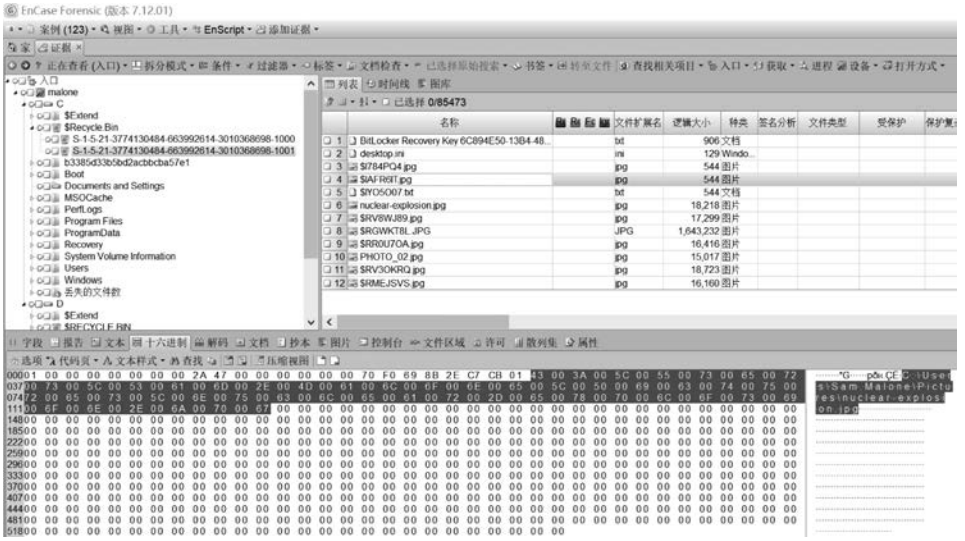


图 3-59　回收站记录文件中文件原始路径解析

图 3-60　文件删除者信息解析

步骤 7：联系所学注册表取证相关知识，在注册表中查看文件删除者的相关信息。在 C:\Windows\System32\Config 下找到注册表文件“SAM”，如图 3-61 所示。

步骤 8：解析该复合文件“SAM”，找到 SAM\Domains\Account\Users 的文件夹，可以看到 000003E9 文件夹(图 3-62)，而十六进制 3E9 的十进制数为 1001，与回收站目录文件夹 SID 中的 UID(如图 3-63 所示：1001)一致，说明删除文件的用户就是 3E9，即该 000003E9 文件夹为 Sam. Malone 的用户文件夹。

步骤 9：解析 000003E9 文件夹中的文件，其中 F 文件记录了用户创建时间等信息，V 文件记录了用户名和 SID 等信息，不再具体一一解析，如图 3-64 所示，即为用户名。

由此，不仅得到被删除文件的相关信息，更进一步获知删除者的相关信息。

图 3-61　注册表文件“SAM”

图 3-62　SAM 注册表文件解析

图 3-63　回收站目录文件夹 SID 中的 UID

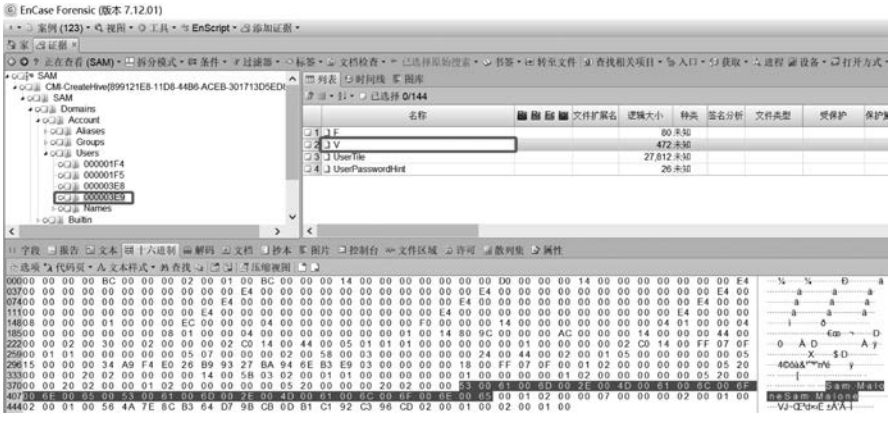


图 3-64　注册表文件中用户信息的解析

3.5.4 实验小结

回收站文件夹中，$R 文件记录原始文件内容，$I 文件记录恢复被删除文件的必要信息（文件大小、被删除时间、原始路径及文件名称）。当恢复被删除文件而原始目录不存在时，利用 $I 文件的信息可以重建目录。同时，联系回收站文件夹与注册表文件，可获知删除者的重要信息。在计算机调查取证过程中，需时刻记得回收站的重要作用，在回收站中出现的只字片语，很有可能对案件的侦破起决定作用。

3.6 分区恢复

3.6.1 预备知识：磁盘分区原理

在使用计算机进行数据存储与读取的过程中，分区丢失是一种比较常见的故障表现形式。由于意外断电、删除、格式化，或罪犯分子为妨碍案件调查而恶意破坏等原因，分区可能会被删除或破坏。因此，无论是在取证调查中为了发现证据，还是为了不影响用户正常使用，恢复被删除的分区都有着重要的意义。

1. 硬盘分区

硬盘分区是在一块物理硬盘上创建多个独立的逻辑单元，这些逻辑单元就是 C 盘、D 盘、E 盘等，又称为逻辑卷。

在实际分区时，通常把硬盘分为主分区和扩展分区，然后根据硬盘大小和使用需要将扩展分区继续划分为几个逻辑分区。建立硬盘分区的步骤是：建立主分区→建立扩展分区→将扩展分区分成多个逻辑分区。硬盘划分多个分区后，可以用于存放不同类型的文件，如存放操作系统、应用程序、数据文件等。

随着科技的发展，硬盘的容量越来越大，市场上 1TB 或 2TB 的大容量硬盘已经很常见。大容量硬盘给用户提供更多存储空间的同时，也使得在创建硬盘分区之前，好好地规划硬盘分区的方案成为必要。合理划分分区可以方便用户更好地管理自己的硬盘。

2. MBR 的数据结构

主引导记录（Master Boot Record，MBR）是采用 MBR 分区表的硬盘的第一个扇区，即 C/H/S 地址的 0 柱面 0 磁头 1 扇区，也叫作 MBR 扇区，共 512 字节。当计算机启动并完成自检后，首先会读取磁盘的 MBR 扇区。MBR 主要由三部分组成：引导程序、分区信息表、结束标志。其中引导程序占用 446 字节，主要用于硬盘启动时将系统控制权转移给用户指定的并在分区表中登记了的某个操作系统区；分区信息表占用 64 字节，主要负责描述磁盘内的各分区情况；结束标志为 2 字节“55 AA”。MBR 数据结构如图 3-65 所示。

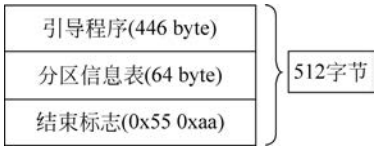


图 3-65 MBR 数据结构

MBR 主要功能如下：

① 首先检查硬盘中分区表是否完好；

② 从分区表查找可引导的“活动”分区；

③ 将活动分区中第一逻辑扇区数据加载到内存中。

在 DOS 分区中，该扇区内容被称为 DOS 引导记录，简称 DBR。

3. 分区表项数据结构

分区表用于记录分区信息，从 MBR 的第 0x1BE 字节开始，共 64 字节，其中共有 4 个分区表项，每个表项 16 字节，各字节含义如图 3-66 所示。

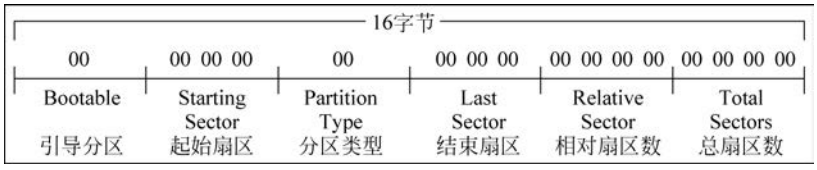


图 3-66　分区表项数据结构

引导标志只有两种可能值，0x80 为可引导(即表示该分区为操作系统分区)，0x00 为不可引导；分区类型标志表明它所描述的分区类型，如 NTFS 的类型值为 0x07、FAT 的类型值为 0x0B；起始扇区及结束扇区均以 CHS 方式表示，CHS 区域的值主要用于较老的操作系统，C、H、S 分别代表磁盘的柱面号、扇区号、磁头号；相对扇区数(分区起始逻辑扇区)及占用总扇区数均以 LBA 方式表示。分区起始 LBA 地址是非常重要的参数，如果该区域数据受到破坏，操作系统将无法找到文件系统分区或扩展分区的起始位置。

如果在没有人为因素改变分区的情况下，因意外而导致的磁盘显示分区丢失、分区显示未格式化等，应该首先检查主分区表是否损坏，通过查看现有分区表描述的各个分区的前后关系是否合理、跳转到分区起始扇区查看是否为正常的 DBR 来判断故障原因。

3.6.2　实验目的与条件

1. 实验目的

通过本实验，读者重点掌握以下内容：

(1) 掌握 MBR 引导扇区中分区表的解析过程；

(2) 掌握使用 EnCase 工具进行分区恢复的方法。

2. 实验条件

本实验所需要的软硬件清单如表 3-8 所示。

表 3-8　分区恢复实验清单

序　号	设　备	数　量	参　数
1	取证工作站	1 台	Windows XP 以上
2	EnCase 软件	1 套	EnCase7
3	检材 U 盘 (包含证据文件“Malone's HDD 1A. Ex01”)	1 个	无

3.6.3　实验过程

步骤 1：打开 EnCase 软件，新建案例并添加证据文件“Malone's HDD 1A. Ex01”。

步骤 2：单击工具栏中的“设备”按钮，选择“磁盘视图”选项，如图 3-67 所示，即可进入磁盘视图。

步骤 3：单击磁盘视图下“查看簇”按钮前的复选框，可在扇区和簇视图之间切换，如图 3-68 所示。

图 3-67　EnCase 磁盘视图

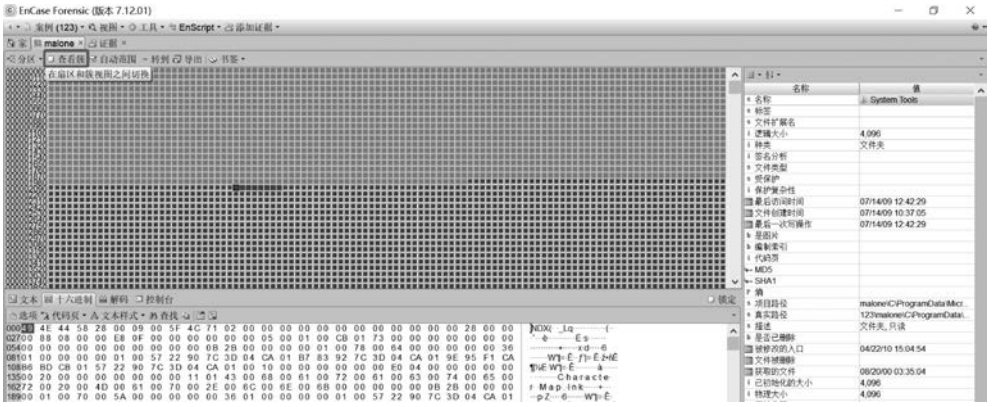


图 3-68　切换扇区和簇视图

步骤 4：在扇区视图下(即显示的是硬盘上各扇区的状态和数据)，单击第一个小方块(LBR0 号扇区)，即为主引导记录 MBR 扇区，单击中间工具栏中的“十六进制”按钮，查看该扇区中的十六进制数据，如图 3-69 所示。

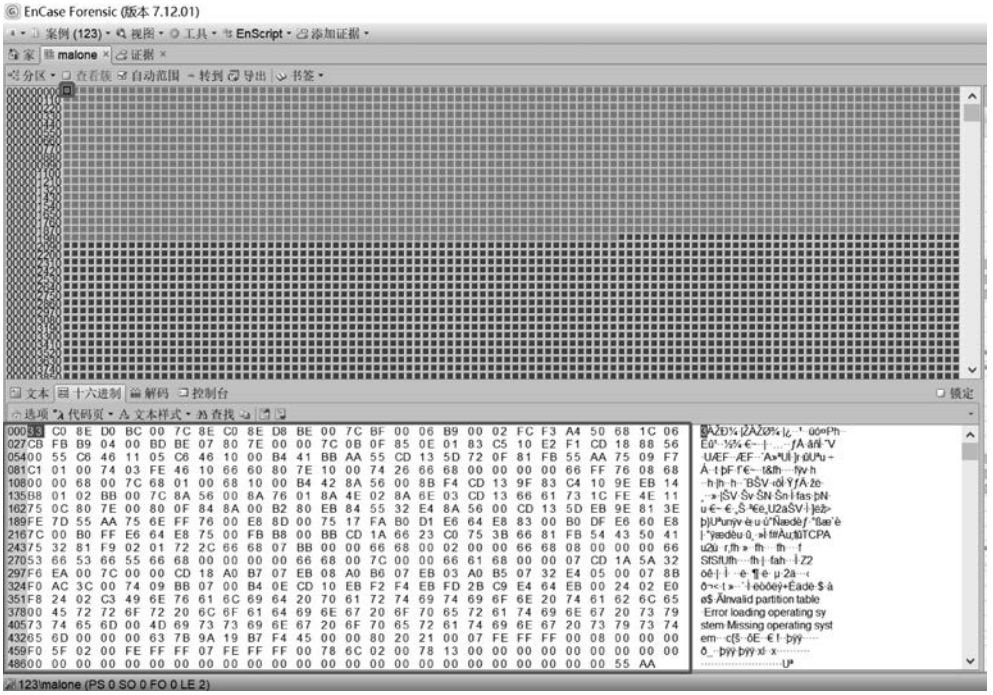


图 3-69　MBR 扇区

步骤 5：联系所学的 MBR 扇区数据结构可知，结束标志“55 AA”前的 64 字节为分区信息表，如图 3-70 所示。共分为 4 个分区表项，每个分区表项为 16 字节，经分析发现仅有 2 个分区表项有数据，即分区信息表中仅存在 2 个分区信息。

