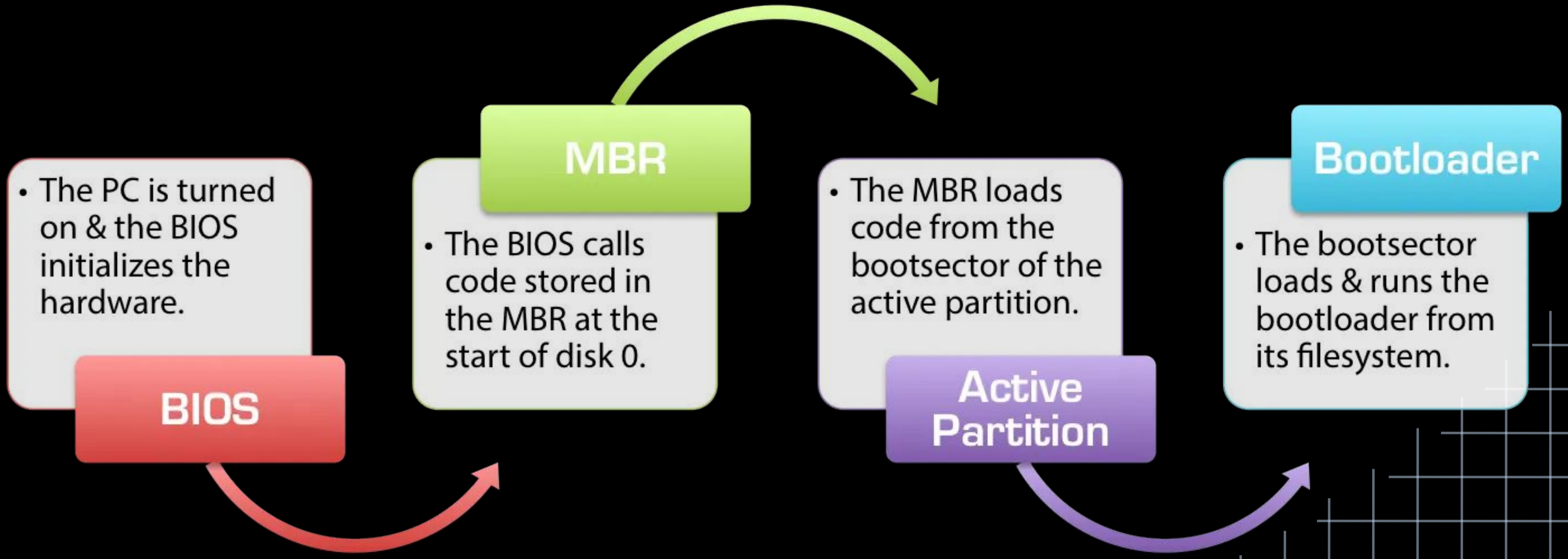
FORFUN

Week 2 NTFS File System

Soraya Harding & Rahim Taheri

What happens when we start our computer?



BIOS

- The PC is turned on & the BIOS initializes the hardware.

MBR

- The BIOS calls code stored in the MBR at the start of disk 0.

Active Partition

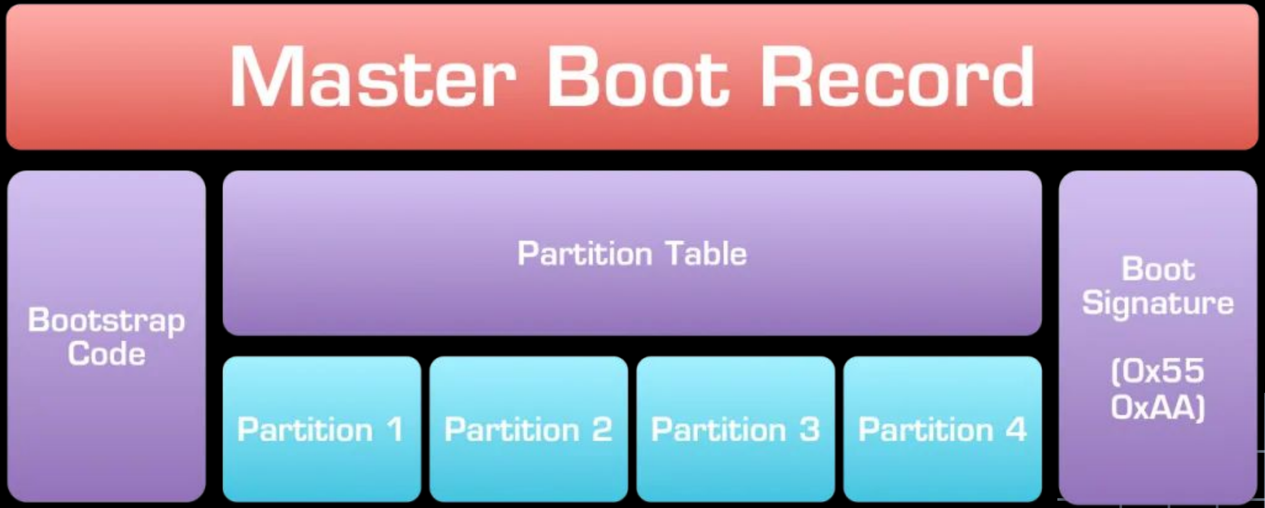
- The MBR loads code from the bootsector of the active partition.

Bootloader

- The bootsector loads & runs the bootloader from its filesystem.

Master Boot Record

- Master Boot Record is a special type of boot sector at the very beginning of partitioned computer storage devices;
- It contains executable code that the system BIOS loads into memory.
- The code scans the MBR to find the partition table to determine which partition is the active, or bootable.
- Boot signature: 0x55AA



Partition Table

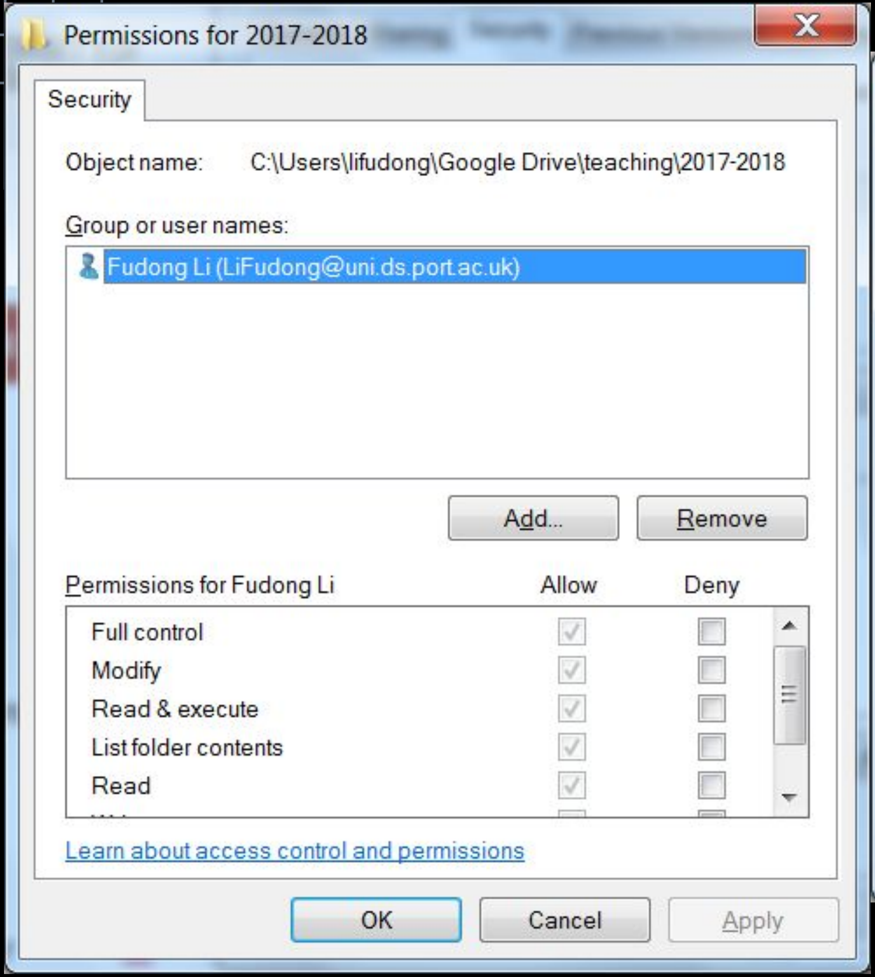
- Information of 4 primary partitions are stored in the partition table and each record contains:
  - 1st byte: 0x80 bootable/active, 0x00 inactive
  - 2-4 bytes: Cylinder-Head-Sector (CHS) of first absolute sector in partition
  - 5th byte: partition type (0x0E: FAT 16; 0x0C; FAT 32; 0x07 NTFS)
  - 6-8 bytes: CHS address of last absolute sector in partition.
  - 9-12 bytes: Logical block addressing of first absolute sector in the partition
  - 13-16 bytes: Number of sectors in partition

