4 文件系统管理

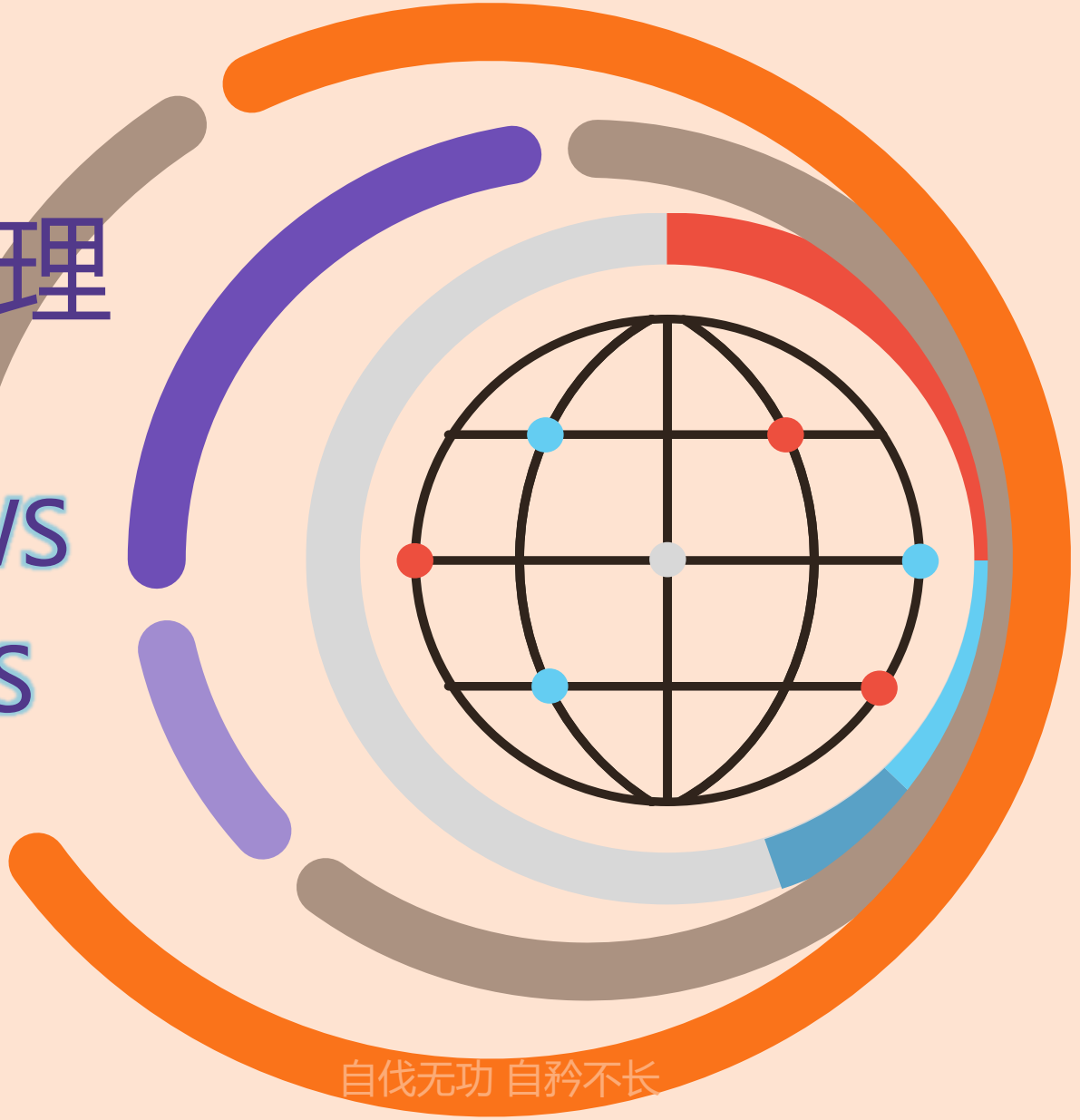
PRINCIPLE OF WINDOWS AND ITS APPLICATIONS

School of CS

Jicheng Hu

jicheng @ yahoo . com

https://gitee.com/wuhanuniversity/



自伐无功 自矜不长

outlines

File Systems in Windows

- File systems in Windows are implemented as file system drivers working above the storage system.
- Every system-supplied file system in Windows is designed to provide reliable data storage with varying features to meet the user's requirements.
- Standard file systems available in Windows include NTFS, ReFS, ExFAT, UDF, and FAT32.

文件系统是操作系统用于明确存储设备（常见的是磁盘，也有基于NAND Flash的固态硬盘）或分区上的文件的方法和数据结构；即在存储设备上组织文件的方法。

文件系统由三部分组成

- 文件系统的接口
- 对对象操纵和管理的软件集合
- 对象及属性

从系统角度来看，文件系统是对文件存储设备的空间进行组织和分配，负责文件存储并对存入的文件进行保护和检索的系统。具体地说，它负责为用户建立文件，存入、读出、修改、转储文件，控制文件的存取，当用户不再使用时撤销文件等。

Windows支持的文件系统

- FAT
- NTFS、 ReFS
- CDFS
- UDF

File system filter drivers

- A file system filter driver intercepts requests targeted at a file system or another file system filter driver.

- By intercepting the request before it reaches its intended target, the filter driver can extend or replace functionality provided by the original target of the request. Examples of filter drivers include:
  - Anti-virus filters
  - Backup agents
  - Encryption products

https://github.com/Microsoft/Windows-driver-samples

Windows Azure Active Directory

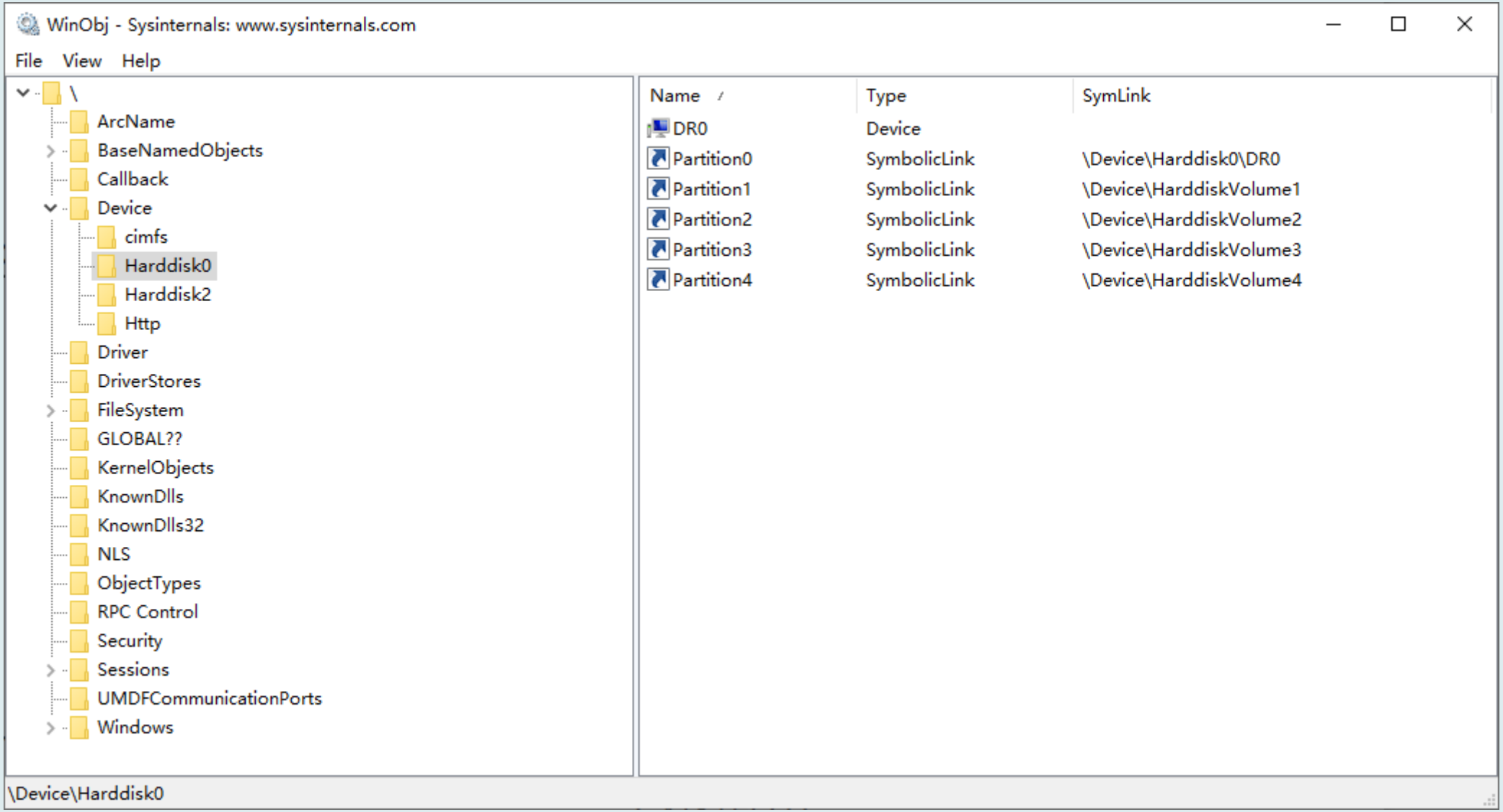
- Windows Azure Active Directory提供了云端的身份和访问管理
- 本质上Windows Azure Active Directory让用户通过认证来使用一些服务
  - 例如Exchange Online邮箱
- Windows Azure Active Directory 有免费、基础和高级版本

https://azure.microsoft.com/zh-cn/services/active-directory/

Azure Active Directory功能

- 简化单一登录，Azure AD 支持超过 2,800 个预先集成的软件即服务 (SaaS) 应用程序
- 通过单一登录，使用户可以在任何平台上从任何位置无缝访问应用，自动化用户生命周期和预配工作流，借助自助服务管理，节省时间和资源，了解单一登录的详细信息
- 实施强身份验证和条件访问策略来保护用户凭据通，过确保正确的人员有权访问所需的资源，有效地管理标识
- 通过一个标识提供者为外部用户获取灵活、可缩放的标识和访问管理，自定义用户旅程并简化访问应用程序的身份验证过程

https://azure.microsoft.com/zh-cn/services/active-directory/


WinObj - Sysinternals: www.sysinternals.com
File View Help
\
ArcName
BaseNamedObjects
Callback
Device
cimfs
Harddisk0
Harddisk2
Http
Driver
DriverStores
FileSystem
GLOBAL??
KernelObjects
KnownDlls
KnownDlls32
NLS
ObjectTypes
RPC Control
Security
Sessions
UMDFCommunicationPorts
Windows
Name
Type
SymLink
DR0
Device
Partition0
SymbolicLink
\Device\Harddisk0\DR0
Partition1
SymbolicLink
\Device\HarddiskVolume1
Partition2
SymbolicLink
\Device\HarddiskVolume2
Partition3
SymbolicLink
\Device\HarddiskVolume3
Partition4
SymbolicLink
\Device\HarddiskVolume4
\Device\Harddisk0

FAT 文件系统简介

- FAT16
  - DOS、Windows 95使用的文件系统
  - 最大可以管理4GB的分区
  - 每个分区最多只能有65525个簇
- FAT32
  - 支持2TB（2048G）的分区
  - 使用的簇比FAT16小

FAT 文件系统的优点

- 文件系统所占容量与计算机的开销少
- 支持各种操作系统 —— 可移植
- 方便的用于传送数据

FAT 文件系统的缺点

- 容易受损害
  - FAT文件系统损坏时，计算机就要瘫痪或者不正常关机
- 单用户
  - 不保存文件的权限信息；只包含隐藏、只读等公共属性
- 非最佳更新策略
  - 在磁盘的第一个扇区保存其目录信息
- 没有防止碎片的最佳措施
- 文件名长度受限
  - 8.3模式

New Technology File System

- 日志类的文件系统，使用NTFS日志记录数据
- 文件夹或者目录最多可以使用 255 个字符
- 可以管理最大256TB的单个文件大小
- 支持文件的安全、存储和容错功能
- 设计目标是在大容量的硬盘上能够很快地执行读、写和搜索等标准的文件操作，包括文件系统恢复等高级操作
- 支持对于关键数据、重要的数据访问控制和私有权限
- 可以为单个文件设定权限

NTFS 优点

- 更为安全的文件保障，提供文件加密，能够大大提高信息的安全性
- 更好的磁盘压缩功能
- 支持最大达2TB的大硬盘，并且随着磁盘容量的增大，NTFS的性能不像FAT那样随之降低
- 可以赋予单个文件和文件夹权限：对同一个文件或者文件夹为不同用户可以指定不同的权限；可以为单个用户设置权限
- 恢复能力：用户在NTFS卷中很少需要运行磁盘修复程序。在系统崩溃事件中，NTFS文件系统使用日志文件和复查点信息自动恢复文件系统的一致性

NTFS 优点

- NTFS文件夹的B-Tree结构使得用户在访问较大文件夹中的文件时，速度甚至较访问卷中较小文件夹中的文件还快
- 可以在NTFS卷中压缩单个文件和文件夹。且用户不需要使用解压软件将这些文件展开，而直接读写压缩文件
- 支持活动目录和域：可以帮助用户方便灵活地查看和控制网络资源
- 支持稀疏文件：应用程序生成的一种特殊文件，它的文件尺寸非常大，但实际上只需要很少的磁盘空间；NTFS只需要给这种文件实际写入的数据分配磁盘存储空间
- 支持磁盘配额：可以管理和控制每个用户所能使用的最大磁盘空间

NTFS的安全特性

- 许可权 —— 定义用户或组可以访问哪些文件或目录，并为不同的用户提供不同的访问等级
- 审计 —— 可将与NTFS安全有关的事件记录到安全记录中，可利用“事件查看器”进行查看
- 拥有权 —— 记住文件的所属关系，创建文件或目录的用户拥有对它的全部权限；管理员或个别具有相应许可的人可以接受文件或目录的拥有权
- 可靠的文件清除 —— NTFS会回收未分配的磁盘扇区中的数据，对这种扇区的访问将返回0值

NTFS的安全特性

- 上次访问时间标记
- 自动缓写功能 —— 基于记录的文件系统，记录文件和目录的变化，记录在系统失效情况下如何取消（undo）和重作（redo）这些变更
- 热修复功能 —— 当扇区发生写故障时，NTFS会自动进行检测，把有故障的簇加上不能使用标记，并写入新簇；
- 磁盘镜像功能
- 有校验的磁盘条带化
- 文件加密

- CDFS（CD-ROM file system）
  - CD-ROM文件系统
  - 只读文件系统驱动
  - 最大尺寸4GB
  - 最多65535个目录