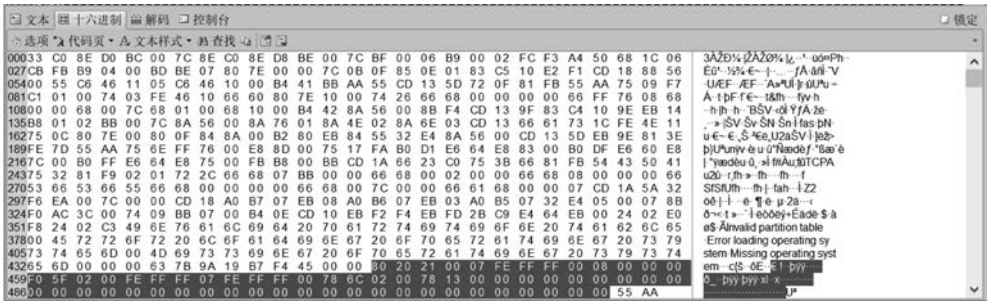


图 3-70　分区信息表

步骤 6：选中该 64 字节分区信息表，单击中间工具栏中的“解码”按钮，选择“Windows-分区入口”选项，可得到解析完成后的分区信息，如图 3-71 所示。

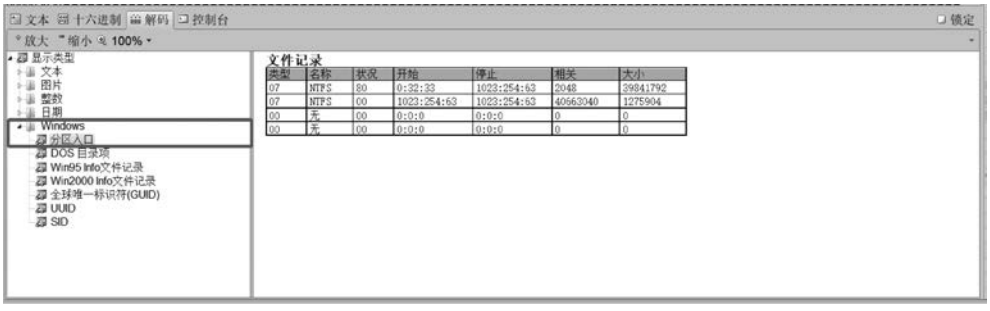


图 3-71　分区信息表解析

可以看出，该磁盘共有 2 个分区，其中第一个分区为活动分区（系统盘）。第一个分区起始位置逻辑扇区为 2048，占用总扇区数为 39841792。第二个分区起始位置逻辑扇区为 40663040，占用总扇区数为 1275904。

步骤 7：在扇区视图下，鼠标停留在第一个扇区并右击，在弹出的快捷菜单中选择“转到”选项，跳转到 2048 号扇区（图 3-72），即为分区 1 的起始扇区 DBR，如图 3-73 所示。

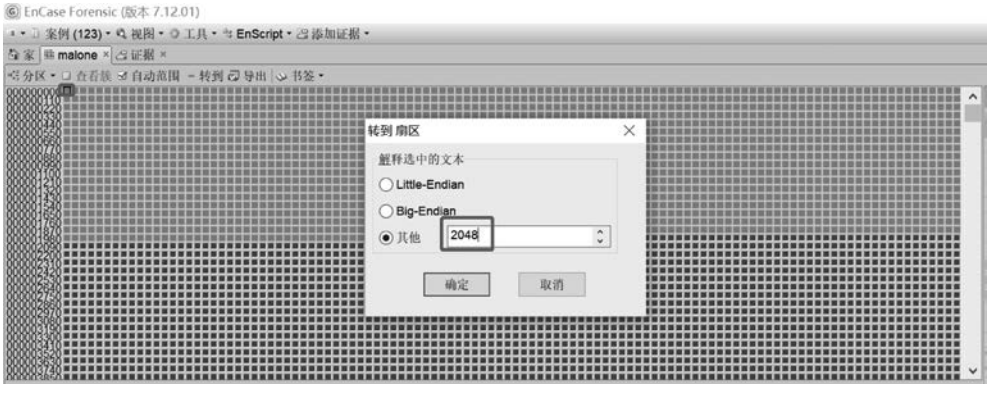


图 3-72　跳转到分区 1 起始扇区

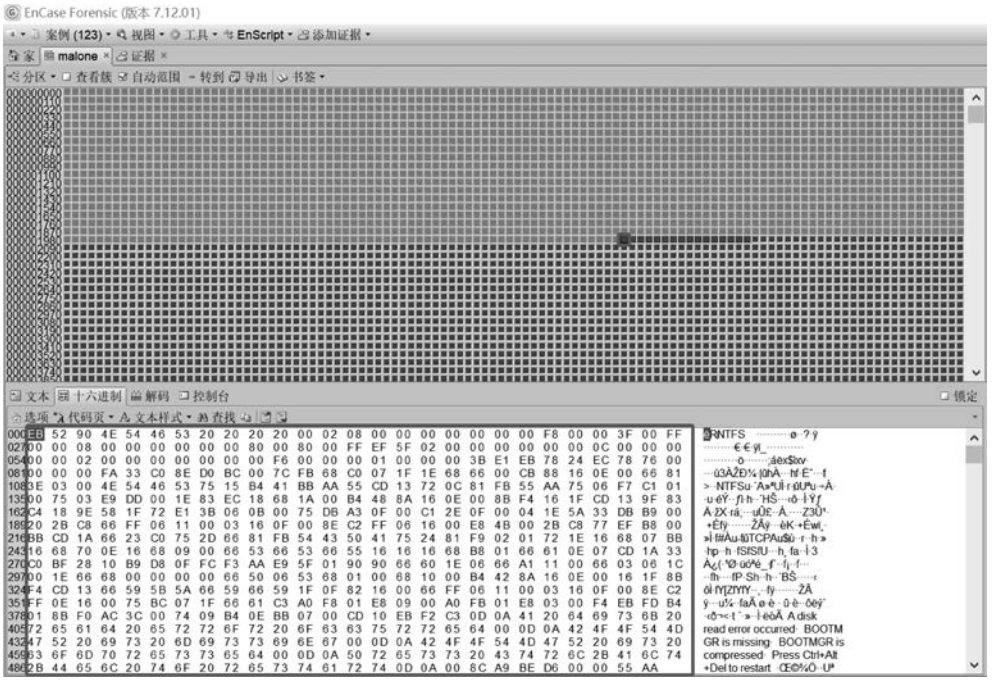


图 3-73　分区 1DBR

步骤 8：同样，在扇区视图下，鼠标停留在第一个扇区并右击，在弹出的快捷菜单中选择“转到”选项，跳转到 40663040 号扇区，即为分区 2 的起始扇区 DBR，如图 3-74 所示。由此找到了磁盘上的两个分区，可进一步对分区引导记录 DBR 解析，获取各分区重要参数信息。

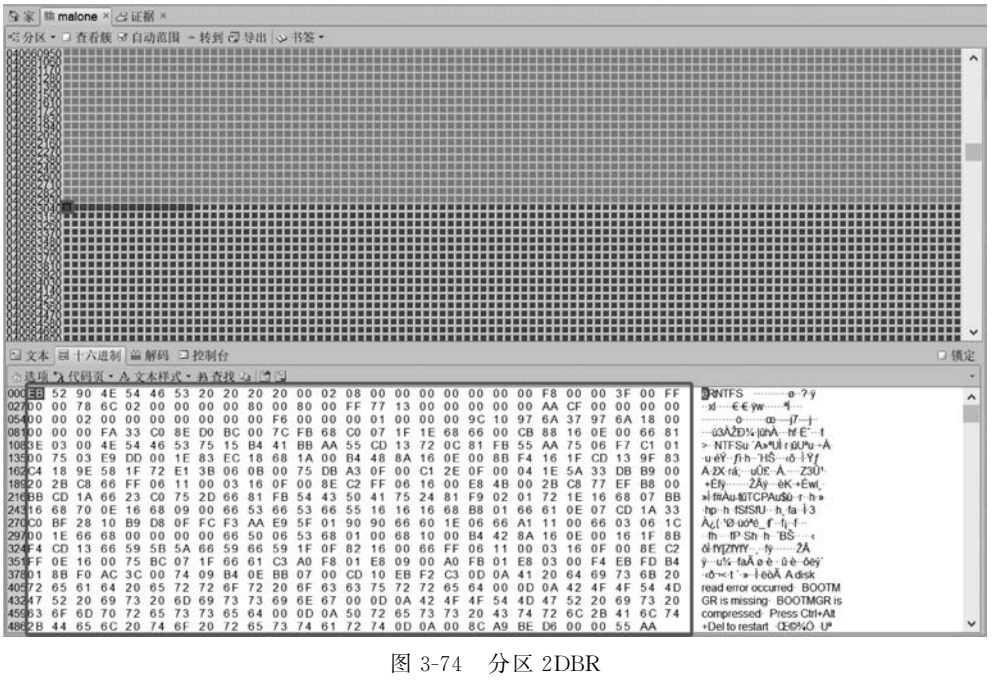


图 3-74　分区 2DBR

步骤 9：分区 1 与分区 2 大小相加为 41117696 扇区，而物理磁盘总大小为 59.8GB (125337600 扇区)，如图 3-75 所示。由此可见，已知的两个分区仅占了该磁盘的小部分空

间。存在未使用磁盘空间或删除分区的情况。

图 3-75　磁盘基本信息

步骤 10：由分区 1 的起始扇区与占用扇区可计算得到分区 1 的结束扇区，发现分区 1 的结束扇区与分区 2 的起始扇区之间存在大量松弛区，不符合常理，判断可能存在删除分区。

步骤 11：跳转到分区 1 结束扇区后的 1 扇区，即 39843840 号扇区，查看该扇区十六进制数据，猜测可能为 DBR，且分区文件系统为 FAT32。单击工具栏中的“分区”按钮，选择“添加分区”选项，如图 3-76 所示。在弹出的“添加分区”对话框中，选择卷类型为“FAT32”，单击“确定”按钮，如图 3-77 所示。

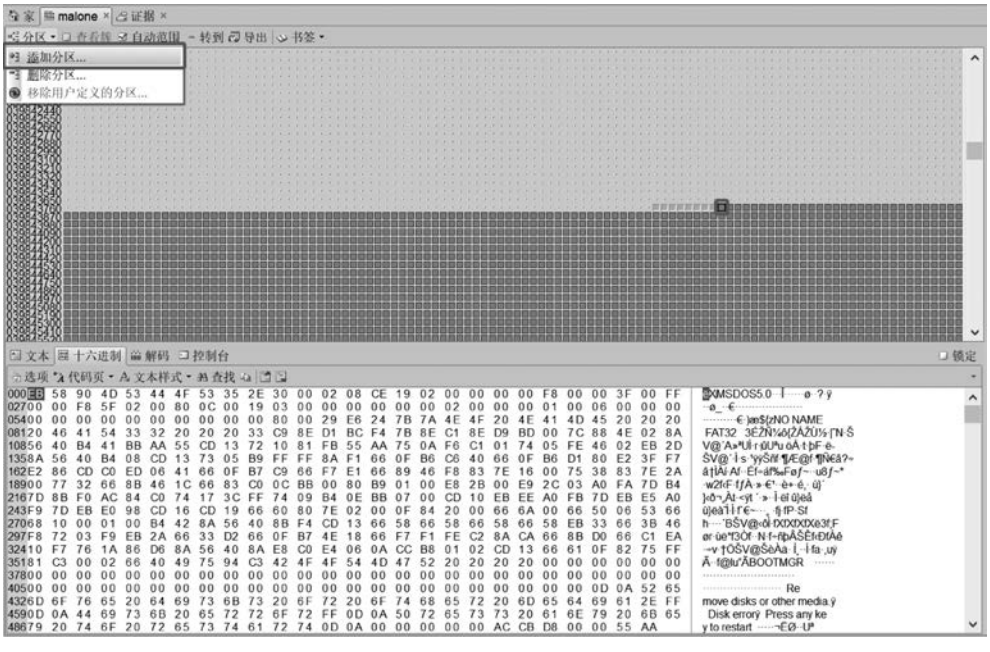


图 3-76　分区的添加

步骤 12：关闭并重新打开该证据文件（EnCase 不能自动刷新，必须手动关闭重新进入），发现除了原有的两个分区外，多了一个分区，即完成了删除分区的恢复，如图 3-78 所示。

图 3-77 “添加分区”对话框

图 3-78 删除分区的信息

3.6.4　实验小结

当磁盘中的某个分区被删除后，分区中的数据并未被真正删除。此时，对应的分区表项会被清零，被删除的分区变为未分配状态且数据不可访问。要恢复被删除的分区，需要找出分区的起始位置、大小和分区类型等重要信息，然后将其写回被清零的分区表项。通常来说，在磁盘中创建多个分区，分区会占用所有磁盘空间。此时，通过查看现有的分区信息，找到被删除分区的位置并不复杂。接着，进一步分析被删除分区的具体数据可以判断出文件系统类型。

EnCase 具备了分区恢复功能，在分区恢复时，应该首先检查主分区表是否损坏，通过查看现有分区表描述的各个分区的前后关系是否合理、跳转到分区起始扇区查看是否为正常的 DBR 等来综合分析、判断并恢复分区。

读者也可利用 WinHex 工具尝试删除分区的恢复操作。

3.7　FAT 文件系统数据恢复

3.7.1　预备知识：FAT 文件系统原理

文件系统是操作系统用于明确磁盘或分区上的文件的保存方法和数据结构，即在磁盘上组织文件的方法。一个分区或磁盘作为文件系统使用前需要初始化，并将数据结构写到磁盘上，这个过程就叫建立文件系统。FAT32、exFAT、NTFS 是目前最常见的三种文件系统。

FAT(file allocation table，文件分配表)文件系统是 Windows 操作系统所使用的一种文件系统，它的发展过程经历了 FAT12、FAT16、FAT32 三个阶段。FAT 文件系统用“簇”作为数据单元。一个“簇”由一组连续的扇区组成，簇所含的扇区数必须是 2 的整数次幂。所有簇从 2 开始进行编号，每个簇都有一个自己的地址编号。用户文件和目录都存储在簇中。

FAT 文件系统由保留扇区、FAT 区和数据区组成，数据结构如图 3-79 所示。

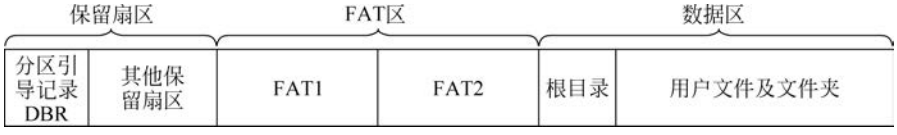


图 3-79　FAT 文件系统数据结构

1. DBR 区

分区引导记录 DBR，也称为操作系统引导记录，位于相对扇区 0 扇区。DBR 由 5 个部分组成：

① 0x00～0x02：跳转指令。FAT32 文件系统跳转指令为“EB 58 90”。在汇编当中 0xEB 是跳转指令，0x58 是跳转的地址，而 0x90 则是空指令。CPU 读取到 EB 58 这个指令时，便跳转到 0x58 这个地址并继续读取指令来执行，而 0x58 地址之后的内容通常都是载入操作系统的指令。