NTFS

NTFS – Overview

- NTFS is a proprietary file system developed by Microsoft in 1993; default file system of Windows NT family
- Notable features of NTFS
  - Security: by using an Access Control List (ACL), an administrator controls who can access specific files.
  - Encryption: Encryption File System (EFS) provides strong and user-transparent encryption of any files or folder on an NTFS volume
  - Performance : B-tree- faster file look up times
  - Journaling: records a transaction before the system carries it out
  - Support large file sizes: up to 16 exbibytes (2GB for FAT16 and 4GB for FAT32)

NTFS – Security



Permissions for 2017-2018

Security

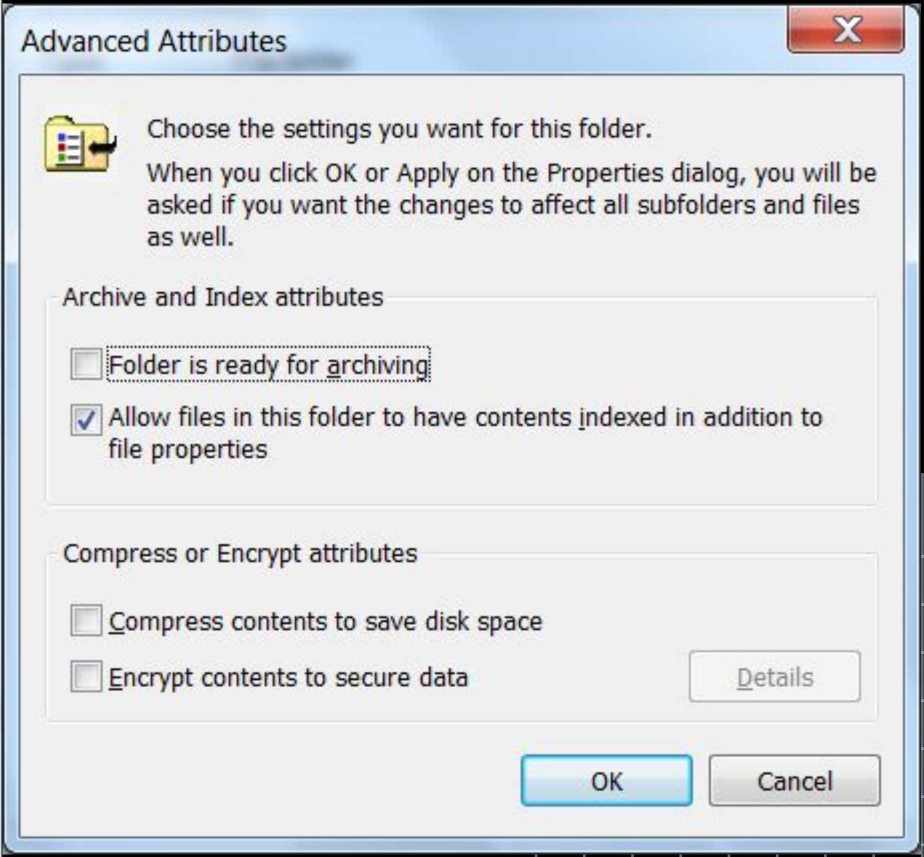
Object name: C:\Users\lifudong\Google Drive\teaching\2017-2018

Group or user names:

Fudong Li (LiFudong@uni.ds.port.ac.uk)

Permissions for Fudong Li	Allow	Deny
Full control	☑	□
Modify	☑	□
Read & execute	☑	□
List folder contents	☑	□
Read	☑	□

Learn about access control and permissions



Advanced Attributes

Choose the settings you want for this folder.

When you click OK or Apply on the Properties dialog, you will be asked if you want the changes to affect all subfolders and files as well.

Archive and Index attributes

□ Folder is ready for archiving

☑ Allow files in this folder to have contents indexed in addition to file properties

