Topic 2
Storage and File Systems II

Learning Outcome

To explain the various physical disk storage and hard disk interfaces

To examine and analyse the file systems used in Windows

To examine and analyse the file systems used in Linux

To explain the file systems used in Mac OS

To explain network file systems

Road Map

Windows File Systems

- FAT
- NTFS

File System

File System Format

- Organizes and stores data of different files in different designated clusters of sectors
- Provide index to the logical location (cluster and sectors number) to individual file on the medium
- Provide date/time information on file creation, modification and access

Windows File Systems

- FAT (File Allocation Table)
- NTFS (New Technology File System)
- exFAT (Extended FAT)
- ReFS (Resilient File System)

File Allocation Table (FAT)

The File Allocation Table (FAT) file system, supported by all versions of Microsoft Windows

- FAT12: Floppy Diskettes
- FAT16: hard disk drive started from 1988
- FAT32: in the Windows 95/98 hard disks; now it is the default FAT format for USB flash drives and SD flash memory in cameras. Maximum size of a file is 4GB
- ExFAT: Extended FAT file system has been adopted by the SD Card Association as the default file system for SDXC cards larger than 32GB (File size limit of 16EB)

Source: http://en.wikipedia.org/wiki/File_Allocation_Table

8.3 Filename Limit

FAT12 and FAT16 had a limit of 8 characters for the file name, and 3 characters for the extension (such as .exe). This is commonly referred to as the 8.3 filename limit.

For backward compatibility, an 8.3 filename is automatically generated for every long filenames

- TextFile1.txt => TEXTFI~1.TXT

To show

- dir /x – shows the short names (if any), and the long names
- dir /-n – shows only the short names

FAT Format

Volume Boot Record (VBR)

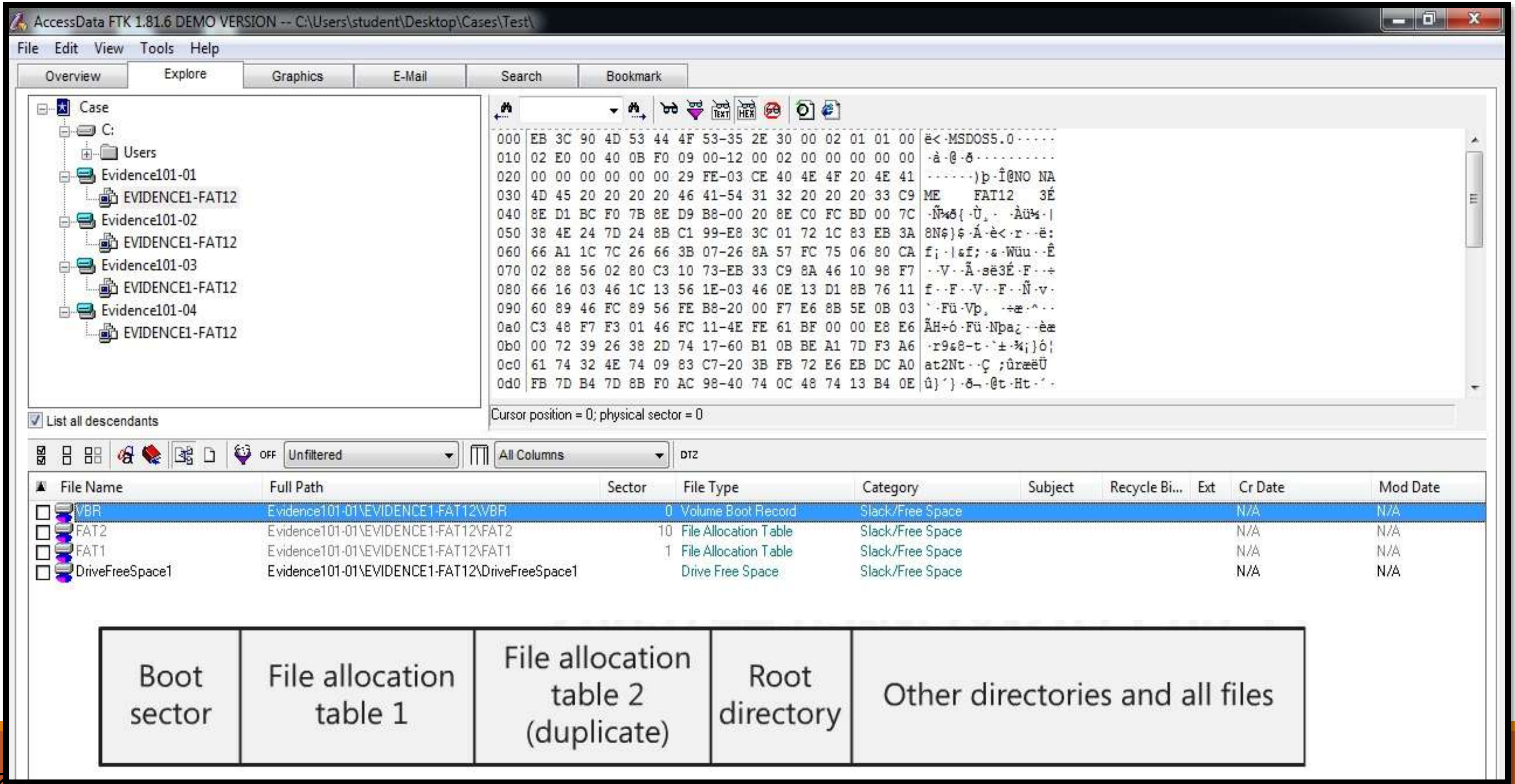
- Sector 0 of the first partition of a file volume (a file volume can have one or more partitions)
- On non-partitioned storage devices, it is the first sector (Sector 0) of the device.
- On partitioned devices, it is the first sector of an individual partition on the device

Question: What is stored at the first sector of the device?