② 0x03～0x0A：OEM(original entrusted manufacture，代工厂商)代号。

③ 0x0B～0x59：BPB(BIOS parameter block，本分区参数记录表)。BPB 参数块记录着本分区的起始扇区、结束扇区、文件存储格式、硬盘介质描述符、根目录大小、FAT 个数、簇的大小等重要参数。具体如表 3-9 所示。

表 3-9 BPB 参数信息

偏 移 量	字 节 数	含 义
0x0B	2	每扇区字数
0x0D	1	每簇扇区数
0x0E	2	保留扇区数
0x10	1	FAT 个数
0x11	2	根目录项数，FAT32 以突破该限制，无效
0x13	2	扇区总数，小于 32M 使用
0x15	1	存储介质描述符
0x16	2	每 FAT 表占用扇区数，小于 32M 使用
0x18	2	逻辑每磁道扇区数
0x1A	2	逻辑磁头数
0x1C	4	系统隐含扇区数
0x20	4	扇区总数，大于 32M 使用
0x24	4	每 FAT 表扇区数，大于 32M 使用
0x28	2	标记
0x2A	2	版本（通常为零）
0x2C	4	根目录起始簇
0x30	2	Boot 占用扇区数
0x32	2	备份引导扇区位置
0x34	14	保留
0x42	1	扩展引导标记
0x43	4	序列号
0x47	10	卷标
0x52	8	文件系统

④ 0x5A～0x1FD：引导程序。在 Windows 98 之前的系统中，这段代码负责完成 DOS 三个系统文件的装入。在 Windows 2000 之后的系统中，这段代码负责完成将系统文件 NTLDR 装入，对于一个没有安装操作系统的分区来讲，这段程序没有用处。

⑤ 0x1FE～0x1FF：结束标志。DBR 的结束标志与 MBR、EBR 的结束标志都相同，为“55 AA”。

2. FAT 区

文件分配表 FAT 是用来描述文件系统内存储单元的分配状态及文件内容前后链接关系的表格。它对于 FAT 文件系统来讲是至关重要的一个组成部分，假若丢失 FAT，那么硬盘上的数据就无法定位，也就不能使用了。由于 FAT 对文件管理的重要性，FAT 有一个备份，即在原 FAT1 后再建一个同样的 FAT2。

根据 FAT 文件系统数据结构可知，FAT1 的起始扇区可由 BPB 中记载的保留扇区数而获知(保留扇区数的信息位于 BPB 模块 0x0E～0x0F 两个字节)，FAT2 的起始扇区可由保留扇区数＋FAT1 占用扇区数(BPB 模块 0x24～0x27)计算所得。

FAT是由一个个表项组成，其中每一个表项的值对应了相应簇的使用情况，如2号表项对应了2号簇的使用情况，3号表项对应了3号簇的使用情况，以此类推（但是第0项和第1项例外）。FAT第0项和第1项是系统保留，记录分区所在的介质类型和分区状态。

FAT32的每个FAT项的大小为32位，相当于4字节，即从00-00-00-00～FF-FF-FF-FF，不同数值具体含义如下：

① 空闲簇（未分配簇）：00-00-00-00；

② 系统保留簇：00-00-00-01；

③ 被占用的簇，其值指向下一个簇号：00-00-00-02～0F-FF-FF-EF；

④ 保留数值：0F-FF-FF-F0～0F-FF-FF-F6；

⑤ 坏簇：0F-FF-FF-F7；

⑥ 文件最后一个簇：0F-FF-FF-F8～0F-FF-FF-FF。

FAT表项的填写规则是：如果该簇是文件的最后一簇，填入的值为0x0F-FF-FF-FF；如果该簇不是文件的最后一簇，则填入的值为该文件占用的下一簇号。

3. FDT区

文件目录表（File Directory Table，FDT）也称为根目录，位于数据区头部（第2簇），用来存放根目录下的文件的目录项。

根据FAT文件系统数据结构可知，根目录起始扇区＝保留扇区数＋FAT扇区数×2。

FDT区是由一个个目录项构成，类似于FAT。每一个目录项占用32字节，记录文件或者文件夹的名称、属性、大小、起始簇号、创建时间、创建日期、最近访问日期、最近修改日期等内容，具体如表3-10所示。

表3-10　FDT信息

偏移量	字节数	含　义
0x00	8	文件名
0x08	3	后缀名
0x0B	1	文件属性（00H读写；01H只读；02H隐藏；04H系统；08H卷标；10H子目录；20H归档）
0x0C	1	系统保留
0x0D	1	创建时间的10毫秒位
0x0E	2	文件创建时间
0x10	2	文件创建日期
0x12	2	文件最后访问日期
0x14	2	文件起始簇号高16位
0x16	2	文件最近修改时间
0x18	2	文件最近修改日期
0x1A	2	文件起始簇号低16位
0x1C	4	文件长度

值得注意的是：FAT分区下，文件在被删除之后，文件对应的文件目录项的第一个字节会被改为0xE5，表示该文件被删除，而文件目录项的其他字节没有变化，所以被删除的文件仍旧能够找到其起始簇，从而使得该文件是可恢复的。

3.7.2 实验目的与条件

1. 实验目的

通过本实验，读者重点掌握以下内容：

(1) 了解 FAT32 文件系统存储原理；

(2) 掌握 FAT32 文件系统各数据结构的解析；

(3) 掌握使用 WinHex 进行 FAT32 文件系统数据恢复的过程。

2. 实验条件

本实验所需要的软硬件清单如表 3-11 所示。

表 3-11 FAT32 文件系统数据恢复实验清单

序　　号	设　　备	数　　量	参　　数
1	取证工作站	1 台	Windows XP 以上
2	WinHex 工具	1 套	无

3.7.3 实验过程

1. 创建虚拟磁盘 VHD

步骤 1：打开计算机管理中的磁盘管理。单击工具栏中的“操作”按钮，选择“创建 VHD”选项，如图 3-80 所示。

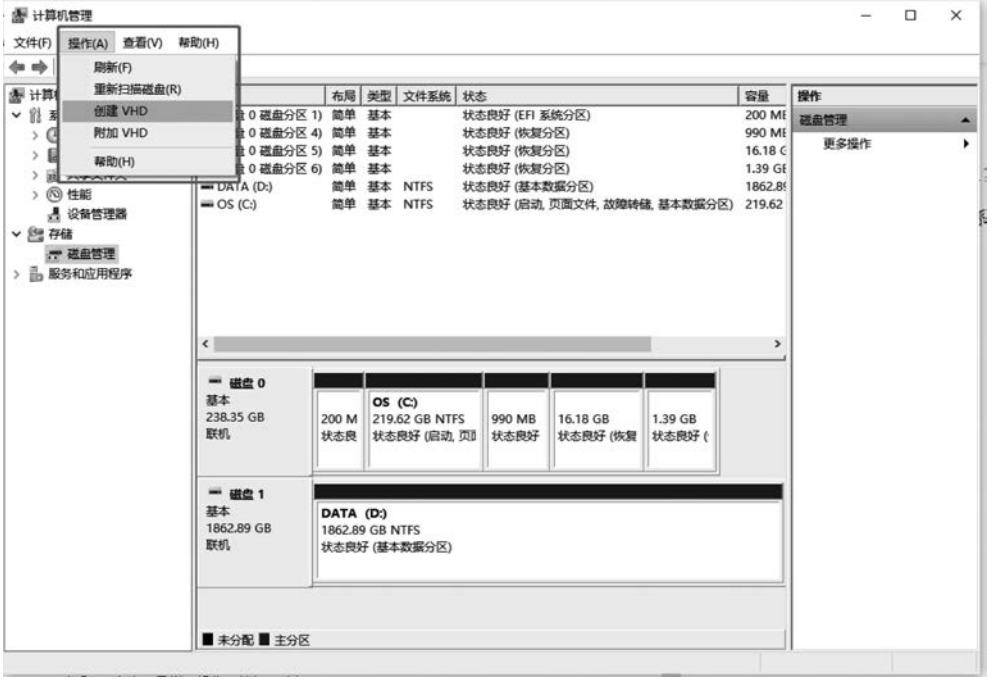


图 3-80 磁盘管理创建 VHD

步骤 2：在弹出的对话框中选择路径及虚拟磁盘大小，单击“确定”按钮，如图 3-81 所示。

步骤 3：此时，在磁盘管理视图下，可以看到多了一个未初始化磁盘。在该磁盘左边部分右击，在弹出的快捷菜单中选择“初始化磁盘”选项，设置磁盘分区形式为“MBR”，如图 3-82 所示。

