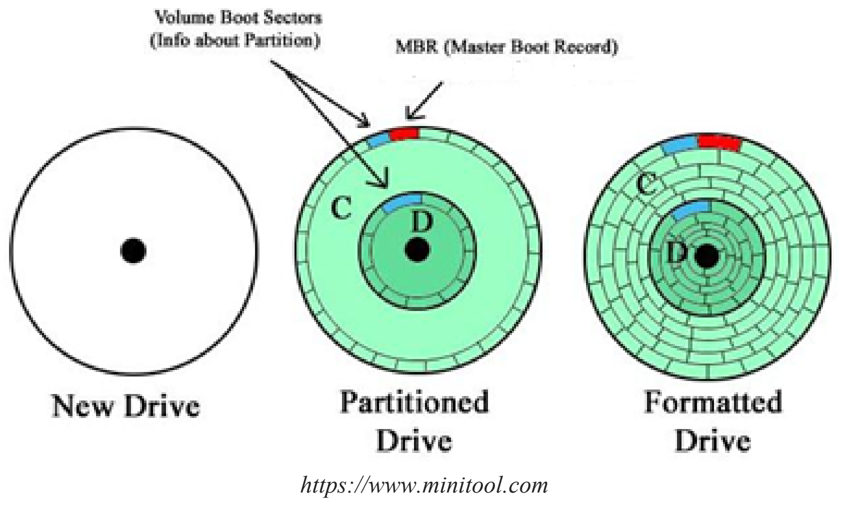
Exploring Microsoft File Structures

- In Microsoft file structures, sectors are grouped to form **clusters**
  - **Clusters can have one sector or more**
- OS assigns these **cluster numbers**, called **logical addresses**
  - *Address point to relative position*
- **Clusters** are numbered sequentially starting at 0 in **NTFS** and 2 in **FAT**
- Sector numbers are called **physical addresses**
  - *From address 0 to last sector on disk*
  - **First sector** of all disks contains a system area, the boot record, and a file structure database
- Clusters and their addresses are specific to a **logical disk drive**, which is a disk partition

Disk Partitions

- A **partition** is a logical drive. *i.e "**C**", "**D**", "**E**" and etc*
- Windows OSs can have three primary partitions followed by an extended partition that can contain one or more logical drives



- **Partition gap**
  - Unused space between partitions
  - ***Can use to hide data!***

One disk can only have one Master Boot Record but multiple volume boot sectors

Disk Partitions (Cont)

- The **partition table(s)** is in the **Master Boot Record (MBR)**
  - Located at **sector 0** (*first 512 bytes*) of the disk drive
- **MBR** stores information about partitions on a disk and their locations, size, and other important items

- In a hexadecimal editor, such as WinHex, you can find the **first** partition at **offset 0x1BE**
  - **File system's hexadecimal code (*partition type*)** is offset **3** bytes from 0x1BE for the first partition
  - **Sector address** of where this partition **starts** on the drive is offset **8** bytes from 0x1BE.
  - **The number of sectors** assigned to the partition are offset **12** bytes for position 0x1BE.

Disk Partitions (Cont)

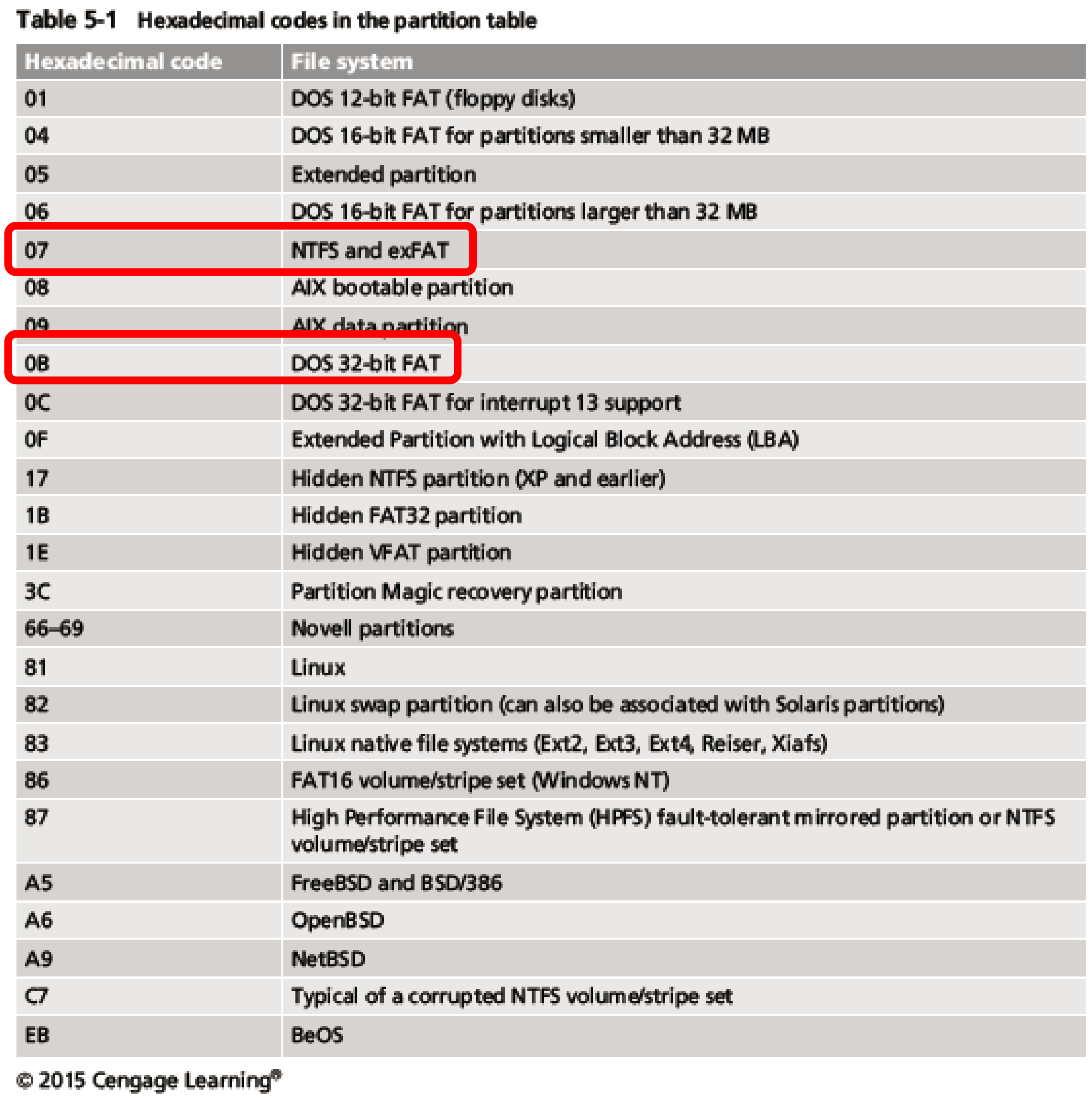


Table 5-1 **Hexadecimal codes in the partition table**

Hexadecimal code	File system
01	DOS 12-bit FAT (floppy disks)
04	DOS 16-bit FAT for partitions smaller than 32 MB
05	Extended partition
06	DOS 16-bit FAT for partitions larger than 32 MB
07	NTFS and exFAT
08	AIX bootable partition
09	AIX data partition
0B	DOS 32-bit FAT
0C	DOS 32-bit FAT for interrupt 13 support
0F	Extended Partition with Logical Block Address (LBA)
17	Hidden NTFS partition (XP and earlier)
1B	Hidden FAT32 partition
1E	Hidden VFAT partition
3C	Partition Magic recovery partition
66–69	Novell partitions
81	Linux
82	Linux swap partition (can also be associated with Solaris partitions)
83	Linux native file systems (Ext2, Ext3, Ext4, Reiser, Xiafs)
86	FAT16 volume/stripe set (Windows NT)
87	High Performance File System (HPFS) fault-tolerant mirrored partition or NTFS volume/stripe set
A5	FreeBSD and BSD/386
A6	OpenBSD
A9	NetBSD
C7	Typical of a corrupted NTFS volume/stripe set
EB	BeOS