File Allocation Table (FAT)

- A table stores the allocation of clusters to individual file which is stored after VBR
  - usually FAT is in sector 1
- A duplicated FAT is stored after the first FAT
  - for FAT recovery when the first FAT is corrupted

Sample FAT12 Layout



AccessData FTK 1.81.6 DEMO VERSION -- C:\Users\student\Desktop\Cases\Test\
File Edit View Tools Help
Overview Explore Graphics E-Mail Search Bookmark
Case
C:
Users
Evidence101-01
EVIDENCE1-FAT12
Evidence101-02
EVIDENCE1-FAT12
Evidence101-03
EVIDENCE1-FAT12
Evidence101-04
EVIDENCE1-FAT12
TEXT
HEX
OFF
DTZ

```
000 EB 3C 90 4D 53 44 4F 53-35 2E 30 00 02 01 01 00 ë<·MSDOS5.0·····
010 02 E0 00 40 0B F0 09 00-12 00 02 00 00 00 00 00 ·à·@·ð··········
020 00 00 00 00 00 00 29 FE-03 CE 40 4E 4F 20 4E 41 ······)þ·Î@NO NA
030 4D 45 20 20 20 20 46 41-54 31 32 20 20 20 33 C9 ME    FAT12   3É
040 8E D1 BC F0 7B 8E D9 B8-00 20 8E C0 FC BD 00 7C ·Ñ¼ð{·Ù¸· ·Àü½·|
050 38 4E 24 7D 24 8B C1 99-E8 3C 01 72 1C 83 EB 3A 8N$}$·Á·è<·r··ë:
060 66 A1 1C 7C 26 66 3B 07-26 8A 57 FC 75 06 80 CA f¡·|&f;·&·Wüu··Ê
070 02 88 56 02 80 C3 10 73-EB 33 C9 8A 46 10 98 F7 ··V··Ã·së3É·F··÷
080 66 16 03 46 1C 13 56 1E-03 46 0E 13 D1 8B 76 11 f··F··V··F··Ñ·v·
090 60 89 46 FC 89 56 FE B8-20 00 F7 E6 8B 5E 0B 03 `·Fü·Vþ¸ ·÷æ·^··
0a0 C3 48 F7 F3 01 46 FC 11-4E FE 61 BF 00 00 E8 E6 ÃH÷ó·Fü·Nþa¿··èæ
0b0 00 72 39 26 38 2D 74 17-60 B1 0B BE A1 7D F3 A6 ·r9&8-t·`±·¾¡}ó¦
0c0 61 74 32 4E 74 09 83 C7-20 3B FB 72 E6 EB DC A0 at2Nt··Ç ;ûræëÜ 
0d0 FB 7D B4 7D 8B F0 AC 98-40 74 0C 48 74 13 B4 0E û}´}·ð¬·@t·Ht·´·
```

Cursor position = 0; physical sector = 0

☑ List all descendants

Unfiltered

All Columns

File Name	Full Path	Sector	File Type	Category	Subject	Recycle Bi...	Ext	Cr Date	Mod Date
□ VBR	Evidence101-01\EVIDENCE1-FAT12\VBR	0	Volume Boot Record	Slack/Free Space				N/A	N/A
□ FAT2	Evidence101-01\EVIDENCE1-FAT12\FAT2	10	File Allocation Table	Slack/Free Space				N/A	N/A
□ FAT1	Evidence101-01\EVIDENCE1-FAT12\FAT1	1	File Allocation Table	Slack/Free Space				N/A	N/A
□ DriveFreeSpace1	Evidence101-01\EVIDENCE1-FAT12\DriveFreeSpace1		Drive Free Space	Slack/Free Space				N/A	N/A

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

Main Components in FAT File System

Volume Boot Record (VBR)

- Store FAT information that includes
  - number of bytes per sector,
  - number of sectors per cluster,
  - number of sectors per FAT

File Allocation Table (FAT)

- Stores addresses of cluster used by individual file
- Special data patterns represent different status of the cluster
  - Unallocated (0x0000)
  - Bad cluster (0xFFF7)
  - Last cluster in a file (0xFFF8 - 0xFFFF)

Root Folder/Directory

- Filenames, Directory names
- Attributes of individual file
  - Date and timestamp, the starting cluster number and status (archived, hidden, system and read-only).

Ref: File System

- Organizes and stores data of different files in different designated clusters of sectors
- Provide index to the logical location (cluster and sectors number) to individual file on the medium
- Provide date/time information on file creation, modification and access

Question: Can you explain how the FAT File System implements the functions of a File System?

Main Components in FAT File System

Sector	Description
0	Volume Boot Record (512 bytes)
1	FAT
10	FAT Backup
19	Root Folder
33	First cluster (cluster 0x0002) stores data of a file

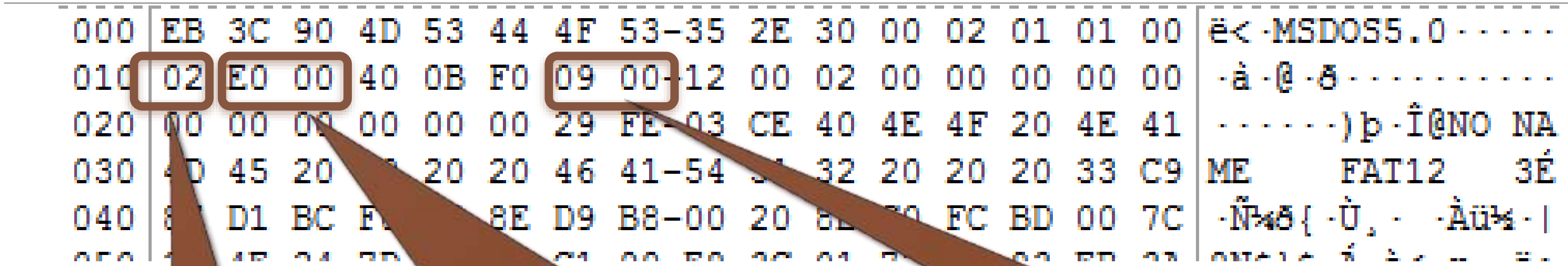
How do we know the sector numbers for
FAT and Root Folder?

Volume Boot Record

It contains the FAT Format Information

Byte Offset	Field Length	Sample Value	Meaning
0x00	3 bytes	EB 3C 90	Jump instruction
0x03	8 bytes		BIOS Parameter Block
0x0B	WORD	0x0002	Bytes per Sector. The size of a hardware sector. For most disks in use in the United States, the value of this field is 512.
0x0D	BYTE	0x08	Sectors Per Cluster. The number of sectors in a cluster. The default cluster size for a volume depends on the volume size and the file system.
0x0E	WORD	0x0100	Reserved Sectors. The number of sectors from the Partition Boot Sector to the start of the first file allocation table, including the Partition Boot Sector. The minimum value is 1. If the value is greater than 1, it means that the bootstrap code is too long to fit completely in the Partition Boot Sector.
0x10	BYTE	0x02	Number of file allocation tables (FATs). The number of copies of the file allocation table on the volume. Typically, the value of this field is 2.
0x11	WORD	0x0002	Root Entries. The total number of file name entries that can be stored in the root folder of the volume. One entry is always used as a Volume Label. Files with long filenames use up multiple entries per file. Therefore, the largest number of files in the root folder is typically 511, but you will run out of entries sooner if you use long filenames.
0x16	WORD	0xC900	Sectors per file allocation table (FAT). Number of sectors occupied by each of the file allocation tables on the volume. By using this information, together with the Number of FATs and Reserved Sectors, you can compute where the root folder begins. By using the number of entries in the root folder, you can also compute where the user data area of the volume begins.