- UDF（Universal Disk Format）
  - 主要是用于存储DVD-ROM文件系统

4.5 支持文件系统的磁盘结构

- 扇区和簇
- 分区和卷
- 分区引导扇区
- BIOS参数块
- 文件分配表FAT
- 主文件表MFT
- 目录
- 附加的索引

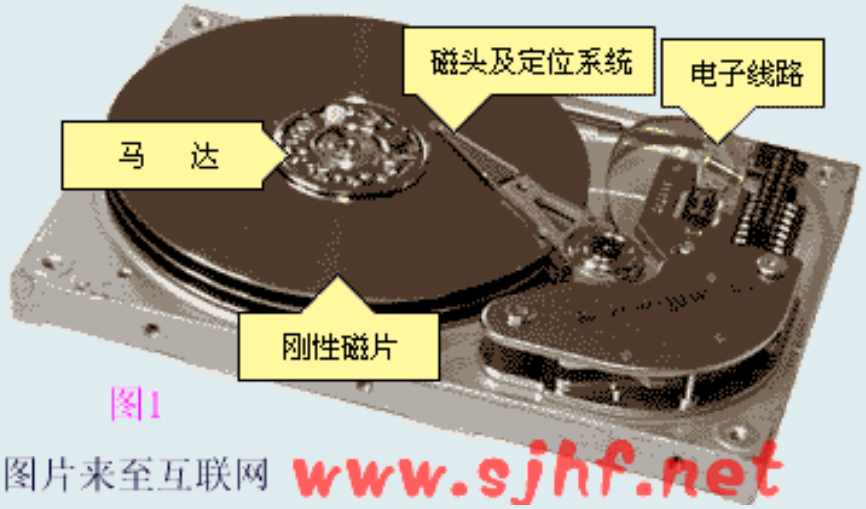
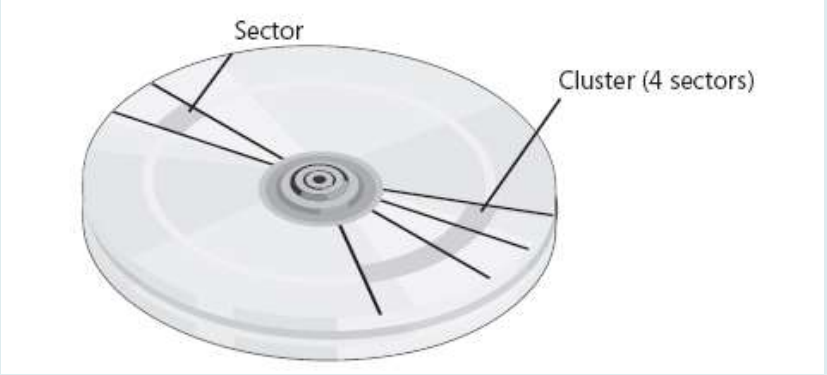


图1

图片来至互联网 www.sjhf.net

扇区Sector和簇Cluster

- 每个扇区512字节
- 若干扇区聚合在一起组成的分配单元构成簇
- FAT: 16位寻址，2^{16}个簇，最大个数2^{16}×512字节＝32MB，卷最大4GB
- FAT32: 32位寻址，最多2^{28}簇，卷理论可达8T，实际最大32GB
- NTFS：64位寻址，卷理论最大值16EB，工业标准卷最大2TB



Default FAT16 Cluster Sizes in Windows

Volume Size	Cluster Size
0–32 MB	512 bytes
33 MB–64 MB	1 KB
65 MB–128 MB	2 KB
129 MB–256 MB	4 KB
257 MB–511 MB	8 KB
512 MB–1023 MB	16 KB
1024 MB–2047 MB	32 KB
2048 MB–4095 MB	64 KB

Default Cluster Sizes for FAT32 Volumes

Partition Size	Cluster Size
32 MB–8 GB	4 KB
8 GB–16 GB	8 KB
16 GB–32 GB	16 KB
32 GB	32 KB

Default Cluster Sizes for NTFS Volumes

Volume Size	Default Cluster Size
512 MB or less	512 bytes
513 MB–1024 MB (1 GB)	1 KB
1025 MB–2048 MB (2 GB)	2 KB
Greater than 2048 MB	4 KB

分区引导扇区

- 分区引导扇区：第一个扇区
- 前16个字节

  EB 3C 90 4D 53 44 4F 53 35 2E 30 00 02 04 01 00 . < .
  MSDOS5.0 。 。 。 。
- BIOS BPB
- 扩展BPB

表 3　FAT32 分区上 DBR 中各部分的位置划分

字节位移	字段长度	字段名	对应图 8 颜色
0x00	3 个字节	跳转指令	
0x03	8 个字节	厂商标志和 os 版本号	
0x0B	53 个字节	BPB	
0x40	26 个字节	扩展 BPB	
0x5A	420 个字节	引导程序代码	
0x01FE	2 个字节	有效结束标志	

FAT BPB - 1

- 每扇区字节数
- 每簇扇区数
- FAT表开始前保留的扇区数
- FAT表副本的数量
- 根目录中项目的最大数量
- 扇区数量
- 介质描述符
- 每个FAT表的扇区数

FAT BPB - 2

- 每个磁道的扇区数
- 扇区总数
- 驱动器类型
- 特殊标志
- 磁盘签名
- 卷的序列号
- 传统卷标
- 文件系统描述符

FAT32 BPB - 1

- 每扇区的字节数
- 每簇的扇区数
- 保留的扇区数
- FAT表的数量
- 根目录的最大项数
- 小扇区数
- 介质描述符
- 每个FAT表含有的扇区数（00 00）

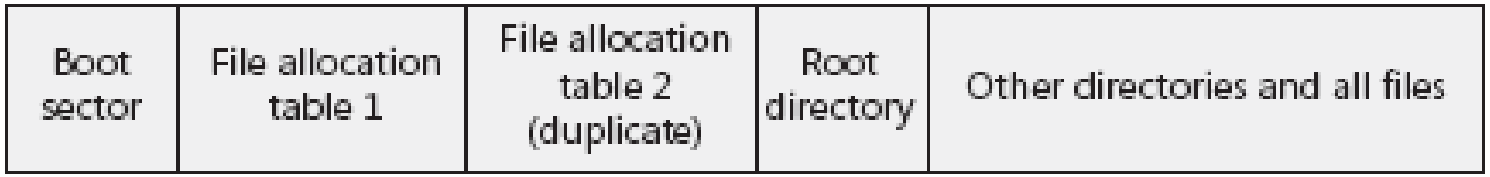
FAT32 BPB - 2

- 每个磁道的扇区数
- 隐藏的扇区数
- 扇区总数
- 每个FAT表含有的扇区数
- 标志位
- 文件系统版本号
- 根目录所在簇
- 文件系统信息扇区

FAT32 BPB - 3

- 引导扇区备份
- 保留域
- 驱动器类型
- 特殊标志
- 签名
- 卷序列号
- 卷标
- 文件系统

FAT结构


Boot sector
File allocation table 1
File allocation table 2 (duplicate)
Root directory
Other directories and all files

FAT表结构

- 文件分配表FAT
- FAT描述了卷中文件的布局和结构
- FAT16用2字节映射分区上的每个簇—16位寻址

  F8FF FFFF FFFF FFFF FFFF FFFF FFFF FFFF

  0900 FFF 0B00 FFFF 0D00 FFFF 0F00 FFFF

  1100 1200 1300 FFFF

2字节为一项，表示一个簇号

FFFF　文件的结尾

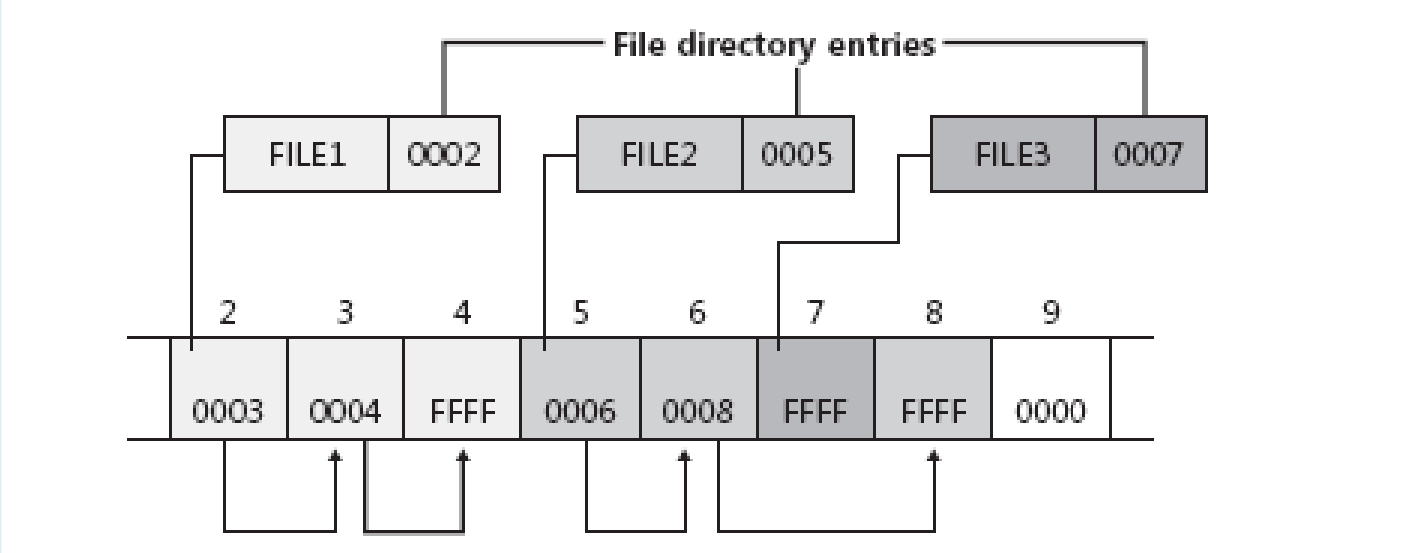
FFF8　坏簇

FFF5　保留簇

- FAT32簇映射
- 每项四个字节
- F7FFFFFF FFFFFFFF FFFFFF0F FFFFFF0F
- FFFFFF0F: 表示文件结束标记
- FAT用目录作为索引
  - 每项都代表一个文件或者子目录
  - 含有与FAT相应的簇号

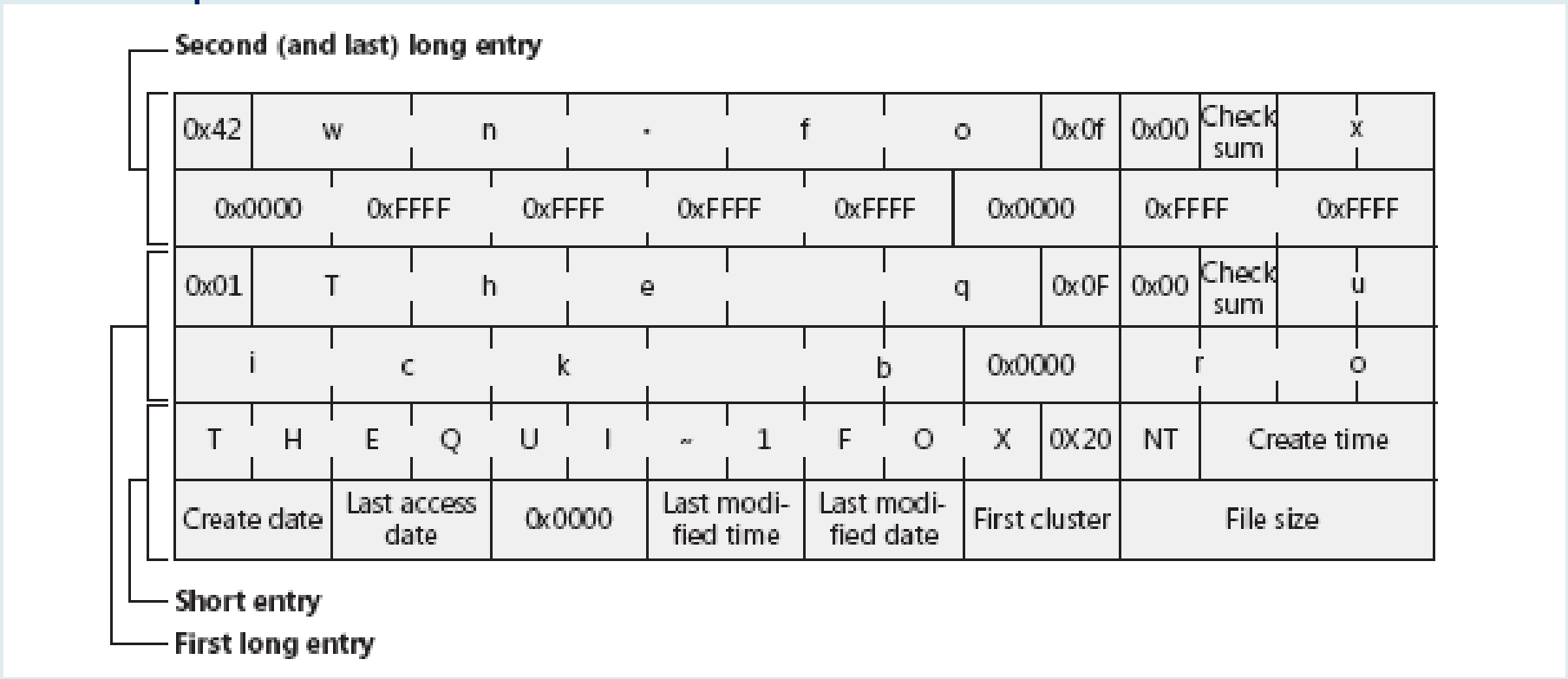
- FAT/FAT32目录列表
  - 文件名
  - 属性
  - 保留
  - 日期和时间戳记
  - 文件长度
- FAT/FAT32文件记录
  - 文件内容记录

