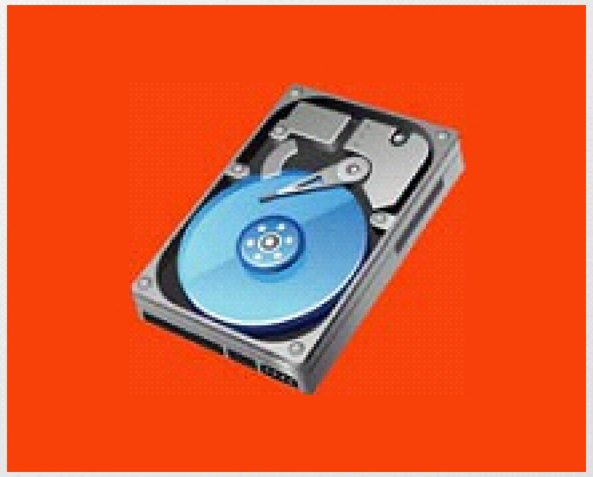
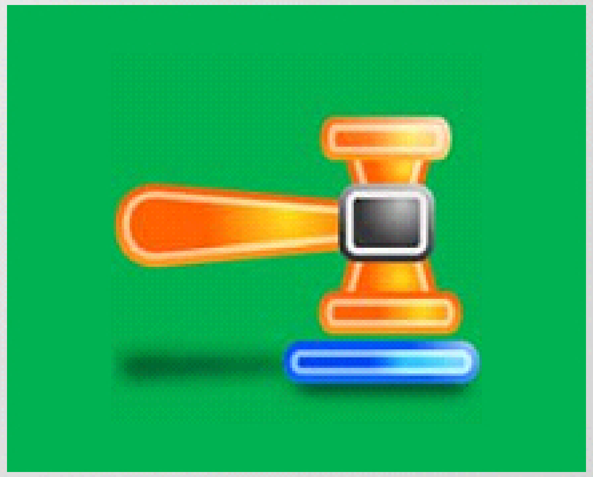
Understanding Hard Disks and File Systems

Module 03

Designed by Cyber Crime Investigators. Presented by Professionals.

C|HFI™
Computer Hacking Forensic Investigator





Module Objectives

After successfully completing this module, you will be able to:

1. Describe the different types of disk drives and their characteristics
2. Understand the physical and logical structure of a hard disk
3. Identify the types of hard disk interfaces and discuss the various hard disk components
4. Describe hard disk partitions
5. Summarize Windows, Mac, and Linux boot Processes
6. Understand various Windows, Linux and Mac OS X file systems
7. Differentiate between various RAID storage systems
8. Demonstrate file system analysis

Disk Drive Overview

HDD

Hard Disk Drive (HDD)

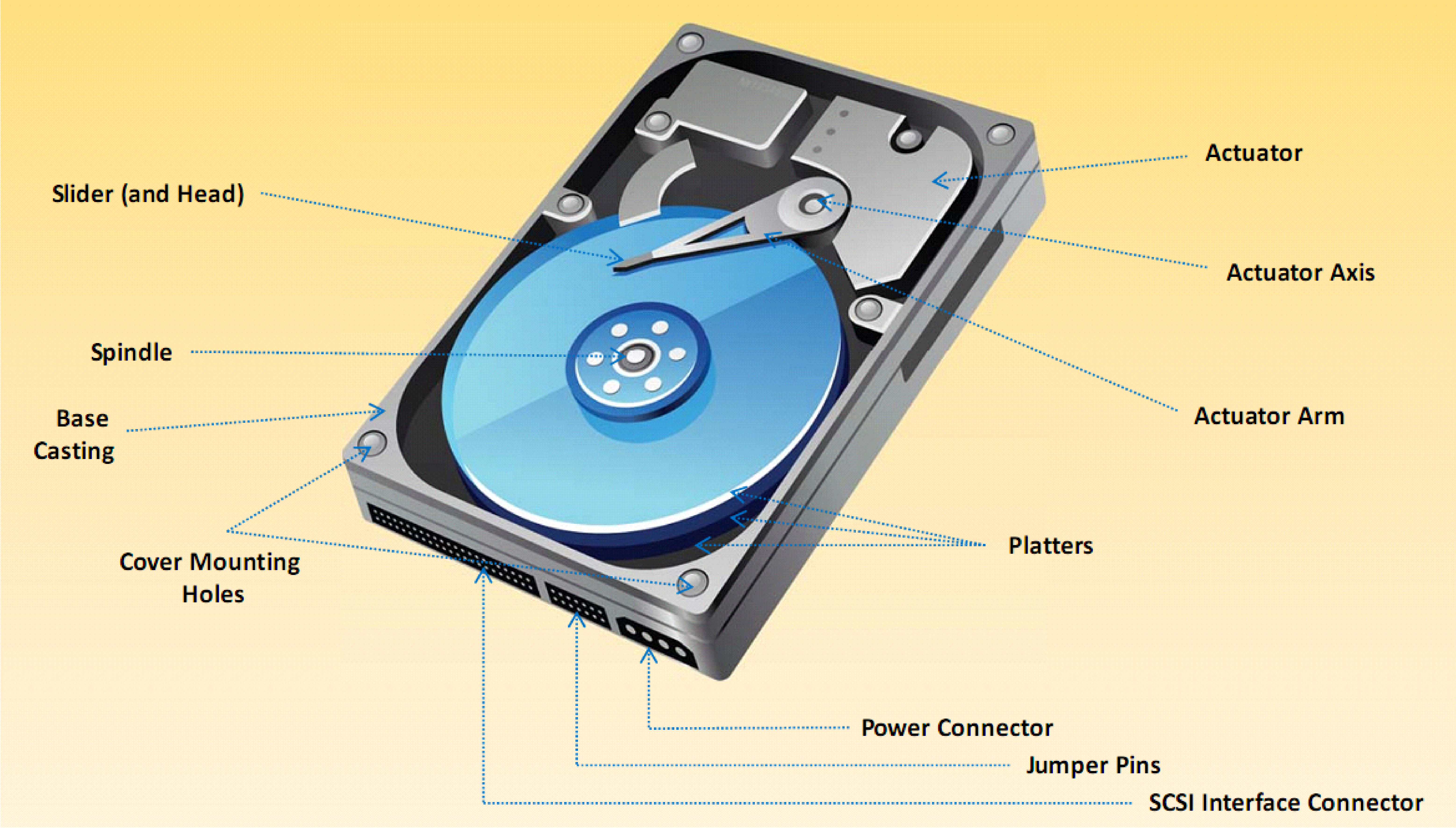
- The HDD is a **non-volatile**, random access digital data storage device used in any computer system
- It utilizes a mechanism that reads data from a disk and writes onto an another disk
- The hard disk **record data magnetically**

SSD

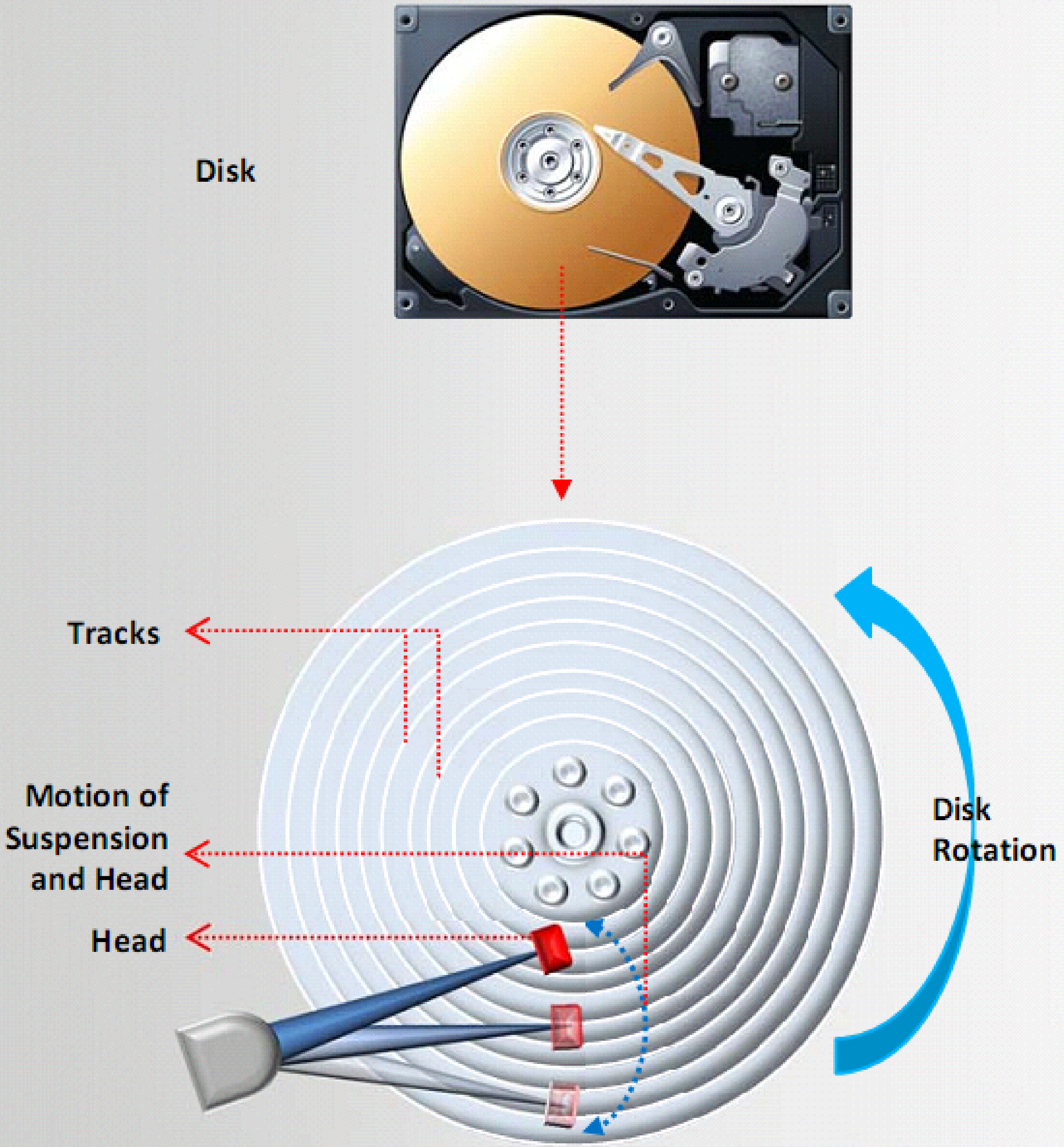
Solid-state Drive (SSD)

- The SSD is a data storage device that uses **solid-state memory** to store data and provides access to the stored data in the same manner as a HDD
- It **uses microchips** to hold data in non-volatile memory chips and does not contain any moving parts
- It is **very expensive** per gigabyte (GB) and supports a restricted number of writes over the life of the device
- It uses two memories:
  - NAND-based flash memory: It retains memory even without power
  - Volatile RAM: It provides faster access

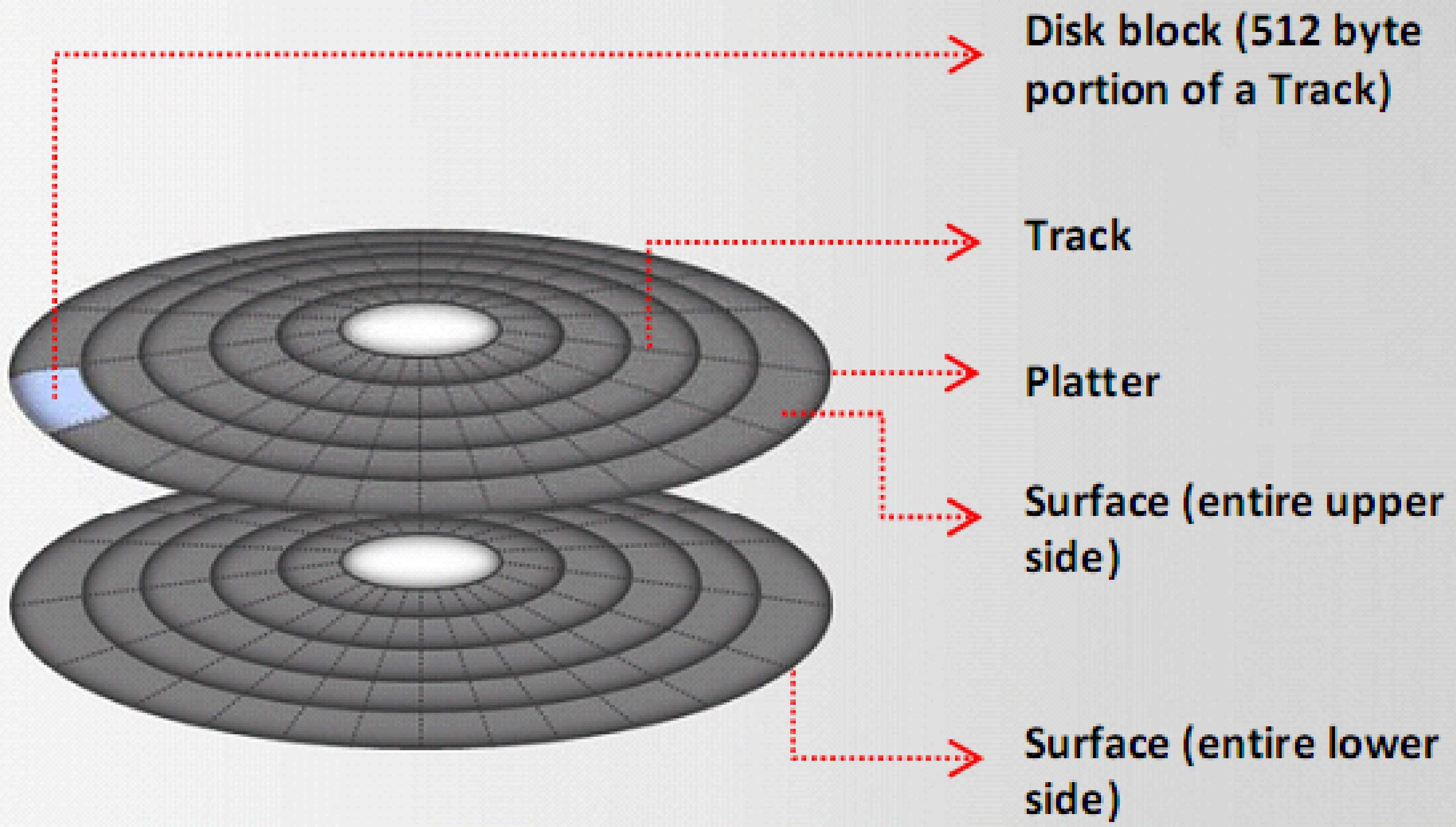
Physical Structure of a Hard Disk

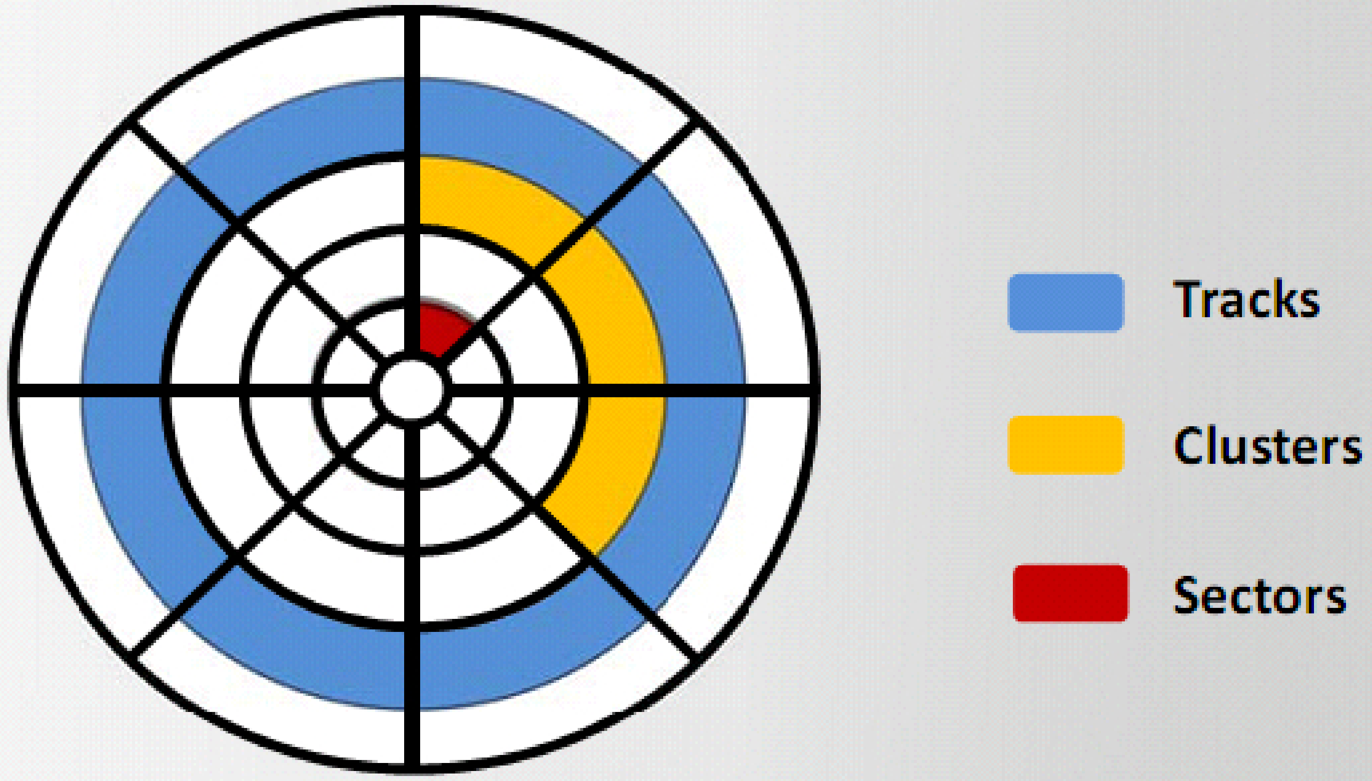


Physical Structure of a Hard Disk (Cont'd)



Magnetized Data on Disk


Disk block (512 byte portion of a Track)
Track
Platter
Surface (entire upper side)
Surface (entire lower side)



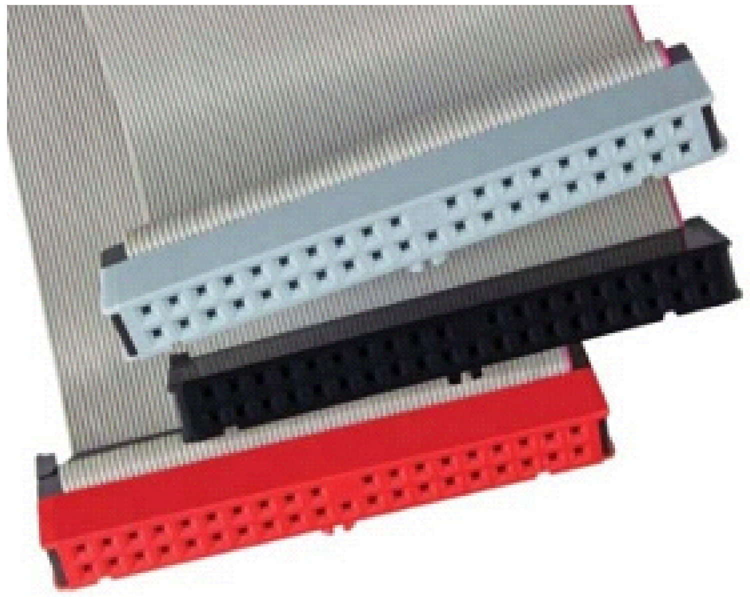
Logical Structure of Hard Disk

- The logical structure of a hard disk is the **file system and software** utilized to control access to the storage on the disk

- The hard disk logical structure has significant influence on the **performance**, **consistency**, **expandability**, and **compatibility** of the storage subsystem of the hard disk

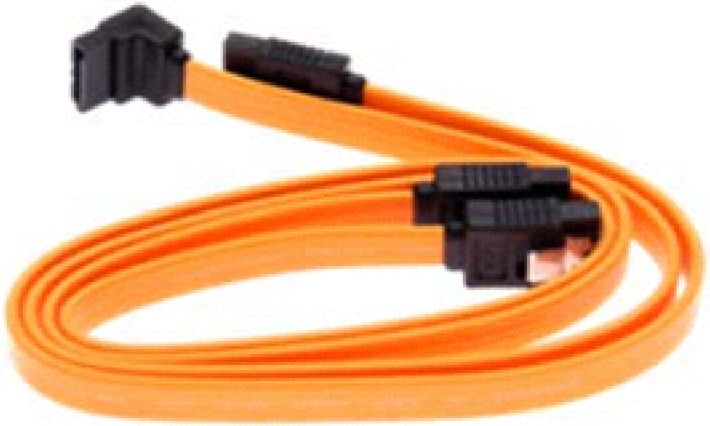
- Different operating systems have different file systems and use various ways of **arranging and controlling access** to data on the hard disk

Hard Disk Interfaces



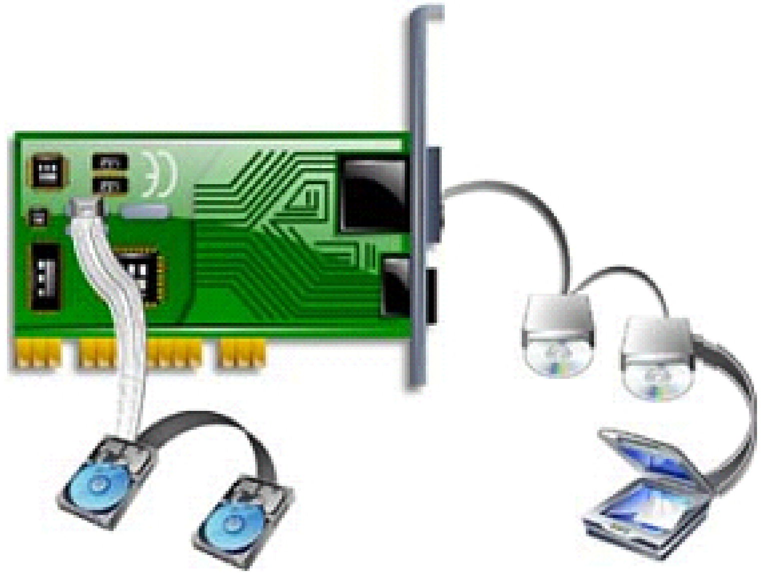
ATA/PATA (IDE/EIDE)

ATA (Advanced Technology Attachment) is the official ANSI name of Integrated Drive Electronics (IDE), a standard interface between a motherboard's data bus and storage discs



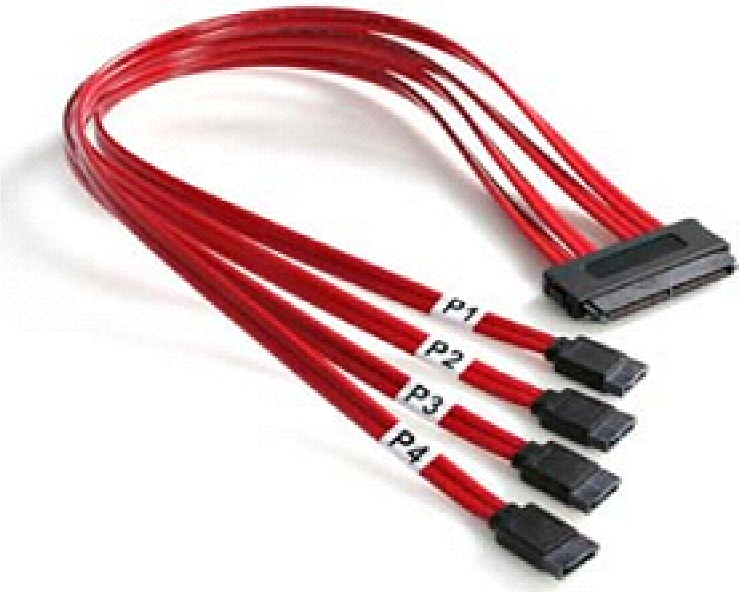
Serial ATA (SATA)

It is an advancement of ATA and uses serial signaling unlike IDE's parallel signaling



SCSI

SCSI (Small Computer System Interface) refers to a set of ANSI standard interfaces, based on the parallel bus structure and designed to connect multiple peripherals to a computer

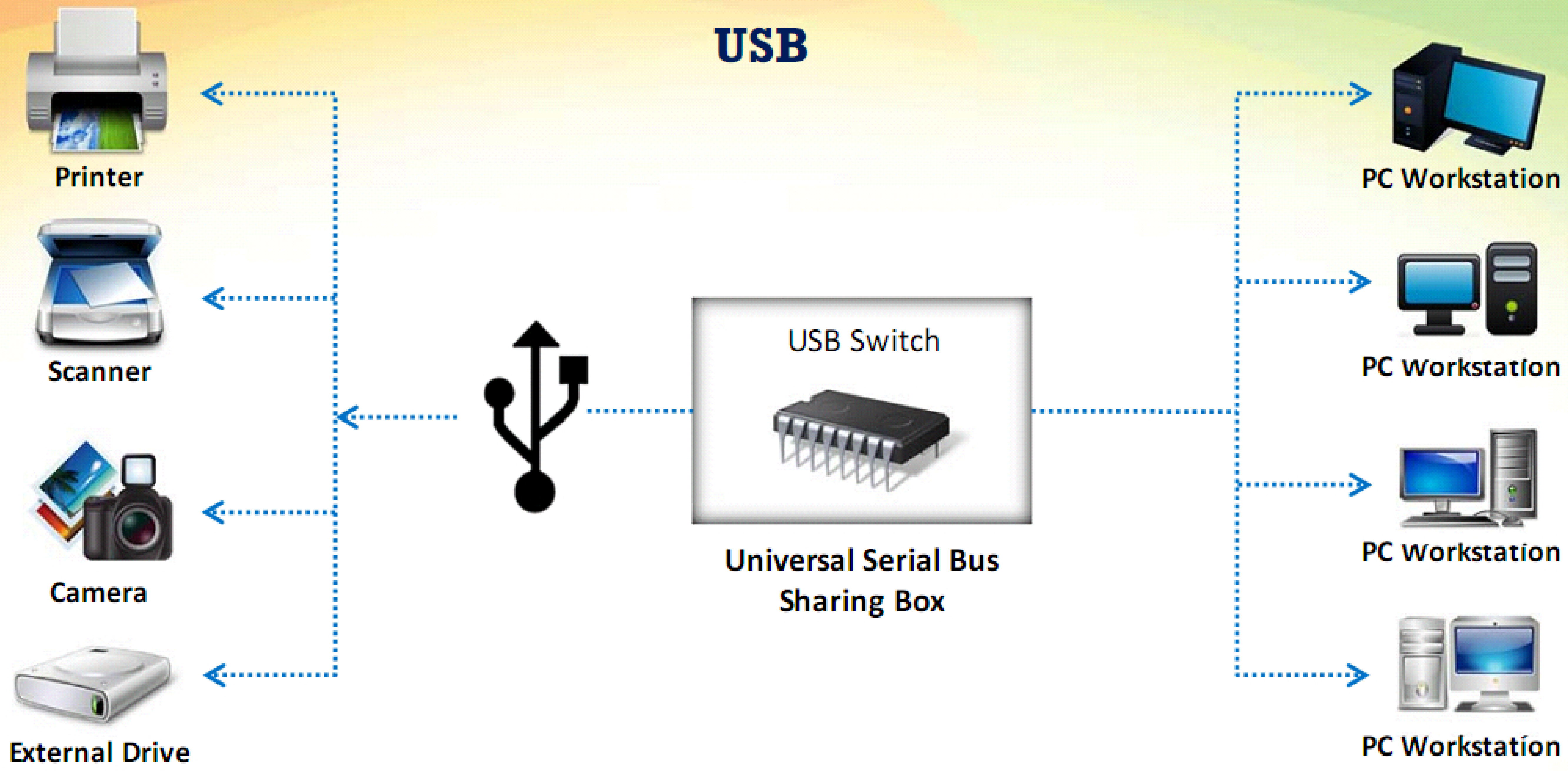


Serial Attached SCSI

SAS is successor and an advanced alternative to parallel SCSI in enterprise environments

Hard Disk Interfaces (Cont'd)

USB



Fibre Channel

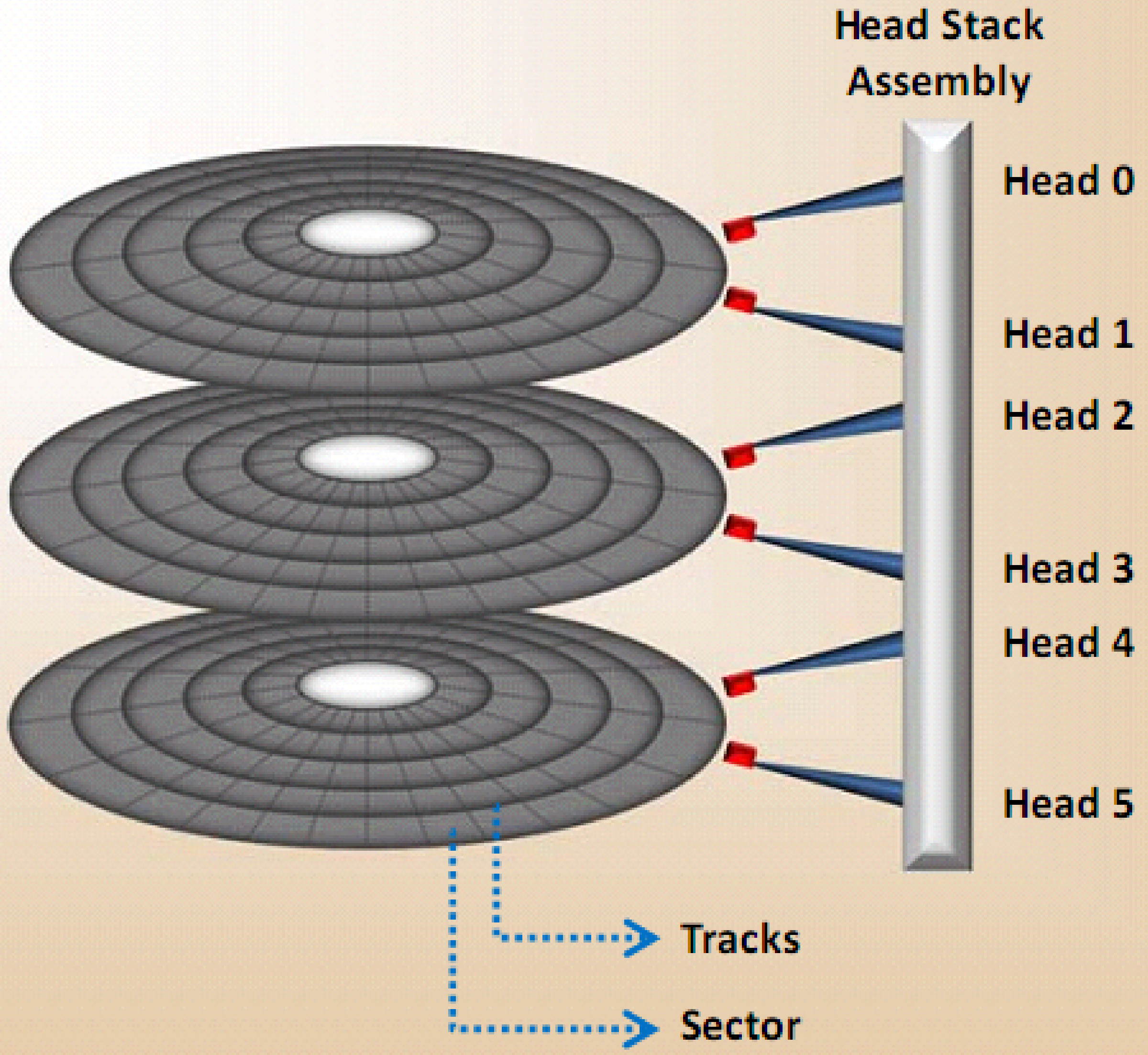
- Fibre Channel (FC) is a **point-to-point bi-directional serial** interface that supports up to 4 Gbps data transfer rates between computer devices
- It is particularly suitable for **linking computer system servers** to shared storage devices and for interconnecting storage controllers and disk drives

Tracks

- Tracks are the **concentric circles on platters** where all the information is stored
- Drive head can **access** these circular rings in one position at a time
- Tracks are numbered for **identification purposes**
- Read-write is done by **rolling headers** from inner to outermost part of the disk

Track Numbering:

- Track numbering on a hard disk **begins** at **0** from the outer edge and moves towards the center, typically **reaching a value of 1023**
- The read/write heads on both surfaces of a platter are tightly packed and locked together on an assembly of head arms
- The arms move in and out together to physically locate all heads at the same **track number**
- Therefore, a track location is often referred by a cylinder number rather than a track number
- A cylinder is a group of all tracks that start at the same head position on the disk



Sector

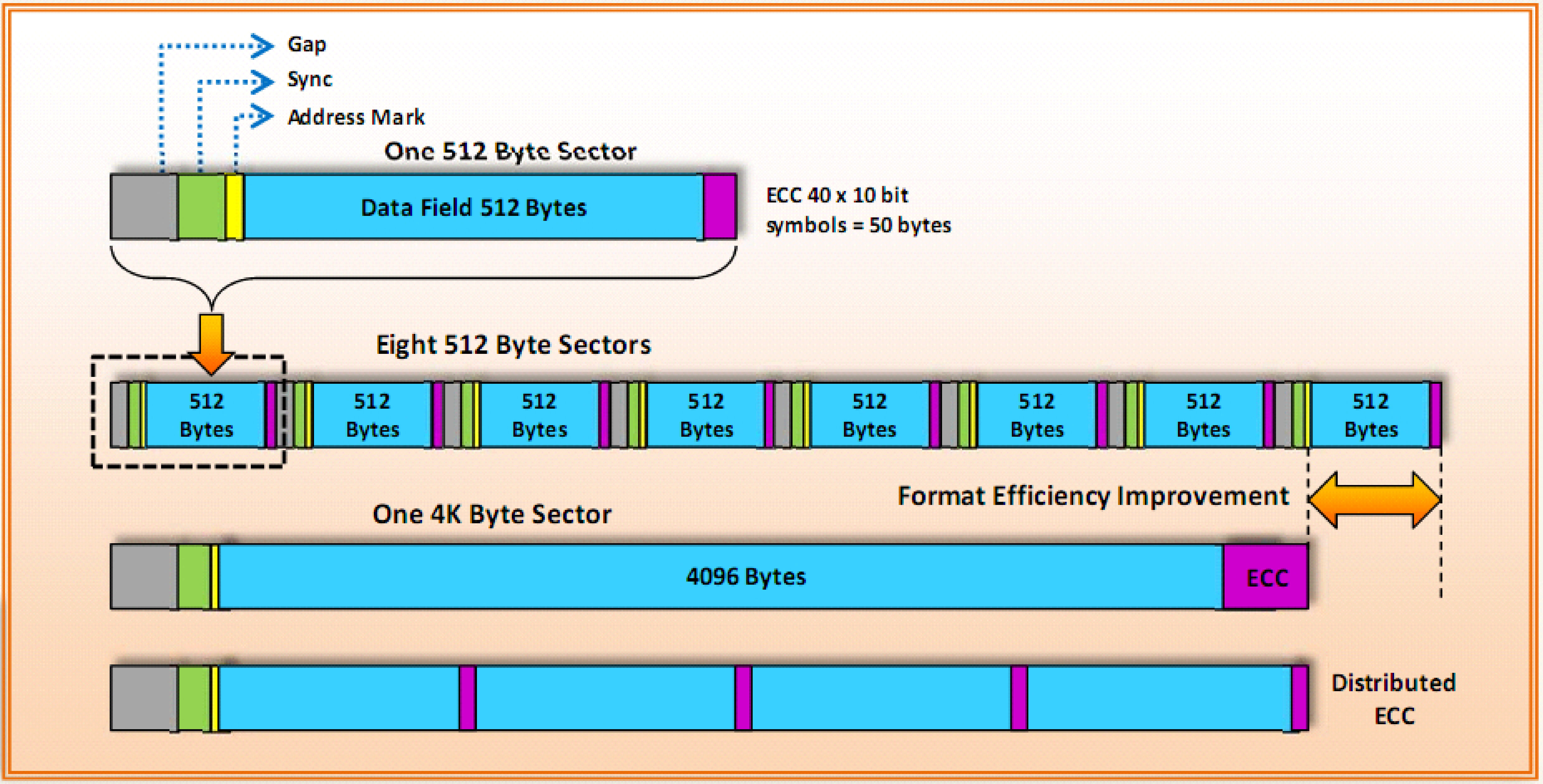
- A sector is the smallest **physical storage unit** on the disk platter
- It is almost always **512 bytes** in size and a few additional bytes for drive control and error correction
- Each disk sector is labelled using the **factory track-positioning data**
- The optimal method of storing a file on a disk is in a **contiguous series**
- For example, if the file size is 600 bytes, two 512 bytes sectors are allocated for the file

Sector Addressing:

- **Cylinders**, **heads** and **sectors** (CHS) determine the address of the individual sectors on the disk
- For example, on formatting a disk, **50 tracks** are divided into 10 sectors each
- Track and sector numbers are used by the **operating system** and **disk drive** to identify the stored information

Advanced Format: Sectors

- New hard drives use **4096 byte** (4 KB or 4K) advanced format sectors
- Generation-one Advanced Format also called as 4K sector technology, efficiently uses the **storage surface media** of a disk efficiently by merging eight 512 byte sectors into one single sector (4096 bytes)
- After merging, the structure of **4K sector** does not disturb the key design elements of the traditional 512-byte sector



Clusters

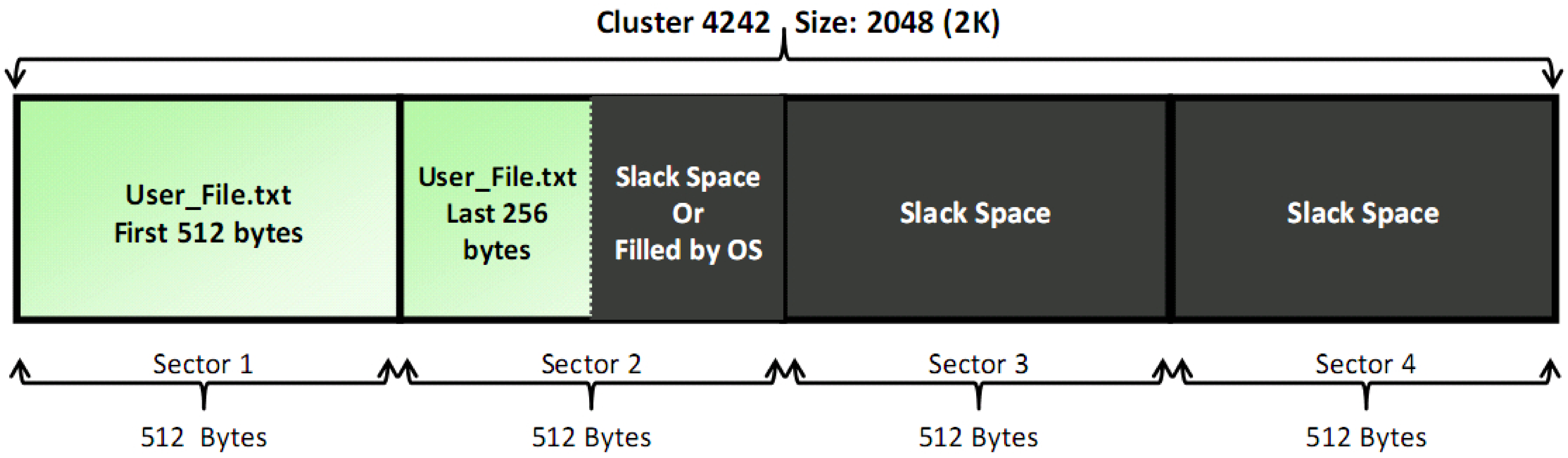
- A cluster is the **smallest logical storage unit** on a hard disk. It is a set of track sectors, ranging from **2 to 32** or more, depending on the formatting scheme in use
- The file system divides the storage on a disk volume into **discreet chunks of data** for efficient disk usage and performance. These chunks are called clusters
- The process by which files are allocated to clusters is called allocation, so clusters are also known as allocation units
- In the File Allocation Table (FAT) file system, the clusters linked with a file keep **track of file data** in the hard disk's file allocation table

Cluster Size

- Cluster sizing has a significant impact on the **performance of an operating system and disk utilization**
- Cluster size can be **altered** for optimum disk storage
- The size of a cluster depends on the **size of the disk partition** and type of file system installed on the partition
- Larger cluster size (greater than one sector):
  - Minimizes the **fragmentation** problem
  - Increases the probability of **unused space** in the cluster
  - Reduces **disk storage** area to save information
  - Reduces the **unused area** on the disk

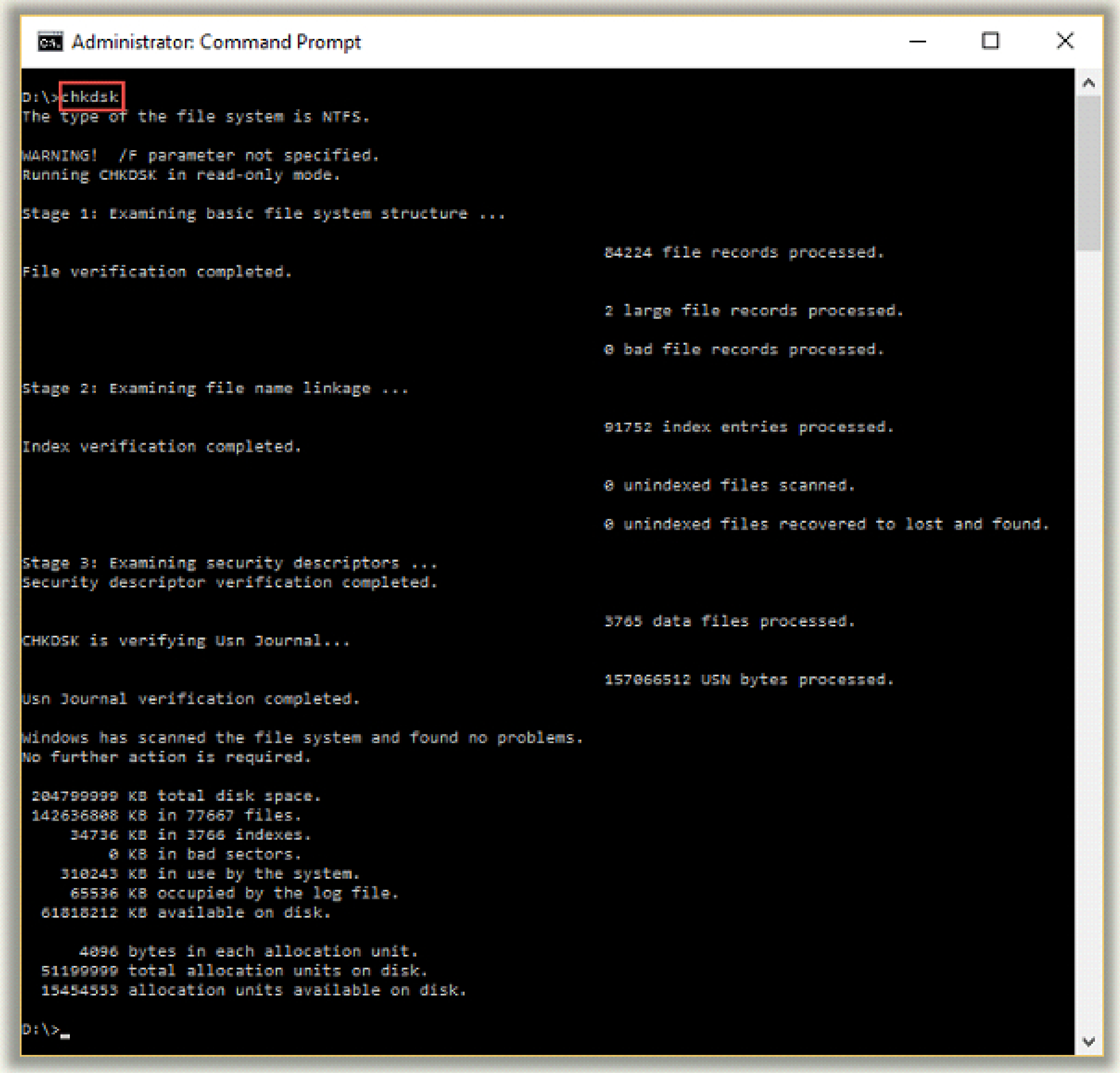
Slack Space

- Slack space is the area of a disk cluster between the **end of the file** and **the end of the cluster**
- If the file size is less than the cluster size, still a full cluster is assigned to that file. The remaining space remains unused and is called **slack space**. This remaining unused space is called slack space.
- For example, if the partition size is 4 GB, each cluster will be 32 KB. Even if a file requires only 10 KB, the entire 32 KB will be allocated to that file, resulting in 22 KB of slack space


Cluster 4242 Size: 2048 (2K)
User_File.txt First 512 bytes
User_File.txt Last 256 bytes
Slack Space Or Filled by OS
Slack Space
Slack Space
Sector 1
512 Bytes
Sector 2
512 Bytes
Sector 3
512 Bytes
Sector 4
512 Bytes

Lost Clusters

1. When the operating system marks clusters, as used, but does not allocate them to any file, such clusters are known as lost clusters
2. A lost cluster is a FAT file system error that results from in what manner the FAT file system allocates space and chains files together
3. It is mainly the result of a logical structure error and not a physical disk error
4. They usually occur because of interrupted file activities such as, 'the file is not correctly completed and closed' thus, the clusters have involved never linked correctly to a file
5. CHKDSK is a system tool in Windows, that authenticates the file system reliability of a volume and repairs logical file system errors



```
D:\>chkdsk
The type of the file system is NTFS.

WARNING!  /F parameter not specified.
Running CHKDSK in read-only mode.

Stage 1: Examining basic file system structure ...
                                                            84224 file records processed.
File verification completed.
                                                            2 large file records processed.

                                                            0 bad file records processed.

Stage 2: Examining file name linkage ...
                                                            91752 index entries processed.
Index verification completed.
                                                            0 unindexed files scanned.

                                                            0 unindexed files recovered to lost and found.

Stage 3: Examining security descriptors ...
Security descriptor verification completed.
                                                            3765 data files processed.
CHKDSK is verifying Usn Journal...
                                                            157066512 USN bytes processed.
Usn Journal verification completed.

Windows has scanned the file system and found no problems.
No further action is required.

 204799999 KB total disk space.
 142636808 KB in 77667 files.
     34736 KB in 3766 indexes.
         0 KB in bad sectors.
    310243 KB in use by the system.
     65536 KB occupied by the log file.
  61818212 KB available on disk.

      4096 bytes in each allocation unit.
  51199999 total allocation units on disk.
  15454553 allocation units available on disk.

D:\>_
```

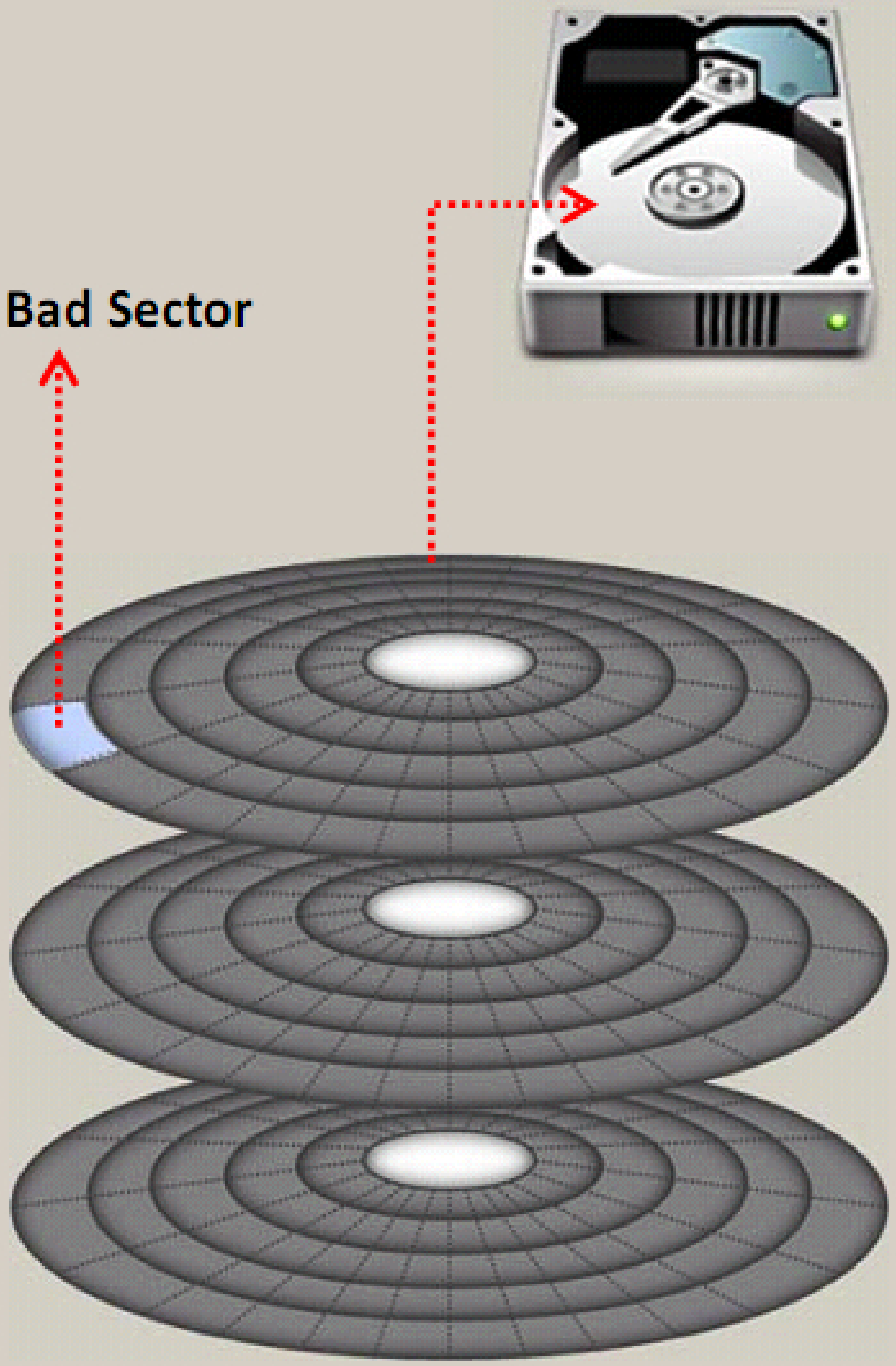
Bad Sectors

Bad sector is a **damaged portion of a disk** on which no read/write operation can be performed

Formatting a disk enables the operating system to **identify unusable sectors** and mark them as bad

Bad sectors are formed due to **configuration problems** or any physical disturbances to the disk

If data is in a sector that becomes bad, then it might not be recoverable
Data can be recovered using software tools such as Chkdsk



Understanding Bit, Nibble and Byte

Bit:

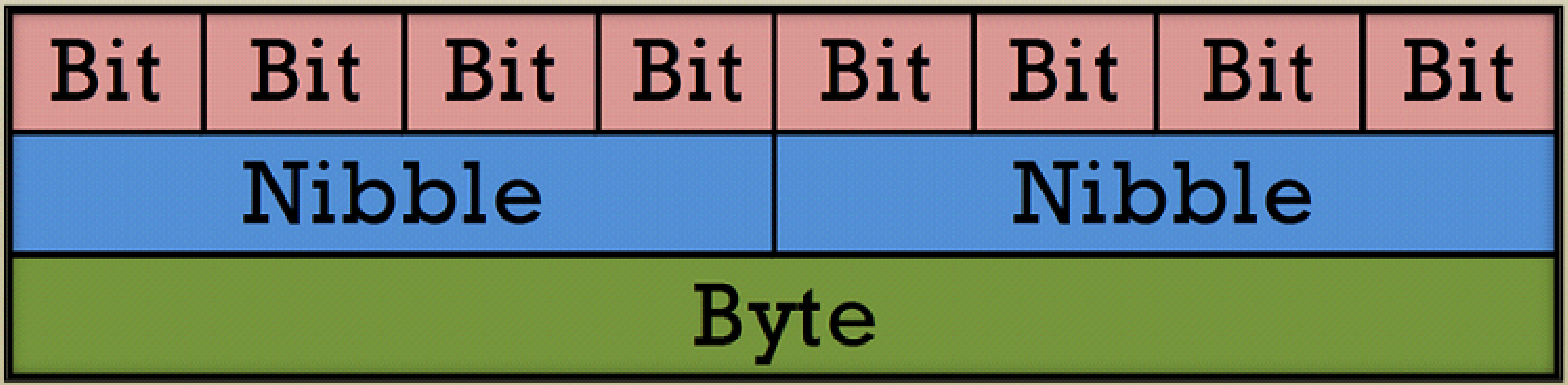
- Short for binary digit
- It is the smallest unit of data stored in a computer and is represented as a binary value, either 1 (true) or 0 (false)

Nibble:

- It is a group of 4 bits and is half the size of a Byte
- Not a common term as most microprocessors use group of 8 bits or higher to process data

Byte:

- It is a group of 8 bits and twice the size of a Nibble
- One single character typed from a keyboard takes one byte of storage



<table>
<tr><td>Bit</td><td>Bit</td><td>Bit</td><td>Bit</td><td>Bit</td><td>Bit</td><td>Bit</td><td>Bit</td></tr>
<tr><td colspan="4">Nibble</td><td colspan="4">Nibble</td></tr>
<tr><td colspan="8">Byte</td></tr>
</table>

Hard Disk Data Addressing

Hard disk data addressing is a method of **allotting addresses** to each physical block of data on a hard disk

Hard Disk Data Addressing Methods

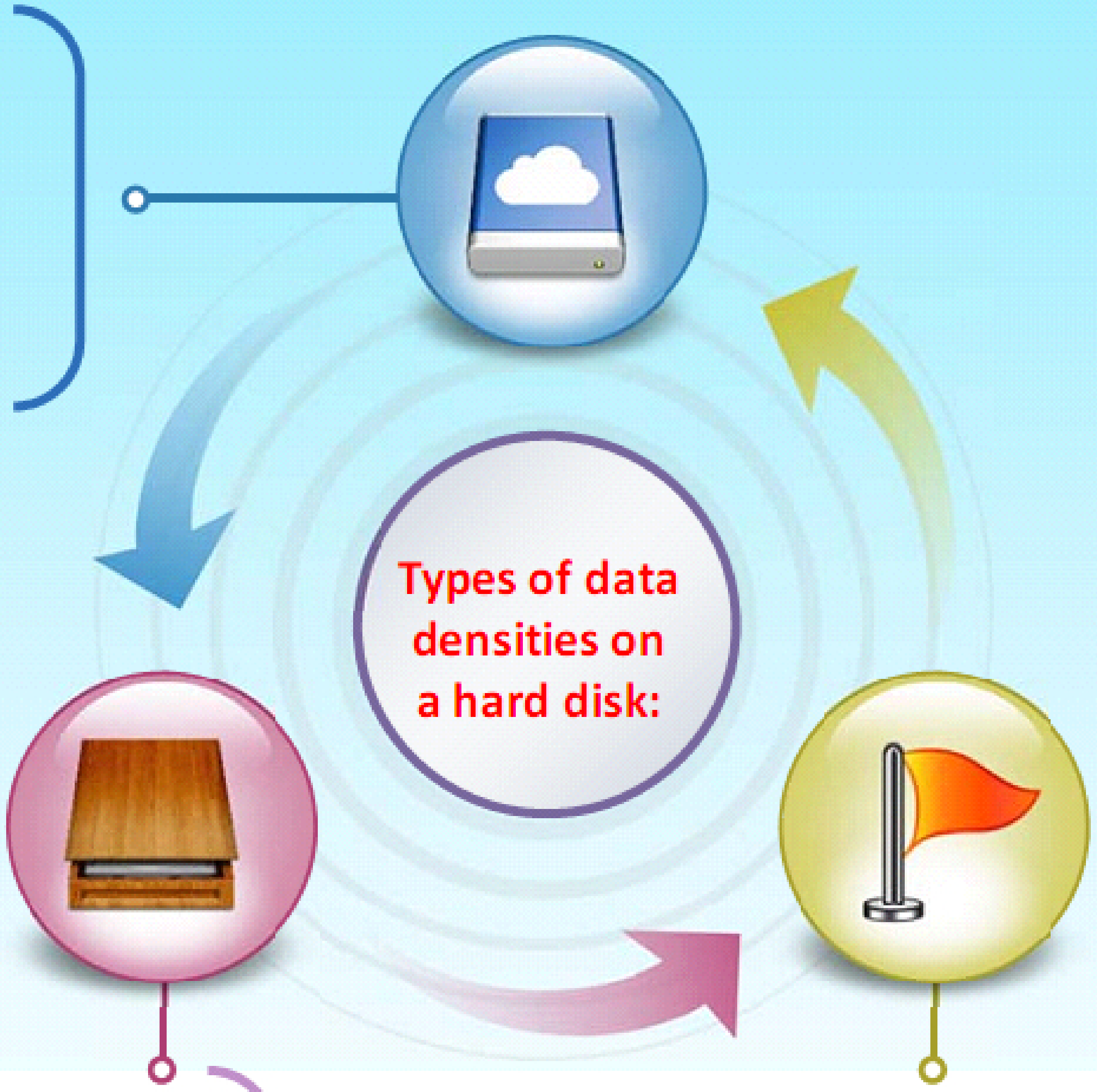
CHS (Cylinder-Head-Sector)

- It addresses data by simply specifying the **cylinder** (radius), **head** (platter side), and **sector** (angular position)
- It is used on most **IDE drives**

LBA (Logical Block Address)

- It addresses data by allotting a **sequential number** to each sector of the hard disk
- It is used on **SCSI** and enhanced **IDE drives**

Data Densities on a Hard Disk



Types of data densities on a hard disk:

Track Density

It is defined as the space between tracks on a disk

Areal Density

It is defined as the number of **bits per square inch** on a platter

Bit Density

It is the bits per unit **length** of track

- Data is recorded onto a hard disk using a method called **zoned bit recording** (also known as a multiple zone recording)
- In this technique, tracks are combined together into zones depending on their distance from the **center of the disk**
- Each zone is assigned a number of sectors per track

Disk Capacity Calculation

Disk Capacity Calculation Question?

A disk drive has 16,384 cylinders, 80 heads, and 63 sectors per track.

Assume - a sector has 512 bytes. What is the capacity of such a disk?