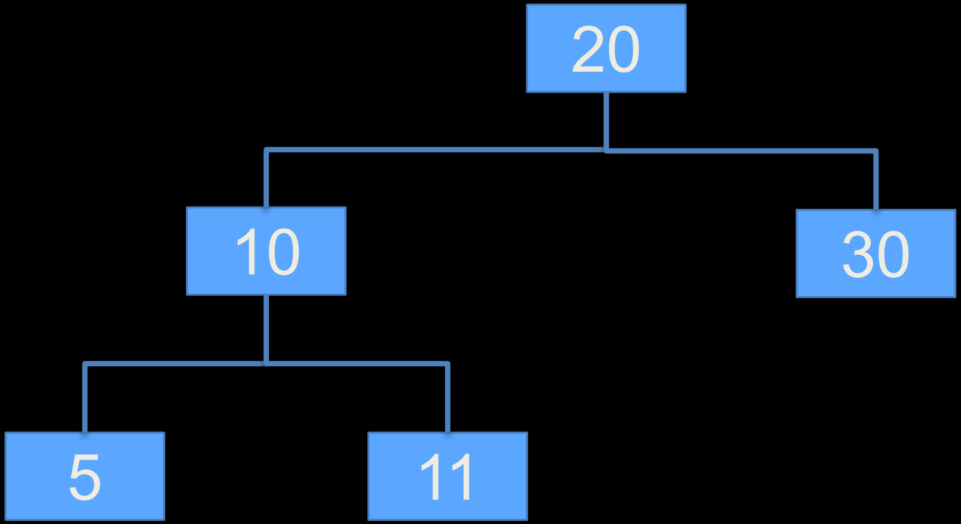
Compress or Encrypt attributes

□ Compress contents to save disk space

□ Encrypt contents to secure data

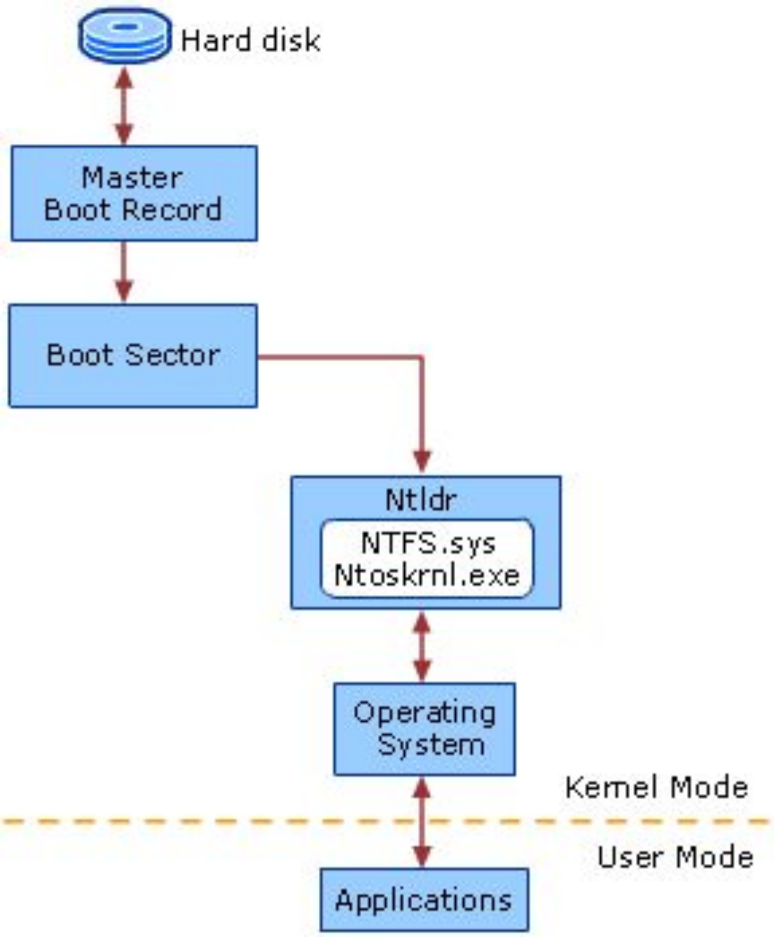
B-tree

- A B-tree is a method of placing and locating files in a file system. It minimises the number of times a medium must be accessed to locate a desired record, hence speeding up the process.



B-tree

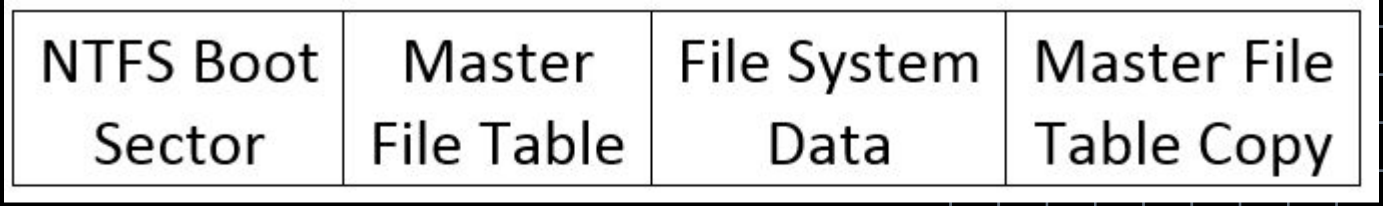
NTFS - Architecture



Source: NTFS Technical Reference – How NTFS works
https://technet.microsoft.com/en-us/library/cc781134(v=ws.10).aspx

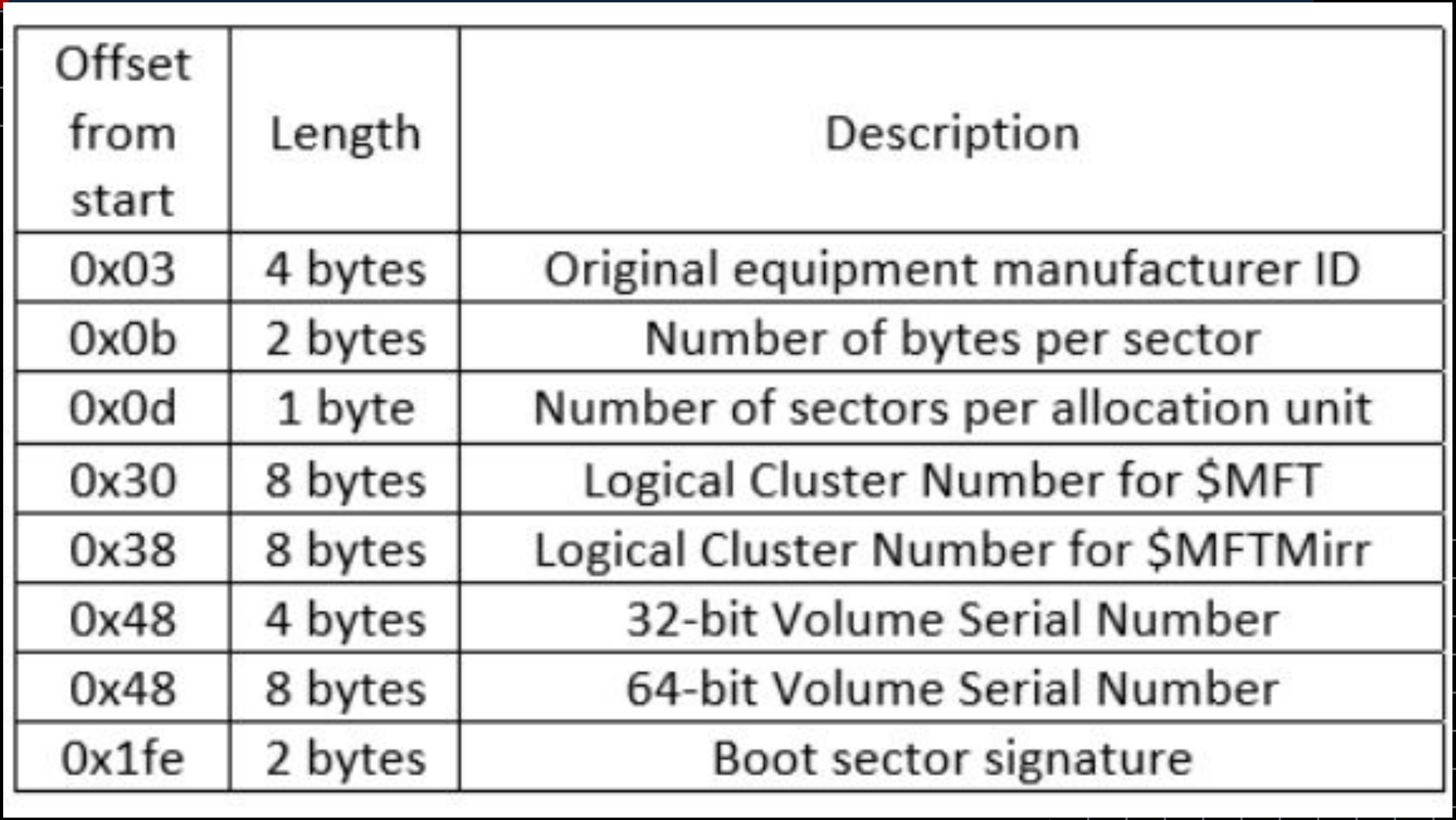
NTFS Partition Organization

- NTFS Boot Sector
  - Contains the BIOS parameter block that stores information about the layout of the volume and the file system structures.
- Master File Table
  - Contains the information necessary to retrieve files from the NTFS partition, such as the attributes of a file
- File System Data
  - Stores data that is not contained within the Master File Table
- Master File Table Copy
  - Includes copies of the records essential for the recovery of the file system if there is a problem with the original copy