FAT文件分配表举例

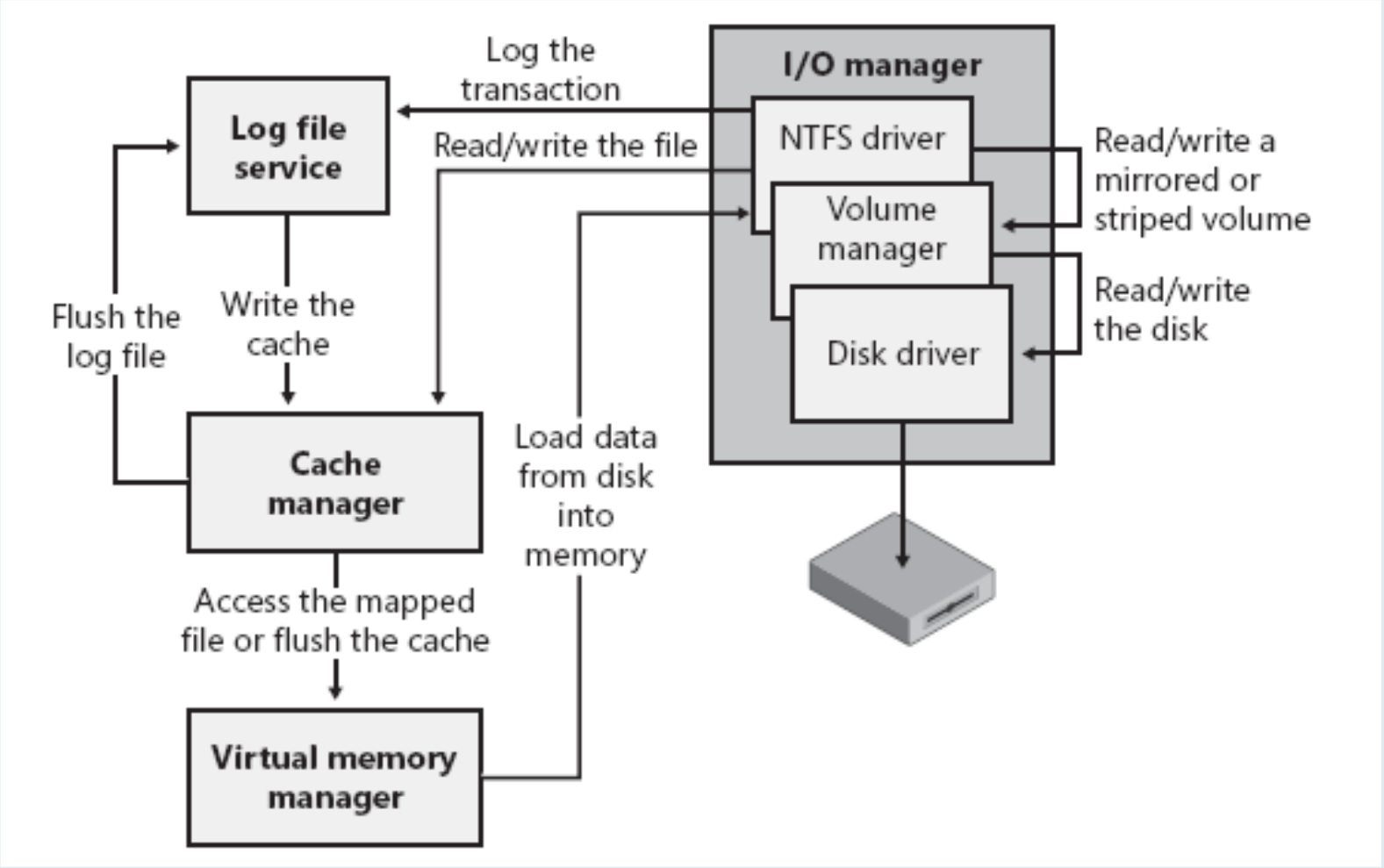

File directory entries
FILE1
0002
FILE2
0005
FILE3
0007
2
3
4
5
6
7
8
9
0003
0004
FFFF
0006
0008
FFFF
FFFF
0000

FAT目录项举例

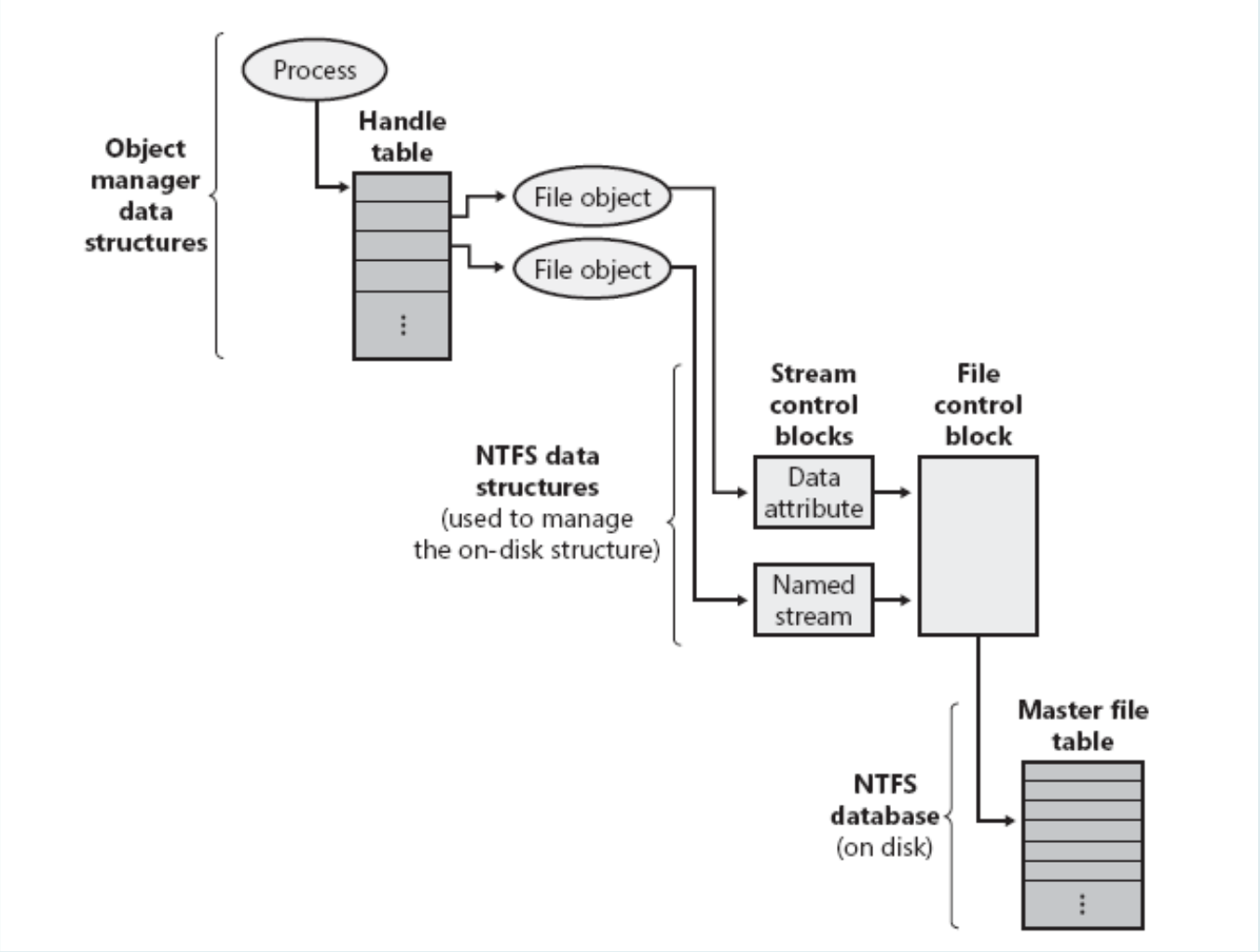
- The quick brown fox



NTFS以及相关组件


Log the transaction
I/O manager
Log file service
Read/write the file
NTFS driver
Read/write a mirrored or striped volume
Volume manager
Flush the log file
Write the cache
Read/write the disk
Disk driver
Cache manager
Load data from disk into memory
Access the mapped file or flush the cache
Virtual memory manager

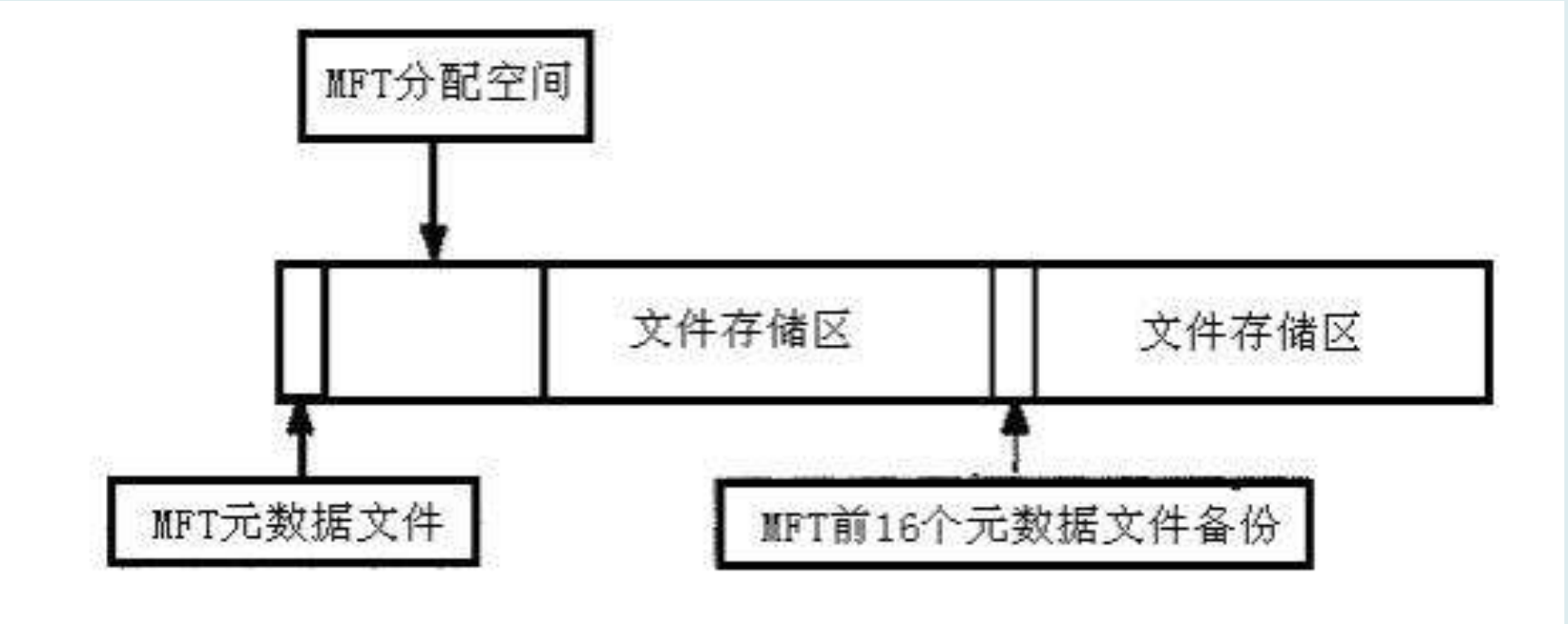
NTFS数据结构


Process
Handle table
Object manager data structures
File object
File object
Stream control blocks
File control block
NTFS data structures
(used to manage the on-disk structure)
Data attribute
Named stream
Master file table
NTFS database
(on disk)

NTFS结构

- 主文件表MFT：文件和目录都用MFT中的记录表示
- MFT是一个数据库而不是简单的簇映射
- MFT的项目比FAT表包含更多的信息，用更多的方式索引
- 分类
  - 文件记录
  - 目录记录
  - 混合记录

- MFT中的文件记录大小一般是固定的，不管簇的大小是多少，均为1KB。
- 文件记录在MFT文件记录数组中物理上是连续的，且从0开始编号，所以，NTFS是预定义文件系统。
- MFT仅供系统本身组织、架构文件系统使用，这在NTFS中称为元数据（metadata，是存储在卷上支持文件系统格式管理的数据。它不能被应用程序访问，只能为系统提供服务）。
- 其中最基本的前16个记录是操作系统使用的非常重要的元数据文件。这些元数据文件的名字都以“$”开始，所以是隐藏文件，在Windows 2000/XP中不能使用dir命令（甚至加上/ah参数）像普通文件一样列出。


MFT分配空间
文件存储区
文件存储区
MFT元数据文件
MFT前16个元数据文件备份

NTFS BPB

- 每扇区的字节数
- 每簇的扇区数
- 保留
- FAT表的数量（00）
- 根目录中的最多项数（00 00）
- 小扇区数（00 00）
- 介质描述符
- 每个FAT表含有的扇区数（00 00）
- 每磁道的扇区数
- 隐藏的扇区数

NTFS BPB

- 扇区总数（00 00 00 00）
- 每个FAT表含有的扇区数（80 00 80 00）
- 扇区总数
- 主文件表的逻辑簇编号
- MFT镜像的逻辑簇编号
- 每个MFT记录占用的簇数
- MFT索引占用的簇数
- 卷序列号
- 校验和

MFT元数据记录

- $ MFT
- $ MFTMirr
- $ LogFile
- $ Volume
- $ AttrDef
- $ \
- $ BitMap
- $ Boot
- $ BadClus
- $ Secure
- $ UpCase
- $ Extend
- $ Quota
- $ ObjID
- $ Reparse
- UsnJrn1

• NTFS元数据文件

元文件	功能
$MFT	主控文件表本身
$MFTMirr	主控文件表的部分镜像
$LogFile	日志文件
$Volume	卷文件
$AttrDef	属性定义列表文件
$Root	根目录
$Bitmap	位图文件，记录了卷种簇的分配情况
$Boot	引导文件，记录了用于系统引导的数据情况
$BadClus	卷的坏簇列表文件
$Quota（NTFS4）	在早期的NT系统中此文件为磁盘配额信息
$Secure	安全文件
$UpCase	大小写字符转换表文件
$Extend metadata directory	扩展元数据目录
$Extend\$Reparse	重解析点文件
$Extend\$UsnJrnl	加密日志文件
$Extend\$Quota	配额管理文件
$Extend\$ObjId	对象ID文件

NTFS属性

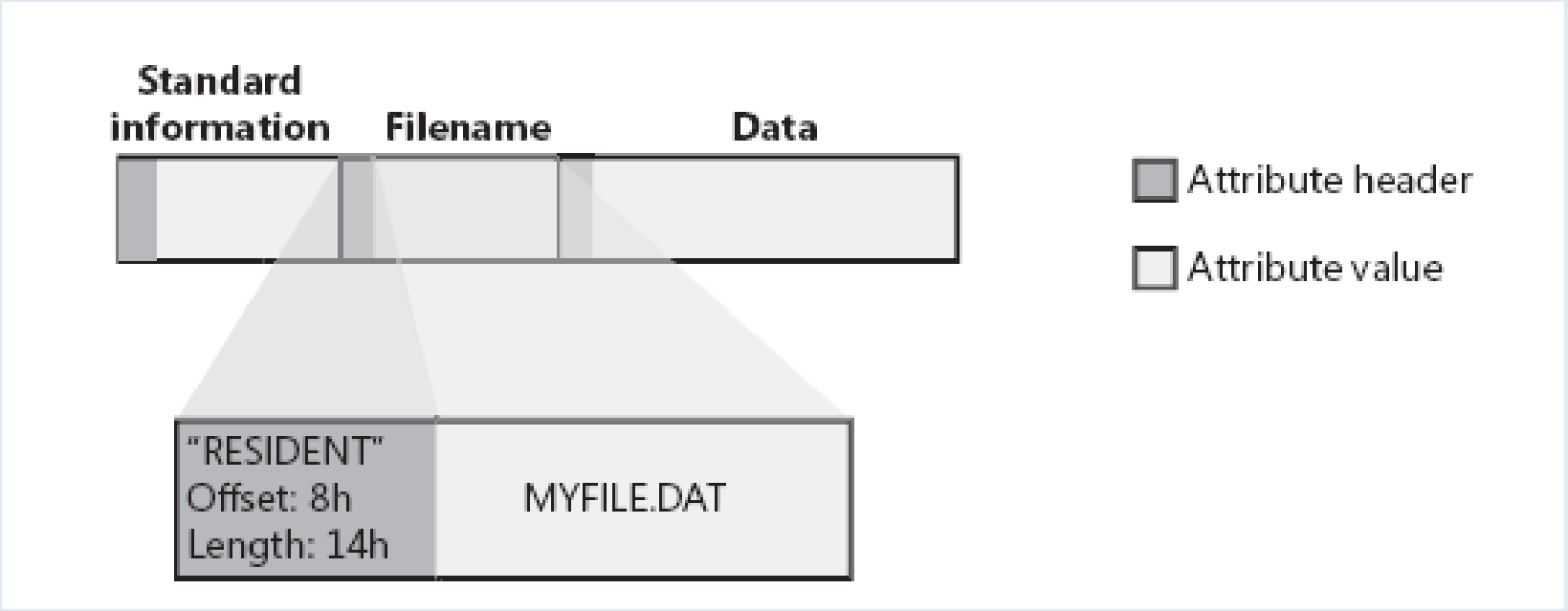
- MFT是一个面向对象的数据库
- 对象由包含特定属性的类派生
- 所有属性都分为两部分：
  - 属性头部分
    - ✓属性的字节数、属性各部分字节数、数据部分的偏移地址、时间戳记、标志位
  - 数据部分
    - ✓包含了属性设计时所要求保存的信息