Answer

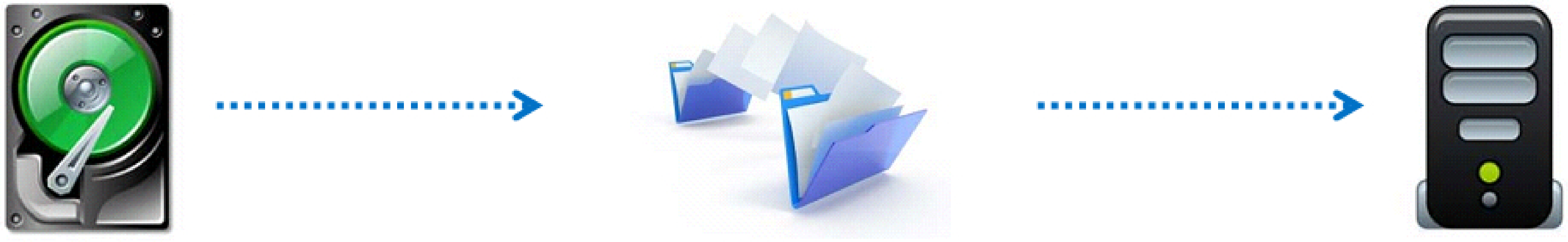
The conversion factors appropriate to this hard disk are:

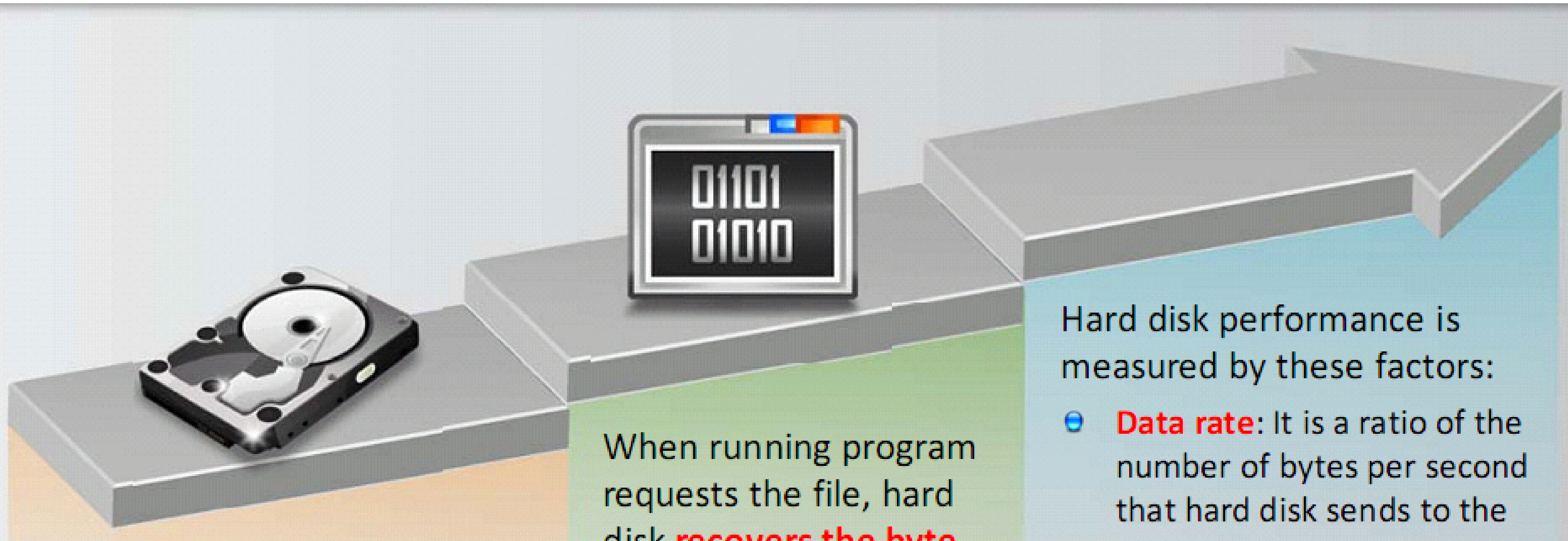
- 16,384 cylinders / disk
- 80 heads / cylinder
- 63 sectors / track
- 512 bytes / sector

Total bytes = 1 disk * (16,384 cylinders / disk) * (80 heads / cylinder) * (1 track / head) * (63 sectors / track) * (512 bytes / sector)

= 42,278,584,320 bytes

Measuring the Hard Disk Performance





Data is stored on the hard disk in the **form of files**

When running program requests the file, hard disk **recovers the byte content** of the file and sends them to the CPU one at a time for further processing

Hard disk performance is measured by these factors:

- **Data rate**: It is a ratio of the number of bytes per second that hard disk sends to the CPU
- **Seek time**: It is the amount of time required to send the first byte of the file to the CPU, when it requests the file

Disk Partitions

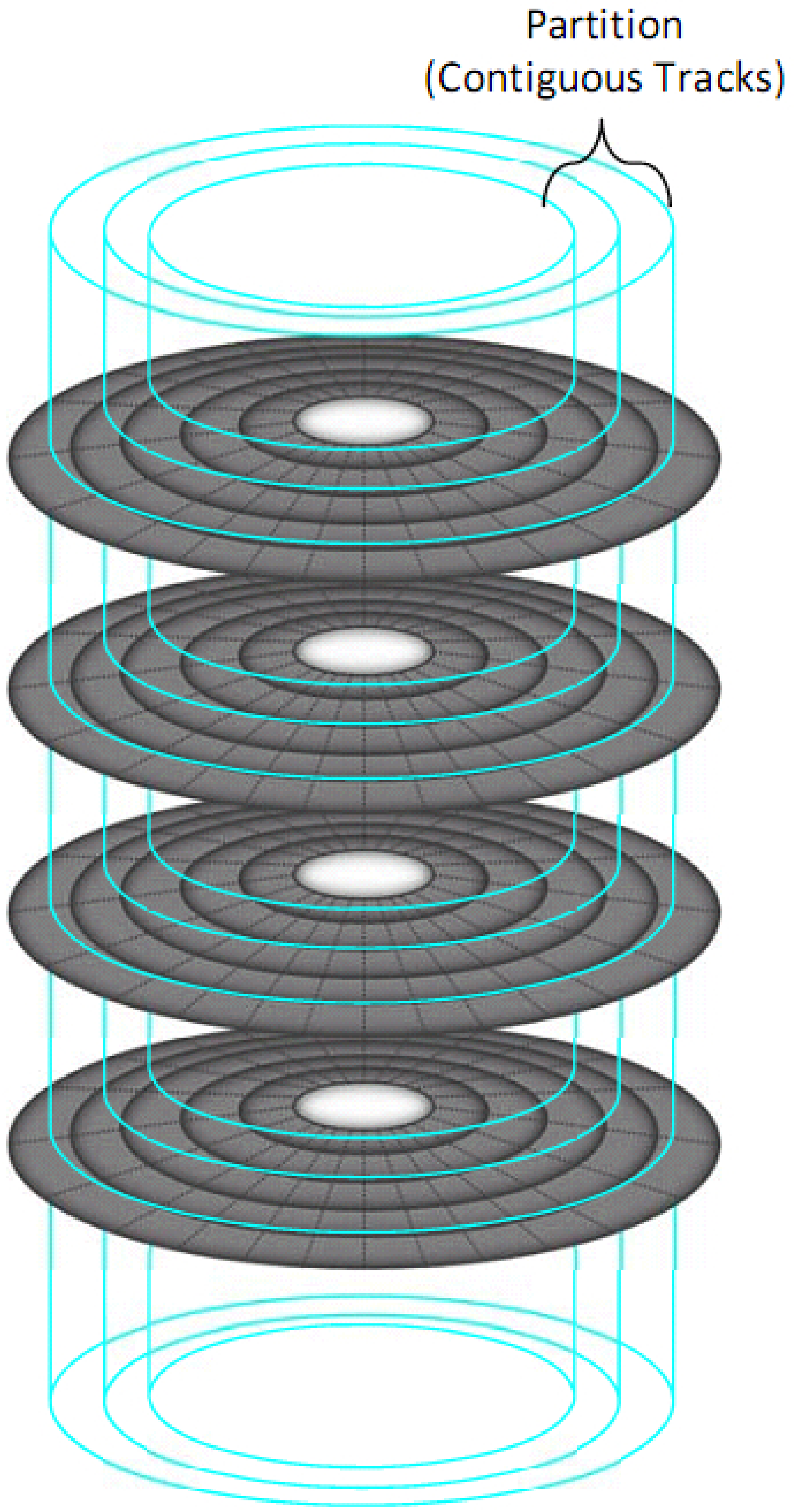
- The HDD partitioning is the **creation of logical divisions** upon a hard disk that allows user to apply operating system-specific logical formatting

Primary Partition

- It is a drive that holds the information regarding **operating system**, **system area**, and other information required for booting
- In MS-DOS and earlier versions of Microsoft Windows systems, the first partition (C:) must be a "primary partition"

Extended Partition

- It is the logical drive that holds the information regarding stored **data and files** in the disk



BIOS Parameter Block (BPB)

- The BIOS parameter block (BPB) is a data structure in the partition boot sector
- It describes the physical layout of a data storage volume, like the number of heads and the size of the tracks on the drive
- BPB in file systems such as FAT12 (except for in DOS 1.x), FAT16, FAT32, HPFS, and NTFS defines the filesystem structure
- The BPB length varies for FAT16, FAT32, and NTFS boot sectors, due to different types of fields and the amount of data stored in them
- BPB assists investigators to locate the file table on the hard drive

Format of full DOS 7.1 Extended BIOS Parameter Block (79 bytes) for FAT32:

Sector offset	BPB offset	Field length	Description
0x00B	0x00	25 BYTEs	DOS 3.31 BPB
0x024	0x19	DWORD	Logical sectors per FAT
0x028	0x1D	WORD	Mirroring flags etc.
0x02A	0x1F	WORD	Version
0x02C	0x21	DWORD	Root directory cluster
0x030	0x25	WORD	Location of FS Information Sector
0x032	0x27	WORD	Location of backup sector(s)
0x034	0x29	12 BYTEs	Reserved (Boot file name)
0x040	0x35	BYTE	Physical drive number
0x041	0x36	BYTE	Flags etc.
0x042	0x37	BYTE	Extended boot signature (0x29)
0x043	0x38	DWORD	Volume serial number
0x047	0x3C	11 BYTEs	Volume label
0x052	0x47	8 BYTEs	File-system type

NTFS - Format of Extended BPB for NTFS (73 bytes):

Sector offset	BPB offset	Field length	Description
0x00B	0x00	25 BYTEs	DOS 3.31 BPB
0x024	0x19	BYTE	Physical drive number (identical to DOS 3.4 EBPB)
0x025	0x1A	BYTE	Flags etc. (identical to DOS 3.4 EBPB)
0x026	0x1B	BYTE	Extended boot signature (0x80 aka "8.0") (similar to DOS 3.4 EBPB and DOS 4.0 EBPB)
0x027	0x1C	BYTE	Reserved
0x028	0x1D	QWORD	Sectors in volume
0x030	0x25	QWORD	MFT first cluster number
0x038	0x2D	QWORD	MFT mirror first cluster number
0x040	0x35	DWORD	MFT record size
0x044	0x39	DWORD	Index block size
0x048	0x3D	QWORD	Volume serial number
0x050	0x45	DWORD	Checksum

Master Boot Record (MBR)

I. A master boot record (MBR) is the **first sector** ("sector zero") of a **data storage device**, such as a hard disk

II. The information regarding the files on the disk, their location, size, and other important data is stored in the **MBR** file

III. In practice, MBR almost always refers to the **512-byte boot sector** or partition sector of a disk

IV. **MBR** is used for:

- **Holding a partition table** which refers to the partitions of a hard disk
- **Bootstrapping** an operating system
- Distinctively recognizing individual hard disk media with a **32-bit disk signature**

Structure of a Master Boot Record

- **Backing up the MBR**

  In UNIX/Linux, dd can be used to backup and restore the MBR

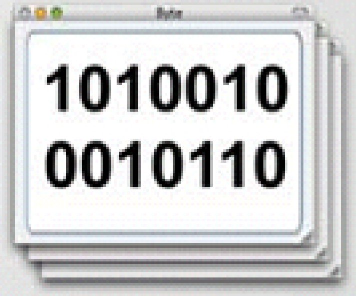
- **Backup the MBR**

  ```
  dd if=/dev/xxx of=mbr.backup bs=512 count=1
  ```

- **Restore the MBR**

  ```
  dd if=mbr.backup of=/dev/xxx bs=512 count=1
  ```

<table>
<tr><th colspan="3">Address</th><th colspan="2" rowspan="2">Description</th><th rowspan="2">Size in bytes</th></tr>
<tr><th>Hex</th><th>Oct</th><th>Dec</th></tr>
<tr><td>0000</td><td>0000</td><td>0</td><td colspan="2">Code Area</td><td>440
(max. 446)</td></tr>
<tr><td>01B8</td><td>0670</td><td>440</td><td colspan="2">Disk Signature (Optional)</td><td>4</td></tr>
<tr><td>01BC</td><td>0674</td><td>444</td><td colspan="2">Usually Nulls; 0x0000</td><td>2</td></tr>
<tr><td>01BE</td><td>0676</td><td>446</td><td colspan="2">Table of Primary Partitions
(Four 16-byte entries, IBM partition table scheme)</td><td>64</td></tr>
<tr><td>01FE</td><td>0776</td><td>510</td><td>55h</td><td rowspan="2">MBR Signature;
0xAA55</td><td rowspan="2">2</td></tr>
<tr><td>01FF</td><td>0777</td><td>511</td><td>AAh</td></tr>
<tr><td colspan="5">MBR, Total Size: 446 + 64 = 2 =</td><td>512</td></tr>
</table>



Structure of a Master Boot Record (Cont'd)

Layout of 16-byte Partition Record

Offset	Description
0x00	Status (0x80 = bootable, 0x00 = non-bootable, other = malformed)
0x01	**Cylinder-head-sector** address of the first sector in the partition
0x04	**Partition type**
0x05	Cylinder-head-sector address of the last sector in the partition
0x08	(4 bytes) **Logical block address** of the first sector in the partition
0x0C	(4 bytes) Length of the partition, in sectors

Layout of IBM Extended Partition Record

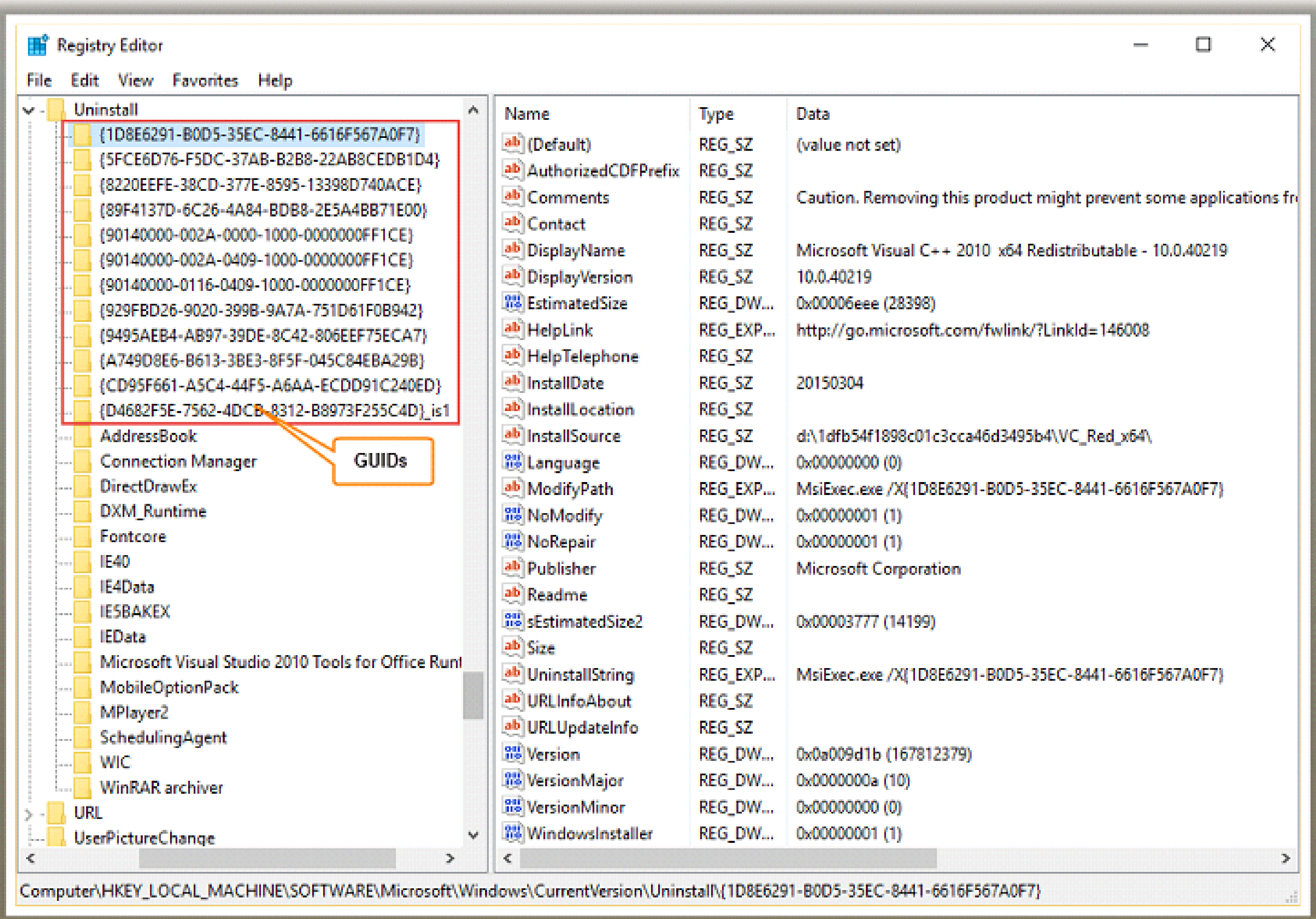
Offset	Description
0x00	Status bits (bit 0 = list on Boot Manager menu, other bits = reserved)
0x01	Space-padded partition name

Globally Unique Identifier (GUID)

- Global Unique Identifier (GUID) is a 128-bit **unique reference number** used as an identifier in computer software
- In general, GUIDs are displayed as **32 hexadecimal digits** with groups separated by hyphens

Common Uses:

- In Windows registry, GUIDs are used to identify COM DLLs
- In database tables, GUIDs are used as primary key values
- Website assigns GUID to a user's browser to record and track the session
- Windows assigns GUID to a username to identify user accounts



Uninstall
{1D8E6291-B0D5-35EC-8441-6616F567A0F7}
{5FCE6D76-F5DC-37AB-B2B8-22AB8CEDB1D4}
{8220EEFE-38CD-377E-8595-13398D740ACE}
{89F4137D-6C26-4A84-BDB8-2E5A4BB71E00}
{90140000-002A-0000-1000-0000000FF1CE}
{90140000-002A-0409-1000-0000000FF1CE}
{90140000-0116-0409-1000-0000000FF1CE}
{929FBD26-9020-399B-9A7A-751D61F0B942}
{9495AEB4-AB97-39DE-8C42-806EEF75ECA7}
{A749D8E6-B613-3BE3-8F5F-045C84EBA29B}
{CD95F661-A5C4-44F5-A6AA-ECDD91C240ED}
{D4682F5E-7562-4DCB-8312-B8973F255C4D}_is1
AddressBook
Connection Manager
DirectDrawEx
DXM_Runtime
Fontcore
IE40
IE4Data
IE5BAKEX
IEData
Microsoft Visual Studio 2010 Tools for Office Runt
MobileOptionPack
MPlayer2
SchedulingAgent
WIC
WinRAR archiver
URL
UserPictureChange

Name	Type	Data
(Default)	REG_SZ	(value not set)
AuthorizedCDFPrefix	REG_SZ	
Comments	REG_SZ	Caution. Removing this product might prevent some applications fr
Contact	REG_SZ	
DisplayName	REG_SZ	Microsoft Visual C++ 2010 x64 Redistributable - 10.0.40219
DisplayVersion	REG_SZ	10.0.40219
EstimatedSize	REG_DW...	0x00006eee (28398)
HelpLink	REG_EXP...	http://go.microsoft.com/fwlink/?LinkId=146008
HelpTelephone	REG_SZ	
InstallDate	REG_SZ	20150304
InstallLocation	REG_SZ	
InstallSource	REG_SZ	d:\1dfb54f1898c01c3cca46d3495b4\VC_Red_x64\
Language	REG_DW...	0x00000000 (0)
ModifyPath	REG_EXP...	MsiExec.exe /X{1D8E6291-B0D5-35EC-8441-6616F567A0F7}
NoModify	REG_DW...	0x00000001 (1)
NoRepair	REG_DW...	0x00000001 (1)
Publisher	REG_SZ	Microsoft Corporation
Readme	REG_SZ	
sEstimatedSize2	REG_DW...	0x00003777 (14199)
Size	REG_SZ	
UninstallString	REG_EXP...	MsiExec.exe /X{1D8E6291-B0D5-35EC-8441-6616F567A0F7}
URLInfoAbout	REG_SZ	
URLUpdateInfo	REG_SZ	
Version	REG_DW...	0x0a009d1b (167812379)
VersionMajor	REG_DW...	0x0000000a (10)
VersionMinor	REG_DW...	0x00000000 (0)
WindowsInstaller	REG_DW...	0x00000001 (1)

Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\{1D8E6291-B0D5-35EC-8441-6616F567A0F7}

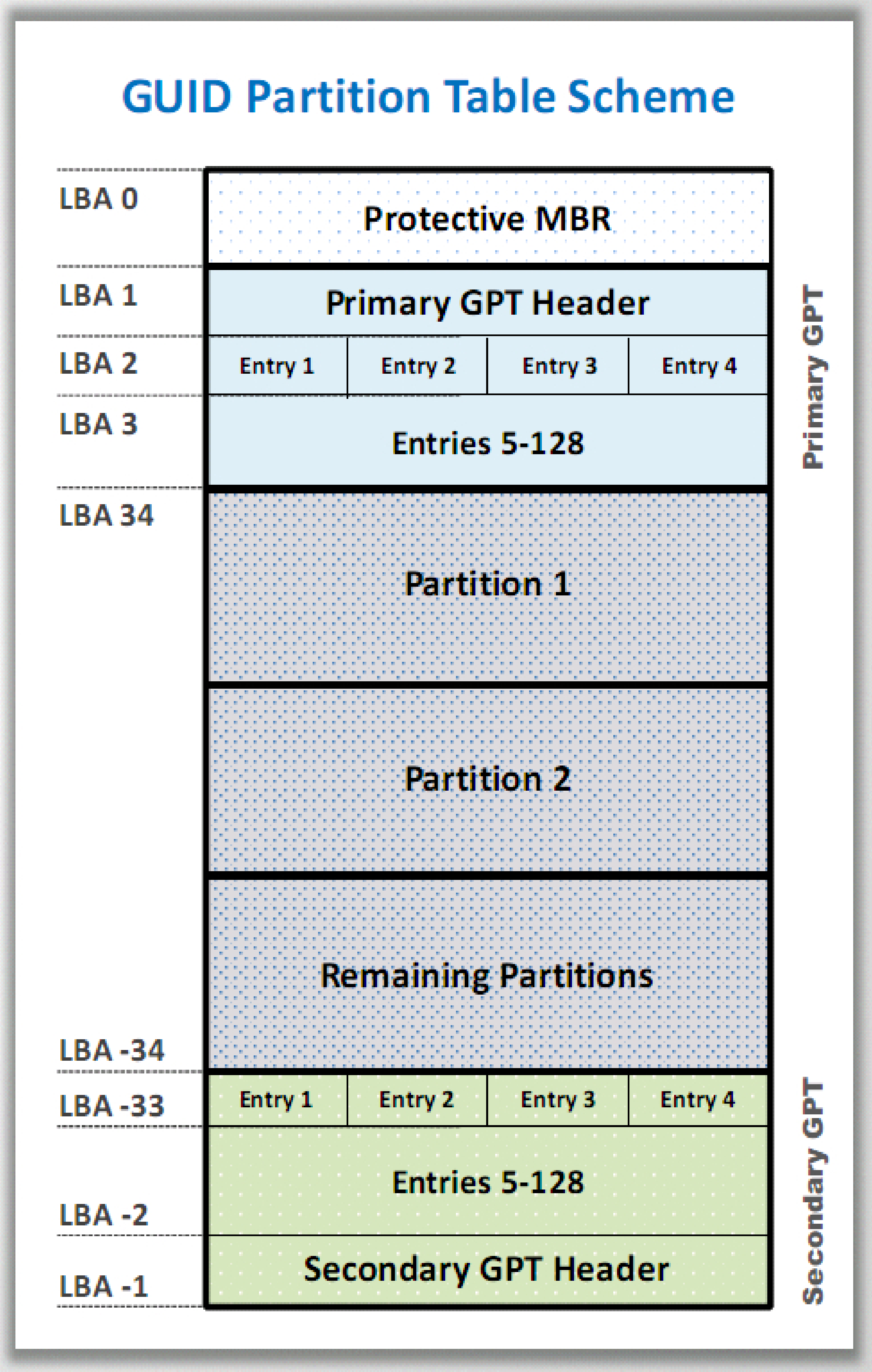
GUID Partition Table (GPT)

- **Unified Extensible Firmware Interface** (UEFI) replaces legacy **BIOS firmware** interfaces
- UEFI is a specification that defines a **software interface** between an OS and platform firmware
- It uses a partition system known as **GUID Partition Table** (GPT) that replaces the traditional **MBR**

Advantages of GPT disk layout:

- Supports up to 128 partitions and uses 64-bit **Logical Block Addresses** (LBAs)
- Supports maximum **partition size** from 2 Tebibyte (TiB) to 8 Zebibyte (ZiB)
- Provides **primary** and **backup partition tables** for redundancy

http://www.invoke-ir.com



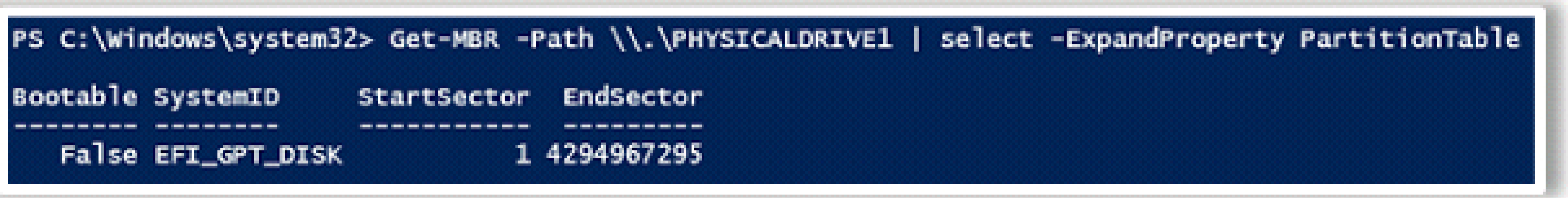
GUID Partition Table (GPT) (Cont'd)

Protective MBR:

- Disk formatted with a GPT disk layout has a **Protective MBR** located at **Logical Block Address** (LBA) 0
- Protective MBR provides compatibility with **legacy tools** that fail to understand the GPT format
- It is alike to the "**legacy**" **MBR** in functionality, but has only one partition of type 0xEE (EFI_GPT_DISK)
- This partition reserves the entire disk for the formal **GUID Partition Table** structure

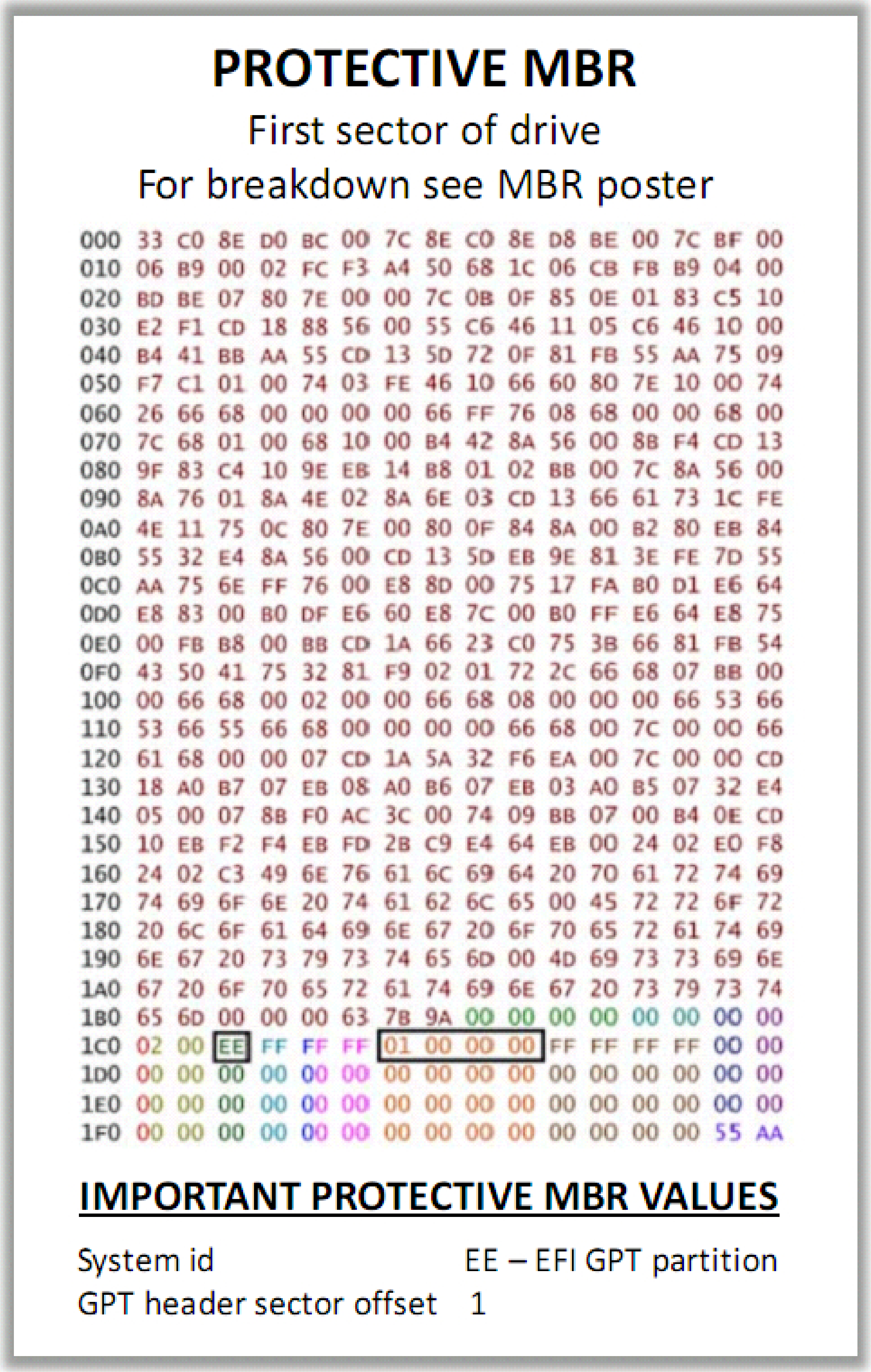
Note: The **UEFI Firmware** does not execute the MBR Boot Code (the first 440 bytes)

- The **Get-MBR** cmdlet displays the MBR Partition Table of a GPT formatted disk



```
PS C:\Windows\system32> Get-MBR -Path \\.\PHYSICALDRIVE1 | select -ExpandProperty PartitionTable

Bootable SystemID      StartSector  EndSector
-------- --------      -----------  ---------
   False EFI_GPT_DISK            1 4294967295
```

http://www.invoke-ir.com



PROTECTIVE MBR

First sector of drive

For breakdown see MBR poster

```
000  33 C0 8E D0 BC 00 7C 8E C0 8E D8 BE 00 7C BF 00
010  06 B9 00 02 FC F3 A4 50 68 1C 06 CB FB B9 04 00
020  BD BE 07 80 7E 00 00 7C 0B 0F 85 0E 01 83 C5 10
030  E2 F1 CD 18 88 56 00 55 C6 46 11 05 C6 46 10 00
040  B4 41 BB AA 55 CD 13 5D 72 0F 81 FB 55 AA 75 09
050  F7 C1 01 00 74 03 FE 46 10 66 60 80 7E 10 00 74
060  26 66 68 00 00 00 00 66 FF 76 08 68 00 00 68 00
070  7C 68 01 00 68 10 00 B4 42 8A 56 00 8B F4 CD 13
080  9F 83 C4 10 9E EB 14 B8 01 02 BB 00 7C 8A 56 00
090  8A 76 01 8A 4E 02 8A 6E 03 CD 13 66 61 73 1C FE
0A0  4E 11 75 0C 80 7E 00 80 0F 84 8A 00 B2 80 EB 84
0B0  55 32 E4 8A 56 00 CD 13 5D EB 9E 81 3E FE 7D 55
0C0  AA 75 6E FF 76 00 E8 8D 00 75 17 FA B0 D1 E6 64
0D0  E8 83 00 B0 DF E6 60 E8 7C 00 B0 FF E6 64 E8 75
0E0  00 FB B8 00 BB CD 1A 66 23 C0 75 3B 66 81 FB 54
0F0  43 50 41 75 32 81 F9 02 01 72 2C 66 68 07 BB 00
100  00 66 68 00 02 00 00 66 68 08 00 00 00 66 53 66
110  53 66 55 66 68 00 00 00 00 66 68 00 7C 00 00 66
120  61 68 00 00 07 CD 1A 5A 32 F6 EA 00 7C 00 00 CD
130  18 A0 B7 07 EB 08 A0 B6 07 EB 03 A0 B5 07 32 E4
140  05 00 07 8B F0 AC 3C 00 74 09 BB 07 00 B4 0E CD
150  10 EB F2 F4 EB FD 2B C9 E4 64 EB 00 24 02 E0 F8
160  24 02 C3 49 6E 76 61 6C 69 64 20 70 61 72 74 69
170  74 69 6F 6E 20 74 61 62 6C 65 00 45 72 72 6F 72
180  20 6C 6F 61 64 69 6E 67 20 6F 70 65 72 61 74 69
190  6E 67 20 73 79 73 74 65 6D 00 4D 69 73 73 69 6E
1A0  67 20 6F 70 65 72 61 74 69 6E 67 20 73 79 73 74
1B0  65 6D 00 00 00 63 7B 9A 00 00 00 00 00 00 00 00
1C0  02 00 EE FF FF FF 01 00 00 00 FF FF FF FF 00 00
1D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 AA
```

IMPORTANT PROTECTIVE MBR VALUES

System id	EE – EFI GPT partition
GPT header sector offset	1

GUID Partition Table (GPT) (Cont'd)

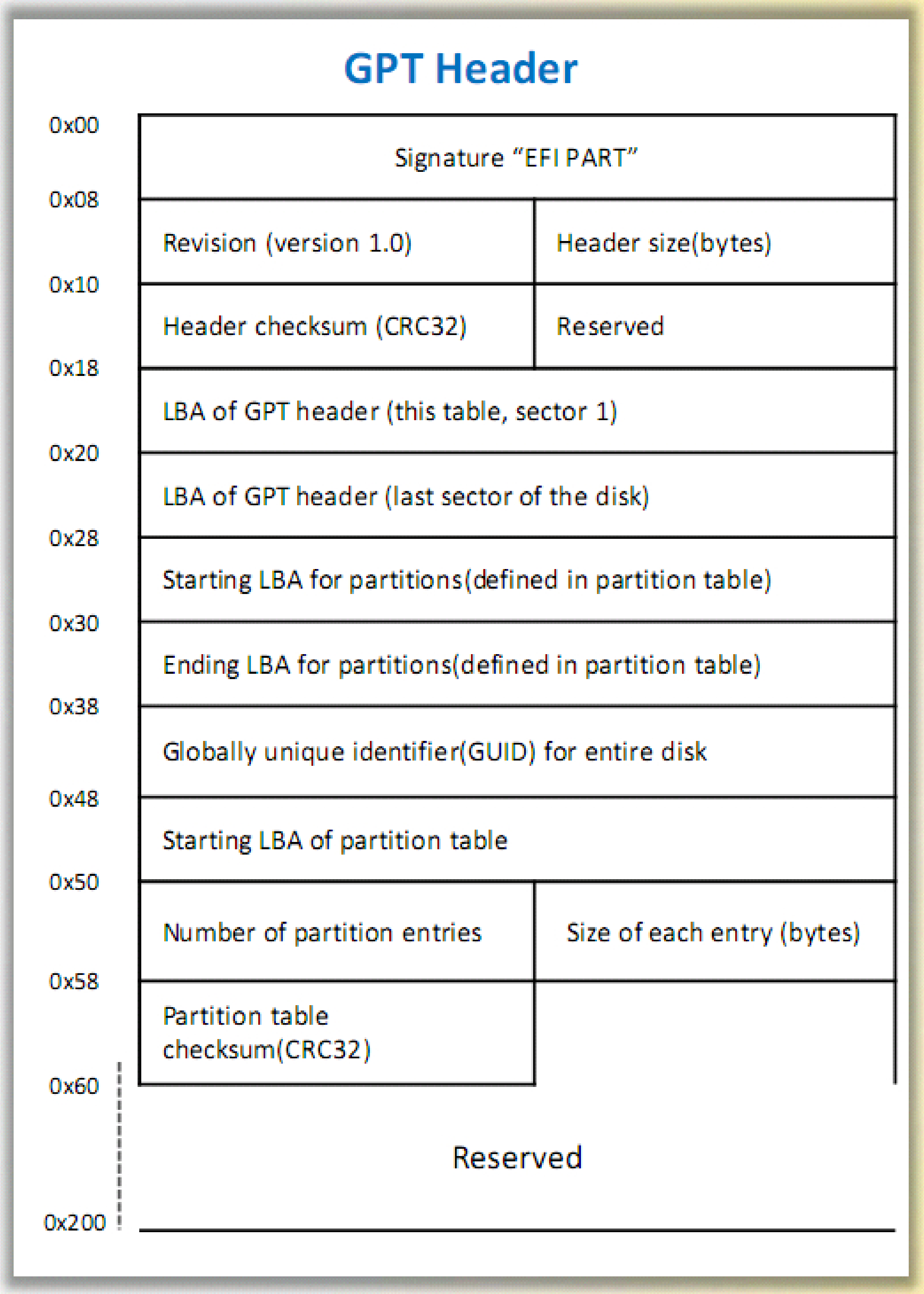
GUID Partition Table:

- The formal GUID Partition Table starts at LBA 1 where the GPT Header is found

GPT Header:

- It is a pointer to the partition table and defines the complete logical layout
- Contains information such as the "EFI PART" signature and a unique GUID of the disk
- The firmware detects GPT corruption using the CRC32 values
- The MyLBA value, always 1, defines the location of GPT header, while the AlternateLBA value represents the backup GPT and will occupy the last sector on a disk
- The backup GPT replaces the original GPT when it is corrupted
- The FirstUsableLBA and LastUsableLBA values point to the disk portion the partitions can use
- The PartitionEntryLBA value represents start of the array, while the NumberOfPartitionEntries and SizeOfPartitionEntry values denote the overall partition size

http://www.invoke-ir.com



GPT Header

0x00	Signature "EFI PART"	
0x08	Revision (version 1.0)	Header size(bytes)
0x10	Header checksum (CRC32)	Reserved
0x18	LBA of GPT header (this table, sector 1)	
0x20	LBA of GPT header (last sector of the disk)	
0x28	Starting LBA for partitions(defined in partition table)	
0x30	Ending LBA for partitions(defined in partition table)	
0x38	Globally unique identifier(GUID) for entire disk	
0x48	Starting LBA of partition table	
0x50	Number of partition entries	Size of each entry (bytes)
0x58	Partition table checksum(CRC32)	
0x60	Reserved	
0x200		

GUID Partition Table (GPT) (Cont'd)

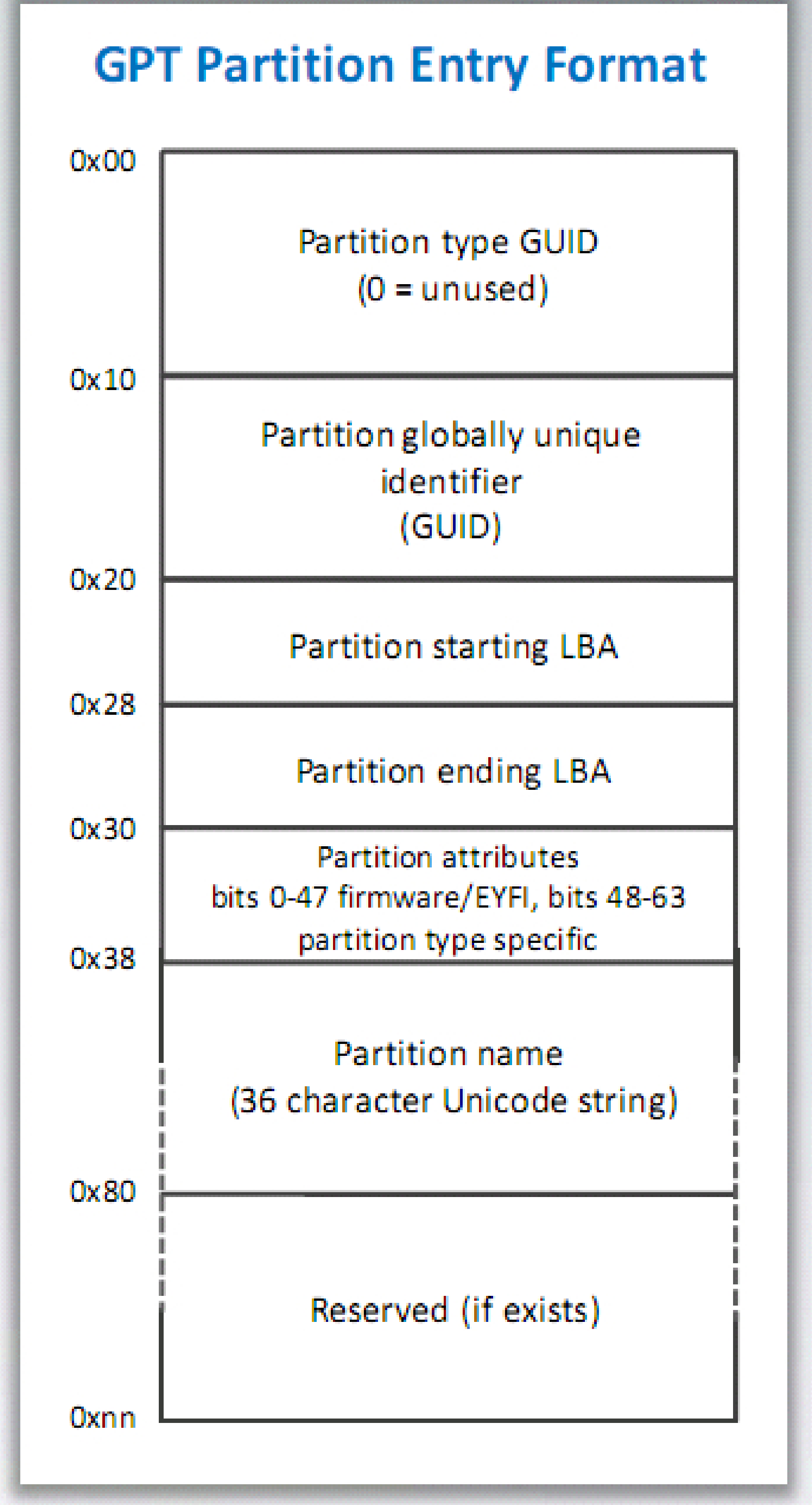
GPT Partition Array:

- The GPT Header points to the Partition Array via the **PartitionEntryLBA** value
- The size and number of partitions are defined in the **GPT Header**

Each partition contains:

- Two GUIDs, one represents the type of partition, and the second uniquely identifies the partition
- Partition **StartingLBA** and **EndingLBA** values describing the location and size of the partition
- Partition type specific attributes
- 36 character user-defined partition name

http://www.invoke-ir.com

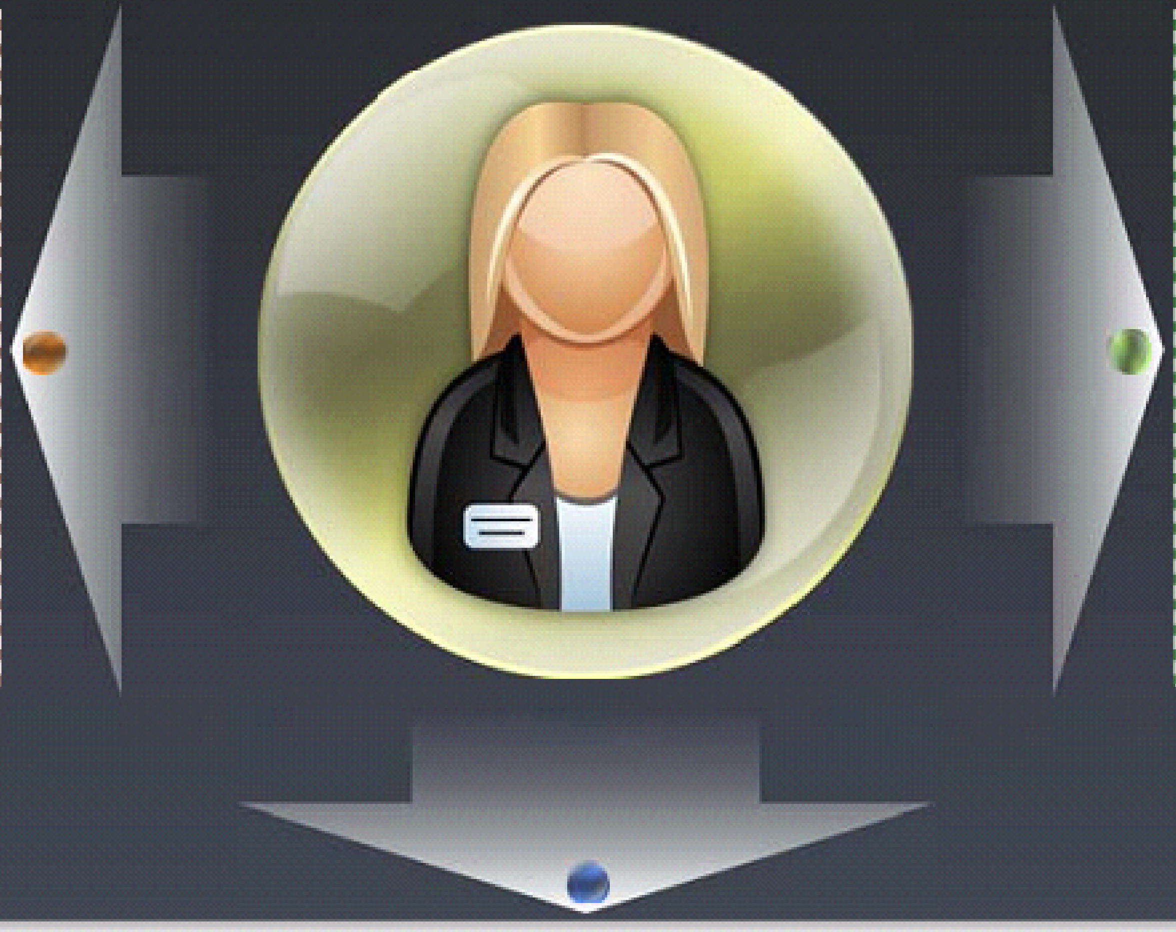


GPT Partition Entry Format

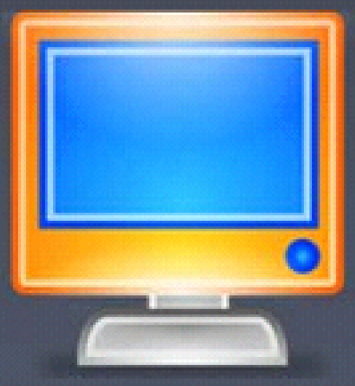
0x00	Partition type GUID (0 = unused)
0x10	Partition globally unique identifier (GUID)
0x20	Partition starting LBA
0x28	Partition ending LBA
0x30	Partition attributes bits 0-47 firmware/EYFI, bits 48-63 partition type specific
0x38	Partition name (36 character Unicode string)
0x80	Reserved (if exists)
0xnn	

What is the Booting Process?

Booting refers to the process of **starting or resetting operating systems** when the user turns on a computer system



It **loads the operating system** (stored in the hard disk) to the RAM (working memory)



Types of Booting

- **Cold boot** **(Hard boot)**

  It is the process of starting a computer from a powered-down or **off** state

- **Warm boot** **(Soft boot)**

  It is the process of restarting a computer that is already turned on through the operating system

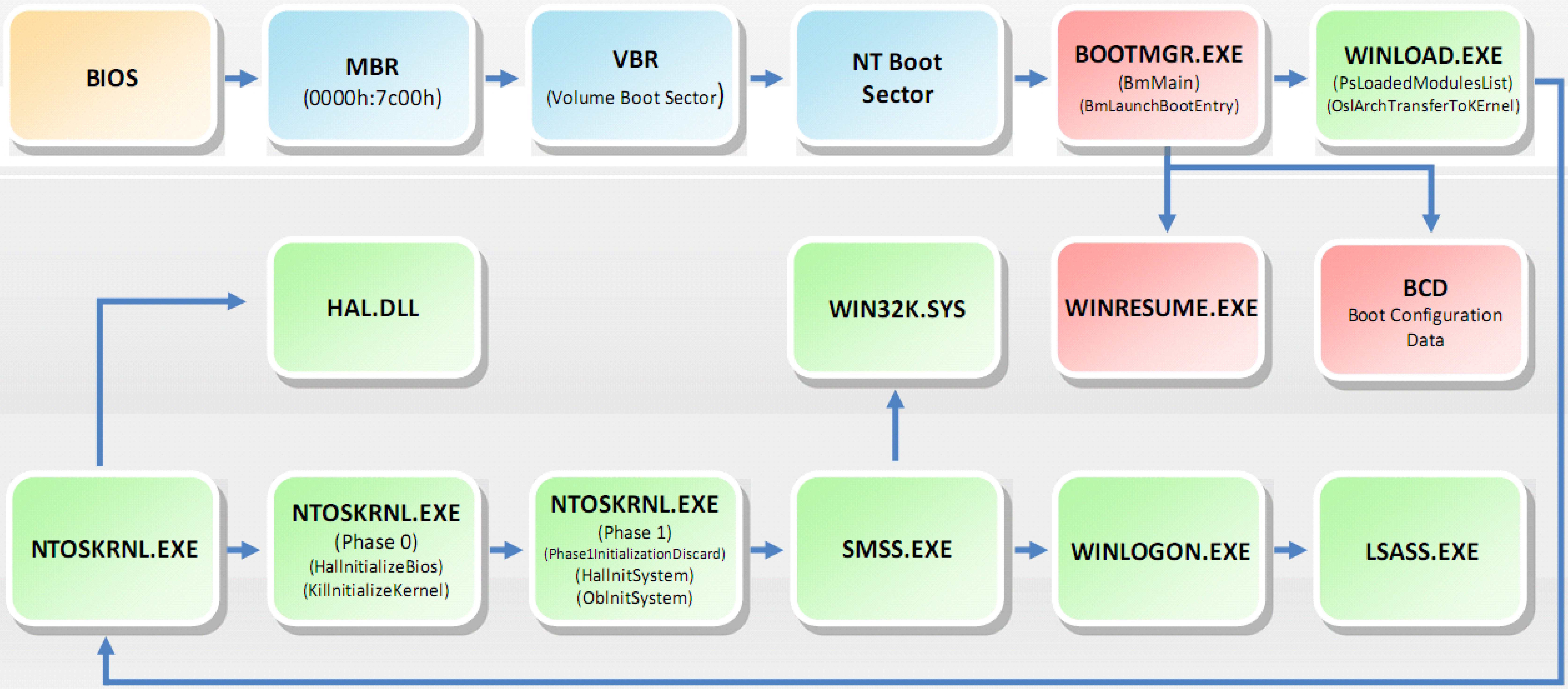
Essential Windows System Files

File Names	Description
Ntoskrnl.exe	Executive and kernel
Ntkrnlpa.exe	Executive and kernel with support for Physical Address Extension (PAE)
Hal.dll	Hardware abstraction layer
Win32k.sys	Kernel-mode part of the Win32 subsystem
Ntdll.dll	Internal support functions and system service dispatch stubs to executive functions
Kernel32.dll	Win32 subsystem DLL files
Advapi32.dll	
User32.dll	
Gdi32.dll	

Windows Boot Process

Windows XP, Vista, and 7 OSs power on and start up using the traditional BIOS-MBR method. Whereas, OSs from Windows 8 and above uses either traditional BIOS-MBR method or newer UEFI-GPT method according to the user choice

BIOS-MBR method



BIOS

MBR (0000h:7c00h)

VBR (Volume Boot Sector)

NT Boot Sector

BOOTMGR.EXE (BmMain) (BmLaunchBootEntry)

WINLOAD.EXE (PsLoadedModulesList) (OslArchTransferToKErnel)

WINRESUME.EXE

BCD Boot Configuration Data

HAL.DLL

WIN32K.SYS

NTOSKRNL.EXE

NTOSKRNL.EXE (Phase 0) (HalInitializeBios) (KiInitializeKernel)

NTOSKRNL.EXE (Phase 1) (Phase1InitializationDiscard) (HalInitSystem) (ObInitSystem)

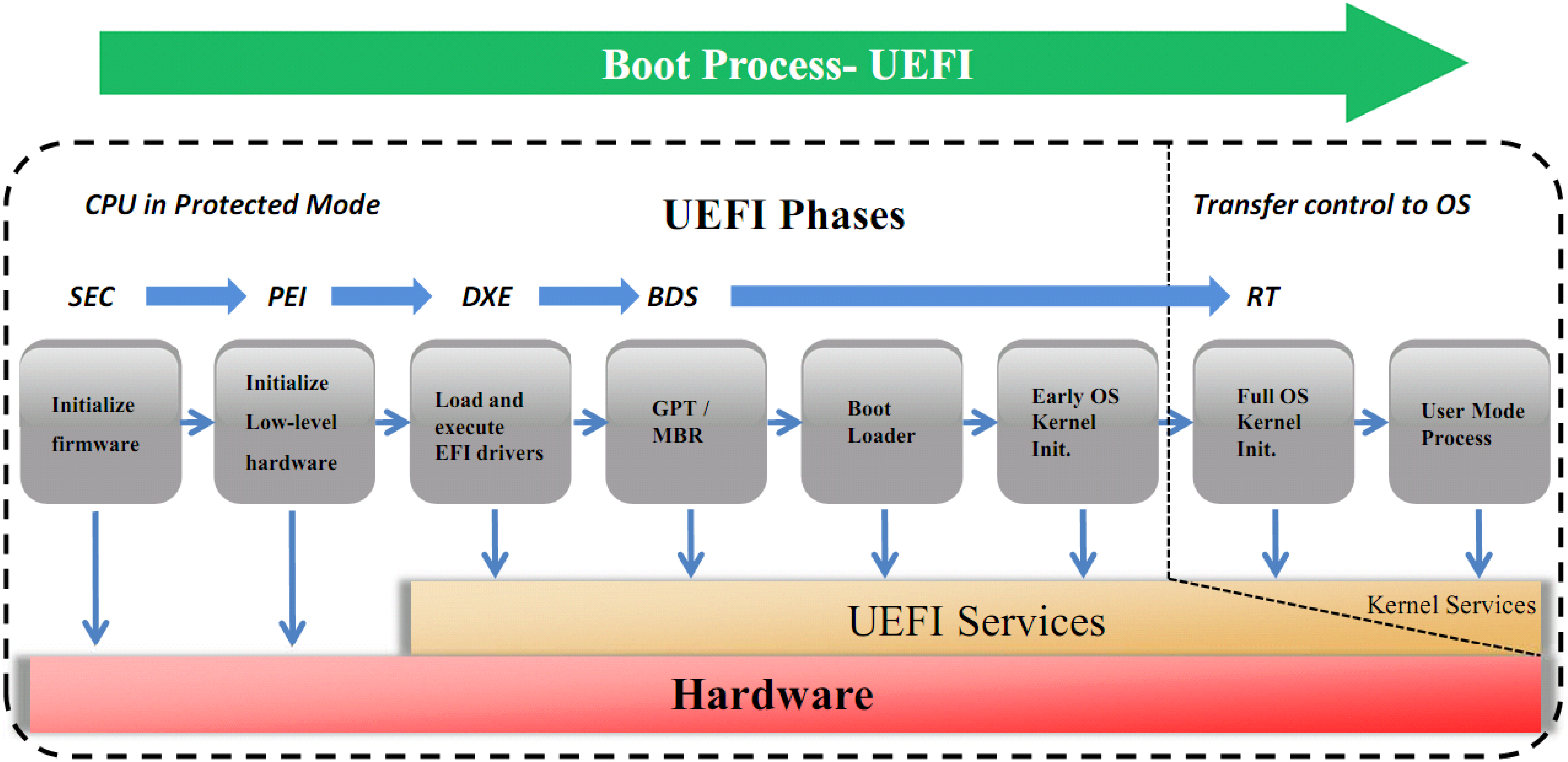
SMSS.EXE

WINLOGON.EXE

LSASS.EXE

http://breaking-the-system.blogspot.in

Windows Boot Process (Cont'd)



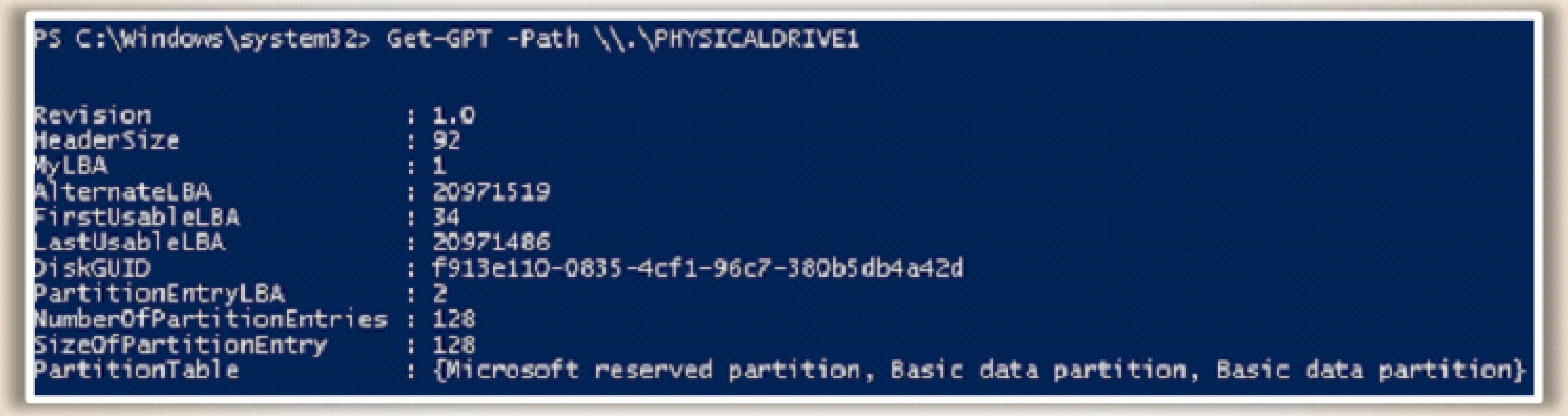
http://answers.microsoft.com

Identifying GUID Partition Table (GPT)

Investigators can use cmdlets given below in Windows PowerShell to identify the presence of GPT:

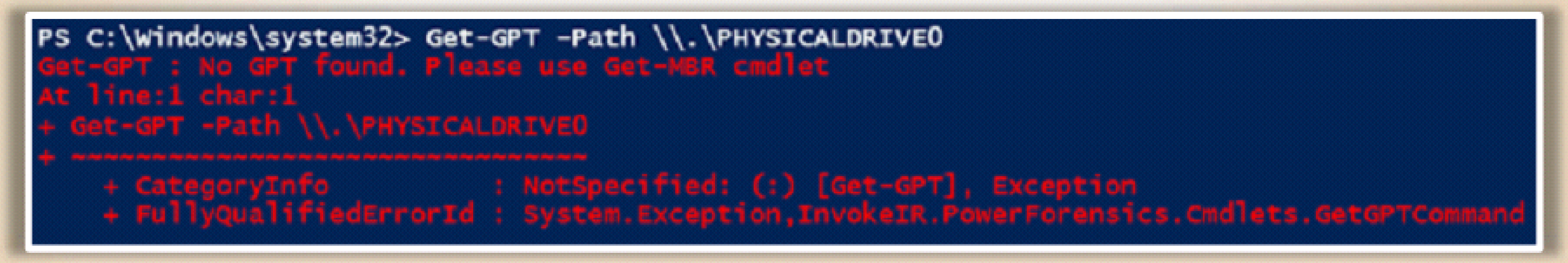
Get-GPT

- It parses the GPT data structure contained within the first few sectors of the device specified
- It requires the use of the -Path parameter which takes the Win32 Device Namespace (ex. \\.\PHYSICALDRIVE1) for the device from which the GPT should be parsed



```
PS C:\Windows\system32> Get-GPT -Path \\.\PHYSICALDRIVE1


Revision                 : 1.0
HeaderSize               : 92
MyLBA                    : 1
AlternateLBA             : 20971519
FirstUsableLBA           : 34
LastUsableLBA            : 20971486
DiskGUID                 : f913e110-0835-4cf1-96c7-380b5db4a42d
PartitionEntryLBA        : 2
NumberOfPartitionEntries : 128
SizeOfPartitionEntry     : 128
PartitionTable           : {Microsoft reserved partition, Basic data partition, Basic data partition}
```

- If Get-GPT is run against a disk formatted with a MBR, it will throw an error prompting to use Get-MBR instead



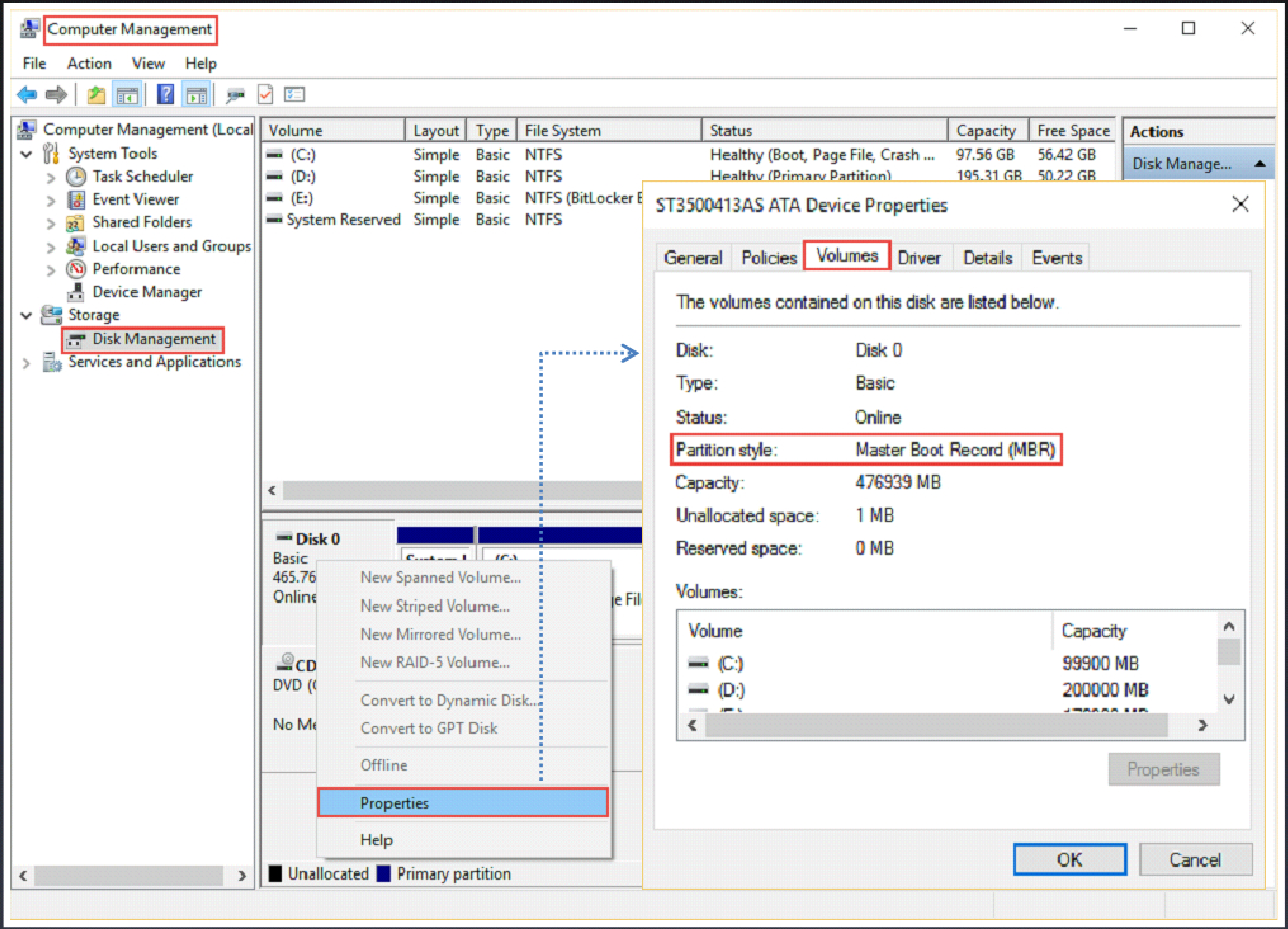
```
PS C:\Windows\system32> Get-GPT -Path \\.\PHYSICALDRIVE0
Get-GPT : No GPT found. Please use Get-MBR cmdlet
At line:1 char:1
+ Get-GPT -Path \\.\PHYSICALDRIVE0
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (:) [Get-GPT], Exception
    + FullyQualifiedErrorId : System.Exception,InvokeIR.PowerForensics.Cmdlets.GetGPTCommand
```

Alternate Method:

- Open “**Computer Management**” application and click “**Disk Management**” on the left pane. Right-click on the primary disk (here, Disk 0) and then click **Properties**.
- In the Device Properties window, click “**Volumes**” tab to see the **Partition style**

http://www.invoke-ir.com

Identifying GUID Partition Table (GPT) (Cont'd)



Volume	Layout	Type	File System	Status	Capacity	Free Space
(C:)	Simple	Basic	NTFS	Healthy (Boot, Page File, Crash ...	97.56 GB	56.42 GB
(D:)	Simple	Basic	NTFS	Healthy (Primary Partition)	195.31 GB	50.22 GB
(E:)	Simple	Basic	NTFS (BitLocker E			
System Reserved	Simple	Basic	NTFS			

ST3500413AS ATA Device Properties

The volumes contained on this disk are listed below.