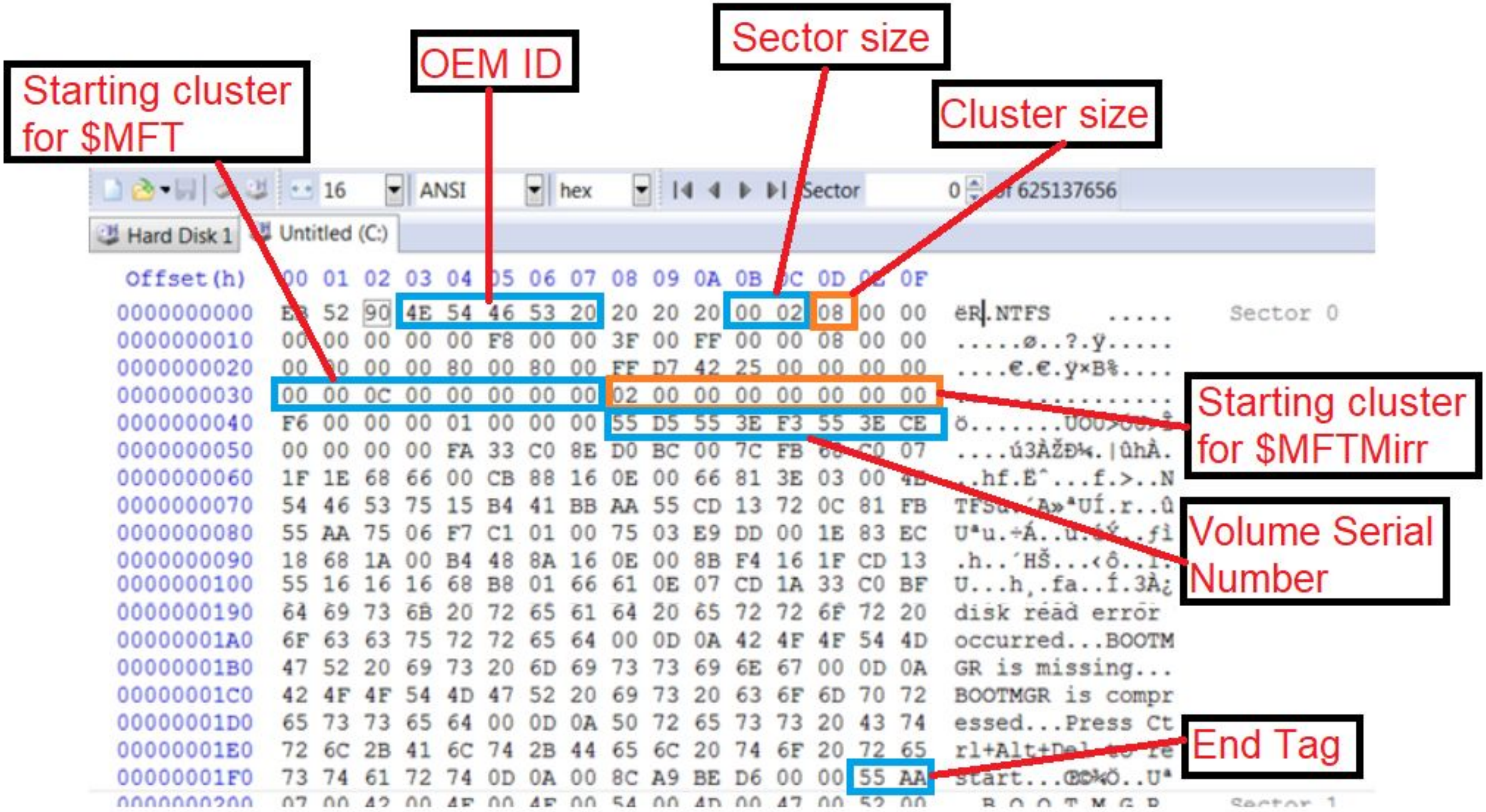
NTFS Boot Sector	Master File Table	File System Data	Master File Table Copy

NTFS Boot Sector



Offset from start	Length	Description
0x03	4 bytes	Original equipment manufacturer ID
0x0b	2 bytes	Number of bytes per sector
0x0d	1 byte	Number of sectors per allocation unit
0x30	8 bytes	Logical Cluster Number for $MFT
0x38	8 bytes	Logical Cluster Number for $MFTMirr
0x48	4 bytes	32-bit Volume Serial Number
0x48	8 bytes	64-bit Volume Serial Number
0x1fe	2 bytes	Boot sector signature

NTFS Boot Sector



Starting cluster for $MFT

OEM ID

Sector size

Cluster size

Starting cluster for $MFTMirr

Volume Serial Number

End Tag

```
Offset(h)   00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

0000000000  EB 52 90 4E 54 46 53 20 20 20 20 00 02 08 00 00  ëR.NTFS    .....     Sector 0
0000000010  00 00 00 00 00 F8 00 00 3F 00 FF 00 00 08 00 00  .....ø..?.ÿ.....
0000000020  00 00 00 00 80 00 80 00 FF D7 42 25 00 00 00 00  ....€.€.ÿ×B%....
0000000030  00 00 0C 00 00 00 00 00 02 00 00 00 00 00 00 00  ................
0000000040  F6 00 00 00 01 00 00 00 55 D5 55 3E F3 55 3E CE  ö.......UÕU>óU>Î
0000000050  00 00 00 00 FA 33 C0 8E D0 BC 00 7C FB 68 C0 07  ....ú3ÀŽÐ¼.|ûhÀ.
0000000060  1F 1E 68 66 00 CB 88 16 0E 00 66 81 3E 03 00 4E  ..hf.Ëˆ...f.>..N
0000000070  54 46 53 75 15 B4 41 BB AA 55 CD 13 72 0C 81 FB  TFSu.´A»ªUÍ.r..û
0000000080  55 AA 75 06 F7 C1 01 00 75 03 E9 DD 00 1E 83 EC  Uªu.÷Á..u.éÝ..ƒì
0000000090  18 68 1A 00 B4 48 8A 16 0E 00 8B F4 16 1F CD 13  .h..´HŠ...‹ô..Í.
0000000100  55 16 16 16 68 B8 01 66 61 0E 07 CD 1A 33 C0 BF  U...h¸.fa..Í.3À¿
0000000190  64 69 73 6B 20 72 65 61 64 20 65 72 72 6F 72 20  disk read error
00000001A0  6F 63 63 75 72 72 65 64 00 0D 0A 42 4F 4F 54 4D  occurred...BOOTM
00000001B0  47 52 20 69 73 20 6D 69 73 73 69 6E 67 00 0D 0A  GR is missing...
00000001C0  42 4F 4F 54 4D 47 52 20 69 73 20 63 6F 6D 70 72  BOOTMGR is compr
00000001D0  65 73 73 65 64 00 0D 0A 50 72 65 73 73 20 43 74  essed...Press Ct
00000001E0  72 6C 2B 41 6C 74 2B 44 65 6C 20 74 6F 20 72 65  rl+Alt+Del to re
00000001F0  73 74 61 72 74 0D 0A 00 8C A9 BE D6 00 00 55 AA  start...Œ©¾Ö..Uª
0000000200  07 00 42 00 4F 00 4F 00 54 00 4D 00 47 00 52 00   B O O T M G R        Sector 1
```