属性头

- 属性的类型号（4字节）
- 属性的总字节数（4字节）
- 保留（8字节）
- 属性的数据部分的字节数（4字节）
- 属性头到属性数据部分的偏移地址（2字节）
- 特殊标志位和属性（10字节）
- 时间戳记（32字节）
- 属性本身专有的定位信息（26字节）

属性部分

- 常驻属性 $AttrDef
- 非常驻属性
  - 运行（run）：数据保存在相邻簇
  - 不连续的运行：每个运行在MFT记录中有一个指针
  - 指针：起始逻辑簇序号LCN
    起始虚拟簇序号VCN
    簇的数量


Standard information
Filename
Data
Attribute header
Attribute value
"RESIDENT"
Offset: 8h
Length: 14h
MYFILE.DAT

MFT属性

- $ Standard_Information
- $ Attribute_List
- $ File_Name
- $ Object_ID
- $ Security_Descriptor
- $ Volume_Name
- $ Volume_Information
- $ Data

MFT属性

- $ Index_Root
- $ Index_Allocation
- $ Bitmap
- $ Reparse_point
- $ Ea_Information
- $Ea
- $ Logged_Utility_Stream

属性名	属性描述
$VOLUME_INFORMATION	卷信息：仅存在于$VOLUME 元数据文件中
$VOLUME_NAME	卷名称或卷标识：仅存在于$VOLUME 元数据文件中
$STANDARD_INFORMATION	标准信息：这包括基本文件属性，如只读、存档；时间标记，如文件的创建时间和最近一次修改的时间；有多少目录指向本文件（即它的硬链接数（HARD LINK COUNT））
$FILE_NAME	文件名：这是以 UNICODE 字符表示的，由于 MS-DOS 不能正确识别 WIN32 子系统创建的文件名，当 WIN32 子系统创建一个文件名时，NTFS 会自动生成一个备用的 MS-DOS 文件名，所以一个文件可以有多种文件名属性
$SECURITY_DESCRIPTOR	安全描述符：这是为了向后兼容而保留的，主要用于保护文件以防止未授权访问，但是，WINDOWS 2000/XP 已将所有文件的安全描述符存放在$SECURE 元数据文件中，以便于共享（NTFS 的早期版本将安全描述符与文件目录一起存放，这不利于共享）
$DATA	文件数据：这是文件的内容（在 NTFS 文件系统中，一个文件除了支持文件数据即未命名的属性外，还可支持其他命名属性，即可以有多个数据属性；目录没有默认的数据属性，但是有可选的命名数据属性）
$INDEX_ROOT	索引根
$INDEX_ALLOCATION	索引分配
$BITMAP	位图
$ATTRIBUTE_LIST	属性列表：当一个文件需要使用多个 MFT 文件记录时，这用来表示该文件的属性列表
$OBJECT_ID	对象 ID:一个具有 64 个字节的标识符，其中最低的 16 个字节对卷来说是唯一的（链接跟踪服务为外壳快捷方式及 OLE 链接源文件赋予对象 ID；NTFS 提供 API 来直接通过这些对象 ID 而不是文件名来打开文件）
$REPARSE_POINT	重解析点：存储文件的重解析点数据（NTFS 的软链接与装配点都包括这个属性）
$EA	扩充属性：主要为与 OS/2 兼容，现已使用不多
$EA_INFORMATION	扩充属性信息：主要为与 OS/2 兼容，现已使用不多
$LOGGED_UTILITY_STREAM	EFS 加密属性：主要为实现 EFS（ENCRYPED FILE SYSTEM）而存储内关加密信息如解码密钥、合法访问的用户列表等。

```
File 0
Master File Table ($Mft)
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)
    $DATA (nonresident)
        logical sectors 32-52447 (0x20-0xccdf)
    $BITMAP (nonresident)
        logical sectors 16-23 (0x10-0x17)

File 1
Master File Table Mirror ($MftMirr)
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)
    $DATA (nonresident)
        logical sectors 2048728-2048735 (0x1f42d8-0x1f42df)

File 2
Log File ($LogFile)
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)
    $DATA (nonresident)
        logical sectors 2048736-2073343 (0x1f42e0-0x1fa2ff)
File 3
DASD ($Volume)
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)
    $OBJECT_ID (resident)
    $SECURITY_DESCRIPTOR (resident)
    $VOLUME_NAME (resident)
    $VOLUME_INFORMATION (resident)
    $DATA (resident)

File 4
Attribute Definition Table ($AttrDef)
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)
    $SECURITY_DESCRIPTOR (resident)
    $DATA (nonresident)
        logical sectors 512256-512263 (0x7d100-0x7d107)
```

```
File 5
Root Directory
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)
    $SECURITY_DESCRIPTOR (resident)
    $INDEX_ROOT $I30 (resident)
    $INDEX_ALLOCATION $I30 (nonresident)
        logical sectors 2073416-2073423 (0x1fa348-0x1fa34f)
    $BITMAP $I30 (resident)

File 6
Volume Bitmap ($BitMap)
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)

    $DATA (nonresident)
        logical sectors 2073424-2073675 (0x1fa350-0x1fa44b)

File 7
Boot Sectors ($Boot)
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)
    $SECURITY_DESCRIPTOR (resident)
    $DATA (nonresident)
        logical sectors 0-15 (0x0-0xf)
File 8
Bad Cluster List ($BadClus)
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)
    $DATA (resident)
    $DATA $Bad (nonresident)
```

```
File 9
Security ($Secure)
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)
    $DATA $SDS (nonresident)
        logical sectors 2073932-2074447 (0x1fa54c-0x1fa74f)
        logical sectors 523160-523163 (0x7fb98-0x7fb9b)
    $INDEX_ROOT $SDH (resident)
    $INDEX_ROOT $SII (resident)
    $INDEX_ALLOCATION $SDH (nonresident)
        logical sectors 1876152-1876159 (0x1ca0b8-0x1ca0bf)
    $INDEX_ALLOCATION $SII (nonresident)
        logical sectors 24-31 (0x18-0x1f)
    $BITMAP $SDH (resident)
    $BITMAP $SII (resident)

File 10
Upcase Table ($UpCase)
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)
    $DATA (nonresident)
        logical sectors 2073676-2073931 (0x1fa44c-0x1fa54b)

File 11
Extend Table ($Extend)
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)
    $INDEX_ROOT $I30 (resident)

File 12
(unknown/unnamed)
    $STANDARD_INFORMATION (resident)
    $SECURITY_DESCRIPTOR (resident)
    $DATA (resident)
```

```
File 13
(unknown/unnamed)
    $STANDARD_INFORMATION (resident)
    $SECURITY_DESCRIPTOR (resident)
    $DATA (resident)

File 14
(unknown/unnamed)
    $STANDARD_INFORMATION (resident)
    $SECURITY_DESCRIPTOR (resident)
    $DATA (resident)

File 15
(unknown/unnamed)
    $STANDARD_INFORMATION (resident)
    $SECURITY_DESCRIPTOR (resident)
    $DATA (resident)

File 24
\$Extend\$Quota
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)
    $INDEX_ROOT $O (resident)
    $INDEX_ROOT $Q (resident)

File 25
\$Extend\$ObjId
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)
    $INDEX_ROOT $O (resident)

File 26 \$Extend\$Reparse
    $STANDARD_INFORMATION (resident)
    $FILE_NAME (resident)
    $INDEX_ROOT $R (resident)
```

通用属性类型

- $ Standard_Information
- $ File_Name
- $ Security _Descriptor

文件记录和$Data属性

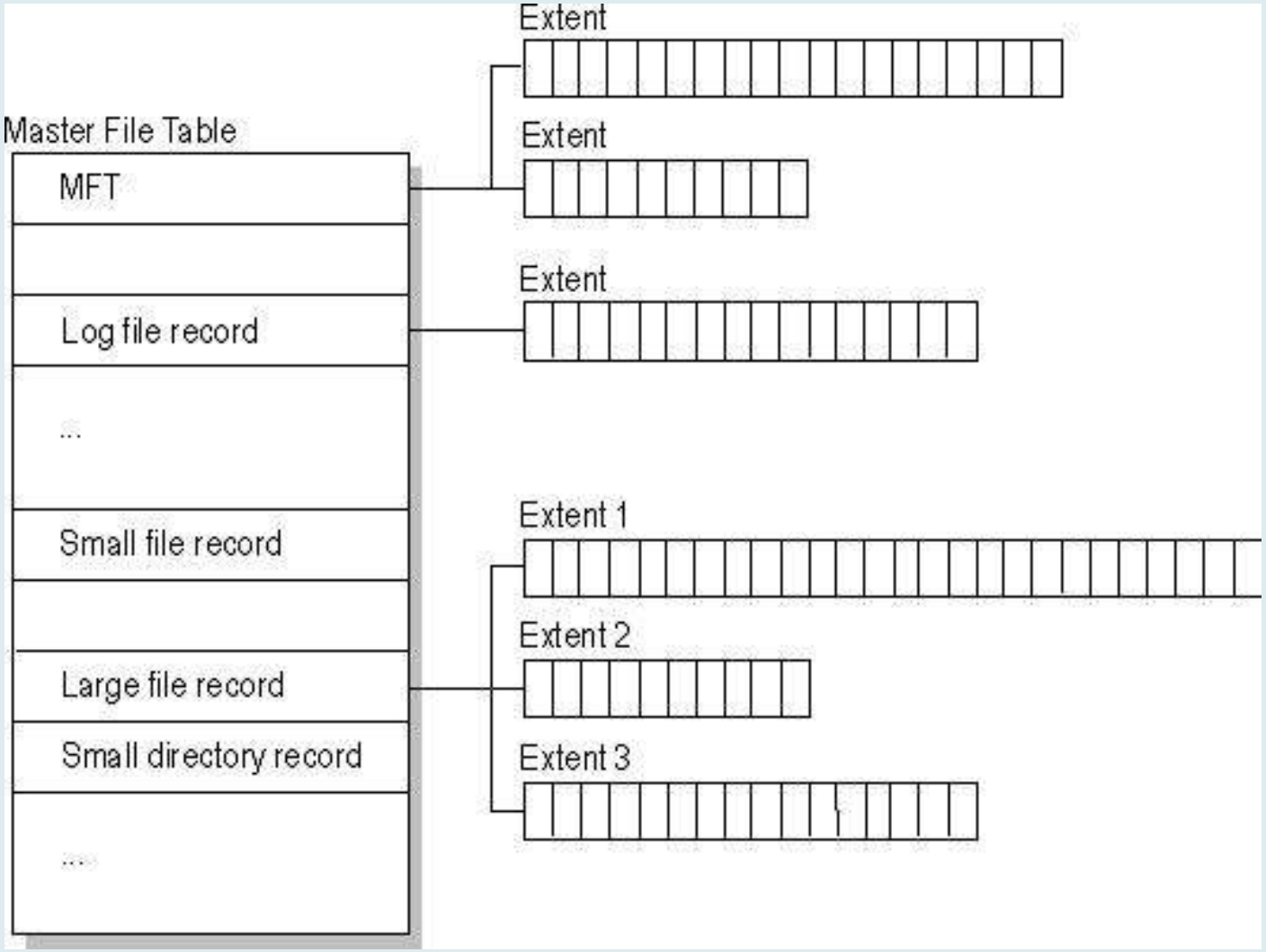
- 文件记录：
  - 三个通用属性
    - ✓$ Standard_Information
    - ✓$ File_Name
    - ✓$ Security _Descriptor
  - $ Data属性
    - ✓所有的文件属性至少有一个$ Data属性
    - ✓如果超出1K，数据部分移动到磁盘上，属性头和一小部分数据部分常驻