VBR at Sector 0



```
000  EB 3C 90 4D 53 44 4F 53-35 2E 30 00 02 01 01 00  ë<·MSDOS5.0·····
010  02 E0 00 40 0B F0 09 00-12 00 02 00 00 00 00 00  ·à·@·ð··········
020  00 00 00 00 00 00 29 FE-03 CE 40 4E 4F 20 4E 41  ······)þ·Î@NO NA
030  4D 45 20 [illegible] 20 20 46 41-54 [illegible] 32 20 20 20 33 C9  ME    FAT12   3É
040  8[illegible] D1 BC F[illegible] [illegible] 8E D9 B8-00 20 8[illegible] [illegible] FC BD 00 7C  ·Ñ¼ð{·Ù¸·  ·Àü½·|
```

Number of file entries in Root Folder is 0x00E0 (224). The size of Root Folder will be 224 x (32 bytes per file entry). That is 7168 bytes or 14 sectors

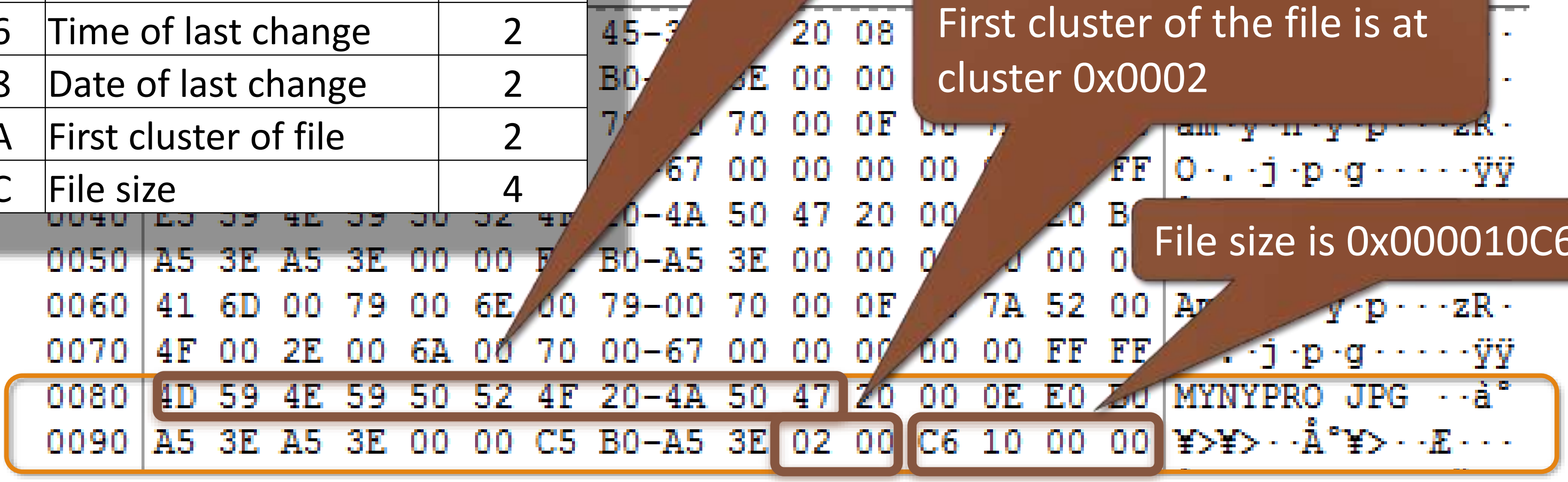
Sectors per FAT is 0x0009. It means that FAT is from sectors 1 to 9 and FAT Backup is at sectors 10 to 18, followed by the Root Folder at sector 19

Number of FAT is 0x02. FAT and FAT backup.

Refer to https://en.wikipedia.org/wiki/Design_of_the_FAT_file_system#Bootsector for detailed descriptions

Root Folder at Sector 19

Offset	Description	Size in bytes
0x0	Filename	8
0x8	File extension	3
0xB	File Attribute	1
0xC	Reserved	10
0x16	Time of last change	2
0x18	Date of last change	2
0x1A	First cluster of file	2
0x1C	File size	4



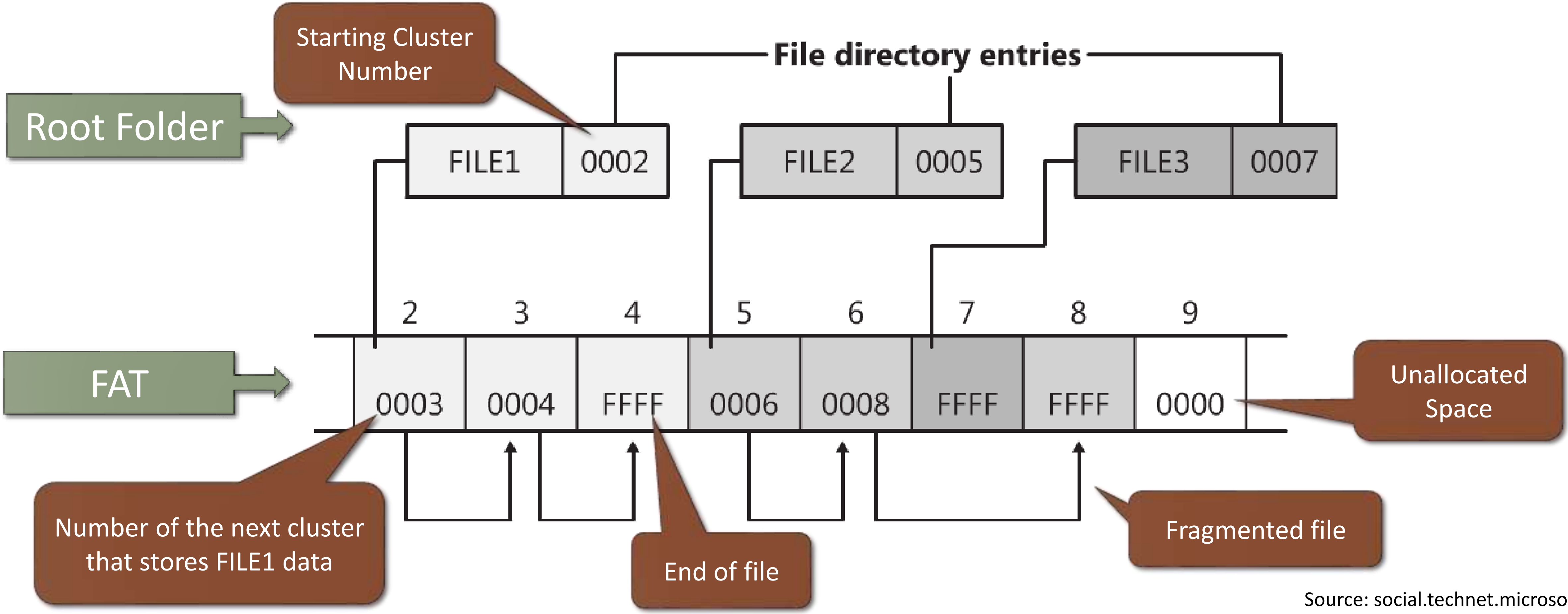
```
0060  41 6D 00 79 00 6E 00 79-00 70 00 0F    7A 52 00  Am  y p···zR·
0070  4F 00 2E 00 6A 00 70 00-67 00 00 00 00 00 FF FF  O·.·j·p·g·····ÿÿ
0080  4D 59 4E 59 50 52 4F 20-4A 50 47 20 00 0E E0 B0  MYNYPRO JPG ··à°
0090  A5 3E A5 3E 00 00 C5 B0-A5 3E 02 00 C6 10 00 00  ¥>¥>··Å°¥>··Æ···
```