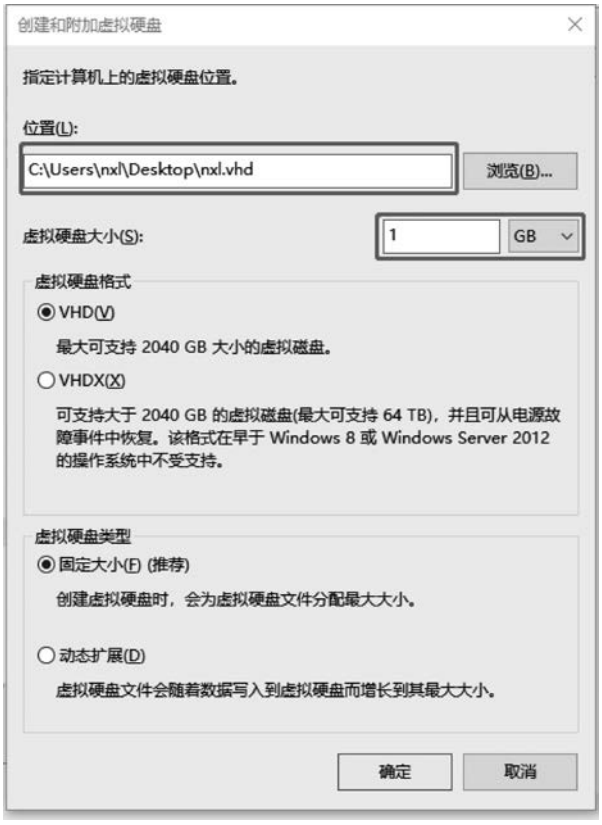


图 3-81　创建和附加虚拟磁盘

图 3-82　初始化磁盘

步骤 4：此时，在磁盘管理视图下可以看到该磁盘显示为“联机”状态，在该磁盘右边部分右击，在弹出的快捷菜单中选择“新建简单卷”选项，如图 3-83 所示。

步骤 5：在新建简单卷向导下按需要设置逻辑卷属性信息，如卷大小、卷标等。在格式化分区步骤，选择文件系统类型为“FAT32”文件系统，如图 3-84 所示。

图 3-83 新建简单卷

图 3-84 格式化分区

步骤 6：完成后，发现在此电脑下，多了一个 FAT32 文件系统的分区 E，如图 3-85 所示，至此，完成 VHD 的创建。

2. 连续存储文件的删除恢复

步骤 1：在上述虚拟磁盘 E 中，存入一个图片文件 lena.jpg 并删除该文件。

步骤 2：用 WinHex 工具打开该 E 盘，找到该分区的起始扇区，即引导记录 DBR，如图 3-86 所示。

步骤 3：分析引导记录 DBR，获得以下信息（小端）：

（1）0x0D：每簇扇区数：08：8；

（2）0x24-0x27：FAT 占用扇区数：00-00-07-F3：2035；

（3）0x0E-0x0F：保留扇区数：10-1A：4122。

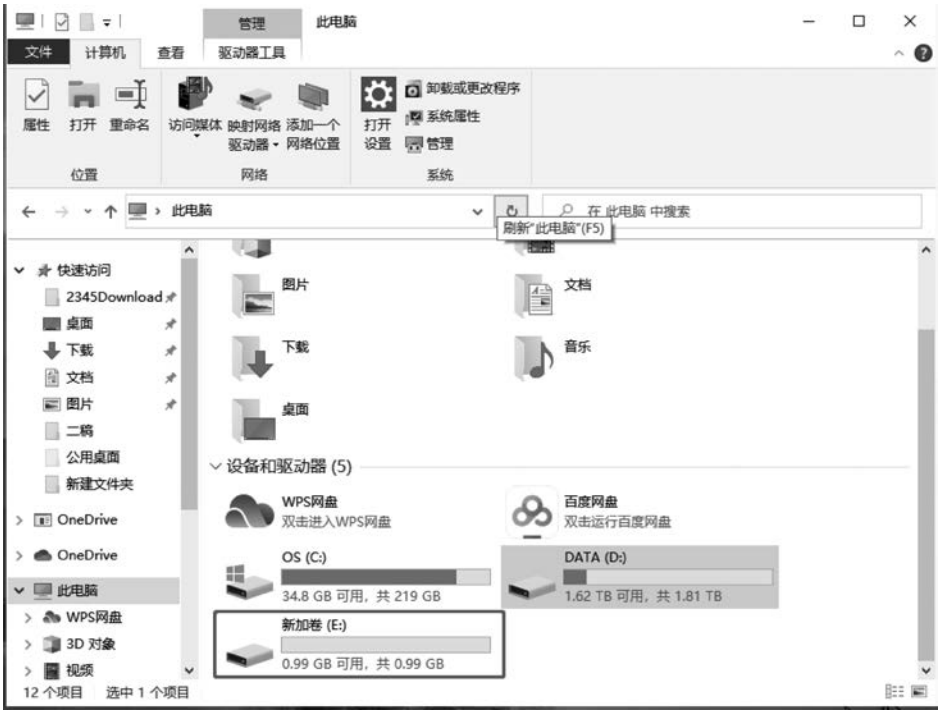


图 3-85　虚拟磁盘 E

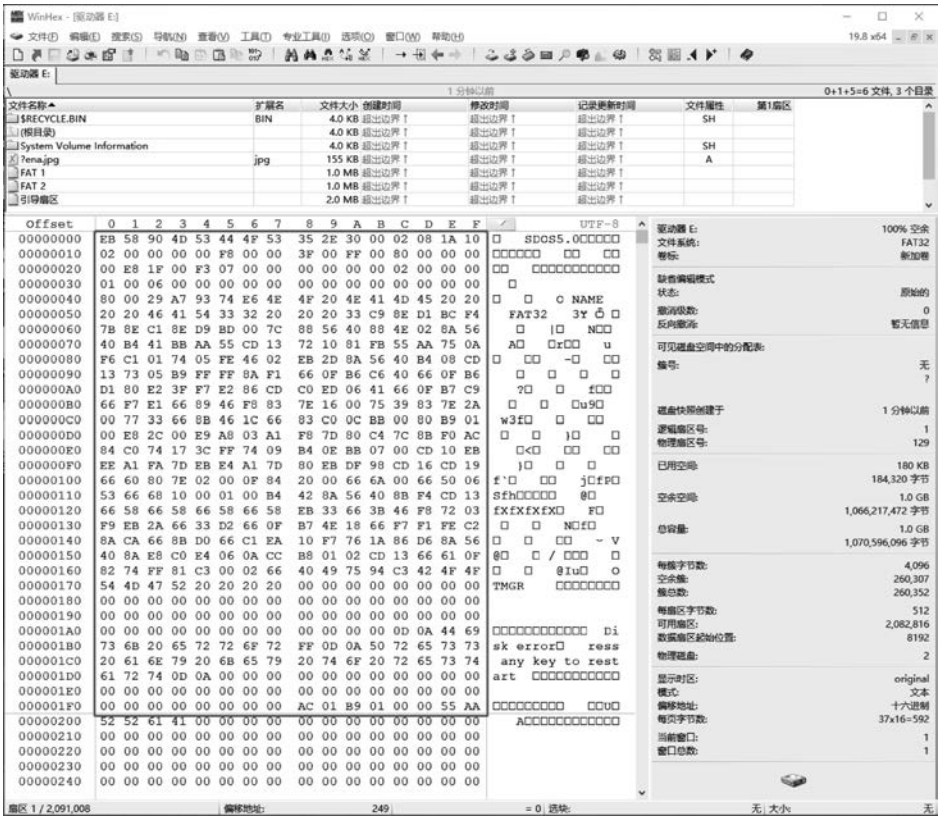


图 3-86　DBR 扇区

步骤 4：单击工具栏中的“跳转扇区”按钮，根据保留扇区数（图 3-87），跳转到 FAT1 起始扇区，如图 3-88 所示。

图 3-87 跳至扇区

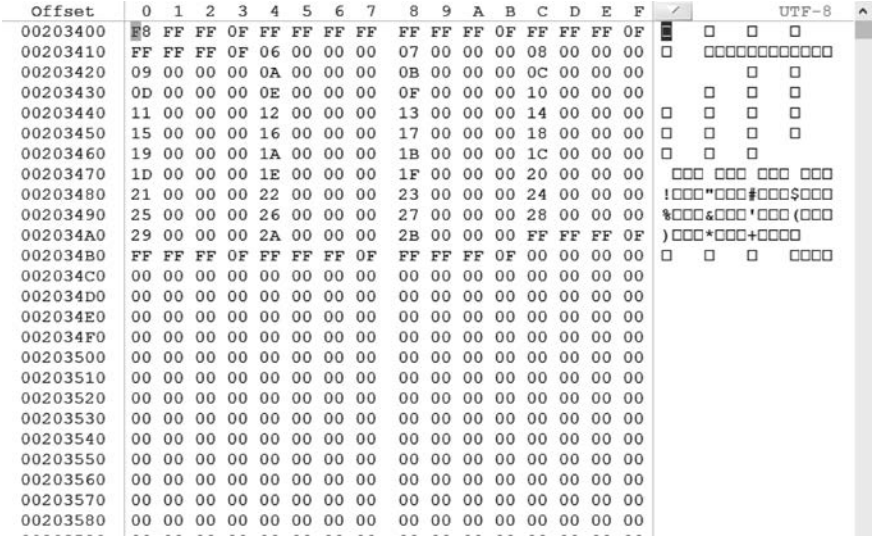


图 3-88 FAT1 起始扇区

步骤 5：同样，根据保留扇区数＋FAT 扇区数×2＝4122＋2035×2＝8192，跳转到数据区的起始 2 号簇，即 FDT 区，如图 3-89 所示。

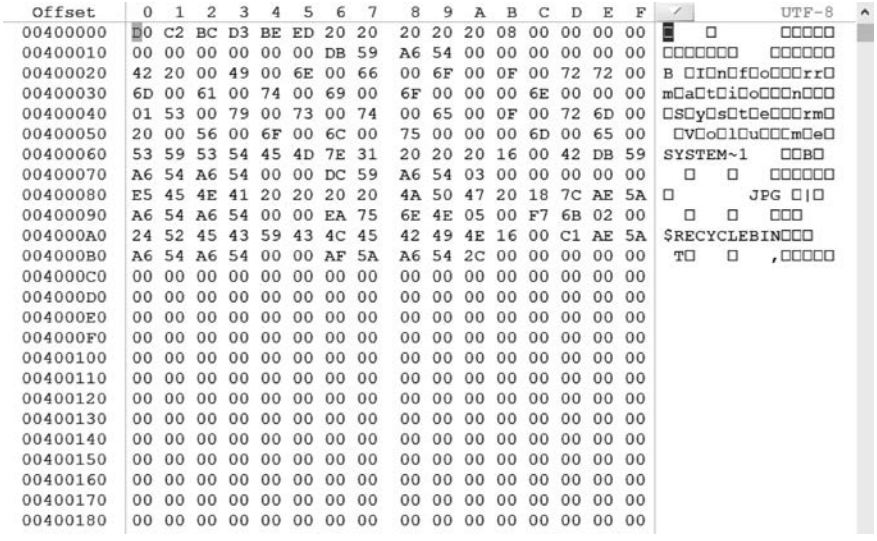
步骤 6：解析 FDT 区的删除图片目录项（图 3-90），获得该文件相关信息：

（1）文件后缀名：JPG；

（2）文件起始簇：0x14-0x15＋0x1A-0x1B：00-00-00-05：5 号簇；

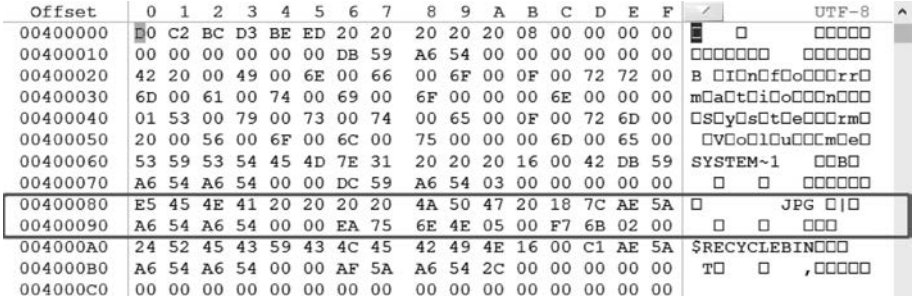
（3）文件大小：0x1C-0x1F：00-02-6B-F7：158711 字节。

步骤 7：分析由于 VHD 分区上只存在过该图片文件，因此该文件应为简单连续存储（查看 FAT 可以验证），既找到了文件的起始簇，又知晓了文件大小，就能找到文件结尾。



Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	UTF-8
00400000	E0	C2	BC	D3	BE	ED	20	20	20	20	20	08	00	00	00	00	□ □ □□□□□
00400010	00	00	00	00	00	00	DB	59	A6	54	00	00	00	00	00	00	□□□□□□□ □□□□□□
00400020	42	20	00	49	00	6E	00	66	00	6F	00	0F	00	72	72	00	B □I□n□f□o□□□rr□
00400030	6D	00	61	00	74	00	69	00	6F	00	00	00	6E	00	00	00	m□a□t□i□o□□□n□□□
00400040	01	53	00	79	00	73	00	74	00	65	00	0F	00	72	6D	00	□S□y□s□t□e□□□rm□
00400050	20	00	56	00	6F	00	6C	00	75	00	00	00	6D	00	65	00	□V□o□l□u□□□m□e□
00400060	53	59	53	54	45	4D	7E	31	20	20	20	16	00	42	DB	59	SYSTEM~1 □□B□
00400070	A6	54	A6	54	00	00	DC	59	A6	54	03	00	00	00	00	00	□ □ □□□□□□
00400080	E5	45	4E	41	20	20	20	20	4A	50	47	20	18	7C	AE	5A	□ JPG □|□
00400090	A6	54	A6	54	00	00	EA	75	6E	4E	05	00	F7	6B	02	00	□ □ □□□
004000A0	24	52	45	43	59	43	4C	45	42	49	4E	16	00	C1	AE	5A	$RECYCLEBIN□□□
004000B0	A6	54	A6	54	00	00	AF	5A	A6	54	2C	00	00	00	00	00	T□ □ ,□□□□□
004000C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
004000D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
004000E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
004000F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00400100	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00400110	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00400120	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00400130	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00400140	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00400150	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00400160	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00400170	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00400180	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	

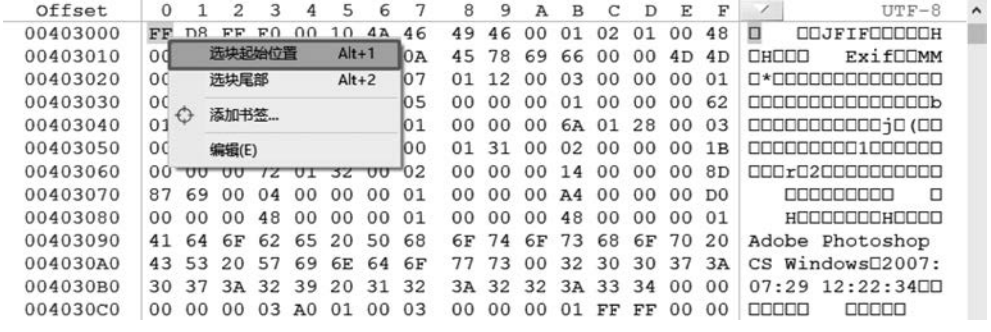
图 3-89　FDT 区



Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	UTF-8
00400000	E0	C2	BC	D3	BE	ED	20	20	20	20	20	08	00	00	00	00	□ □ □□□□□
00400010	00	00	00	00	00	00	DB	59	A6	54	00	00	00	00	00	00	□□□□□□□ □□□□□□
00400020	42	20	00	49	00	6E	00	66	00	6F	00	0F	00	72	72	00	B □I□n□f□o□□□rr□
00400030	6D	00	61	00	74	00	69	00	6F	00	00	00	6E	00	00	00	m□a□t□i□o□□□n□□□
00400040	01	53	00	79	00	73	00	74	00	65	00	0F	00	72	6D	00	□S□y□s□t□e□□□rm□
00400050	20	00	56	00	6F	00	6C	00	75	00	00	00	6D	00	65	00	□V□o□l□u□□□m□e□
00400060	53	59	53	54	45	4D	7E	31	20	20	20	16	00	42	DB	59	SYSTEM~1 □□B□
00400070	A6	54	A6	54	00	00	DC	59	A6	54	03	00	00	00	00	00	□ □ □□□□□□
00400080	E5	45	4E	41	20	20	20	20	4A	50	47	20	18	7C	AE	5A	□ JPG □|□
00400090	A6	54	A6	54	00	00	EA	75	6E	4E	05	00	F7	6B	02	00	□ □ □□□
004000A0	24	52	45	43	59	43	4C	45	42	49	4E	16	00	C1	AE	5A	$RECYCLEBIN□□□
004000B0	A6	54	A6	54	00	00	AF	5A	A6	54	2C	00	00	00	00	00	T□ □ ,□□□□□
004000C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	

图 3-90　删除文件的目录项

步骤 8：跳转到 5 号簇，即为文件的起始簇，选中第一个字节，即为文件首字节，右击该字节，在弹出的快捷菜单中选择“选块起始位置”选项，如图 3-91 所示。



Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	UTF-8
00403000	FF	D8	FF	E0	00	10	4A	46	49	46	00	01	02	01	00	48	□ □□JFIF□□□□□H
00403010	00						0A	45	78	69	66	00	00	4D	4D		□H□□□ Exif□□MM
00403020	00						07	01	12	00	03	00	00	00	01		□*□□□□□□□□□□□□□□□
00403030	00						05	00	00	00	01	00	00	00	62		□□□□□□□□□□□□□□□b
00403040	01						01	00	00	00	6A	01	28	00	03		□□□□□□□□□□□j□(□□
00403050	00						00	01	31	00	02	00	00	00	1B		□□□□□□□□□1□□□□□□
00403060	00	00	00	72	01	32	00	02	00	00	00	14	00	00	00	8D	□□□r□2□□□□□□□□□□
00403070	87	69	00	04	00	00	00	01	00	00	00	A4	00	00	00	D0	□□□□□□□□□ □
00403080	00	00	00	48	00	00	00	01	00	00	00	48	00	00	00	01	H□□□□□□□H□□□□
00403090	41	64	6F	62	65	20	50	68	6F	74	6F	73	68	6F	70	20	Adobe Photoshop
004030A0	43	53	20	57	69	6E	64	6F	77	73	00	32	30	30	37	3A	CS Windows□2007:
004030B0	30	37	3A	32	39	20	31	32	3A	32	32	3A	33	34	00	00	07:29 12:22:34□□
004030C0	00	00	00	03	A0	01	00	03	00	00	00	01	FF	FF	00	00	□□□□□ □□□□□

图 3-91　图片文件首字节

步骤 9：相对于当前文件首字节位置，根据文件大小 0x26BF7＝158711 字节，跳转到文件的尾字节，如图 3-92、图 3-93 所示。

步骤 10：在文件尾字节“D9”上右击，在弹出的快捷菜单中选择“选块尾部”选项，即文件首尾之间所有字节都被选中（文件所有字节）。在选中的字节上右击，在弹出的快捷菜单中选择“编辑”→“复制选块”→“至新文件”选项，如图 3-94 所示，将选中区域保存成一个文件。