$ Data数据部分

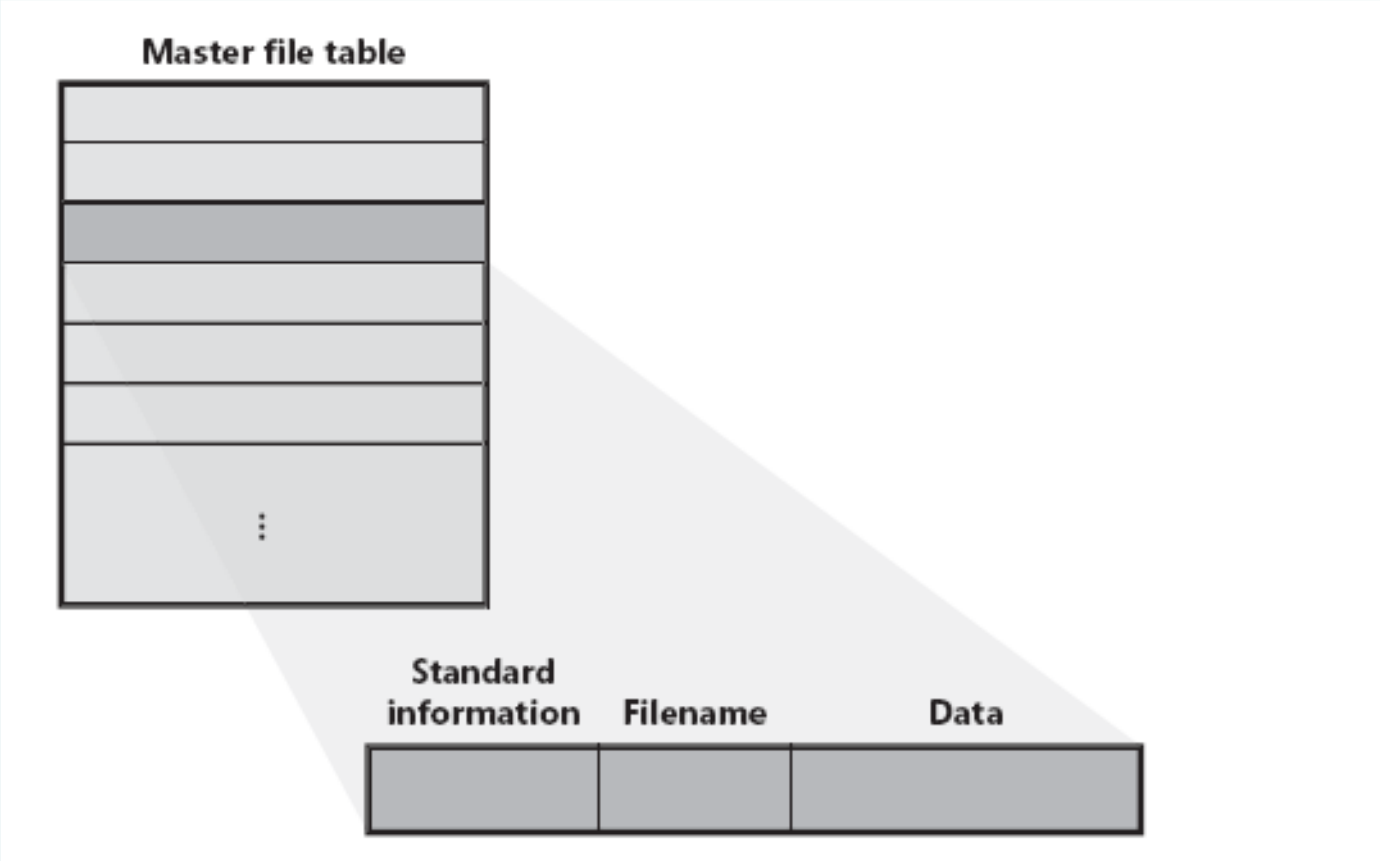
- 常驻属性头部分
- 常驻数据部分包括
  - 非常驻部分信息
  - 非常驻部分所在运行的指针
  - 簇数
  - 保留
  - 磁盘上的大小
  - 文件大小
  - 标志位
  - 位置指针

多个$ Data属性

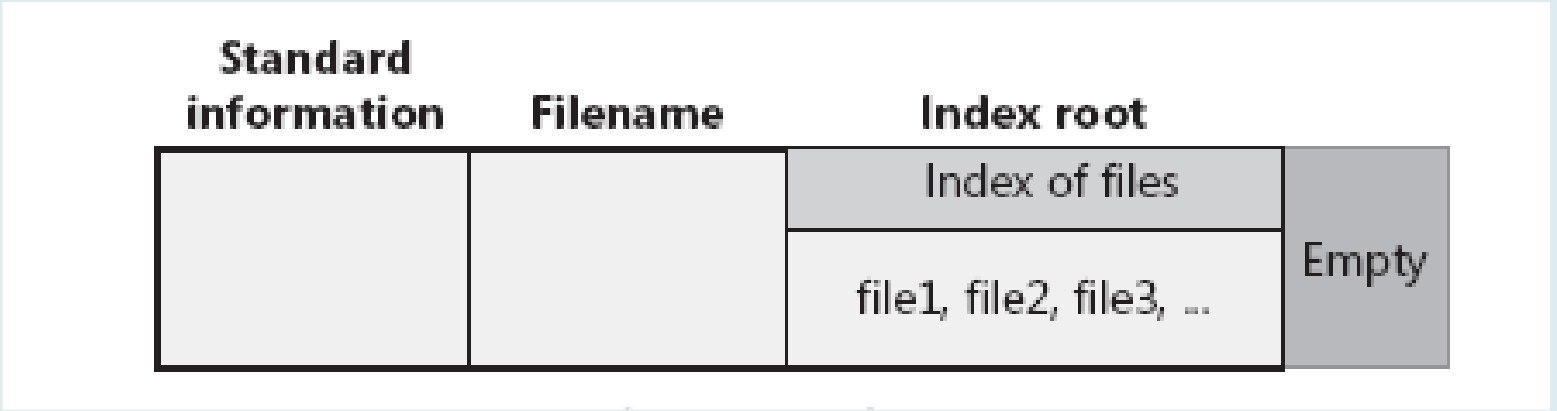
- 默认的$ Data属性没有名字
- 额外的$ Data属性必须有名字
- 命名数据流
  - 使用MORE命令将命名数据流通过管道输出
  - C:\more<super.txt
    It’s a example.


Master File Table
MFT
Log file record
...
Small file record
Large file record
Small directory record
...
Extent
Extent
Extent
Extent 1
Extent 2
Extent 3

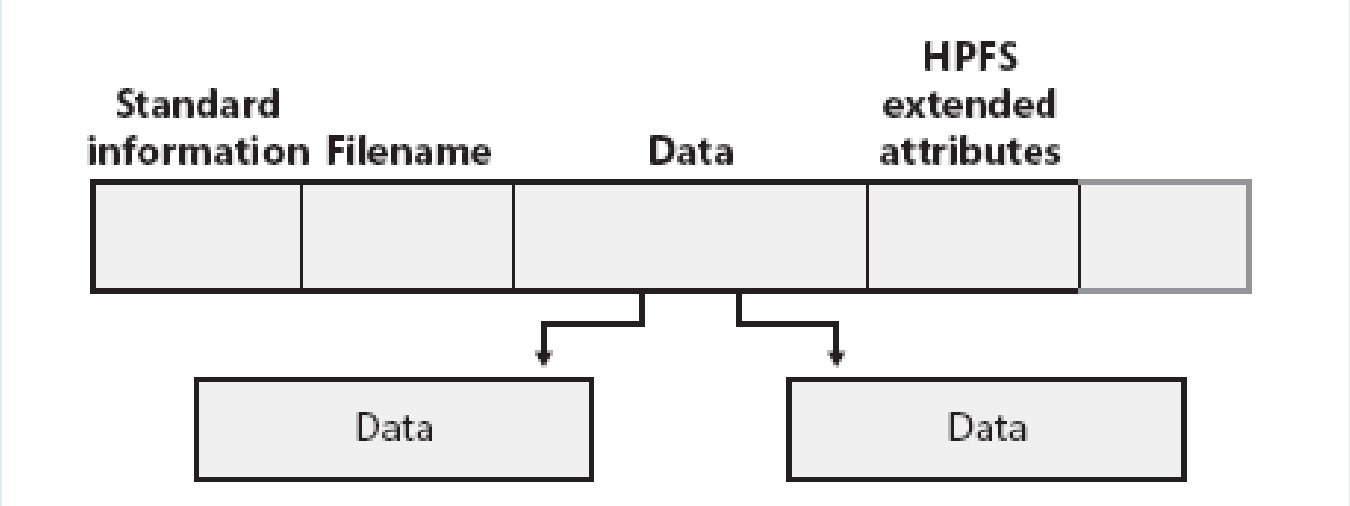
小文件的MFT记录



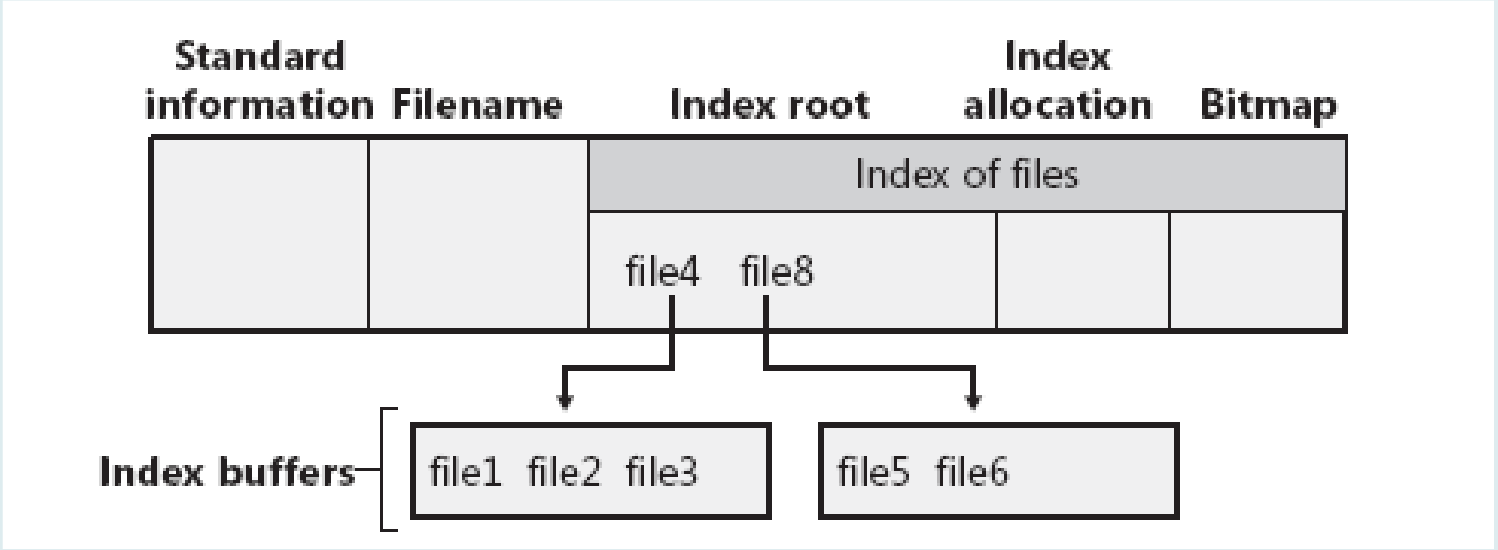
小型目录的MFT记录


Standard information
Filename
Index root
Index of files
file1, file2, file3, ...
Empty