*A **partition table** is a **table** maintained on disk by the operating system describing the **partition** on that disk.*

key hexadecimal codes** is used by OS to identify and maintain the file system. **i.e 0x07 => NTFS

Drive Slack, RAM Slack, File Slack

- Microsoft OSs allocate disk space for files by **clusters**
  - Results in **Drive Slack**
    - Unused space in a cluster between the end of an active file and the end of the cluster

- **Drive Slack** includes:
  - **RAM slack** (*unused space between EOF and end of last used sector*) and
  - **File slack** (*unused space allocated for a file*)

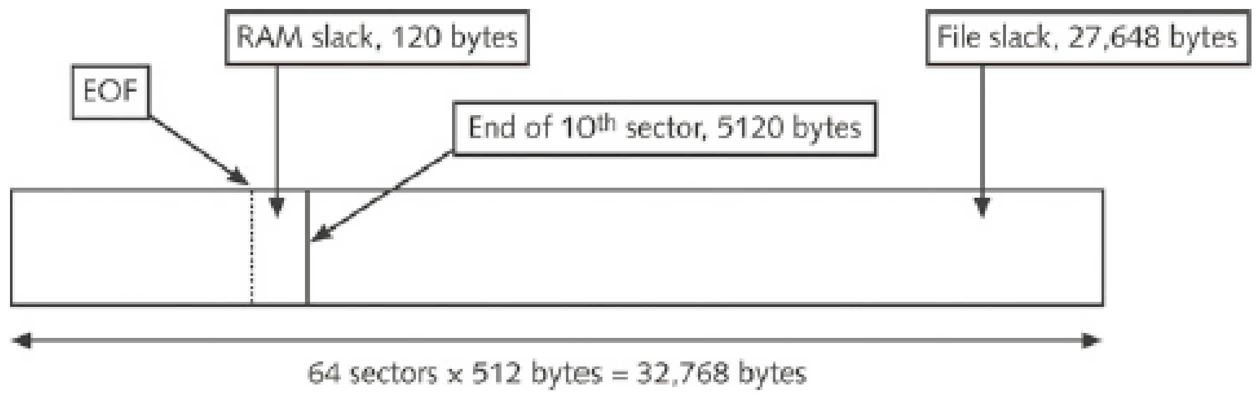

RAM slack, 120 bytes
File slack, 27,648 bytes
EOF
End of 10th sector, 5120 bytes
64 sectors x 512 bytes = 32,768 bytes

Figure 5-8 **File slack space**
© 2015 Cengage Learning®

- *Document size is 5000 bytes of data.*
- *Assuming hard disk size is 1.6G, the OS allocates about 32,000 bytes, or 64 sectors (512 bytes per sector), for this file.*
- *The unused space, 27,000 bytes, is the file slack.*
- *RAM slack is the portion between EOF and the last sector (10th sector) used. So the RAM slack is 120 bytes. The remaining sectors are referred to as "file slack" and is equal to 27648 bytes.*
- *See diagram above*

*Note : The data used to fill the **120-byte** void (RAM Slack) is pulled from RAM. Any information in RAM at that point, such as logon IDs or passwords, is placed in RAM slack!!*

Default : 1 sector = 512 bytes

File Systems

FAT

NTFS

Examining FAT Disks

- **File Allocation Table (FAT)**
  - File structure database that Microsoft originally designed for floppy disks
- FAT database is typically written to a disk’s outermost track and contains:
  - Filenames, directory names, date and time stamps, the starting cluster number, and file attributes
- Three current FAT versions
  - FAT16, FAT32, and exFAT (used by Xbox game systems)

Examining FAT Disks

- Cluster sizes vary according to disk drive size and file system

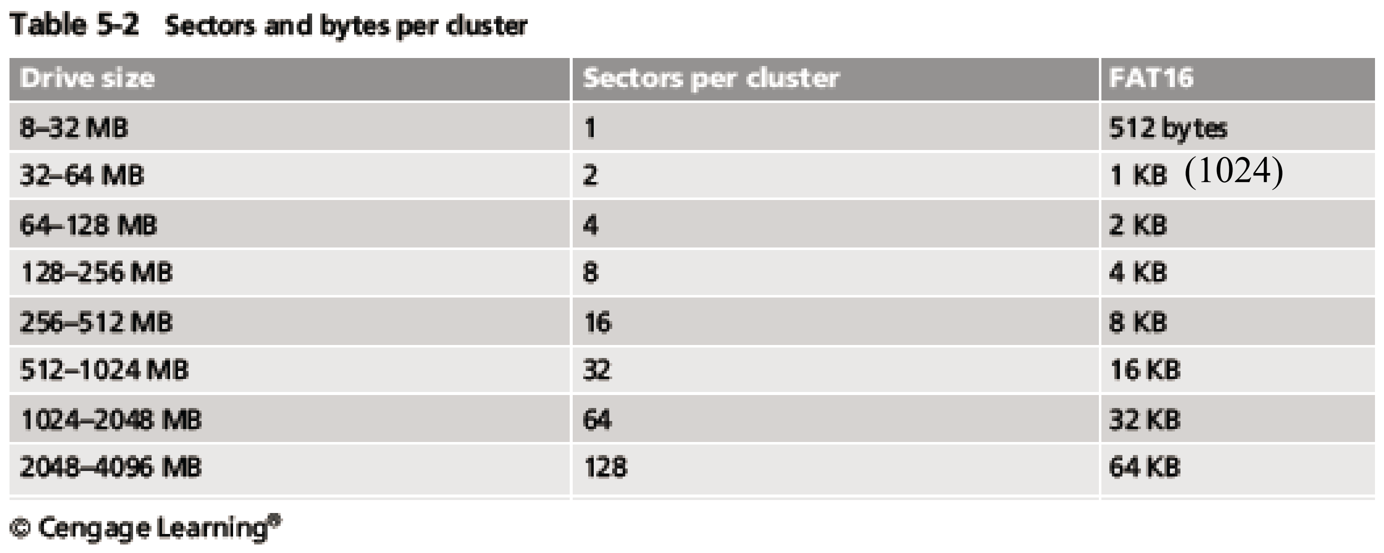


Table 5-2 Sectors and bytes per cluster

Drive size	Sectors per cluster	FAT16
8–32 MB	1	512 bytes
32–64 MB	2	1 KB (1024)
64–128 MB	4	2 KB
128–256 MB	8	4 KB
256–512 MB	16	8 KB
512–1024 MB	32	16 KB
1024–2048 MB	64	32 KB
2048–4096 MB	128	64 KB