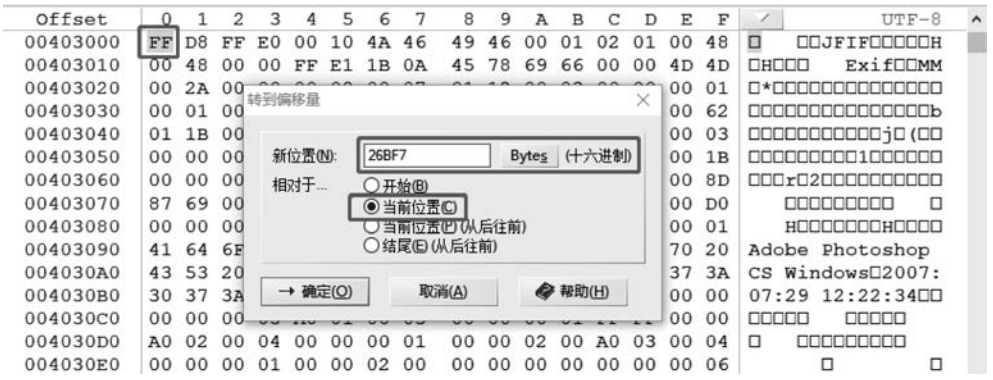


图 3-92 转到偏移量

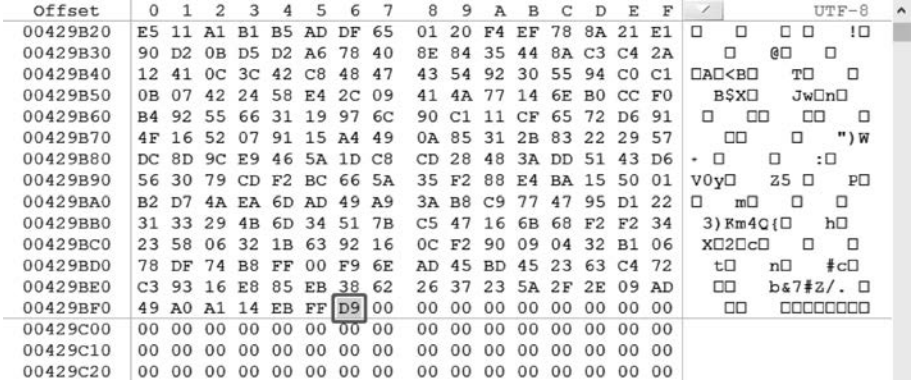


图 3-93 图片文件尾字节

图 3-94 复制文件选块

步骤 11：选择保存路径为桌面，文件名为“恢复.jpg”(图 3-95)，单击“保存”按钮，即在桌面上成功恢复该图片文件，如图 3-96 所示。

3. 非连续存储文件的删除恢复

步骤 1：在 FAT32 分区(VHD)中创建一个文本文件 test.txt，写入一些数据后保存。

图 3-95　恢复图片的保存路径

图 3-96　桌面上恢复的图片文件

然后在该分区中存入另一文件，再次打开 test.txt 文件对其中内容进行增加（增加内容超过1簇），可多次重复该操作，文件会更“碎片”地存储。最后删除 test.txt 文件。

步骤 2：用 WinHex 打开该分区，找到该分区的起始扇区，即引导记录 DBR。通过解析获知每簇扇区数、FAT 区起始扇区、FDT 区起始扇区。解析过程同上，不再赘述。

步骤 3：找到该删除文件的目录项，如图 3-97 所示。

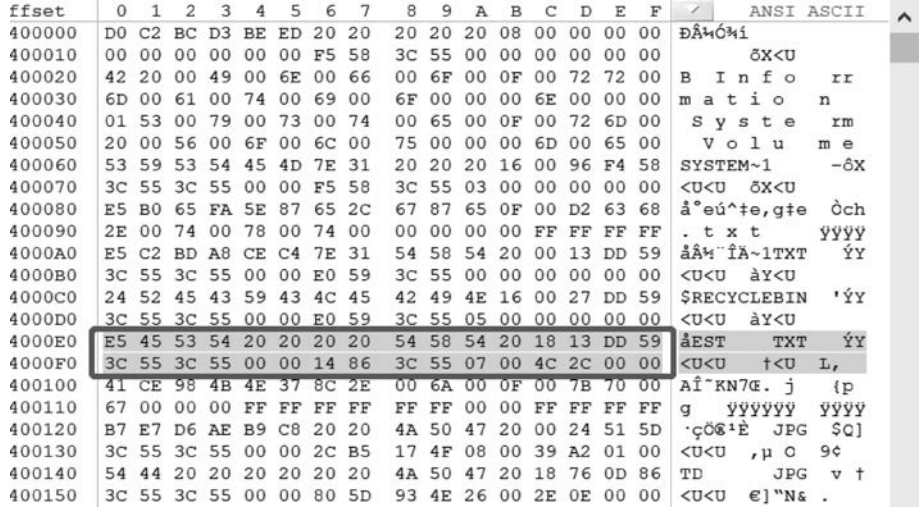


图 3-97　删除文件的目录项

步骤 4：分析该文件目录项，获得以下信息：

(1) 文件的起始簇号高 16 位：0x14～0x15：00-00；

(2) 文件的起始簇号低 16 位：0x1A～0x1B：00-07；

(3) 文件的长度 0x1C～0x1F：00-00-2C-4C。

可得：

文件的起始簇为：7 号簇；

文件大小为：11340 字节。

步骤 5：跳转到 FAT1 位置，查看 FAT 中 7 号表项内容(对应数据区 7 号簇的状态)，7 号表项内容为 00-00-00-25(小端)，如图 3-98 所示，说明文件存储的下一簇为 0x25，即 37 号簇。

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ANSI ASCII
00203400	F8	FF	FF	0F	FF	FF	FF	FF	FF	FF	FF	0F	FF	FF	FF	0F	øÿÿ ÿÿÿÿÿÿÿ ÿÿÿ
00203410	FF	FF	FF	0F	FF	FF	FF	0F	FF	FF	FF	0F	25	00	00	00	ÿÿÿ ÿÿÿ ÿÿÿ %
00203420	09	00	00	00	0A	00	00	00	0B	00	00	00	0C	00	00	00	
00203430	0D	00	00	00	0E	00	00	00	0F	00	00	00	10	00	00	00	
00203440	11	00	00	00	12	00	00	00	13	00	00	00	14	00	00	00	
00203450	15	00	00	00	16	00	00	00	17	00	00	00	18	00	00	00	
00203460	19	00	00	00	1A	00	00	00	1B	00	00	00	1C	00	00	00	
00203470	1D	00	00	00	1E	00	00	00	1F	00	00	00	20	00	00	00	
00203480	21	00	00	00	22	00	00	00	FF	FF	FF	0F	FF	FF	FF	0F	! " ÿÿÿ ÿÿÿ
00203490	FF	FF	FF	0F	27	00	00	00	FF	FF	FF	0F	FF	FF	FF	0F	ÿÿÿ ' ÿÿÿ ÿÿÿ
002034A0	FF	FF	FF	0F	00	00	00	00	00	00	00	00	00	00	00	00	ÿÿÿ

图 3-98 FAT1 中 7 号表项内容

步骤 6：同样地在 FAT 中查看 37 号簇的状态，FAT 中 37 号表项为 00-00-00-27(39)，如图 3-99 所示，说明 37 号簇的下一簇为 39 号簇。

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ANSI ASCII
00203400	F8	FF	FF	0F	FF	FF	FF	FF	FF	FF	FF	0F	FF	FF	FF	0F	øÿÿ ÿÿÿÿÿÿÿ ÿÿÿ
00203410	FF	FF	FF	0F	FF	FF	FF	0F	FF	FF	FF	0F	25	00	00	00	ÿÿÿ ÿÿÿ ÿÿÿ %
00203420	09	00	00	00	0A	00	00	00	0B	00	00	00	0C	00	00	00	
00203430	0D	00	00	00	0E	00	00	00	0F	00	00	00	10	00	00	00	
00203440	11	00	00	00	12	00	00	00	13	00	00	00	14	00	00	00	
00203450	15	00	00	00	16	00	00	00	17	00	00	00	18	00	00	00	
00203460	19	00	00	00	1A	00	00	00	1B	00	00	00	1C	00	00	00	
00203470	1D	00	00	00	1E	00	00	00	1F	00	00	00	20	00	00	00	
00203480	21	00	00	00	22	00	00	00	FF	FF	FF	0F	FF	FF	FF	0F	! " ÿÿÿ ÿÿÿ
00203490	FF	FF	FF	0F	27	00	00	00	FF	FF	FF	0F	FF	FF	FF	0F	ÿÿÿ ' ÿÿÿ ÿÿÿ
002034A0	FF	FF	FF	0F	00	00	00	00	00	00	00	00	00	00	00	00	ÿÿÿ
002034B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	

图 3-99 FAT1 中 37 号表项内容

步骤 7：以此类推，可以发现 39 号表项的状态为 0F-FF-FF-FF(结束标识)。

由此可知，文件一共分为 3 块存储：7 号簇、37 号簇、39 号簇，其中 39 号簇为文件存储的最后一簇，一般情况下未存满。

步骤 8：选中 7 号簇(8 扇区)所有字节，保存为新文件在桌面上，命名为“1”，如图 3-100 所示。同样，选中 37 号簇所有字节，保存在桌面上，命名为“2”。

步骤 9：找到 39 号簇的起始字节，右击，在弹出的快捷菜单中选择“选块起始”选项，找到 39 号簇中最后一个字节，右击，在弹出的快捷菜单中选择“选块结束”选项，如图 3-101 所示，保存为新文件在桌面上，命名为“3”。此时，文件的 3 个分块都已导出在桌面上，如图 3-102 所示。

步骤 10：打开 cmd 命令行工具，进入 3 个文件分块所在的目录下，利用 cmd 命令行工具“copy”命令，将文件的 3 个分块组合成 1 个文件，如图 3-103 所示。即在桌面上生成了 txt 文件，完成了碎片文件的恢复，如图 3-104 所示。

图 3-100　导出 7 号簇所有字节数据

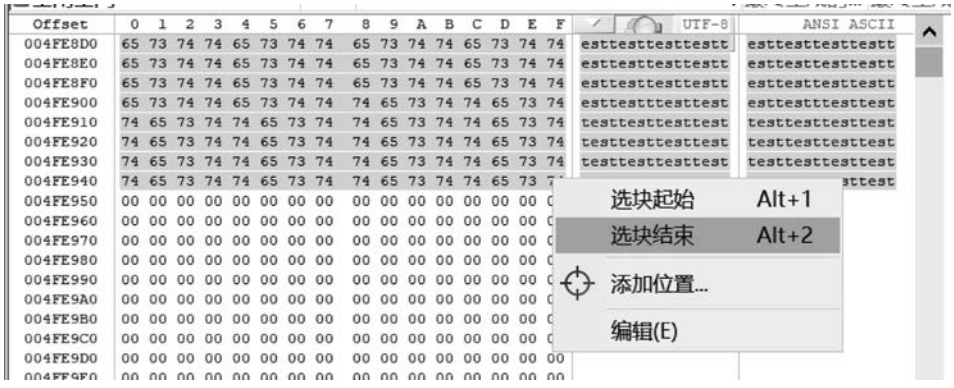


图 3-101　导出文件第三块内容

图 3-102　恢复的文件三个分片

图 3-103 “copy”命令组合文件碎片

图 3-104 拼接完成的 txt 文件

3.7.4 实验小结

磁盘上的文件常常要进行创建、删除、增长、缩短等操作。这样的操作越多，盘上的文件就被存储得越零散。即同一个文件的数据并不一定完整地存放在磁盘的一个连续的区域内，而往往会分成若干段，像一条链子一样存放，这种存储方式称为文件的链式存储。而FAT是实现文件链式存储的关键。

在取证过程中，恢复被删除或丢失的数据是一项重要的工作。当一个文件被删除时，文件系统中存储的文件内容等数据不会立即消失。已删除的数据在被其他的新数据覆盖之前，一直都完整地存在于原始位置。当在FAT文件系统中删除一个文件时，操作系统会更新目录项，将文件目录项的第一个字节设置为一个特殊字符，即十六进制的0xE5，表示这是一个被删除的目录项。除第一个字节之外，目录项中的其他位置信息都未发生变化。换句话说，目录项中剩余的文件名、扩展名、创建日期和时间、权限、大小、文件起始簇地址等均保持不变。且文件的数据在数据区中保持不变。

基于此，我们找到文件的目录项并解析后，可根据FAT进一步获知文件的存储“链”如何构成，而后即可在数据区找到文件的各个分片，拼接出完整的文件内容。

3.8 NTFS文件系统数据恢复

3.8.1 预备知识：NTFS文件系统原理

NTFS的英文全称为“new technology file system”，中文意为NT文件系统，是Windows NT以及之后操作系统的标准文件系统，具有安全性、可恢复性、容错性、文件压缩、硬盘配额等

优势。FAT32 文件系统的出现对于 FAT16 而言，可以说是有了比较明显的改善，但 NTFS 对 FAT32 的改进，就必须得用“卓越”来形容了。

NTFS 文件系统同 FAT32 文件系统一样，也是用“簇”为存储单位，一个文件总是占用一个或多个簇。但与 FAT32 文件系统不同的是，NTFS 文件系统将所有的数据，包括文件系统管理数据都作为文件进行管理，所以 NTFS 文件系统中所有扇区都被分配以簇号，并从 0 开始对所有簇进行编号，文件系统的 0 号扇区为 0 号簇的起始位置。

NTFS 文件系统使用逻辑簇号(LCN)和虚拟簇号(VCN)对分区进行管理。

逻辑簇号：即对分区内的第一个簇到最后一个簇进行编号，NTFS 使用逻辑簇号对簇进行定位。

虚拟簇号：即将文件所占用的簇从开头到结尾进行编号，虚拟簇号不要求在物理上是连续的。

一个 NTFS 系统是由分区引导扇区、主文件表(MFT)和数据区组成，另外 MFT 有一部分重要备份在数据区，数据结构如图 3-105 所示。

图 3-105　NTFS 文件系统数据结构

1. 分区引导扇区

分区引导扇区包含了 NTFS 文件系统结构的关键信息。与 FAT 的引导扇区类似，NTFS 的引导扇区描述了文件系统的数据结构，如簇大小、MFT 项(MFT entry，或称 MFT 文件记录项)大小及 MFT 起始簇地址等。

分区引导扇区中的第一个扇区为 DBR，由“跳转指令”“OEM 代号”“BPB”“引导程序”和“结束标志”组成，这里和 FAT32 文件系统的 DBR 一样，具体如下：

① 0x00～0x02：跳转指令。NTFS 文件系统中跳转指令为“EB 52 90”，意为转到 0x52 字节。

② 0x03～0x0A：OEM 代号。固定为“4E-54-46-53-20-20-20-20”，表示“NTFS”。

③ 0x0B～0x53：BPB。记录了有关该文件系统的重要信息，共 73 字节，具体见表 3-12。

表 3-12　BPB 参数信息

偏　移　量	字　节　数	含　　义
0x0B	2	每扇区字数
0x0D	1	每簇扇区数
0x0E	2	保留扇区数
0x10	3	总为 0
0x13	1	不使用
0x14	2	存储介质描述符，硬盘为 F8
0x16	2	总为 0
0x18	2	逻辑每磁道扇区数
0x1A	2	逻辑磁头数
0x1C	4	系统隐含扇区数