Filename MYNYPRO.JPG

First cluster of the file is at cluster 0x0002

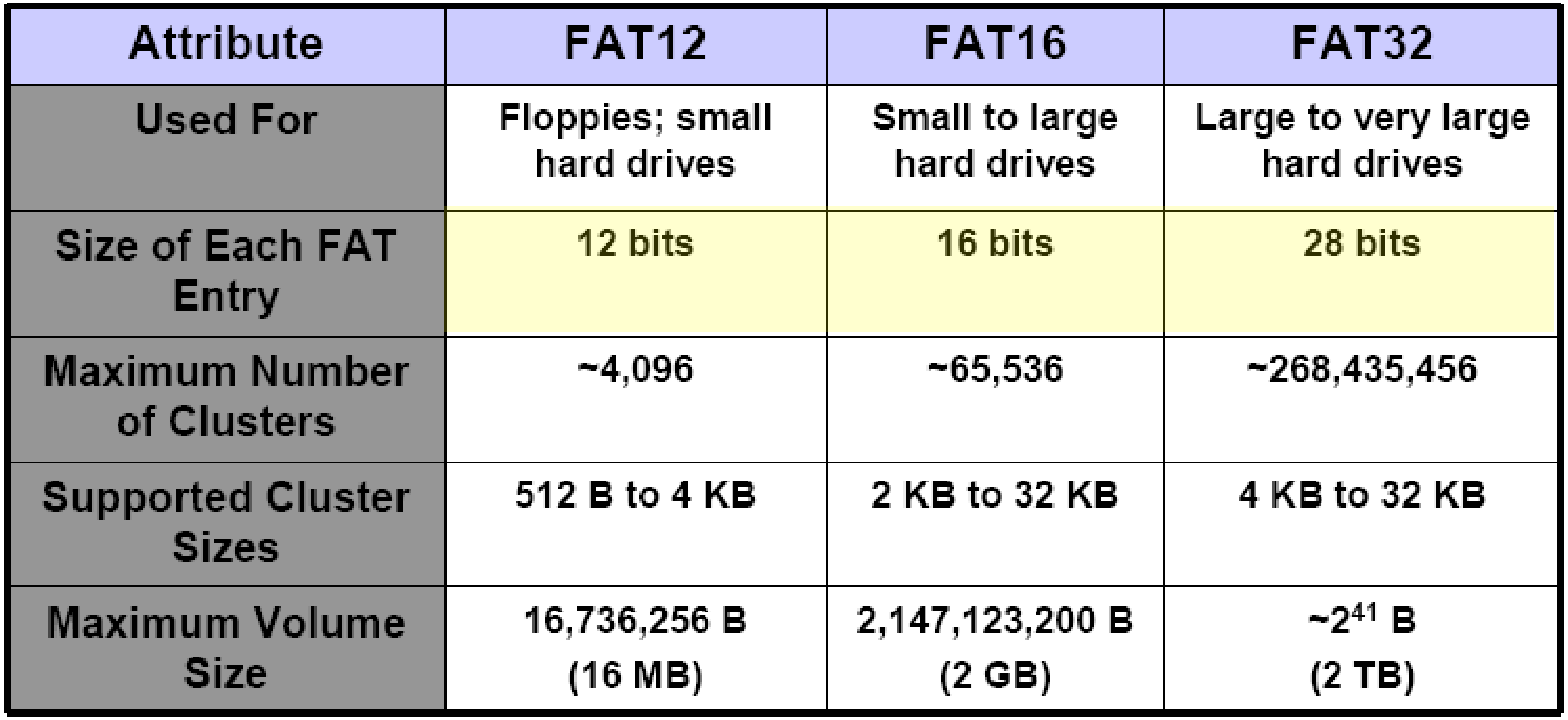
File size is 0x000010C6 bytes

Root Folder and FAT Working Together



Source: social.technet.microsoft.com

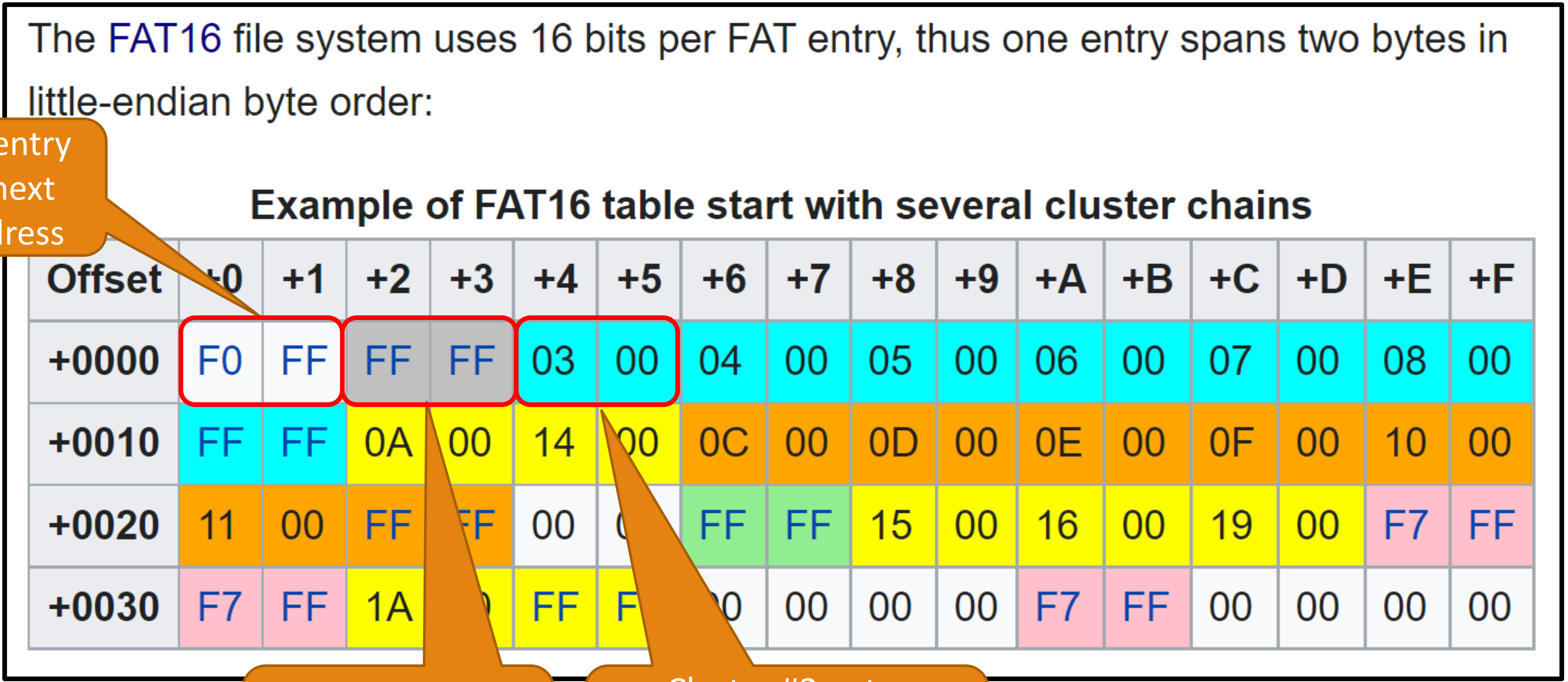
FATs Compared



Attribute	FAT12	FAT16	FAT32
Used For	Floppies; small hard drives	Small to large hard drives	Large to very large hard drives
Size of Each FAT Entry	12 bits	16 bits	28 bits
Maximum Number of Clusters	~4,096	~65,536	~268,435,456
Supported Cluster Sizes	512 B to 4 KB	2 KB to 32 KB	4 KB to 32 KB
Maximum Volume Size	16,736,256 B (16 MB)	2,147,123,200 B (2 GB)	~2^{41} B (2 TB)