Disk:	Disk 0
Type:	Basic
Status:	Online
Partition style:	Master Boot Record (MBR)
Capacity:	476939 MB
Unallocated space:	1 MB
Reserved space:	0 MB

Volumes:

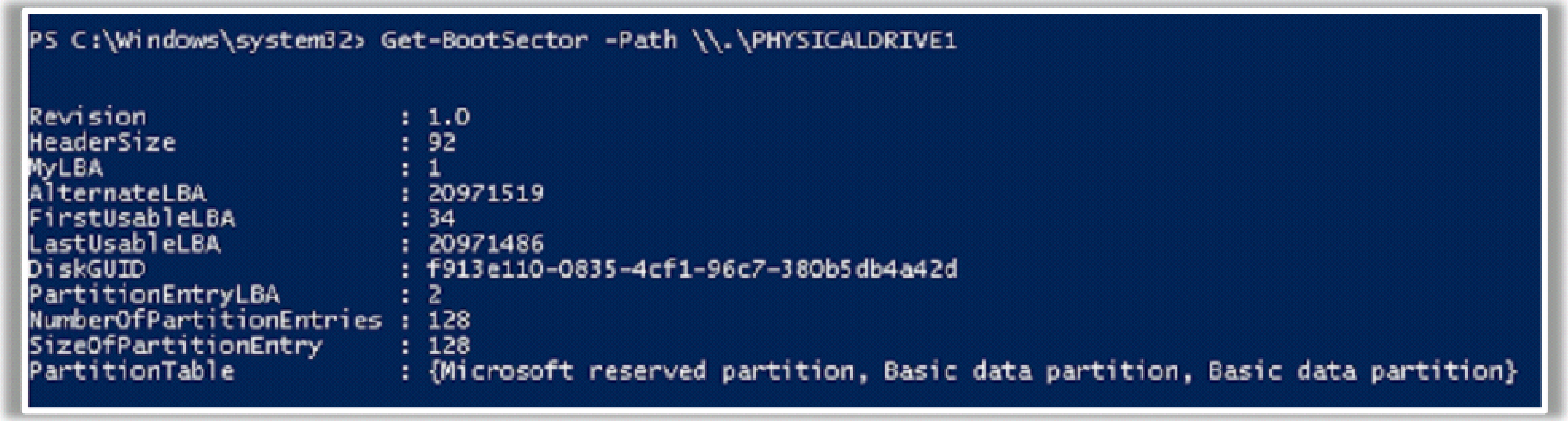
Volume	Capacity
(C:)	99900 MB
(D:)	200000 MB

Identifying GUID Partition Table (GPT) (Cont'd)

Get-BootSector

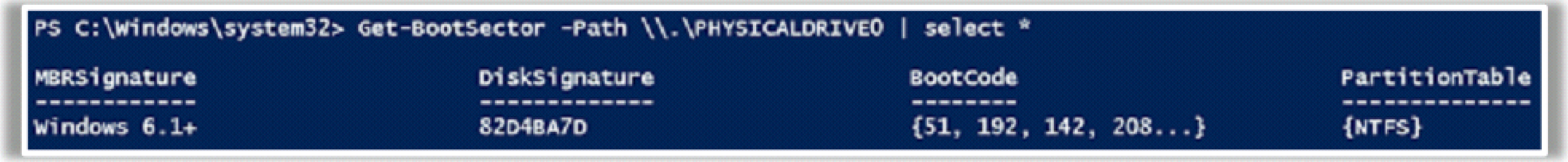
- It **reviews the hard drive's first sector** and determines if the disk is formatted using the MBR or GPT partitioning scheme. Once done, it acts just as Get-MBR or Get-GPT would, respectively.

Get-BootSector run against a disk formatted using the GPT partitioning scheme:



```
PS C:\Windows\system32> Get-BootSector -Path \\.\PHYSICALDRIVE1


Revision                 : 1.0
HeaderSize               : 92
MyLBA                    : 1
AlternateLBA             : 20971519
FirstUsableLBA           : 34
LastUsableLBA            : 20971486
DiskGUID                 : f913e110-0835-4cf1-96c7-380b5db4a42d
PartitionEntryLBA        : 2
NumberOfPartitionEntries : 128
SizeOfPartitionEntry     : 128
PartitionTable           : {Microsoft reserved partition, Basic data partition, Basic data partition}
```

Get-BootSector run against a disk formatted using the MBR partitioning scheme:



```
PS C:\Windows\system32> Get-BootSector -Path \\.\PHYSICALDRIVE0 | select *

MBRSignature              DiskSignature            BootCode                 PartitionTable
------------              -------------            --------                 --------------
Windows 6.1+              82D4BA7D                 {51, 192, 142, 208...}   {NTFS}
```

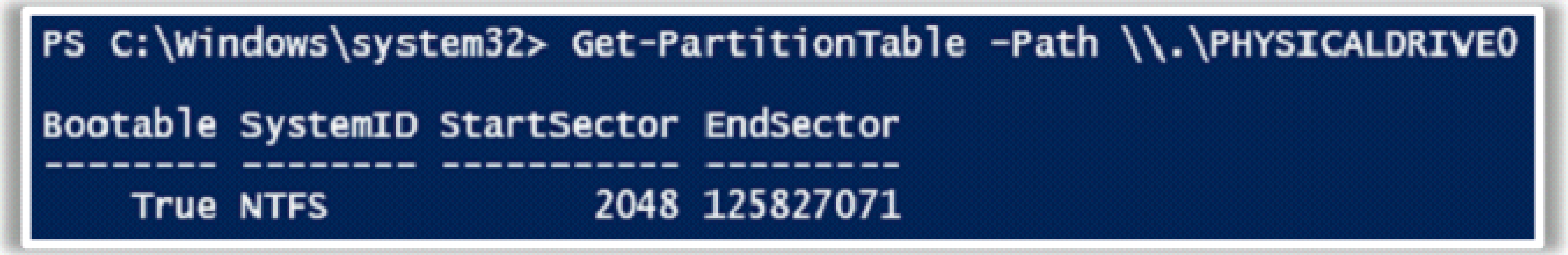
http://www.invoke-ir.com

Identifying GUID Partition Table (GPT) (Cont'd)

Get-PartitionTable

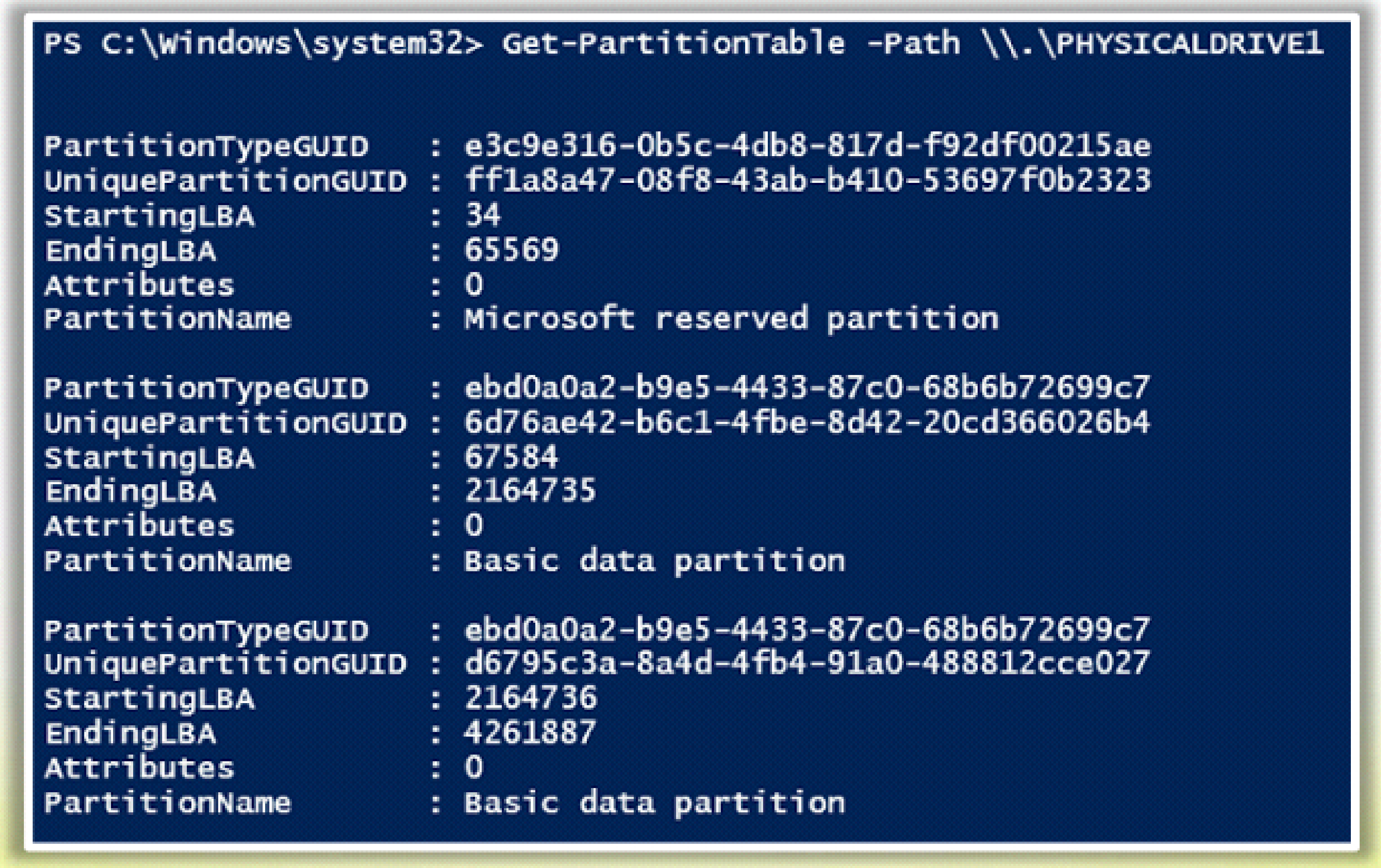
- It **determines the type of boot sector** (MBR or GPT) and returns the correct partition object (PartitionEntry or GuidPartitionTableEntry)

Get-PartitionTable run against an MBR formatted disk, returning an PartitionEntry object:



```
PS C:\Windows\system32> Get-PartitionTable -Path \\.\PHYSICALDRIVE0

Bootable SystemID StartSector EndSector
-------- -------- ----------- ---------
    True NTFS            2048 125827071
```

Get-PartitionTable run against a GPT formatted disk, returning an array of GuidPartitionTableEntry Objects:



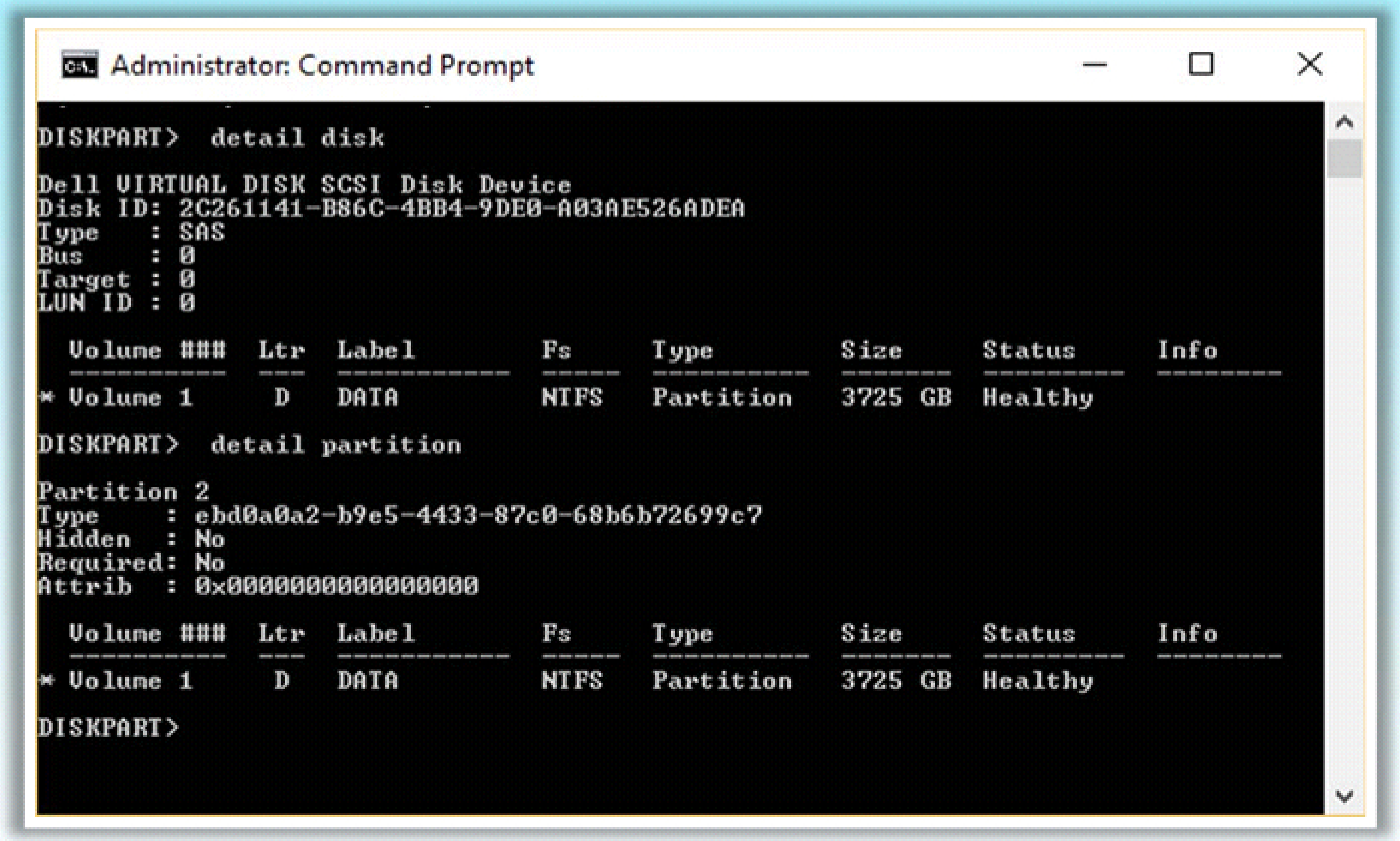
```
PS C:\Windows\system32> Get-PartitionTable -Path \\.\PHYSICALDRIVE1


PartitionTypeGUID   : e3c9e316-0b5c-4db8-817d-f92df00215ae
UniquePartitionGUID : ff1a8a47-08f8-43ab-b410-53697f0b2323
StartingLBA         : 34
EndingLBA           : 65569
Attributes          : 0
PartitionName       : Microsoft reserved partition

PartitionTypeGUID   : ebd0a0a2-b9e5-4433-87c0-68b6b72699c7
UniquePartitionGUID : 6d76ae42-b6c1-4fbe-8d42-20cd366026b4
StartingLBA         : 67584
EndingLBA           : 2164735
Attributes          : 0
PartitionName       : Basic data partition

PartitionTypeGUID   : ebd0a0a2-b9e5-4433-87c0-68b6b72699c7
UniquePartitionGUID : d6795c3a-8a4d-4fb4-91a0-488812cce027
StartingLBA         : 2164736
EndingLBA           : 4261887
Attributes          : 0
PartitionName       : Basic data partition
```

http://www.invoke-ir.com

Analyzing the GPT Header and Entries

- Most of the operating systems that support GPT disk access come up with a basic partitioning tool, which **displays details about GPTs**

  E.g.: DiskPart tool (Windows), OS X Disk utility (Mac), GNU parted tool (Linux)



```
DISKPART> detail disk

Dell VIRTUAL DISK SCSI Disk Device
Disk ID: 2C261141-B86C-4BB4-9DE0-A03AE526ADEA
Type   : SAS
Bus    : 0
Target : 0
LUN ID : 0

  Volume ###  Ltr  Label        Fs     Type        Size     Status     Info
  ----------  ---  -----------  -----  ----------  -------  ---------  --------
* Volume 1     D   DATA         NTFS   Partition   3725 GB  Healthy

DISKPART> detail partition

Partition 2
Type    : ebd0a0a2-b9e5-4433-87c0-68b6b72699c7
Hidden  : No
Required: No
Attrib  : 0x0000000000000000

  Volume ###  Ltr  Label        Fs     Type        Size     Status     Info
  ----------  ---  -----------  -----  ----------  -------  ---------  --------
* Volume 1     D   DATA         NTFS   Partition   3725 GB  Healthy

DISKPART>
```

- Sleuthkit (mmls command) can be used to view detailed partition layout for GPT disk
- Alternatively, details about GPT header and partition entries can be obtained via manual analysis using a hex editor

GPT Artifacts

Deleted and Overwritten GUID Partitions

Case 1:

- If the MBR disk is repartitioned or converted to GPT, then the sector zero will be generally overwritten with a protective MBR
- To recover data from the previous MBR partitioned volumes, investigators can use standard forensic methods used to perform an extensive search for file systems

Case 2:

- If the GPT disk is repartitioned or converted to MBR, then the GPT header and tables may remain intact based on the tool used
- Implementation of general partition deletion tools on a GPT disk might only delete the protective MBR, which can be recreated by simply reconstructing the disk

As per UEFI specification, if all the fields in a partition entry are zeroed, it implies that the entry is not in use. In this case, data recovery about deleted GUID partition entries is not possible

GUID Identifiers

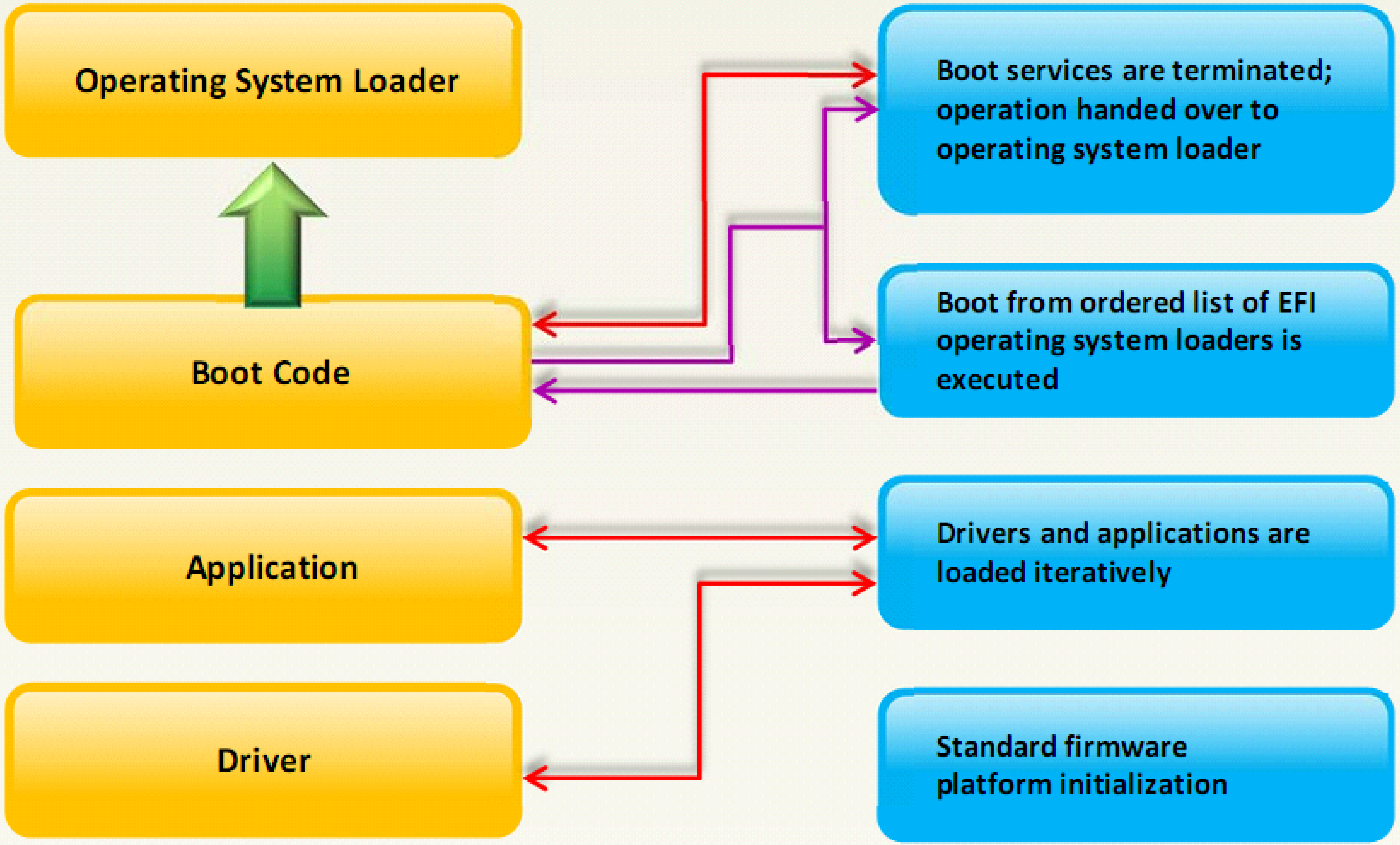
- The GPT scheme provides GUIDs which are of investigative value as they are unique and hold potential information within them
- GUIDs possess unique identifying information for both disks and individual partitions
- Investigators can use tools such as uuid to decode various versions of GUID/UUID

Hidden Information on GPT Disks

- Intruders may hide data on GPT disks as they do it on traditional MBR disks
- Data hiding places on GPT disks may be inter-partition gaps, unpartitioned space towards the end of the disk, GPT header, and reserved areas

Current forensic methods and tools to perform GPT analysis are not satisfactory

Macintosh Boot Process



Operating System Loader

Boot Code

Application

Driver

Boot services are terminated; operation handed over to operating system loader

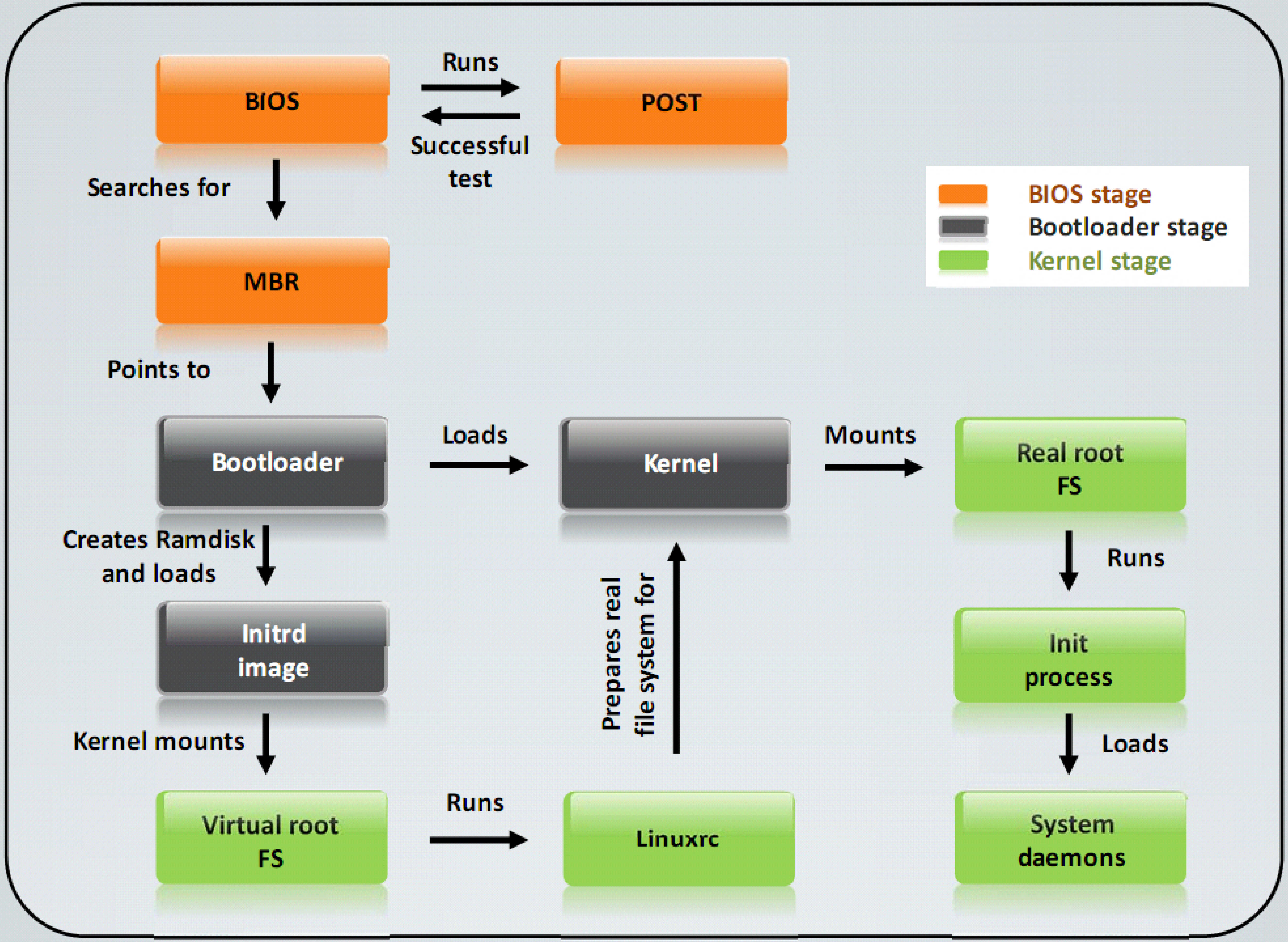
Boot from ordered list of EFI operating system loaders is executed

Drivers and applications are loaded iteratively

Standard firmware platform initialization

Legend	
🟨	EFI Binaries
🟦	Boot Manager
➡	Value Add Implementation
→	API – Specified
→	Upon Encountering an Error

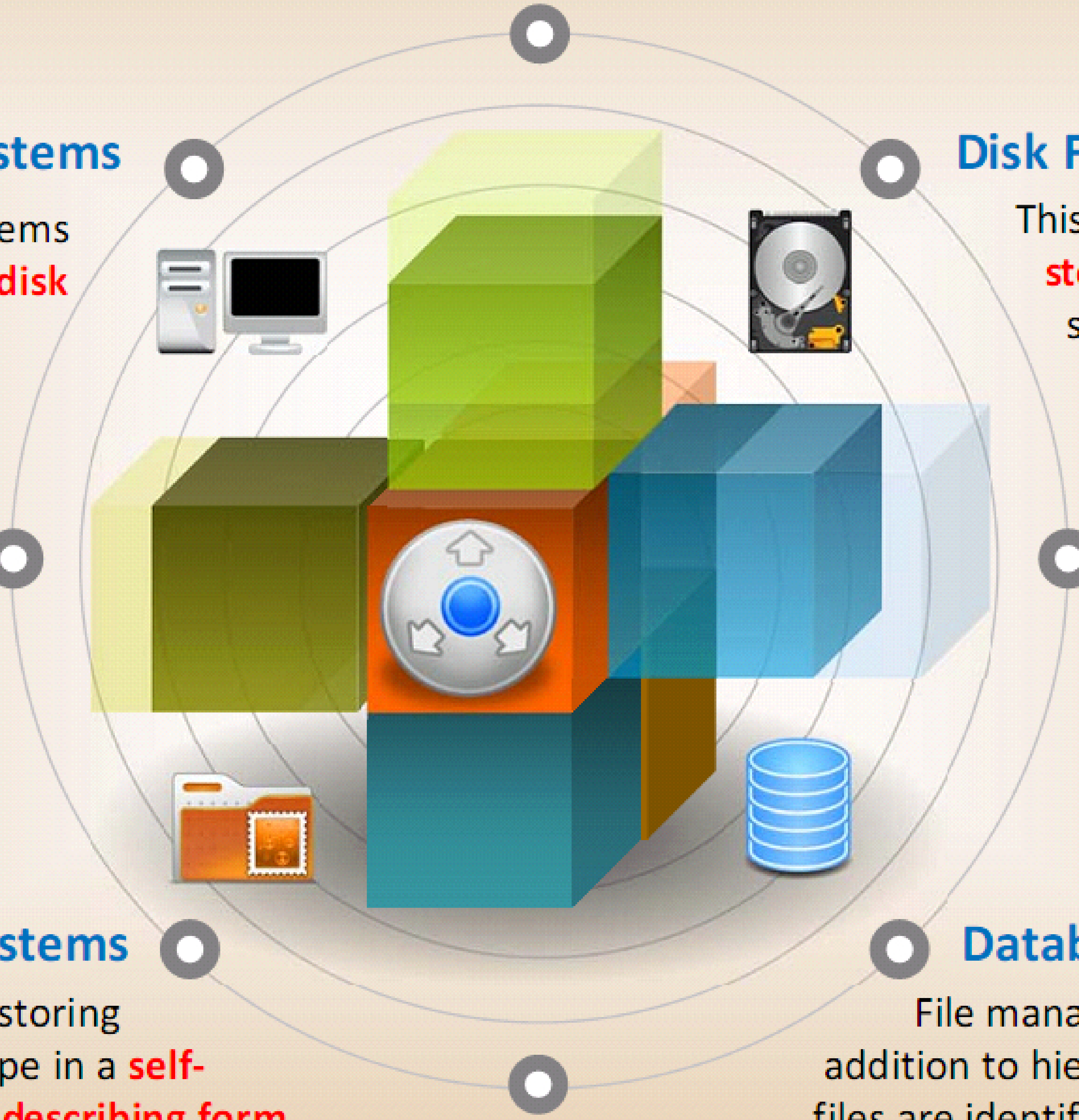
Linux Boot Process



Understanding File Systems

- The file system is a **set of data types**, which is employed for storage, hierarchical categorization, management, navigation, access, and recovering the data
- It provides a mechanism for users to store data logically in a **hierarchy of files and directories**
- It also includes a **format** for specifying the path to a file through the structure of directories
- They are organized in the form of **tree-structured directories**, and directories require access authorization
- Major file systems include FAT, NTFS, HFS, HFS+, Ext2, Ext3, Ext4, etc.

Types of File Systems



Shared Disk File Systems

In this file system, a number of systems (servers) can access same **external disk subsystem**

Disk File Systems

This file system is designed for **storing and recovering** the file on the storage devices, usually a hard disk

Special Purpose File Systems

In this file system ,files are arranged dynamically by software, intended for such purposes as communication between **computer processes or temporary file space**

Network File Systems

This file system is created to access the files on other computers that are **connected by a network**

Tape File Systems

This file system is designed for storing and recovering the file on the tape in a **self-describing form**

Database File Systems

File management, where, instead of or in addition to hierarchically structured management, files are identified by their **characteristics**, such as the type of file, topic, author, or similar metadata

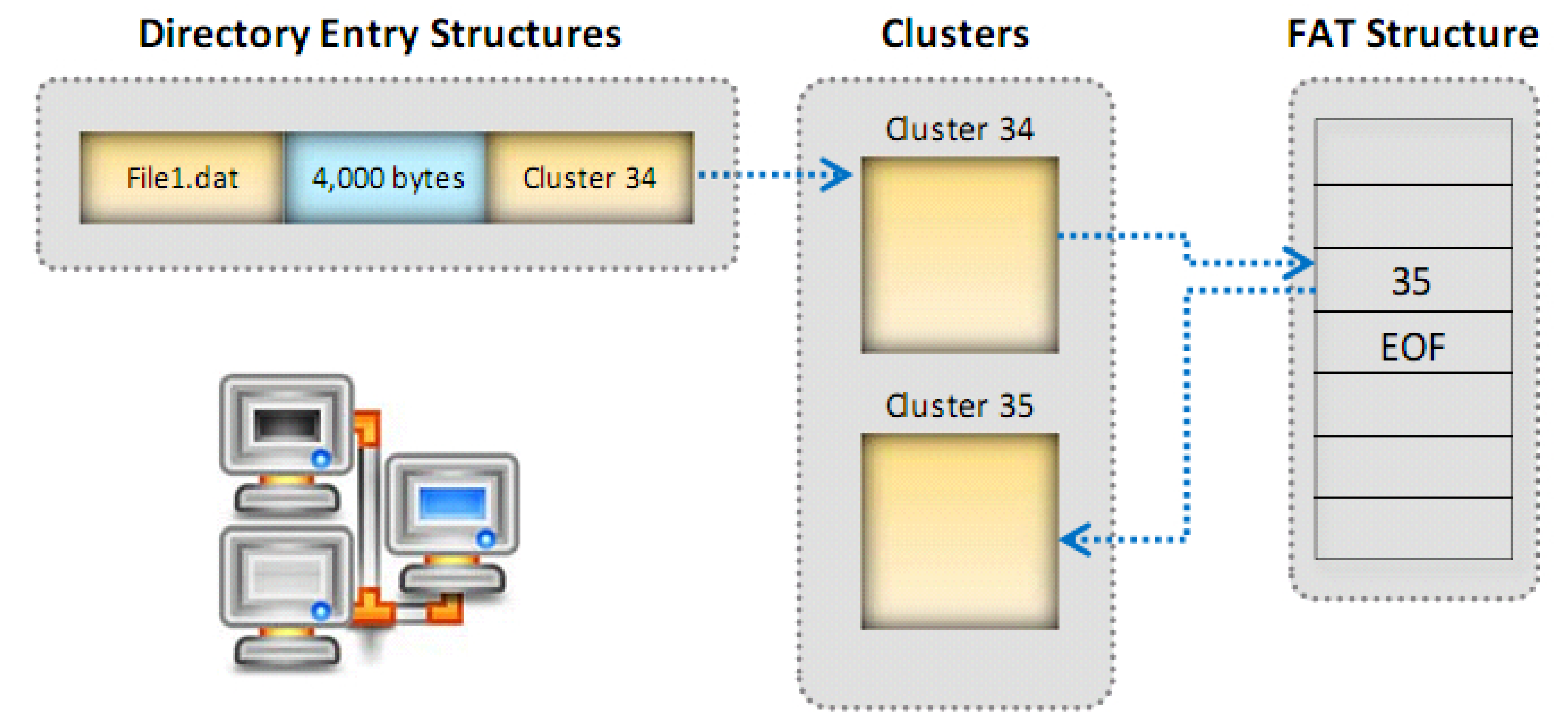
Flash File Systems

This file system is designed for storing and recovering the file on the **flash memory devices**

Windows File Systems

File Allocation Table (FAT)

- The FAT file system is used with DOS, and it was the first file system used with the Windows OS
- It is named for its method of organization, the file allocation table, which resides at the **beginning of the volume**
- FAT contains three different versions (FAT12, FAT16, and FAT32) and differs due to the **size of the entries in the FAT structure**



Relationship between the directory entry structures, clusters, and FAT structure

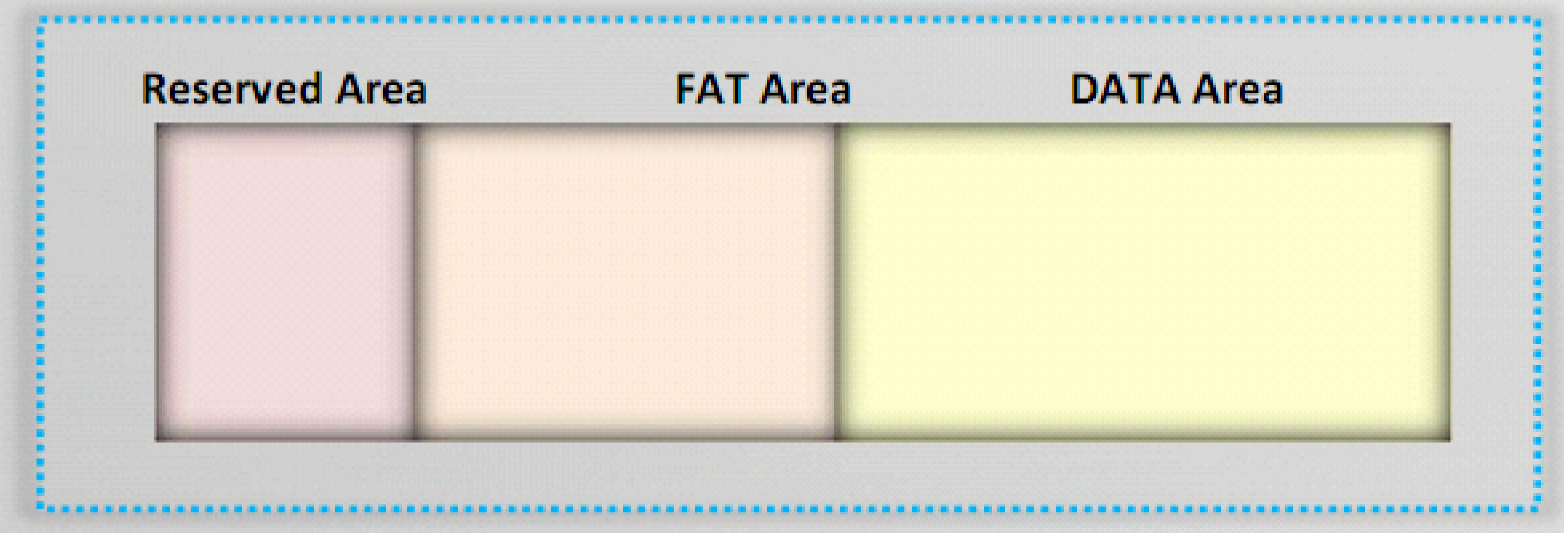
System	Bytes Per Cluster within File Allocation Table	Cluster Limit
FAT12	1.5	Fewer than 4087 clusters
FAT16	2	Between 4,087 and 65,526 clusters, inclusive
FAT32	4	Between 65,526 and 268,435,456 clusters, inclusive

FAT File System Layout

Reserved Area → It is 1 sector in size and includes **data** in the file system category

FAT Area → It contains the **FAT structures** and its size is calculated based on their number and size

Data Area → It contains the **clusters allocated** to store the content of the file and directory



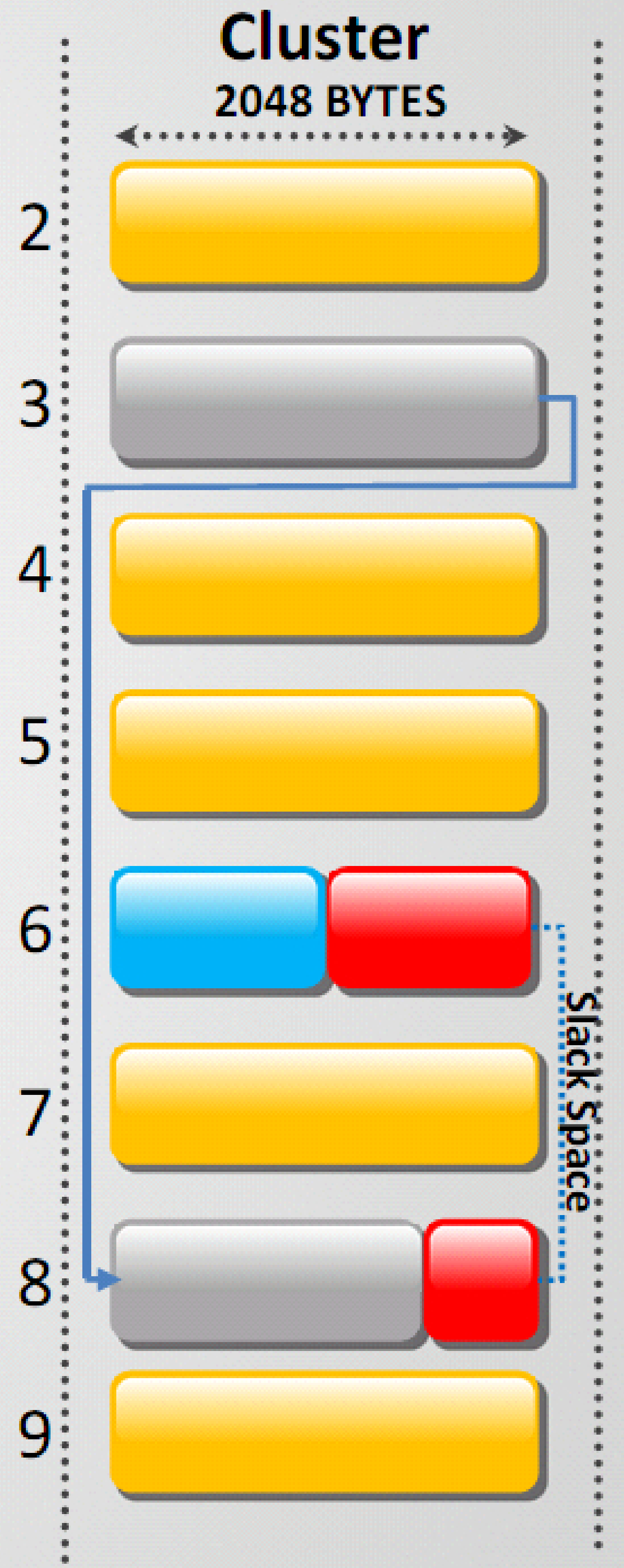
Root Directory

File Name	Size	Cluster
hello.jpg	3973	3
gary.txt	1034	6

File Allocation Table

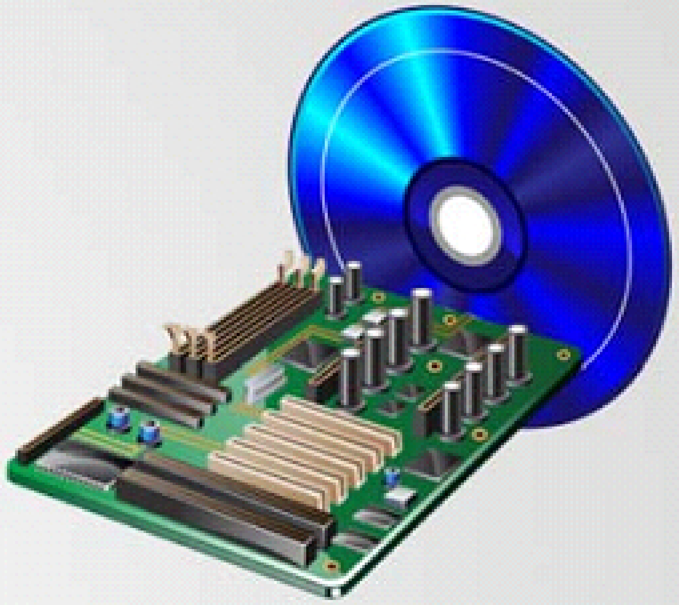
Cluster	Next
2	0x0000
3	8
4	0XFFF7 (Bad Cluster)
5	0x0000
6	0xFFFF
7	0x0000
8	0xFFFF
9	0x0000

Cluster = 2048 = 4 Sectors

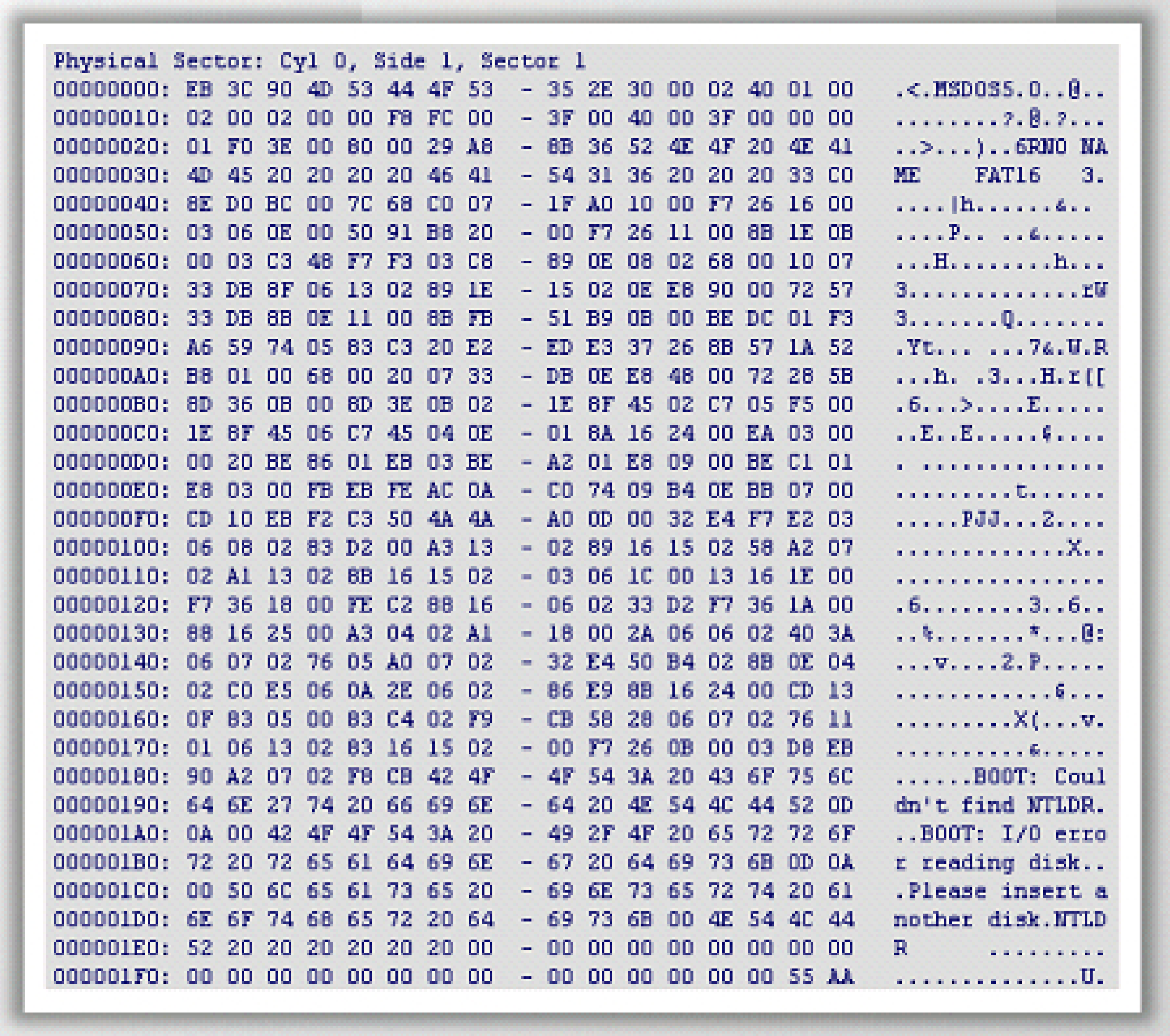


FAT Partition Boot Sector

- Boot Sector is the first sector (512 bytes) of a FAT file system
- FAT partition boot sector holds data that the file system uses to access the partition or volume
- MBR of x86-based computer systems uses this boot sector on the system partition to load the system kernel files



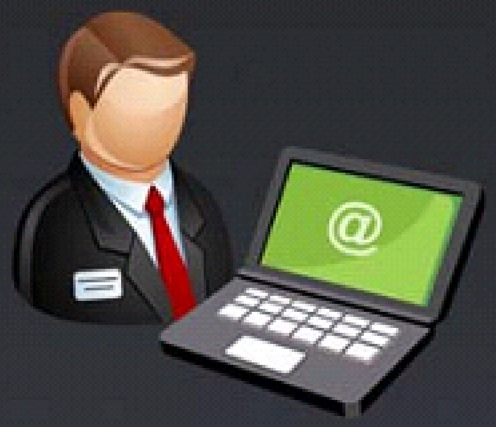
Byte Offset (in Hex)	Field Length	Sample Value	Meaning
00	3 bytes	EB 3C 90	Jump instruction
03	8 bytes	MSDOS5.0	OEM name in text
0B	25 bytes		BIOS Parameter Block (BPB)
24	26 bytes		Extended BIOS parameter block
3E	448 bytes		Bootstrap code
1FE	2 bytes	0x55AA	End of the sector marker

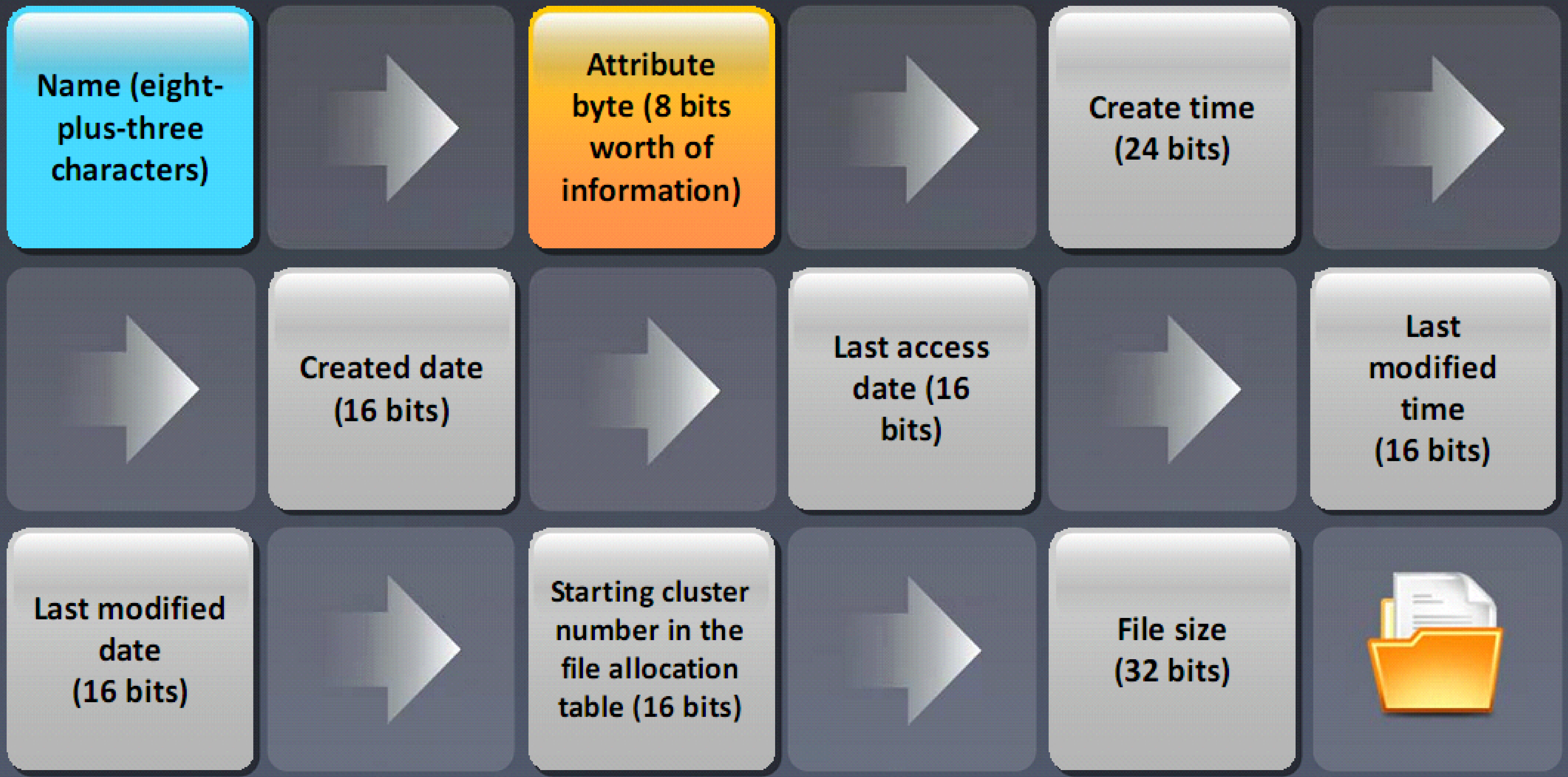


```
Physical Sector: Cyl 0, Side 1, Sector 1
00000000: EB 3C 90 4D 53 44 4F 53  - 35 2E 30 00 02 40 01 00   .<.MSDOS5.0..@..
00000010: 02 00 02 00 00 F8 FC 00  - 3F 00 40 00 3F 00 00 00   ........?.@.?...
00000020: 01 F0 3E 00 80 00 29 A8  - 8B 36 52 4E 4F 20 4E 41   ..>...)..6RNO NA
00000030: 4D 45 20 20 20 20 46 41  - 54 31 36 20 20 20 33 C0   ME    FAT16   3.
00000040: 8E D0 BC 00 7C 68 C0 07  - 1F A0 10 00 F7 26 16 00   ....|h......&..
00000050: 03 06 0E 00 50 91 B8 20  - 00 F7 26 11 00 8B 1E 0B   ....P.. ..&.....
00000060: 00 03 C3 48 F7 F3 03 C8  - 89 0E 08 02 68 00 10 07   ...H........h...
00000070: 33 DB 8F 06 13 02 89 1E  - 15 02 0E E8 90 00 72 57   3.............rW
00000080: 33 DB 8B 0E 11 00 8B FB  - 51 B9 0B 00 BE DC 01 F3   3.......Q.......
00000090: A6 59 74 05 83 C3 20 E2  - ED E3 37 26 8B 57 1A 52   .Yt... ...7&.W.R
000000A0: B8 01 00 68 00 20 07 33  - DB 0E E8 48 00 72 28 5B   ...h. .3...H.r([
000000B0: 8D 36 0B 00 8D 3E 0B 02  - 1E 8F 45 02 C7 05 F5 00   .6...>....E.....
000000C0: 1E 8F 45 06 C7 45 04 0E  - 01 8A 16 24 00 EA 03 00   ..E..E.....$....
000000D0: 00 20 BE 86 01 EB 03 BE  - A2 01 E8 09 00 BE C1 01   . ..............
000000E0: E8 03 00 FB EB FE AC 0A  - C0 74 09 B4 0E BB 07 00   .........t......
000000F0: CD 10 EB F2 C3 50 4A 4A  - A0 0D 00 32 E4 F7 E2 03   .....PJJ...2....
00000100: 06 08 02 83 D2 00 A3 13  - 02 89 16 15 02 58 A2 07   .............X..
00000110: 02 A1 13 02 8B 16 15 02  - 03 06 1C 00 13 16 1E 00   ................
00000120: F7 36 18 00 FE C2 88 16  - 06 02 33 D2 F7 36 1A 00   .6........3..6..
00000130: 88 16 25 00 A3 04 02 A1  - 18 00 2A 06 06 02 40 3A   ..%.......*...@:
00000140: 06 07 02 76 05 A0 07 02  - 32 E4 50 B4 02 8B 0E 04   ...v....2.P.....
00000150: 02 C0 E5 06 0A 2E 06 02  - 86 E9 8B 16 24 00 CD 13   ............$...
00000160: 0F 83 05 00 83 C4 02 F9  - CB 58 28 06 07 02 76 11   .........X(...v.
00000170: 01 06 13 02 83 16 15 02  - 00 F7 26 0B 00 03 D8 EB   ..........&.....
00000180: 90 A2 07 02 F8 CB 42 4F  - 4F 54 3A 20 43 6F 75 6C   ......BOOT: Coul
00000190: 64 6E 27 74 20 66 69 6E  - 64 20 4E 54 4C 44 52 0D   dn't find NTLDR.
000001A0: 0A 00 42 4F 4F 54 3A 20  - 49 2F 4F 20 65 72 72 6F   ..BOOT: I/O erro
000001B0: 72 20 72 65 61 64 69 6E  - 67 20 64 69 73 6B 0D 0A   r reading disk..
000001C0: 00 50 6C 65 61 73 65 20  - 69 6E 73 65 72 74 20 61   .Please insert a
000001D0: 6E 6F 74 68 65 72 20 64  - 69 73 6B 00 4E 54 4C 44   nother disk.NTLD
000001E0: 52 20 20 20 20 20 20 00  - 00 00 00 00 00 00 00 00   R      .........
000001F0: 00 00 00 00 00 00 00 00  - 00 00 00 00 00 00 55 AA   ..............U.
```

FAT Folder Structure

- FAT file systems have a set of **32-byte folder entries** for every file
- Folder entries in FAT system:





Name (eight-plus-three characters)

Attribute byte (8 bits worth of information)

Create time (24 bits)

Created date (16 bits)

Last access date (16 bits)

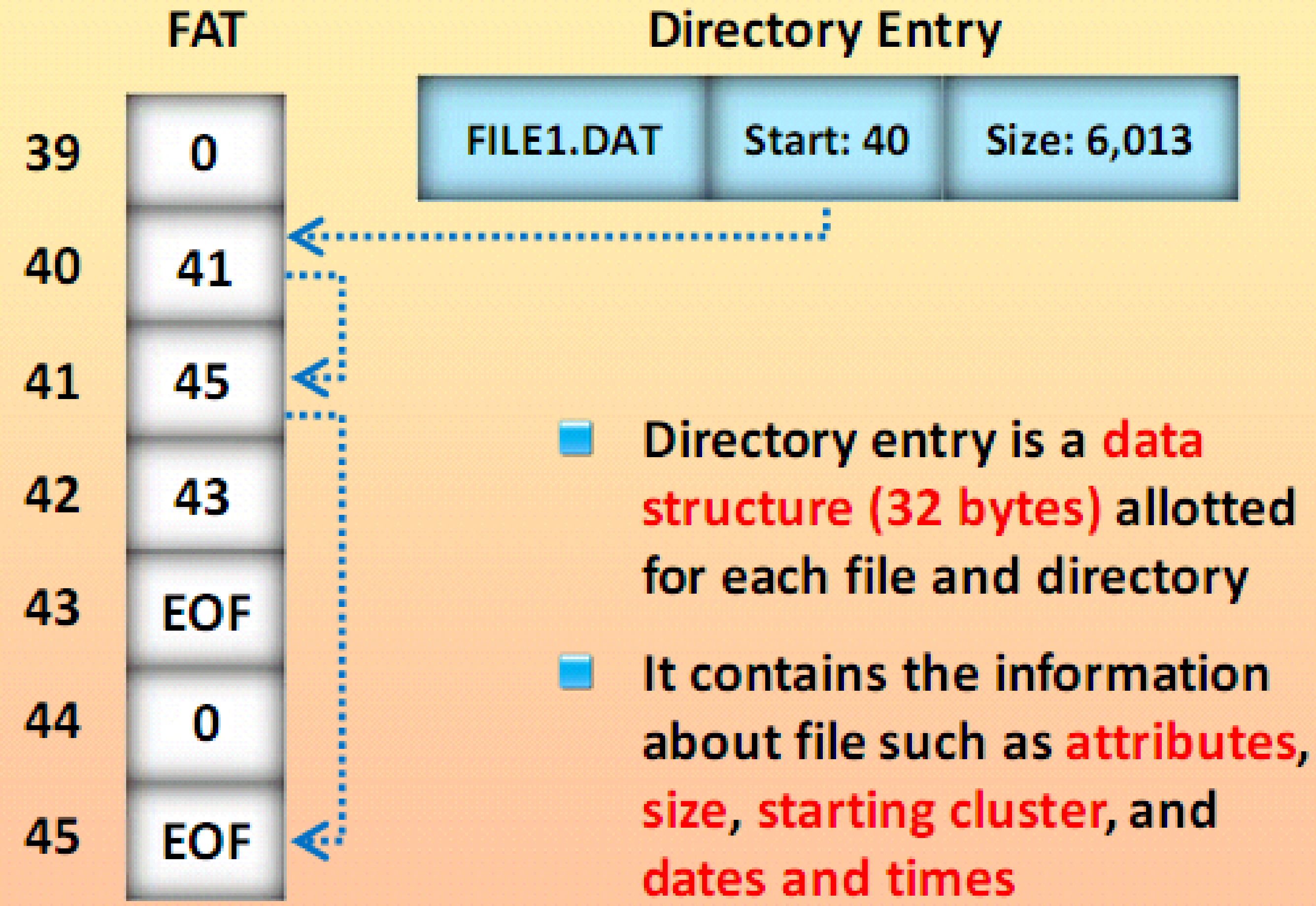
Last modified time (16 bits)