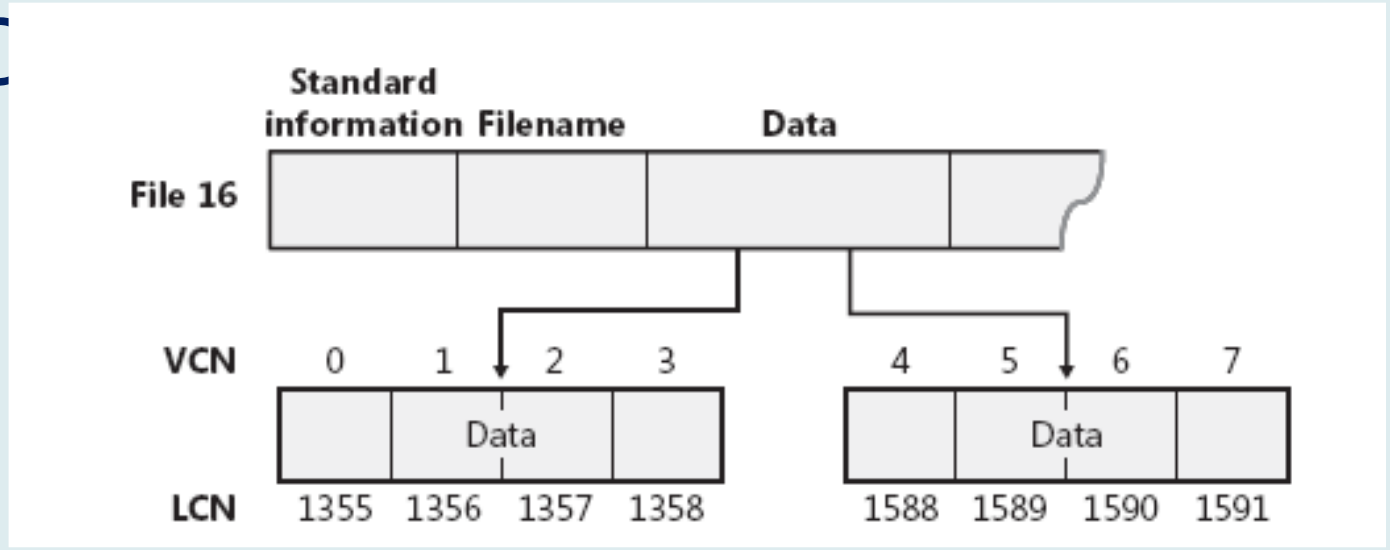
大文件的MFT文件记录

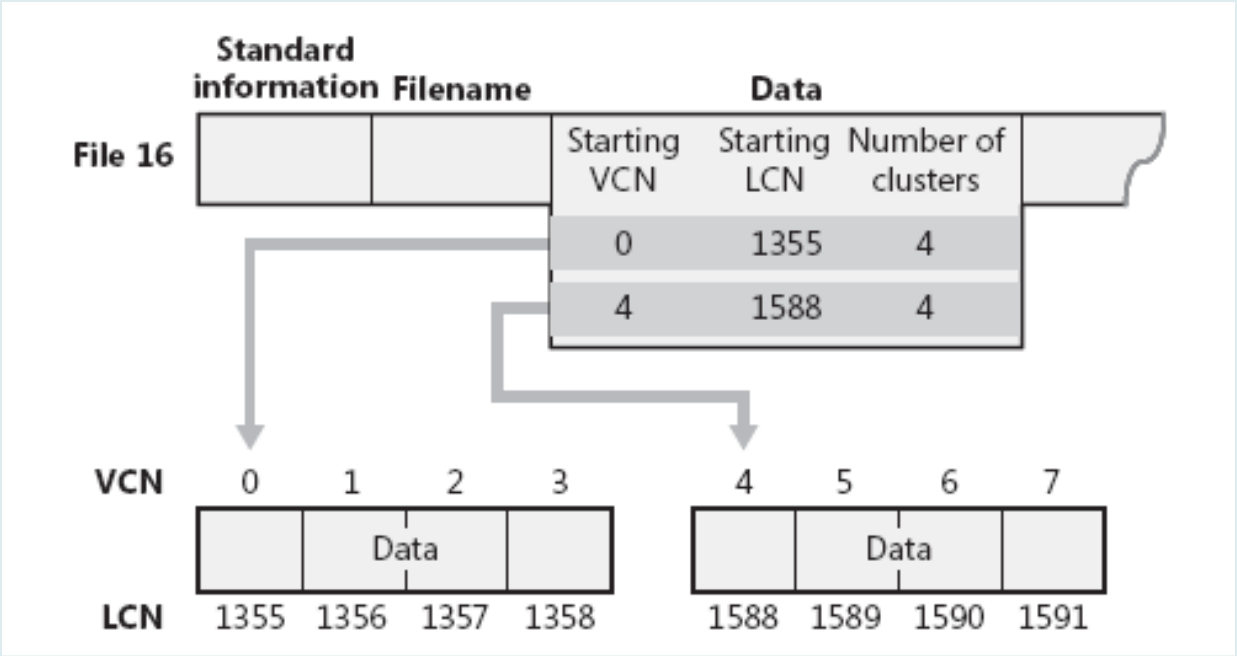


大型目录的MFT文件记录



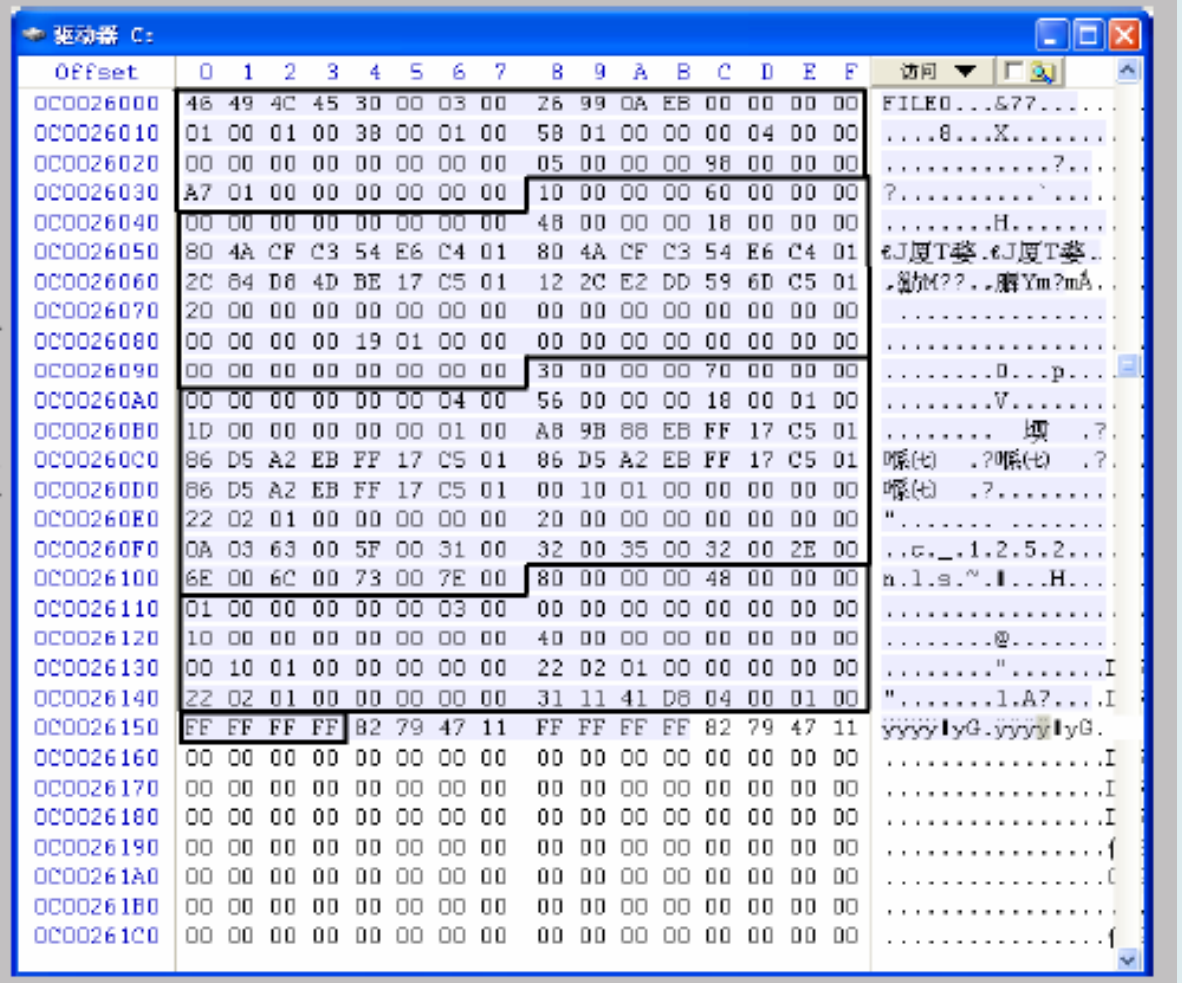
VC


Standard information
Filename
Data
File 16
VCN
0 1 2 3
4 5 6 7
Data
Data
LCN
1355 1356 1357 1358
1588 1589 1590 1591


Standard information
Filename
Data
File 16
Starting VCN
Starting LCN
Number of clusters
0 1355 4
4 1588 4
VCN
0 1 2 3
4 5 6 7
Data
Data
LCN
1355 1356 1357 1358
1588 1589 1590 1591

普通文件的MFT分析

- MFT头
- 10H类型属性（标准属性信息）
- 30H类型属性（文件名属性
- 80H类型属性（数据属性）
- MFT结束标志



4.6 管理文件与文件夹的访问许可权

- NTFS文件权限的类型
- 设置安全的访问许可权
- 文件与文件夹的访问许可冲突
- 查看文件与文件夹的访问许可权
- 更改文件或文件夹的访问许可权

4.6.1 NTFS文件夹权限的类型

- 读取
- 写入
- 列出文件夹目录
- 读取及运行
- 修改
- 完全控制

安全策略

4.6.2 设置安全的访问许可权

- 对服务器上的所有文件，实施强有力的基于许可的安全措施；
- 对中低安全性的安装，除系统卷和引导卷外，所有驱动器上均实施域用户（Domain User）管理，避免使用缺省的每个用户（Everyone）、完全控制（Full control）许可等安全措施；
- 对于高安全性安装，去掉所有Everyone、Full control许可权；
- 以机构中的自然关系为基础建立组，按组分配文件许可权；
- 利用第三方的许可审计软件管理复杂环境中的许可权问题 。

4.6.3 用户的有效权限(1)

- 权限具有累加性

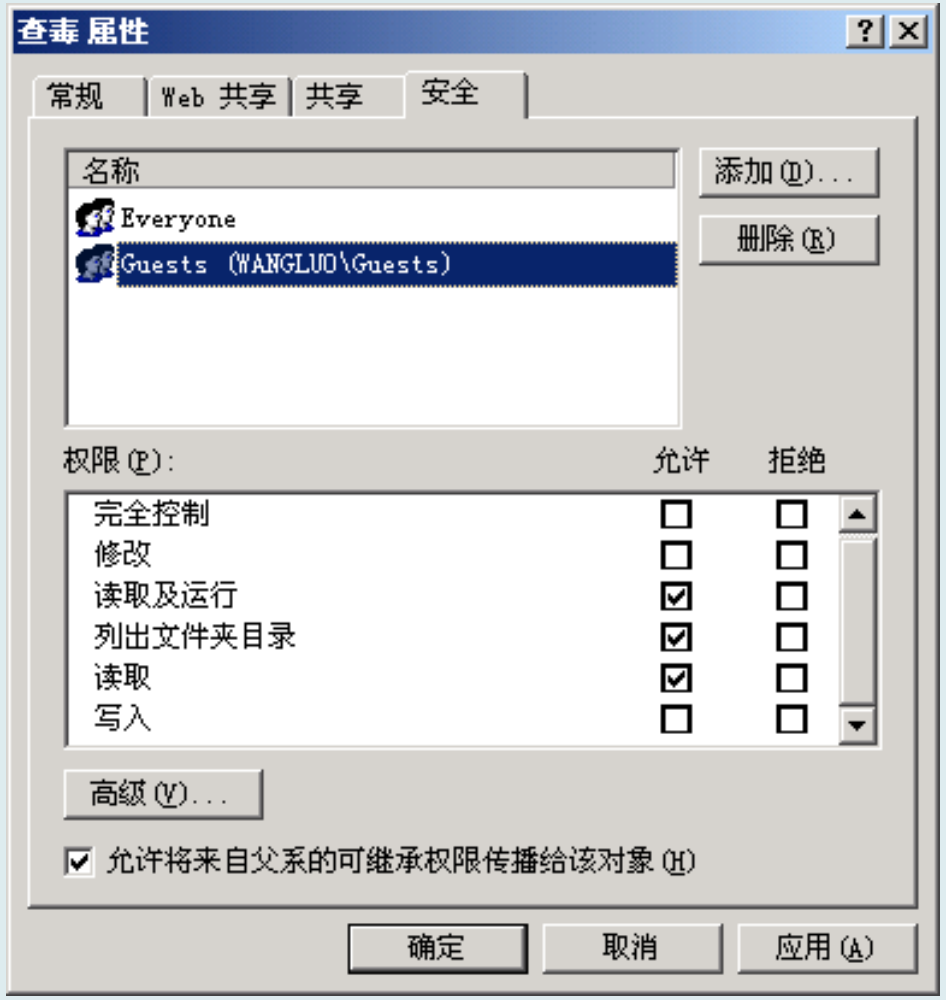
用户或组	权限
用户**A**	写入
组**Sales**	读取
组**Manager**	读取及运行
用户**A**最后的有效权限为	写入＋读取＋运行

4.6.3 用户的有效权限(2)

- 拒绝权限会覆盖所有其他权限
  - 用户拒绝权限可覆盖改用户、组其他权限
  - 在属性对话框“完全控制”处选择
- 文件权限会覆盖文件夹的权限
  - 文件的设置权限优先
  - 直接利用完整路径或共享文件夹来访问文件

4.6.4 查看文件与文件夹的访问许可权(1)

- 选定文件或文件夹的图标，单击鼠标右键打开快捷菜单
- 然后选择“属性”命令
- 在打开的文件或文件夹的属性对话框中单击“安全”标签。



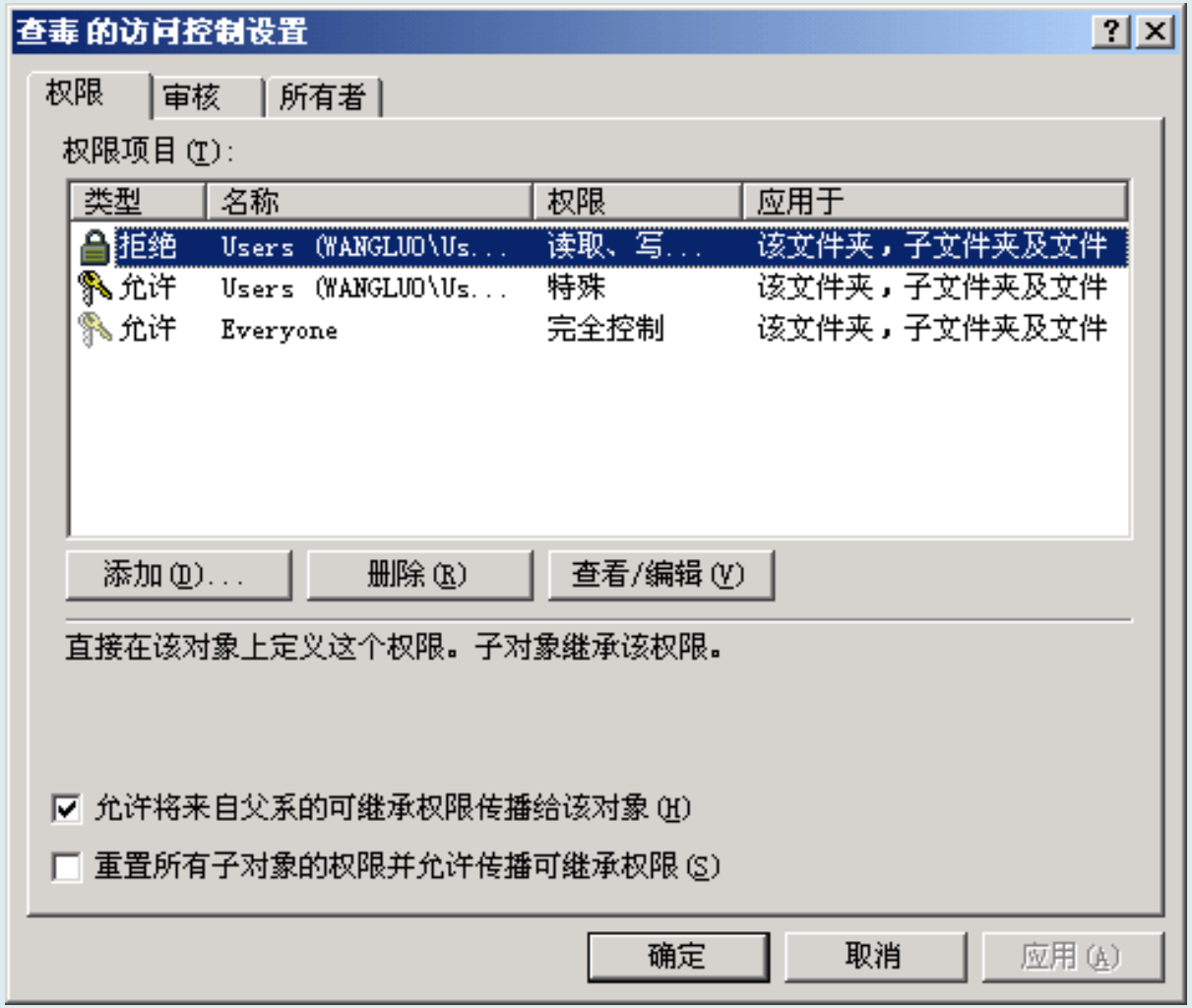
查看文件或文件夹的访问许可权

4.6.4 查看文件与文件夹的访问许可权(2)

没有列出来的用户（属于该选项中列出的某个组）也可能具有对文件或文件夹的访问许可权。因此，最好不要把对文件的访问许可权分配给各个用户，而把许可权分配给组，然后把用户添加到组中。这样需要更改的时候只需要更改整个组的访问许可权，而不必逐个修改每个用户。

4.6.5 更改文件或文件夹的访问许可权(3)