Source: http://www.c-jump.com/CIS24/Slides/FAT/lecture.html#F01_0200_fats_compared

See How FAT16 works at FAT (Wikipedia.org: Design of the FAT file system)



The FAT16 file system uses 16 bits per FAT entry, thus one entry spans two bytes in little-endian byte order:

Example of FAT16 table start with several cluster chains

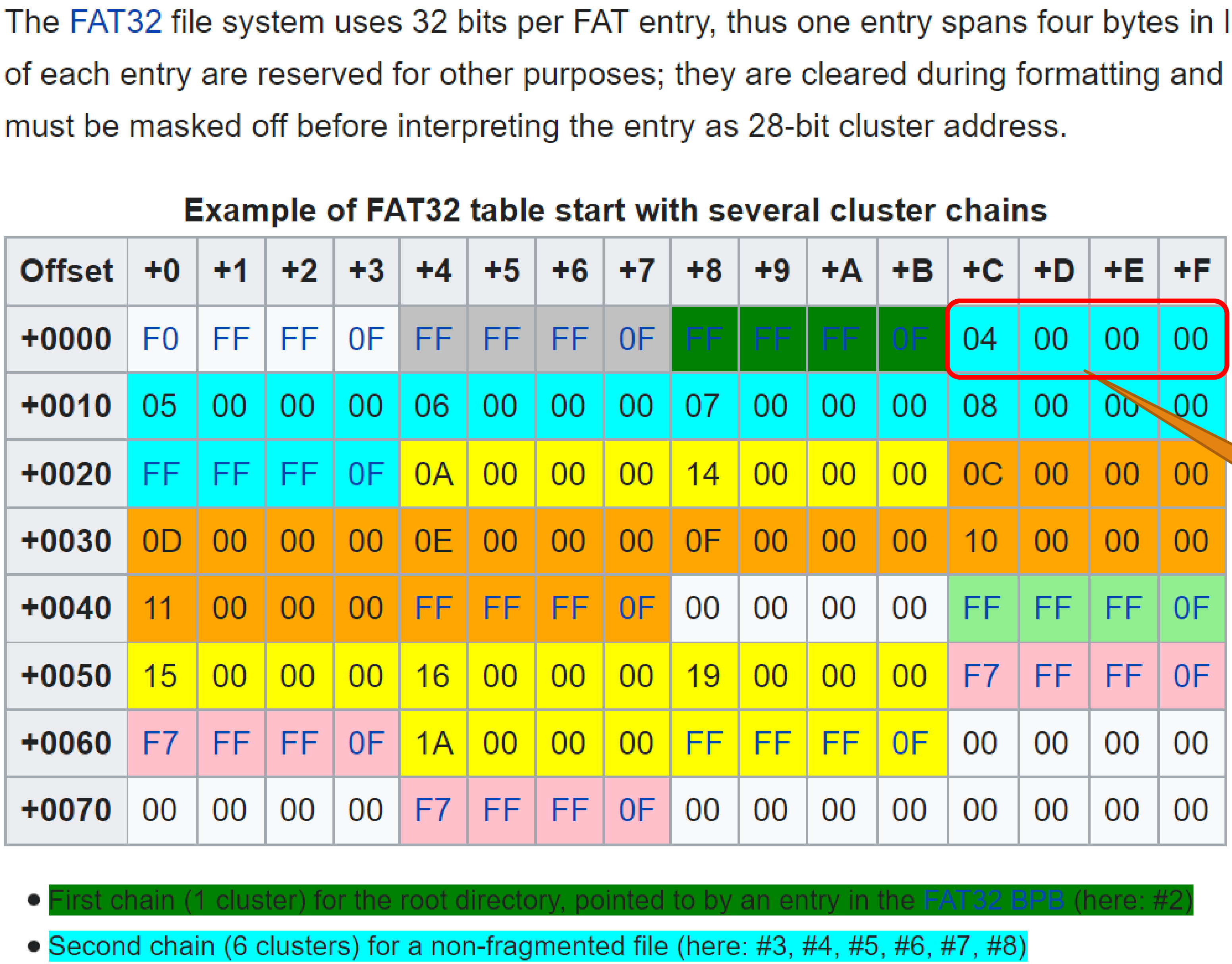
Offset	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
+0000	F0	FF	FF	FF	03	00	04	00	05	00	06	00	07	00	08	00
+0010	FF	FF	0A	00	14	00	0C	00	0D	00	0E	00	0F	00	10	00
+0020	11	00	FF	FF	00	00	FF	FF	15	00	16	00	19	00	F7	FF
+0030	F7	FF	1A	00	FF	FF	00	00	00	00	F7	FF	00	00	00	00