© Cengage Learning®

Note : sector size = 512 bytes

Examining NTFS Disks (Cont)

- In NTFS, everything written to the disk is considered a **file**
- On an **NTFS** disk
  - First data set is the **Partition Boot Sector**
  - Next is **Master File Table (MFT)**
- In NTFS, clusters are smaller for smaller disk drives
- NTFS results in much less file slack space

Examining NTFS Disks (Cont)

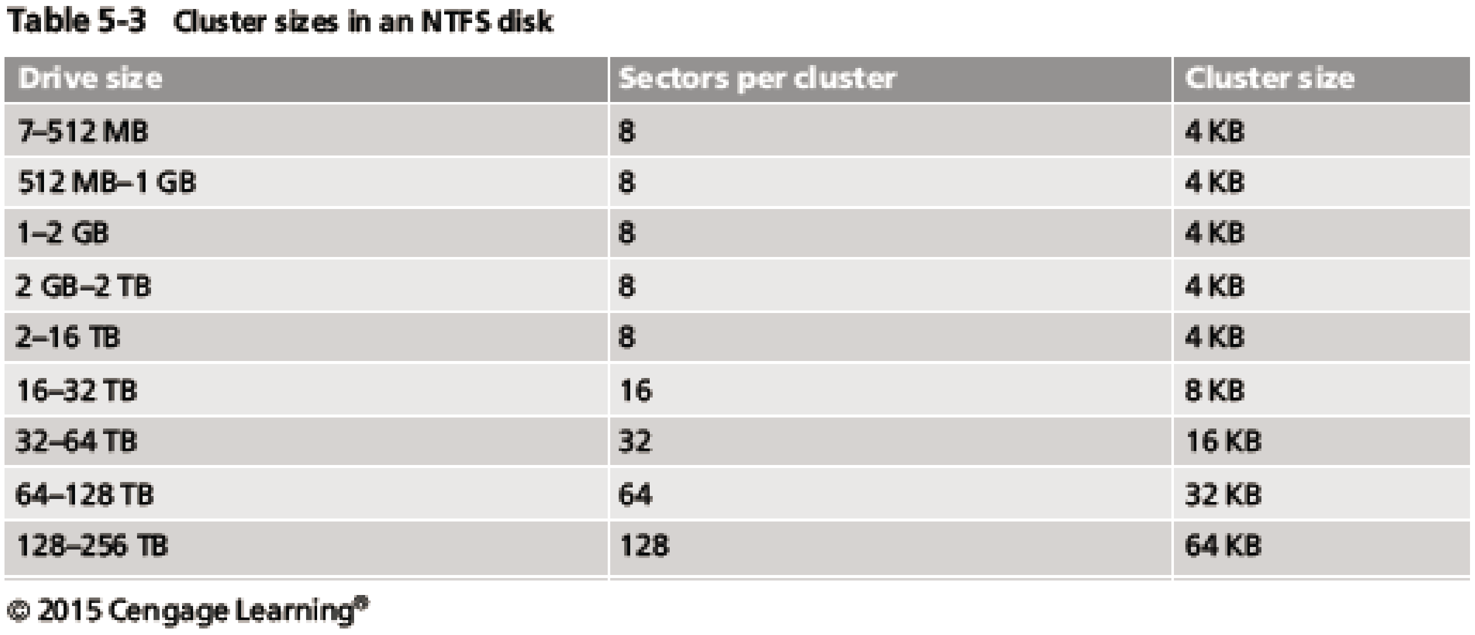
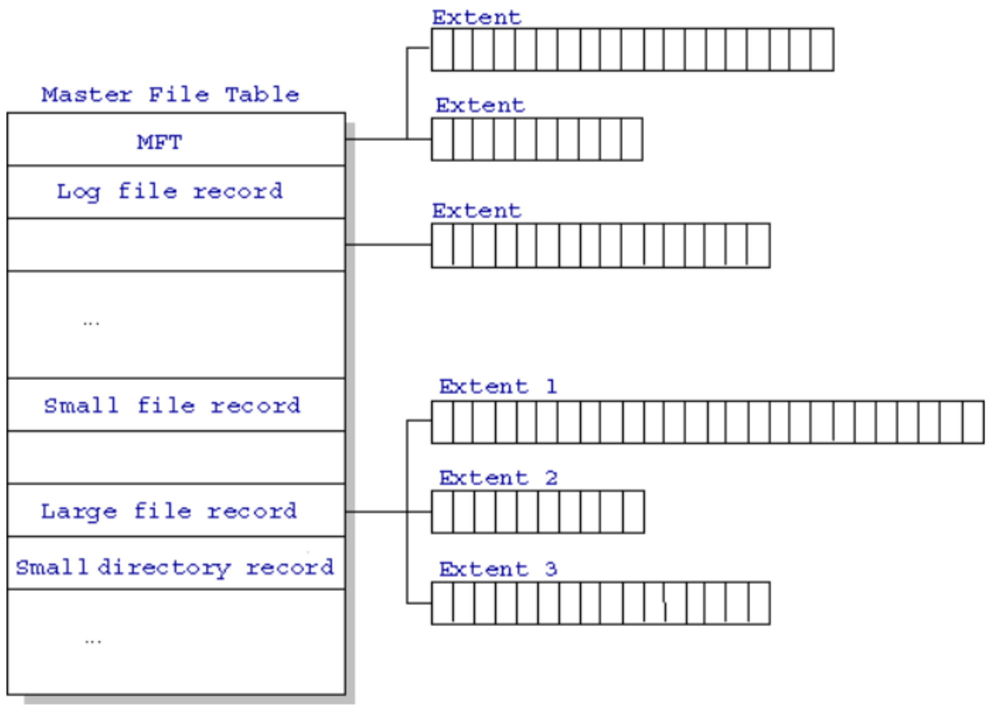


Table 5-3 Cluster sizes in an NTFS disk

Drive size	Sectors per cluster	Cluster size
7–512 MB	8	4 KB
512 MB–1 GB	8	4 KB
1–2 GB	8	4 KB
2 GB–2 TB	8	4 KB
2–16 TB	8	4 KB
16–32 TB	16	8 KB
32–64 TB	32	16 KB
64–128 TB	64	32 KB
128–256 TB	128	64 KB

Note : sector size = 512 bytes

MFT Structure



The master file table allocates a certain amount of space for each file record. The attributes of a file are written to the allocated space in the MFT. Small files and directories (typically 512 bytes or smaller), such as the file illustrated in next figure, can entirely be contained within the master file table record.