Last modified date (16 bits)

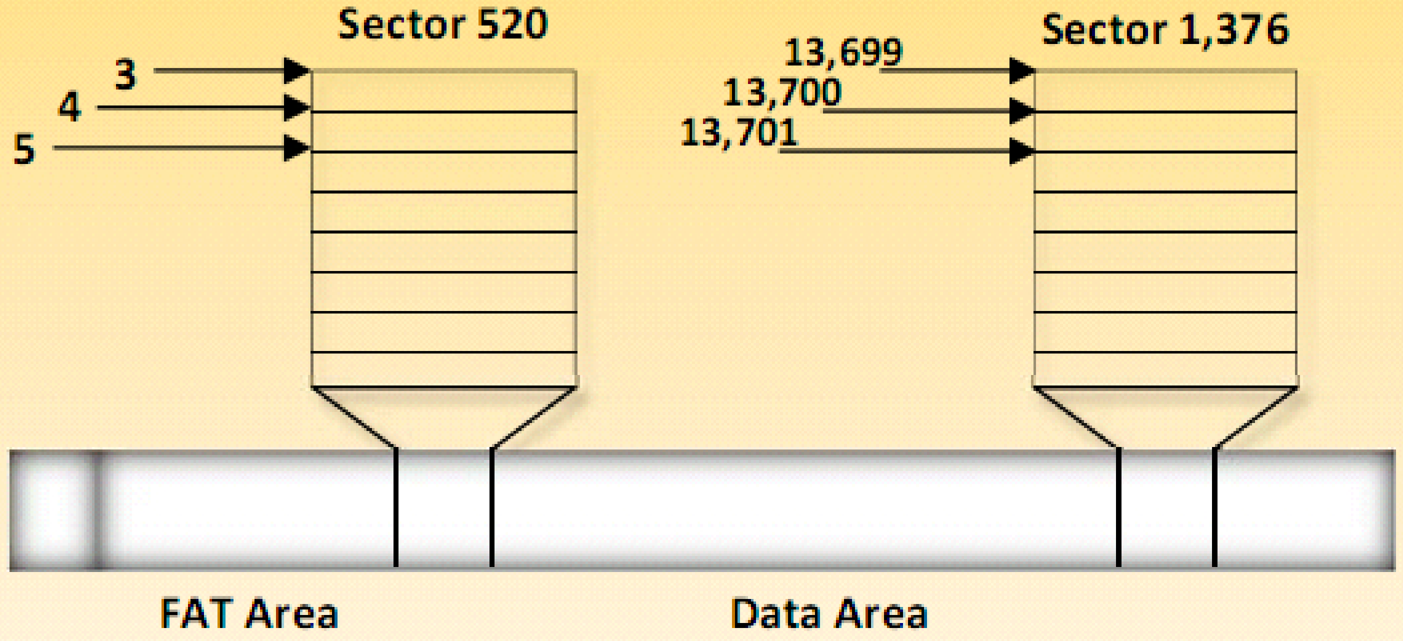
Starting cluster number in the file allocation table (16 bits)

File size (32 bits)

Directory Entries and Cluster Chains



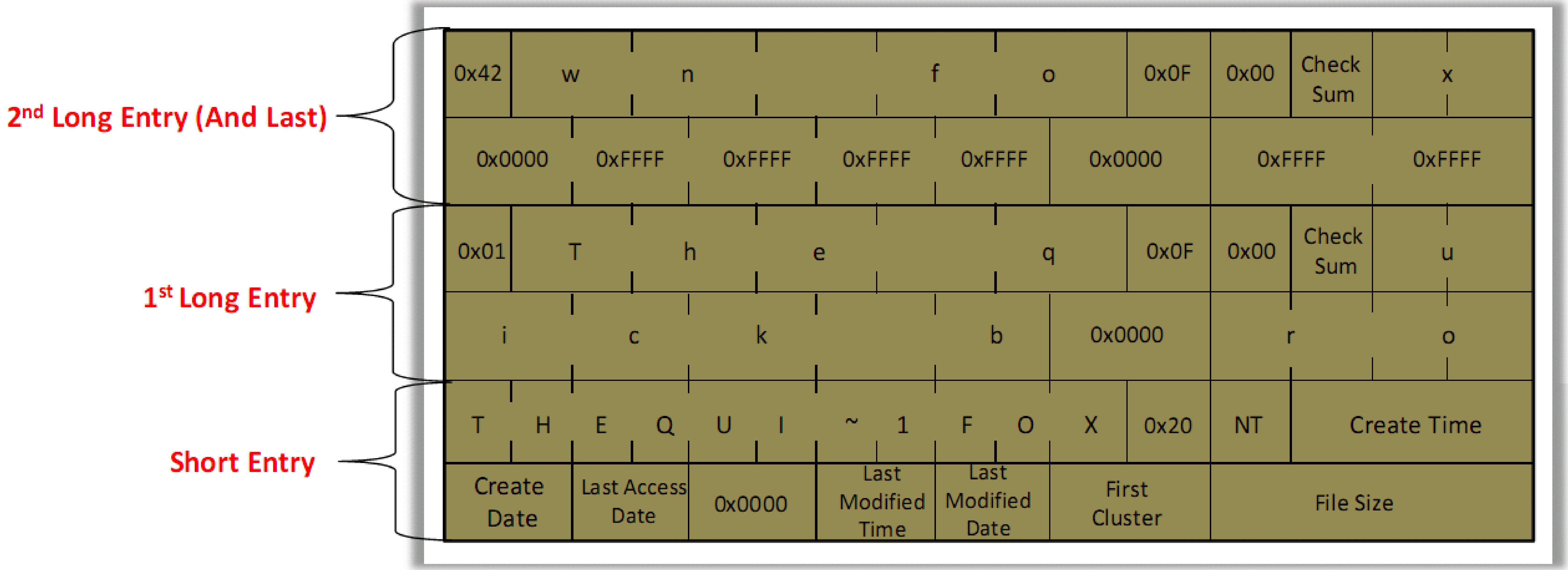
- Directory entry is a data structure (32 bytes) allotted for each file and directory
- It contains the information about file such as attributes, size, starting cluster, and dates and times



Byte Range	Description
0 – 0	First character of file name in ASCII and allocation status (0xe5 or 0x00 if unallocated)
1 – 10	Characters 2 to 11 of file name in ASCII
11 – 11	File Attributes
12 – 12	Reserved
13 – 13	Created time (tenths of second)
14 – 15	Created time (hours, minutes, seconds)
16 – 17	Created day
18 – 19	Accessed day
20 – 21	High 2 bytes of first cluster address (0 for FAT12 and FAT16)
22 – 23	Written time (hours, minutes, seconds)
24 – 25	Written day
26 – 27	Low 2 bytes of first cluster address
28 – 31	Size of file (0 for directories)

Filenames on **FAT Volumes**

- Whenever users create or rename a file on FAT volume, Windows uses **attribute bits** to support long file names and creates an eight-plus-three name for the file
- Windows also creates many **secondary folder entries** for the file
- Below diagram shows all of the folder entries for the file **Thequi~1.fox**, which has a long name of **The quick brown.fox**



2nd Long Entry (And Last)	0x42	w	n		f	o	0x0F	0x00	Check Sum	x
	0x0000	0xFFFF	0xFFFF	0xFFFF	0xFFFF	0x0000	0xFFFF	0xFFFF		
1st Long Entry	0x01	T	h	e		q	0x0F	0x00	Check Sum	u
	i	c	k		b	0x0000	r	o		
Short Entry	T H E Q U I ~ 1 F O X	0x20	NT	Create Time						
	Create Date	Last Access Date	0x0000	Last Modified Time	Last Modified Date	First Cluster	File Size			

http://technet.microsoft.com

FAT32

- FAT32 file system is derived from a **FAT file system** and supports drives up to **2 terabytes** in size
- It uses drive space efficiently and uses **small clusters**
- It takes backup of the **file allocation table** instead of the default copy

Offset	Description	Size
000h	Executable Code (Boots Computer)	446 Bytes
1BEh	1st Position Entry	16 Bytes
1CEh	2nd Position Entry	16 Bytes
1DEh	3rd Position Entry	16 Bytes
1EEh	4th Position Entry	16 Bytes
1FEh	Boot Record Signature	2 Bytes

MBR table of FAT32

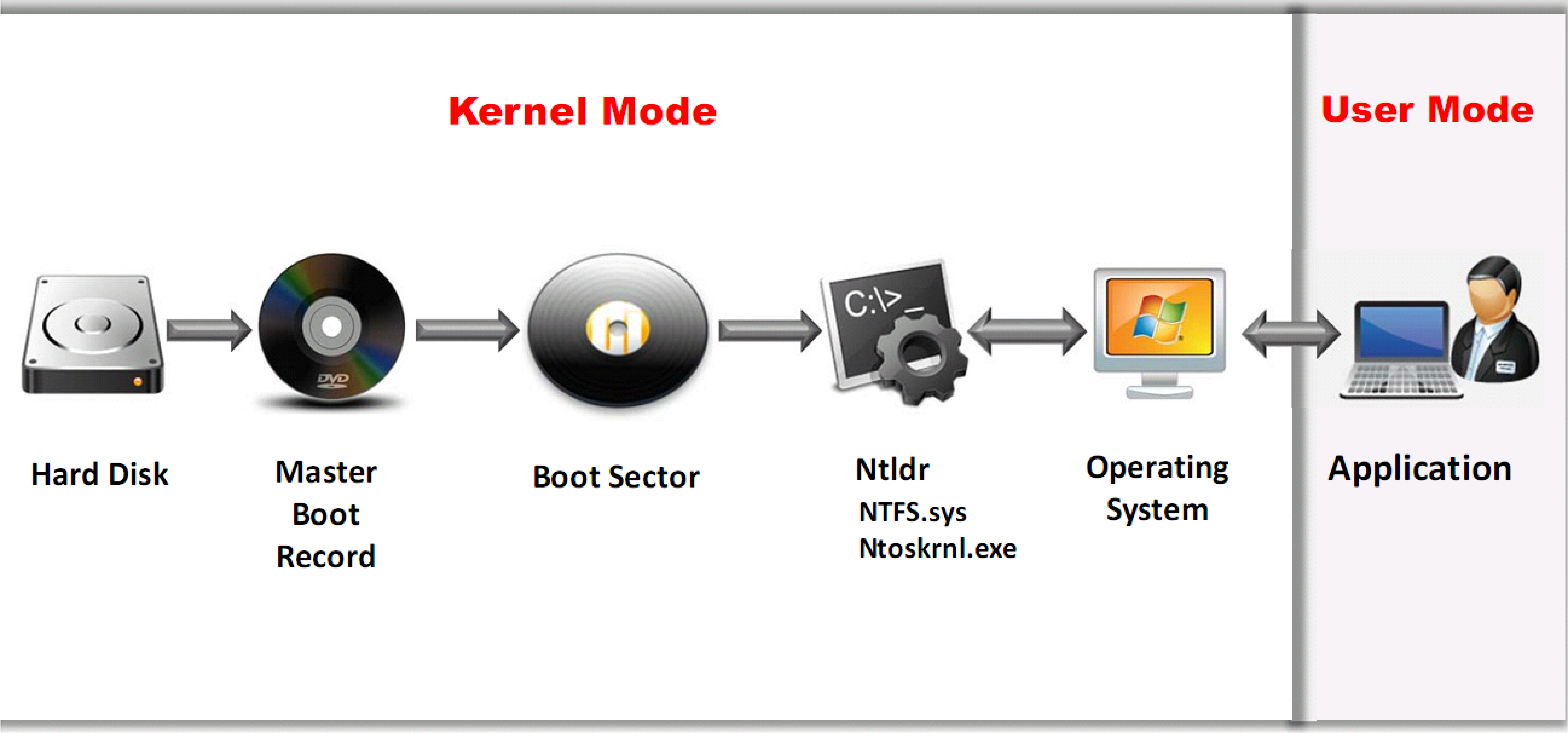
New Technology File System (NTFS)

- NTFS is the standard file system of Windows NT and its descendants Windows XP, Vista, 7, 8.1,10, server 2003, server 2008, and server 2012

- From Windows NT 3.1, it is the default file system of Windows NT family

- It has several improvements over FAT such as improved support for metadata and the use of advanced data structures to improve performance, reliability, and disk space utilization plus additional extensions such as security access control lists and file system journaling

NTFS Architecture



NTFS System Files

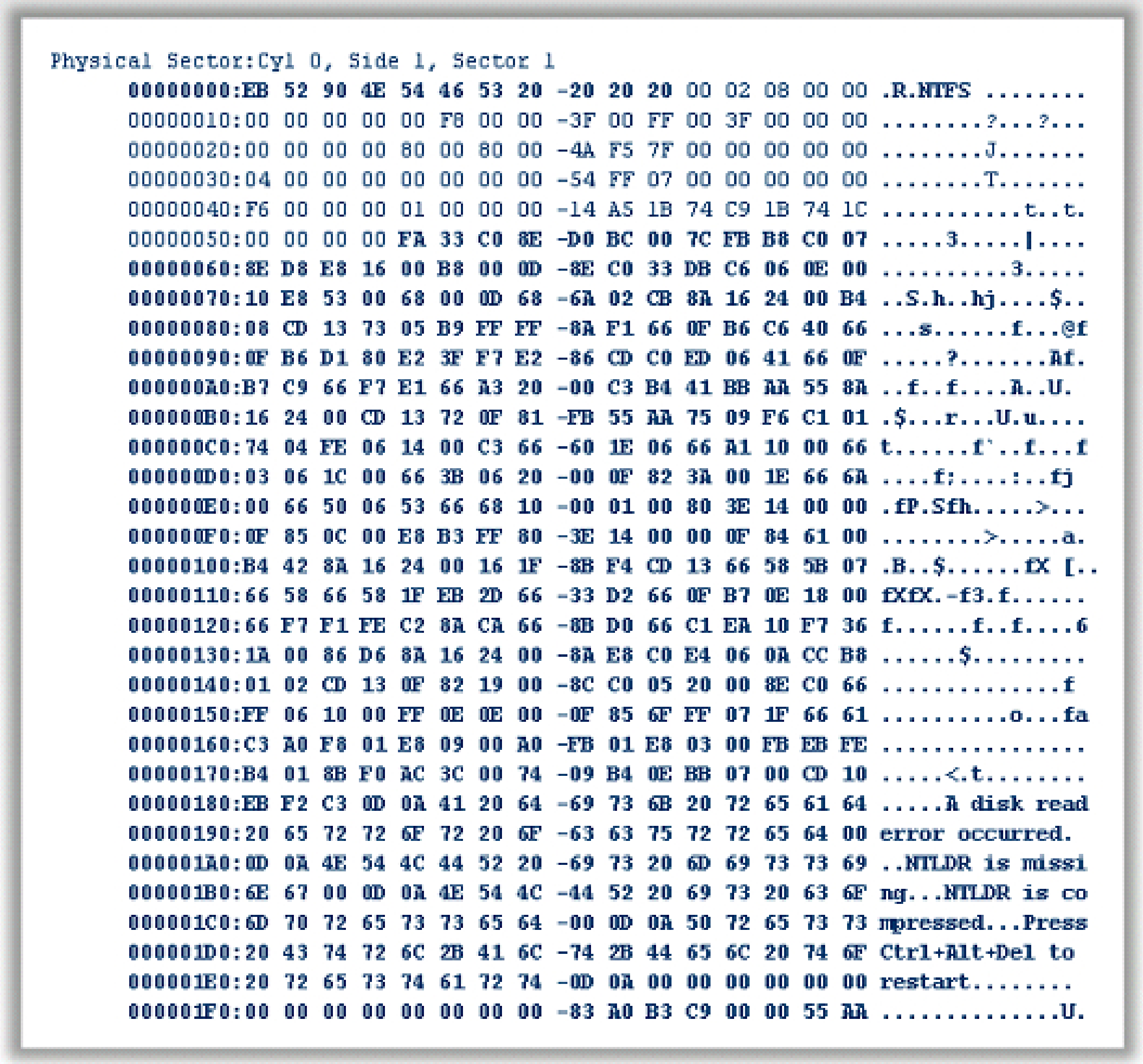
File Name	Description
$attrdef	Contains definitions of all system-and user-defined attributes of the volume
$badclus	Contains all the bad clusters
$bitmap	Contains bitmap for the entire volume
$boot	Contains the volume's bootstrap
$logfile	Used for recovery purposes
$mft	Contains a record for every file
$mftmirr	Mirror of the MFT used for recovering files
$quota	Indicates disk quota for each user
$upcase	Converts characters into uppercase Unicode
$volume	Contains volume name and version number

NTFS Partition Boot Sector

- When a volume is formatted as an NTFS volume, the format program allocates the first 16 sectors for the boot sector and the bootstrap code
- Partition identifier: 0x07 (MBR) EBD0A0A2-B9E5-4433-87C0-68B6B72699C7 (GPT)

Boot Sector of an NTFS Volume:

Byte Offset	Field Length	Field Name
0x00	3 bytes	Jump Instruction
0x03	LONGLONG	OEM ID
0x0B	25 bytes	BIOS Parameter Block (BPB)
0x24	48 bytes	Extended BPB
0x54	426 bytes	Bootstrap Code
0x01FE	WORD	End of Sector Marker



```
Physical Sector:Cyl 0, Side 1, Sector 1
        00000000:EB 52 90 4E 54 46 53 20 -20 20 20 00 02 08 00 00 .R.NTFS ........
        00000010:00 00 00 00 00 F8 00 00 -3F 00 FF 00 3F 00 00 00 ........?...?...
        00000020:00 00 00 00 80 00 80 00 -4A F5 7F 00 00 00 00 00 ........J.......
        00000030:04 00 00 00 00 00 00 00 -54 FF 07 00 00 00 00 00 ........T.......
        00000040:F6 00 00 00 01 00 00 00 -14 A5 1B 74 C9 1B 74 1C ...........t..t.
        00000050:00 00 00 00 FA 33 C0 8E -D0 BC 00 7C FB B8 C0 07 .....3.....|....
        00000060:8E D8 E8 16 00 B8 00 0D -8E C0 33 DB C6 06 0E 00 ..........3.....
        00000070:10 E8 53 00 68 00 0D 68 -6A 02 CB 8A 16 24 00 B4 ..S.h..hj....$..
        00000080:08 CD 13 73 05 B9 FF FF -8A F1 66 0F B6 C6 40 66 ...s......f...@f
        00000090:0F B6 D1 80 E2 3F F7 E2 -86 CD C0 ED 06 41 66 0F .....?.......Af.
        000000A0:B7 C9 66 F7 E1 66 A3 20 -00 C3 B4 41 BB AA 55 8A ..f..f....A..U.
        000000B0:16 24 00 CD 13 72 0F 81 -FB 55 AA 75 09 F6 C1 01 .$...r...U.u....
        000000C0:74 04 FE 06 14 00 C3 66 -60 1E 06 66 A1 10 00 66 t......f`..f...f
        000000D0:03 06 1C 00 66 3B 06 20 -00 0F 82 3A 00 1E 66 6A ....f;....:..fj
        000000E0:00 66 50 06 53 66 68 10 -00 01 00 80 3E 14 00 00 .fP.Sfh.....>...
        000000F0:0F 85 0C 00 E8 B3 FF 80 -3E 14 00 00 0F 84 61 00 .........>.....a.
        00000100:B4 42 8A 16 24 00 16 1F -8B F4 CD 13 66 58 5B 07 .B..$......fX [..
        00000110:66 58 66 58 1F EB 2D 66 -33 D2 66 0F B7 0E 18 00 fXfX.-f3.f......
        00000120:66 F7 F1 FE C2 8A CA 66 -8B D0 66 C1 EA 10 F7 36 f......f..f....6
        00000130:1A 00 86 D6 8A 16 24 00 -8A E8 C0 E4 06 0A CC B8 ......$.........
        00000140:01 02 CD 13 0F 82 19 00 -8C C0 05 20 00 8E C0 66 ..............f
        00000150:FF 06 10 00 FF 0E 0E 00 -0F 85 6F FF 07 1F 66 61 ..........o...fa
        00000160:C3 A0 F8 01 E8 09 00 A0 -FB 01 E8 03 00 FB EB FE ................
        00000170:B4 01 8B F0 AC 3C 00 74 -09 B4 0E BB 07 00 CD 10 .....<.t........
        00000180:EB F2 C3 0D 0A 41 20 64 -69 73 6B 20 72 65 61 64 .....A disk read
        00000190:20 65 72 72 6F 72 20 6F -63 63 75 72 72 65 64 00 error occurred.
        000001A0:0D 0A 4E 54 4C 44 52 20 -69 73 20 6D 69 73 73 69 ..NTLDR is missi
        000001B0:6E 67 00 0D 0A 4E 54 4C -44 52 20 69 73 20 63 6F ng...NTLDR is co
        000001C0:6D 70 72 65 73 73 65 64 -00 0D 0A 50 72 65 73 73 mpressed...Press
        000001D0:20 43 74 72 6C 2B 41 6C -74 2B 44 65 6C 20 74 6F Ctrl+Alt+Del to
        000001E0:20 72 65 73 74 61 72 74 -0D 0A 00 00 00 00 00 00 restart........
        000001F0:00 00 00 00 00 00 00 00 -83 A0 B3 C9 00 00 55 AA ..............U.
```

Cluster Sizes of NTFS Volume

- A cluster is the smallest **allocation unit on the hard disk** that is used to hold a file
- NTFS uses **clusters of different sizes** to hold the files, depending on the size of the NTFS volume
- List of the default cluster sizes for NTFS volume:

Volume Size	Sectors Per Cluster	Default Cluster Size
512 MB or less	1	512 bytes
513 MB – 1024 MB (1 GB)	2	1024 bytes (1 GB)
1024 – 2048 MB (2 GB)	4	2048 bytes (2 GB)
Greater than 2049 MB	8	4 KB

NTFS Master File Table (MFT)

Each file on an NTFS volume is represented by a record in a special file called the **master file table** (MFT)

It **reserves the first 16 records** of the table for special information

The first record of this table describes the master file table itself, followed by an **MFT mirror record**

If the first MFT record is corrupted, NTFS reads the second record to find the **MFT mirror file**, whose first record is identical to the first record of the MFT

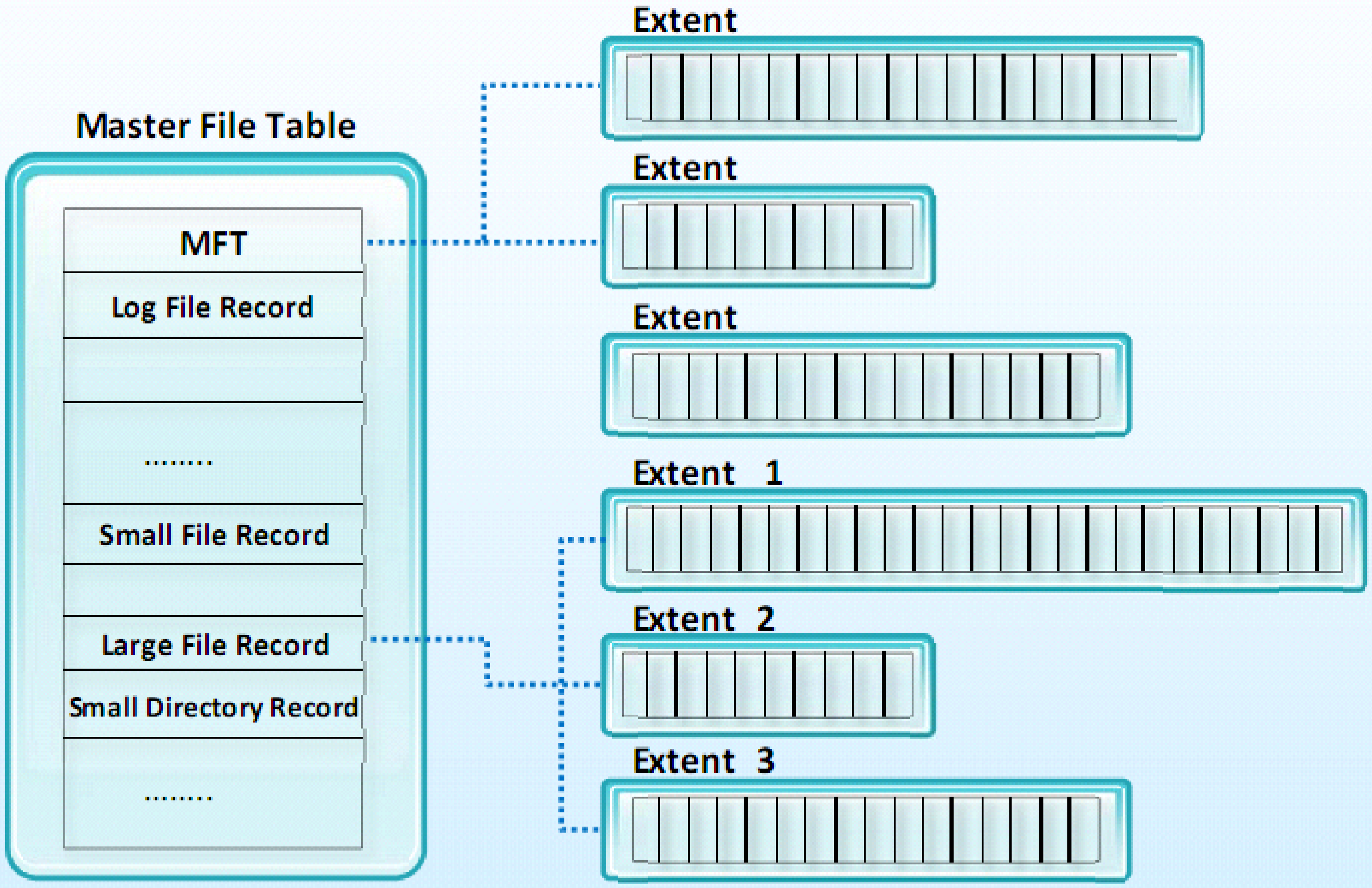
The locations of the data segments for both the MFT and MFT mirror file are recorded in the boot sector, a **duplicate** of the boot sector is located at the **logical center of the disk**

The third record of the MFT is the log file, used for file recovery. The **seventeenth and following records** of the master file table are for each file and directory (also viewed as a file by NTFS) on the volume

NTFS Master File Table (MFT) (Cont'd)

- MFT is a relational database, which consists of information related to the **files** and the **file attributes**
- The rows consist of **file records** and the columns consist of **file attributes**
- It has information of every file on the **NTFS volume** including its own information
- It has 16 records reserved for **system files**
- For small folder, MFT is represented as follows:

Standard Information	File or Directory Name	Data or Index	Unused Space



Structure of a Master File Table on an NTFS volume

Metadata Files Stored in the MFT

System File	File Name	MFT Record	Purpose of the File
Master file table	$Mft	0	It contains one base file record for each file and folder on an NTFS volume
Master file table mirror	$MftMirr	1	It guarantees access to the MFT in case of a single-sector failure
Log file	$LogFile	2	It contains information used by NTFS for faster recoverability
Volume	$Volume	3	It contains information about the volume
Attribute definitions	$AttrDef	4	It lists attribute names, numbers, and descriptions
Root file name index		5	The root folder
Cluster bitmap	$Bitmap	6	It represents the volume by showing free and unused clusters
Boot sector	$Boot	7	It includes the BPB used to mount the volume
Bad cluster file	$BadClus	8	It contains bad clusters for a volume
Security file	$Secure	9	It contains unique security descriptors for all files within a volume
Upcase table	$Upcase	10	It converts lowercase characters to matching Unicode uppercase characters
NTFS extension file	$Extend	11	It is used for various optional extensions

NTFS Attributes

- Every file has unique identities such as **name**, **security information**, and **metadata of file system** in the file

- Every attribute is identified by an **attribute type code** and **attribute name**

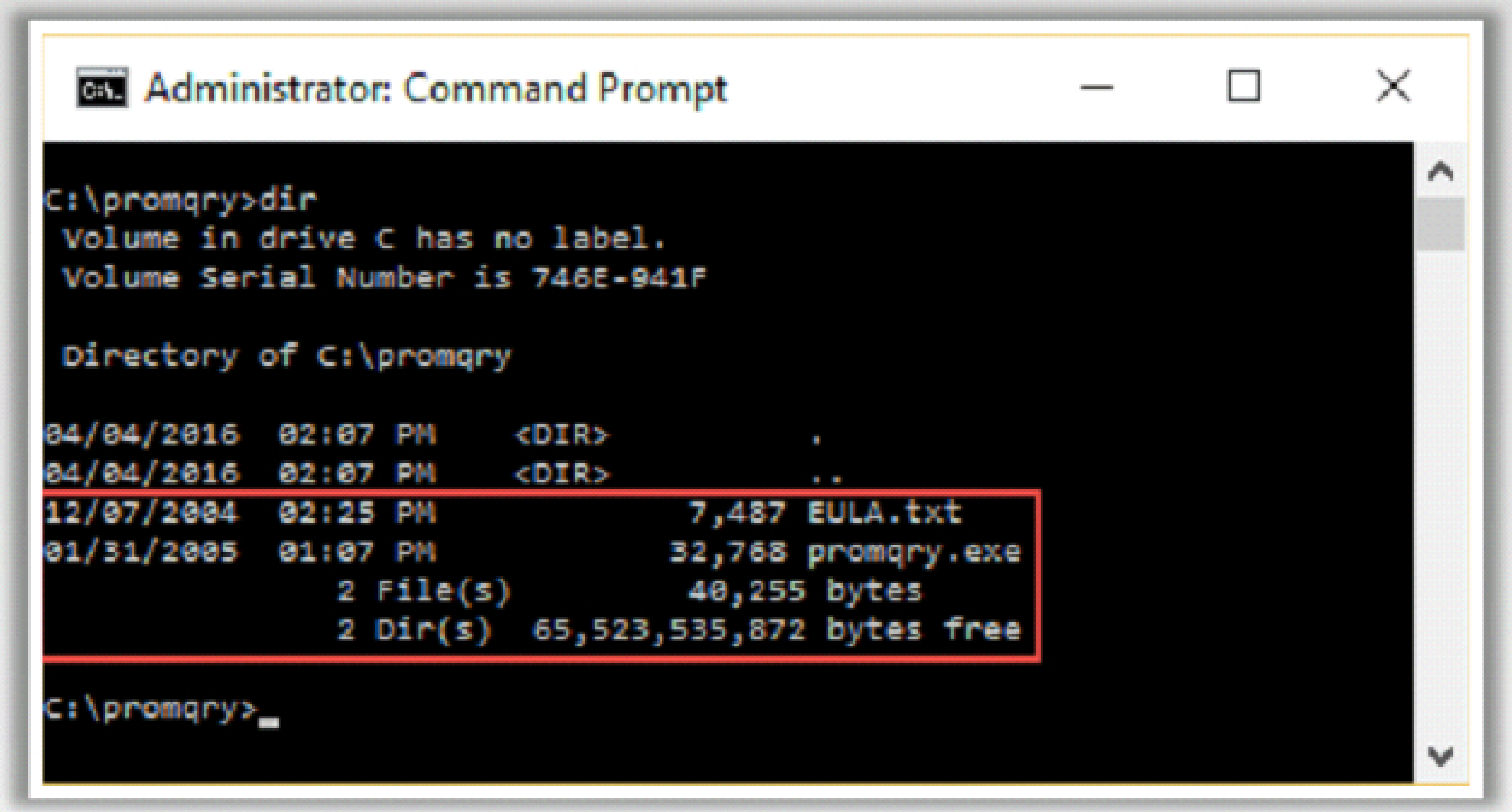
- There are two categories of attributes:
  - **Resident attributes**: These are the attributes that are contained in the MFT
  - **Non-resident attributes**: These are the attributes that are allocated with one or more clusters of disk space

Attribute Type	Purpose of the Attribute
Standard information	Lists the information regarding the time stamp data and link count information
Attribute list	List of attributes that are not in the MFT
File name	The file name is stored here and can be a long or short name
Security descriptor	Ownership and access rights to the file are listed here
Data	Stores file data
Object id	File identifier volume –unique identifier
Logged tool stream	Used by the encrypted file system service
Reparse point	Volume mount point used for installable file system filter drivers
Index root	Employed for use of folders and files
Index allocation	Employed for use of folders and files
Bitmap	Employed for use of folders and files
Volume information	Version number of the volume is listed
Volume name	The volume label is listed here

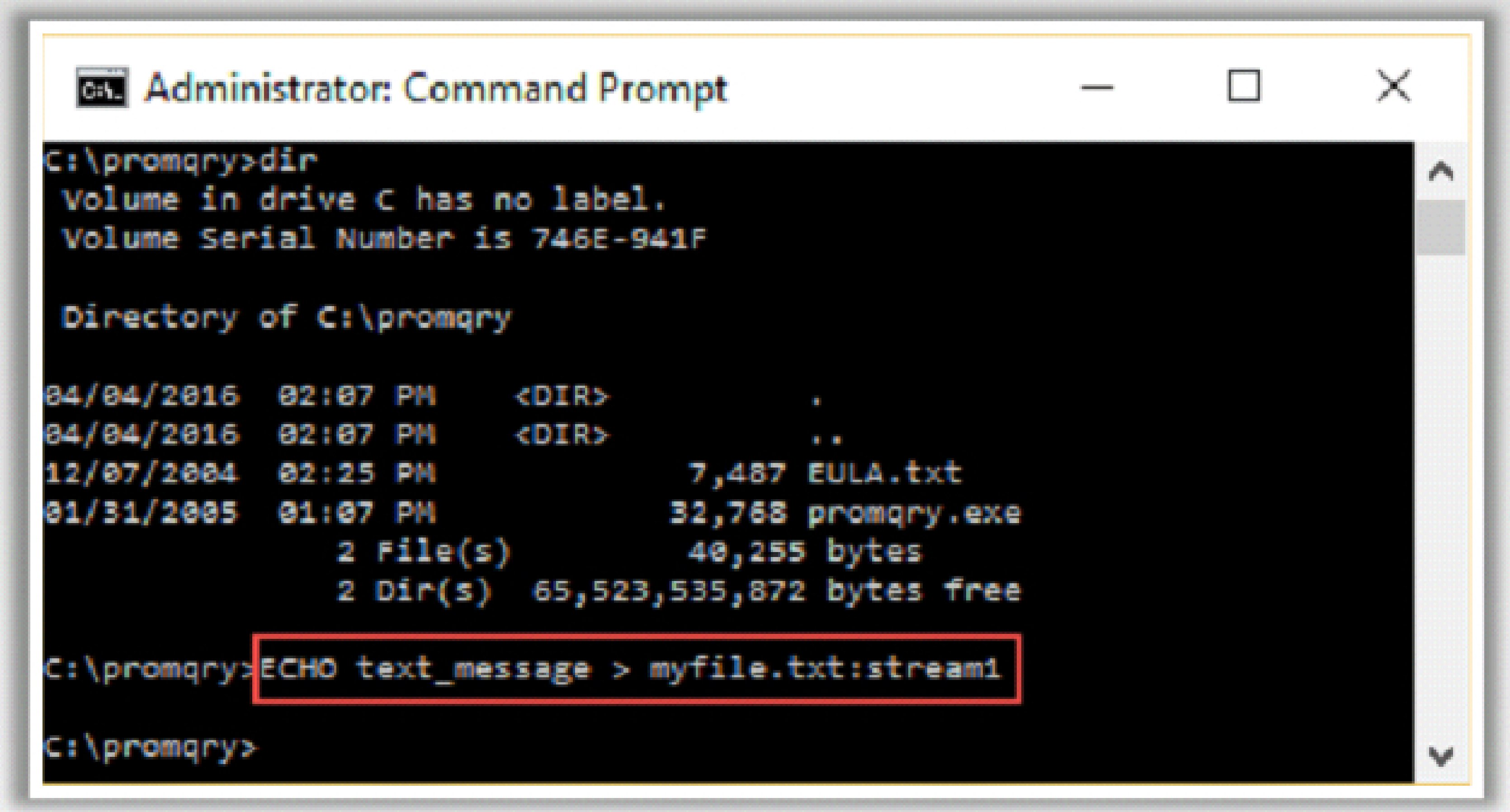
NTFS Data Stream

- NTFS data stream is a **unique set of file attributes**
- NTFS supports **multiple data streams per file**, where the stream name **identifies a new data attribute** on the file
- Command that creates a data stream in an existing file on an NTFS volume:
  `C:\>ECHO text_message > myfile.txt:stream1`
- Command to display the contents of the data stream: `C:\>MORE < myfile.txt:stream1`
- A data stream does not appear when a file is opened in a text editor. The only way to see if a data stream is attached to a file is by examining the MFT entry for the file
- When you copy an **NTFS** file to a **FAT volume**, such as a floppy disk, data streams and other attributes not supported by FAT are lost

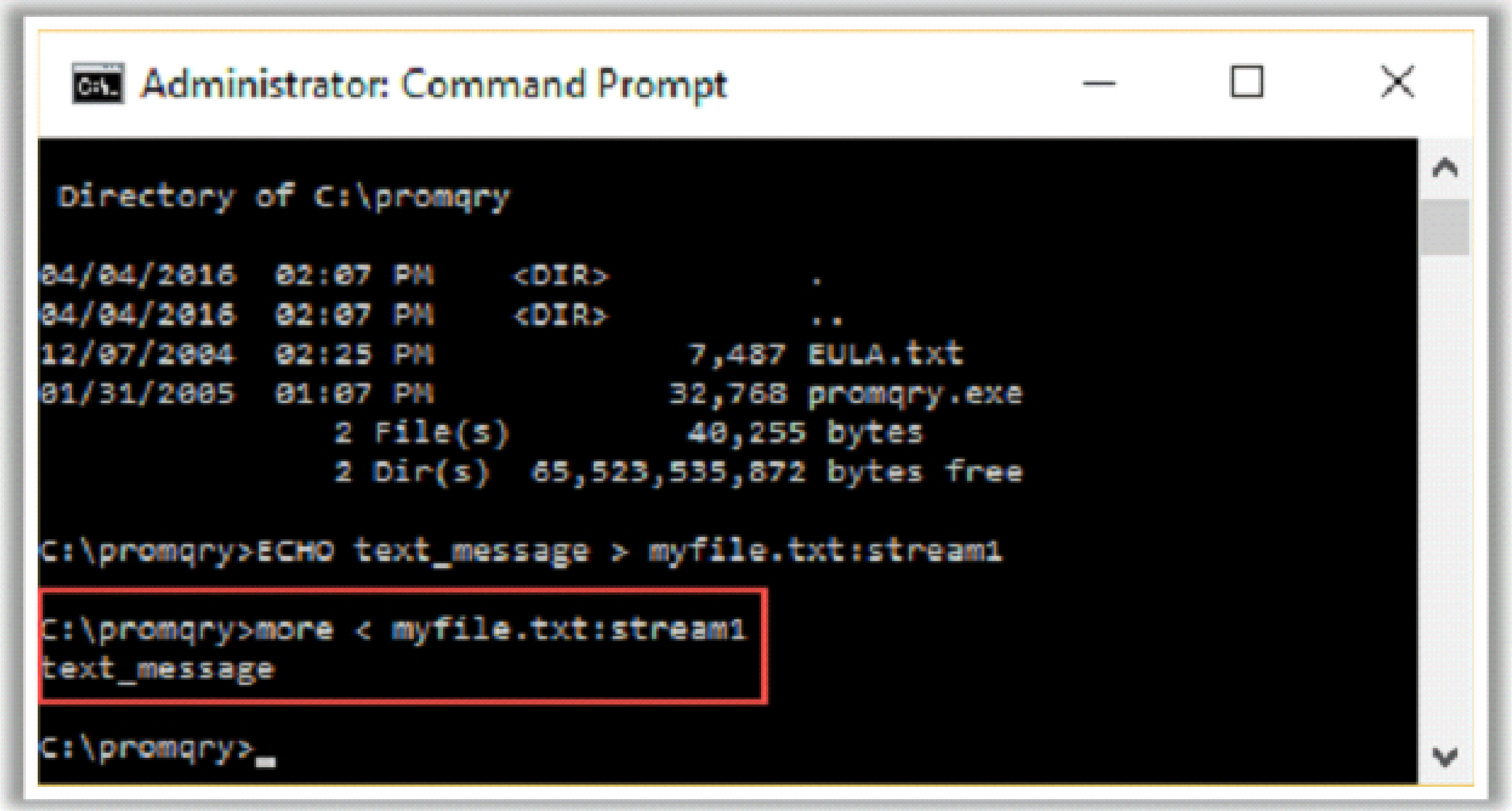
NTFS Data Stream (Cont'd)



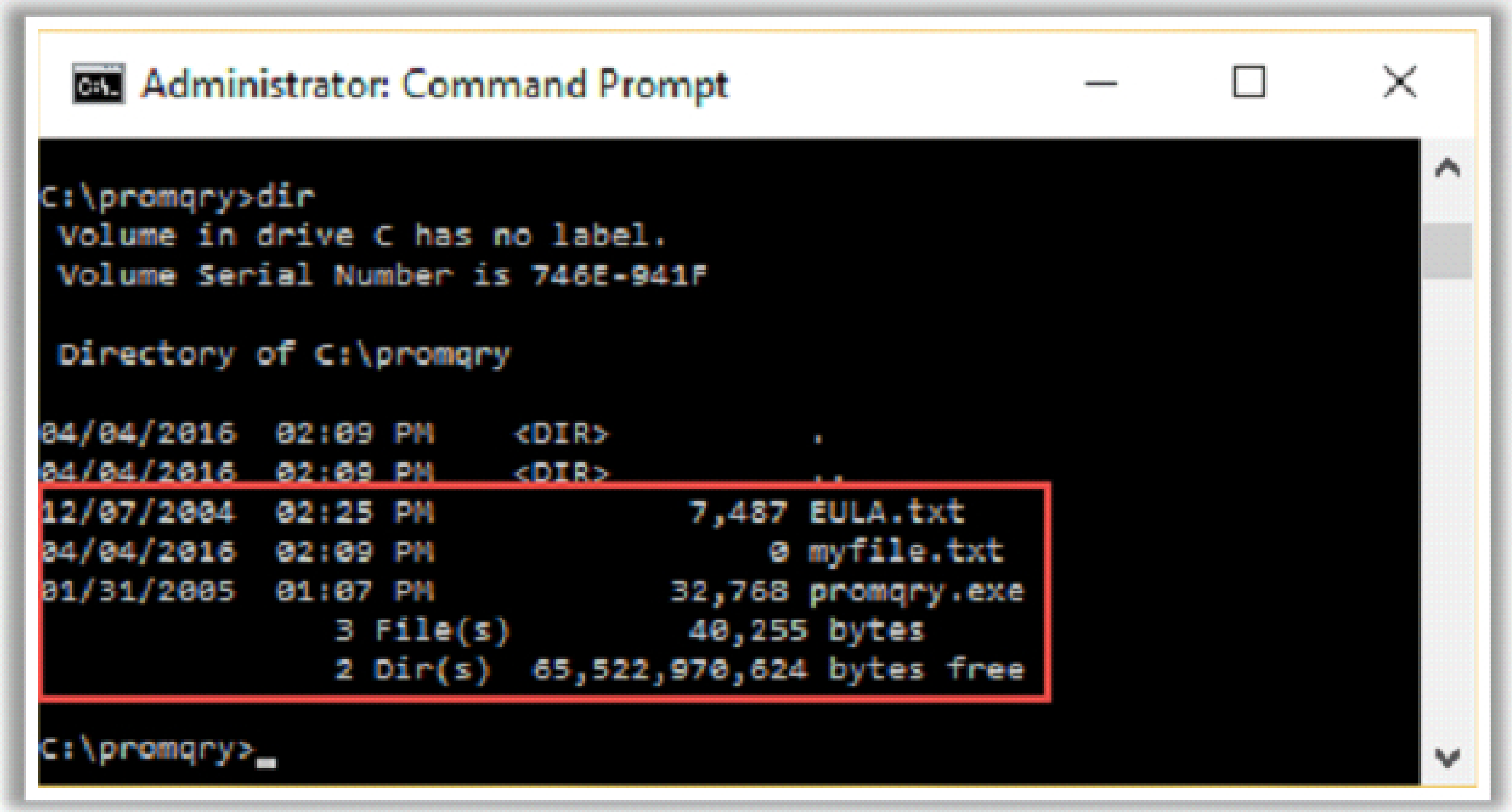
```
C:\promqry>dir
 Volume in drive C has no label.
 Volume Serial Number is 746E-941F

 Directory of C:\promqry

04/04/2016  02:07 PM    <DIR>          .
04/04/2016  02:07 PM    <DIR>          ..
12/07/2004  02:25 PM             7,487 EULA.txt
01/31/2005  01:07 PM            32,768 promqry.exe
               2 File(s)         40,255 bytes
               2 Dir(s)  65,523,535,872 bytes free

C:\promqry>_
```



```
C:\promqry>dir
 Volume in drive C has no label.
 Volume Serial Number is 746E-941F

 Directory of C:\promqry

04/04/2016  02:07 PM    <DIR>          .
04/04/2016  02:07 PM    <DIR>          ..
12/07/2004  02:25 PM             7,487 EULA.txt
01/31/2005  01:07 PM            32,768 promqry.exe
               2 File(s)         40,255 bytes
               2 Dir(s)  65,523,535,872 bytes free

C:\promqry>ECHO text_message > myfile.txt:stream1

C:\promqry>
```



```
 Directory of C:\promqry

04/04/2016  02:07 PM    <DIR>          .
04/04/2016  02:07 PM    <DIR>          ..
12/07/2004  02:25 PM             7,487 EULA.txt
01/31/2005  01:07 PM            32,768 promqry.exe
               2 File(s)         40,255 bytes
               2 Dir(s)  65,523,535,872 bytes free

C:\promqry>ECHO text_message > myfile.txt:stream1

C:\promqry>more < myfile.txt:stream1
text_message

C:\promqry>_
```



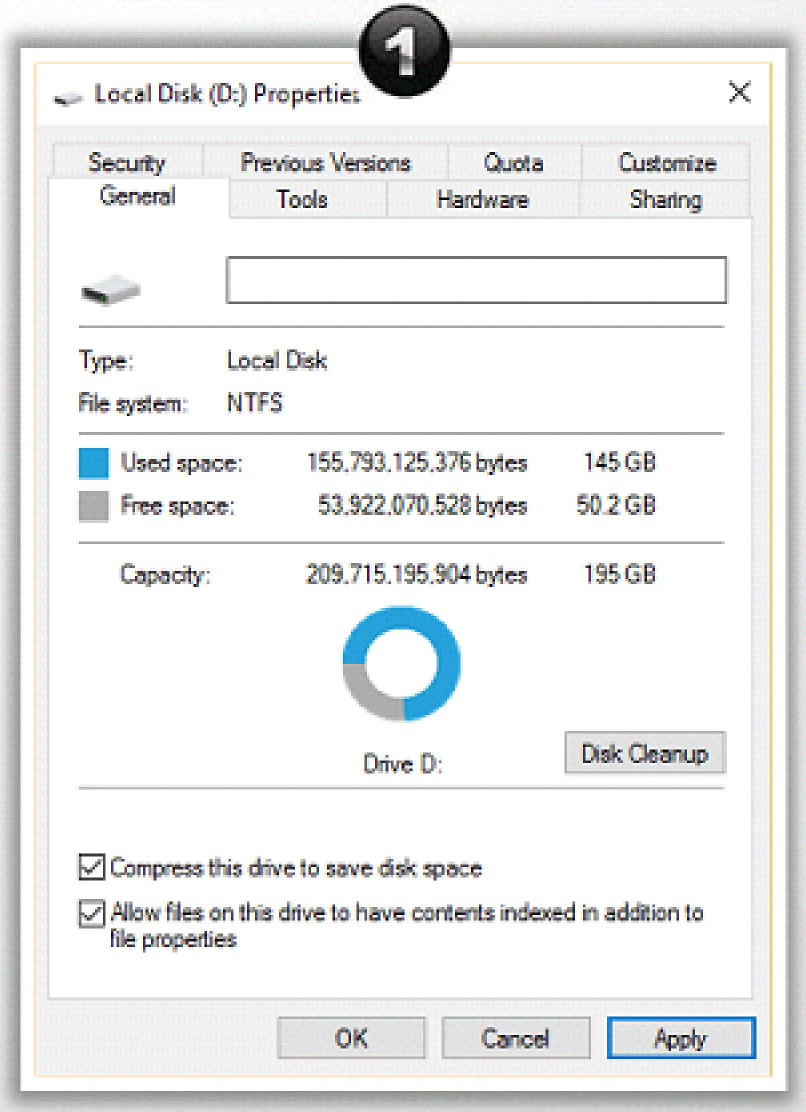
```
C:\promqry>dir
 Volume in drive C has no label.
 Volume Serial Number is 746E-941F

 Directory of C:\promqry

04/04/2016  02:09 PM    <DIR>          .
04/04/2016  02:09 PM    <DIR>          ..
12/07/2004  02:25 PM             7,487 EULA.txt
04/04/2016  02:09 PM                 0 myfile.txt
01/31/2005  01:07 PM            32,768 promqry.exe
               3 File(s)         40,255 bytes
               2 Dir(s)  65,522,970,624 bytes free

C:\promqry>_
```

NTFS Compressed Files

- The compressed files present on an **NTFS volume can be read and written** by any Windows-based application without **first being decompressed by another program**
- NTFS **promotes compression** of individual files, all the files within a folder, and all the files/folders within an NTFS volume
- The file is automatically **decompressed by filter driver** when Windows applications requests the access
- NTFS compression algorithms **support cluster sizes of up to 4 KB**

Setting the Compression State of a Volume:

- Right-click on the drive that is to be compressed and click **Properties**
- On the **General** tab, choose "**Compress this drive to save disk space**" check box and click **Apply**
- In the **Confirm Attribute Changes** dialog box, choose an option and click **OK**



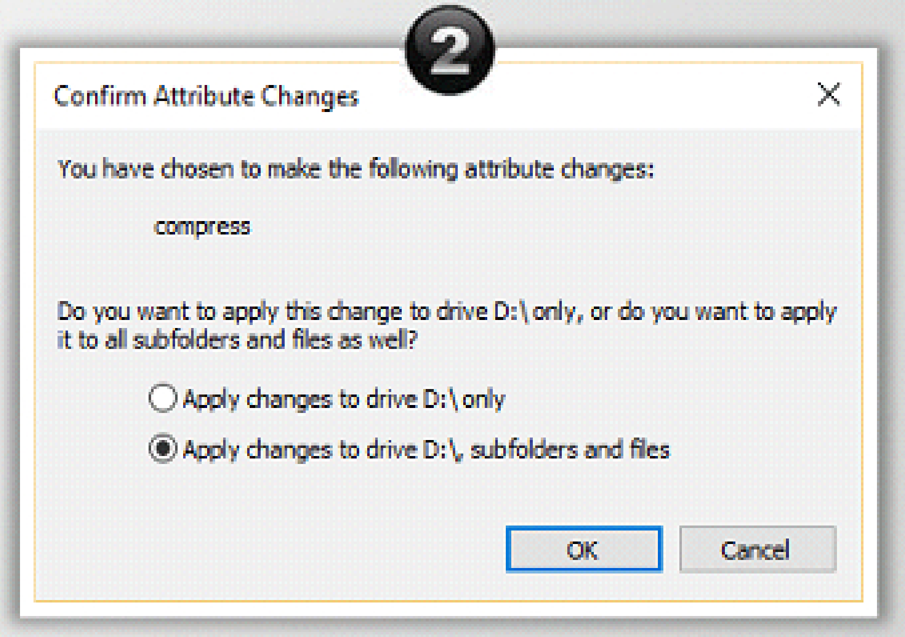
Type: Local Disk

File system: NTFS

🟦 Used space:	155,793,125,376 bytes	145 GB
▨ Free space:	53,922,070,528 bytes	50.2 GB
Capacity:	209,715,195,904 bytes	195 GB

☑ Compress this drive to save disk space

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



You have chosen to make the following attribute changes:

compress

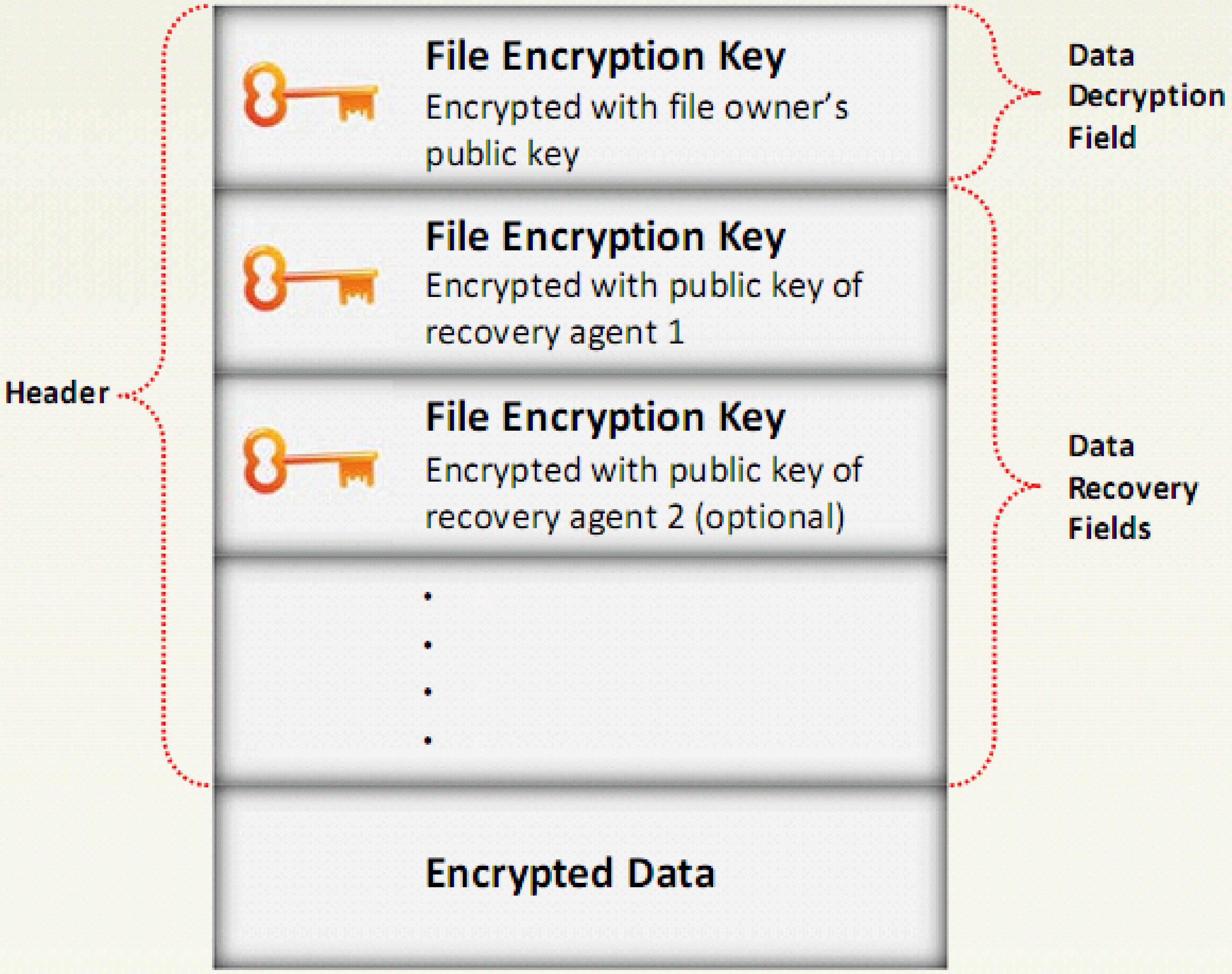
Do you want to apply this change to drive D:\ only, or do you want to apply it to all subfolders and files as well?