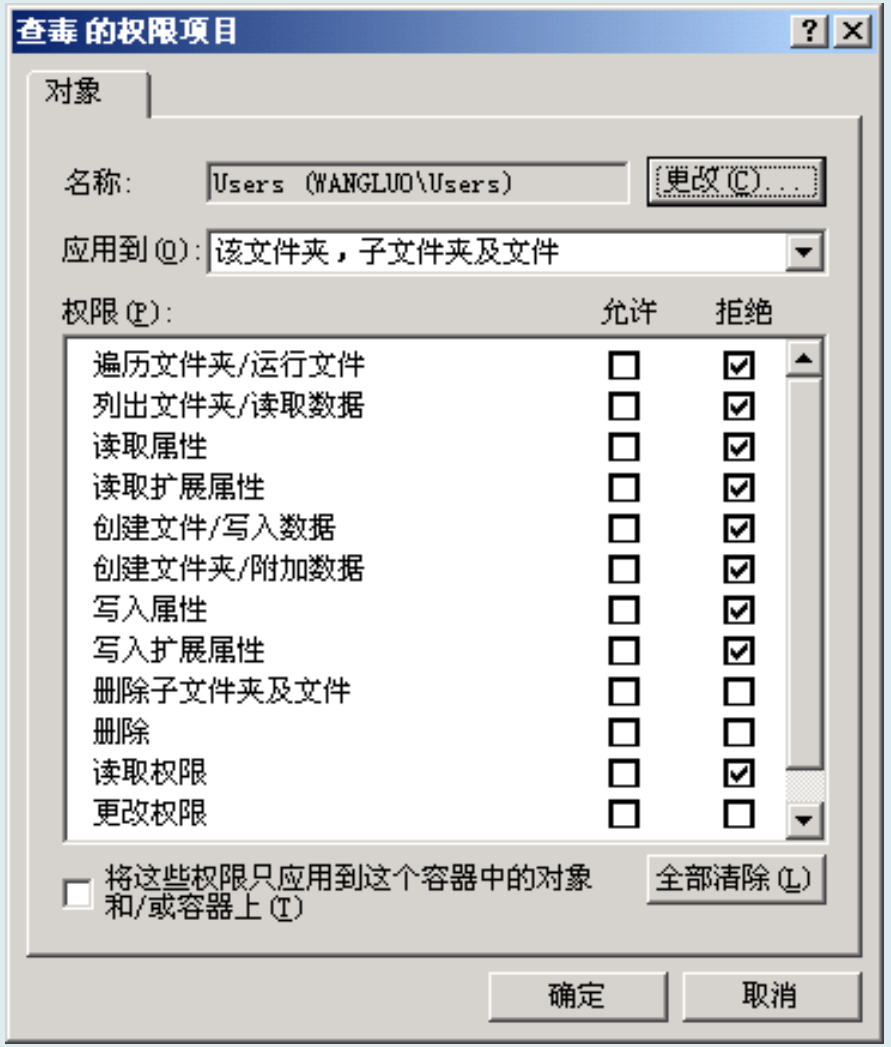
- 在如图所示的对话框中，选择需要设置的用户或组，简单地选定或取消对应权限后面的复选框；
- 单击“安全”标签下单击“高级”按钮，可以打开访问控制对话框。进一步设置一些额外的高级访问权限。



设置文件或文件夹的高级访问权限

4.6.5 更改文件或文件夹的访问许可权(4)

- 单击“查看/编辑”，打开选定对象的权限项目对话框，
- 用户可以通过“应用到”下拉列表框选择需设定用户或组，并对选定对象的访问权限进行更加全面的设置。



为用户或组设置额外的高级访问权限

4.7 共享文件夹

- 共享文件夹概念
- 共享文件夹权限
- 添加共享文件夹
- 停止共享文件夹
- 修改共享文件夹的属性
- 映射网络驱动器
- 断开网络驱动器

4.7.1 共享文件夹

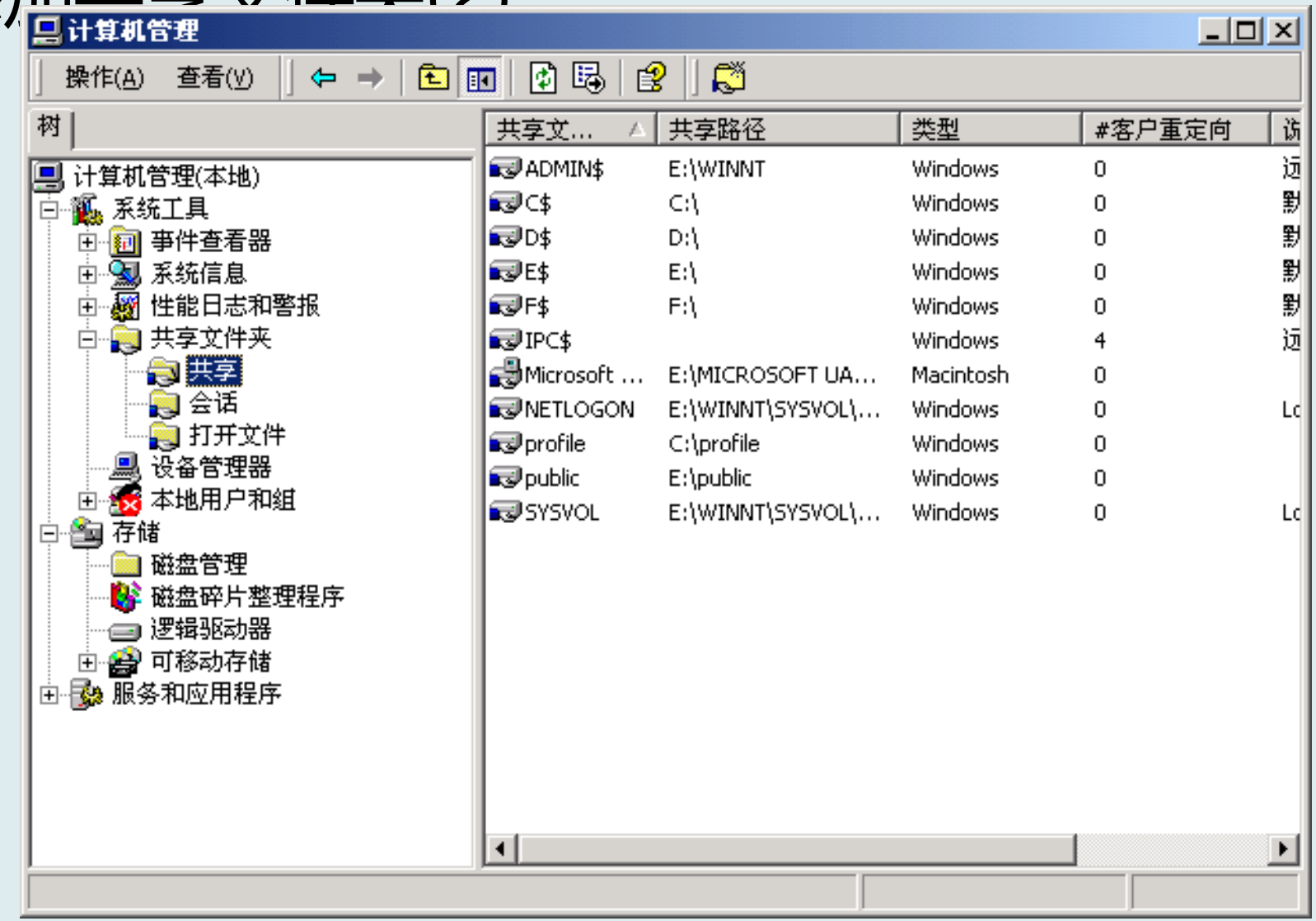
- 概念
- 权限

具备的能力	读取	修改	完全控制
查看该共享文件夹内的文件名称、子文件夹名称	V	V	V
查看文件内数据、运行程序	V	V	V
遍历子文件夹	V	V	V
添加文件、子文件夹		V	V
修改文件内的数据		V	V
删除文件与子文件夹			V
修改权限			V
取得所有权			V

4.7.2 添加共享文件夹(1)

- 步骤一，打开“开始”菜单，选择“程序”/“管理工具”/“计算机管理”命令后，打开“计算机管理”窗口，然后点击“共享文件夹”/“共享”子节点，打开如图所示窗口。

4.7.2 添加共享文件夹(2)

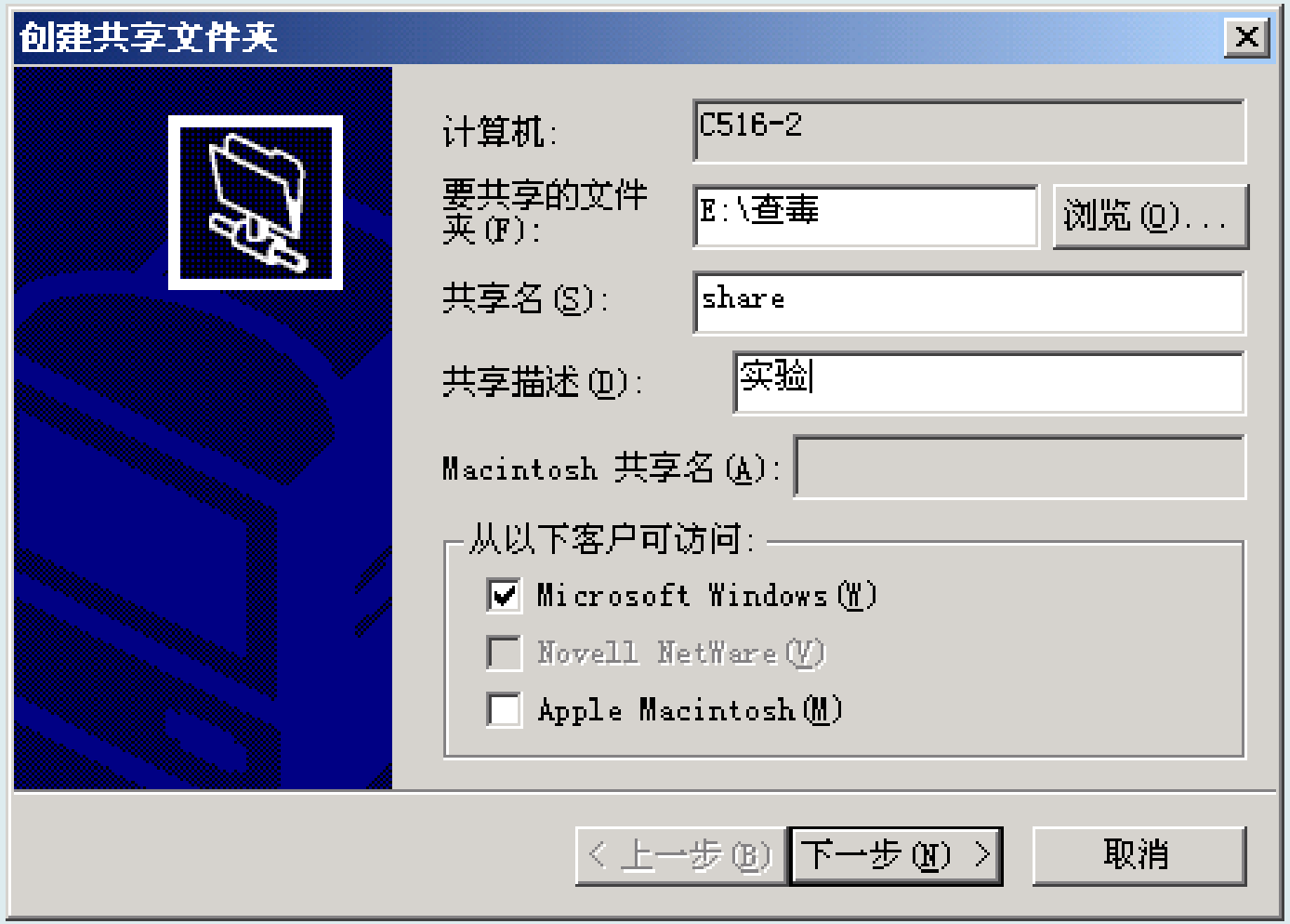


计算机管理窗口

4.7.2 添加共享文件夹(3)

- 步骤二，在窗口的右边显示出了计算机中所有共享文件夹的信息。
- 如果要建立新的共享文件夹，可通过选择主菜单“操作”中的“新文件共享”子菜单，或者在右侧窗口单击鼠标右键选择“共享”菜单，打开如图7-5所示对话框。输入要共享的文件夹、共享名、共享描述，在共享描述中可输入一些该资源的描述性信息，以方便用户了解其内容。

4.7.2 添加共享文件夹(4)

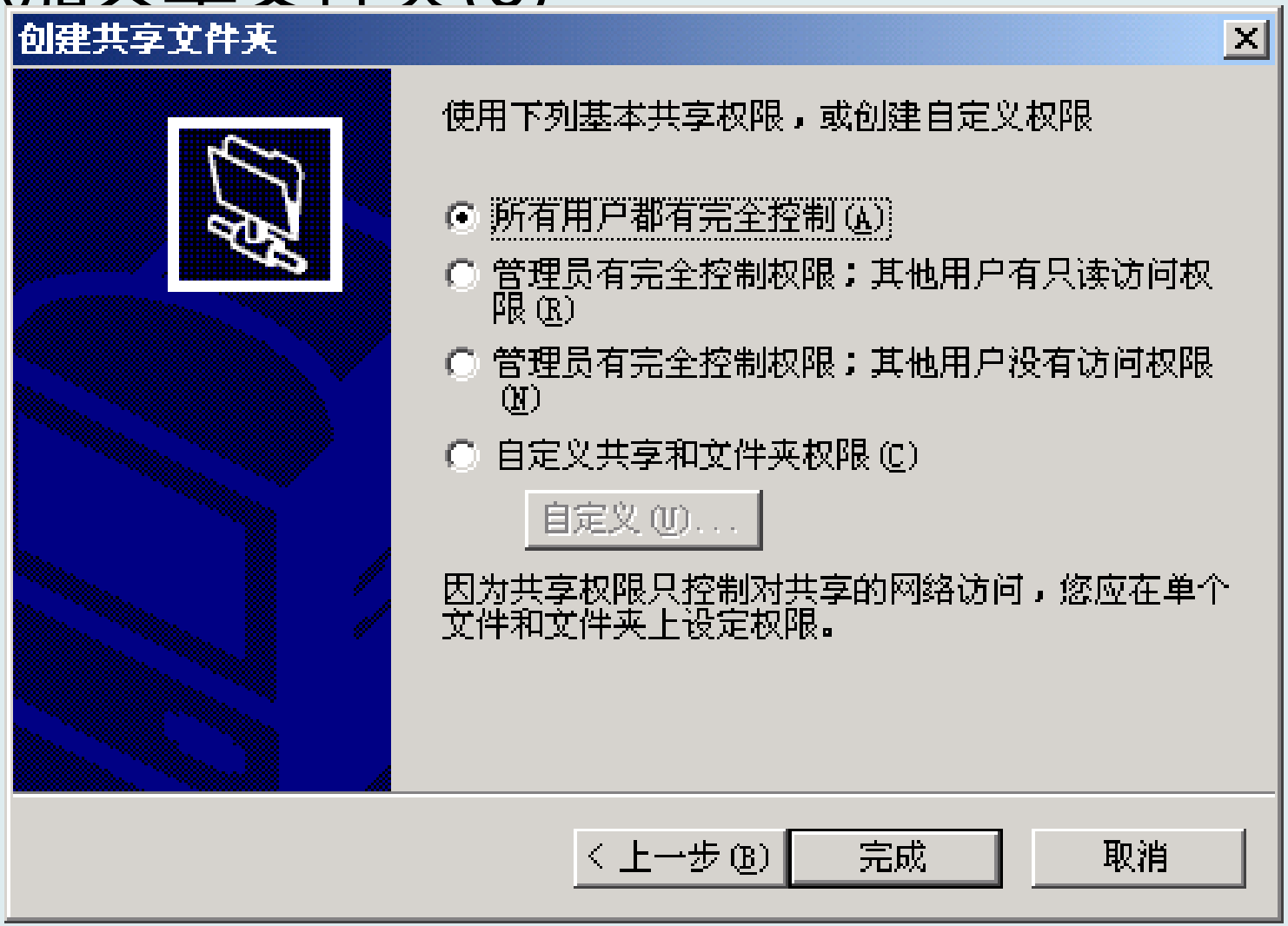


创建共享文件夹

4.7.2 添加共享文件夹(5)