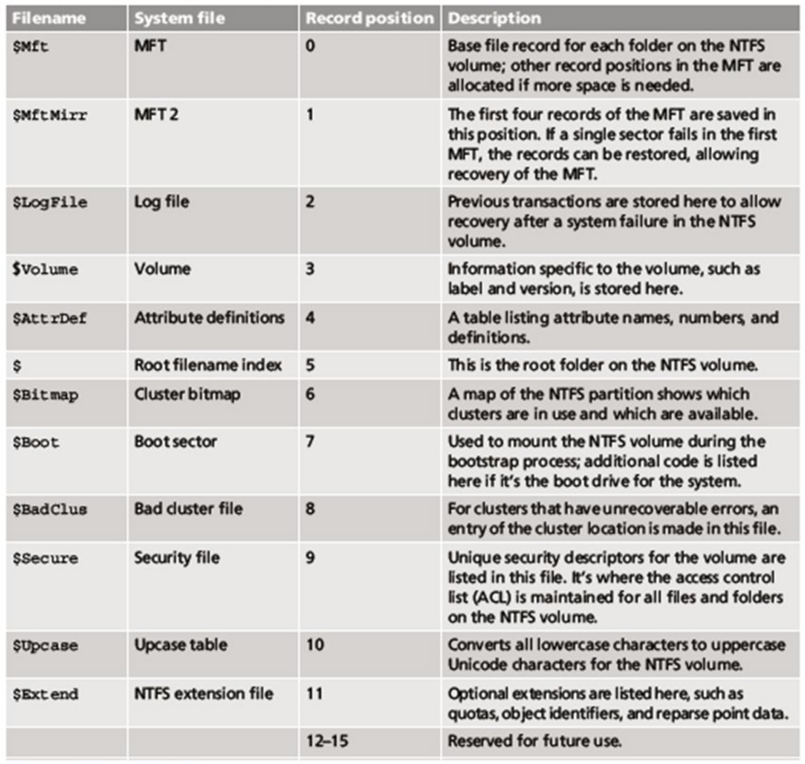
Ref :ntfs.com/ntfs-mft.htm

NTFS System Files

Master File Table (MFT) contains information about **all files** on the disk including the system files OS uses

- In the MFT, the first 15 records are reserved for system files used to describe MFT. They have file names beginning with a $ sign
- Records in the MFT are called **metadata**

Table 5-4 Metadata records in the MFT



Filename	System file	Record position	Description
`$Mft`	MFT	0	Base file record for each folder on the NTFS volume; other record positions in the MFT are allocated if more space is needed.
`$MftMirr`	MFT 2	1	The first four records of the MFT are saved in this position. If a single sector fails in the first MFT, the records can be restored, allowing recovery of the MFT.
`$LogFile`	Log file	2	Previous transactions are stored here to allow recovery after a system failure in the NTFS volume.
`$Volume`	Volume	3	Information specific to the volume, such as label and version, is stored here.
`$AttrDef`	Attribute definitions	4	A table listing attribute names, numbers, and definitions.
`$`	Root filename index	5	This is the root folder on the NTFS volume.
`$Bitmap`	Cluster bitmap	6	A map of the NTFS partition shows which clusters are in use and which are available.
`$Boot`	Boot sector	7	Used to mount the NTFS volume during the bootstrap process; additional code is listed here if it's the boot drive for the system.
`$BadClus`	Bad cluster file	8	For clusters that have unrecoverable errors, an entry of the cluster location is made in this file.
`$Secure`	Security file	9	Unique security descriptors for the volume are listed in this file. It's where the access control list (ACL) is maintained for all files and folders on the NTFS volume.
`$Upcase`	Upcase table	10	Converts all lowercase characters to uppercase Unicode characters for the NTFS volume.
`$Extend`	NTFS extension file	11	Optional extensions are listed here, such as quotas, object identifiers, and reparse point data.
		12–15	Reserved for future use.

© 2015 Cengage Learning®

MFT File Attributes

- In NTFS MFT, all files and folders are stored in separate **records** of **1024 bytes** each

- Each record contains file or folder information
  - This information is divided into record fields
  - A record field is referred to as an **attribute ID**

- File or folder information is typically stored in one of two ways in an MFT record:
  - Resident and nonresident

MFT and File Attributes (Cont)

- All **MFT record (*files or folders*)** starts with a **56 byte** value known as **MFT header**.
- For very small files(*resident*), about 512 bytes or less, all file metadata and data are stored in the MFT record.
- Files larger than 512 bytes (*nonresident*) are stored outside the MFT
  - MFT record provides **cluster addresses** where the file is stored on the drive's partition
    - Referred to as **data runs**

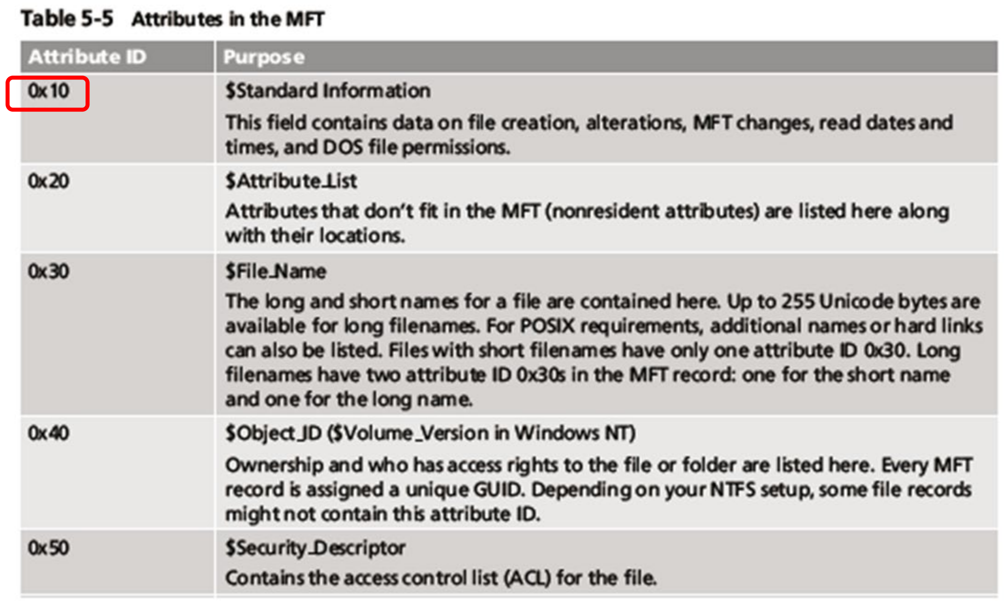


Table 5-5 Attributes in the MFT

Attribute ID	Purpose
0x10	$Standard Information This field contains data on file creation, alterations, MFT changes, read dates and times, and DOS file permissions.
0x20	$Attribute_List Attributes that don't fit in the MFT (nonresident attributes) are listed here along with their locations.
0x30	$File_Name The long and short names for a file are contained here. Up to 255 Unicode bytes are available for long filenames. For POSIX requirements, additional names or hard links can also be listed. Files with short filenames have only one attribute ID 0x30. Long filenames have two attribute ID 0x30s in the MFT record: one for the short name and one for the long name.
0x40	$Object_ID ($Volume_Version in Windows NT) Ownership and who has access rights to the file or folder are listed here. Every MFT record is assigned a unique GUID. Depending on your NTFS setup, some file records might not contain this attribute ID.
0x50	$Security_Descriptor Contains the access control list (ACL) for the file.