Master File Table

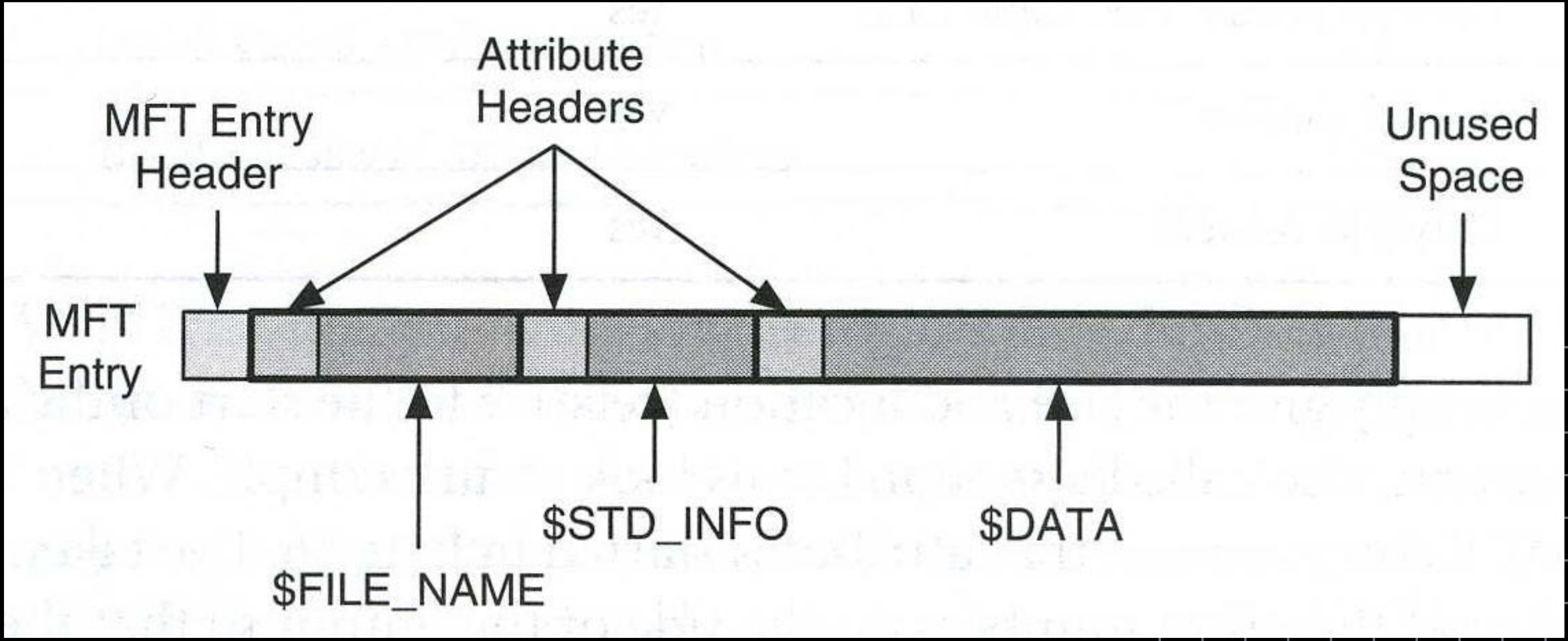
- Each file on an NTFS volume is represented by a record in a special file called the master file table (MFT)
- Starting location of the MFT is given in the boot sector;
- 12.5% of space allocated but only used when necessary
- Each entry is 1024 bytes (1Kibi Bytes)
  - Only first 42 bytes defined, containing 12 fields
  - The rest are allocated to numerous/various attributes
- 0x46494C45 (FILE): beginning of each record
- 0xFFFFFFFF: end of record marker for each record

$MFT Entries

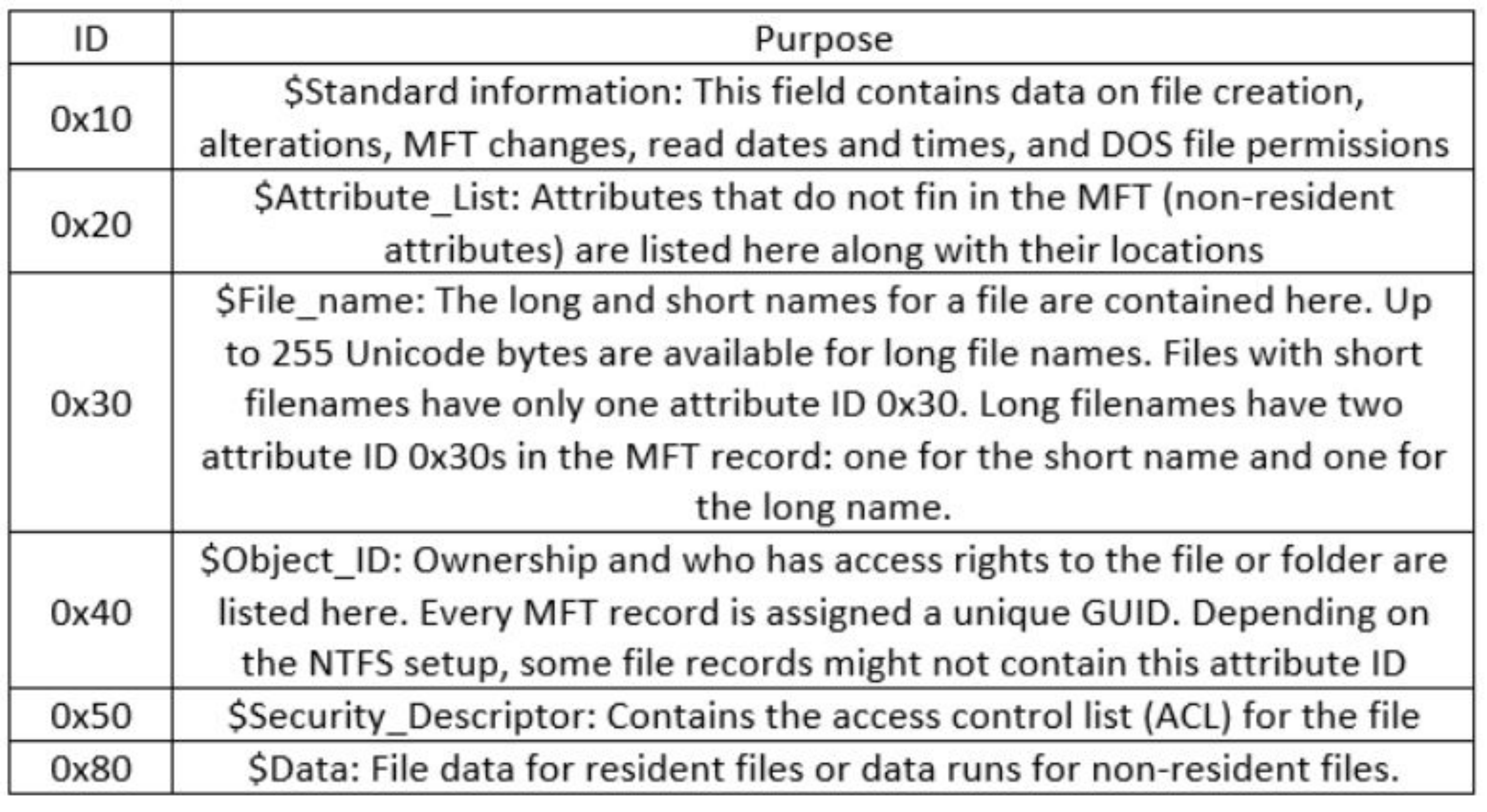
Entry	Filename	Description
0	$MFT	The entry for the MFT itself
1	$MFTMirr	Backup of the MFT
2	$LogFile	Journal containing records of metadata transactions
3	$Volume	Volume information
4	$AttrDef	Attribute information (identifier values, name)
5	$.	Root directory of the file system
6	$Bitmap	Allocation status of each cluster in the file system
7	$Boot	Boot sector and boot code for the file system
8	$BadClus	Clusters that have bad sectors
9	$Secure	Security and access control for the files
10	$Upcase	Contains the uppercase version of every Unicode character
11	$Extend	Directory containing files for optional extensions

Single File Record in $MFT

- NTFS reads attributes from the record – not files – files are simply one of the attributes