○ Apply changes to drive D:\ only

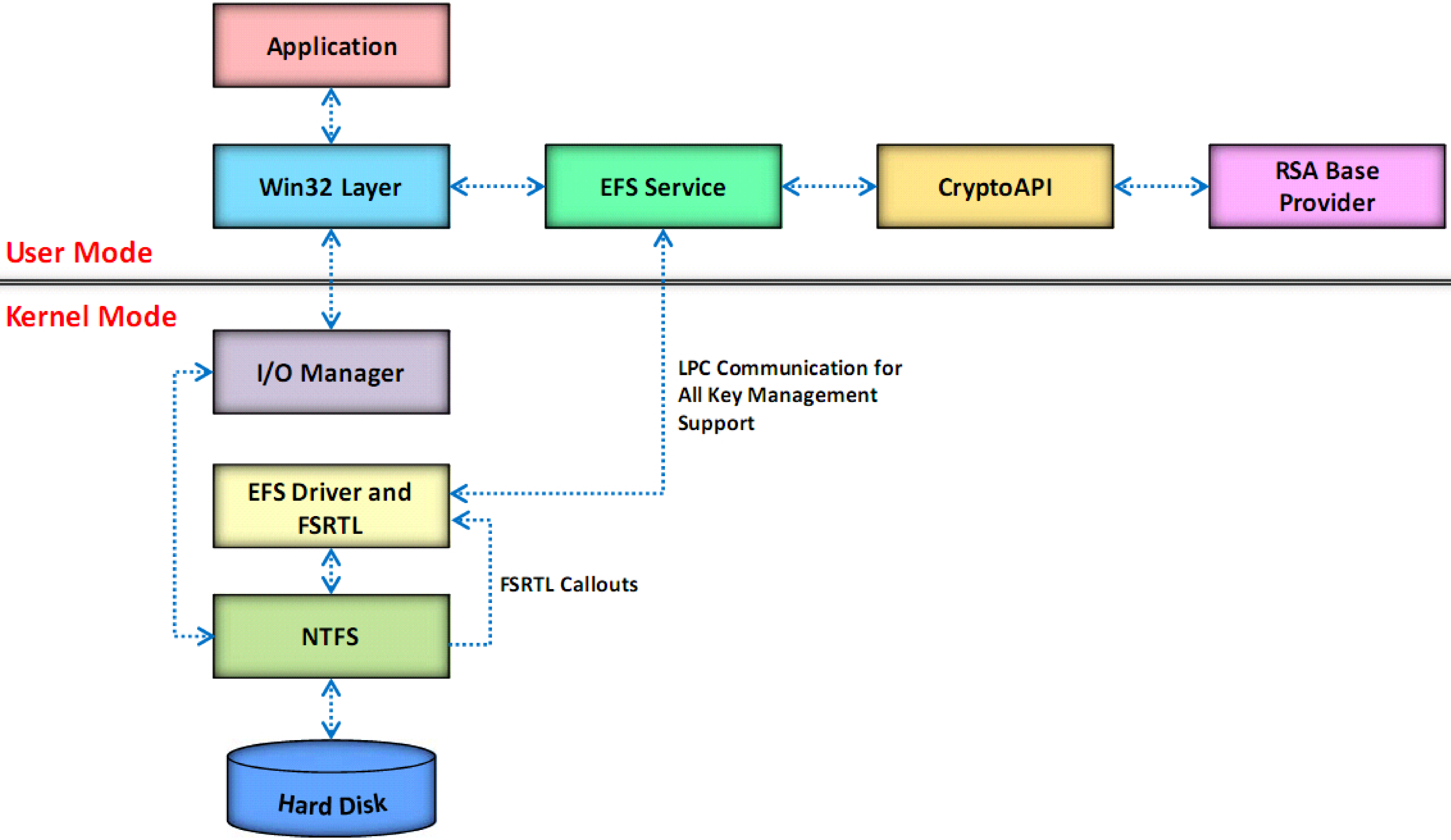
● Apply changes to drive D:\, subfolders and files

Encrypting File Systems (EFS)

- Encrypting File System (EFS) was first introduced in version 3.0 of NTFS, that offers filesystem-level encryption
- This encryption technology maintains a level of transparency to the user who encrypted the file, which means there is no need for users to decrypt the file to access it to make changes
- After a user is done with the file, the encryption policy is automatically restored
- When any unauthorized user tries to access an encrypted file, he or she is denied access
- To enable the encryption and decryption facilities, a user has to set the encryption attributes of the files and folders that the user wants to encrypt or decrypt

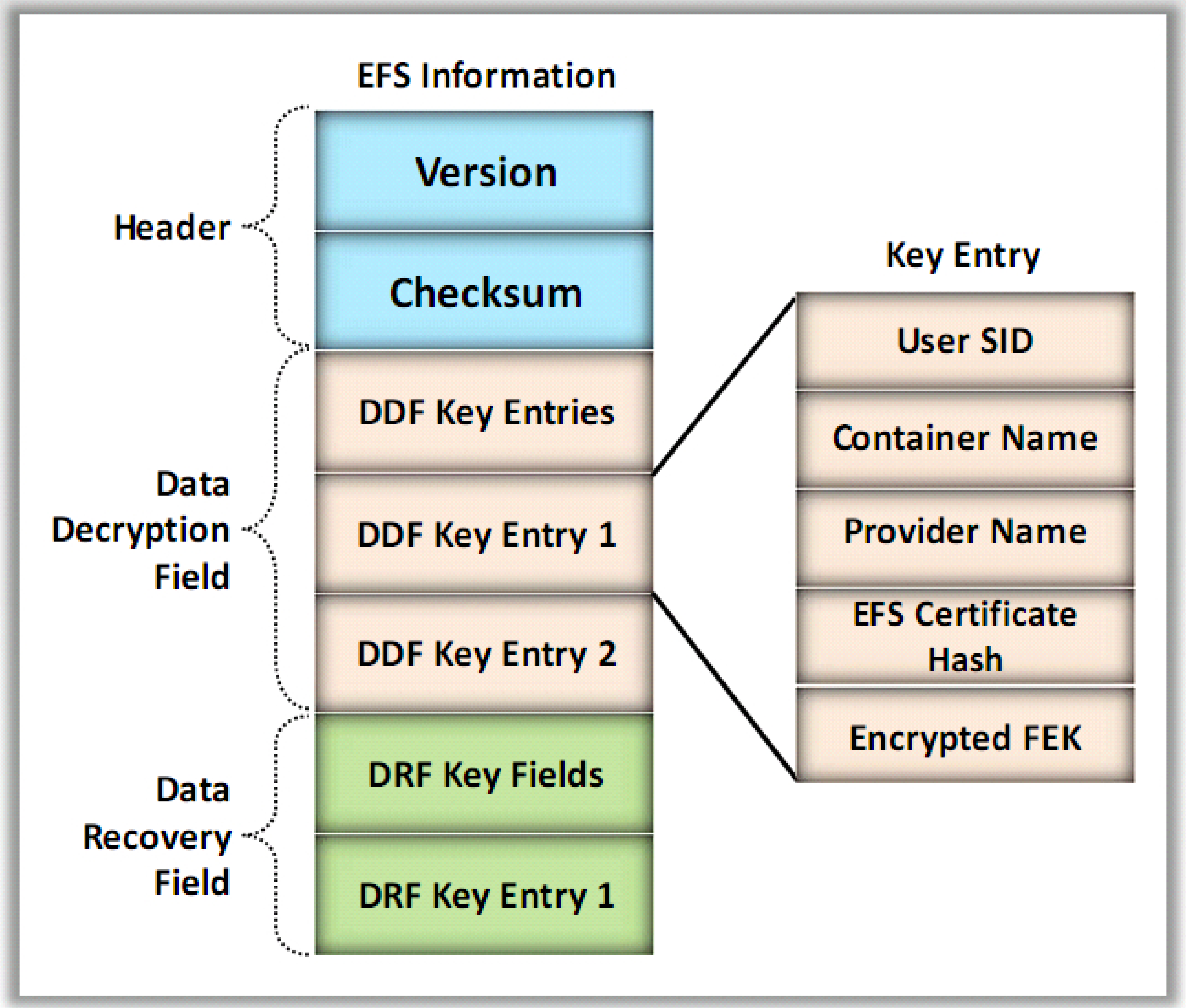

File Encryption Key
Encrypted with file owner's public key
File Encryption Key
Encrypted with public key of recovery agent 1
File Encryption Key
Encrypted with public key of recovery agent 2 (optional)
Encrypted Data
Header
Data Decryption Field
Data Recovery Fields

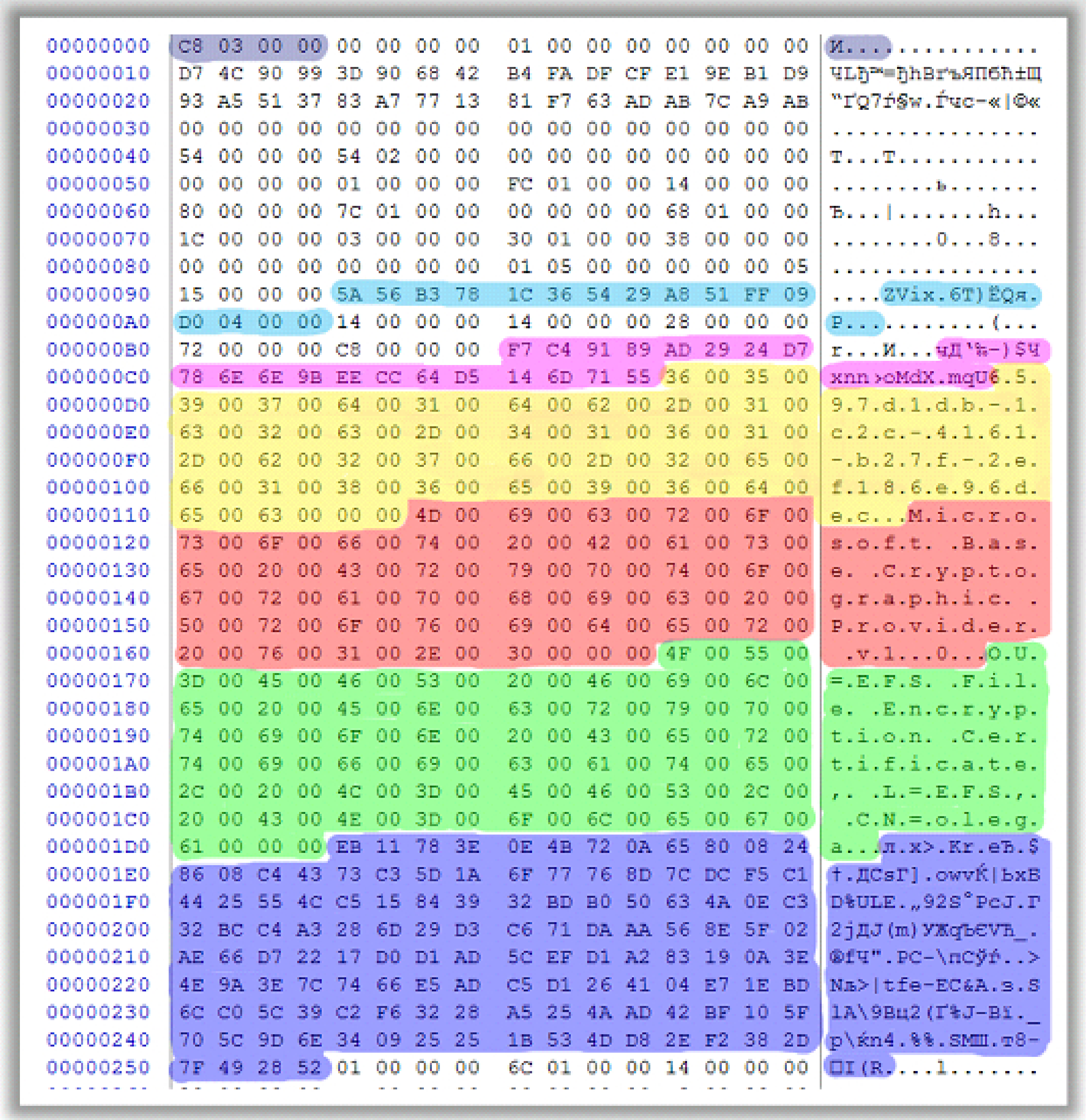
Components of EFS



EFS Attribute

- NTFS sets a flag for the file once you encrypt it and creates an **EFS attribute** where it stores **Data Decryption Field** (DDF) and **Data Recovery Field** (DDR)
- This attribute has **Attribute ID = 0x100** in NTFS





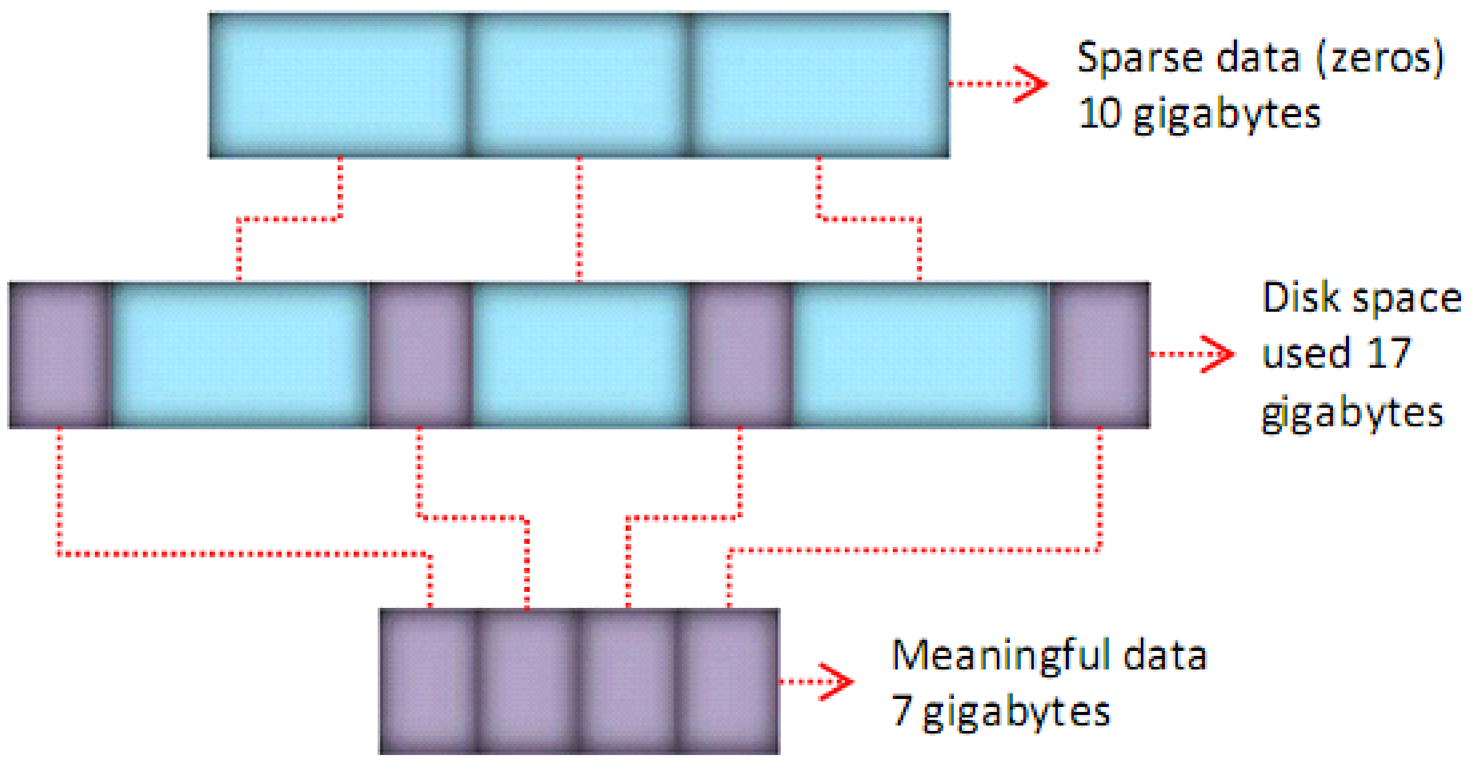
```
00000000  C8 03 00 00 00 00 00 00  01 00 00 00 00 00 00 00  И...............
00000010  D7 4C 90 99 3D 90 68 42  B4 FA DF CF E1 9E B1 D9  ЧLђ™=ђhBґъЯПбћ±Щ
00000020  93 A5 51 37 83 A7 77 13  81 F7 63 AD AB 7C A9 AB  “ҐQ7ѓ§w.Ѓчc-«|©«
00000030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00000040  54 00 00 00 54 02 00 00  00 00 00 00 00 00 00 00  T...T...........
00000050  00 00 00 00 01 00 00 00  FC 01 00 00 14 00 00 00  ........ь.......
00000060  80 00 00 00 7C 01 00 00  00 00 00 00 68 01 00 00  Ђ...|.......h...
00000070  1C 00 00 00 03 00 00 00  30 01 00 00 38 00 00 00  ........0...8...
00000080  00 00 00 00 00 00 00 00  01 05 00 00 00 00 00 05  ................
00000090  15 00 00 00 5A 56 B3 78  1C 36 54 29 A8 51 FF 09  ....ZVіx.6T)ЁQя.
000000A0  D0 04 00 00 14 00 00 00  14 00 00 00 28 00 00 00  Р...........(...
000000B0  72 00 00 00 C8 00 00 00  F7 C4 91 89 AD 29 24 D7  r...И...чД‘‰-)$Ч
000000C0  78 6E 6E 9B EE CC 64 D5  14 6D 71 55 36 00 35 00  xnn›оМdХ.mqU6.5.
000000D0  39 00 37 00 64 00 31 00  64 00 62 00 2D 00 31 00  9.7.d.1.d.b.-.1.
000000E0  63 00 32 00 63 00 2D 00  34 00 31 00 36 00 31 00  c.2.c.-.4.1.6.1.
000000F0  2D 00 62 00 32 00 37 00  66 00 2D 00 32 00 65 00  -.b.2.7.f.-.2.e.
00000100  66 00 31 00 38 00 36 00  65 00 39 00 36 00 64 00  f.1.8.6.e.9.6.d.
00000110  65 00 63 00 00 00 4D 00  69 00 63 00 72 00 6F 00  e.c...M.i.c.r.o.
00000120  73 00 6F 00 66 00 74 00  20 00 42 00 61 00 73 00  s.o.f.t. .B.a.s.
00000130  65 00 20 00 43 00 72 00  79 00 70 00 74 00 6F 00  e. .C.r.y.p.t.o.
00000140  67 00 72 00 61 00 70 00  68 00 69 00 63 00 20 00  g.r.a.p.h.i.c. .
00000150  50 00 72 00 6F 00 76 00  69 00 64 00 65 00 72 00  P.r.o.v.i.d.e.r.
00000160  20 00 76 00 31 00 2E 00  30 00 00 00 4F 00 55 00   .v.1...0...O.U.
00000170  3D 00 45 00 46 00 53 00  20 00 46 00 69 00 6C 00  =.E.F.S. .F.i.l.
00000180  65 00 20 00 45 00 6E 00  63 00 72 00 79 00 70 00  e. .E.n.c.r.y.p.
00000190  74 00 69 00 6F 00 6E 00  20 00 43 00 65 00 72 00  t.i.o.n. .C.e.r.
000001A0  74 00 69 00 66 00 69 00  63 00 61 00 74 00 65 00  t.i.f.i.c.a.t.e.
000001B0  2C 00 20 00 4C 00 3D 00  45 00 46 00 53 00 2C 00  ,. .L.=.E.F.S.,.
000001C0  20 00 43 00 4E 00 3D 00  6F 00 6C 00 65 00 67 00   .C.N.=.o.l.e.g.
000001D0  61 00 00 00 EB 11 78 3E  0E 4B 72 0A 65 80 08 24  a...л.x>.Kr.eЂ.$
000001E0  86 08 C4 43 73 C3 5D 1A  6F 77 76 8D 7C DC F5 C1  †.ДCsГ].owvЌ|ЬхБ
000001F0  44 25 55 4C C5 15 84 39  32 BD B0 50 63 4A 0E C3  D%ULЕ.„92Ѕ°PcJ.Г
00000200  32 BC C4 A3 28 6D 29 D3  C6 71 DA AA 56 8E 5F 02  2јДЈ(m)УЖqЪЄVЋ_.
00000210  AE 66 D7 22 17 D0 D1 AD  5C EF D1 A2 83 19 0A 3E  ®fЧ".РС-\пСўѓ..>
00000220  4E 9A 3E 7C 74 66 E5 AD  C5 D1 26 41 04 E7 1E BD  Nљ>|tfе-ЕС&A.з.Ѕ
00000230  6C C0 5C 39 C2 F6 32 28  A5 25 4A AD 42 BF 10 5F  lА\9Вц2(Ґ%J-Bї._
00000240  70 5C 9D 6E 34 09 25 25  1B 53 4D D8 2E F2 38 2D  p\ќn4.%%.SMШ.т8-
00000250  7F 49 28 52 01 00 00 00  6C 01 00 00 14 00 00 00  .I(R....l.......
```

Sparse Files

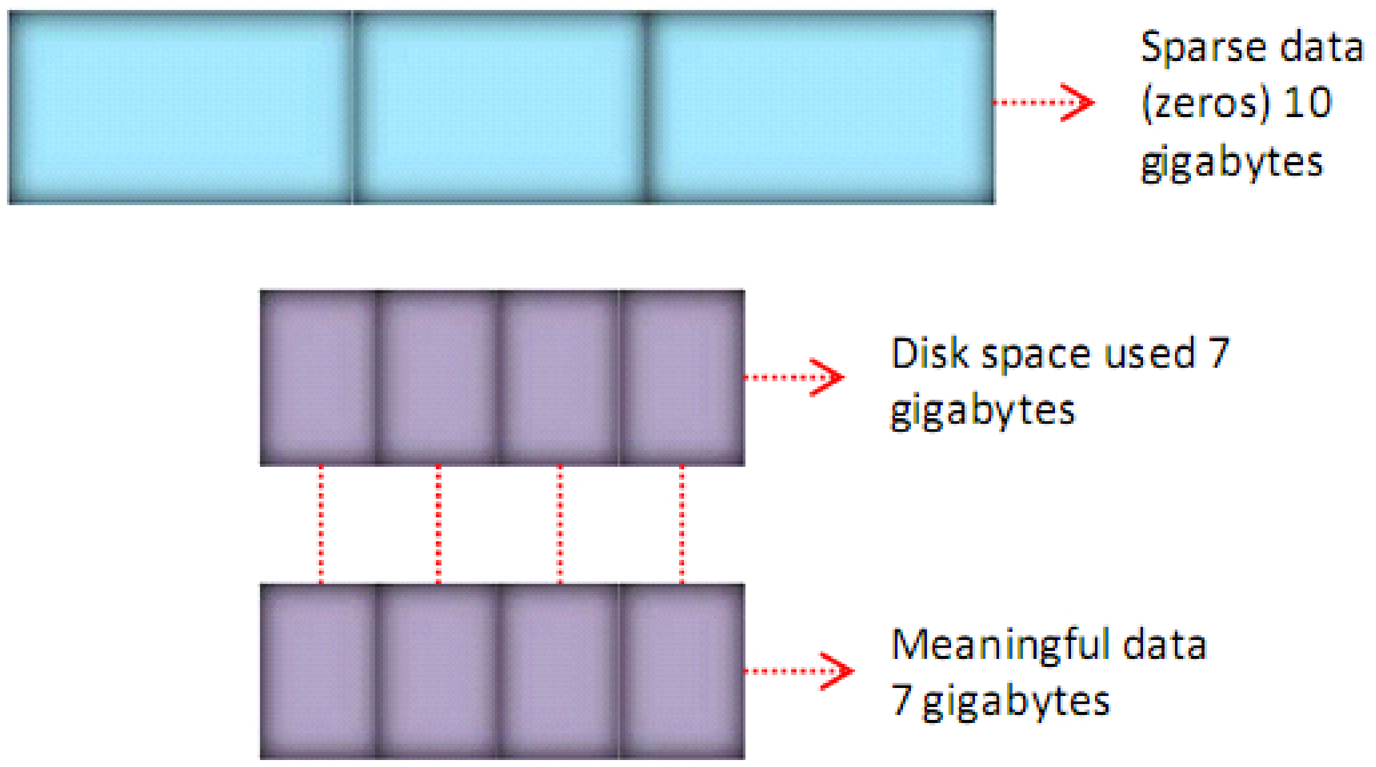
Sparse files provide a method of saving disk space for files by allowing I/O subsystem to allocate only meaningful (nonzero) data

If NTFS file is marked as sparse, it assigns hard disk cluster only for the data defined by the application

Non-defined data of the file are represented by non-allocated space on the disk


Without Sparse File Attribute Set

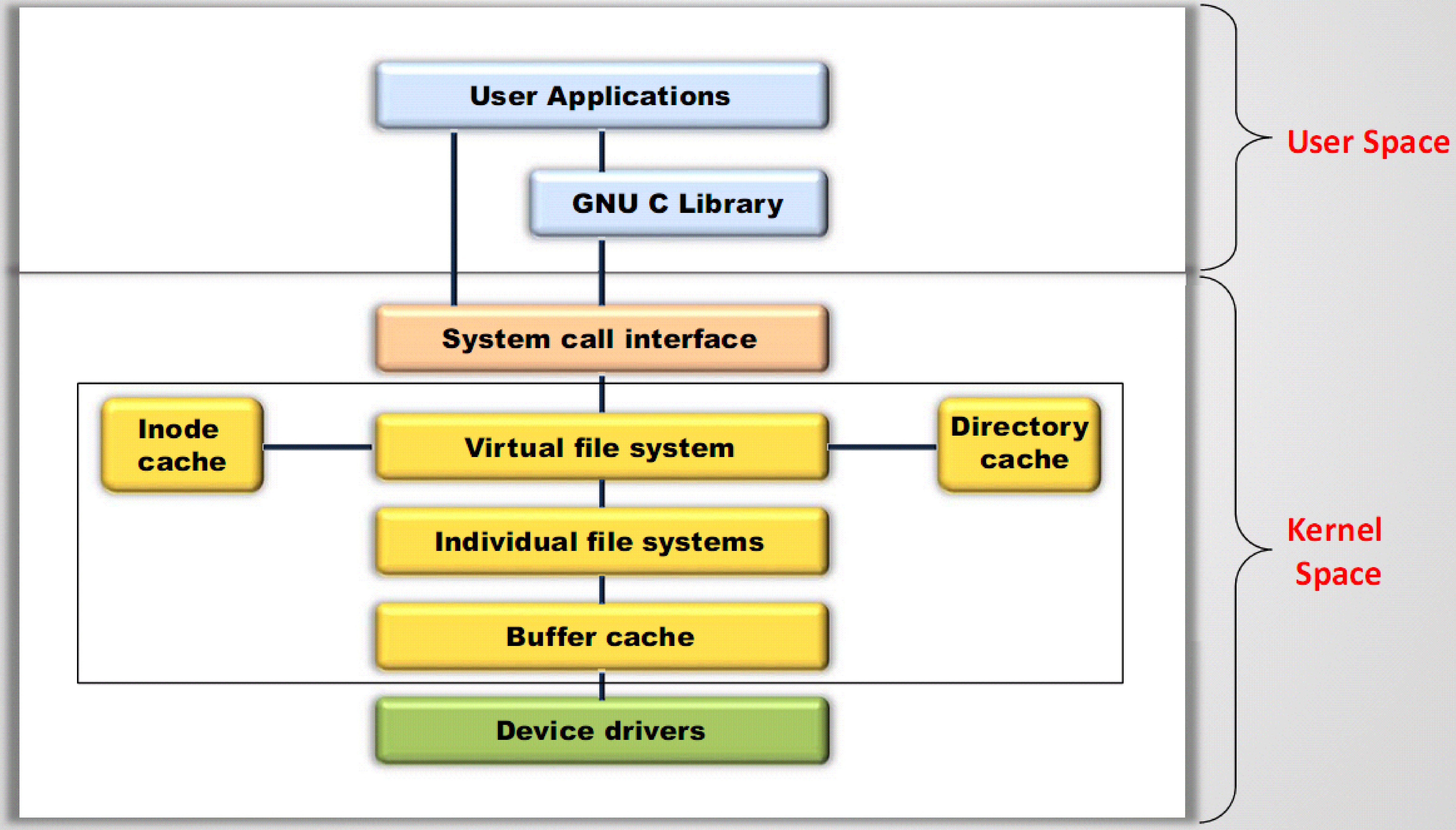
Sparse data (zeros) 10 gigabytes
Disk space used 17 gigabytes
Meaningful data 7 gigabytes


With Sparse File Attribute Set

Sparse data (zeros) 10 gigabytes
Disk space used 7 gigabytes
Meaningful data 7 gigabytes

Linux File Systems

Linux File System Architecture



Filesystem Hierarchy Standard (FHS)

- The **Filesystem Hierarchy Standard (FHS)** defines the directory structure and its contents in Linux and Unix-like operating systems
- In the **FHS**, all files and directories are present under the root directory (represented by /)

Table displaying directories and their description specific to the FHS

Directory	Description
/bin	Essential command binaries. Ex: cat, ls, cp.
/boot	Static files of the boot loader. Ex: Kernels, Initrd
/dev	Essential device files. Ex: /dev/null
/etc	Host-specific system configuration files
/home	Users' home directories, holding saved files, personal settings, etc.
/lib	Essential libraries for the binaries in /bin/ and /sbin/
/media	Mount points for removable media
/mnt	Temporarily mounted filesystems
/opt	Add-on application software packages
/root	Home directory for the root user
/proc	Virtual file system providing process and kernel information as files
/run	Information about running processes. Ex: running daemons, currently logged-In users
/sbin	Contains the binary files required for working
/srv	Site-specific data for services provided by the system
/tmp	Temporary files
/usr	Secondary hierarchy for read-only user data
/var	Variable data. Ex: logs, spool files, etc.

Extended File System (EXT)

- First file system for the Linux operating system to overcome certain limitations of the **Minix file system**
- It has a maximum partition size of 2 GB and a maximum file name size of 255 characters
- It removes the two major Minix file system limitations of a **64 MB partition size** and **short file names**
- The major limitation of this file system is that it doesn't support separate access, inode modification, and data modification time stamps
- It is replaced by the **second extended file system**

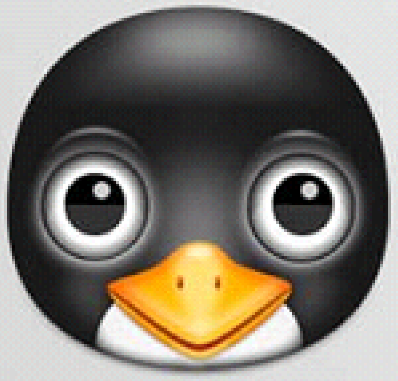
Second Extended File System (EXT2)

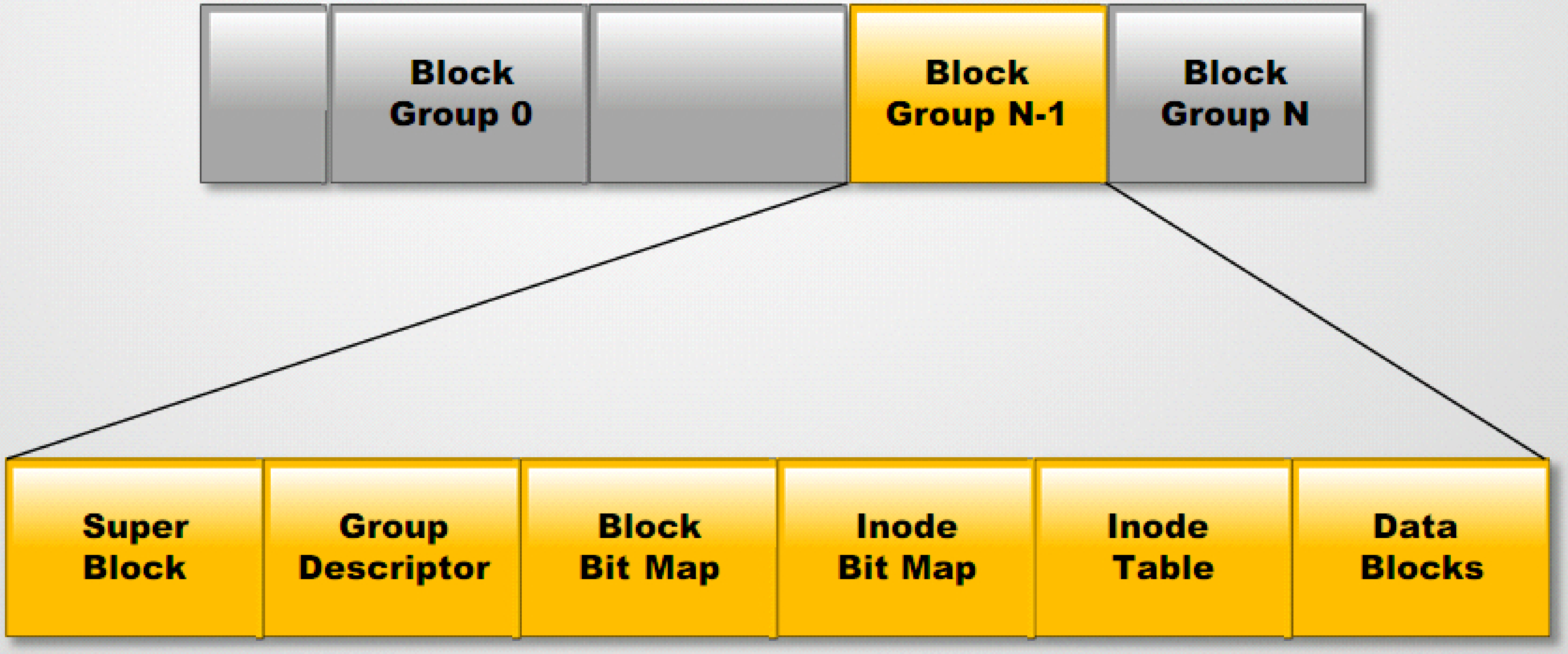
I EXT2 is a standard file system that uses improved algorithms, which greatly enhances its speed significantly, and it maintains additional time stamps

II It maintains a special field in the superblock that keeps track of the file system status and identifies it as either clean or dirty

III Its major shortcomings are the risk of file system corruption when writing to EXT2, and it is not a journaling file system

IV Physical layout of the EXT2 File system:

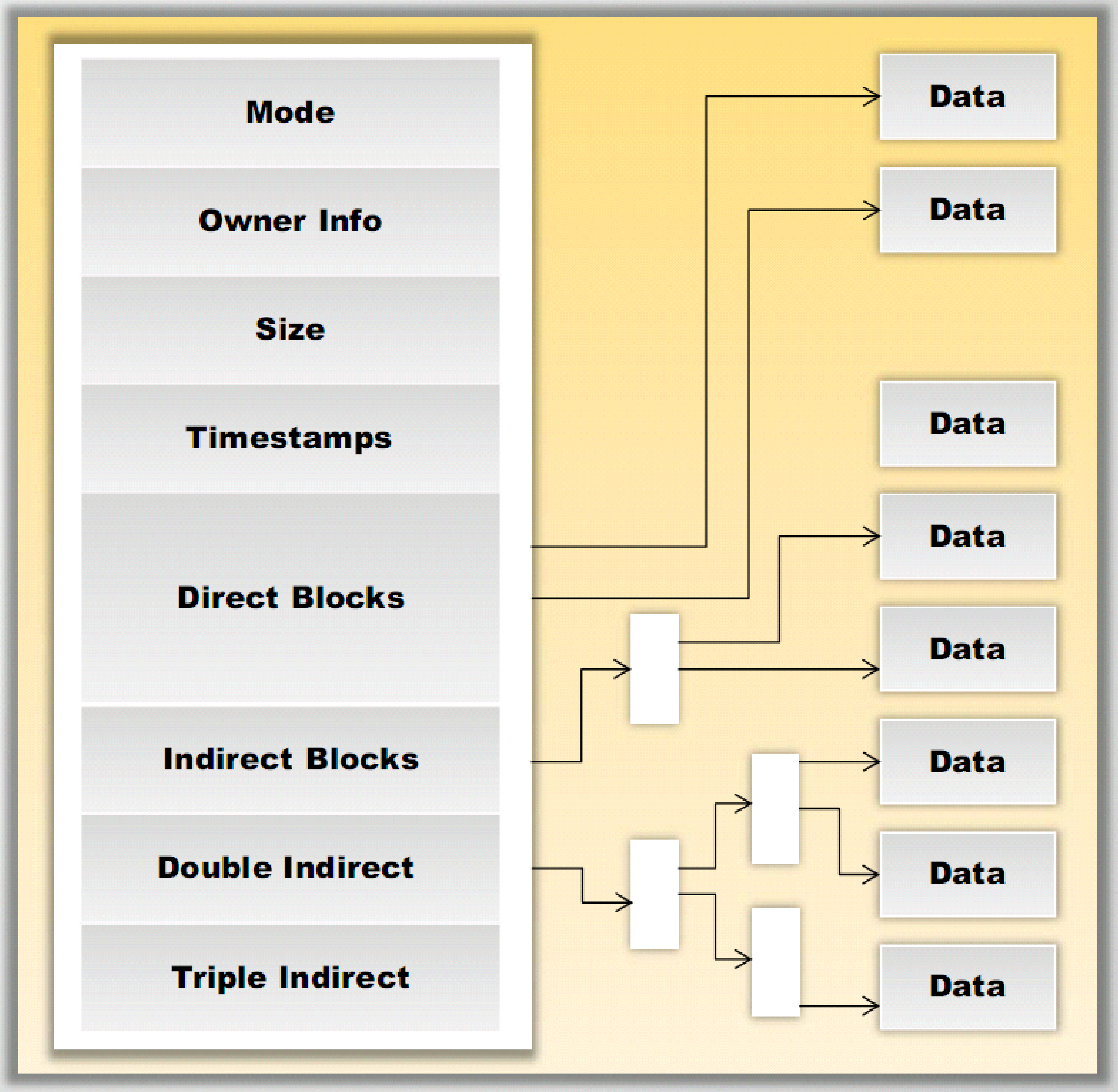




Second Extended File System (EXT2) (Cont'd)

EXT2 Inode

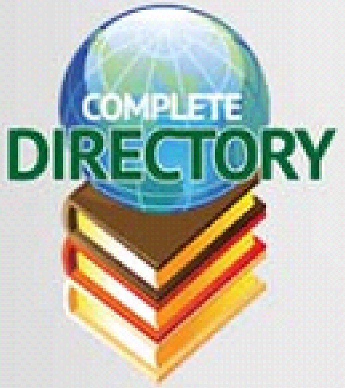
- The inode is a **basic building block** of the EXT2 file system
- Each **file** and **directory** are described by a single inode
- The inodes for each file system block are placed together in an **inode table**

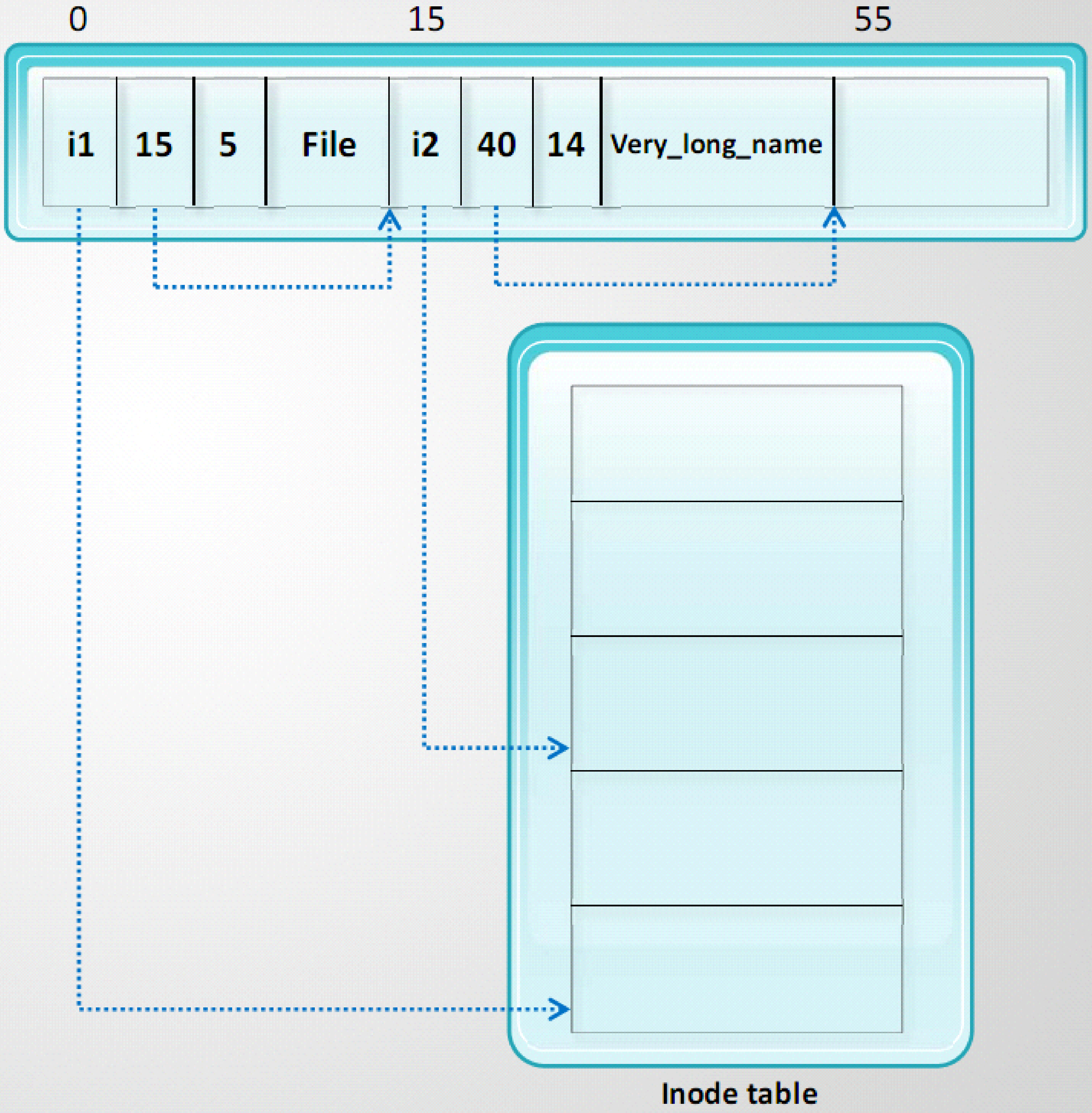


Second Extended File System (EXT2) (Cont'd)

EXT2 Directories

- EXT2 directories are particular files that create and hold the **access path** of the files in the file system
- These files contain the list of **directory entries** with the following information:
  - Directory inode
  - Length of the file name
  - Name of the directory





Third Extended File System (Ext3)

- Ext3 is a journaling version of the EXT2 file system and is greatly used with the Linux operating system
- It is the enhanced version of the **EXT2** file system
- It uses **file system maintenance utilities** (like fsck) for maintaining and repairing alike EXT2 file system
- Command to convert EXT2 to EXT3 file system:

```
➢ # /sbin/tune2fs -j <partition-name>
```

Ext3 Features

Data Integrity

It provides stronger **data integrity** for events that occur due to computer system shutdowns

Speed

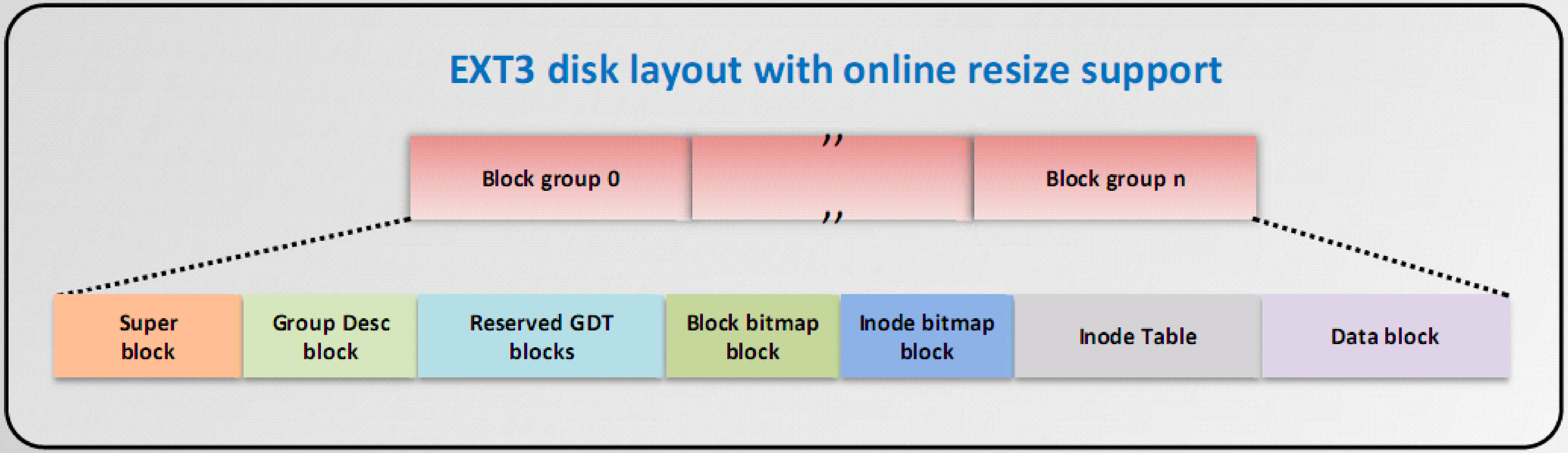
As the EXT3 file system is journaling the file system, it has **higher throughput** in most cases than EXT2

Easy Transition

The user can easily change the file system from EXT2 to EXT3 and **increase the performance** of the system

Third Extended File System (EXT3) (Cont'd)

- **File System Journaling**:
  - It records file system updates which help in quick file system recovering in case of a **system crash**
  - The EXT3 journal uses **inode 8** and its location is stated in the superblock
  - The first block in the EXT3 journal is for **superblock** and contains general information



Fourth Extended File System (EXT4)

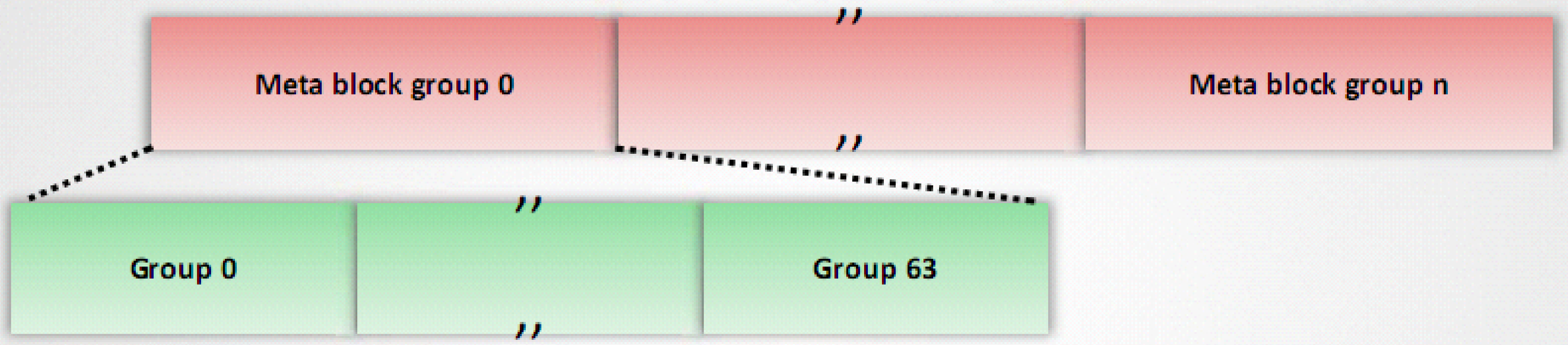
- EXT4 is a journaling file system, developed as the **replacement of commonly used EXT3 file system**
- With incorporation of new features, EXT4 has **significant advantages over EXT3 and EXT2** file systems particularly in terms of performance, scalability, and reliability
- Supports Linux Kernel v2.6.19 onwards

Key Features

- File System Size - supports maximum individual file size 16TB and overall maximum EXT4 file system size 1EB (exabyte)
- Extents - replaces block mapping scheme used by EXT2 and EXT3, improving large file performance and reducing fragmentation
- Delayed allocation - improves performance and reduces fragmentation by effectively allocating larger amounts of data at a time
- Multi-block allocation - allocates files contiguously on disk
- fsck speed - supports faster file system checking
- Journal checksumming - uses checksums in the journal to improve reliability
- Persistent preallocation - preallocates on-disk space for a file
- Improved Timestamps - provides timestamps measured in nanoseconds
- Backward compatibility – makes it possible to mount EXT3 and EXT2 as EXT4

Fourth Extended File System (EXT4) (Cont'd)

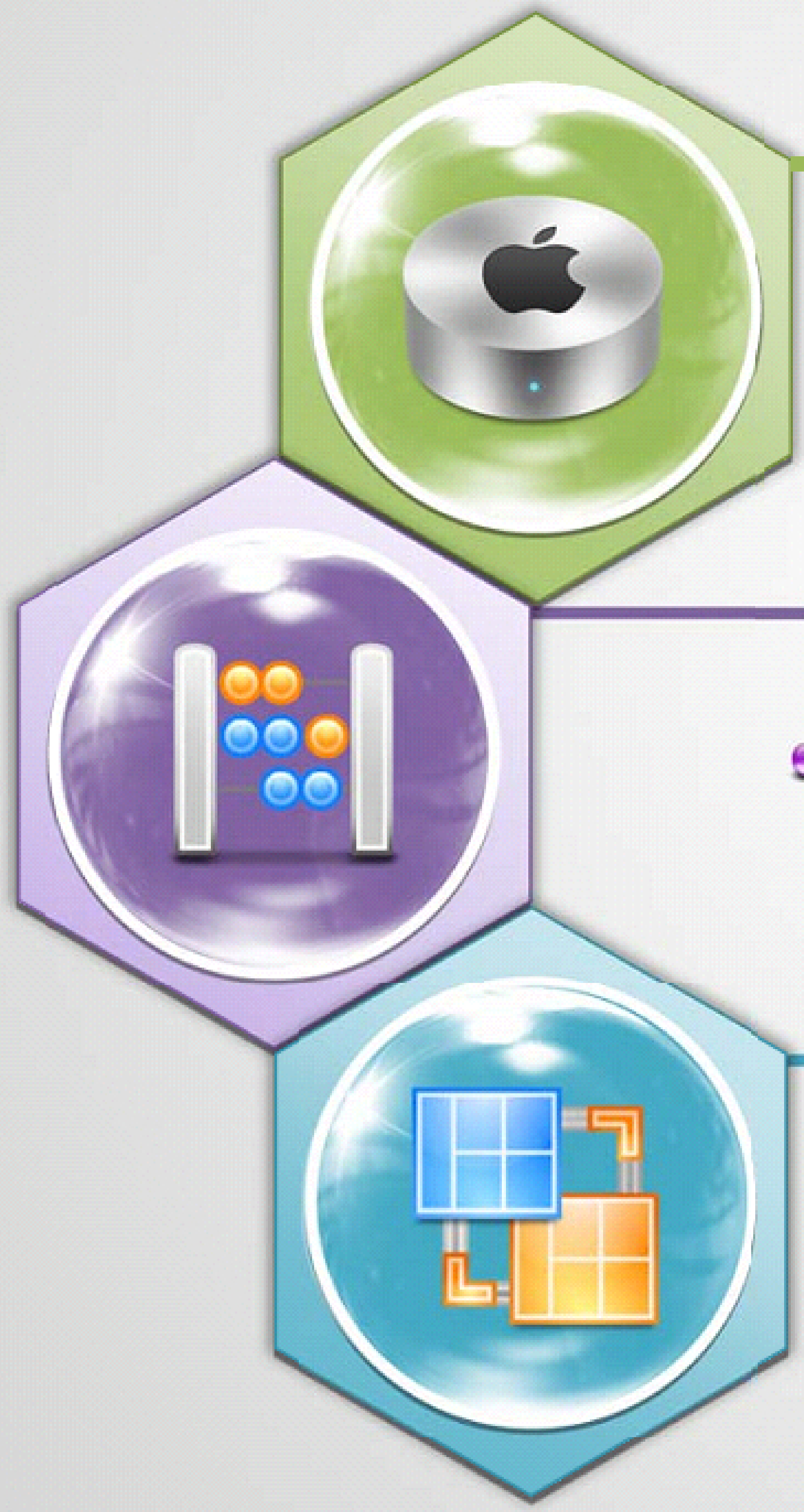
EXT4 disk layout with meta block group



Group 0	Super block	Group Desc block	Block bitmap block	Inode bitmap block	Inode Table	Data block
Group 1	Super block	Group Desc block	Block bitmap block	Inode bitmap block	Inode Table	Data block
Group 2			Block bitmap block	Inode bitmap block	Inode Table	Data block
Group 3		Group Desc block	Block bitmap block	Inode bitmap block	Inode Table	Data block

Mac OS X File Systems

Mac OS X File Systems



Hierarchical File System (HFS)

- Developed by **Apple Computer** to support Mac operating system

HFS Plus

- HFS Plus (HFS+) is a successor of HFS and is used as a **primary file system** in Macintosh

UNIX File System (UFS)

- Derived from the **Berkeley Fast File System (FFS)** that was originally developed at Bell Laboratories from the first version of UNIX FS
- All BSD UNIX derivatives including FreeBSD, NetBSD, OpenBSD, NeXTStep, and Solaris use a variant of UFS
- Acts as a substitute for HFS in Mac OS X

HFS vs. HFS Plus

Feature	HFS	HFS Plus	Benefit/Comment
User visible name	Mac OS Standard		
Number of allocation blocks	16 bits worth	32 bits worth	Radical decrease in disk space used on large volumes, and a large number of files per volume
Long file names	31 characters	255 characters	Obvious user benefit; also improves cross-platform compatibility
File name encoding	MacRoman	Unicode	Allows for international-friendly file names, including mixed script names
File/folder attributes	Support for fixed size attributes (FileInfo and ExtendedFileInfo)	Allows for future metadata extensions	Future systems may use metadata for a richer Finder experience
OS startup support	System Folder ID	Also supports a dedicated startup file	May help non-Mac OS systems to boot from HFS Plus Volumes
Catalog node size	512 bytes	4 KB	Maintains efficiency in the face of the other changes. (This larger catalog node size is due to the much longer file names [512 bytes as opposed to 32 bytes], and larger catalog records (because of more/larger fields)).
Maximum file size	2^{31} bytes	2^{63} bytes	Obvious user benefits, especially for multimedia content creators.

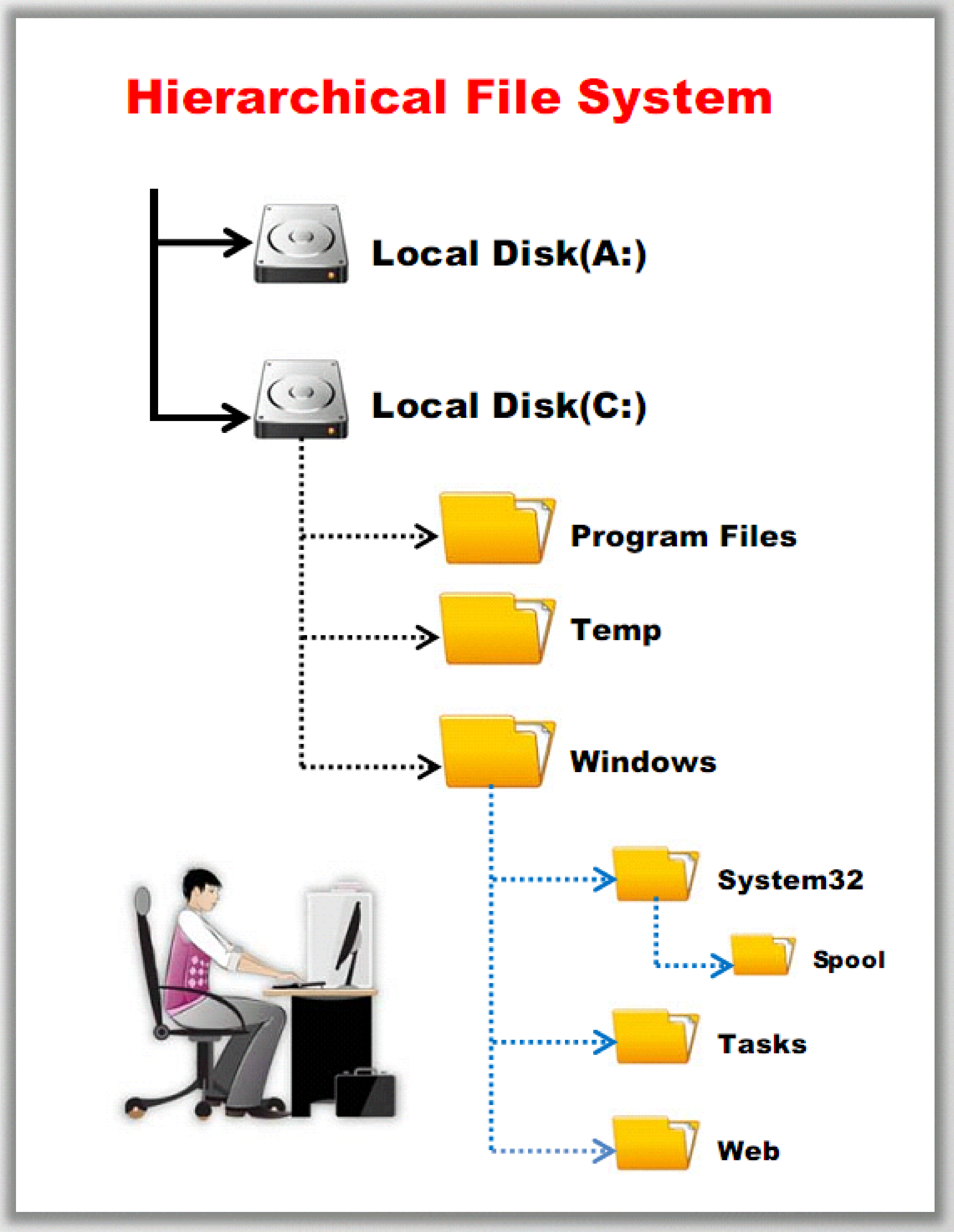
Hierarchical File System (HFS)

Hierarchical File System (HFS, also referred as Mac OS Standard) is a file system designed by Apple in 1985 for MAC operating system

It groups files into directories and each directory also groups with other directories

It displays drives, directories, and files in groups

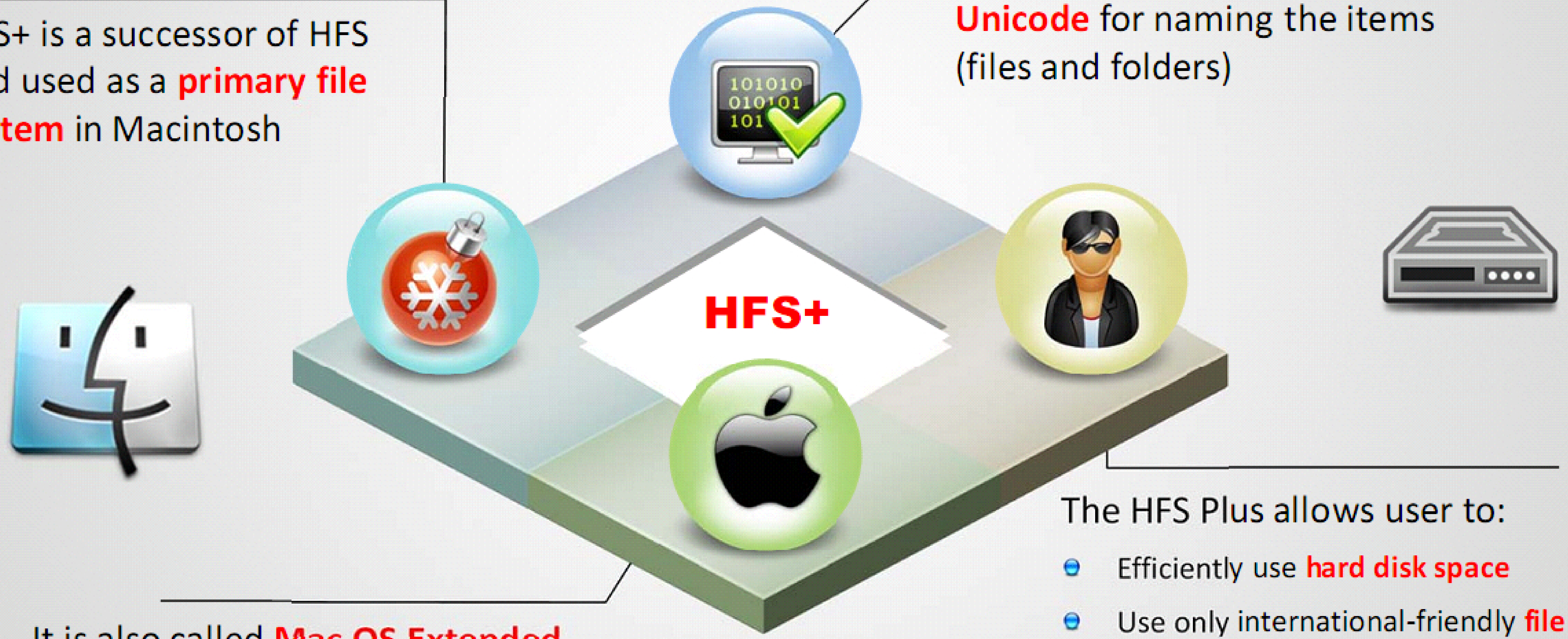
It divides a logical volume into logical blocks of 512 bytes



Hierarchical File System Plus (HFS+)

HFS+ is a successor of HFS and used as a **primary file system** in Macintosh

It supports large files and uses **Unicode** for naming the items (files and folders)



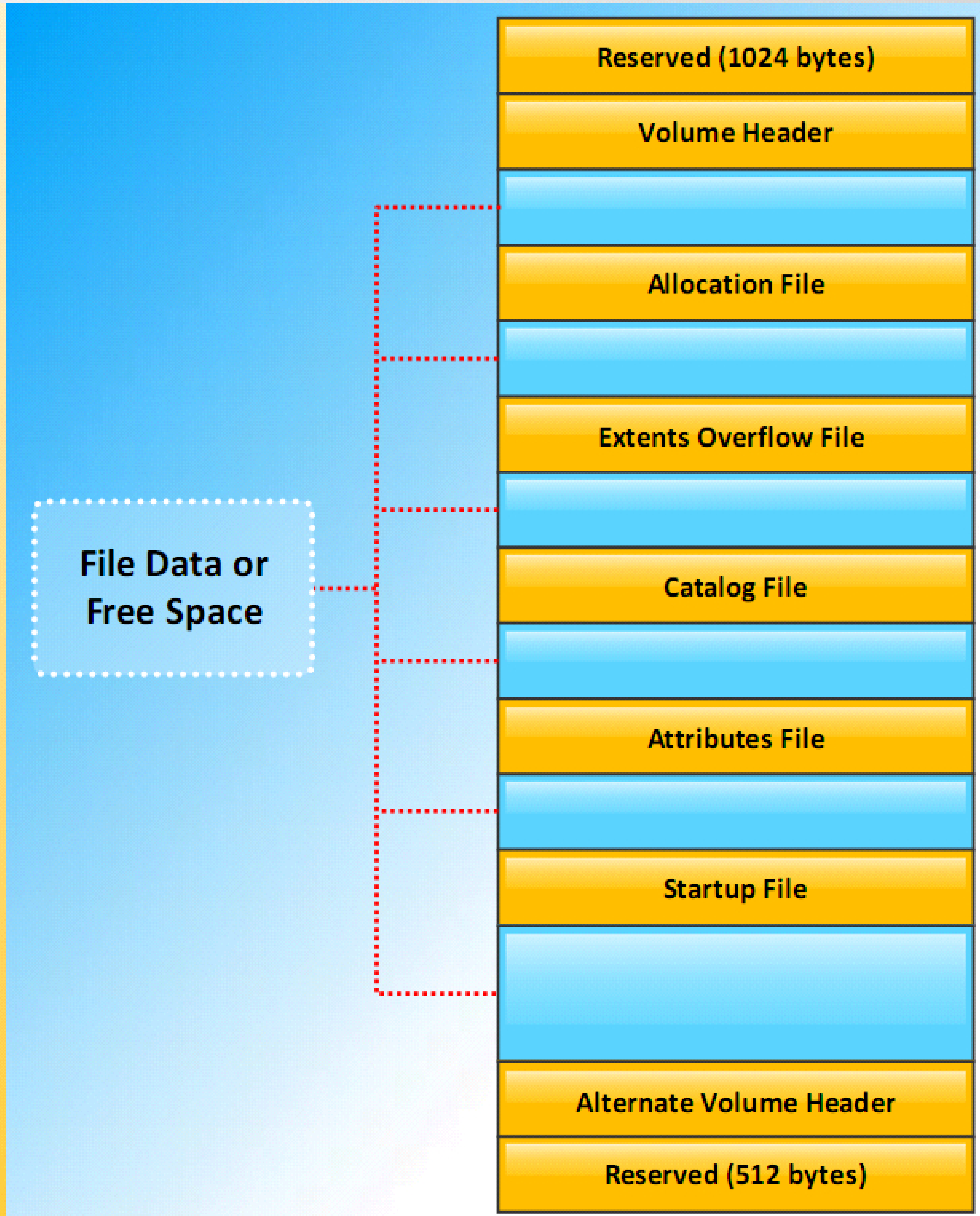
It is also called **Mac OS Extended** (HFS Extended) and is one of the formats used in the Apple iPod

The HFS Plus allows user to:

- Efficiently use **hard disk space**
- Use only international-friendly **file names**
- Easily boot on **non-Mac OS operating systems**

HFS+ Volumes

- HFS+ volumes are divided into **logical blocks** (sectors) of 512 bytes
- These sectors are **clustered** into allocation blocks
- The total number of allocation blocks depends on the **volume size**
- The bulk of an HFS+ volume contains seven types of sectors:
  - User file fork
  - Allocation file
  - Catalog file
  - Extent overflow file
  - Attribute file
  - Startup file
  - Unused space