*Basic information of a file in MFT starts at **0x10** after the MFT header (after 56 byte)*

MFT and File Attributes (Cont)

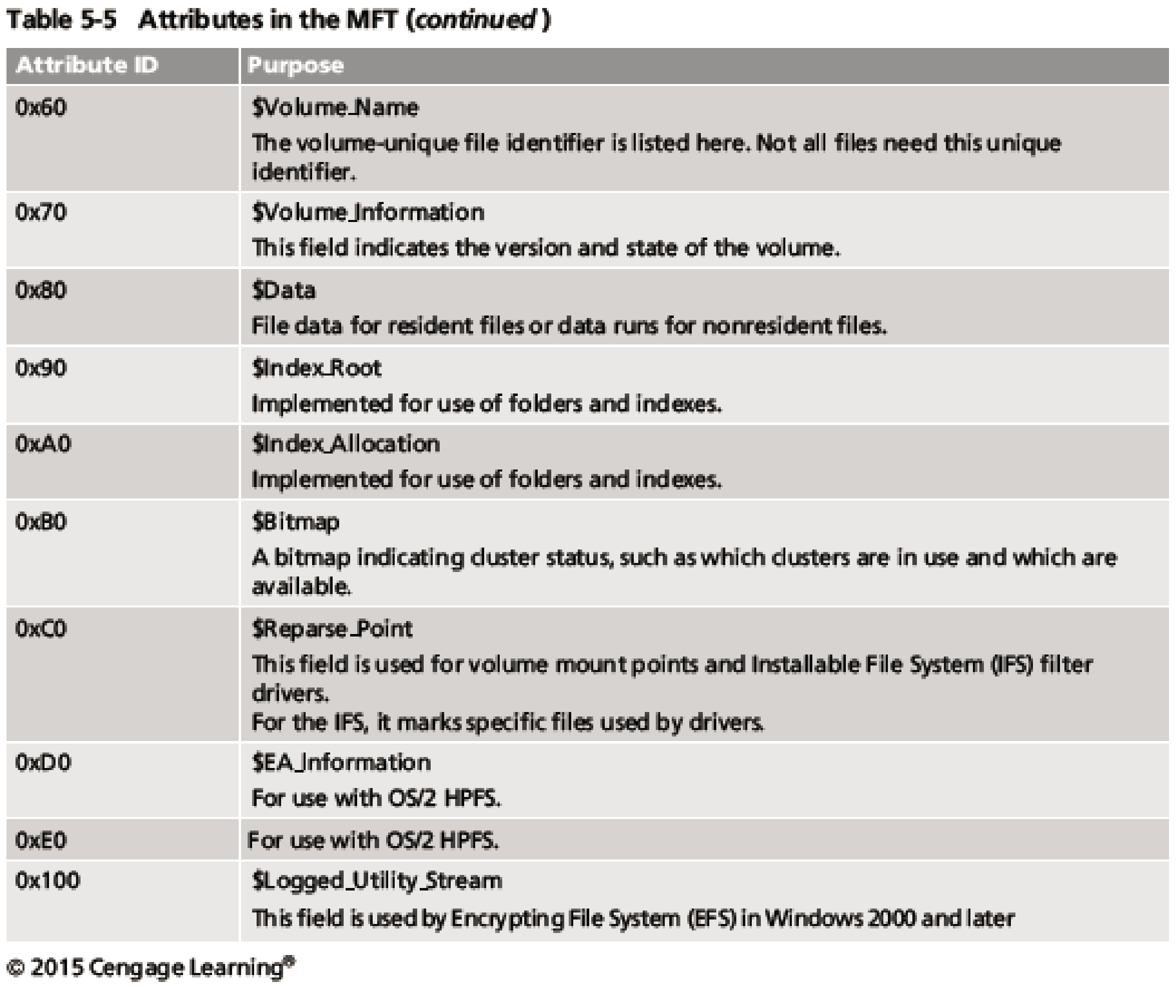
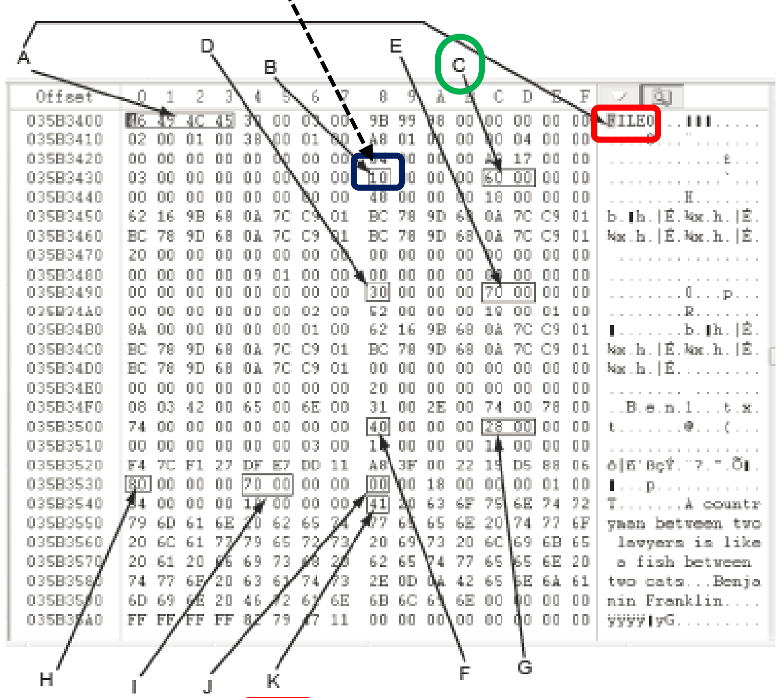


Table 5-5 Attributes in the MFT (*continued*)

Attribute ID	Purpose
0x60	$Volume_Name The volume-unique file identifier is listed here. Not all files need this unique identifier.
0x70	$Volume_Information This field indicates the version and state of the volume.
0x80	$Data File data for resident files or data runs for nonresident files.
0x90	$Index_Root Implemented for use of folders and indexes.
0xA0	$Index_Allocation Implemented for use of folders and indexes.
0xB0	$Bitmap A bitmap indicating cluster status, such as which clusters are in use and which are available.
0xC0	$Reparse_Point This field is used for volume mount points and Installable File System (IFS) filter drivers. For the IFS, it marks specific files used by drivers.
0xD0	$EA_Information For use with OS/2 HPFS.
0xE0	For use with OS/2 HPFS.
0x100	$Logged_Utility_Stream This field is used by Encrypting File System (EFS) in Windows 2000 and later

MFT and File Attributes (Cont)