Cluster #0 entry indicates next cluster address

Cluster #1 entry

Cluster #2 entry indicates next cluster address is Cluster #3

See How FAT32 works at FAT



The FAT32 file system uses 32 bits per FAT entry, thus one entry spans four bytes in little-endian byte order. The four top bits of each entry are reserved for other purposes; they are cleared during formatting and should not be changed otherwise. They must be masked off before interpreting the entry as 28-bit cluster address.

Example of FAT32 table start with several cluster chains

Offset	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
+0000	F0	FF	FF	0F	FF	FF	FF	0F	FF	FF	FF	0F	04	00	00	00
+0010	05	00	00	00	06	00	00	00	07	00	00	00	08	00	00	00
+0020	FF	FF	FF	0F	0A	00	00	00	14	00	00	00	0C	00	00	00
+0030	0D	00	00	00	0E	00	00	00	0F	00	00	00	10	00	00	00
+0040	11	00	00	00	FF	FF	FF	0F	00	00	00	00	FF	FF	FF	0F
+0050	15	00	00	00	16	00	00	00	19	00	00	00	F7	FF	FF	0F
+0060	F7	FF	FF	0F	1A	00	00	00	FF	FF	FF	0F	00	00	00	00
+0070	00	00	00	00	F7	FF	FF	0F	00	00	00	00	00	00	00	00

- First chain (1 cluster) for the root directory, pointed to by an entry in the FAT32 BPB (here: #2)
- Second chain (6 clusters) for a non-fragmented file (here: #3, #4, #5, #6, #7, #8)

Cluster #3 entry indicates next cluster address is Cluster #4

Other FATs

VFAT

- Invented to handle long file names
- Used in 1st version of Windows 95 & Windows for Workgroups

exFAT

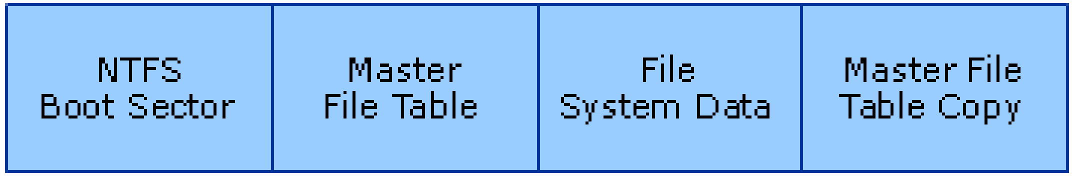
- Developed by Microsoft, designed specifically for flash drive
- Used when NTFS is not a feasible solution due to data structure overhead; or need to go beyond FAT32 size limit

NTFS

NTFS - New Technology File System

- Partition Boot Sector (PBR)
  - Similar to VBR in FAT
  - Occupies the first 16 sectors
- Master File Table (MFT)
  - Similar to directory entry in FAT
  - Entry for every file and directory including itself ($MFT)
  - Contains file metadata
  - The starting location of MFT is given in the boot sector

- $bitmap
  - Similar to the file allocation table
  - Represents cluster allocation



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

NTFS File System Metadata Files

File Name	Description
$MFT	Entry of MFT itself
$MFTMirr	Backup of the first entries in the MFT
$LogFile	Journal that records the metadata transactions
$Volume	Volume information, such as the label and version
$AttrDef	Attribute information such as identifier values, name and size
.	Root directory of the file system
$Bitmap	Allocation status of each cluster in the file system
$Boot	Boot sector and boot code for the file system
$BadClus	Clusters that have bad sectors
$Secure	Information about the security and access control
$Upcase	Uppercase version of every Unicode character
$Extend	A directory that contains files for optional extension

NTFS

NTFS was introduced with the Windows NT operating system. It provides

- File owner information
- Access Control List in each file/folder header
- System time zone information
- Alternate Data Stream (ADS)
- File storage quota tracking and control
- Encryption File System
- File compression
- Volume shadow copy

Alternate Data Streams (ADS)

NTFS ADS were introduced in Windows NT 3.1

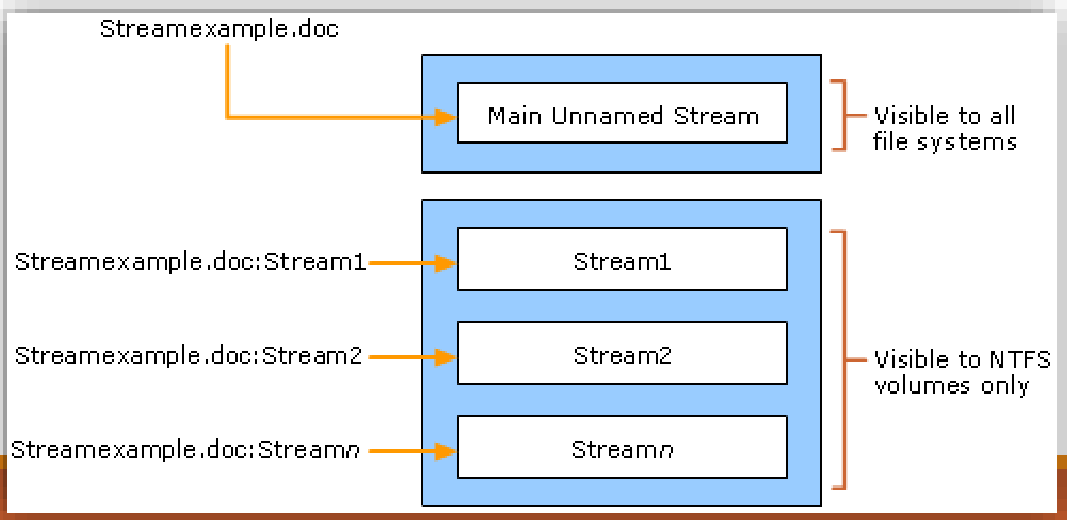
- For compatibility with the Mac HFS
  - HFS stores icon and other information in an alternative scream.