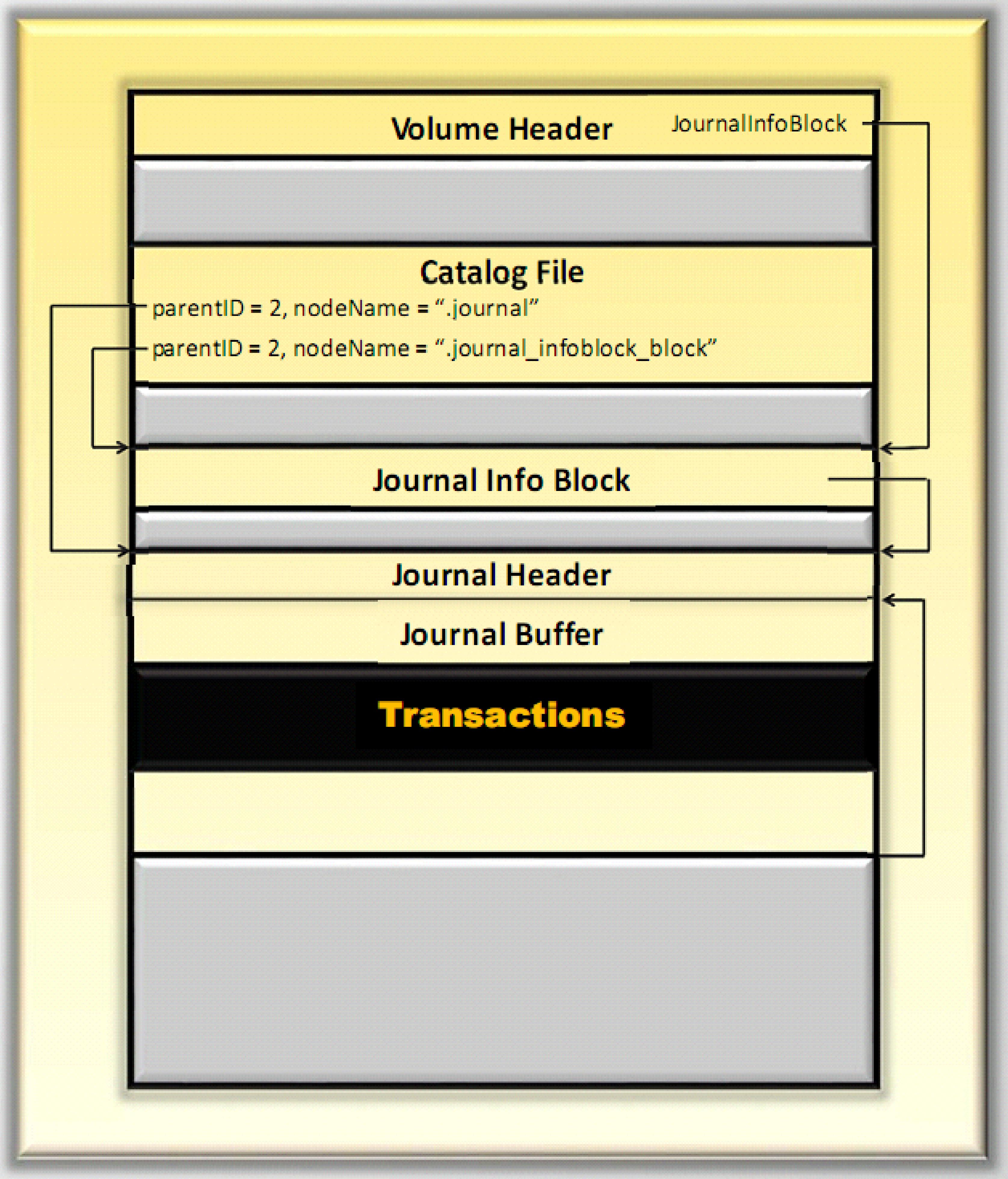


HFS+ Journal

HFS+ volume has an optional journal which helps in mounting an **unmounted volume** in the case of a system crash

Journal **restores** the volume structures to a trustworthy state without scanning all of the structures

The journal info block (.journal_info_block) is stored as a file on the HFS+ volume's **root directory**



Oracle Solaris 11 File System: ZFS

- ZFS is the default **disk-based and root file system** used in the Oracle Solaris 11
- It **provides a simple management interface**, which is robust, scalable, and easy to administer

Features:

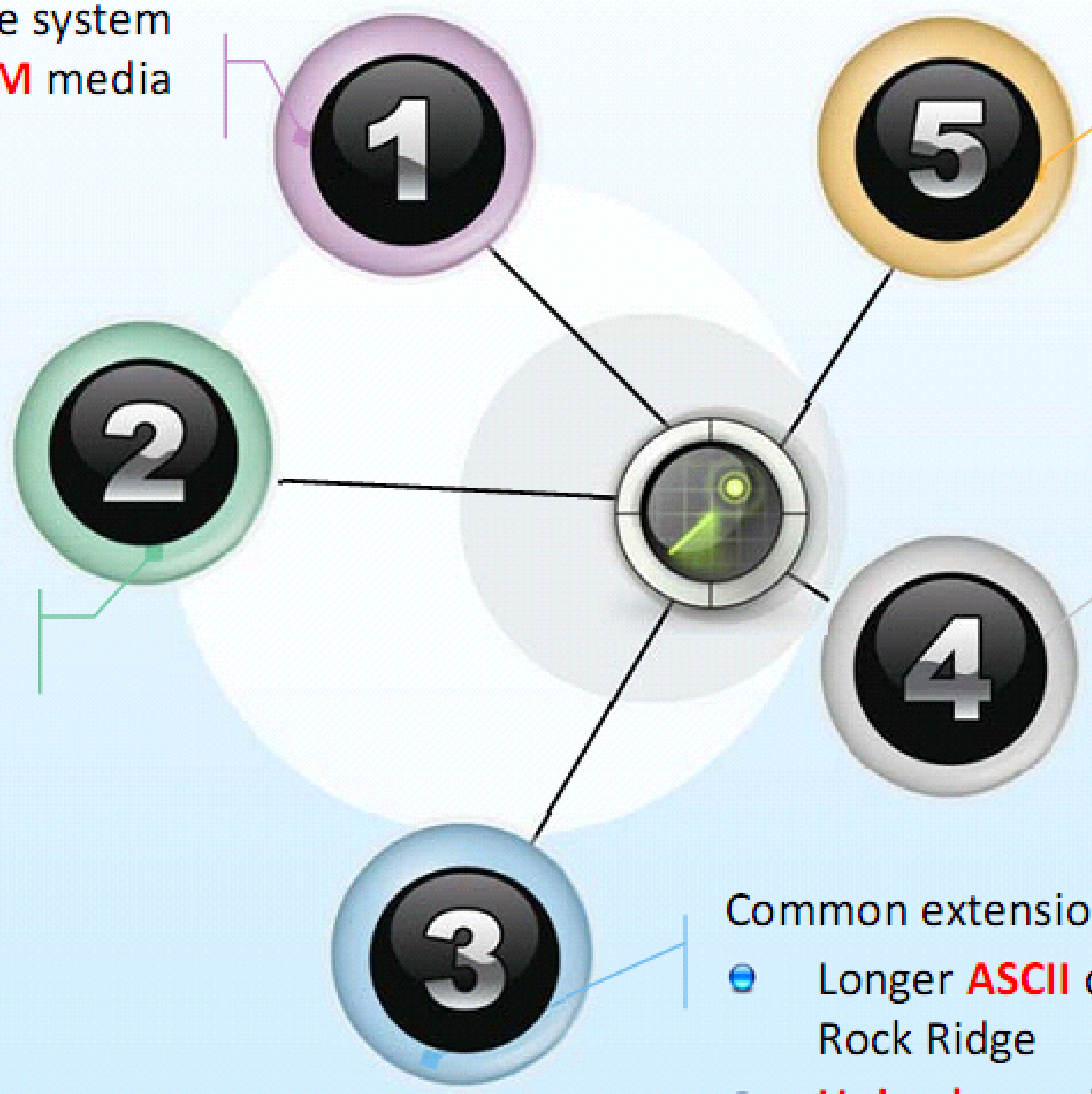
- ZFS Pooled Storage Model
- Data integrity Model
- Simplified Administration
- Copy-on-Write transactional model
- End-to-End Checksums
- Self-Healing Data
- Unparalled Scalability
- ZFS and Solid-State Storage
- Snapshots and Clones
- Encryption
- Deduplication
- Compression

Benefits:

- Simplifies and reduces storage management tasks
- Increases storage agility and data protection
- Delivers superior performance and availability

http://www.oracle.com

CD-ROM/DVD File System



The ISO (International Organization for Standardization) 9660 defines a file system for **CD-ROM** and **DVD-ROM** media

To **exchange data**, it supports various computer operating systems such as Microsoft Windows, Mac OS, and UNIX-based systems

Common extensions to ISO 9660 were to deal with the limitations:

- Longer **ASCII** coded names and UNIX permissions are facilitated by Rock Ridge
- **Unicode** naming (like non-Roman scripts)are also supported by Joliet
- Bootable CDs are facilitated by El Torito

ISO 13490 is a combination of ISO 9660 with multisession support

Windows supports two types of file systems on CD-ROM and Digital Versatile Disk (DVD):

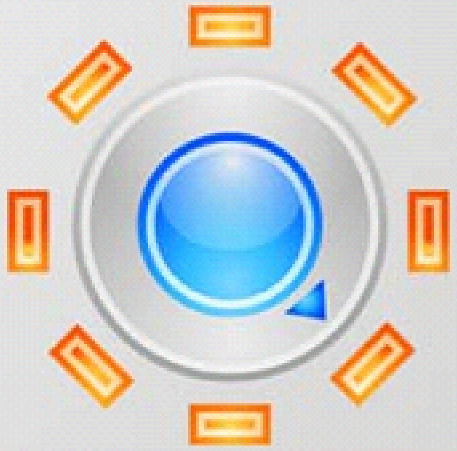
- **Compact Disc File System** (CDFS)
- **Universal Disk Format** (UDF)

Compact Disc File System (CDFS)

1. CD File System (CDFS) is a file system for **Linux operating system**

2. It transfers all **tracks** and **boot images** on a CD as normal files

3. It unlocks the information in old **ISO images**



For e.g., suppose multisession CD contains two ISO images, mounting the CD with CDFS file system, results in two sessions as files:

```
[root@k6 /root]# mount -t cdfs -o ro /dev/cdrom /mnt/cdfs
```

```
[root@k6 /root]# ls -l /mnt/cdfs
total 33389
-r--r--r-- 1 ronsse ronsse 33503232 Aug 8 19:36 sessions_1-1.iso
-r--r--r-- 1 ronsse ronsse 34121728 Aug 8 19:99 sessions_1-2.iso
```

Virtual File System (VFS) and Universal Disk Format File System (UDF)

Virtual File System (VFS)

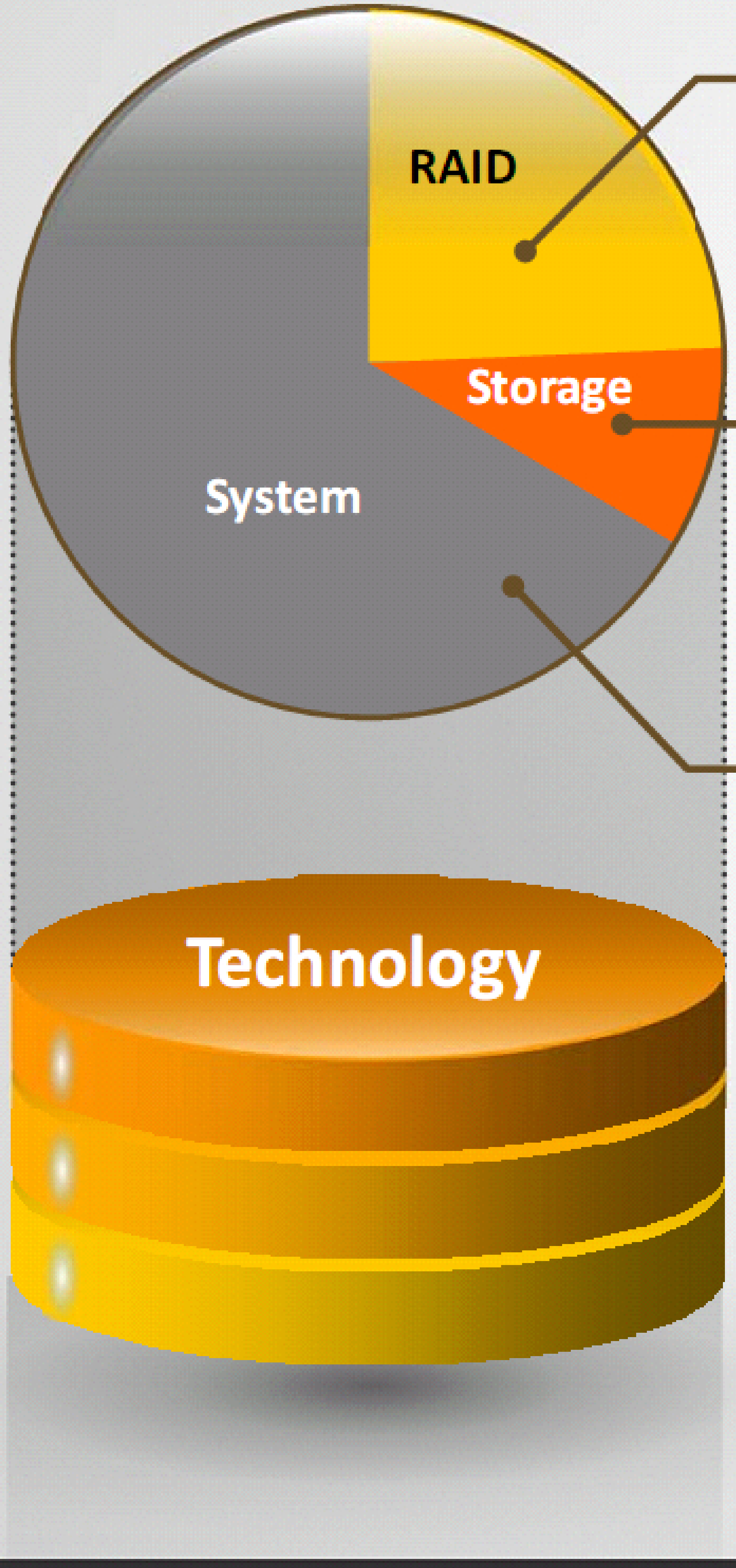
- A VFS is programming that specifies an interface between the OS's kernel and the different file systems
- VFS gives client applications access to the various concrete file system in a systematic manner

  E.g.: provision of transparent access to local and network storage devices without noticeable difference by the client application
- VMware Virtual Machine File System (VMFS), New Technology File System (NTFS), Global File System (GFS) and the Oracle Clustered File System (OCFS) are some of the VFS examples

Universal Disk Format File System (UDF)

- UDF is a file system specification defined by the Optical Storage Technology Association (OSTA), aimed to replace the ISO9660 file system on optical media and also FAT on removable media
- It is an open source file system based on ISO/IEC 13346 and ECMA-167 standards that defines how data is stored and interchanged on a wide variety of optical media

RAID Storage System



- Redundant Array of Independent Disks (RAID) is a technology that uses **multiple smaller disks** simultaneously which function as a **single large volume**

- It provides a particular method of accessing one or many separate hard disks, thereby **decreasing the risk of losing all data** in case of hard disk failures or damage, and improve access time

- This technology is developed to:
  - Maintain a **large** amount of **data storage**
  - Achieve a greater level of **input/output performance**
  - Achieve a greater reliability through **data redundancy**

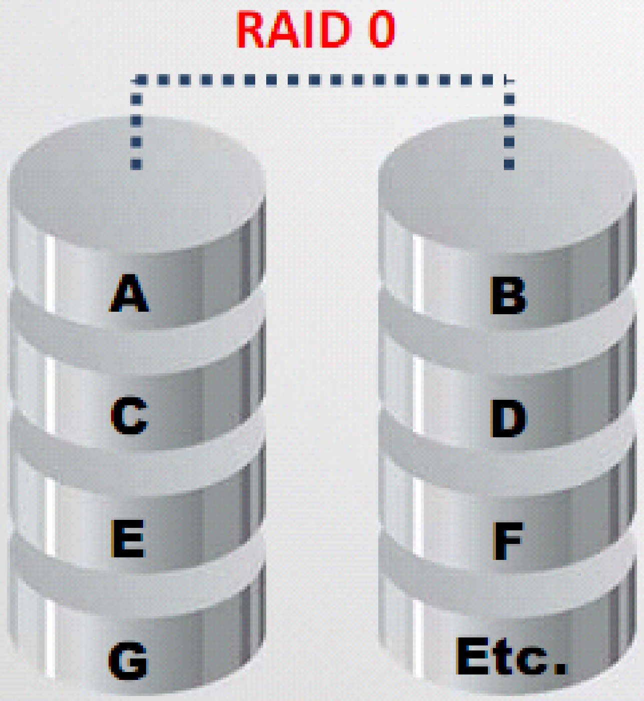
Levels of RAID Storage System

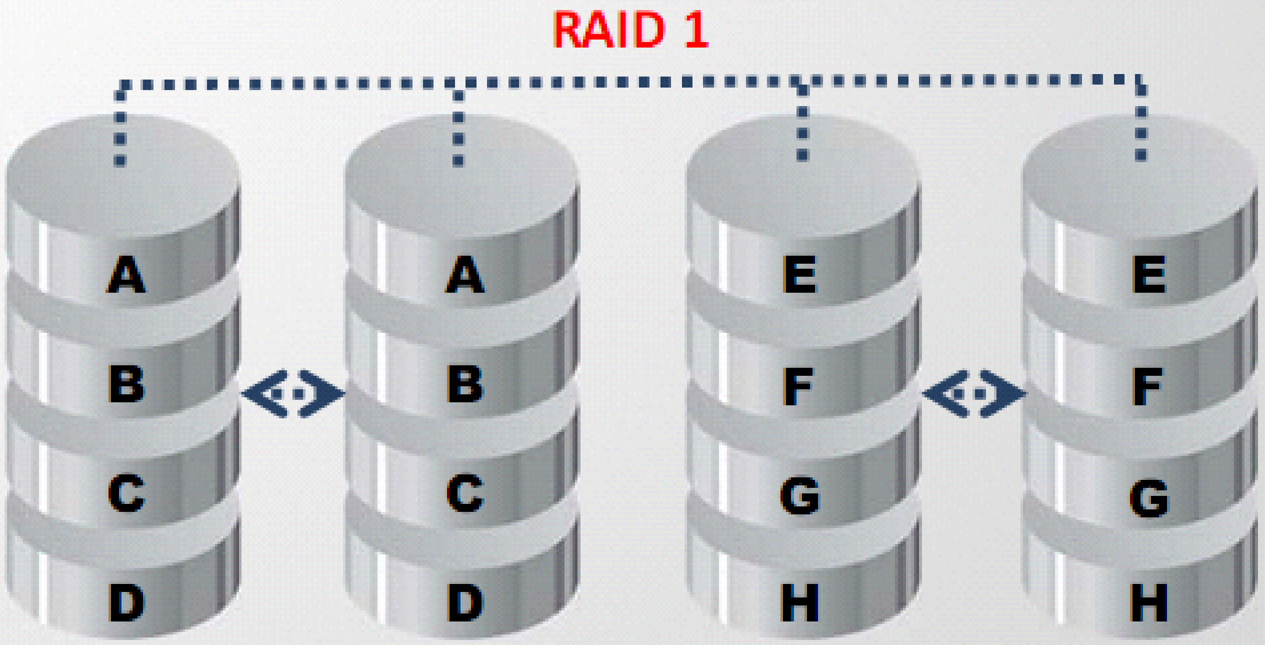
RAID 0

- Data is split into blocks and written equally across **multiple hard drives**
- It improves I/O performance by spreading the **I/O load** across many channels and disk drives
- If any **drive fails**, data recovery is not possible
- It does not provide **data redundancy**
- It requires **minimum two drives** for setting up

RAID 1

- It is made up of two disks for each volume and is designed for data recovery in the **event of disk failure**
- The contents of the two disks are **identical**
- It ensures that data is not lost and assists in preventing **computer downtime**





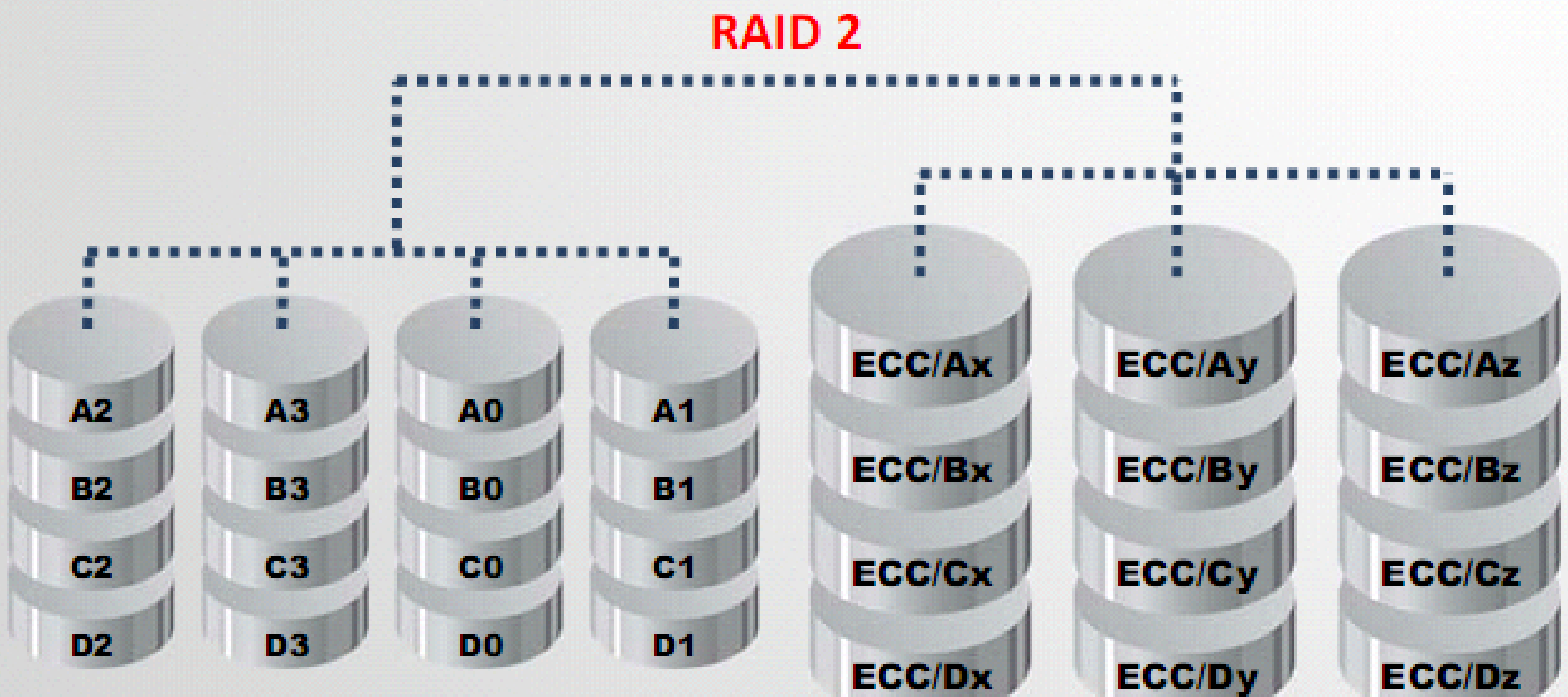
Levels of RAID Storage System (Cont'd)

RAID 2

- It provides **rapid access and increased storage** by configuring two or more disks as one large volume, similar to RAID 1
- Data is written to a disk on a bit level
- **Error correcting code (ECC)** is used to verify whether the write is successful
- It has better **data integrity** checking and is slower than RAID 0

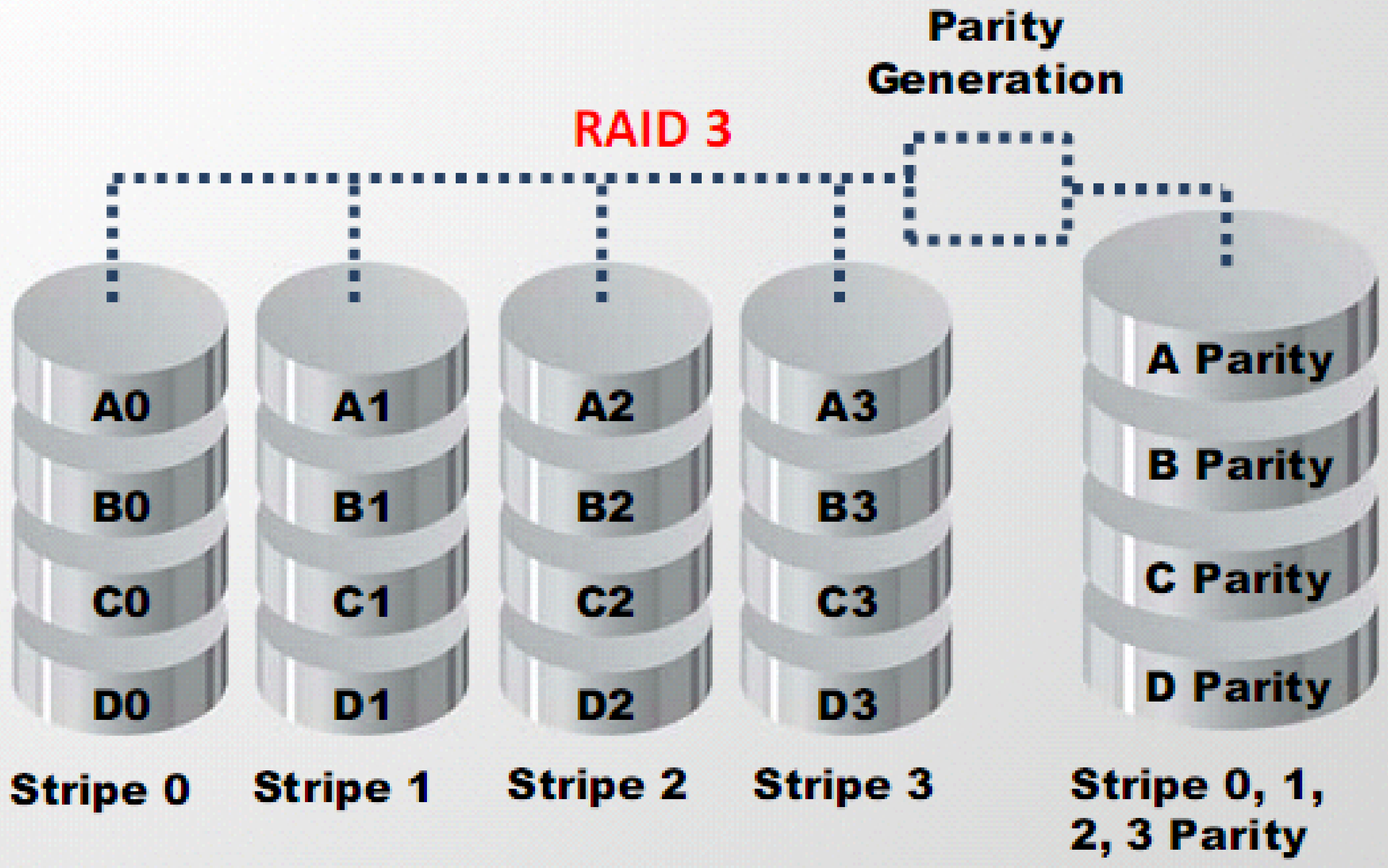
RAID 3

- It uses **data stripping and dedicated parity**, and requires at least three disks
- Data is striped at a **byte level** across multiple drives and one drive is set to store parity information
- If any drive fails, **data recovery and error correction** is possible through the parity drive



A0 to A3 = Word A; B0 to B3 =Word B;
C0 to C3 = Word C; D0 to D3 = Word D

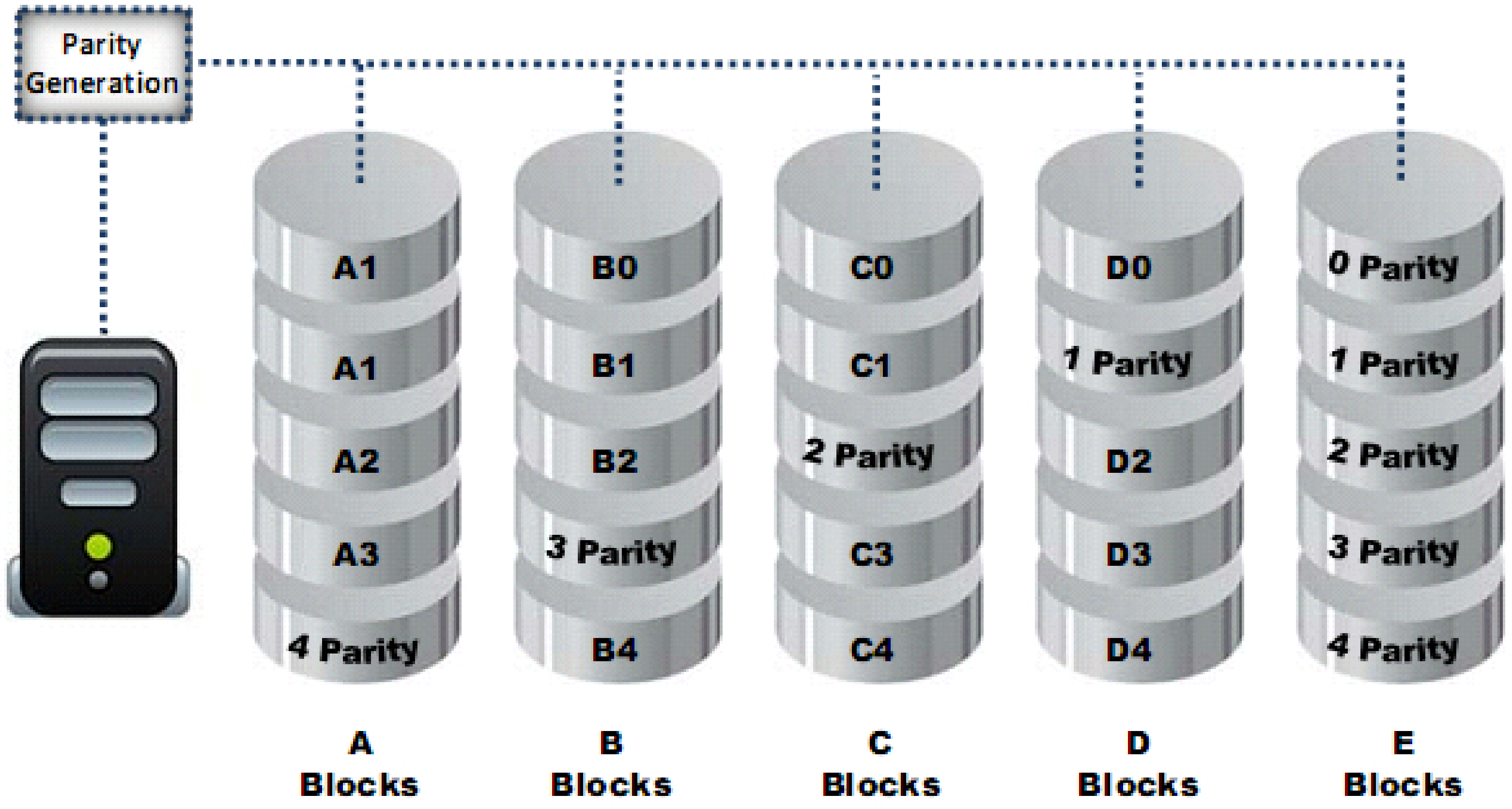
ECC/Ax to Az = Word A ECC;
ECC/Bx to Bz = Word B ECC;
ECC/Cx to Cz = Word C ECC;
ECC/Dx to Dz = Word D ECC;



Levels of RAID Storage System (Cont'd)

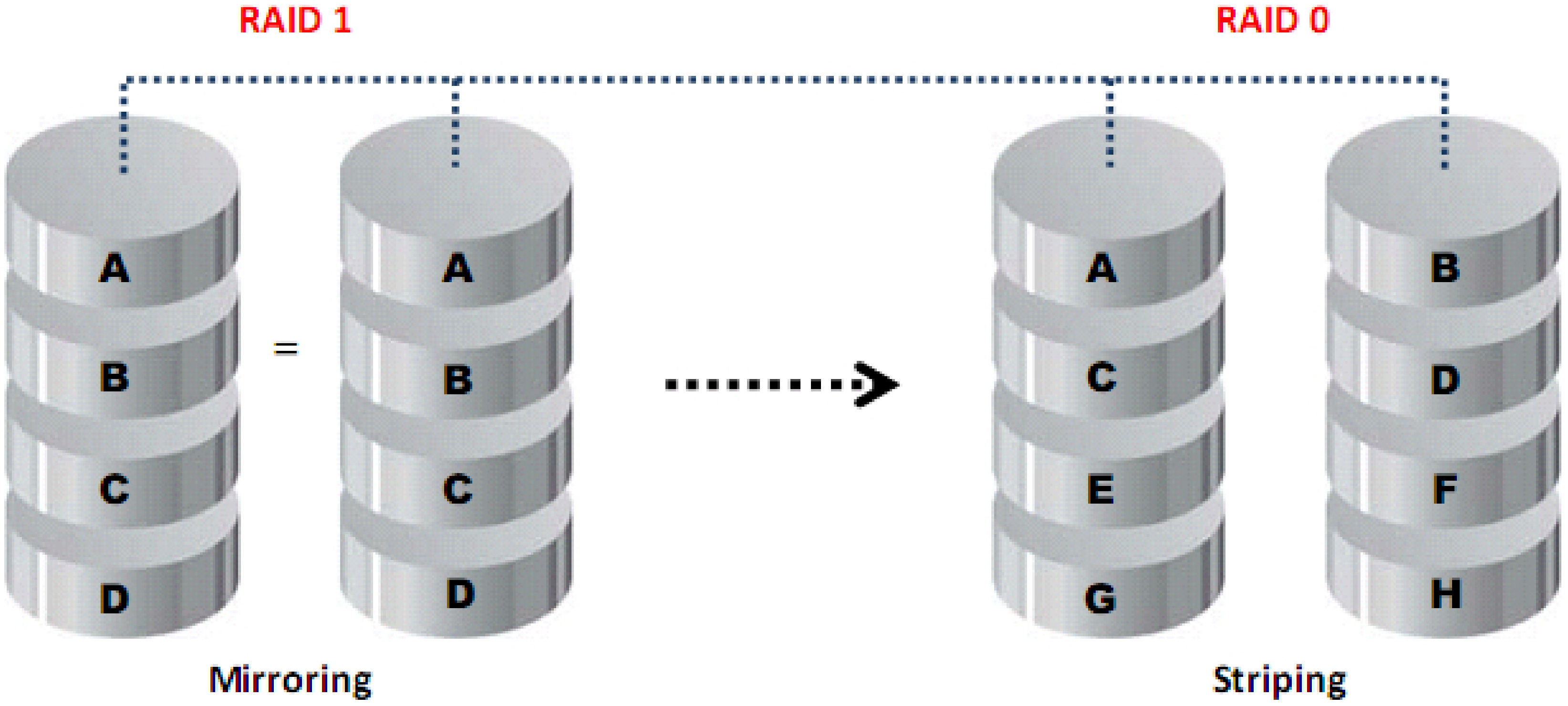
RAID 5

- Data is striped at a byte level across **multiple drives** and **parity information** is distributed among all member drives
- **Data writing** process is slow
- It requires a minimum of **three drives for setup**



RAID 10 or Mirrored Striping

- It is a combination of RAID 0 (Striping Volume Data) and RAID 1 (Disk Mirroring) and requires at least **four drives to implement**
- It has same **fault tolerance as RAID level 1** and the same overheads as mirroring alone
- It allows mirroring of disks in pairs for **redundancy** and **improved performance**, and then data is striped across multiple disks for maximum performance



Host Protected Areas (HPA) and Device Configuration Overlays (DCO)

- Host Protected Areas (HPA) and Device Configuration Overlays (DCO) are the hidden areas of a hard disk

  HPA:

  - HPA is the reserved area on a HDD, meant to store data in a way that the user, BIOS, or OS cannot modify, change, or access it
  - Information about HDD utilities, diagnostic tools, boot sector code, etc. is available in this area

  DCO:

  - DCO is an additional hidden area available on modern hard disks, which enables system vendors to buy HDDs of varying sizes from different manufacturers and configure all of them to have equal number of sectors
  - It can also be used to enable/disable features on the HDD

- With an intent to hide information, intruders use certain tools to modify and write to the HPA and DCO areas on the HDD
- HPA and DCO areas are of concern during the investigation as many tools fail to detect their presence
- Investigators can use tools such as EnCase, TAFT (an ATA (IDE) forensics tool), Sleuth Kit, etc. to detect and image HPA and/or DCO areas

File System Analysis

Understanding American Standard Code for Information Interchange (ASCII), Unicode, and Offset

- ASCII and Unicode are the two most common techniques used to encode the characters
- File contents can be viewed in ASCII or by extracting the ASCII strings from binary files for analysis

ASCII:

- It is a character encoding scheme developed from telegraphic codes
- ASCII encodes 128 specified characters into 7-bit integers. The encoded characters are:
  - Numbers 0 to 9
  - Lowercase letters a to z
  - Uppercase letters A to Z
  - Basic punctuation symbols
  - Control codes that originated with Teletype machines
  - A space
- The ASCII table has 3 divisions namely, non printable (system codes between 0 and 31), lower ASCII (codes between 32 and 127) and higher ASCII (codes between 128 and 255)

File System Analysis (Cont'd)

UNICODE:

- It is an international encoding standard which supports consistent encoding, representation, and management of text expressed in many writing systems
- It provides a unique number for every character, irrespective of the platform, program, and language
- UTF-8 and UTF-16 are the most widely used UTF character encodings

UTF Encoding	Description
UTF-8	a 8-bit, variable-width encoding. Maximizes compatibility with ASCII
UTF-16	a 16-bit, variable-width encoding
UTF-32	a 32-bit, fixed-width encoding

OFFSET:

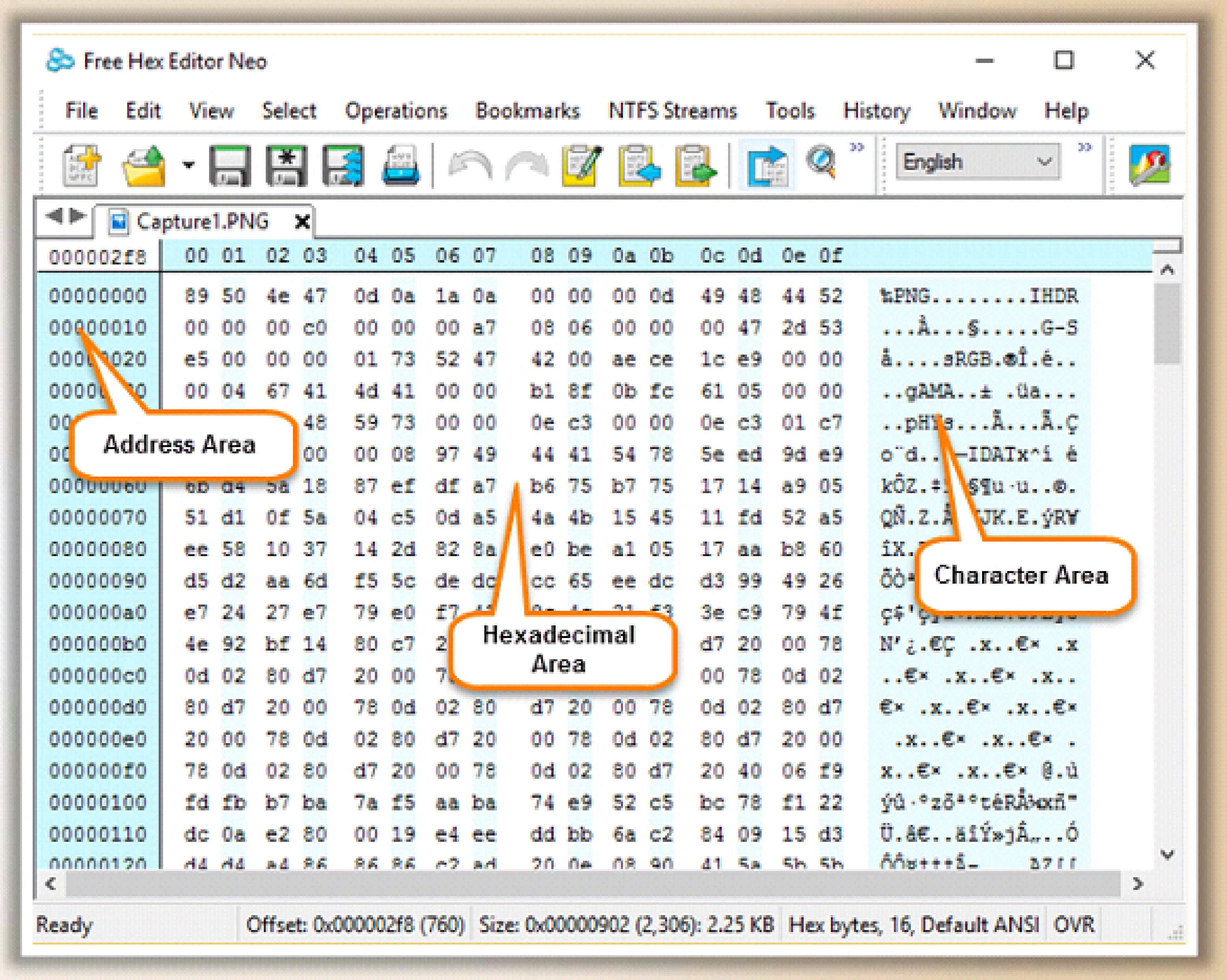
- In computing, an offset usually refers to either the start of a file or the start of a memory address
- Its value is added to a base address to derive the actual address

Example: If "A" denotes address 80, then the expression A+20 implies the address 100, where 20 in the expression is the offset

File System Analysis (Cont'd)

Understanding Hex Editor:

- Hex editors are the programs meant to examine or modify the physical (i.e. byte per byte) structure of a binary file
- Usually, hex-editors have three areas:
    - **Address area:** located on the left and exhibits the address of the first byte of each line usually in hexadecimal format
    - **Hexadecimal area:** located in the center, lists each byte of the file in a table, usually 16 bytes per line
    - **Character area:** located on the right, exhibits the ASCII representation of each of the bytes in the hexadecimal area
- In forensics, hex editors are of use to view stored or deleted data from both files and disk sectors
- In general, investigators use hex editors to examine evidence in particular parts of a disk
- Apart from the hexadecimal view of the data, many hex editors also display data both in binary and ASCII forms


Free Hex Editor Neo
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
Capture1.PNG
Address Area
Hexadecimal Area
Character Area

```
000002f8  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000  89 50 4e 47  0d 0a 1a 0a  00 00 00 0d  49 48 44 52   ‰PNG........IHDR
00000010  00 00 00 c0  00 00 00 a7  08 06 00 00  00 47 2d 53   ...À...§.....G-S
00000020  e5 00 00 00  01 73 52 47  42 00 ae ce  1c e9 00 00   å....sRGB.®Î.é..
00000030  00 04 67 41  4d 41 00 00  b1 8f 0b fc  61 05 00 00   ..gAMA..± .üa...
00                 48  59 73 00 00  0e c3 00 00  0e c3 01 c7   ..pH s...Ã...Ã.Ç
00                 00  00 08 97 49  44 41 54 78  5e ed 9d e9   o¨d.. IDATx^í é
00000060  6b d4 5a 18  87 ef df a7  b6 75 b7 75  17 14 a9 05   kÔZ.‡  §¶u·u..©.
00000070  51 d1 0f 5a  04 c5 0d a5  4a 4b 15 45  11 fd 52 a5   QÑ.Z.Å JK.E.ýR¥
00000080  ee 58 10 37  14 2d 82 8a  e0 be a1 05  17 aa b8 60   îX.
00000090  d5 d2 aa 6d  f5 5c de dc  cc 65 ee dc  d3 99 49 26   õÒ
000000a0  e7 24 27 e7  79 e0 f7                  3e c9 79 4f   ç$'
000000b0  4e 92 bf 14  80 c7 2                   d7 20 00 78   N'¿.€Ç .x..€× .x
000000c0  0d 02 80 d7  20 00 7                   00 78 0d 02   ..€× .x..€× .x..
000000d0  80 d7 20 00  78 0d 02 80  d7 20 00 78  0d 02 80 d7   €× .x..€× .x..€×
000000e0  20 00 78 0d  02 80 d7 20  00 78 0d 02  80 d7 20 00    .x..€× .x..€× .
000000f0  78 0d 02 80  d7 20 00 78  0d 02 80 d7  20 40 06 f9   x..€× .x..€× @.ù
00000100  fd fb b7 ba  7a f5 aa ba  74 e9 52 c5  bc 78 f1 22   ýû·ºzõªºtéRÅ¼xñ"
00000110  dc 0a e2 80  00 19 e4 ee  dd bb 6a c2  84 09 15 d3   Ü.â€..äîÝ»jÂ„..Ó
00000120  d4 d4 a4 86  86 86 c2 ad  20 0e 08 90  41 5a 5b 5b
```

Ready | Offset: 0x000002f8 (760) | Size: 0x00000902 (2,306): 2.25 KB | Hex bytes, 16, Default ANSI | OVR

http://www.hhdsoftware.com

File System Analysis (Cont'd)

Understanding Hexadecimal Notation:

- Hexadecimal numeral system, also known as hex, is a numeral system with base 16
- In hexadecimal notation 0–9 represent zero to nine values and English alphabets A, B, C, D, E, and F represent ten to fifteen values

  E.g.: 2BA in hexadecimal is the same as 0010 1011 1010 binary
- Hexadecimal notation allows using powers of 2 easily instead of writing the whole thing in binary

HEX	Binary	Base 10
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

File Carving

- It is a technique to **recover files and fragments of files** from unallocated space of the hard disk in the absence of file metadata
- In this technique, **file identification and extraction is based on certain characteristics** like file header or footer rather than the file extension or metadata

- A file header is a **signature** (also known as a magic number) which is a constant numeric or text value that determines a file format

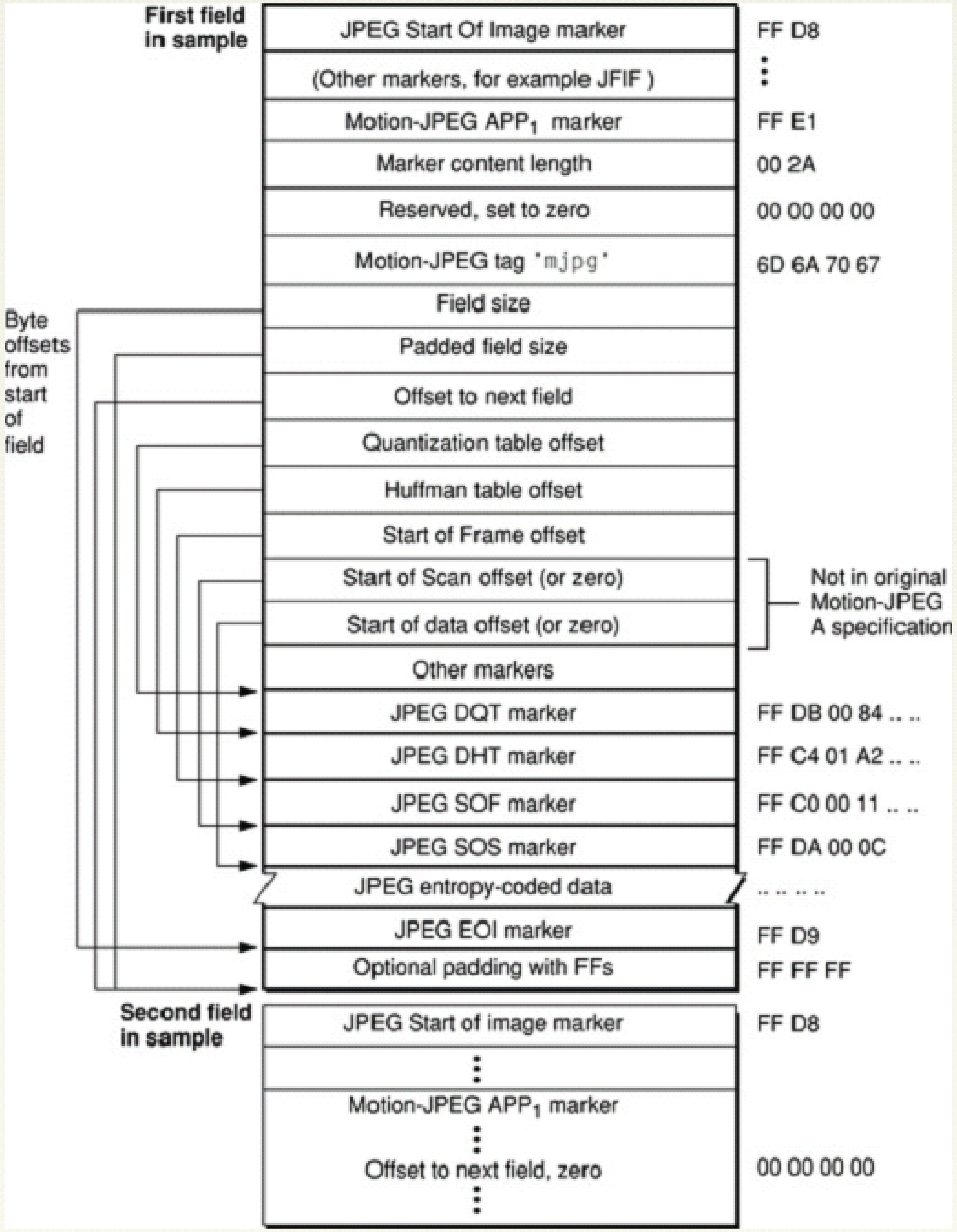
  Example: A suspect may try to hide an image from being detected by investigators by changing the file extension from .jpg to .dll

  However, changing the file extension does not change to the file header, and analysis tells the actual file format

  Example: A file format is confirmed as .jpg if it shows "JFIF" in the file header and hex signature as "4A 46 49 46"

- Investigators can take a look at file headers to verify the file format using tools such as 010 Editor, CI Hex Viewer, Hexinator, Hex Editor Neo, Qiew, WinHex, etc.

Image File Formats: JPEG



First field in sample

JPEG Start Of Image marker	FF D8
(Other markers, for example JFIF)	⋮
Motion-JPEG APP$_1$ marker	FF E1
Marker content length	00 2A
Reserved, set to zero	00 00 00 00
Motion-JPEG tag 'mjpg'	6D 6A 70 67
Field size	
Padded field size	
Offset to next field	
Quantization table offset	
Huffman table offset	
Start of Frame offset	
Start of Scan offset (or zero)	
Start of data offset (or zero)	
Other markers	
JPEG DQT marker	FF DB 00 84 .. ..
JPEG DHT marker	FF C4 01 A2 .. ..
JPEG SOF marker	FF C0 00 11 .. ..
JPEG SOS marker	FF DA 00 0C
JPEG entropy-coded data	.. ..
JPEG EOI marker	FF D9
Optional padding with FFs	FF FF FF

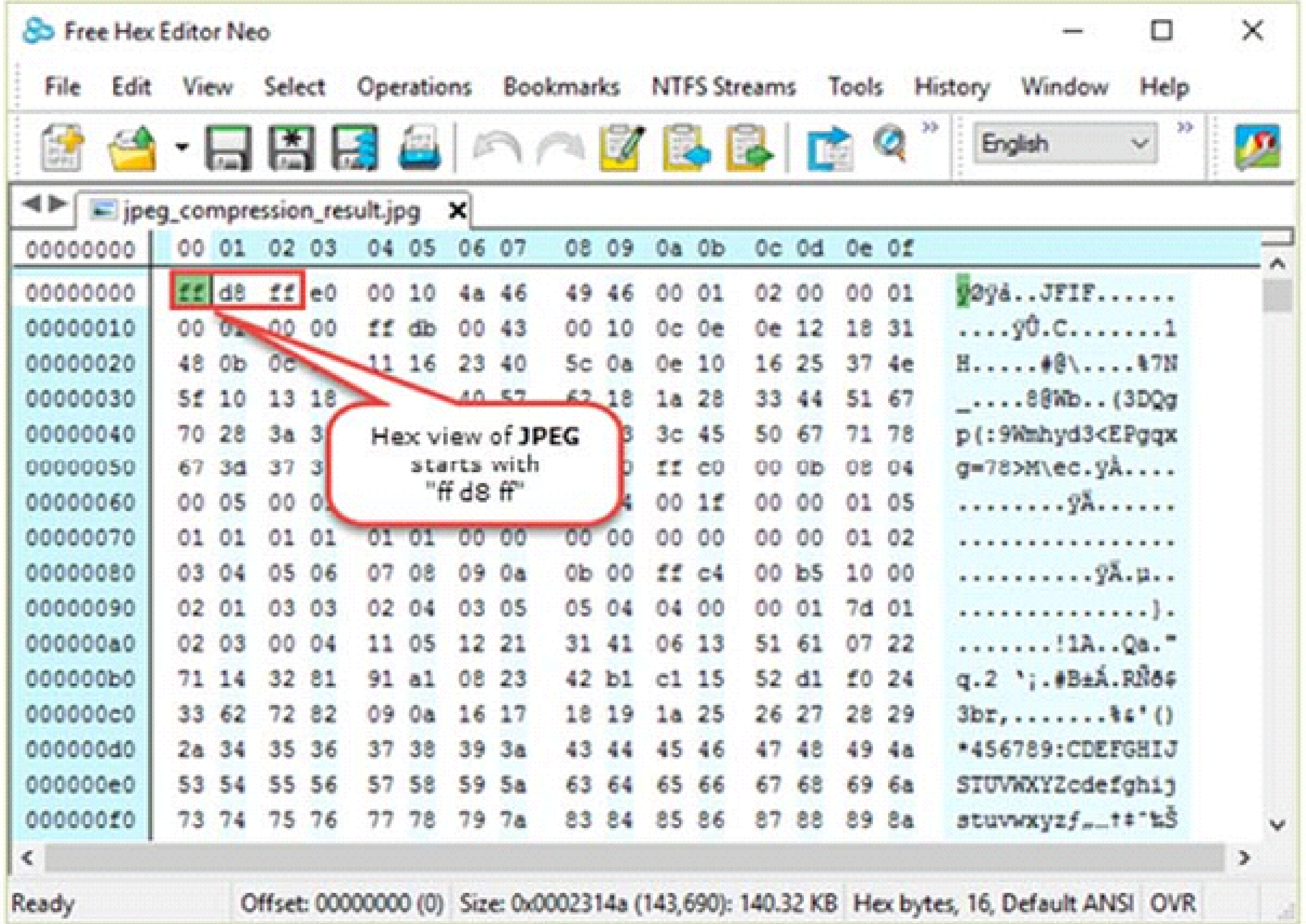
Second field in sample

JPEG Start of image marker	FF D8
⋮	
Motion-JPEG APP$_1$ marker ⋮ Offset to next field, zero ⋮	00 00 00 00

Byte offsets from start of field
Not in original Motion-JPEG A specification

https://developer.apple.com

- The JPEG is a commonly used method to compress photographic images
- It uses a compression algorithm to minimize the size of the file without affecting the quality of the image



Free Hex Editor Neo
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
jpeg_compression_result.jpg

```
00000000   00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000   ff d8 ff e0  00 10 4a 46  49 46 00 01  02 00 00 01   ÿØÿà..JFIF......
00000010   00 01 00 00  ff db 00 43  00 10 0c 0e  0e 12 18 31   ....ÿÛ.C.......1
00000020   48 0b 0c        11 16 23 40  5c 0a 0e 10  16 25 37 4e   H.....#@\....%7N
00000030   5f 10 13 18        40 57  62 18 1a 28  33 44 51 67   _....8@Wb..(3DQg
00000040   70 28 3a 3                 3c 45  50 67 71 78   p(:9Wmhyd3<EPgqx
00000050   67 3d 37 3                 ff c0  00 0b 08 04   g=78>M\ec.yÀ....
00000060   00 05 00 0                 00 1f  00 00 01 05   ........ÿÄ......
00000070   01 01 01 01  01 01 00 00  00 00 00 00  00 00 01 02   ................
00000080   03 04 05 06  07 08 09 0a  0b 00 ff c4  00 b5 10 00   ..........ÿÄ.µ..
00000090   02 01 03 03  02 04 03 05  05 04 04 00  00 01 7d 01   ..............}.
000000a0   02 03 00 04  11 05 12 21  31 41 06 13  51 61 07 22   .......!1A..Qa."
000000b0   71 14 32 81  91 a1 08 23  42 b1 c1 15  52 d1 f0 24   q.2 ';.#B±Á.RÑð$
000000c0   33 62 72 82  09 0a 16 17  18 19 1a 25  26 27 28 29   3br,.......%&'()
000000d0   2a 34 35 36  37 38 39 3a  43 44 45 46  47 48 49 4a   *456789:CDEFGHIJ
000000e0   53 54 55 56  57 58 59 5a  63 64 65 66  67 68 69 6a   STUVWXYZcdefghij
000000f0   73 74 75 76  77 78 79 7a  83 84 85 86  87 88 89 8a   stuvwxyzƒ„…†‡ˆ‰Š
```

Ready | Offset: 00000000 (0) | Size: 0x0002314a (143,690): 140.32 KB | Hex bytes, 16, Default ANSI | OVR

Hex view of **JPEG** starts with "ff d8 ff"

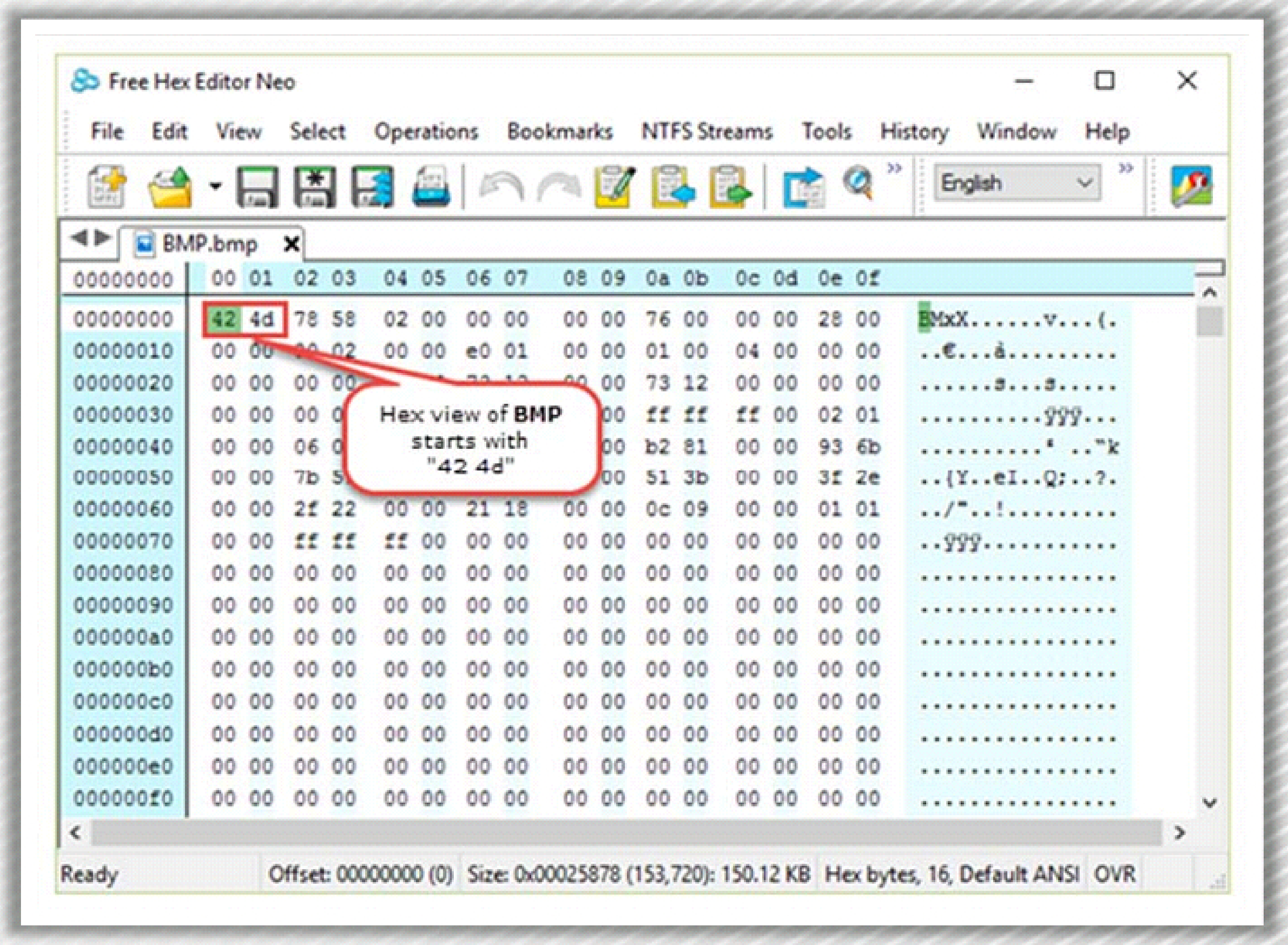
Hex View of JPEG File Format

Image File Formats: **BMP**

- The Bitmap (BMP) is a standard file format for a Windows Device Independent Bitmap (DIB) file
- The size and color of these images can vary from 1 bit per pixel (black and white) to 24-bit color (16.7 million colors)