Start of attribute **0x10** after 56 bytes header info



```
Offset     0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
035B3400  46 49 4C 45 30 00 03 00  9B 99 98 00 00 00 00 00  FILE0..▮▮▮.....
035B3410  02 00 01 00 38 00 01 00  A8 01 00 00 00 04 00 00  ....8...".......
035B3420  00 00 00 00 00 00 00 00  04 00 00 00 A3 17 00 00  ............£...
035B3430  03 00 00 00 00 00 00 00  10 00 00 00 60 00 00 00  ............`...
035B3440  00 00 00 00 00 00 00 00  48 00 00 00 18 00 00 00  ........H.......
035B3450  62 16 9B 68 0A 7C C9 01  BC 78 9D 68 0A 7C C9 01  b.▮h.|É.¼x.h.|É.
035B3460  BC 78 9D 68 0A 7C C9 01  BC 78 9D 68 0A 7C C9 01  ¼x.h.|É.¼x.h.|É.
035B3470  20 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ...............
035B3480  00 00 00 00 09 01 00 00  00 00 00 00 00 00 00 00  ................
035B3490  00 00 00 00 00 00 00 00  30 00 00 00 70 00 00 00  ........0...p...
035B34A0  00 00 00 00 00 00 02 00  52 00 00 00 18 00 01 00  ........R.......
035B34B0  8A 00 00 00 00 00 01 00  62 16 9B 68 0A 7C C9 01  ▮.......b.▮h.|É.
035B34C0  BC 78 9D 68 0A 7C C9 01  BC 78 9D 68 0A 7C C9 01  ¼x.h.|É.¼x.h.|É.
035B34D0  BC 78 9D 68 0A 7C C9 01  00 00 00 00 00 00 00 00  ¼x.h.|É.........
035B34E0  00 00 00 00 00 00 00 00  20 00 00 00 00 00 00 00  ........ .......
035B34F0  08 03 42 00 65 00 6E 00  31 00 2E 00 74 00 78 00  ..B.e.n.1...t.x.
035B3500  74 00 00 00 00 00 00 00  40 00 00 00 28 00 00 00  t.......@...(...
035B3510  00 00 00 00 00 00 03 00  10 00 00 00 18 00 00 00  ................
035B3520  F4 7C F1 27 DF E7 DD 11  A8 3F 00 22 15 D5 88 06  ô|ñ'ßçÝ.¨?."?Õ▮.
035B3530  80 00 00 00 70 00 00 00  00 00 18 00 00 00 01 00  ▮...p...........
035B3540  54 00 00 00 18 00 00 00  41 20 63 6F 75 6E 74 72  T.......A countr
035B3550  79 6D 61 6E 20 62 65 74  77 65 65 6E 20 74 77 6F  yman between two
035B3560  20 6C 61 77 79 65 72 73  20 69 73 20 6C 69 6B 65   lawyers is like
035B3570  20 61 20 66 69 73 68 20  62 65 74 77 65 65 6E 20   a fish between 
035B3580  74 77 6F 20 63 61 74 73  2E 0D 0A 42 65 6E 6A 61  two cats...Benja
035B3590  6D 69 6E 20 46 72 61 6E  6B 6C 69 6E 00 00 00 00  min Franklin....
035B35A0  FF FF FF FF 82 79 47 11  00 00 00 00 00 00 00 00  ÿÿÿÿ▮yG.........
```

A: All MFT records start with FILE0
B: Start of attribute 0x10
C: Length of attribute 0x10 (value 60)
D: Start of attribute 0x30
E: Length of attribute 0x30 (value 70)
F: Start of attribute 0x40
G: Length of attribute 0x40 (value 28)
H: Start of attribute 0x80
I : Length of attribute 0x80 (value 70)
J : Attribute 0x80 resident flag
K: Starting position of resident data

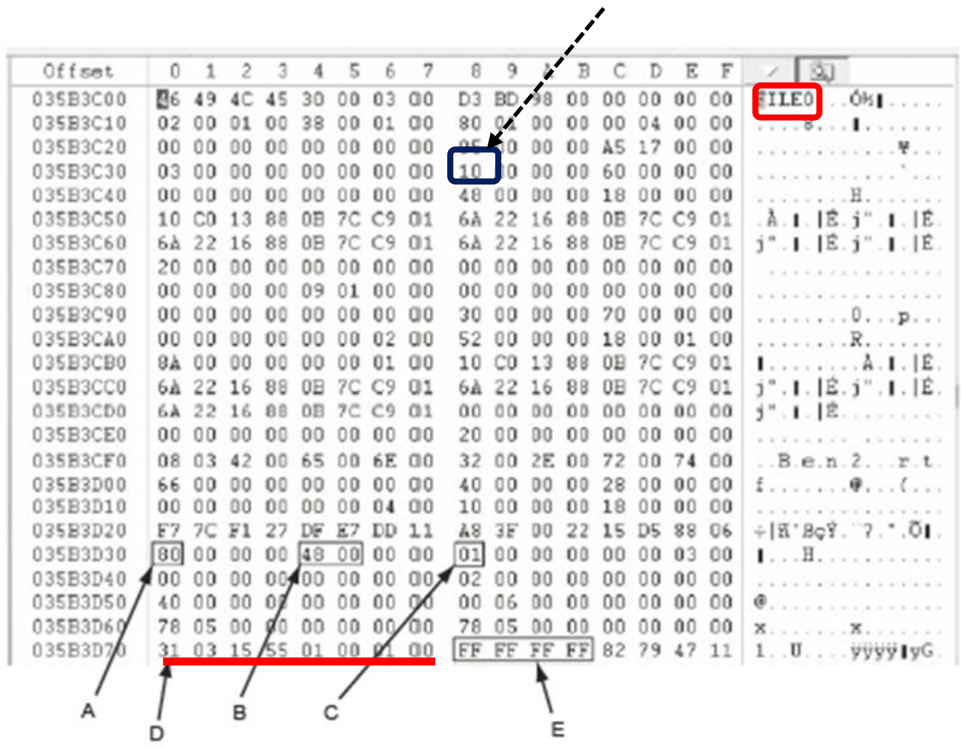
60 00 → 00 60 (HEX) => 96 (DEC)

70 00 → 00 70

Figure 5-10 Resident file in an MFT record
Courtesy of X-Ways AG, *www.x-ways.net*