ADS are used for other purposes in Windows 2000 and XP

- Applications can create additional named streams and access these streams by referring to their names, which allows related data to be managed as a single unit.
  - Thumbnails
  - Internet explorer add zone identifier into files downloaded from Internet

Can be used to hide executable content

- Perl scripts
- Windows Scripting Host files
- Malware!


Streamexample.doc
Main Unnamed Stream
Visible to all file systems
Streamexample.doc:Stream1
Stream1
Streamexample.doc:Stream2
Stream2
Streamexample.doc:Stream*n*
Stream*n*
Visible to NTFS volumes only

Alternate Data Streams (ADS)

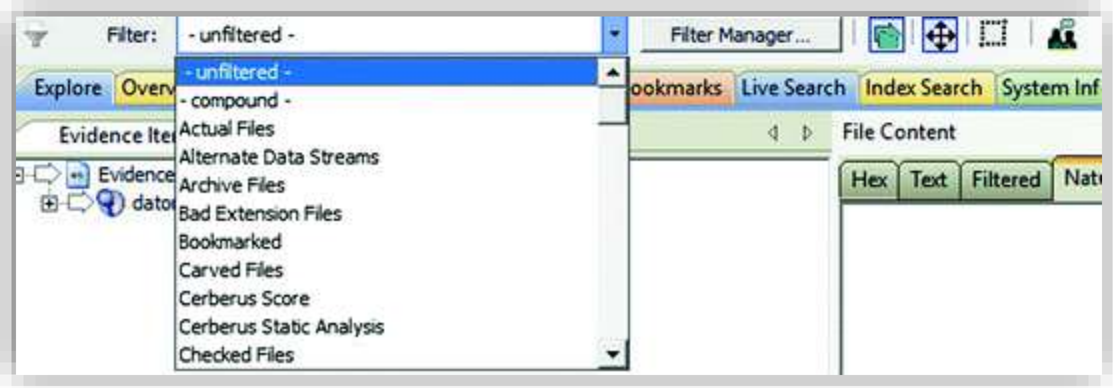
To create an ADS file

- echo “this is an ADS” > myfile.txt:ads.txt
- myfile.txt will also be created but is zero bytes in size

To identify an ADS file

- Viewing of NTFS ADS is available for Windows Vista and above
  - Use “dir /r” command
  - myfile.txt:ads.txt:$DATA

Commercial forensic applications will usually be abled to recognize ADS files.



Screenshot of AccessData FTK GUI

Encrypting File System

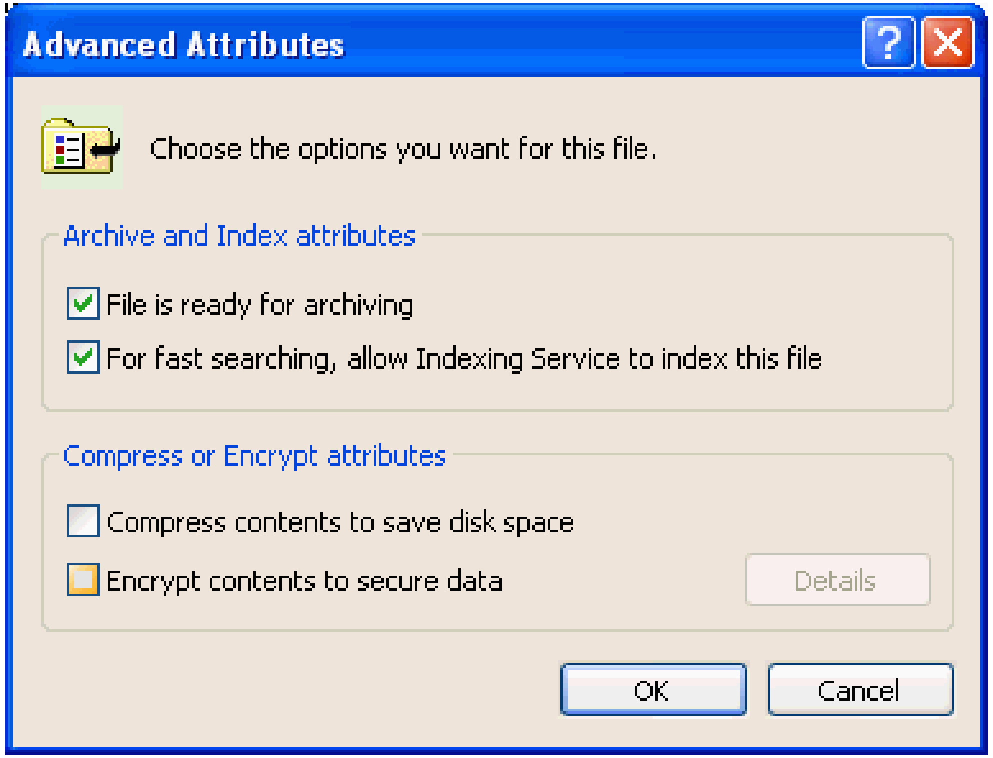
Allows users to encrypt individual files or entire folders

Built into Windows 2000 and XP Professional and later

Encrypted files are only viewable by the user who encrypted them or by designated recovery agents

- Decryption is automatic without the need to enter password

Can invoke feature by selecting checkbox in Advanced Attributes property of files



Advanced Attributes

Choose the options you want for this file.