Basic BMP File Format

Name			Size
Header			**14 bytes**
	Signature		2 bytes
	File Size		4 bytes
	reserved		4 bytes
	DataOffset		4 bytes
Color Table			**4 * NumColors bytes**
		Red	1 byte
		Green	1 byte
		Blue	1 byte
		reserved	1 byte
	Repeated NumColors times		
Raster Data			**Info ImageSize bytes**



```
00000000  00 01  02 03  04 05  06 07   08 09  0a 0b  0c 0d  0e 0f
00000000  42 4d  78 58  02 00  00 00   00 00  76 00  00 00  28 00   BMxX......v...(.
00000010  00 00  [illegible] 02  00 00  e0 01   00 00  01 00  04 00  00 00   ..€...à.........
00000020  00 00  00 00  [illegible]   00 00  73 12  00 00  00 00   ......ƨ...ƨ.....
00000030  00 00  00 0[illegible]   00  ff ff  ff 00  02 01   .........ÿÿÿ...
00000040  00 00  06 0[illegible]   00  b2 81  00 00  93 6b   .........ʻ .."k
00000050  00 00  7b 5[illegible]   00  51 3b  00 00  3f 2e   ..{Y..eI..Q;..?.
00000060  00 00  2f 22  00 00  21 18   00 00  0c 09  00 00  01 01   ../"..!.........
00000070  00 00  ff ff  ff 00  00 00   00 00  00 00  00 00  00 00   ..ÿÿÿ...........
00000080  00 00  00 00  00 00  00 00   00 00  00 00  00 00  00 00   ................
00000090  00 00  00 00  00 00  00 00   00 00  00 00  00 00  00 00   ................
000000a0  00 00  00 00  00 00  00 00   00 00  00 00  00 00  00 00   ................
000000b0  00 00  00 00  00 00  00 00   00 00  00 00  00 00  00 00   ................
000000c0  00 00  00 00  00 00  00 00   00 00  00 00  00 00  00 00   ................
000000d0  00 00  00 00  00 00  00 00   00 00  00 00  00 00  00 00   ................
000000e0  00 00  00 00  00 00  00 00   00 00  00 00  00 00  00 00   ................
000000f0  00 00  00 00  00 00  00 00   00 00  00 00  00 00  00 00   ................
```

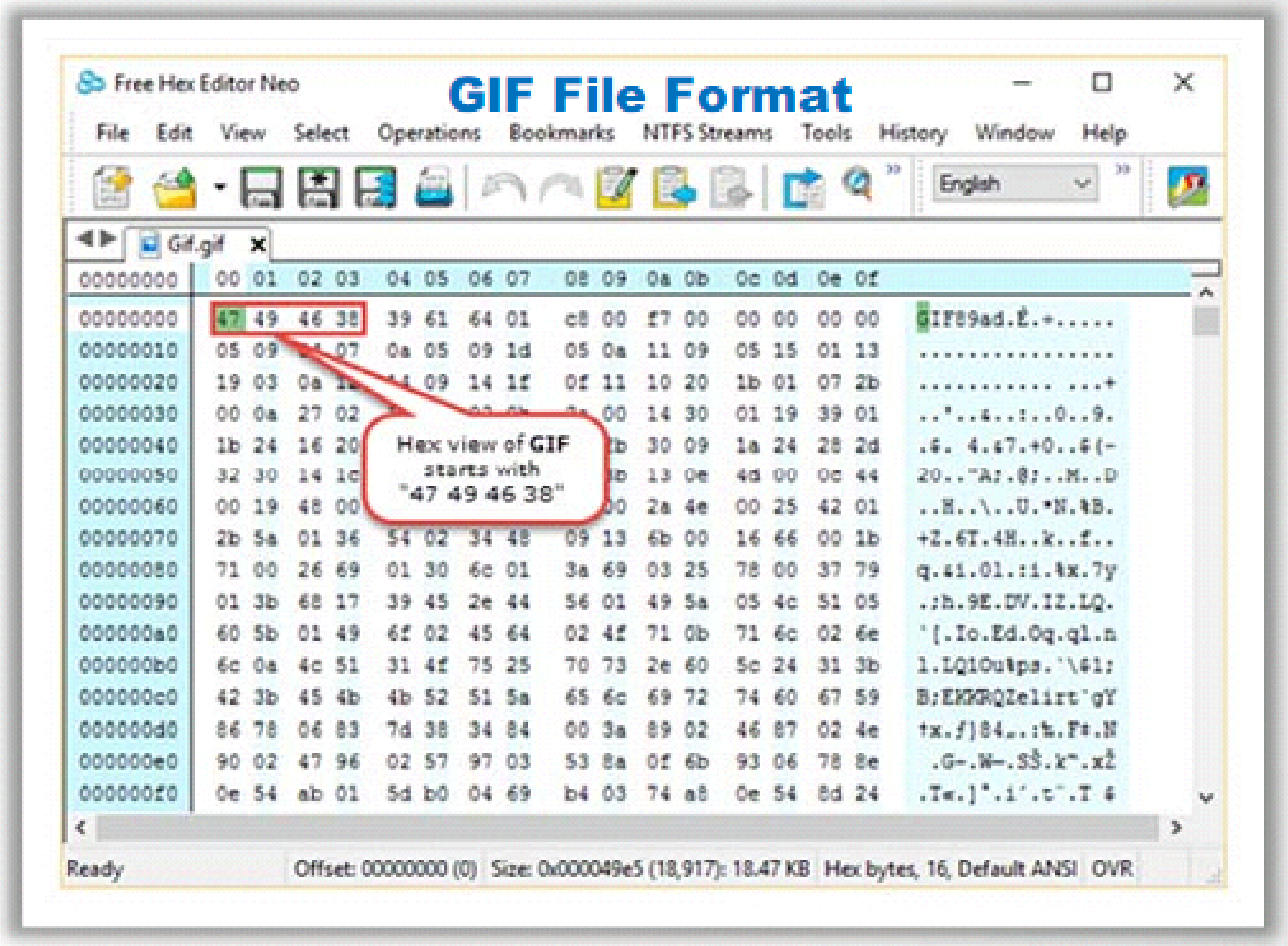
Hex view of **BMP** starts with "42 4d"

```
Ready | Offset: 00000000 (0) | Size: 0x00025878 (153,720): 150.12 KB | Hex bytes, 16, Default ANSI | OVR
```

Hex View of BMP File Format

Hex View of Popular Image File Formats

- The GIF is a file format that contains **8 bits per pixel** and displays 256 colors per frame
- It **uses lossless data compression techniques**, which maintain the visual quality of the image

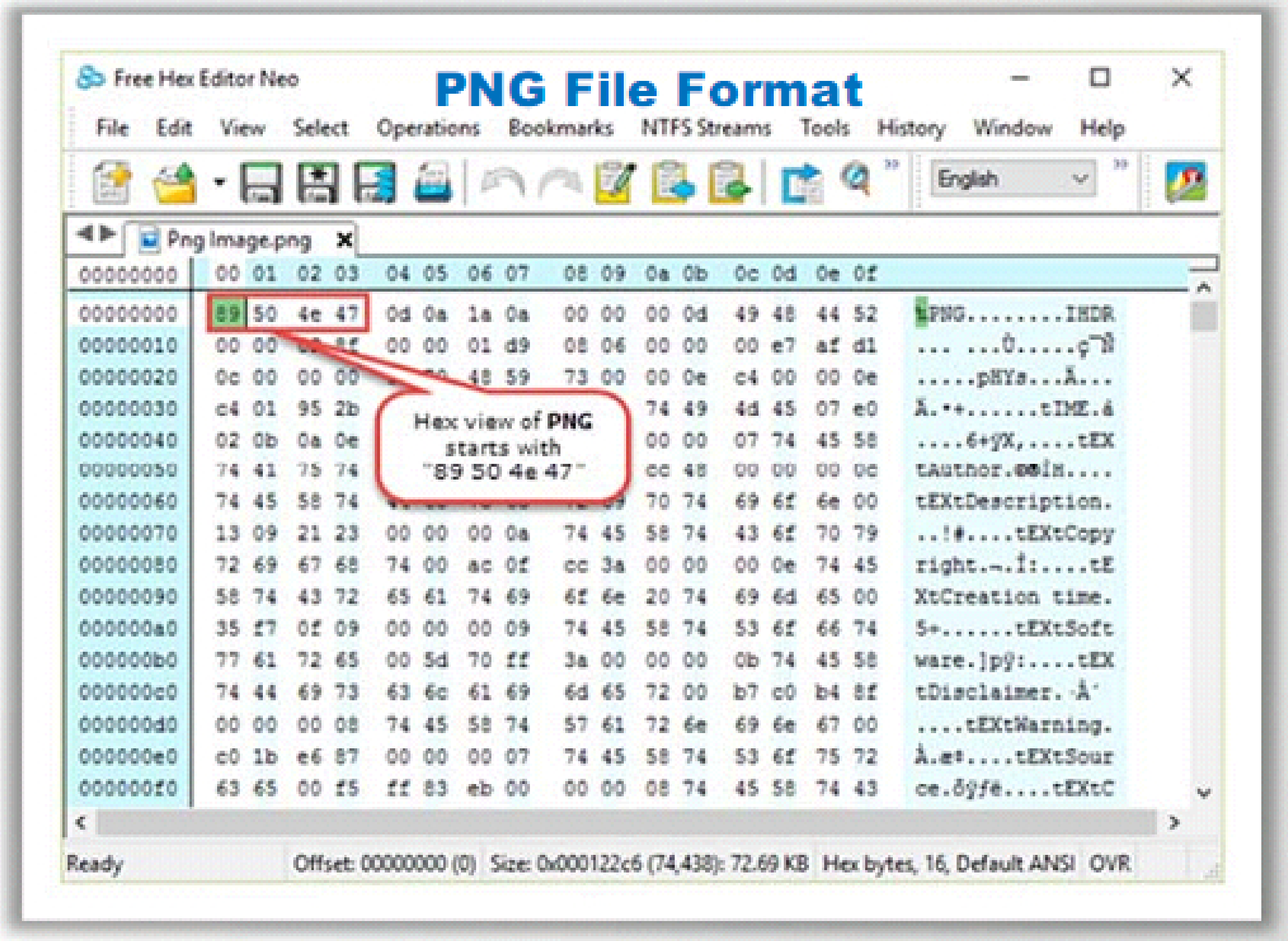


Free Hex Editor Neo
GIF File Format
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
Gif.gif
Hex view of GIF starts with "47 49 46 38"

```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000  47 49 46 38  39 61 64 01  c8 00 f7 00  00 00 00 00   GIF89ad.È.÷.....
00000010  05 09    07  0a 05 09 1d  05 0a 11 09  05 15 01 13   ................
00000020  19 03 0a        09 14 1f  0f 11 10 20  1b 01 07 2b   ........... ...+
00000030  00 0a 27 02                  00 14 30  01 19 39 01   ..'...4..:..0..9.
00000040  1b 24 16 20                     30 09  1a 24 28 2d   .6. 4.&7.+0..$(-
00000050  32 30 14 1c                     13 0e  4d 00 0c 44   20..˜A:.@:..M..D
00000060  00 19 48 00                  00 2a 4e  00 25 42 01   ..H..\..U.*N.%B.
00000070  2b 5a 01 36  54 02 34 48  09 13 6b 00  16 66 00 1b   +Z.6T.4H..k..f..
00000080  71 00 26 69  01 30 6c 01  3a 69 03 25  78 00 37 79   q.&i.0l.:i.%x.7y
00000090  01 3b 68 17  39 45 2e 44  56 01 49 5a  05 4c 51 05   .;h.9E.DV.IZ.LQ.
000000a0  60 5b 01 49  6f 02 45 64  02 4f 71 0b  71 6c 02 6e   `[.Io.Ed.Oq.ql.n
000000b0  6c 0a 4c 51  31 4f 75 25  70 73 2e 60  5c 24 31 3b   l.LQ1Ou%ps.`\$1;
000000c0  42 3b 45 4b  4b 52 51 5a  65 6c 69 72  74 60 67 59   B;EKKRQZelirt`gY
000000d0  86 78 06 83  7d 38 34 84  00 3a 89 02  46 87 02 4e   †x.ƒ}84„.:‰.F‡.N
000000e0  90 02 47 96  02 57 97 03  53 8a 0f 6b  93 06 78 8e    .G–.W—.SŠ.k".xŽ
000000f0  0e 54 ab 01  5d b0 04 69  b4 03 74 a8  0e 54 8d 24   .T«.]°.i´.t¨.T $
```

Ready | Offset: 0000000 (0) | Size: 0x000049e5 (18,917): 18.47 KB | Hex bytes, 16, Default ANSI | OVR

- The **PNG** is a lossless data compression image format, intended to replace the GIF and TIFF formats
- It supports:
  - **Indexed / Palette-based images (24-bit RGB or 32-bit RGBA colors)**
  - **Grayscale images (with or without alpha channel)**
  - **Transparency (both normal and alpha channel)**



Free Hex Editor Neo
PNG File Format
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
Png Image.png
Hex view of PNG starts with "89 50 4e 47"

```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000  89 50 4e 47  0d 0a 1a 0a  00 00 00 0d  49 48 44 52   ‰PNG........IHDR
00000010  00 00        00 00 01 d9  08 06 00 00  00 e7 af d1   ... ...Ù.....ç¯Ñ
00000020  0c 00 00 00        48 59  73 00 00 0e  c4 00 00 0e   .....pHYs...Ä...
00000030  c4 01 95 2b                     74 49  4d 45 07 e0   Ä.•+......tIME.à
00000040  02 0b 0a 0e                     00 00  07 74 45 58   ....6+ÿX,....tEX
00000050  74 41 75 74                     cc 48  00 00 00 0c   tAuthor.ÌHÍH....
00000060  74 45 58 74                     70 74  69 6f 6e 00   tEXtDescription.
00000070  13 09 21 23  00 00 00 0a  74 45 58 74  43 6f 70 79   ..!#....tEXtCopy
00000080  72 69 67 68  74 00 ac 0f  cc 3a 00 00  00 0e 74 45   right.¬.Ì:....tE
00000090  58 74 43 72  65 61 74 69  6f 6e 20 74  69 6d 65 00   XtCreation time.
000000a0  35 f7 0f 09  00 00 00 09  74 45 58 74  53 6f 66 74   5÷......tEXtSoft
000000b0  77 61 72 65  00 5d 70 ff  3a 00 00 00  0b 74 45 58   ware.]pÿ:....tEX
000000c0  74 44 69 73  63 6c 61 69  6d 65 72 00  b7 c0 b4 8f   tDisclaimer.·À´
000000d0  00 00 00 08  74 45 58 74  57 61 72 6e  69 6e 67 00   ....tEXtWarning.
000000e0  c0 1b e6 87  00 00 00 07  74 45 58 74  53 6f 75 72   À.æ‡....tEXtSour
000000f0  63 65 00 f5  ff 83 eb 00  00 00 08 74  45 58 74 43   ce.õÿƒë....tEXtC
```

Ready | Offset: 0000000 (0) | Size: 0x000122c6 (74,438): 72.69 KB | Hex bytes, 16, Default ANSI | OVR

PDF File Format

- Attackers use PDF and Microsoft Office (Word, PowerPoint, and Excel) documents as attack vectors because of their wide usage by individuals and organizations
- Thus, it is essential for an investigator to understand the PDF and Microsoft Office file formats and structures, which may assist during malicious document analysis

PDF File Structure:

Header:

The first line of the PDF file specifies the version number of a PDF file format

Ex: %PDF-1.3

Body:

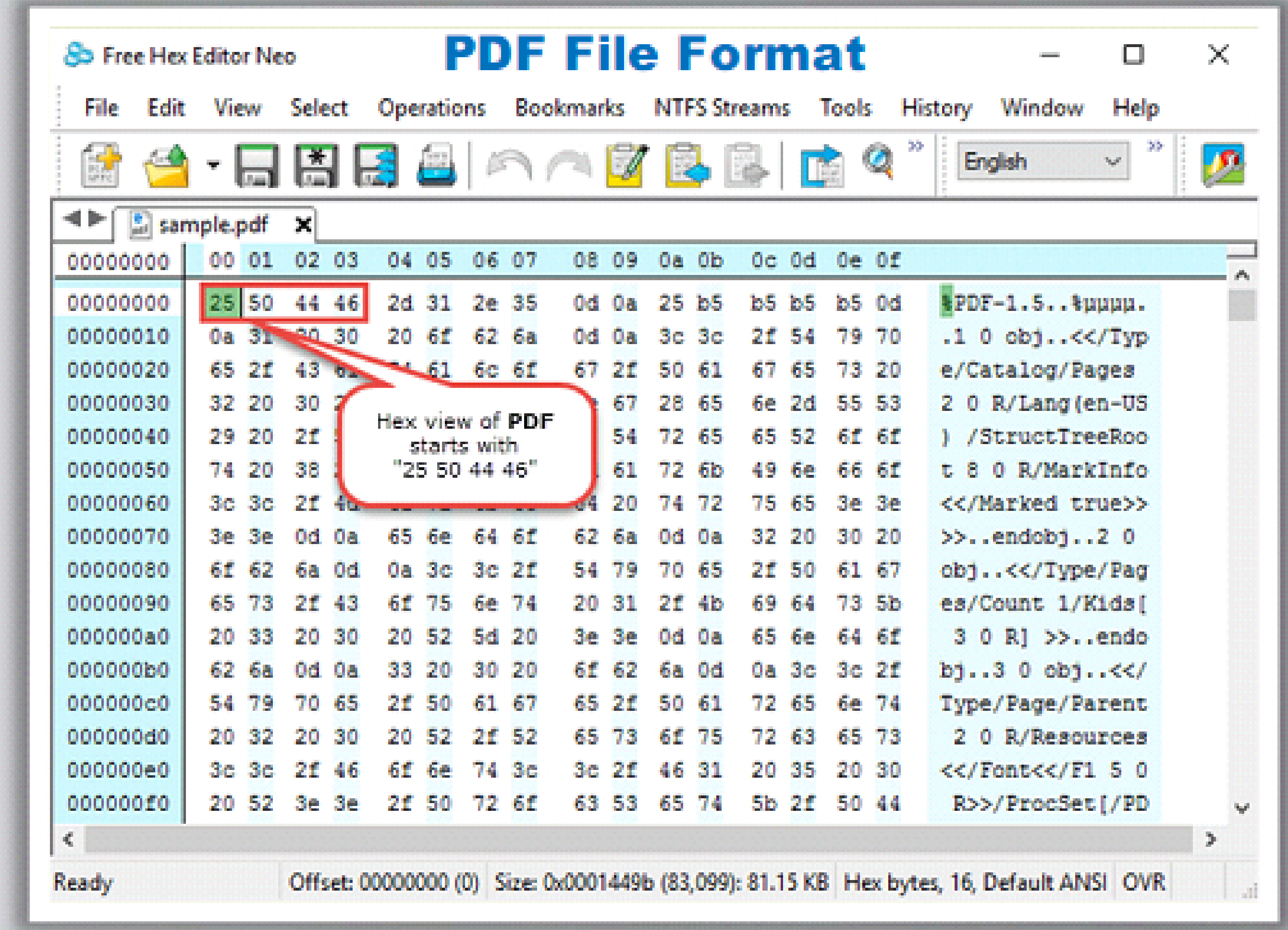
Consists of objects (images, fonts, form fields, text streams, annotations, bookmarks, etc.) that constitute the contents of the document

The Cross-reference table (xref table):

- Contains links to all the objects in a file
- Allows random access to objects
- Allows to trace updated changes made to the PDF file

The Trailer:

- Contains links to the xref table and to main objects in the trailer dictionary
- Ends with %%EOF to recognize end of file



PDF File Format

```
00000000   00 01  02 03   04 05  06 07   08 09  0a 0b   0c 0d  0e 0f

00000000   25 50  44 46   2d 31  2e 35   0d 0a  25 b5   b5 b5  b5 0d   %PDF-1.5..%µµµµ.
00000010   0a 31  20 30   20 6f  62 6a   0d 0a  3c 3c   2f 54  79 70   .1 0 obj..<</Typ
00000020   65 2f  43 6       61  6c 6f   67 2f  50 61   67 65  73 20   e/Catalog/Pages
00000030   32 20  30 2                      67  28 65   6e 2d  55 53   2 0 R/Lang(en-US
00000040   29 20  2f                        54  72 65   65 52  6f 6f   ) /StructTreeRoo
00000050   74 20  38                        61  72 6b   49 6e  66 6f   t 8 0 R/MarkInfo
00000060   3c 3c  2f 4                   4  20  74 72   75 65  3e 3e   <</Marked true>>
00000070   3e 3e  0d 0a   65 6e  64 6f   62 6a  0d 0a   32 20  30 20   >>..endobj..2 0
00000080   6f 62  6a 0d   0a 3c  3c 2f   54 79  70 65   2f 50  61 67   obj..<</Type/Pag
00000090   65 73  2f 43   6f 75  6e 74   20 31  2f 4b   69 64  73 5b   es/Count 1/Kids[
000000a0   20 33  20 30   20 52  5d 20   3e 3e  0d 0a   65 6e  64 6f    3 0 R] >>..endo
000000b0   62 6a  0d 0a   33 20  30 20   6f 62  6a 0d   0a 3c  3c 2f   bj..3 0 obj..<</
000000c0   54 79  70 65   2f 50  61 67   65 2f  50 61   72 65  6e 74   Type/Page/Parent
000000d0   20 32  20 30   20 52  2f 52   65 73  6f 75   72 63  65 73    2 0 R/Resources
000000e0   3c 3c  2f 46   6f 6e  74 3c   3c 2f  46 31   20 35  20 30   <</Font<</F1 5 0
000000f0   20 52  3e 3e   2f 50  72 6f   63 53  65 74   5b 2f  50 44    R>>/ProcSet[/PD
```

Ready | Offset: 00000000 (0) | Size: 0x0001449b (83,099): 81.15 KB | Hex bytes, 16, Default ANSI | OVR

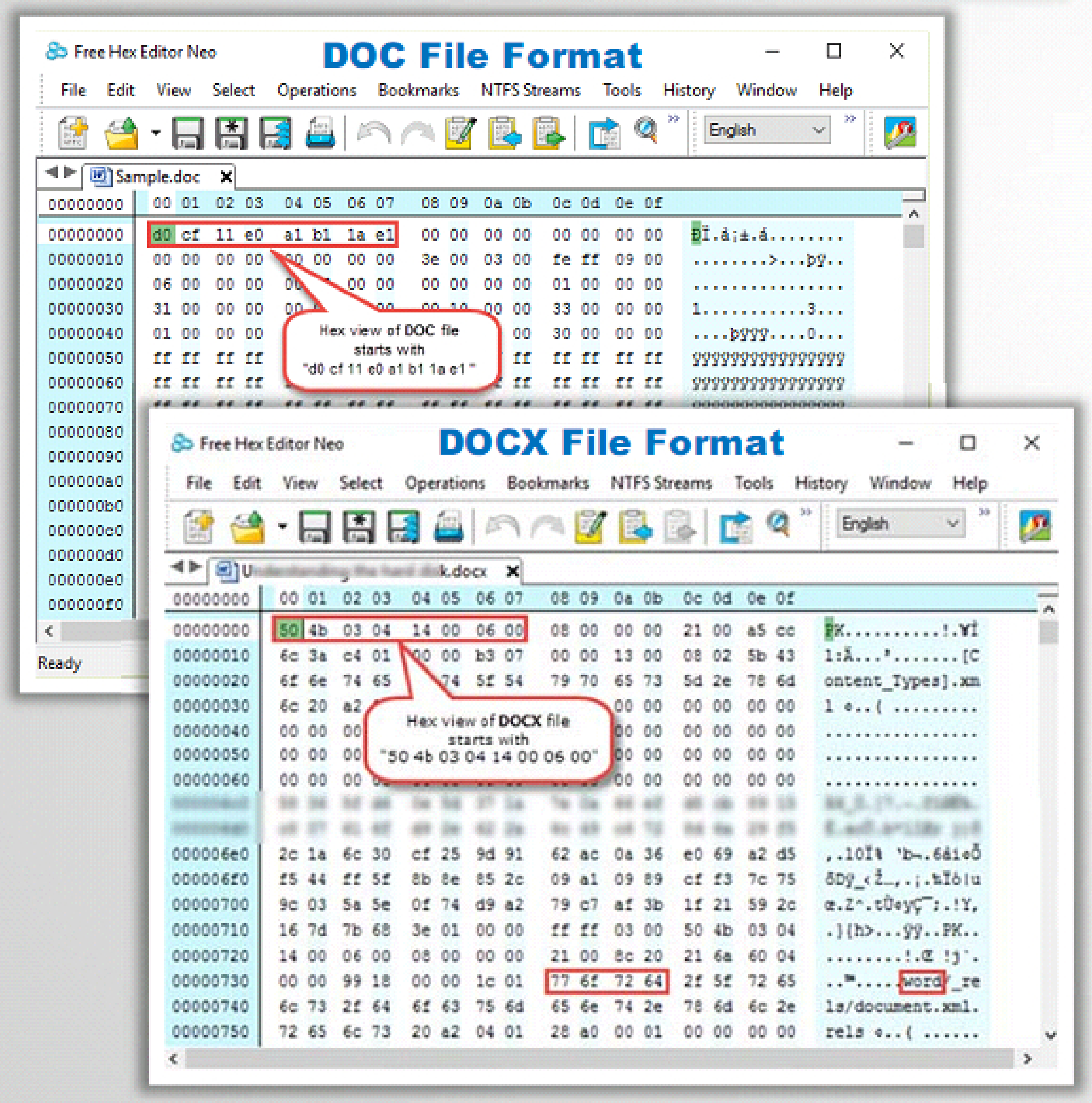
Hex view of **PDF** starts with "25 50 44 46"

Word File Formats

MS Office Documents – File Format:

Binary File Format

- Usually, attackers target MS Office documents of the binary format because they are still in use and the file structures are highly complex



Free Hex Editor Neo
DOC File Format
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
Sample.doc

```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f

00000000  d0 cf 11 e0  a1 b1 1a e1  00 00 00 00  00 00 00 00  ÐÏ.à¡±.á........
00000010  00 00 00 00  00 00 00 00  3e 00 03 00  fe ff 09 00  ........>...þÿ..
00000020  06 00 00 00  00           00 00 00 00  01 00 00 00  ................
00000030  31 00 00 00  00                 00 00  33 00 00 00  1...........3...
00000040  01 00 00 00                        00  30 00 00 00  ....þÿÿÿ....0...
00000050  ff ff ff ff                        ff  ff ff ff ff  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000060  ff ff ff ff                        ff  ff ff ff ff  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000070
00000080
00000090
000000a0
000000b0
000000c0
000000d0
000000e0
000000f0
```

Hex view of DOC file starts with "d0 cf 11 e0 a1 b1 1a e1 "

Free Hex Editor Neo
DOCX File Format
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
U[redacted]k.docx

```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f

00000000  50 4b 03 04  14 00 06 00  08 00 00 00  21 00 a5 cc  PK..........!.¥Ì
00000010  6c 3a c4 01  00 00 b3 07  00 00 13 00  08 02 5b 43  l:Ä...³.......[C
00000020  6f 6e 74 65     74 5f 54  79 70 65 73  5d 2e 78 6d  ontent_Types].xm
00000030  6c 20 a2                        00 00  00 00 00 00  l ¢..( .........
00000040  00 00 00                        00 00  00 00 00 00  ................
00000050  00 00 00                        00 00  00 00 00 00  ................
00000060  00 00 00                        00 00  00 00 00 00  ................
[redacted]
000006e0  2c 1a 6c 30  cf 25 9d 91  62 ac 0a 36  e0 69 a2 d5  ,.l0Ï% 'b¬.6ài¢Õ
000006f0  f5 44 ff 5f  8b 8e 85 2c  09 a1 09 89  cf f3 7c 75  õDÿ_‹Ž…,.¡.‰Ïó|u
00000700  9c 03 5a 5e  0f 74 d9 a2  79 c7 af 3b  1f 21 59 2c  œ.Z^.tÙ¢yÇ¯;.!Y,
00000710  16 7d 7b 68  3e 01 00 00  ff ff 03 00  50 4b 03 04  .}{h>...ÿÿ..PK..
00000720  14 00 06 00  08 00 00 00  21 00 8c 20  21 6a 60 04  ........!.Œ !j`.
00000730  00 00 99 18  00 00 1c 01  77 6f 72 64  2f 5f 72 65  ..™.....word/_re
00000740  6c 73 2f 64  6f 63 75 6d  65 6e 74 2e  78 6d 6c 2e  ls/document.xml.
00000750  72 65 6c 73  20 a2 04 01  28 a0 00 01  00 00 00 00  rels ¢..( ......
```

Hex view of **DOCX** file starts with "50 4b 03 04 14 00 06 00"

Microsoft Word File Structure (.doc/.docx):

WordDocument Stream/Main Stream

- Contains binary data of the Word document and Word file header (known as the File Information Block (FIB)) located at the offset 0
- FIB contains information about the document, file length, and specifies pointers to elements in the document file

Summary Information Streams

- The summary information is stored in two storage streams: Summary Information and DocumentSummaryInformation

Table Stream (0Table or 1Table)

- Contains data referenced from the FIB and other parts of the file
- stores various plex of character positions (PLCs) and tables defining document's structure
- Has predefined structure only for encrypted files

Data Stream (Optional)

- No predefined structure
- Contains data referenced from the FIB in the mainstream or other parts of the file

Object Streams

- Holds binary data for embedded OLE objects within the .doc file

PowerPoint File Formats

Microsoft PowerPoint Presentation File Structure (.ppt/.pptx):

Current User Stream

- Maintains CurrentUserAtom record, that identifies the last user's name to open/modify a target PPT and location of the most recent user edit

PowerPoint Document Stream

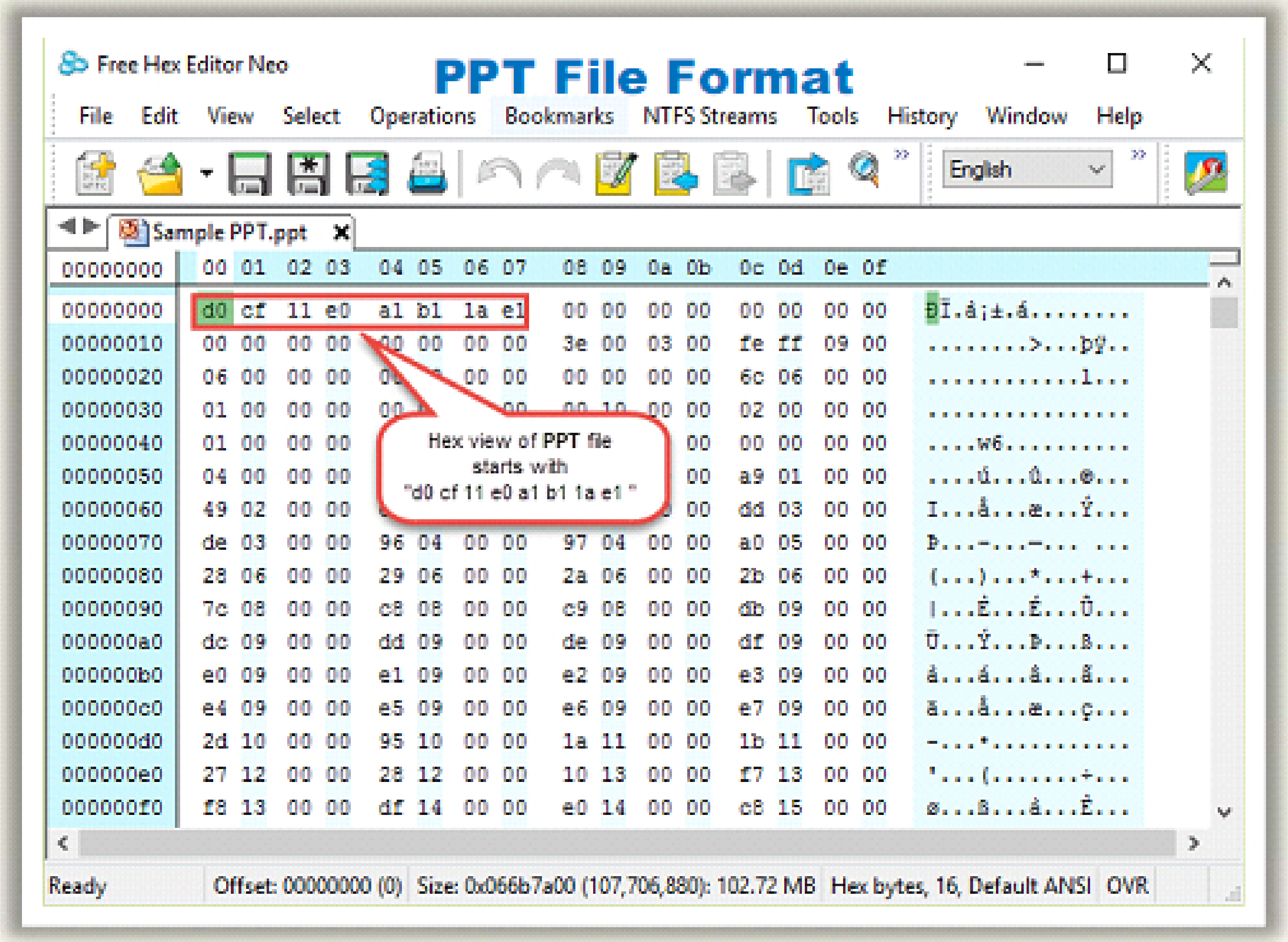
- Contains information about the presentation layout and its contents

Pictures Stream (Optional)

- Contains information about embedded image files within the presentation

Summary Information Streams (Optional)

- The summary information is stored in two storage streams: SummaryInformation and DocumentSummaryInformation



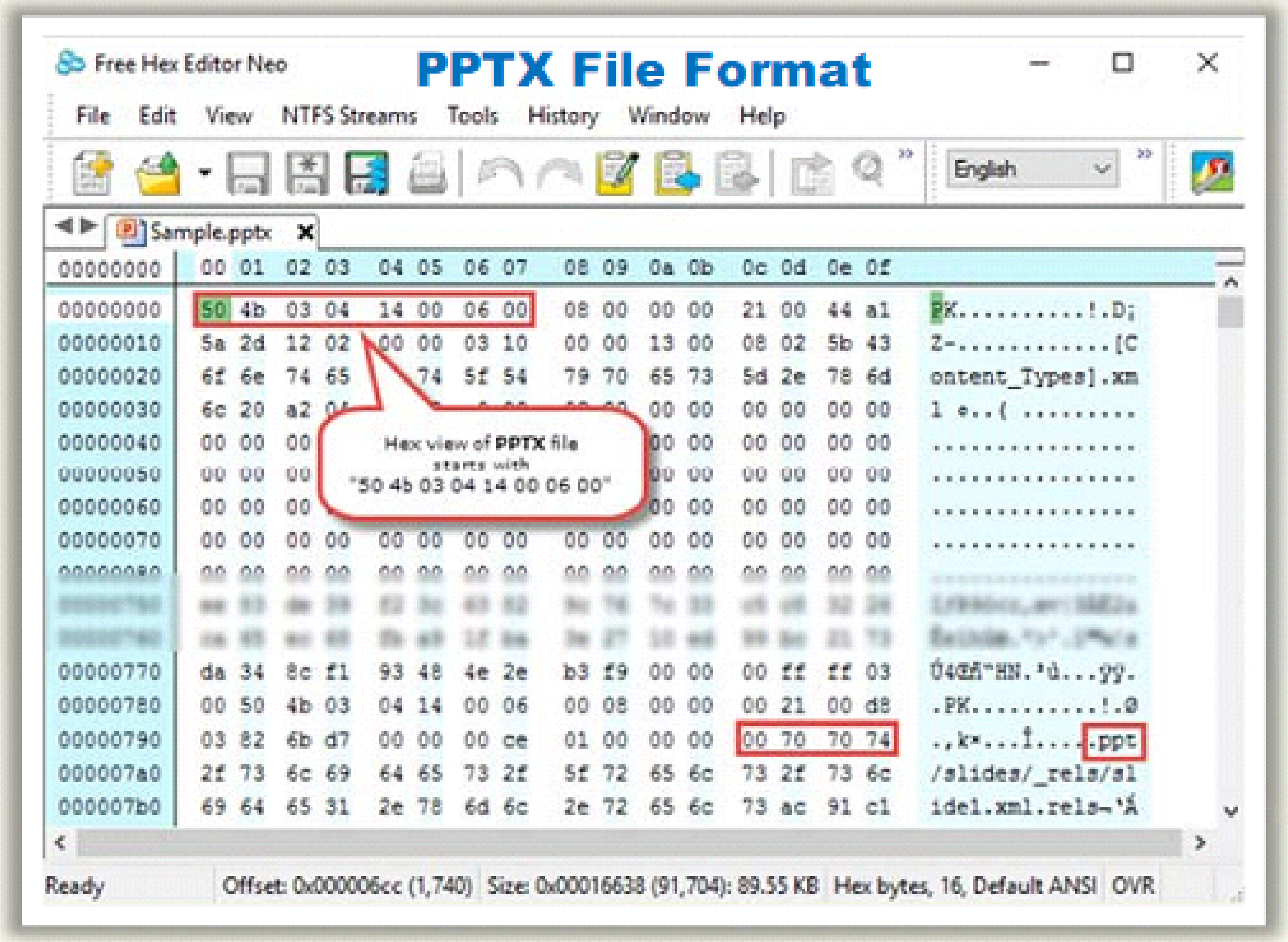
PPT File Format

Free Hex Editor Neo
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
Sample PPT.ppt

```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000  d0 cf 11 e0  a1 b1 1a e1  00 00 00 00  00 00 00 00  ÐÏ.á¡±.á........
00000010  00 00 00 00  00 00 00 00  3e 00 03 00  fe ff 09 00  ........>...þÿ..
00000020  06 00 00 00  0           00 00 00 00  6c 06 00 00  ............l...
00000030  01 00 00 00  00           00 10 00 00  02 00 00 00  ................
00000040  01 00 00 00                        00  00 00 00 00  ....w6..........
00000050  04 00 00 00                        00  a9 01 00 00  ....ú...û...©...
00000060  49 02 00 00                        00  dd 03 00 00  I...å...æ...Ý...
00000070  de 03 00 00  96 04 00 00  97 04 00 00  a0 05 00 00  Þ...–...—... ...
00000080  28 06 00 00  29 06 00 00  2a 06 00 00  2b 06 00 00  (...)...*...+...
00000090  7c 08 00 00  c8 08 00 00  c9 08 00 00  db 09 00 00  |...È...É...Û...
000000a0  dc 09 00 00  dd 09 00 00  de 09 00 00  df 09 00 00  Ü...Ý...Þ...ß...
000000b0  e0 09 00 00  e1 09 00 00  e2 09 00 00  e3 09 00 00  à...á...â...ã...
000000c0  e4 09 00 00  e5 09 00 00  e6 09 00 00  e7 09 00 00  ä...å...æ...ç...
000000d0  2d 10 00 00  95 10 00 00  1a 11 00 00  1b 11 00 00  -...•...........
000000e0  27 12 00 00  28 12 00 00  10 13 00 00  f7 13 00 00  '...(.......÷...
000000f0  f8 13 00 00  df 14 00 00  e0 14 00 00  c8 15 00 00  ø...ß...à...È...
```

Hex view of PPT file starts with "d0 cf 11 e0 a1 b1 1a e1 "

Ready | Offset: 00000000 (0) | Size: 0x066b7a00 (107,706,880): 102.72 MB | Hex bytes, 16, Default ANSI | OVR



PPTX File Format

```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000  50 4b 03 04  14 00 06 00  08 00 00 00  21 00 44 a1  PK..........!.D¡
00000010  5a 2d 12 02  00 00 03 10  00 00 13 00  08 02 5b 43  Z-............[C
00000020  6f 6e 74 65     74 5f 54  79 70 65 73  5d 2e 78 6d  ontent_Types].xm
00000030  6c 20 a2 0                      00 00  00 00 00 00  l ¢..( .........
00000040  00 00 00                        00 00  00 00 00 00  ................
00000050  00 00 00                        00 00  00 00 00 00  ................
00000060  00 00 00                        00 00  00 00 00 00  ................
00000070  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000080  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
[illegible]
[illegible]
00000770  da 34 8c f1  93 48 4e 2e  b3 f9 00 00  00 ff ff 03  Ú4Œñ"HN.³ù...ÿÿ.
00000780  00 50 4b 03  04 14 00 06  00 08 00 00  00 21 00 d8  .PK..........!.Ø
00000790  03 82 6b d7  00 00 00 ce  01 00 00 00  00 70 70 74  ..k×...Î.....ppt
000007a0  2f 73 6c 69  64 65 73 2f  5f 72 65 6c  73 2f 73 6c  /slides/_rels/sl
000007b0  69 64 65 31  2e 78 6d 6c  2e 72 65 6c  73 ac 91 c1  ide1.xml.rels¬'Á
```

Hex view of PPTX file starts with "50 4b 03 04 14 00 06 00"

Ready | Offset: 0x000006cc (1,740) | Size: 0x00016638 (91,704): 89.55 KB | Hex bytes, 16, Default ANSI | OVR

Excel File Formats

Microsoft Excel File Structure (.xls/.xlsx):

An OLE compound file saved in Binary Interchange File Format (BIFF)

Streams

- Workbook stream is the primary stream in .xls file, which contains many substreams

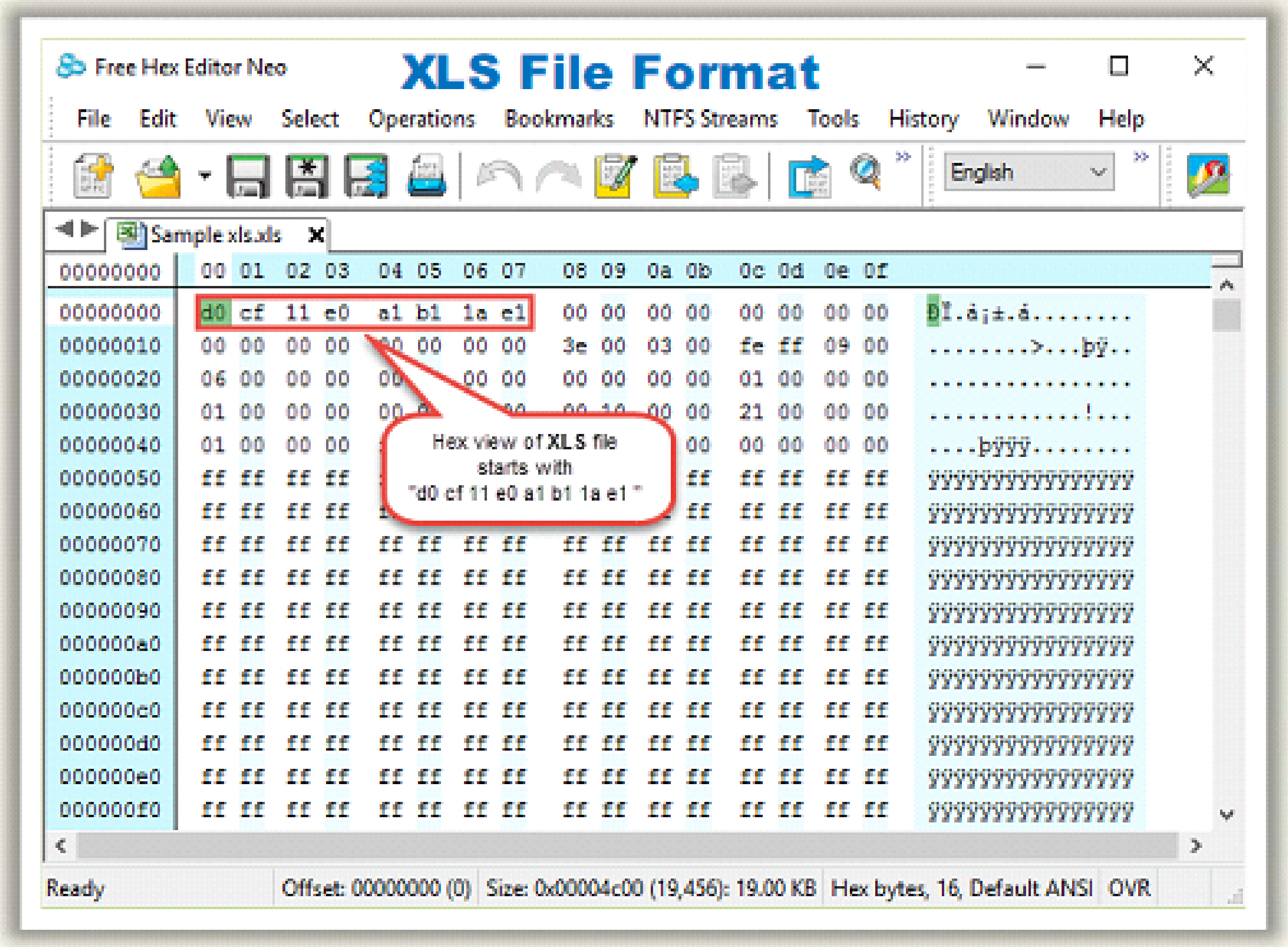
Substreams

- Global substream - specifies global properties and data in a workbook
- Worksheet substream - specifies a sheet in a workbook

Records

- Holds information about each workbook's feature~~s~~
- Components include record size, record type, and record data

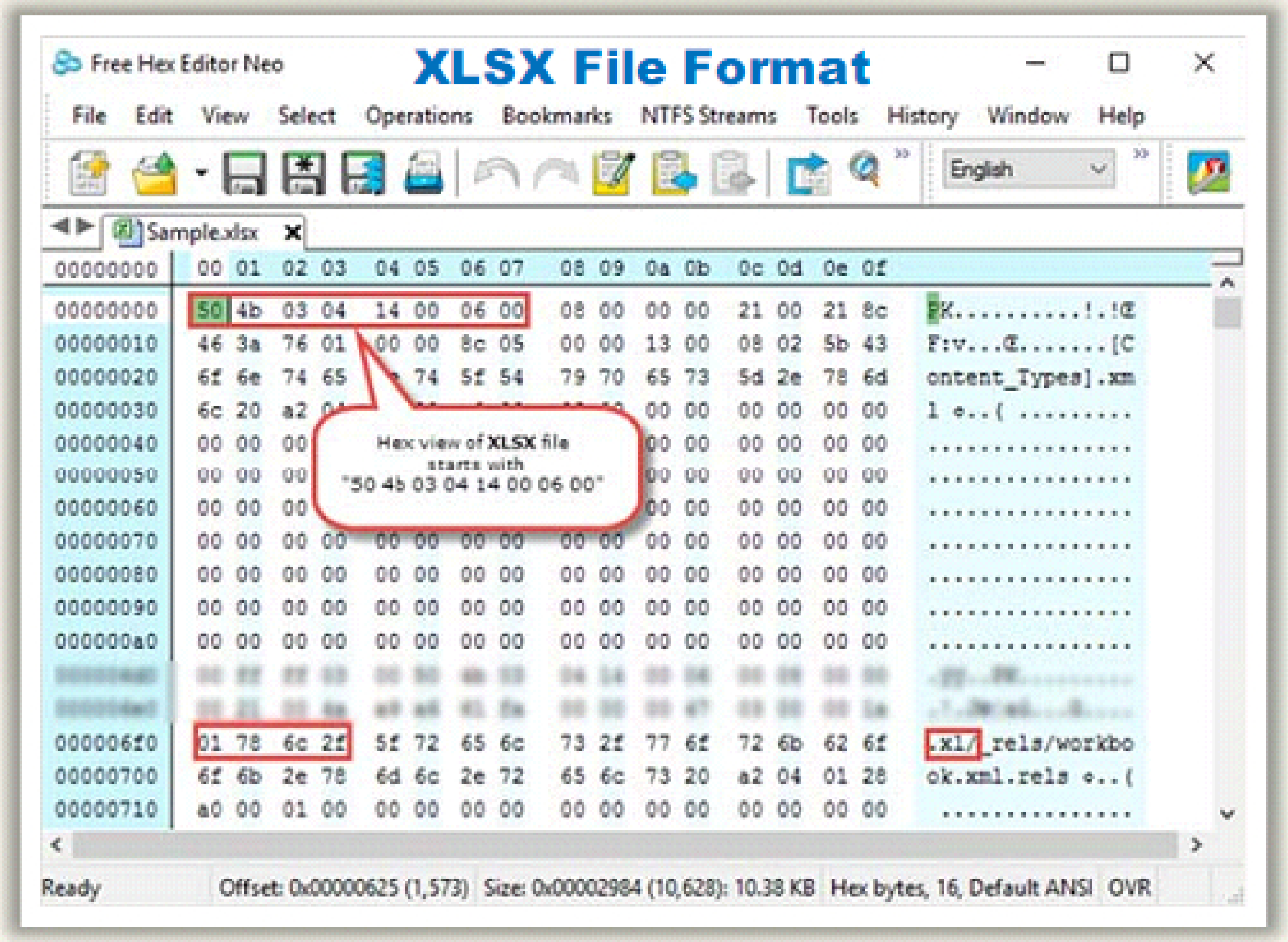
Note: Office Open XML Format (MS Office 2007 and above) is less vulnerable compared to the binary format and is therefore not widely used by attackers as a vector of attack


Free Hex Editor Neo
XLS File Format
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
Sample xls.xls

```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000  d0 cf 11 e0  a1 b1 1a e1  00 00 00 00  00 00 00 00  ÐÏ.à¡±.á........
00000010  00 00 00 00  00 00 00 00  3e 00 03 00  fe ff 09 00  ........>...þÿ..
00000020  06 00 00 00  00       00 00  00 00 00 00  01 00 00 00  ................
00000030  01 00 00 00  00           00 10 00 00  21 00 00 00  ............!...
00000040  01 00 00 00                     00  00 00 00 00  ....þÿÿ.........
00000050  ff ff ff ff                     ff  ff ff ff ff  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000060  ff ff ff ff                     ff  ff ff ff ff  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000070  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000080  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000090  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000a0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000b0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000c0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000d0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000e0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000f0  ff ff ff ff  ff ff ff ff  ff ff ff ff  ff ff ff ff  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

Hex view of **XLS** file starts with "d0 cf 11 e0 a1 b1 1a e1 "

Ready | Offset: 00000000 (0) | Size: 0x00004c00 (19,456): 19.00 KB | Hex bytes, 16, Default ANSI | OVR

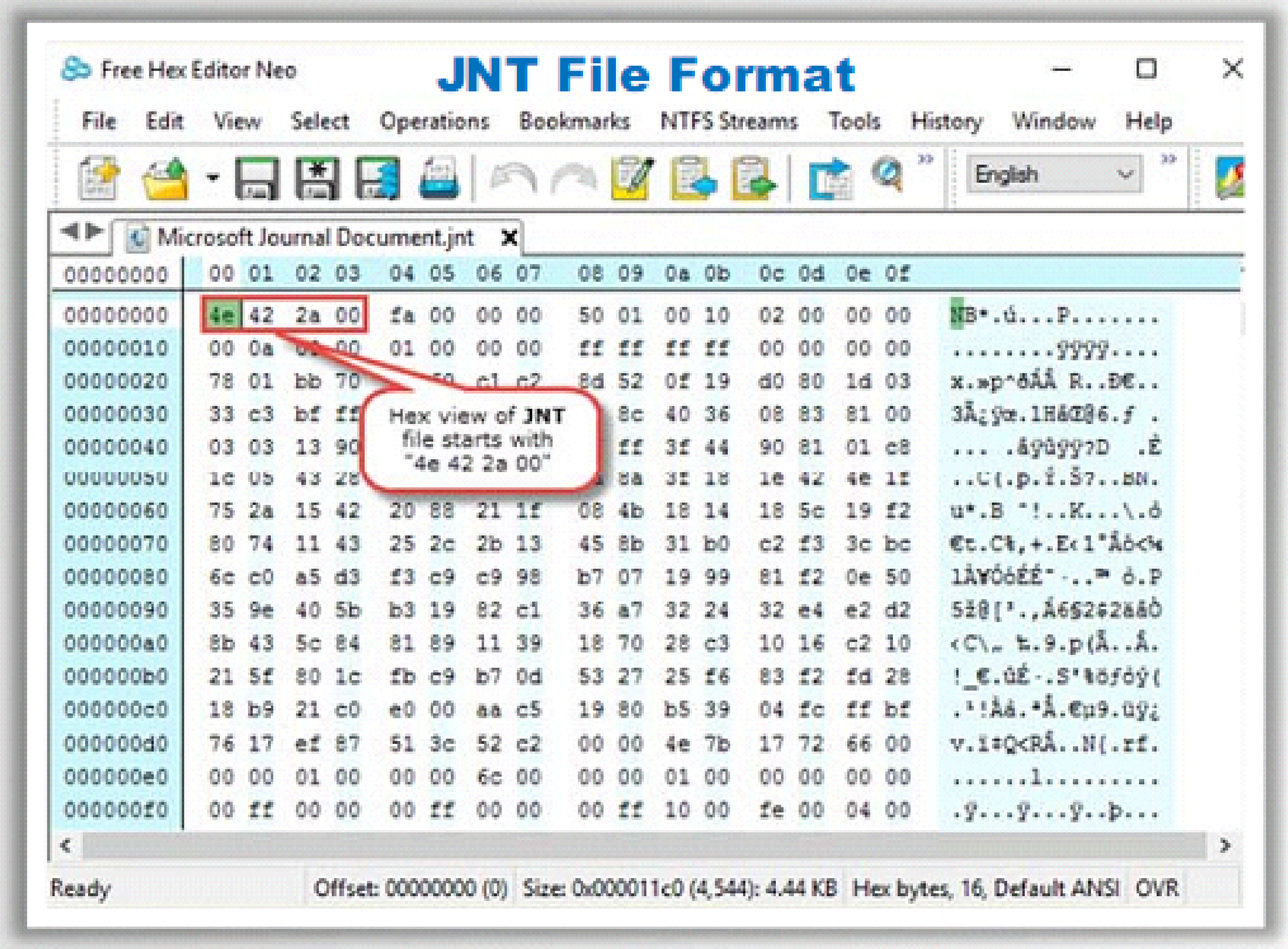

Free Hex Editor Neo
XLSX File Format
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
Sample.xlsx

```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000  50 4b 03 04  14 00 06 00  08 00 00 00  21 00 21 8c  PK..........!.!Œ
00000010  46 3a 76 01  00 00 8c 05  00 00 13 00  08 02 5b 43  F:v...Œ.......[C
00000020  6f 6e 74 65     74 5f 54  79 70 65 73  5d 2e 78 6d  ontent_Types].xm
00000030  6c 20 a2 0                     00 00  00 00 00 00  l ¢..( ........
00000040  00 00 00                       00 00  00 00 00 00  ................
00000050  00 00 00                       00 00  00 00 00 00  ................
00000060  00 00 00                       00 00  00 00 00 00  ................
00000070  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000080  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000090  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
000000a0  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
[redacted]
000006f0  01 78 6c 2f  5f 72 65 6c  73 2f 77 6f  72 6b 62 6f  .xl/_rels/workbo
00000700  6f 6b 2e 78  6d 6c 2e 72  65 6c 73 20  a2 04 01 28  ok.xml.rels ¢..(
00000710  a0 00 01 00  00 00 00 00  00 00 00 00  00 00 00 00   ...............
```

Hex view of **XLSX** file starts with "50 4b 03 04 14 00 06 00"

Ready | Offset: 0x00000625 (1,573) | Size: 0x00002984 (10,628): 10.38 KB | Hex bytes, 16, Default ANSI | OVR

Hex View of Other Popular File Formats

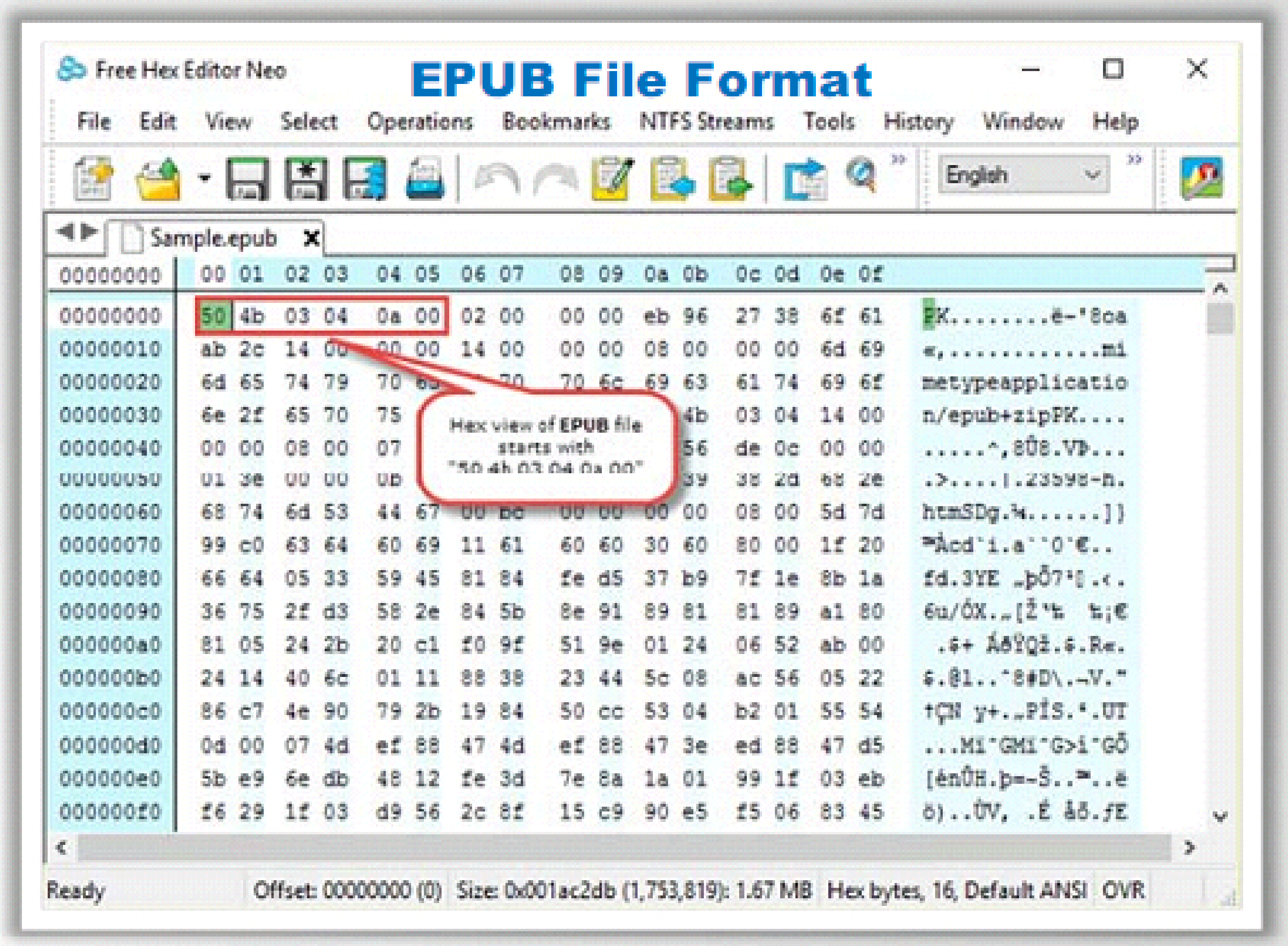

Free Hex Editor Neo
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
Microsoft Journal Document.jnt

JNT File Format

```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000  4e 42 2a 00  fa 00 00 00  50 01 00 10  02 00 00 00  NB*.ú...P.......
00000010  00 0a    00  01 00 00 00  ff ff ff ff  00 00 00 00  ........ÿÿÿÿ....
00000020  78 01 bb 70        c1 c2  8d 52 0f 19  d0 80 1d 03  x.»p^ðÁÂ R..Ð€..
00000030  33 c3 bf ff                  8c 40 36  08 83 81 00  3Ã¿ÿœ.1H‰Œ@6.ƒ .
00000040  03 03 13 90                  ff 3f 44  90 81 01 c8  .... .åÿûÿ?D  .È
00000050  1c 05 43 28                  8a 3f 18  1e 42 4e 1f  ..C(.p.î.Š?..BN.
00000060  75 2a 15 42  20 88 21 1f  08 4b 18 14  18 5c 19 f2  u*.B ˆ!..K...\.ò
00000070  80 74 11 43  25 2c 2b 13  45 8b 31 b0  c2 f3 3c bc  €t.C%,+.E‹1°Âó<¼
00000080  6c c0 a5 d3  f3 c9 c9 98  b7 07 19 99  81 f2 0e 50  lÀ¥ÓóÉÉ˜·..™ ò.P
00000090  35 9e 40 5b  b3 19 82 c1  36 a7 32 24  32 e4 e2 d2  5ž@[³.‚Á6§2$2äâÒ
000000a0  8b 43 5c 84  81 89 11 39  18 70 28 c3  10 16 c2 10  ‹C\„ ‰.9.p(Ã..Â.
000000b0  21 5f 80 1c  fb c9 b7 0d  53 27 25 f6  83 f2 fd 28  !_€.ûÉ·.S'%öƒòý(
000000c0  18 b9 21 c0  e0 00 aa c5  19 80 b5 39  04 fc ff bf  .¹!Àà.ªÅ.€µ9.üÿ¿
000000d0  76 17 ef 87  51 3c 52 c2  00 00 4e 7b  17 72 66 00  v.ï‡Q<RÂ..N{.rf.
000000e0  00 00 01 00  00 00 6c 00  00 00 01 00  00 00 00 00  ......l.........
000000f0  00 ff 00 00  00 ff 00 00  00 ff 10 00  fe 00 04 00  .ÿ...ÿ...ÿ..þ...
```

Hex view of **JNT** file starts with "4e 42 2a 00"