Resident file attributes

Start of attribute **0x10** after 56 bytes header info



```
Offset     0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
035B3C00  46 49 4C 45 30 00 03 00  D3 BD 98 00 00 00 00 00  FILE0...ÓЅ▮.....
035B3C10  02 00 01 00 38 00 01 00  80 01 00 00 00 04 00 00  ....8...▮.......
035B3C20  00 00 00 00 00 00 00 00  05 00 00 00 A5 17 00 00  ............¥...
035B3C30  03 00 00 00 00 00 00 00  10 00 00 00 60 00 00 00  ............`...
035B3C40  00 00 00 00 00 00 00 00  48 00 00 00 18 00 00 00  ........H.......
035B3C50  10 C0 13 88 0B 7C C9 01  6A 22 16 88 0B 7C C9 01  .À.▮.|É.j".▮.|É.
035B3C60  6A 22 16 88 0B 7C C9 01  6A 22 16 88 0B 7C C9 01  j".▮.|É.j".▮.|É.
035B3C70  20 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ...............
035B3C80  00 00 00 00 09 01 00 00  00 00 00 00 00 00 00 00  ................
035B3C90  00 00 00 00 00 00 00 00  30 00 00 00 70 00 00 00  ........0...p...
035B3CA0  00 00 00 00 00 00 02 00  52 00 00 00 18 00 01 00  ........R.......
035B3CB0  8A 00 00 00 00 00 01 00  10 C0 13 88 0B 7C C9 01  ▮........À.▮.|É.
035B3CC0  6A 22 16 88 0B 7C C9 01  6A 22 16 88 0B 7C C9 01  j".▮.|É.j".▮.|É.
035B3CD0  6A 22 16 88 0B 7C C9 01  00 00 00 00 00 00 00 00  j".▮.|É.........
035B3CE0  00 00 00 00 00 00 00 00  20 00 00 00 00 00 00 00  ........ .......
035B3CF0  08 03 42 00 65 00 6E 00  32 00 2E 00 72 00 74 00  ..B.e.n.2...r.t.
035B3D00  66 00 00 00 00 00 00 00  40 00 00 00 28 00 00 00  f.......@...(...
035B3D10  00 00 00 00 00 00 04 00  10 00 00 00 18 00 00 00  ................
035B3D20  F7 7C F1 27 DF E7 DD 11  A8 3F 00 22 15 D5 88 06  ÷|ñ'ßçÝ.¨?."?Õ▮.
035B3D30  80 00 00 00 48 00 00 00  01 00 00 00 00 00 03 00  ▮...H...........
035B3D40  00 00 00 00 00 00 00 00  02 00 00 00 00 00 00 00  ................
035B3D50  40 00 00 00 00 00 00 00  00 06 00 00 00 00 00 00  @...............
035B3D60  78 05 00 00 00 00 00 00  78 05 00 00 00 00 00 00  x.......x.......
035B3D70  31 03 15 55 01 00 01 00  FF FF FF FF 82 79 47 11  1..U....ÿÿÿÿ▮yG.
```

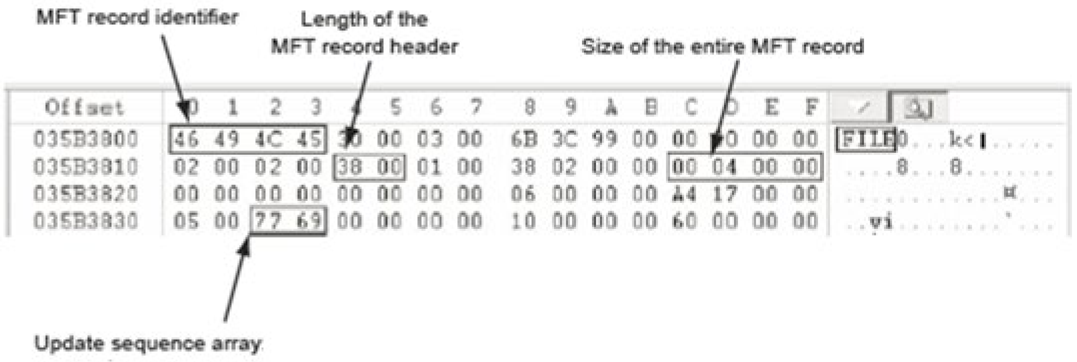
A: Start of nonresident attribute 0x80
B: Length of nonresident attribute 0x80
C: Attribute 0x80 nonresident flag
D: Starting point of data run
E: End-of-record marker (FF FF FF FF) for the MFT record

Figure 5-12 Nonresident file in an MFT record
Courtesy of X-Ways AG, *www.x-ways.net*

Non-resident file attributes

MFT Structures for File Data

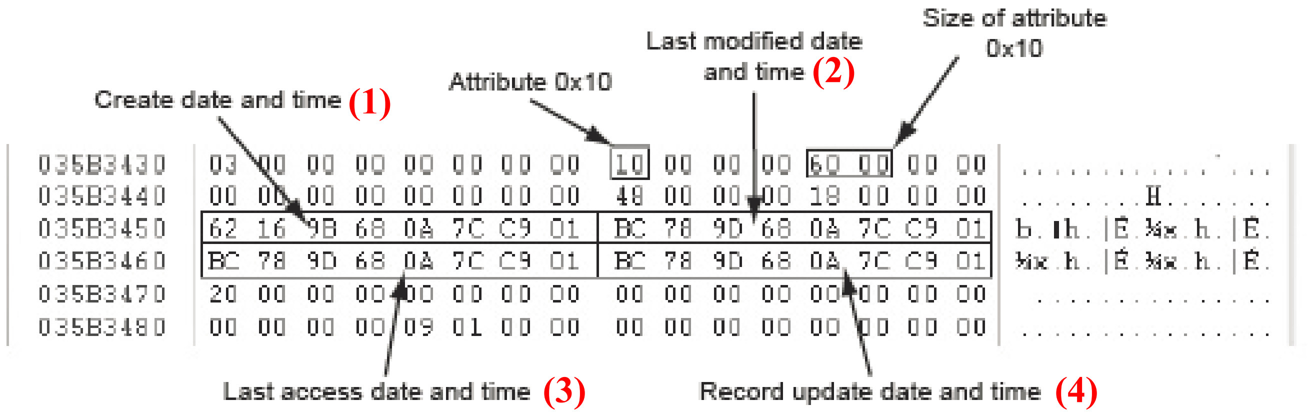
- For the **header** of all MFT records, the record fields of interest are as follows:
  - *At offset 0x00* - the MFT record identifier **"FILE"**
  - *At offset 0x14* - length of the header (indicates where the next attribute starts) *38 00 → 00 38* = ***56 bytes****!!*
  - *At offset 0x1C to 0x1F* - size of the entire MFT record
  - *At offset 0x32 and 0x33* - the update sequence array, which stores the last 2 bytes of the first sector of the MFT record
    - *The update sequence array is used as a checksum for record integrity validation*


MFT record identifier
Length of the MFT record header
Size of the entire MFT record
Update sequence array

```
Offset     0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
035B3800  46 49 4C 45 30 00 03 00  6B 3C 99 00 00 00 00 00  FILE0...k<......
035B3810  02 00 02 00 38 00 01 00  38 02 00 00 00 04 00 00  ....8...8.......
035B3820  00 00 00 00 00 00 00 00  06 00 00 00 A4 17 00 00  ............¤...
035B3830  05 00 77 69 00 00 00 00  10 00 00 00 60 00 00 00  ..wi........`...
```

Figure 5-13 An MFT header
Courtesy of X-Ways AG, *www.x-ways.net*

MFT Structures for File Data (cont)


Create date and time (1)
Attribute 0x10
Last modified date and time (2)
Size of attribute 0x10
Last access date and time (3)
Record update date and time (4)

```
035B3430  03 00 00 00 00 00 00 00  10 00 00 00 60 00 00 00  ............`...
035B3440  00 00 00 00 00 00 00 00  48 00 00 00 18 00 00 00  ........H.......
035B3450  62 16 9B 68 0A 7C C9 01  BC 78 9D 68 0A 7C C9 01  b.›h.|É.¼x.h.|É.
035B3460  BC 78 9D 68 0A 7C C9 01  BC 78 9D 68 0A 7C C9 01  ¼x.h.|É.¼x.h.|É.
035B3470  20 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ...............
035B3480  00 00 00 00 09 01 00 00  00 00 00 00 00 00 00 00  ................
```