Archive and Index attributes

☑ File is ready for archiving

☑ For fast searching, allow Indexing Service to index this file

Compress or Encrypt attributes

□ Compress contents to save disk space

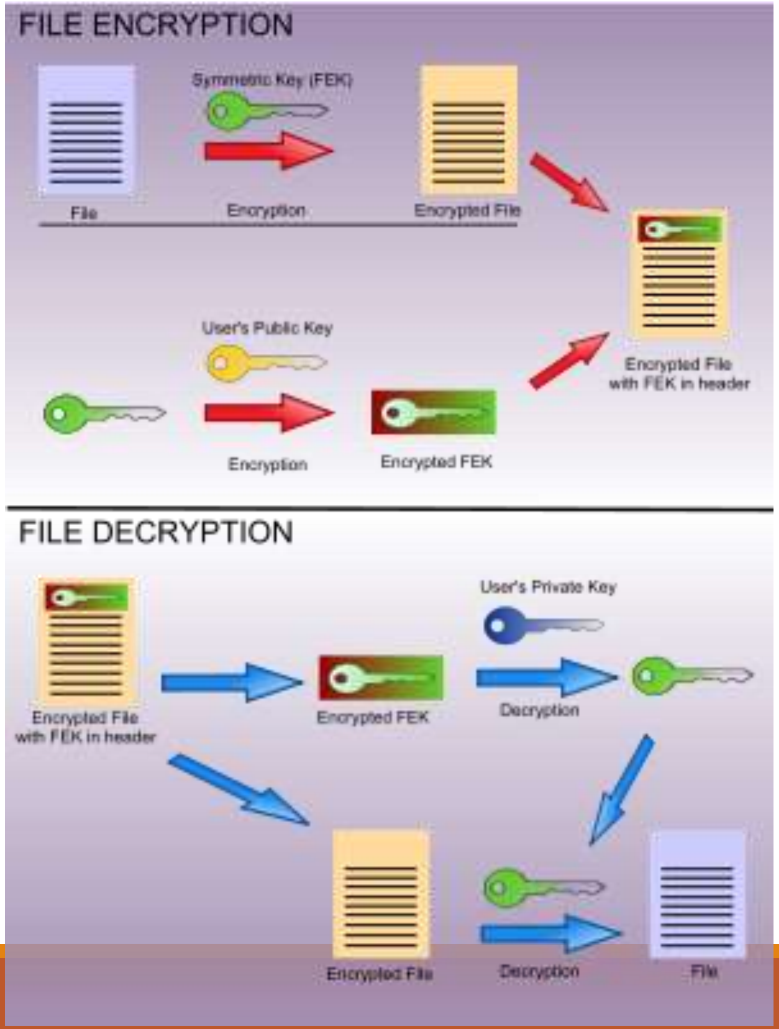
□ Encrypt contents to secure data

Encrypting File System

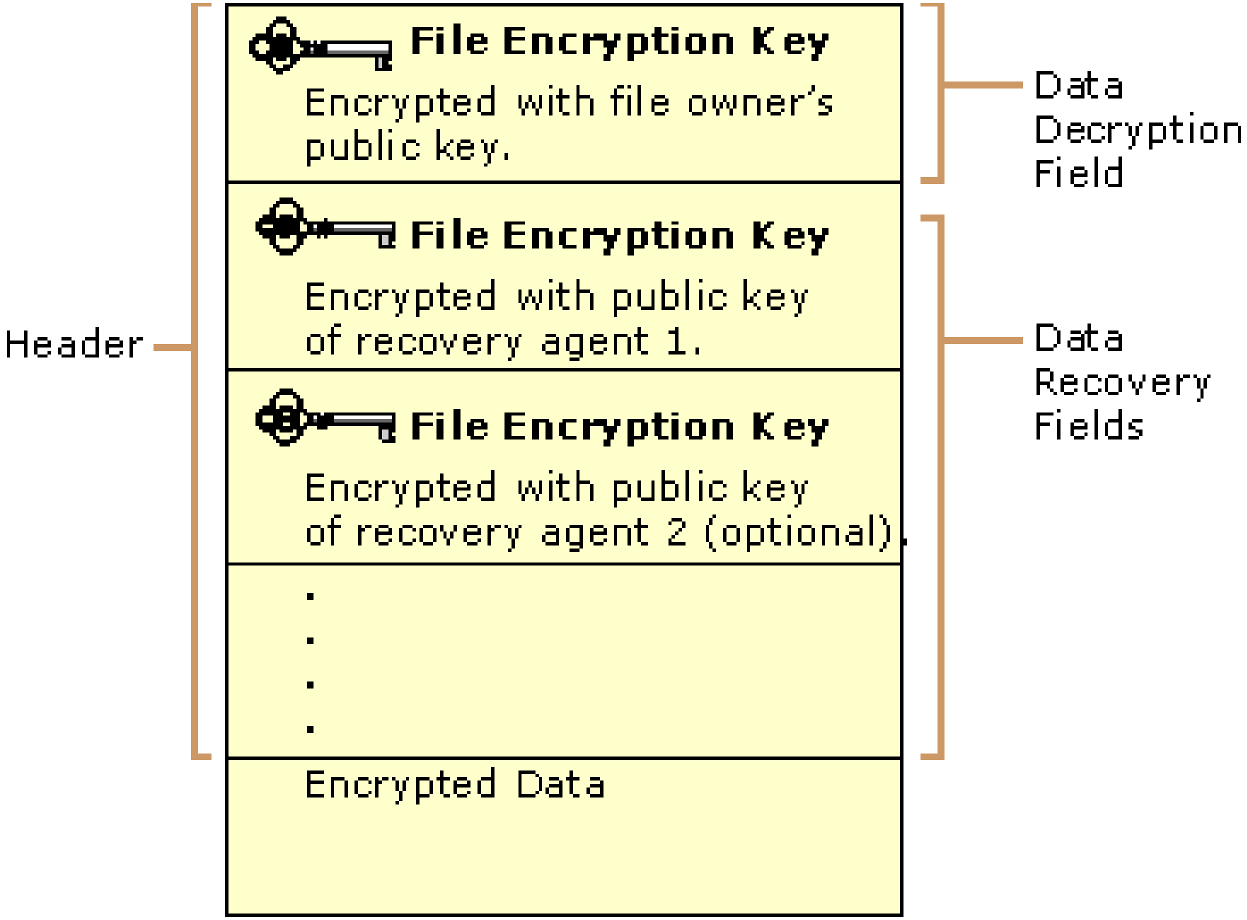
When EFS is activated

1. User logon password => Passkey
2. Passkey + User’s protected information => Master Key
3. A pair of private and public is created
   - Unique for each user
4. Master Key encrypts the private key

5.



Encrypting File System



Header

File Encryption Key

Encrypted with file owner's public key.

File Encryption Key

Encrypted with public key of recovery agent 1.

File Encryption Key

Encrypted with public key of recovery agent 2 (optional).

.
.
.
.

Encrypted Data

Data Decryption Field

Data Recovery Fields

Summary

FAT File System

- File System Structure: VBR, FAT, Root Folder
- 8.3 file name limitation

NTFS

- File System Structure Overview
- NTFS features: ADS, EFS

References

1. File System Forensic Analysis, Brian Carrier, 2005, Addison Wesley
2. http://social.technet.microsoft.com/wiki/contents/articles/6771.the-fat-file-system-en-us.aspx
3. http://en.wikipedia.org/wiki/File_Allocation_Table
4. http://en.wikipedia.org/wiki/NTFS
5. Hacking Exposed Computer Forensics Second Edition, Aaron Philipp, 2010, McGraw-Hill
6. Guide to Integrating Forensic Techniques into Incident Response SP800-86 NIST, csrc.nist.org
7. Cyber Forensics – From Data to Digital Evidence, Albert J. Marcella JR, Frederic Gullossou, 2012, Wiley