续表

偏 移 量	字 节 数	含 义
0x20	4	不使用
0x24	4	不使用,总为 80 00 80 00
0x28	8	扇区总数,即分区大小
0x30	8	$MFT 的开始簇号
0x38	8	$MFTmirr 的开始簇号
0x40	4	每个 MFT 记录的簇数
0x44	4	每索引的簇数
0x48	8	分区的逻辑序列号
0x50	4	校验和,一般都为 0

④ 0x54～0x1FD:引导程序。负责将系统文件 NTLDR 装入,对于没有安装系统的分区是无效的。

⑤ 0x1FE～0x1FF:结束标志。为“55 AA”。

2. MFT

MFT(master file table,主文件表)对于 NTFS 文件系统来说尤为重要,在 NTFS 文件系统中,磁盘上的所有数据都是以文件的形式存储,其中包括元文件。每个文件都有一个或多个文件记录,每个文件记录占用两个扇区。MFT 的前 16 个文件记录总是元文件的,并且顺序是固定不变的,如表 3-13 所示。第一个 MFT(MFT 0 或 $MFT)用于描述 MFT 本身,记录了 MFT 的大小和位置。第二个 MFT(MFT 1 或 $MFTMirr)是 MFT 中第一个表项的备份。后续保存的是每一个文件和每一个目录所对应的 MFT 项。

表 3-13 NTFS 文件系统元文件

序 号	元 文 件	描 述
0	$MFT	主文件表
1	$MFTMirr	主文件表前几项的备份
2	$LogFile	日志文件,记录元数据变化
3	$Volume	卷文件,包含卷标及版本信息等
4	$AttrDef	属性定义列表,定义每种属性的名字和类型
5	$Root	根目录文件
6	$Bitmap	位图文件,每一个二进制位对应一个簇的状态,1 表示该簇已分配,0 表示该簇未分配
7	$Boot	引导文件,DBR 扇区是引导文件的第一个扇区
8	$BadClus	坏簇记录文件,防止文件系统再次分配这些簇
9	$Secure	文件的安全属性和访问控制(仅用于 Windows 2000 和 Windows XP)
10	$UpCase	大小写字符转换表文件
11	$Extend	扩展属性如 $Quota(磁盘配额)、$ObjId(对象 ID 文件)和 $Reparse(重解析点文件)
12～15	⋮	其他属性预留

由于 NTFS 文件系统是通过 MFT 来确定文件在磁盘上的位置以及文件的属性,所以 MFT 是非常重要的,MFT 的起始位置在 DBR 中有描述。

3. 文件记录

文件记录由三部分组成，一部分是文件记录头，然后是属性列表，最后结尾为 4 字节的“FF”，文件记录的结构如图 3-106 所示。

图 3-106　文件记录结构

在同一个操作系统中，文件记录头的长度和偏移位置的数据含义是基本不变的，属性列表会随着数据的不同而不同，不同的属性有着不同的含义。如图 3-107 所示，偏移量 0x00～0x37 是一个文件记录头，文件记录头各字节的具体含义如表 3-14 所示。

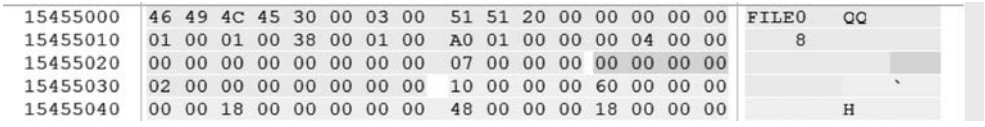


图 3-107　文件记录头

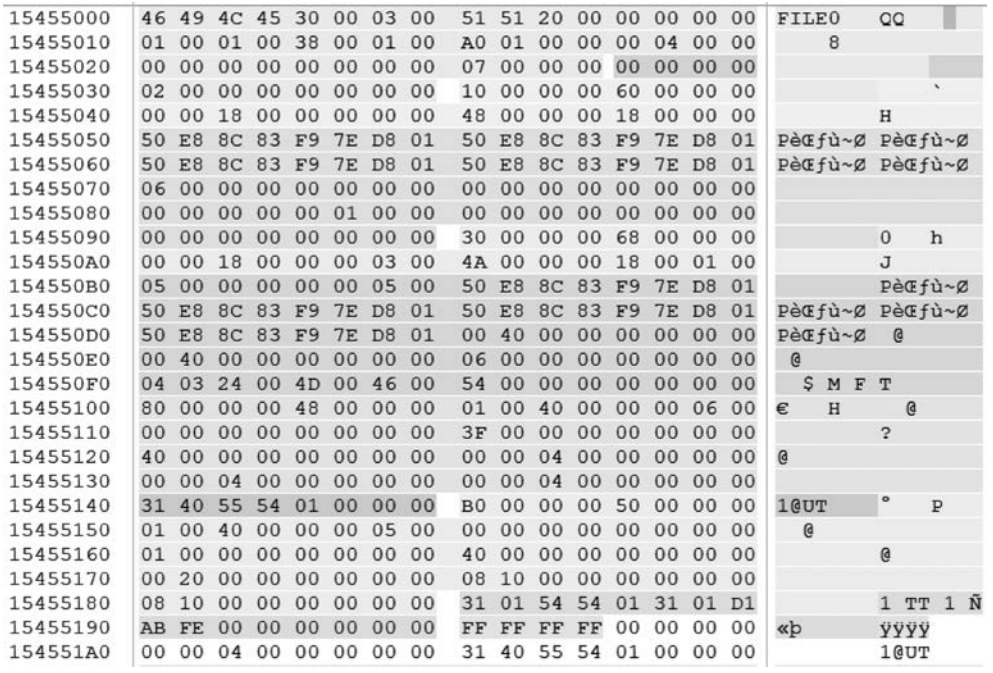
表 3-14　文件记录头各字节含义

偏　移　量	字　节　数	含　　义
0x00	4	固定值，总为“FILE”
0x04	2	更新序列号的偏移
0x06	2	更新序列号与更新数组大小(以字为单位)
0x08	8	日志文件序列号(每次记录修改，该序列号加 1)
0x10	2	序列号
0x12	2	硬连接数，即有多少目录指向该文件
0x14	2	第一个属性的偏移地址
0x16	2	标志字节，0x00 表示删除文件，0x01 表示正常文件，0x02 表示删除目录，0x03 表示正常目录
0x18	4	文件记录实际大小
0x1C	4	文件记录分配大小
0x20	8	基本文件记录的文件索引号
0x28	2	下一属性 ID，当增加新的属性时，将该值分配给新属性，然后该值增加，如果 MFT 记录重新使用，则将它置为 0，第一个实例总是 0
0x2A	2	边界，Windows XP 中使用，本记录使用的两个扇区的最后两个字节的值
0x2C	4	Windows XP 中使用，本文件记录号
0x30	2	更新序列号
0x32	4	更新数组

在 NTFS 文件系统中所有与文件相关的数据结构均被认为是属性，包括文件的内容。文件记录是一个与文件相对应的文件属性数据库，它记录了文件的所有属性。每个文件记录中都有多个属性，它们相对独立，有各自的类型和名称。如图 3-108 所示，在 0x38 之后的 4 大块颜色数据是 4 条属性，描述名称、时间、索引等信息，最后以“FF FF FF FF”结束。

每个属性都由两部分组成，即属性头和属性体，如图 3-109 所示。其中，属性头的前 4 字节为属性的类型。

另外，属性还有常驻与非常驻之分。当一个文件很小时，其所有属性体都可以存放在文件记录中，该属性就称为常驻属性。如果某个文件很大，1KB(2 个扇区)的文件记录无法记



15455000	46	49	4C	45	30	00	03	00	51	51	20	00	00	00	00	00	FILE0 QQ
15455010	01	00	01	00	38	00	01	00	A0	01	00	00	00	04	00	00	8
15455020	00	00	00	00	00	00	00	00	07	00	00	00	00	00	00	00	
15455030	02	00	00	00	00	00	00	00	10	00	00	00	60	00	00	00	`
15455040	00	00	18	00	00	00	00	00	48	00	00	00	18	00	00	00	H
15455050	50	E8	8C	83	F9	7E	D8	01	50	E8	8C	83	F9	7E	D8	01	PèŒfù~Ø PèŒfù~Ø
15455060	50	E8	8C	83	F9	7E	D8	01	50	E8	8C	83	F9	7E	D8	01	PèŒfù~Ø PèŒfù~Ø
15455070	06	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
15455080	00	00	00	00	00	01	00	00	00	00	00	00	00	00	00	00	
15455090	00	00	00	00	00	00	00	00	30	00	00	00	68	00	00	00	0 h
154550A0	00	00	18	00	00	00	03	00	4A	00	00	00	18	00	01	00	J
154550B0	05	00	00	00	00	00	05	00	50	E8	8C	83	F9	7E	D8	01	PèŒfù~Ø
154550C0	50	E8	8C	83	F9	7E	D8	01	50	E8	8C	83	F9	7E	D8	01	PèŒfù~Ø PèŒfù~Ø
154550D0	50	E8	8C	83	F9	7E	D8	01	00	40	00	00	00	00	00	00	PèŒfù~Ø @
154550E0	00	40	00	00	00	00	00	00	06	00	00	00	00	00	00	00	@
154550F0	04	03	24	00	4D	00	46	00	54	00	00	00	00	00	00	00	$ M F T
15455100	80	00	00	00	48	00	00	00	01	00	40	00	00	00	06	00	€ H @
15455110	00	00	00	00	00	00	00	00	3F	00	00	00	00	00	00	00	?
15455120	40	00	00	00	00	00	00	00	00	00	04	00	00	00	00	00	@
15455130	00	00	04	00	00	00	00	00	00	00	04	00	00	00	00	00	
15455140	31	40	55	54	01	00	00	00	B0	00	00	00	50	00	00	00	1@UT ° P
15455150	01	00	40	00	00	00	05	00	00	00	00	00	00	00	00	00	@
15455160	01	00	00	00	00	00	00	00	40	00	00	00	00	00	00	00	@
15455170	00	20	00	00	00	00	00	00	08	10	00	00	00	00	00	00	
15455180	08	10	00	00	00	00	00	00	31	01	54	54	01	31	01	D1	1 TT 1 Ñ
15455190	AB	FE	00	00	00	00	00	00	FF	FF	FF	FF	00	00	00	00	«þ ÿÿÿÿ
154551A0	00	00	04	00	00	00	00	00	31	40	55	54	01	00	00	00	1@UT

图 3-108 文件记录

0000003000	46	49	4C	45	30	00	03	00	DC	B3	72	C5	02	00	00	00
0000003010	01	00	01	00	38	[illegible]			A8	01	00	00	00	04	00	00
0000003020	00	00	00	00	00	[illegible]			06	00	00	00	00	00	00	00
0000003030	AB	83	00	00	00	00	00	00	10	00	00	00	60	00	00	00
0000003040	00	00	18	00	00	00	00	00	48	00	00	00	18	00	00	00
0000003050	60	0C	4E	B6	D1	78	CF	01	60	0C	4E	B6	D1	78	CF	01
0000003060	60	0C	4E	B6	D1	78	CF	01	60	0C	4E	B6	D1	78	CF	01
0000003070	06	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0000003080	00	00	00	00	00	01	00	00	00	00	00	00	00	00	00	00
0000003090	00	00	00	00	00	00	00	00	[illegible]				00	68	00	00
00000030A0	00	00	18	00	00	00	03	00	[illegible]				00	18	00	01

图 3-109 属性结构

录所有属性时，则文件系统会在 MFT 元文件之外的区域（也称数据流）存放该文件的其他文件记录属性，这些存放在非 MFT 元文件内的记录就称为非常驻属性。出现非常驻属性一般是由于 DATA 文件较大，即 80H 属性大。

属性头中包含了该属性的重要信息，如属性类型、属性大小、是否为常驻属性等。而常驻属性与非常驻属性的属性头结构略有不同。常驻属性的属性头信息如表 3-15 所示。非常驻属性的属性头信息如表 3-16 所示。

表 3-15 常驻属性的属性头各字节含义

偏 移 量	字 节 数	含 义
0x00	4	属性类型
0x04	4	整个属性的长度
0x08	1	是否为常驻属性，0x00 表示常驻
0x09	1	属性名的长度，0x00 表示无属性名
0x0A	2	属性名的开始偏移

续表

偏 移 量	字 节 数	含 义
0x0C	2	标志位(压缩、加密、稀疏)
0x0E	2	属性 ID
0x10	4	属性体的长度
0x14	2	属性体的开始偏移位置
0x16	1	索引标志
0x17	1	填充
0x18	—	属性体开始

表 3-16 非常驻属性的属性头各字节含义

偏 移 量	字 节 数	含 义
0x00	4	属性类型
0x04	4	整个属性的长度
0x08	1	是否为常驻属性,0x01 表示非常驻
0x09	1	属性名的长度,0x00 表示无属性名
0x0A	2	属性名的开始偏移
0x0C	2	标志位(压缩、加密、稀疏)
0x0E	2	属性 ID
0x10	8	属性体的起始虚拟簇号 VCN
0x18	8	属性体的结束虚拟簇号 VCN
0x20	2	Data Run 的偏移地址
0x22	2	压缩单位大小,2 的 N 次方
0x24	4	不使用
0x28	8	属性体的分配大小
0x30	8	属性体的实际大小
0x38	8	属性体的初始大小
0x40	—	Data Run 信息开始

属性的种类很多,因此各属性体的含义也不同。NTFS 文件系统中常见的文件属性如表 3-17 所示。

表 3-17 NTFS 文件系统常见属性类型

属性类型(属性偏移 0x00～0x03 数据,小端)	属 性 名 称	属 性 含 义
10H	$ STANDARD_INFORMATION	标准属性,包含文件的基本属性(如只读、系统、存档),时间属性,硬连接数等
20H	$ ATTRIBUTE_LIST	属性列表,当一个文件需要多个文件记录时,描述文件的属性列表
30H	$ FILE_NAME	文件名属性(UNICODE 编码)
40H	$ OBJECT_ID	对象 ID 属性,64 字节的标志符,其中最低 16 位对卷来说是唯一的
50H	$ SECURITY_DESCRIPTOR	安全描述符属性,文件访问控制安全属性
60H	$ VOLUME_NAME	卷名属性
70H	$ VOLUME_INFOMATION	卷信息属性