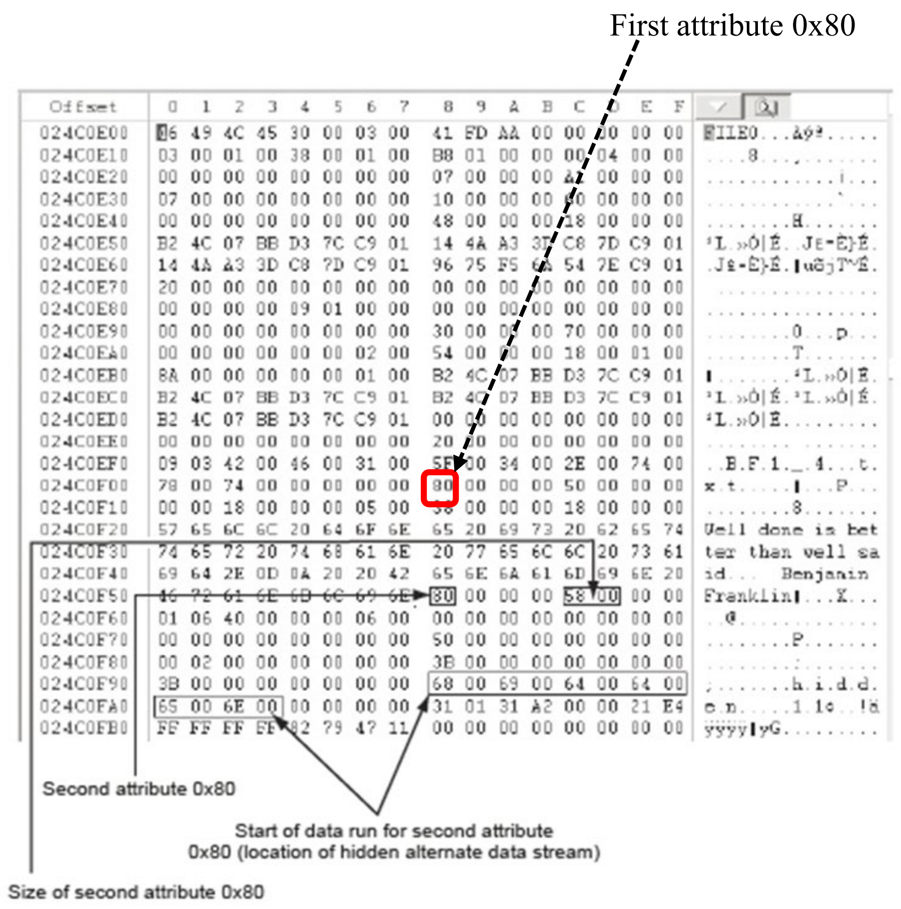
Figure 5-14 **Attribute 0x10: Standard Information**
Courtesy of X-Ways AG, *www.x-ways.net*

Standard Information attribute –

- *Create date and time* **(1)**
- *Last modified date and time* **(2)**
- *Last access date and time* **(3)**
- *Record update date and time* **(4)**

NTFS Alternate Data Streams

- **Alternate data streams**
  - Ways data can be appended to existing files
  - Can obscure valuable evidentiary data, intentionally or by coincidence
- In NTFS, an alternate data stream becomes an additional file attribute
  - Allows the file to be associated with different applications
- You can only tell whether a file has a data stream attached by examining that file's MFT entry!!


First attribute 0x80
Second attribute 0x80
Start of data run for second attribute 0x80 (location of hidden alternate data stream)
Size of second attribute 0x80

```
Offset      0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
024C0E00   46 49 4C 45 30 00 03 00  41 FD AA 00 00 00 00 00   FILE0...Aýª.....
024C0E10   03 00 01 00 38 00 01 00  B8 01 00 00 00 04 00 00   ....8...¸.......
024C0E20   00 00 00 00 00 00 00 00  07 00 00 00 A1 00 00 00   ............¡...
024C0E30   07 00 00 00 00 00 00 00  10 00 00 00 60 00 00 00   ............`...
024C0E40   00 00 00 00 00 00 00 00  48 00 00 00 18 00 00 00   ........H.......
024C0E50   B2 4C 07 BB D3 7C C9 01  14 4A A3 3D C8 7D C9 01   ²L.»Ó|É..J£=È}É.
024C0E60   14 4A A3 3D C8 7D C9 01  96 75 F5 6A 54 7E C9 01   .J£=È}É.–uõjT~É.
024C0E70   20 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
024C0E80   00 00 00 00 09 01 00 00  00 00 00 00 00 00 00 00   ................
024C0E90   00 00 00 00 00 00 00 00  30 00 00 00 70 00 00 00   ........0...p...
024C0EA0   00 00 00 00 00 00 02 00  54 00 00 00 18 00 01 00   ........T.......
024C0EB0   8A 00 00 00 00 00 01 00  B2 4C 07 BB D3 7C C9 01   Š.......²L.»Ó|É.
024C0EC0   B2 4C 07 BB D3 7C C9 01  B2 4C 07 BB D3 7C C9 01   ²L.»Ó|É.²L.»Ó|É.
024C0ED0   B2 4C 07 BB D3 7C C9 01  00 00 00 00 00 00 00 00   ²L.»Ó|É.........
024C0EE0   00 00 00 00 00 00 00 00  20 00 00 00 00 00 00 00   ................
024C0EF0   09 03 42 00 46 00 31 00  5F 00 34 00 2E 00 74 00   ..B.F.1._.4...t.
024C0F00   78 00 74 00 00 00 00 00  80 00 00 00 50 00 00 00   x.t.....€...P...
024C0F10   00 00 18 00 00 00 05 00  38 00 00 00 18 00 00 00   ........8.......
024C0F20   57 65 6C 6C 20 64 6F 6E  65 20 69 73 20 62 65 74   Well done is bet
024C0F30   74 65 72 20 74 68 61 6E  20 77 65 6C 6C 20 73 61   ter than well sa
024C0F40   69 64 2E 0D 0A 20 20 42  65 6E 6A 61 6D 69 6E 20   id...  Benjamin 
024C0F50   46 72 61 6E 6B 6C 69 6E  80 00 00 00 58 00 00 00   Franklin€...X...
024C0F60   01 06 40 00 00 00 06 00  00 00 00 00 00 00 00 00   ..@.............
024C0F70   00 00 00 00 00 00 00 00  50 00 00 00 00 00 00 00   ........P.......
024C0F80   00 02 00 00 00 00 00 00  3B 00 00 00 00 00 00 00   ........;.......
024C0F90   3B 00 00 00 00 00 00 00  68 00 69 00 64 00 64 00   ;.......h.i.d.d.
024C0FA0   65 00 6E 00 00 00 00 00  31 01 31 A2 00 00 21 E4   e.n.....1.1¢..!ä
024C0FB0   FF FF FF FF 82 79 47 11  00 00 00 00 00 00 00 00   ÿÿÿÿ‚yG.........
```

Figure 5-24 A text alternate data stream
Courtesy of X-Ways AG, *www.x-ways.net*