- 步骤三，点击“下一步”，打开如图8-8所示“创建共享文件夹”对话框。用户可以根据自己的需要设置网络用户的访问权限。或者选择“自定义”自己定义网络用户的访问权限。

4.7.2 添加共享文件夹(6)



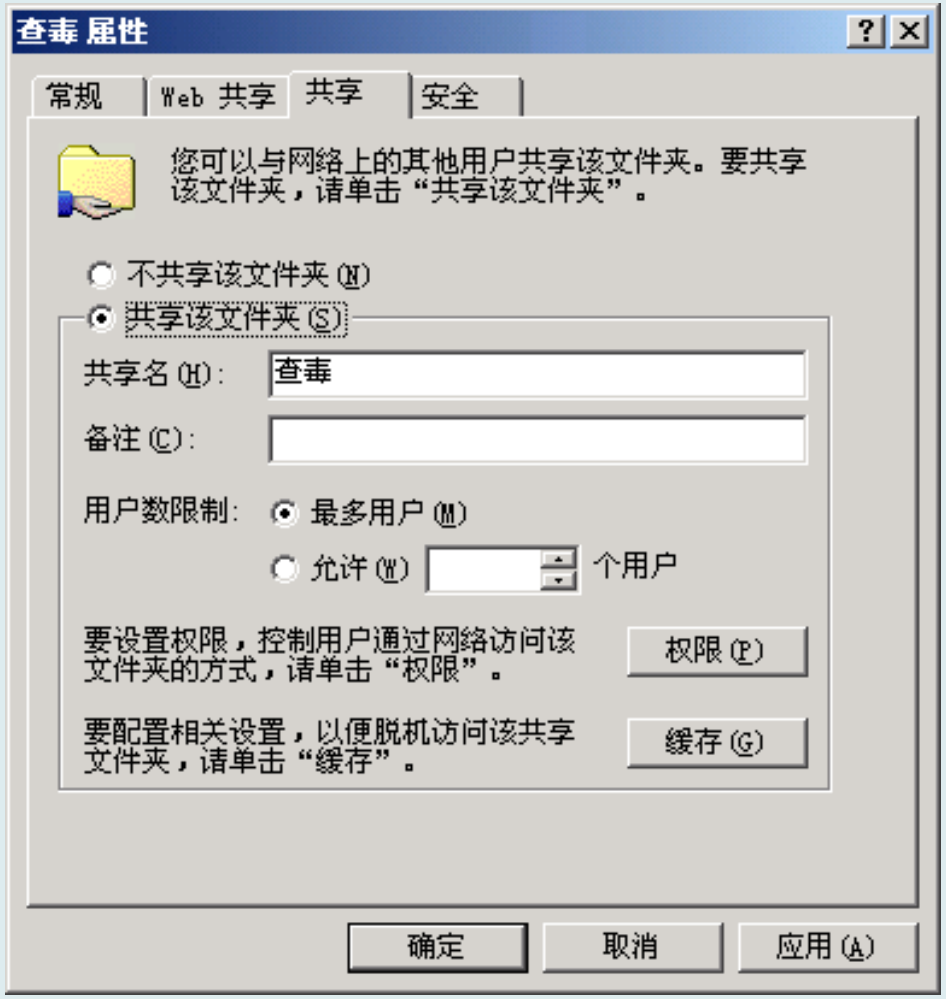
创建共享文件夹

4.7.2 添加共享文件夹(8)

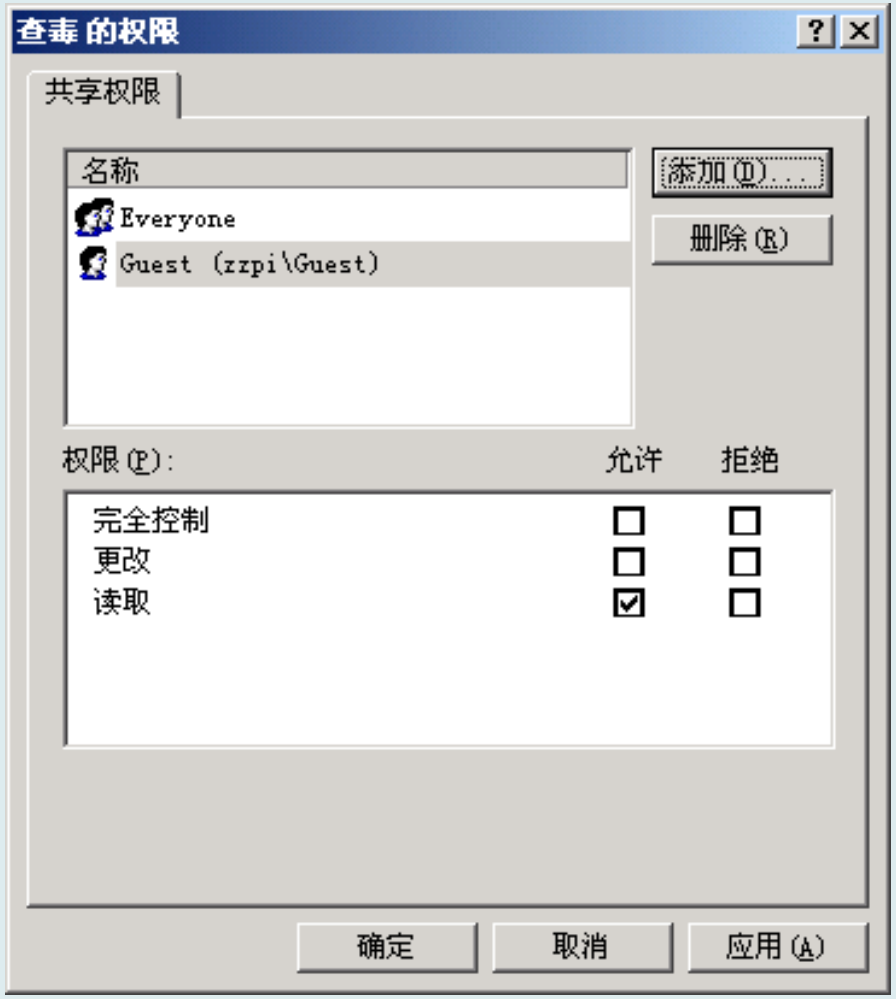
另一种方法：

- 双击“我的电脑”，然后选择要设置为共享文件夹的驱动器并选定文件夹。
- 鼠标右键激活快捷菜单，选择“共享”菜单项，打开如图所示7-7窗口。
- 然后进行相应的设置，如更改共享名，设定用户连接数量，点击“权限”按钮，如图7-8所示，设置允许访问的用户权限。

4.7.2 添加共享文件夹(8)



文件夹的共享选项



文件夹的共享许可权限

4.7.3 停止共享文件夹

方法1：

- 在“计算机管理”窗口中，选择要停止共享的文件夹；
- 点击右键，选择“停止共享”；
- 在弹出的对话框里，点击“确定”按钮即可。

方法2：

- 双击“我的电脑”图标，选定已经设为共享的文件夹；
- 右击该文件夹，选择“共享”命令，打开共享“选项卡”；
- 单击“不共享该文件夹”，点击“确定”按钮即可。

4.7.4 修改共享文件夹的属性

- 选择共享文件夹，点击右键，选择“属性”，打开如图所示对话框；
- 在“常规”对话框里，可以设置允许多少用户同时访问该共享文件夹以及缓存设置；
- 可以通过选择“共享权限”、“安全”选项卡，修改组和用户的共享访问许可，或该文件/文件夹访问许可的设置；
- 点击“确定”按钮即可使配置生效。

4.7.4 修改共享文件夹的属性

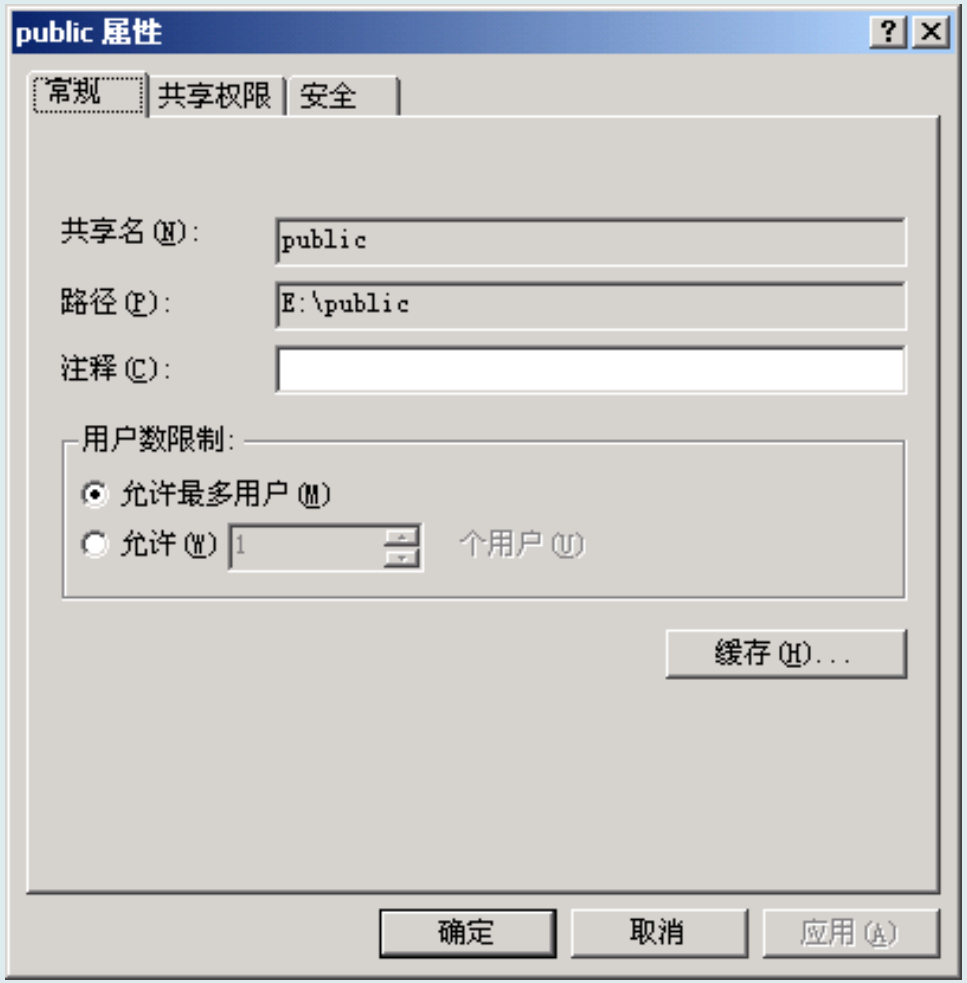


图7-10 “Public”对话框

4.7.5 映射网络驱动器

- 右击“我的电脑”，选择“映射网络驱动器”，打开如图7-11所示对话框；
- 在“驱动器”下拉列表框中，选择一个本机没有的盘符作为共享文件夹的映射驱动器符号。输入要共享的文件夹名及路径；或者点击“浏览”按钮打开“浏览文件夹”对话框，选择要映射的文件夹；
- 如果需要下次登录时自动建立同共享文件夹的连接，选定“登陆时重新连接”复选框；
- 单击“完成”，即可完成对共享文件夹到本机的映射。

4.7.5 映射网络驱动器

映射网络驱动器对话框

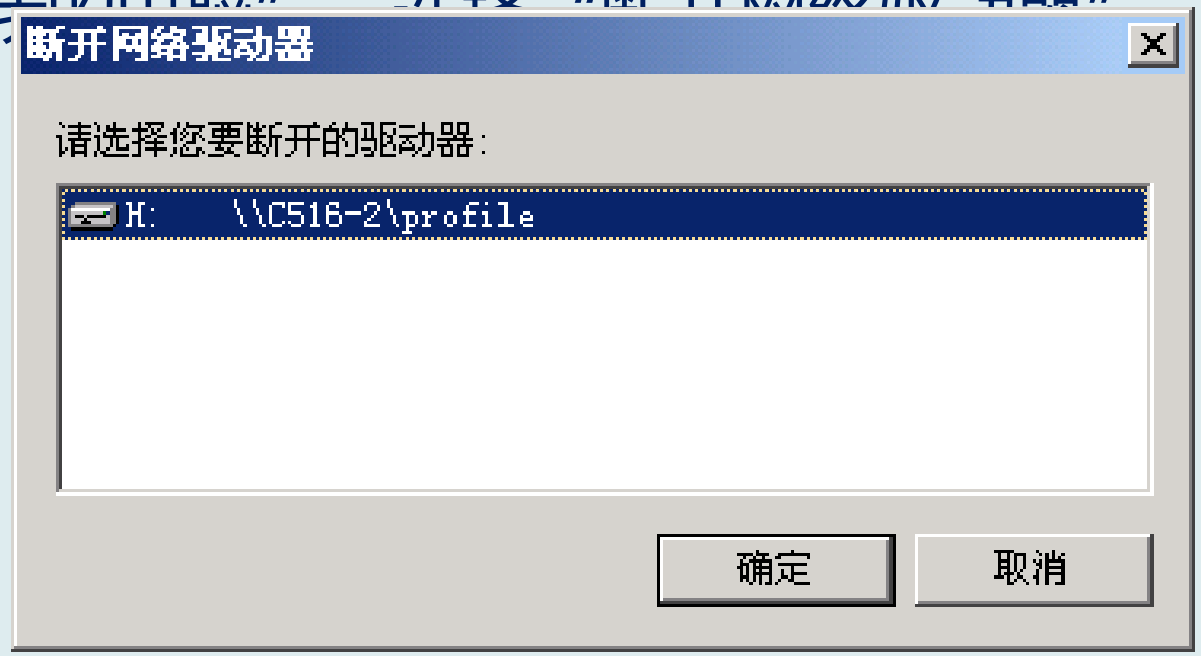
4.7.5 映射网络驱动器

- 打开“我的电脑”，将发现本机多了一个驱动器符，通过该驱动器符可以访问该共享文件夹，如同访问本机的物理磁盘

通过映射的驱动器访问共享文件夹

4.7.5 断开网络驱动器

- 右击“我的电脑”，选择“断开网络驱动器”



- 选择要断开的网络驱动器，点击“确定”即可

上机练习作业

- 采用文件读写方式，按指定顺序合并某个文件夹中的文本文件集