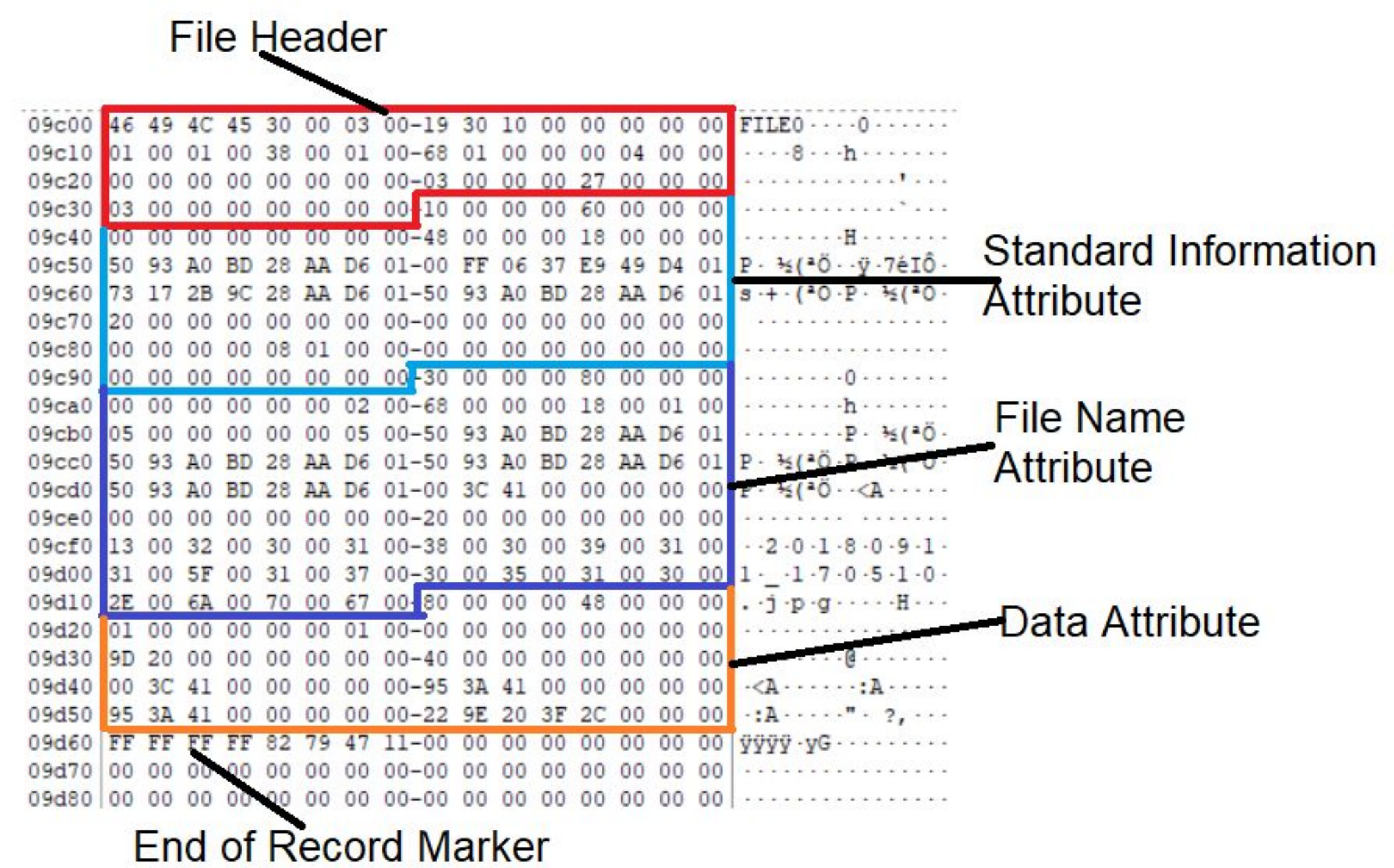


$MFT Entry Attribute Types



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 do not fin in the MFT (non-resident 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 file names. 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: Ownership and who has access rights to the file or folder are listed here. Every MFT record is assigned a unique GUID. Depending on the NTFS setup, some file records might not contain this attribute ID
0x50	$Security_Descriptor: Contains the access control list (ACL) for the file
0x80	$Data: File data for resident files or data runs for non-resident files.

$MFT Entry Attribute Types



File Header

```
09c00 46 49 4C 45 30 00 03 00-19 30 10 00 00 00 00 00 FILE0····0······
09c10 01 00 01 00 38 00 01 00-68 01 00 00 00 04 00 00 ····8···h·······
09c20 00 00 00 00 00 00 00 00-03 00 00 00 27 00 00 00 ············'···
09c30 03 00 00 00 00 00 00 00-10 00 00 00 60 00 00 00 ············`···
09c40 00 00 00 00 00 00 00 00-48 00 00 00 18 00 00 00 ········H·······
09c50 50 93 A0 BD 28 AA D6 01-00 FF 06 37 E9 49 D4 01 P· ½(ªÖ··ÿ·7éIÔ·
09c60 73 17 2B 9C 28 AA D6 01-50 93 A0 BD 28 AA D6 01 s·+·(ªÖ·P· ½(ªÖ·
09c70 20 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ···············
09c80 00 00 00 00 08 01 00 00-00 00 00 00 00 00 00 00 ················
09c90 00 00 00 00 00 00 00 00-30 00 00 00 80 00 00 00 ········0·······
09ca0 00 00 00 00 00 00 02 00-68 00 00 00 18 00 01 00 ········h·······
09cb0 05 00 00 00 00 00 05 00-50 93 A0 BD 28 AA D6 01 ········P· ½(ªÖ·
09cc0 50 93 A0 BD 28 AA D6 01-50 93 A0 BD 28 AA D6 01 P· ½(ªÖ·P· ½(ªÖ·
09cd0 50 93 A0 BD 28 AA D6 01-00 3C 41 00 00 00 00 00 P· ½(ªÖ··<A·····
09ce0 00 00 00 00 00 00 00 00-20 00 00 00 00 00 00 00 ········ ·······
09cf0 13 00 32 00 30 00 31 00-38 00 30 00 39 00 31 00 ··2·0·1·8·0·9·1·
09d00 31 00 5F 00 31 00 37 00-30 00 35 00 31 00 30 00 1·_·1·7·0·5·1·0·
09d10 2E 00 6A 00 70 00 67 00-80 00 00 00 48 00 00 00 .·j·p·g·····H···
09d20 01 00 00 00 00 00 01 00-00 00 00 00 00 00 00 00 ················
09d30 9D 20 00 00 00 00 00 00-40 00 00 00 00 00 00 00 · ······@·······
09d40 00 3C 41 00 00 00 00 00-95 3A 41 00 00 00 00 00 ·<A······:A·····
09d50 95 3A 41 00 00 00 00 00-22 9E 20 3F 2C 00 00 00 ·:A·····"· ?,···
09d60 FF FF FF FF 82 79 47 11-00 00 00 00 00 00 00 00 ÿÿÿÿ·yG·········
09d70 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ················
09d80 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ················
```