续表

属性类型(属性偏移0x00～0x03数据,小端)	属性名称	属性含义
80H	$DATA	文件的数据属性
90H	$INDEX_ROOT	索引根属性
A0H	$INDEX_ALLOCATION	索引分配,90H属性的拓展版(90H属性只能在MFT内记录文件列表,A0H属性将文件列表记录到数据区可以记录更多文件)
B0H	$BITMAP	位图属性
C0H	$REPARSE_POINT	重解析点属性
D0H	$EA_INFORMATION	拓展属性信息
E0H	$EA	拓展属性
100H	$LOGGED_UTILITY_STREAM	EFS加密属性

上表中的两个属性$STANDARD_INFORMATION和$FILE_NAME包含了文件系统的所有4个时间戳信息(创建时间、修改时间、更改时间、访问时间)。操作系统在更新时间戳信息时应该同时更新两个属性,但实际研究表明,不同操作系统的具体表现有所不同,有些只更新$STANDARD_INFORMATION,有些只更新$FILE_NAME,所以在分析NTFS文件系统的时间属性时需要格外注意。

NTFS文件系统里每个文件至少要占用一个MFT项,而MFT项的大小只有1024字节,如果一个文件有太多属性,那么这些属性就需要占用其他MFT项。在NTFS文件系统里,增加的MFT项使用$ATTRIBUTE_LIST属性进行记录。每种属性的结构不尽相同。取证中关注的几个重要属性如下:

① 10H属性被称为标准信息属性,英文标识为$STANDARD_INFORMATION。10H属性是所有文件记录所必备的属性,它包含了许多文件或文件夹的基本信息,如:文件或文件夹的创建时间、文件或文件夹的修改时间、目录硬连接数等。

② 30H属性被称为文件名属性,英文标识为$FILE_NAME。30H属性常紧跟于10H属性之后,用于描述文件名以及文件或文件夹更详细的信息,如:文件名长度、文件大小、文件名命名空间、文件名Unicode码等。

③ 80H属性被称为数据属性,英文标识为$DATA。该属性容纳着文件的数据内容。80H属性是整个文件属性中最重要的部分,可大概分为3种类型:一是只有属性头无属性体,这种情况主要存在于小型文本文件中,文件大小为0字节,即文件内容为空,因此数据属性为空,无需属性体记录数据。二是常驻属性,这种情况存在于文件内容简短的情况下,为了节省空间,不额外分配簇进行存储,直接在80H属性体内存储文件数据,最后以“FF FF FF FF”为结束标志。三是非常驻属性,这是80H属性中最复杂最重要的类型。在此种情况中,文件内容大于80H属性体最大长度,因此采用数据运行列表(data run list)的方式存储数据信息,80H属性体中记录数据运行列表。

④ 90H属性被称为索引根属性,英文标识为$INDEX_ROOT。90H属性主要存在于$MFT的文件夹记录中,一般为常驻属性。该属性是实现NTFS文件系统的B+树索引的根节点。

⑤ A0H属性被称为索引根拓展属性,英文标识为$INDEX_ALLOCATION。该属性

包含一个 B 树的子节点，是一个非常驻属性。对于小型目录，此属性不存在，所有信息将保存在 $INDEX_ROOT 结构中。这个属性的内容是一个或多个索引记录(index rocord)，每个索引节点(这里是 B 树节点)有一个记录。每个索引记录包含一个或多个索引条目(index entry)结构，这些结构与 $INDEX_ROOT 相同。

通过以上几个文件重要属性的解析，就能获知在取证中较为关注的文件信息，如文件名、文件的时间、文件的位置、文件的数据内容等。

3.8.2　实验目的与条件

1. 实验目的

通过本实验，读者重点掌握以下内容：

(1) 了解 NTFS 文件系统存储原理；

(2) 掌握 NTFS 文件系统各数据结构的解析；

(3) 掌握使用 WinHex 进行 NTFS 文件系统数据恢复的过程。

2. 实验条件

本实验所需要的软硬件清单如表 3-18 所示。

表 3-18　NTFS 文件系统数据恢复实验清单

序　号	设　备	数　量	参　数
1	取证工作站	1 台	Windows XP 以上
2	WinHex 工具	1 套	无

3.8.3　实验过程

1. 常驻文件的删除恢复

步骤 1：创建虚拟磁盘 VHD，并在该磁盘上建立 NTFS 分区。具体步骤见 3.7.3 节。

步骤 2：在新建的 VHD(E 盘)中创建一个 txt 文件，写入较少的内容并保存，如图 3-110 所示。

图 3-110　虚拟磁盘中新建小文件

步骤 3：按下“Shift＋Delete”，将“test. txt”文件删除。

步骤 4：用 WinHex 工具打开 E 盘，可以看到该磁盘中的各文件情况，如图 3-111 所示。

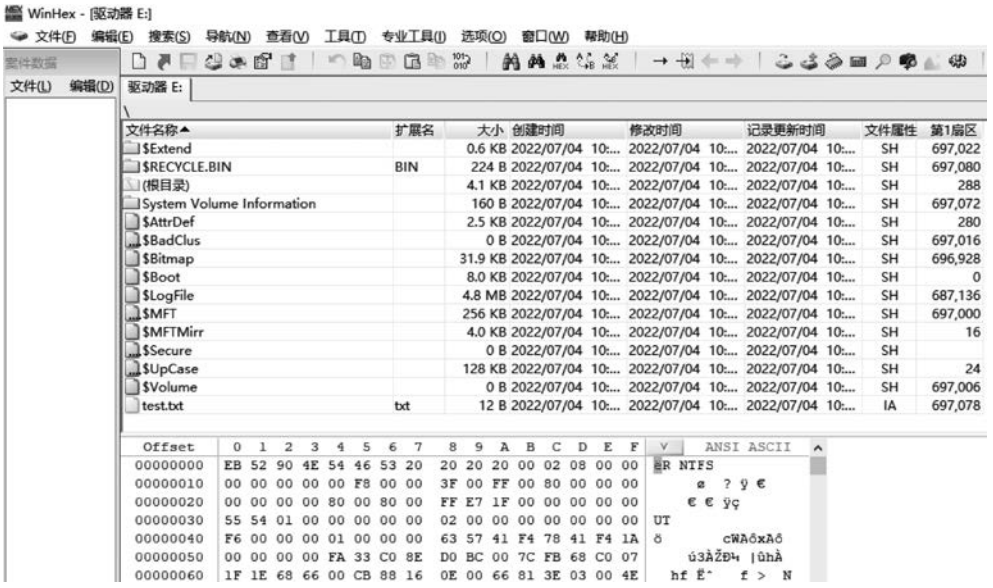


图 3-111　WinHex 中磁盘情况

步骤 5：要找到删除文件的内容，按照 NTFS 文件系统原理，必须先找到其文件记录。在该文件的文件记录中，会记录文件的名称、时间、数据等属性。现已知文件名为“test. txt”，由于 WinHex 以十六进制为核心，因此先将文件名转为十六进制 ASCII 编码（可用网页在线工具转换），如图 3-112 所示，为“00740065007300740002E007400780074”。

图 3-112　文件名 ASCII 编码

步骤 6：在 $MFT 中，向下搜索十六进制数据“00740065007300740002E007400780074”，如图 3-113 所示，单击“确定”按钮，即跳转到 $MFT 中文件“test. txt”所属文件记录，如图 3-114 所示。

步骤 7：按照 3.8.1 节理论知识，分析该文件记录，找到 80H 属性，即为文件的数据属性。在 80H 的属性头中 0x08 位置即为是否为常驻属性的标志位，可以看出该文件的 80H 属性为常驻属性，意味着该文件的数据内容直接记录在了 80H 属性体中。

步骤 8：在该 80H 属性的属性头中，0x10～0x13 位置为属性体的大小（00 00 00 0C），

图 3-113　在 $MFT 中搜索十六进制数据

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	ANSI ASCII
1545EC00	46	49	4C	45	30	00	03	00	10	7E	20	00	00	00	00	00	FILE0 ~
1545EC10	02	00	01	00	38	00	00	00	60	01	00	00	00	04	00	00	8 `
1545EC20	00	00	00	00	00	00	00	00	05	00	00	00	27	00	00	00	'
1545EC30	04	00	00	00	00	00	00	00	10	00	00	00	60	00	00	00	`
1545EC40	00	00	00	00	00	00	00	00	48	00	00	00	18	00	00	00	H
1545EC50	16	33	A8	4A	51	8F	D8	01	7D	9C	22	58	51	8F	D8	01	3¨JQ Ø }œ"XQ Ø
1545EC60	7D	9C	22	58	51	8F	D8	01	7D	9C	22	58	51	8F	D8	01	}œ"XQ Ø }œ"XQ Ø
1545EC70	20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
1545EC80	00	00	00	00	08	01	00	00	00	00	00	00	00	00	00	00	
1545EC90	E8	08	00	00	00	00	00	00	30	00	00	00	70	00	00	00	è 0 p
1545ECA0	00	00	00	00	00	00	03	00	52	00	00	00	18	00	01	00	R
1545ECB0	05	00	00	00	00	00	05	00	16	33	A8	4A	51	8F	D8	01	3¨JQ Ø
1545ECC0	16	33	A8	4A	51	8F	D8	01	16	33	A8	4A	51	8F	D8	01	3¨JQ Ø 3¨JQ Ø
1545ECD0	16	33	A8	4A	51	8F	D8	01	00	00	00	00	00	00	00	00	3¨JQ Ø
1545ECE0	00	00	00	00	00	00	00	00	20	00	00	00	00	00	00	00	
1545ECF0	08	00	74	00	65	00	73	00	74	00	2E	00	74	00	78	00	t e s t . t x
1545ED00	74	00	00	00	00	00	00	00	40	00	00	00	28	00	00	00	t @ (
1545ED10	00	00	00	00	00	00	04	00	10	00	00	00	18	00	00	00	
1545ED20	83	C0	09	24	42	FB	EC	11	97	5A	C8	94	02	E2	28	D8	ƒÀ $Bûì –ZÈ" â(Ø
1545ED30	80	00	00	00	28	00	00	00	00	00	18	00	00	00	01	00	€ (
1545ED40	0C	00	00	00	18	00	00	00	48	65	6C	6C	6F	20	77	6F	Hello wo
1545ED50	72	6C	64	21	00	00	00	00	FF	FF	FF	FF	82	79	47	11	rld! ÿÿÿÿ‚yG

图 3-114　test. txt 的文件记录

0x18 位置即为常驻属性属性体的开始，即属性体中内容为“48 65 6C 6C 6F 20 77 6F 72 6C 64 21”，如图 3-115 所示。

步骤 9：按照 ASCII 编码规则转换十六进制数据“48 65 6C 6C 6F 20 77 6F 72 6C 64 21”为“Hello World!”，即为删除文件“test. txt”的内容。

至此完成了常驻文件的删除恢复！

2. 非常驻文件的删除恢复

步骤 1：在新建的 VHD(NTFS 文件系统)中存入图片文件“lena. jpg”。

步骤 2：按下“Shift＋Delete”，将“lena. jpg”文件删除，如图 3-116 所示。

步骤 3：用 WinHex 工具打开 E 盘，查看磁盘基本信息。

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	UTF-8
1545EC00	46	49	4C	45	30	00	03	00	10	7E	20	00	00	00	00	00	FILE0□□□□~ □□□□□
1545EC10	02	00	01	00	38	00	00	00	60	01	00	00	00	04	00	00	□□□□8□□□`□□□□□□□
1545EC20	00	00	00	00	00	00	00	00	05	00	00	00	27	00	00	00	□□□□□□□□□□□□'□□□
1545EC30	04	00	00	00	00	00	00	00	10	00	00	00	60	00	00	00	□□□□□□□□□□□□`□□□
1545EC40	00	00	00	00	00	00	00	00	48	00	00	00	18	00	00	00	□□□□□□□□H□□□□□□□
1545EC50	16	33	A8	4A	51	8F	D8	01	7D	9C	22	58	51	8F	D8	01	□3□ □ "XQ□
1545EC60	7D	9C	22	58	51	8F	D8	01	7D	9C	22	58	51	8F	D8	01	□ □ □ □
1545EC70	20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
1545EC80	00	00	00	00	08	01	00	00	00	00	00	00	00	00	00	00	□
1545EC90	E8	08	00	00	00	00	00	00	30	00	00	00	70	00	00	00	□ □□□□0□□□p□□□
1545ECA0	00	00	00	00	00	00	03	00	52	00	00	00	18	00	01	00	□□□□□□□□R□□□□□□□
1545ECB0	05	00	00	00	00	00	05	00	16	33	A8	4A	51	8F	D8	01	□□□□□□□□□3□ □
1545ECC0	16	33	A8	4A	51	8F	D8	01	16	33	A8	4A	51	8F	D8	01	□ □ □ □
1545ECD0	16	33	A8	4A	51	8F	D8	01	00	00	00	00	00	00	00	00	□ □ □□□□□□
1545ECE0	00	00	00	00	00	00	00	00	20	00	00	00	00	00	00	00	□□□□□□□□ □□□□□□□
1545ECF0	08	00	74	00	65	00	73	00	74	00	2E	00	74	00	78	00	□□t□e□s□t□.□t□x□
1545ED00	74	00	00	00	00	00	00	00	40	00	00	00	28	00	00	00	t□□□□□□□@□□□(□□□
1545ED10	00	00	00	00	00	00	04	00	10	00	00	00	18	00	00	00	□ □ □
1545ED20	83	C0	09	24	42	FB	EC	11	97	5A	C8	94	02	E2	28	D8	□ B□ ZÙ □□
1545ED30	80	00	00	00	28	00	00	00	00	00	18	00	00	00	01	00	□□□(□□□□□□□□□□
1545ED40	0C	00	00	00	18	00	00	00	48	65	6C	6C	6F	20	77	6F	□□□□□□□□Hello wo
1545ED50	72	6C	64	21	00	00	00	00	FF	FF	FF	FF	82	79	47	11	rld!□□□□□ □

图 3-115 80H 属性解析

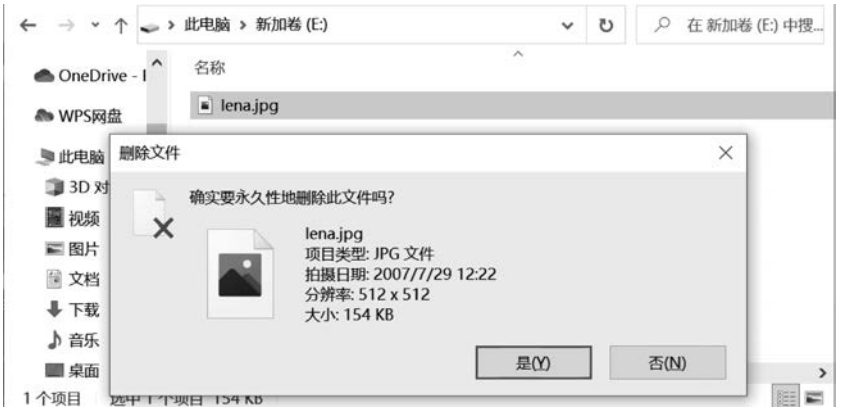


图 3-116 删除图片文件“lena.jpg”