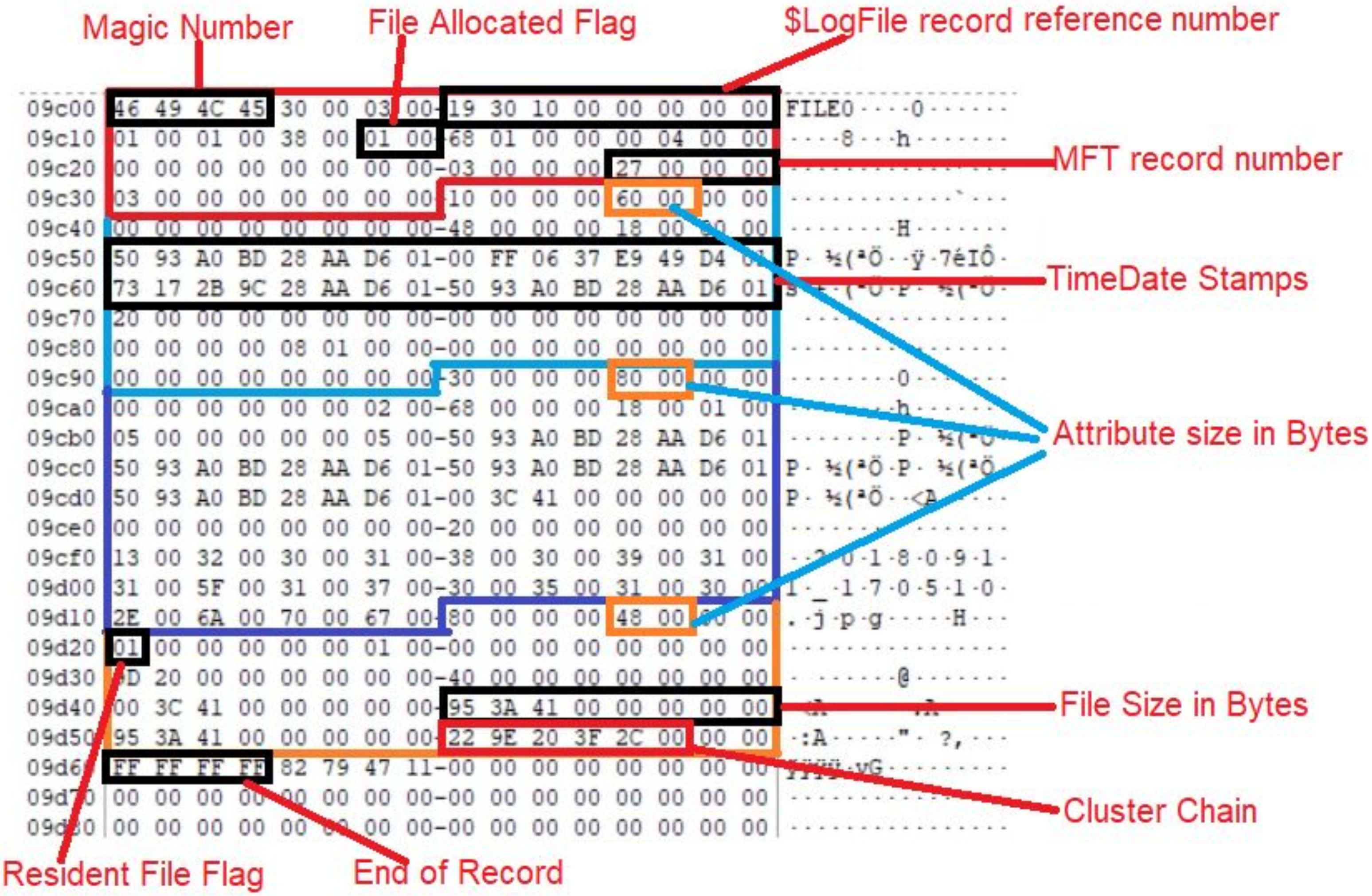
Standard Information Attribute

File Name Attribute

Data Attribute

End of Record Marker

$MFT Record



```
09c00 46 49 4C 45 30 00 03 00-19 30 10 00 00 00 00 00 FILE0····0······
09c10 01 00 01 00 38 00 01 00-68 01 00 00 00 04 00 00 ····8···h·······
09c20 00 00 00 00 00 00 00 00-03 00 00 00 27 00 00 00 ················
09c30 03 00 00 00 00 00 00 00-10 00 00 00 60 00 00 00 ············`···
09c40 00 00 00 00 00 00 00 00-48 00 00 00 18 00 00 00 ········H·······
09c50 50 93 A0 BD 28 AA D6 01-00 FF 06 37 E9 49 D4 01 P·½(ªÖ··ÿ·7éIÔ·
09c60 73 17 2B 9C 28 AA D6 01-50 93 A0 BD 28 AA D6 01 s·+·(ªÖ·P·½(ªÖ·
09c70 20 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ················
09c80 00 00 00 00 08 01 00 00-00 00 00 00 00 00 00 00 ················
09c90 00 00 00 00 00 00 00 00-30 00 00 00 80 00 00 00 ········0·······
09ca0 00 00 00 00 00 00 02 00-68 00 00 00 18 00 01 00 ········h·······
09cb0 05 00 00 00 00 00 05 00-50 93 A0 BD 28 AA D6 01 ········P·½(ªÖ·
09cc0 50 93 A0 BD 28 AA D6 01-50 93 A0 BD 28 AA D6 01 P·½(ªÖ·P·½(ªÖ·
09cd0 50 93 A0 BD 28 AA D6 01-00 3C 41 00 00 00 00 00 P·½(ªÖ··<A······
09ce0 00 00 00 00 00 00 00 00-20 00 00 00 00 00 00 00 ················
09cf0 13 00 32 00 30 00 31 00-38 00 30 00 39 00 31 00 ··2·0·1·8·0·9·1·
09d00 31 00 5F 00 31 00 37 00-30 00 35 00 31 00 30 00 1·_·1·7·0·5·1·0·
09d10 2E 00 6A 00 70 00 67 00-80 00 00 00 48 00 00 00 .·j·p·g·····H···
09d20 01 00 00 00 00 00 01 00-00 00 00 00 00 00 00 00 ················
09d30 9D 20 00 00 00 00 00 00-40 00 00 00 00 00 00 00 ········@·······
09d40 00 3C 41 00 00 00 00 00-95 3A 41 00 00 00 00 00 ·<A······:A·····
09d50 95 3A 41 00 00 00 00 00-22 9E 20 3F 2C 00 00 00 ·:A·····"· ?,···
09d60 FF FF FF FF 82 79 47 11-00 00 00 00 00 00 00 00 ÿÿÿÿ·yG·········
09d70 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ················
09d80 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ················
```