步骤 4：使用编码转换工具将所要恢复文件的文件名“lena.jpg”转换为十六进制 ASCII 编码为“006c0065006e0061002e006a00700067”。

步骤 5：单击 $MFT，跳转到 $MFT 起始扇区，单击工具栏中的“查找十六进制数值”按钮，在弹出的对话框中，向下搜索十六进制数据“006c0065006e0061002e006a00700067”，具体如图 3-117 所示。

步骤 6：单击“确定”按钮，成功跳转到 $MFT 中“lena.jpg”的文件记录，如图 3-118 所示。

步骤 7：解析该文件记录，找到 80H 属性（数据属性），其中 0x08 位置为“01”，即该条属性为非常驻属性。也即是说，文件的数据属性较大，在文件记录中存储不下，因此，文件的数据内容存放于簇流之中。此时，80H 属性的属性体中存放的是指向文件簇流的索引（Data Runs）。

步骤 8：非常驻属性的属性体开始位置为 0x40，即 80H 属性的属性体中十六进制数据

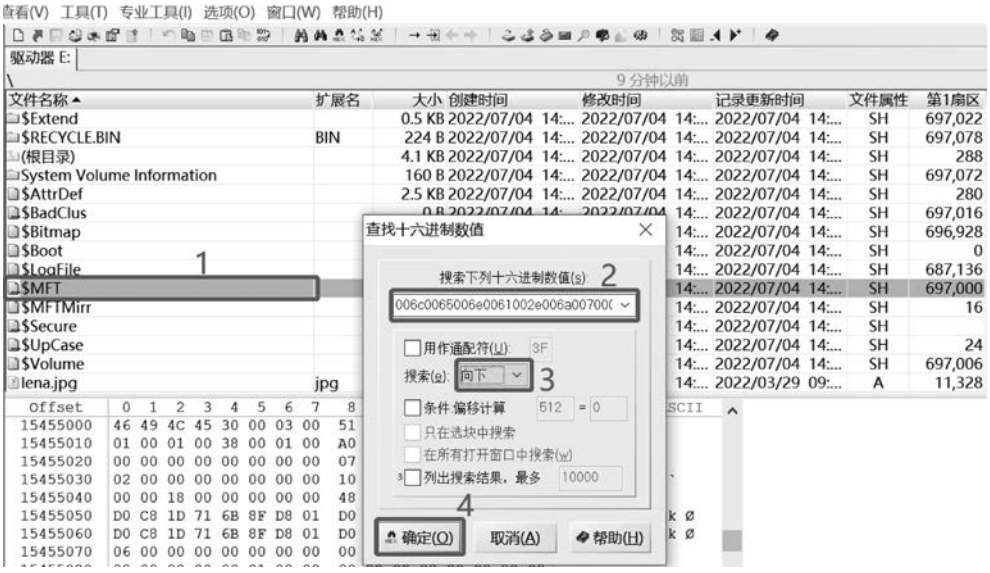


图 3-117　在 $MFT 中搜索十六进制数据

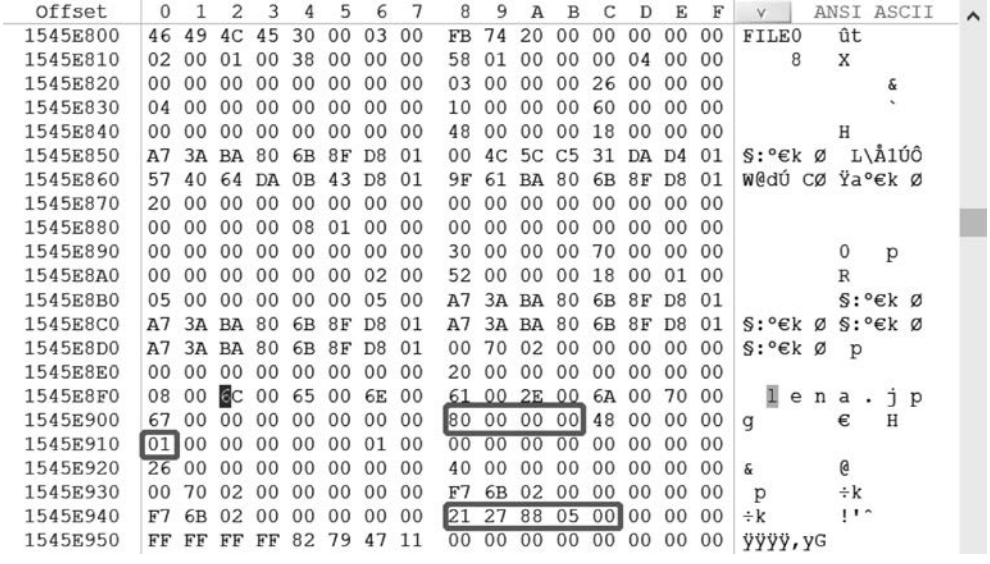


图 3-118　“lena.jpg”文件记录

为“21 27 88 05 00”，按照 Data Runs 的解析规则，该删除文件的数据内容位于从 0x588 开始的 0x27 个簇中。

步骤 9：单击工具栏中的“跳至扇区”按钮，输入簇号“1416(0x588)”，单击“确定”按钮，如图 3-119 所示。即跳转至删除文件内容起始簇。

步骤 10：在 1416 号簇的首字节处右击，在弹出的快捷菜单中选择“选块起始位置”选项，如图 3-120 所示。

步骤 11：由于该删除文件的数据内容位于从 0x588(1416)开始的 0x27(39)个簇中。因此，文件的最后一簇为 1416＋39－1＝1454 号簇。同上，跳转到 1454 号簇，如图 3-121 所示。

图 3-119 跳转至删除文件内容起始簇

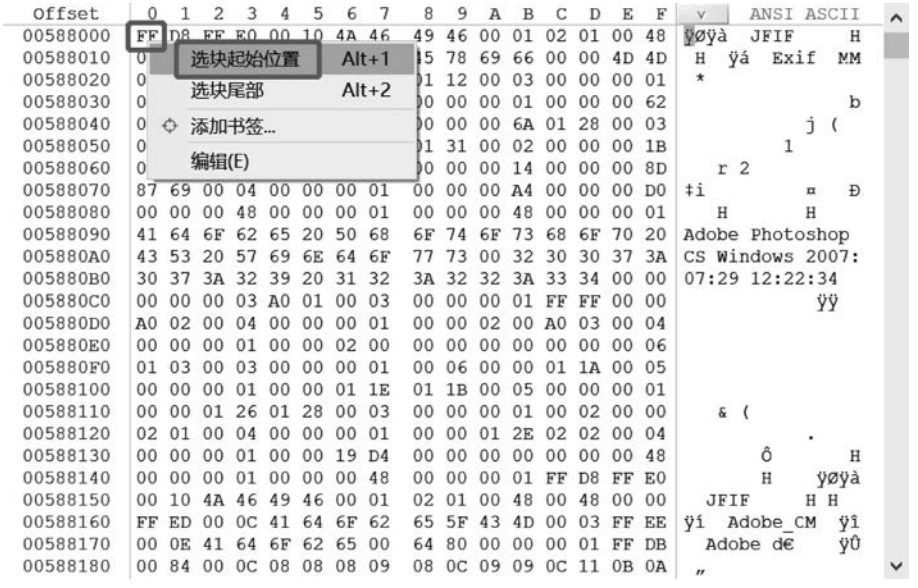


图 3-120 删除文件存储的起始簇

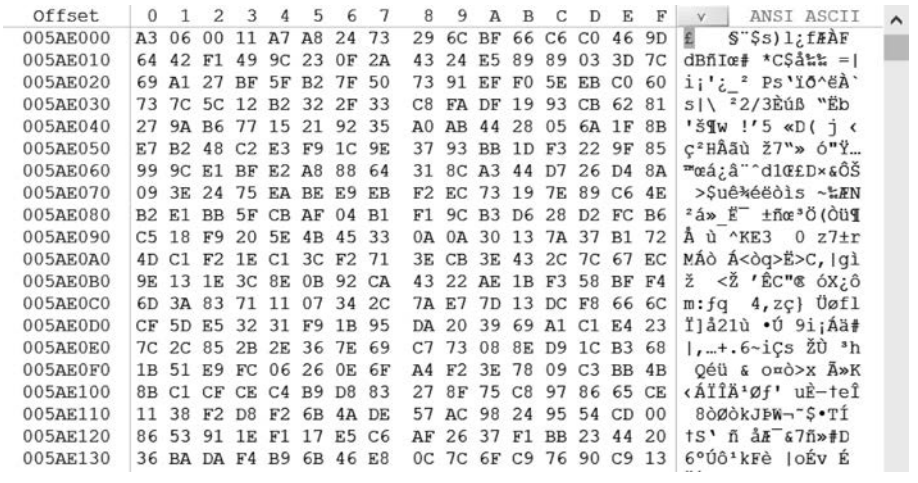


图 3-121 删除文件存储的最后一簇

步骤 12：由分区的 DBR 解析可知，该 E 分区中簇的大小为每簇 8 扇区，在该最后一簇中向下拉，直至找到文件的最后一字节（位于该簇的第 6 个扇区中），在最后一字节上右击，在弹出的快捷菜单中选择“选块尾部”选项，如图 3-122 所示。

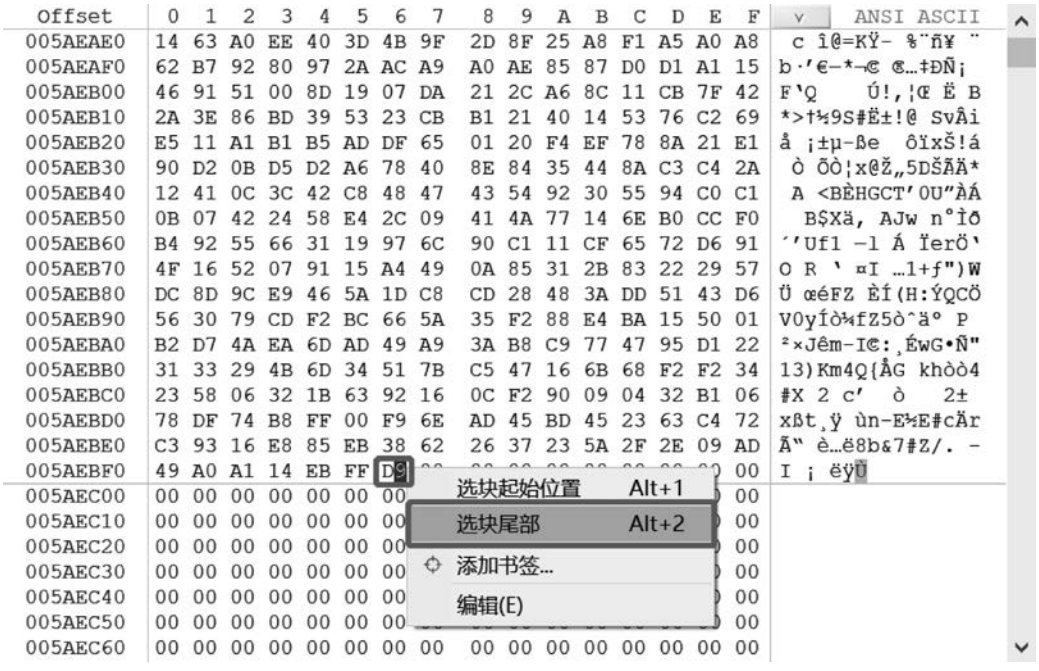


图 3-122　删除文件的最后一字节

步骤 13：由此，该删除文件的所有数据字节均被选中，在选中块上右击，在弹出的快捷菜单中选择“编辑”→“复制选块”→“至新文件”选项，如图 3-123 所示，保存在桌面上，命名为“huifu.jpg”，如图 3-124 所示。

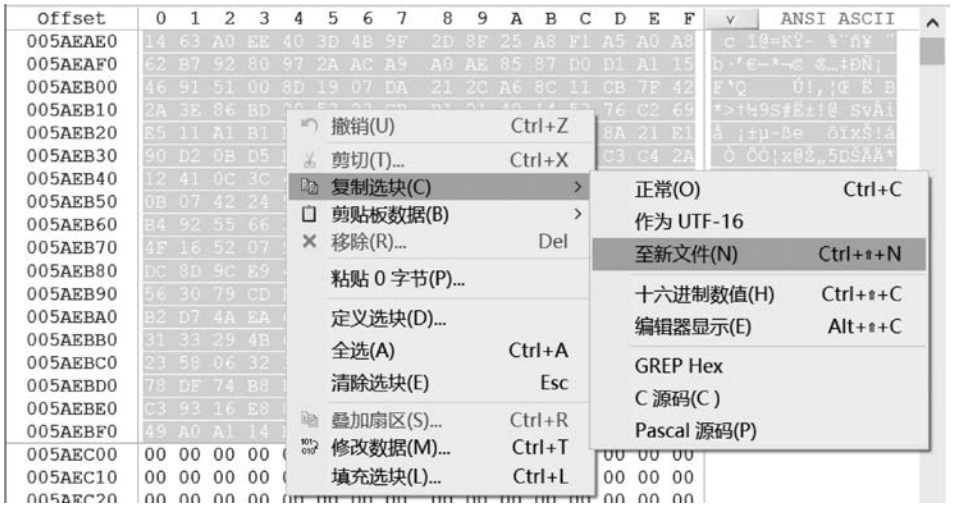


图 3-123　复制文件数据块至新文件

图 3-124　恢复删除图片文件至桌面

至此完成非常驻文件的删除恢复!

3.8.4　实验小结

NTFS 文件系统与 FAT 文件系统差别很大,在 NTFS 中所有数据均以文件的形式存在,MFT 文件记录尤为重要,通过 MFT 可以定位每一个文件。

本实验意在让读者掌握 NTFS 文件系统的基本原理,如数据结构、索引存储等,实验环境较为理想。在实际应用中,不连续存储的文件删除后,对于某些类型的文件,比如视频文件,如果根据数据结构上下文搜索匹配进行恢复,运算量较大。