Magic Number

File Allocated Flag

$LogFile record reference number

MFT record number

TimeDate Stamps

Attribute size in Bytes

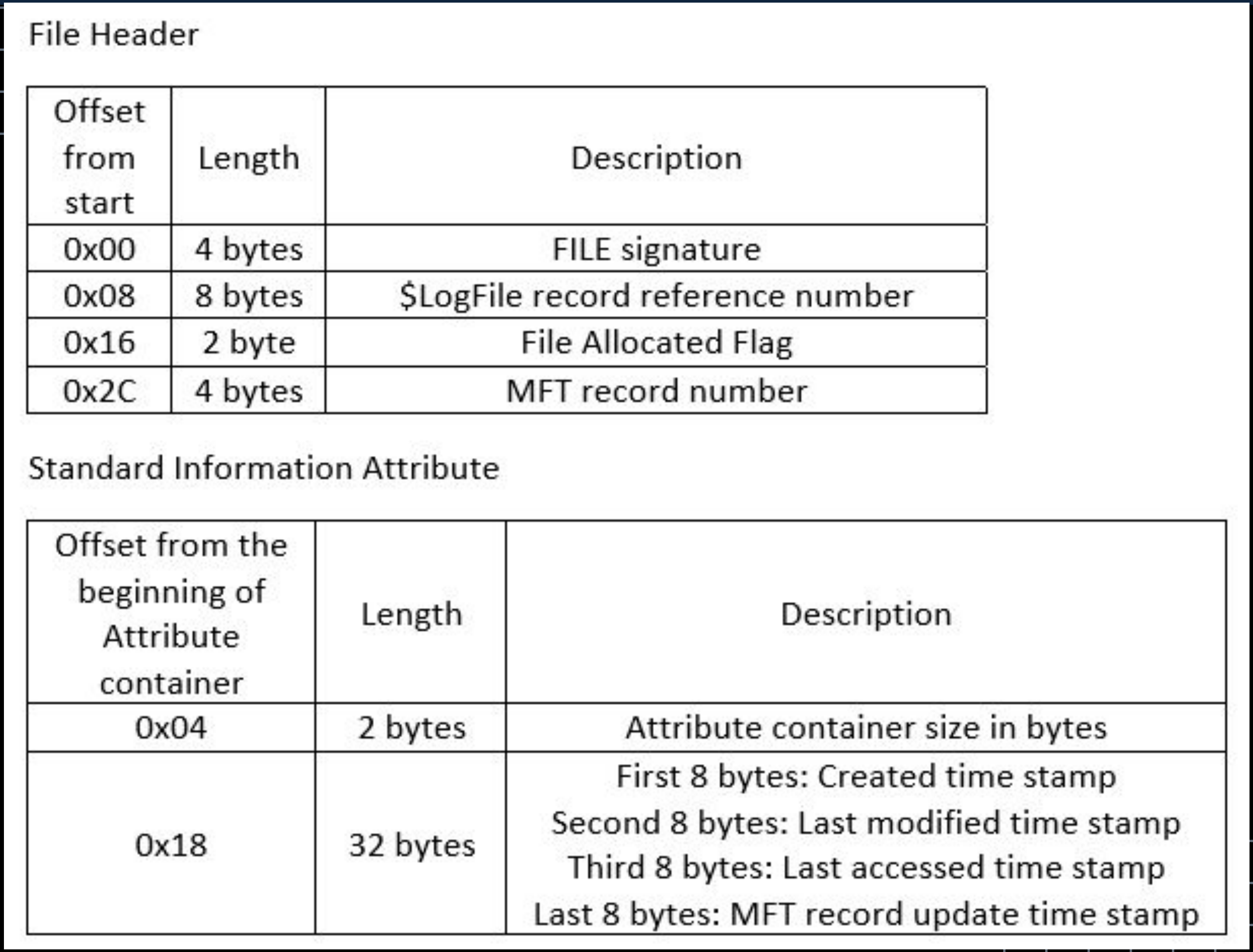
File Size in Bytes

Cluster Chain

Resident File Flag

End of Record

$MFT Record Offset Info



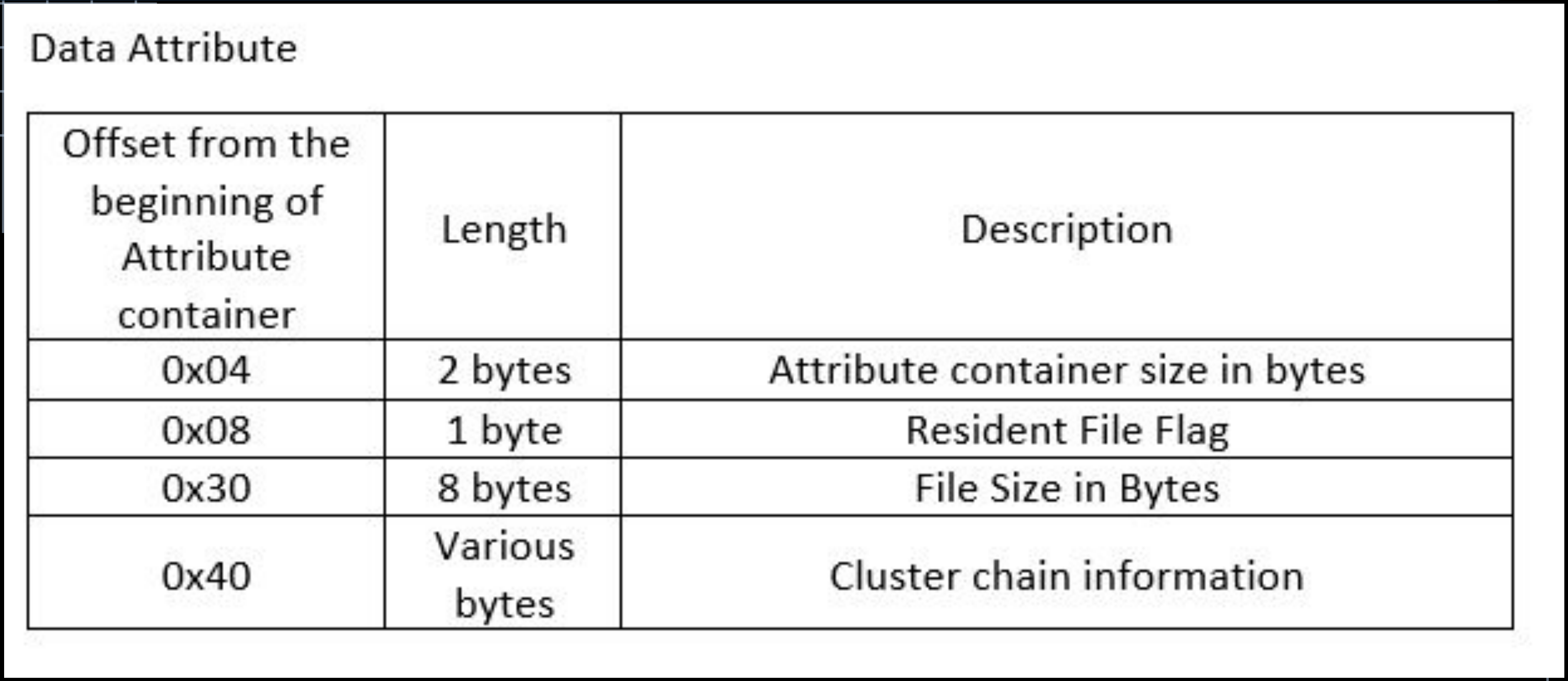
File Header

Offset from start	Length	Description
0x00	4 bytes	FILE signature
0x08	8 bytes	$LogFile record reference number
0x16	2 byte	File Allocated Flag
0x2C	4 bytes	MFT record number

Standard Information Attribute

Offset from the beginning of Attribute container	Length	Description
0x04	2 bytes	Attribute container size in bytes
0x18	32 bytes	First 8 bytes: Created time stamp Second 8 bytes: Last modified time stamp Third 8 bytes: Last accessed time stamp Last 8 bytes: MFT record update time stamp

$MFT Record Offset Info

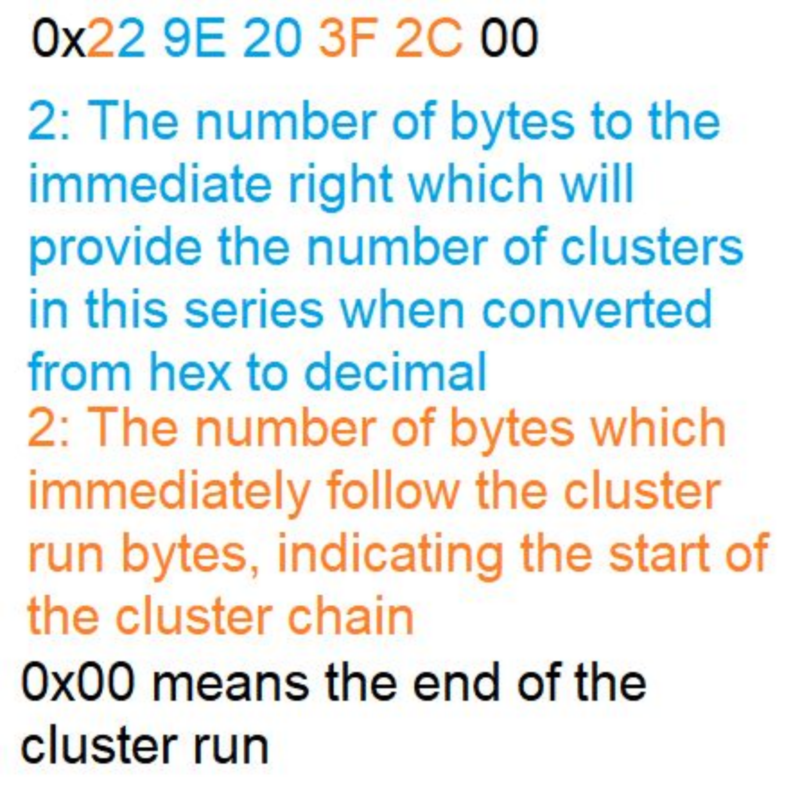


Data Attribute

Offset from the beginning of Attribute container	Length	Description
0x04	2 bytes	Attribute container size in bytes
0x08	1 byte	Resident File Flag
0x30	8 bytes	File Size in Bytes
0x40	Various bytes	Cluster chain information

Decoding Cluster Chain

- Cluster chain mapping information starts at decimal offset 64 of the Data Attribute (0x40) block of the MFT record



0x22 9E 20 3F 2C 00

2: The number of bytes to the immediate right which will provide the number of clusters in this series when converted from hex to decimal

2: The number of bytes which immediately follow the cluster run bytes, indicating the start of the cluster chain

0x00 means the end of the cluster run