```
Ready   Offset: 00000000 (0)   Size: 0x000011c0 (4,544): 4.44 KB   Hex bytes, 16, Default ANSI   OVR
```

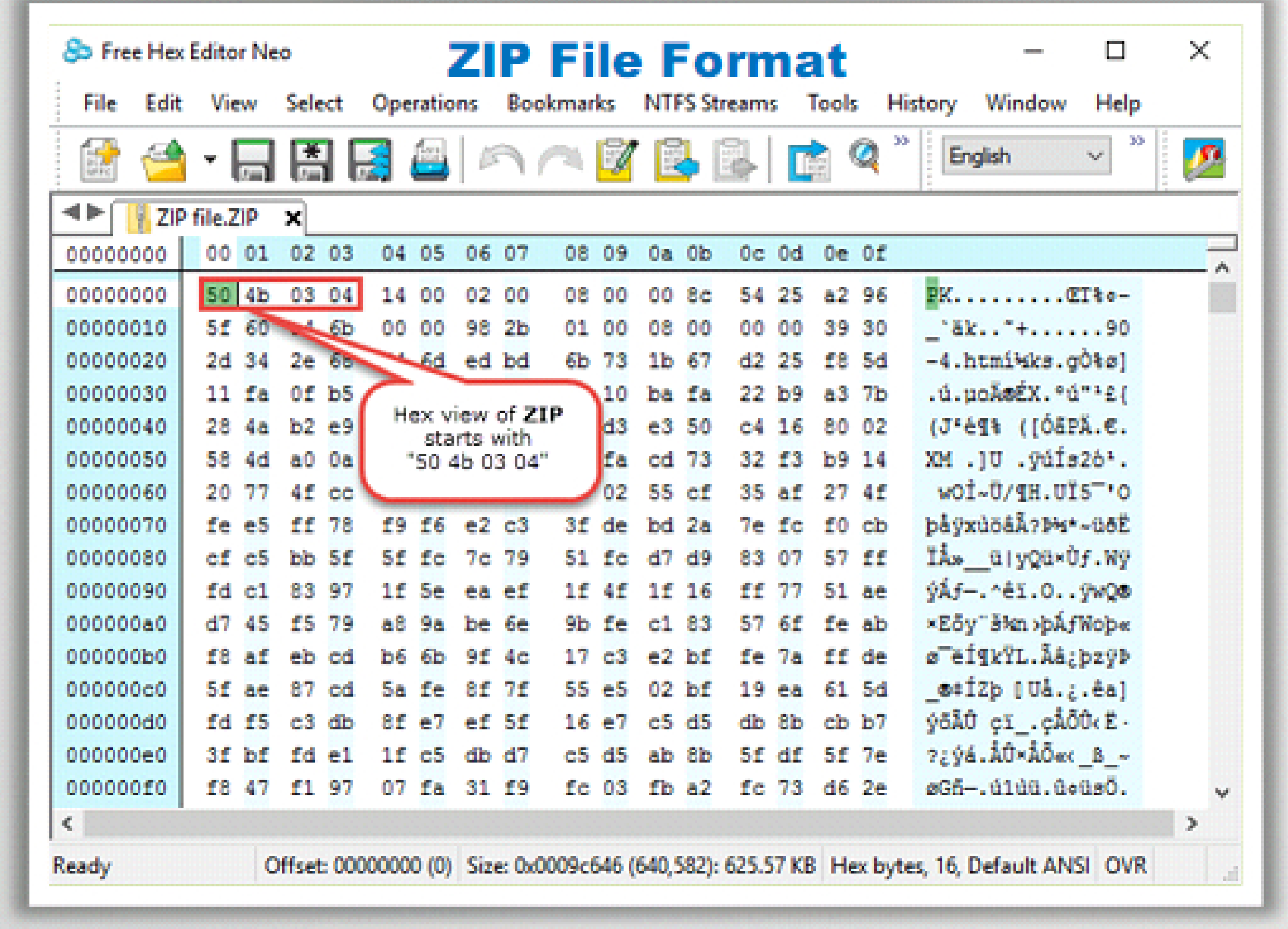

Free Hex Editor Neo
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
Sample.epub

EPUB File Format

```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000  50 4b 03 04  0a 00 02 00  00 00 eb 96  27 38 6f 61  PK........ë–'8oa
00000010  ab 2c 14 00  00 00 14 00  00 00 08 00  00 00 6d 69  «,............mi
00000020  6d 65 74 79  70 65        70 6c 69 63  61 74 69 6f  metypeapplicatio
00000030  6e 2f 65 70  75              4b  03 04 14 00  n/epub+zipPK....
00000040  00 00 08 00  07              56  de 0c 00 00  ......^,8Û8.VÞ...
00000050  01 3e 00 00  0b              39  38 2d 68 2e  .>....|.23598-h.
00000060  68 74 6d 53  44 67 00 bc  00 00 00 00  08 00 5d 7d  htmSDg.¼......]}
00000070  99 c0 63 64  60 69 11 61  60 60 30 60  80 00 1f 20  ™Àcd`i.a``0`€..
00000080  66 64 05 33  59 45 81 84  fe d5 37 b9  7f 1e 8b 1a  fd.3YE „þÕ7¹.‹.
00000090  36 75 2f d3  58 2e 84 5b  8e 91 89 81  81 89 a1 80  6u/ÓX.„[Ž‘‰  ‰¡€
000000a0  81 05 24 2b  20 c1 f0 9f  51 9e 01 24  06 52 ab 00   .$+ ÁðŸQž.$.R«.
000000b0  24 14 40 6c  01 11 88 38  23 44 5c 08  ac 56 05 22  $.@l..ˆ8#D\.¬V."
000000c0  86 c7 4e 90  79 2b 19 84  50 cc 53 04  b2 01 55 54  †ÇN y+.„PÌS.².UT
000000d0  0d 00 07 4d  ef 88 47 4d  ef 88 47 3e  ed 88 47 d5  ...MïˆGMïˆG>íˆGÕ
000000e0  5b e9 6e db  48 12 fe 3d  7e 8a 1a 01  99 1f 03 eb  [énÛH.þ=~Š..™..ë
000000f0  f6 29 1f 03  d9 56 2c 8f  15 c9 90 e5  f5 06 83 45  ö)..ÙV, .É åõ.ƒE
```

Hex view of **EPUB** file starts with "50 4b 03 04 0a 00"

```
Ready   Offset: 00000000 (0)   Size: 0x001ac2db (1,753,819): 1.67 MB   Hex bytes, 16, Default ANSI   OVR
```

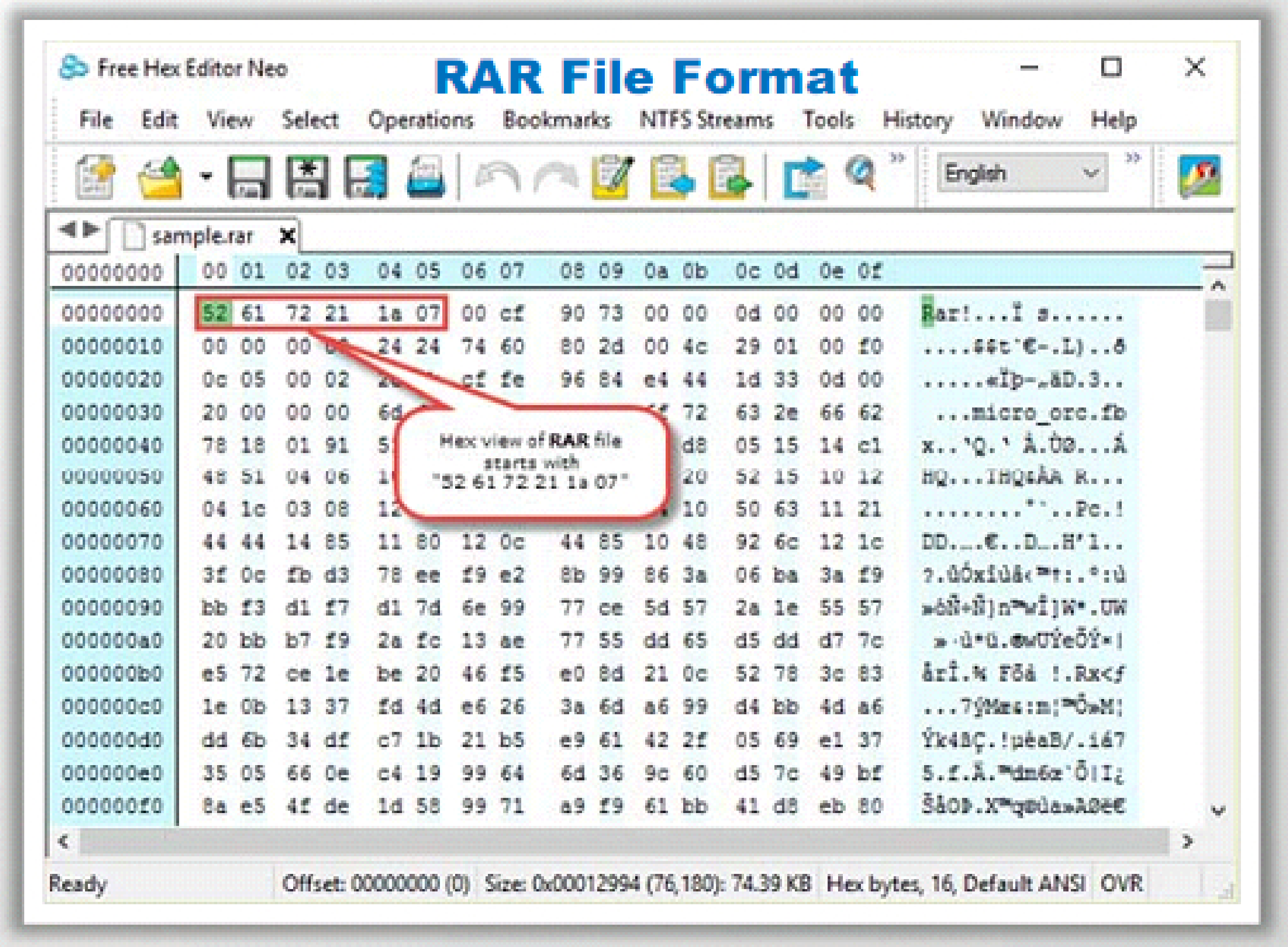

Free Hex Editor Neo
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
ZIP file.ZIP

ZIP File Format

```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000  50 4b 03 04  14 00 02 00  08 00 00 8c  54 25 a2 96  PK........ŒT%¢–
00000010  5f 60    6b  00 00 98 2b  01 00 08 00  00 00 39 30  _`äk..˜+......90
00000020  2d 34 2e 68     6d ed bd  6b 73 1b 67  d2 25 f8 5d  -4.htmí½ks.gÒ%ø]
00000030  11 fa 0f b5                10 ba fa  22 b9 a3 7b  .ú.µoÃ®ÉX.ºú"¹£{
00000040  28 4a b2 e9                d3 e3 50  c4 16 80 02  (J²é¶ ([ÓãPÄ.€.
00000050  58 4d a0 0a                fa cd 73  32 f3 b9 14  XM .]Ü .úÍs2ó¹.
00000060  20 77 4f cc                02 55 cf  35 af 27 4f   wOÌ~Ü/¶H.UÏ5¯'O
00000070  fe e5 ff 78  f9 f6 e2 c3  3f de bd 2a  7e fc f0 cb  þåÿxùöâÃ?Þ½*~üðË
00000080  cf c5 bb 5f  5f fc 7c 79  51 fc d7 d9  83 07 57 ff  ÏÅ»__ü|yQü×Ùƒ.Wÿ
00000090  fd c1 83 97  1f 5e ea ef  1f 4f 1f 16  ff 77 51 ae  ýÁƒ—.^êï.O..ÿwQ®
000000a0  d7 45 f5 79  a8 9a be 6e  9b fe c1 83  57 6f fe ab  ×Eõy¨š¾n›þÁƒWoþ«
000000b0  f8 af eb cd  b6 6b 9f 4c  17 c3 e2 bf  fe 7a ff de  ø¯ëÍ¶kŸL.Ãâ¿þzÿÞ
000000c0  5f ae 87 cd  5a fe 8f 7f  55 e5 02 bf  19 ea 61 5d  _®‡ÍZþ .Uå.¿.êa]
000000d0  fd f5 c3 db  8f e7 ef 5f  16 e7 c5 d5  db 8b cb b7  ýõÃÛ çï_.çÅÕÛ‹Ë·
000000e0  3f bf fd e1  1f c5 db d7  c5 d5 ab 8b  5f df 5f 7e  ?¿ýá.ÅÛ×ÅÕ«‹_ß_~
000000f0  f8 47 f1 97  07 fa 31 f9  fc 03 fb a2  fc 73 d6 2e  øGñ—.ú1ùü.û¢üsÖ.
```

Hex view of **ZIP** starts with "50 4b 03 04"

```
Ready   Offset: 00000000 (0)   Size: 0x0009c646 (640,582): 625.57 KB   Hex bytes, 16, Default ANSI   OVR
```


Free Hex Editor Neo
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
sample.rar

RAR File Format

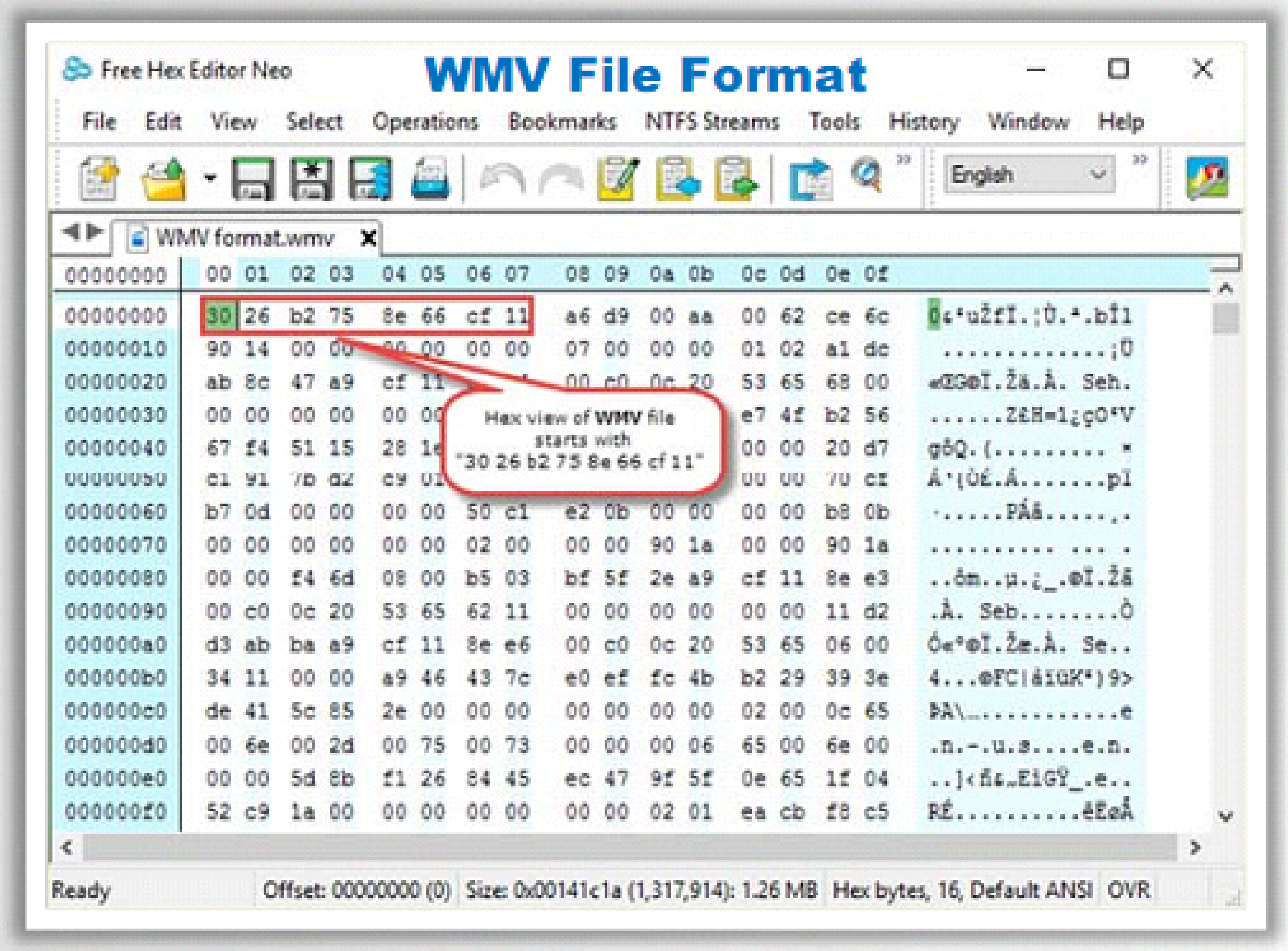
```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000  52 61 72 21  1a 07 00 cf  90 73 00 00  0d 00 00 00  Rar!...Ï s......
00000010  00 00 00     24 24 74 60  80 2d 00 4c  29 01 00 f0  ....$$t`€-.L)..ð
00000020  0c 05 00 02        cf fe  96 84 e4 44  1d 33 0d 00  ......«Ïþ–„äD.3..
00000030  20 00 00 00  6d              72  63 2e 66 62   ...micro_orc.fb
00000040  78 18 01 91  5               d8  05 15 14 c1  x..‘Q.' À.ÙØ...Á
00000050  48 51 04 06  1               20  52 15 10 12  HQ...IHQ¤ÀÀ R...
00000060  04 1c 03 08  12              10  50 63 11 21  ........"`..Pc.!
00000070  44 44 14 85  11 80 12 0c  44 85 10 48  92 6c 12 1c  DD.….€..D….H'l..
00000080  3f 0c fb d3  78 ee f9 e2  8b 99 86 3a  06 ba 3a f9  ?.ûÓxîùâ‹™†:.º:ù
00000090  bb f3 d1 f7  d1 7d 6e 99  77 ce 5d 57  2a 1e 55 57  »óÑ÷Ñ}n™wÎ]W*.UW
000000a0  20 bb b7 f9  2a fc 13 ae  77 55 dd 65  d5 dd d7 7c   »·ù*ü.®wUÝeÕÝ×|
000000b0  e5 72 ce 1e  be 20 46 f5  e0 8d 21 0c  52 78 3c 83  årÎ.¾ Fõà !.Rx<ƒ
000000c0  1e 0b 13 37  fd 4d e6 26  3a 6d a6 99  d4 bb 4d a6  ...7ýMæ&:m¦™Ô»M¦
000000d0  dd 6b 34 df  c7 1b 21 b5  e9 61 42 2f  05 69 e1 37  Ýk4ßÇ.!µéaB/.iá7
000000e0  35 05 66 0e  c4 19 99 64  6d 36 9c 60  d5 7c 49 bf  5.f.Ä.™dm6œ`Õ|I¿
000000f0  8a e5 4f de  1d 58 99 71  a9 f9 61 bb  41 d8 eb 80  ŠåOÞ.X™q©ùa»AØë€
```

Hex view of **RAR** file starts with "52 61 72 21 1a 07"

```
Ready   Offset: 00000000 (0)   Size: 0x00012994 (76,180): 74.39 KB   Hex bytes, 16, Default ANSI   OVR
```

Hex View of Popular Video File Formats

CHFI Computer Hacking Forensic Investigator


Free Hex Editor Neo
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English

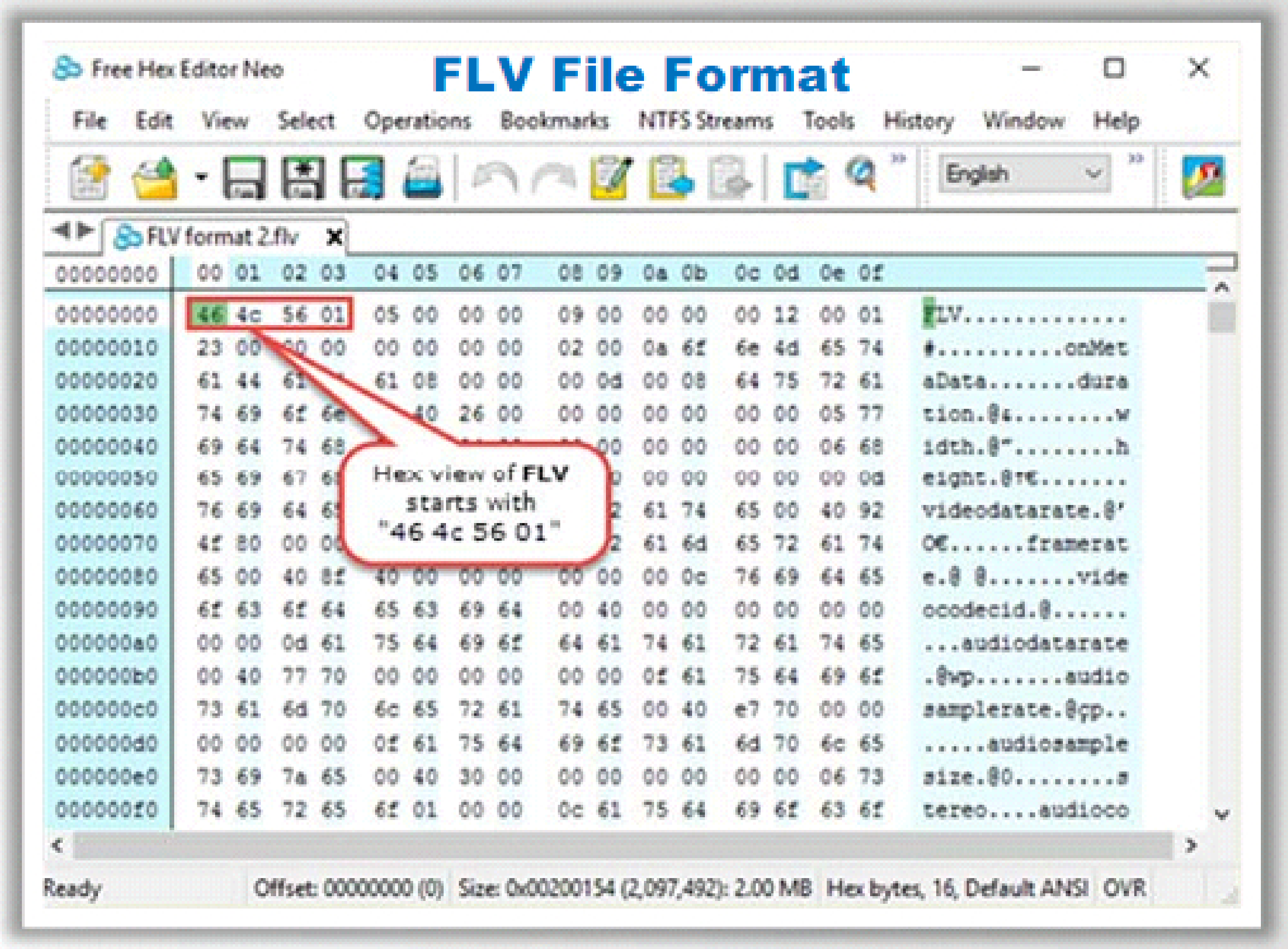
WMV File Format

WMV format.wmv

```
00000000   00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000   30 26 b2 75  8e 66 cf 11  a6 d9 00 aa  00 62 ce 6c   0&²uŽfÏ.¦Ù.ª.bÎl
00000010   90 14 00 00  00 00 00 00  07 00 00 00  01 02 a1 dc   ..............¡Ü
00000020   ab 8c 47 a9  cf 11        00 c0 0c 20  53 65 68 00   «ŒG©Ï.Žä.À. Seh.
00000030   00 00 00 00  00 00                     e7 4f b2 56   ......Z£H=1¿çO²V
00000040   67 f4 51 15  28 1e                     00 00 20 d7   gôQ.(......... ×
00000050   c1 91 7b d2  c9 01                     00 00 70 cf   Á'{ÒÉ.Á.......pÏ
00000060   b7 0d 00 00  00 00 50 c1  e2 0b 00 00  00 00 b8 0b   ·.....PÁâ.....¸.
00000070   00 00 00 00  00 00 02 00  00 00 90 1a  00 00 90 1a   ................
00000080   00 00 f4 6d  08 00 b5 03  bf 5f 2e a9  cf 11 8e e3   ..ôm..µ.¿_.©Ï.Žã
00000090   00 c0 0c 20  53 65 62 11  00 00 00 00  00 00 11 d2   .À. Seb.......Ò
000000a0   d3 ab ba a9  cf 11 8e e6  00 c0 0c 20  53 65 06 00   Ó«º©Ï.Žæ.À. Se..
000000b0   34 11 00 00  a9 46 43 7c  e0 ef fc 4b  b2 29 39 3e   4...©FC|àïüK²)9>
000000c0   de 41 5c 85  2e 00 00 00  00 00 00 00  02 00 0c 65   ÞA\…...........e
000000d0   00 6e 00 2d  00 75 00 73  00 00 00 06  65 00 6e 00   .n.-.u.s....e.n.
000000e0   00 00 5d 8b  f1 26 84 45  ec 47 9f 5f  0e 65 1f 04   ..]‹ñ&„EìGŸ_.e..
000000f0   52 c9 1a 00  00 00 00 00  00 00 02 01  ea cb f8 c5   RÉ..........êËøÅ
```

Hex view of **WMV** file starts with "30 26 b2 75 8e 66 cf 11"

Ready | Offset: 0000000 (0) | Size: 0x00141c1a (1,317,914): 1.26 MB | Hex bytes, 16, Default ANSI | OVR


Free Hex Editor Neo
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English

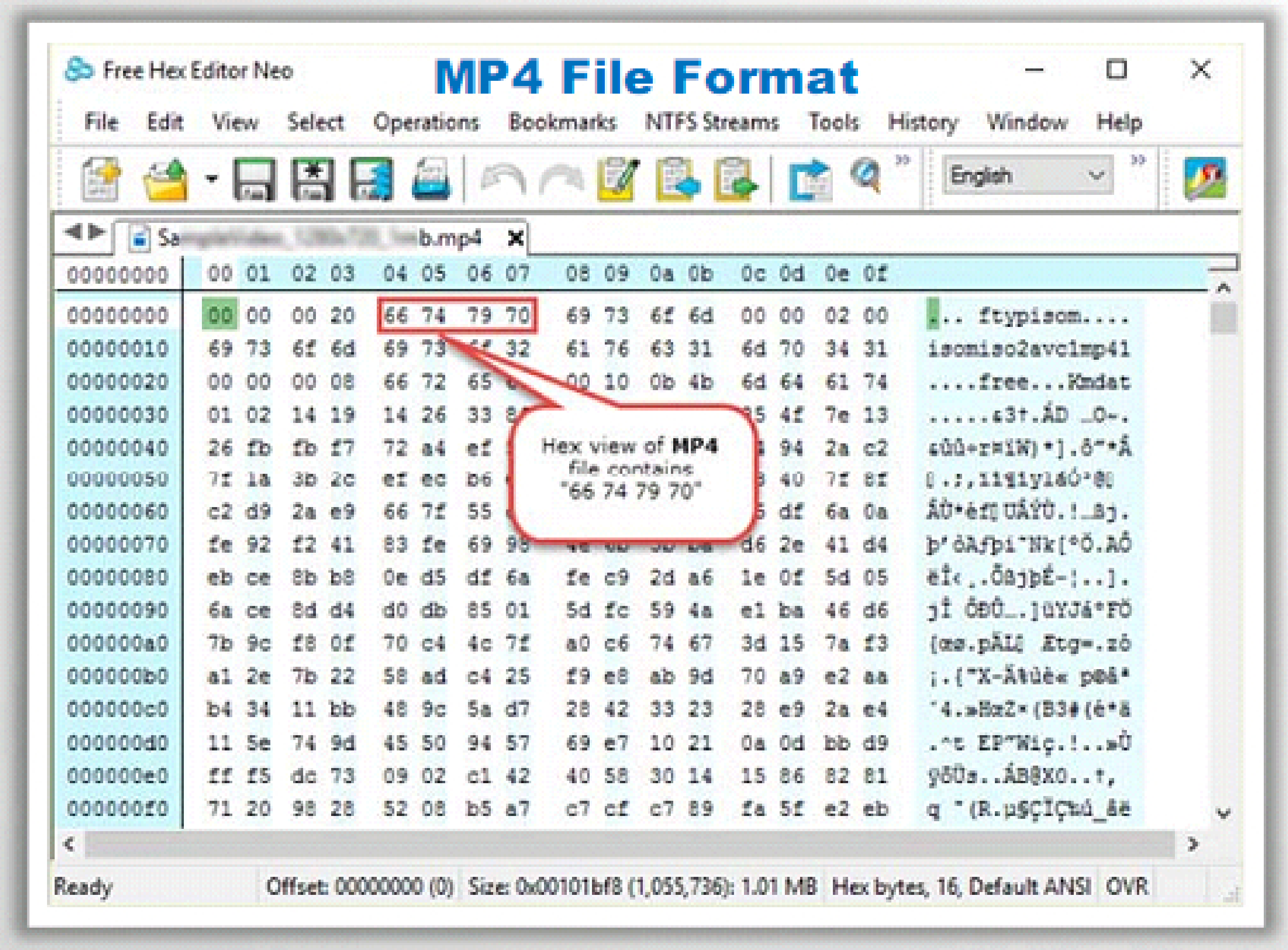
FLV File Format

FLV format 2.flv

```
00000000   00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000   46 4c 56 01  05 00 00 00  09 00 00 00  00 12 00 01   FLV.............
00000010   23 00 00 00  00 00 00 00  02 00 0a 6f  6e 4d 65 74   #.........onMet
00000020   61 44 61        61 08 00 00  00 0d 00 08  64 75 72 61   aData.......dura
00000030   74 69 6f 6e     40 26 00  00 00 00 00  00 00 05 77   tion.@&........w
00000040   69 64 74 68                  00 00 00  00 00 06 68   idth.@"........h
00000050   65 69 67                        00 00  00 00 00 0d   eight.@†€.......
00000060   76 69 64 65                     61 74  65 00 40 92   videodatarate.@'
00000070   4f 80 00 00                     61 6d  65 72 61 74   O€......framerat
00000080   65 00 40 8f  40 00 00 00  00 00 00 0c  76 69 64 65   e.@ @.......vide
00000090   6f 63 6f 64  65 63 69 64  00 40 00 00  00 00 00 00   ocodecid.@......
000000a0   00 00 0d 61  75 64 69 6f  64 61 74 61  72 61 74 65   ...audiodatarate
000000b0   00 40 77 70  00 00 00 00  00 00 0f 61  75 64 69 6f   .@wp.......audio
000000c0   73 61 6d 70  6c 65 72 61  74 65 00 40  e7 70 00 00   samplerate.@çp..
000000d0   00 00 00 00  0f 61 75 64  69 6f 73 61  6d 70 6c 65   .....audiosample
000000e0   73 69 7a 65  00 40 30 00  00 00 00 00  00 00 06 73   size.@0........s
000000f0   74 65 72 65  6f 01 00 00  0c 61 75 64  69 6f 63 6f   tereo....audioco
```

Hex view of **FLV** starts with "46 4c 56 01"

Ready | Offset: 00000000 (0) | Size: 0x00200154 (2,097,492): 2.00 MB | Hex bytes, 16, Default ANSI | OVR


Free Hex Editor Neo
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English

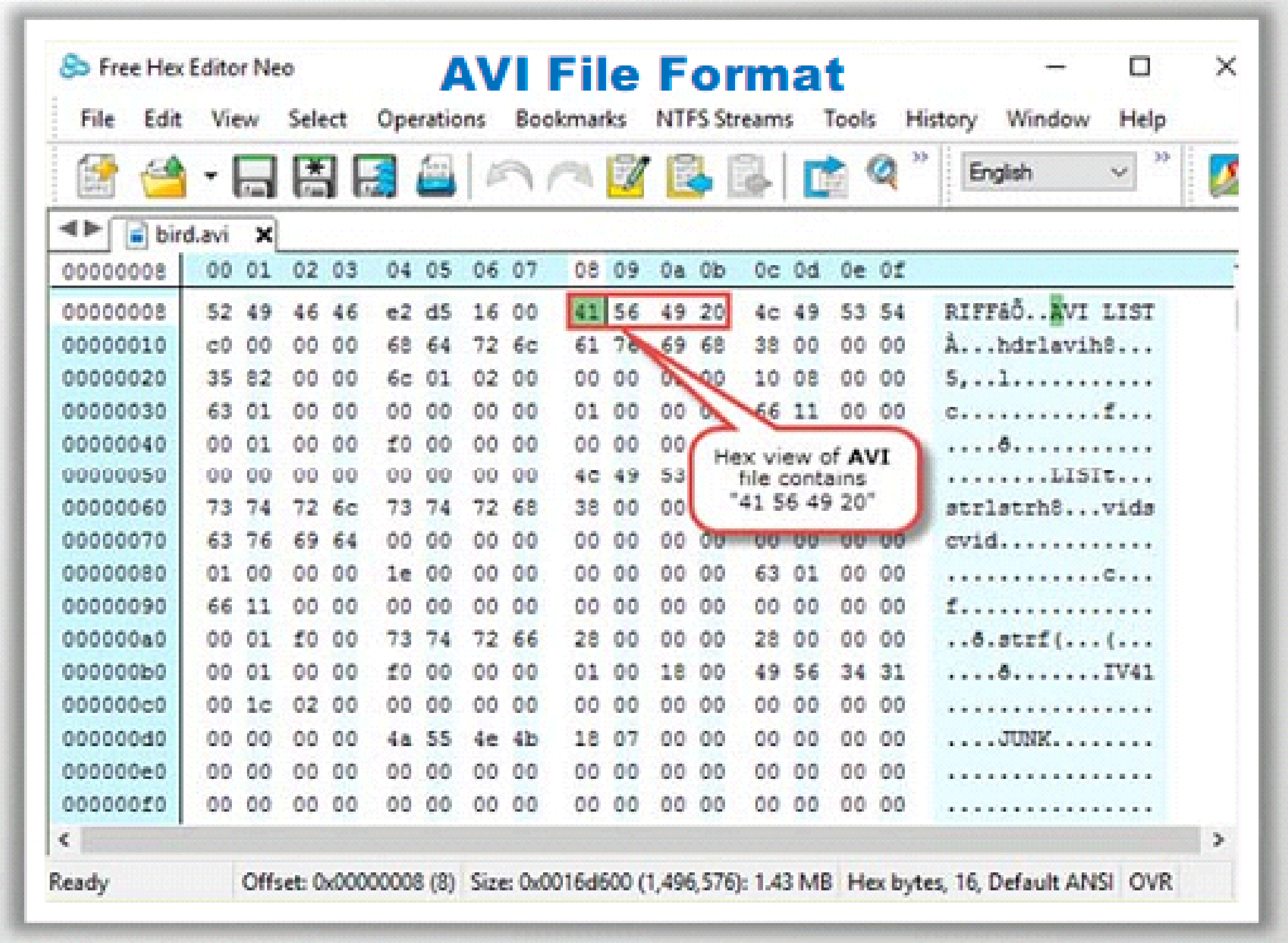
MP4 File Format

Sa[redacted]b.mp4

```
00000000   00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000   00 00 00 20  66 74 79 70  69 73 6f 6d  00 00 02 00   ... ftypisom....
00000010   69 73 6f 6d  69 73 6f 32  61 76 63 31  6d 70 34 31   isomiso2avc1mp41
00000020   00 00 00 08  66 72 65        00 10 0b 4b  6d 64 61 74   ....free...Kmdat
00000030   01 02 14 19  14 26 33                     4f 7e 13   .....&3†.ÅD …O~.
00000040   26 fb fb f7  72 a4 ef                     94 2a c2   &ûû÷r¤ïiW)•].ô"*Â
00000050   7f 1a 3b 2c  ef ec b6                     40 7f 8f   .;,ïì¶ýiáÓ°@.
00000060   c2 d9 2a e9  66 7f 55                     df 6a 0a   ÂÙ*éf.UÁÝÚ.!…ßj.
00000070   fe 92 f2 41  83 fe 69               d6 2e  41 d4   þ'òAƒþi˜Nk[°Ö.AÔ
00000080   eb ce 8b b8  0e d5 df 6a  fe c9 2d a6  1e 0f 5d 05   ëÎ‹¸.ÕßjþÉ-¦..].
00000090   6a ce 8d d4  d0 db 85 01  5d fc 59 4a  e1 ba 46 d6   jÎ ÔÐÛ…]üYJáºFÖ
000000a0   7b 9c f8 0f  70 c4 4c 7f  a0 c6 74 67  3d 15 7a f3   {œø.pÄL Ætg=.zó
000000b0   a1 2e 7b 22  58 ad c4 25  f9 e8 ab 9d  70 a9 e2 aa   ¡.{"X-Ä%ùè« p©âª
000000c0   b4 34 11 bb  48 9c 5a d7  28 42 33 23  28 e9 2a e4   ´4.»HœZ×(B3#(é*ä
000000d0   11 5e 74 9d  45 50 94 57  69 e7 10 21  0a 0d bb d9   .^t EP"Wiç.!..»Ù
000000e0   ff f5 dc 73  09 02 c1 42  40 58 30 14  15 86 82 81   ÿõÜs..ÁB@X0..†,
000000f0   71 20 98 28  52 08 b5 a7  c7 cf c7 89  fa 5f e2 eb   q ˜(R.µ§ÇÏÇ‰ú_âë
```

Hex view of **MP4** file contains "66 74 79 70"

Ready | Offset: 00000000 (0) | Size: 0x00101bf8 (1,055,736): 1.01 MB | Hex bytes, 16, Default ANSI | OVR


Free Hex Editor Neo
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English

AVI File Format

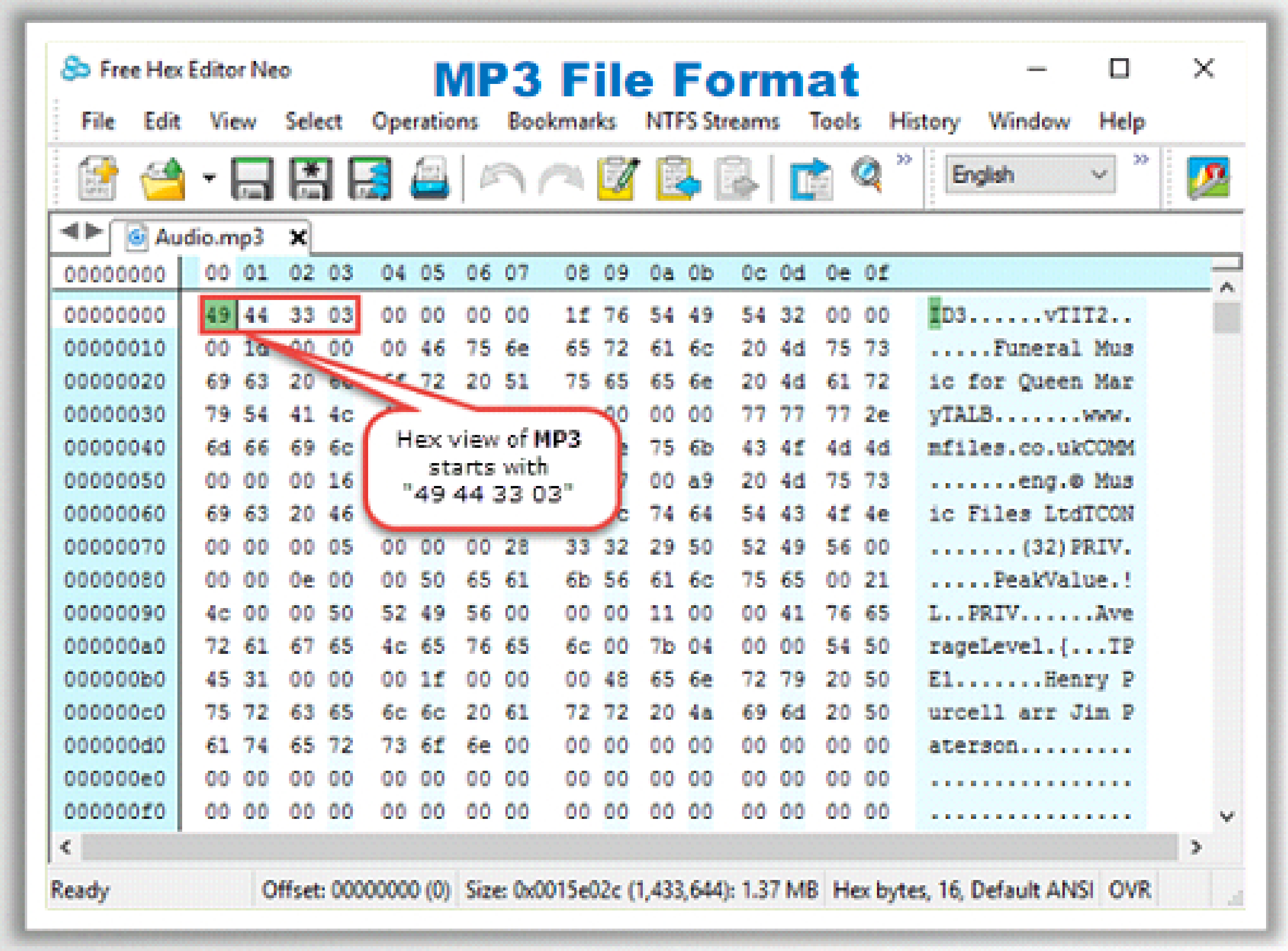
bird.avi

```
00000008   00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000008   52 49 46 46  e2 d5 16 00  41 56 49 20  4c 49 53 54   RIFFâÕ..AVI LIST
00000010   c0 00 00 00  68 64 72 6c  61 76 69 68  38 00 00 00   À...hdrlavih8...
00000020   35 82 00 00  6c 01 02 00  00 00        10 08 00 00   5,..l...........
00000030   63 01 00 00  00 00 00 00  01 00 00        11 00 00   c...........f...
00000040   00 01 00 00  f0 00 00 00  00 00 00                   ....ð...........
00000050   00 00 00 00  00 00 00 00  4c 49 53                   ........LIST...
00000060   73 74 72 6c  73 74 72 68  38 00 00                   strlstrh8...vids
00000070   63 76 69 64  00 00 00 00  00 00 00 00  00 00 00 00   cvid............
00000080   01 00 00 00  1e 00 00 00  00 00 00 00  63 01 00 00   ............c...
00000090   66 11 00 00  00 00 00 00  00 00 00 00  00 00 00 00   f...............
000000a0   00 01 f0 00  73 74 72 66  28 00 00 00  28 00 00 00   ..ð.strf(...(...
000000b0   00 01 00 00  f0 00 00 00  01 00 18 00  49 56 34 31   ....ð.......IV41
000000c0   00 1c 02 00  00 00 00 00  00 00 00 00  00 00 00 00   ................
000000d0   00 00 00 00  4a 55 4e 4b  18 07 00 00  00 00 00 00   ....JUNK........
000000e0   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00   ................
000000f0   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00   ................
```

Hex view of **AVI** file contains "41 56 49 20"

Ready | Offset: 0x00000008 (8) | Size: 0x0016d600 (1,496,576): 1.43 MB | Hex bytes, 16, Default ANSI | OVR

Hex View of Popular Audio File Formats

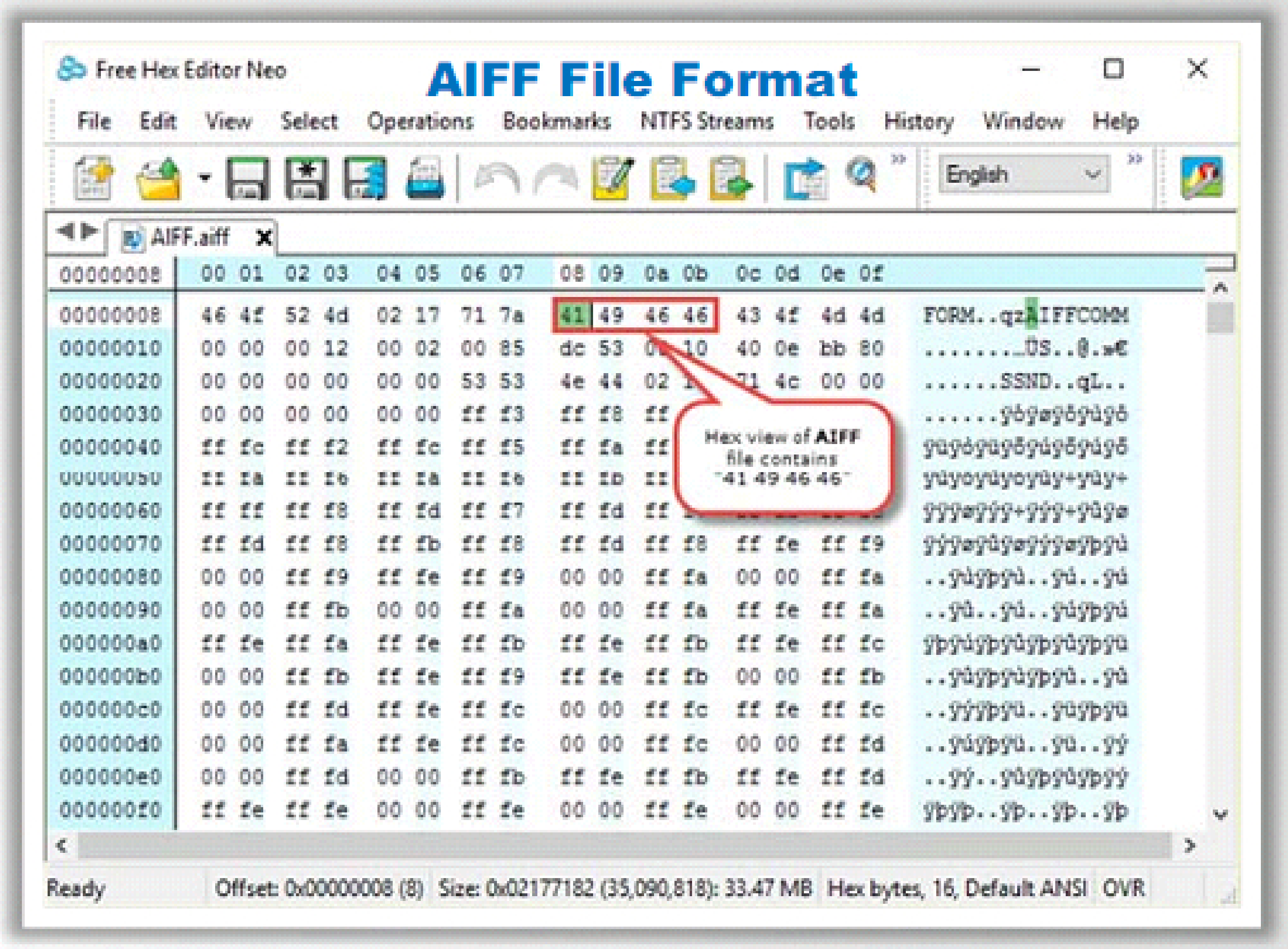


MP3 File Format

```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f

00000000  49 44 33 03  00 00 00 00  1f 76 54 49  54 32 00 00   ID3......vTIT2..
00000010  00 1d 00 00  00 46 75 6e  65 72 61 6c  20 4d 75 73   .....Funeral Mus
00000020  69 63 20 66  6f 72 20 51  75 65 65 6e  20 4d 61 72   ic for Queen Mar
00000030  79 54 41 4c                  00 00 00  77 77 77 2e   yTALB.......www.
00000040  6d 66 69 6c                  e 75 6b  43 4f 4d 4d   mfiles.co.ukCOMM
00000050  00 00 00 16                  7 00 a9  20 4d 75 73   .......eng.© Mus
00000060  69 63 20 46                  c 74 64  54 43 4f 4e   ic Files LtdTCON
00000070  00 00 00 05  00 00 00 28  33 32 29 50  52 49 56 00   .......(32)PRIV.
00000080  00 00 0e 00  00 50 65 61  6b 56 61 6c  75 65 00 21   .....PeakValue.!
00000090  4c 00 00 50  52 49 56 00  00 00 11 00  00 41 76 65   L..PRIV......Ave
000000a0  72 61 67 65  4c 65 76 65  6c 00 7b 04  00 00 54 50   rageLevel.{...TP
000000b0  45 31 00 00  00 1f 00 00  00 48 65 6e  72 79 20 50   E1.......Henry P
000000c0  75 72 63 65  6c 6c 20 61  72 72 20 4a  69 6d 20 50   urcell arr Jim P
000000d0  61 74 65 72  73 6f 6e 00  00 00 00 00  00 00 00 00   aterson.........
000000e0  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00   ................
000000f0  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00   ................
```

Hex view of **MP3** starts with "49 44 33 03"

Ready | Offset: 00000000 (0) | Size: 0x0015e02c (1,433,644): 1.37 MB | Hex bytes, 16, Default ANSI | OVR

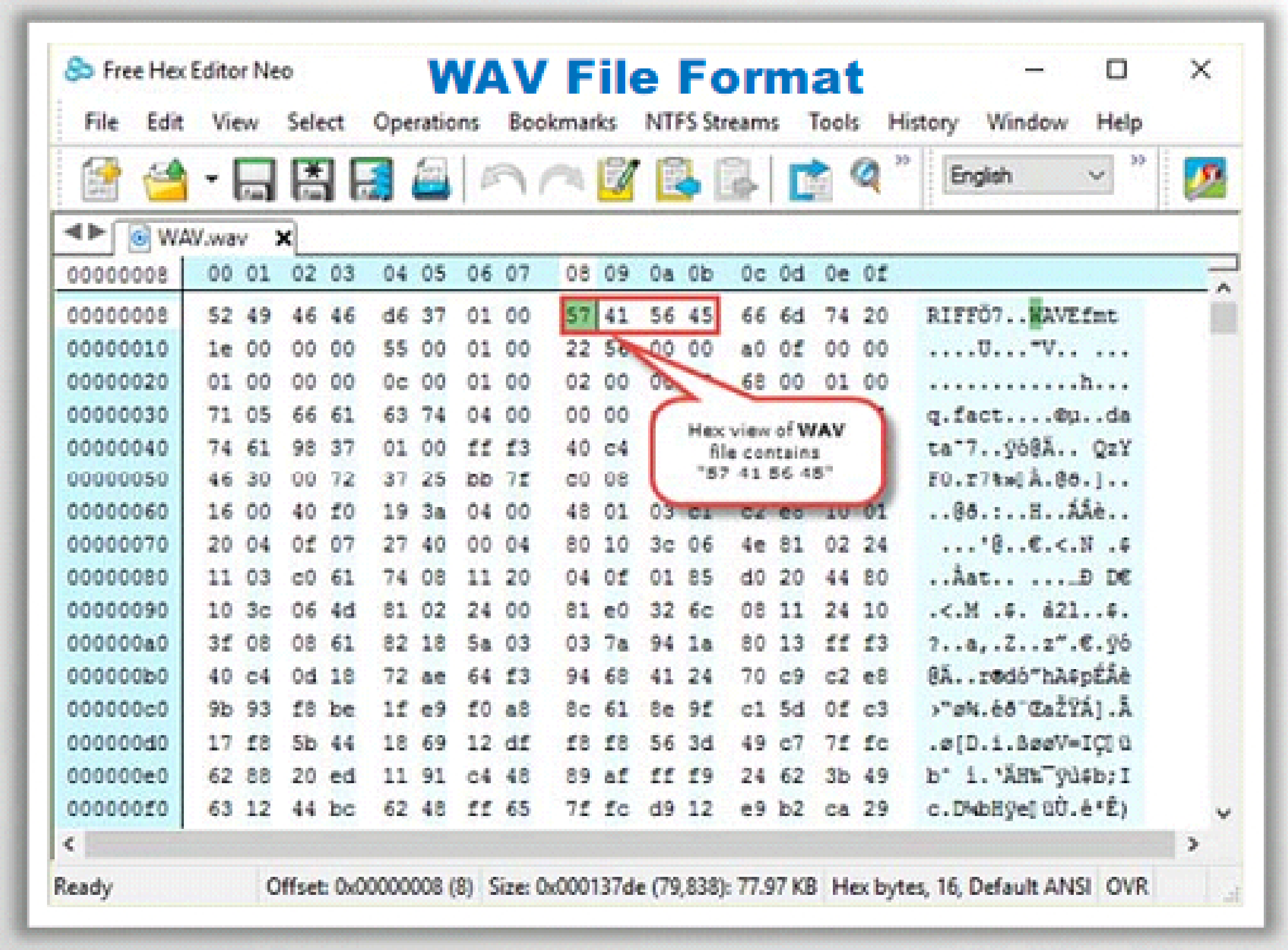


AIFF File Format

```
00000008  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f

00000008  46 4f 52 4d  02 17 71 7a  41 49 46 46  43 4f 4d 4d   FORM..qzAIFFCOMM
00000010  00 00 00 12  00 02 00 85  dc 53 0  10  40 0e bb 80   ........ÜS..@.»€
00000020  00 00 00 00  00 00 53 53  4e 44 02 1   1 4c 00 00   ......SSND..qL..
00000030  00 00 00 00  00 00 ff f3  ff f8 ff                   ......ÿóÿøÿõÿùÿõ
00000040  ff fc ff f2  ff fc ff f5  ff fa ff                   ÿüÿòÿüÿõÿúÿõÿúÿõ
00000050  ff fa ff f6  ff fa ff f6  ff fb ff                   ÿúÿöÿúÿöÿûÿ+ÿúÿ+
00000060  ff ff ff f8  ff fd ff f7  ff fd ff                   ÿÿÿøÿýÿ÷ÿýÿ÷ÿúÿø
00000070  ff fd ff f8  ff fb ff f8  ff fd ff f8  ff fe ff f9   ÿýÿøÿûÿøÿýÿøÿþÿù
00000080  00 00 ff f9  ff fe ff f9  00 00 ff fa  00 00 ff fa   ..ÿùÿþÿù..ÿú..ÿú
00000090  00 00 ff fb  00 00 ff fa  00 00 ff fa  ff fe ff fa   ..ÿû..ÿú..ÿúÿþÿú
000000a0  ff fe ff fa  ff fe ff fb  ff fe ff fb  ff fe ff fc   ÿþÿúÿþÿûÿþÿûÿþÿü
000000b0  00 00 ff fb  ff fe ff f9  ff fe ff fb  00 00 ff fb   ..ÿûÿþÿùÿþÿû..ÿû
000000c0  00 00 ff fd  ff fe ff fc  00 00 ff fc  ff fe ff fc   ..ÿýÿþÿü..ÿüÿþÿü
000000d0  00 00 ff fa  ff fe ff fc  00 00 ff fc  00 00 ff fd   ..ÿúÿþÿü..ÿü..ÿý
000000e0  00 00 ff fd  00 00 ff fb  ff fe ff fb  ff fe ff fd   ..ÿý..ÿûÿþÿûÿþÿý
000000f0  ff fe ff fe  00 00 ff fe  00 00 ff fe  00 00 ff fe   ÿþÿþ..ÿþ..ÿþ..ÿþ
```

Hex view of **AIFF** file contains "41 49 46 46"

Ready | Offset: 0x00000008 (8) | Size: 0x02177182 (35,090,818): 33.47 MB | Hex bytes, 16, Default ANSI | OVR

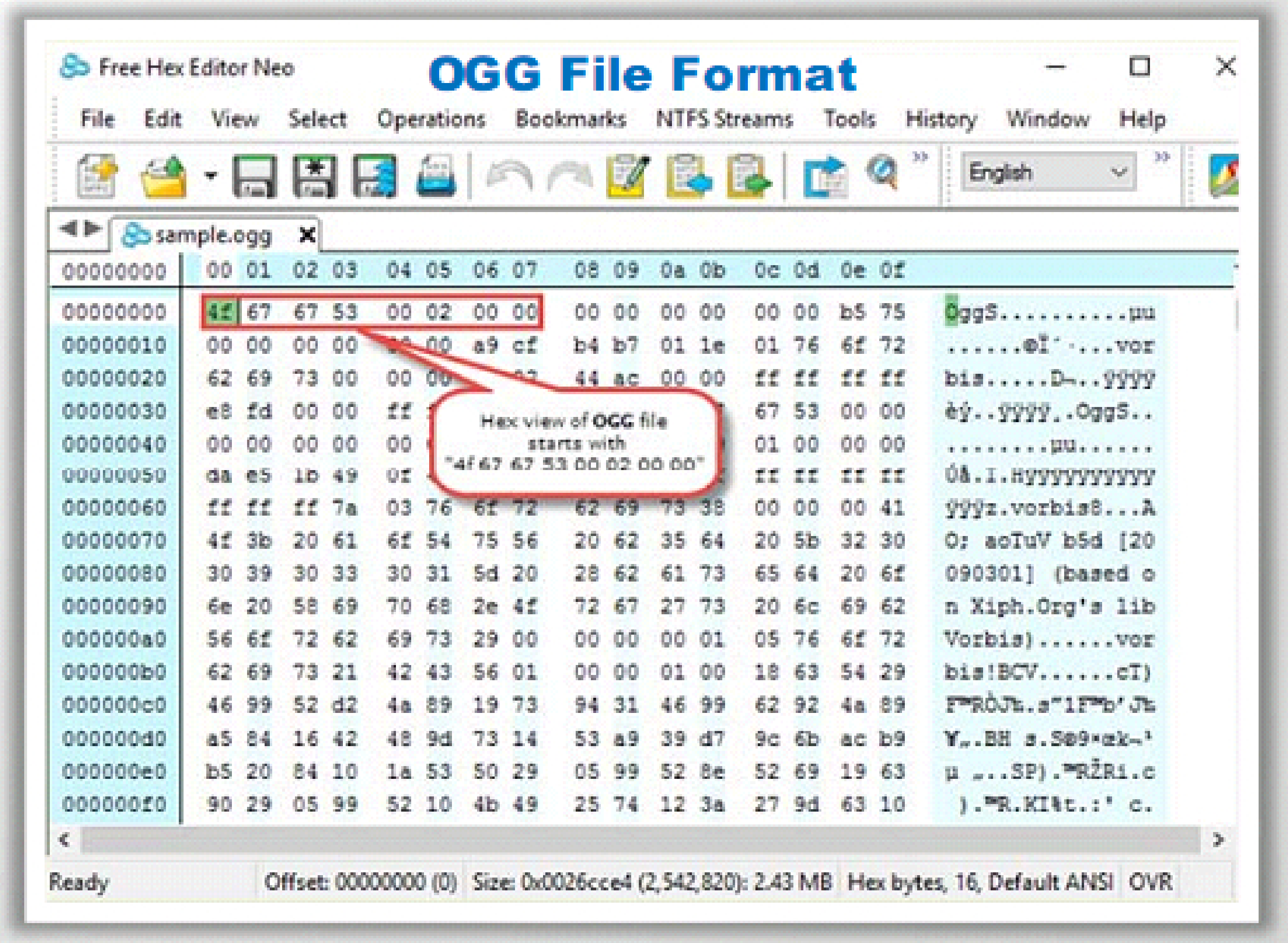


WAV File Format

```
00000008  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f

00000008  52 49 46 46  d6 37 01 00  57 41 56 45  66 6d 74 20   RIFFÖ7..WAVEfmt
00000010  1e 00 00 00  55 00 01 00  22 56 00 00  a0 0f 00 00   ....U..."V.. ...
00000020  01 00 00 00  0c 00 01 00  02 00 0       68 00 01 00   ............h...
00000030  71 05 66 61  63 74 04 00  00 00                      q.fact....@µ..da
00000040  74 61 98 37  01 00 ff f3  40 c4                      ta˜7..ÿó@Ä.. QzY
00000050  46 30 00 72  37 25 bb 7f  c0 08                      F0.r7%»[À.@e.]..
00000060  16 00 40 f0  19 3a 04 00  48 01 03 c1  c2 e5 10 01   ..@ð.:..H..ÁÂå..
00000070  20 04 0f 07  27 40 00 04  80 10 3c 06  4e 81 02 24    ...'@..€.<.N .$
00000080  11 03 c0 61  74 08 11 20  04 0f 01 85  d0 20 44 80   ..Àat.. ....Ð D€
00000090  10 3c 06 4d  81 02 24 00  81 e0 32 6c  08 11 24 10   .<.M .$. à2l..$.
000000a0  3f 08 08 61  82 18 5a 03  03 7a 94 1a  80 13 ff f3   ?..a‚.Z..z".€.ÿó
000000b0  40 c4 0d 18  72 ae 64 f3  94 68 41 24  70 c9 c2 e8   @Ä..r®dó"hA$pÉÂè
000000c0  9b 93 f8 be  1f e9 f0 a8  8c 61 8e 9f  c1 5d 0f c3   ›"ø¾.éð¨ŒaŽŸÁ].Ã
000000d0  17 f8 5b 44  18 69 12 df  f8 f8 56 3d  49 c7 7f fc   .ø[D.i.ßøøV=IÇ.ü
000000e0  62 88 20 ed  11 91 c4 48  89 af ff f9  24 62 3b 49   bˆ í.'ÄH‰¯ÿù$b;I
000000f0  63 12 44 bc  62 48 ff 65  7f fc d9 12  e9 b2 ca 29   c.D¼bHÿe.üÙ.é²Ê)
```

Hex view of **WAV** file contains "57 41 56 45"

Ready | Offset: 0x00000008 (8) | Size: 0x000137de (79,838): 77.97 KB | Hex bytes, 16, Default ANSI | OVR



OGG File Format

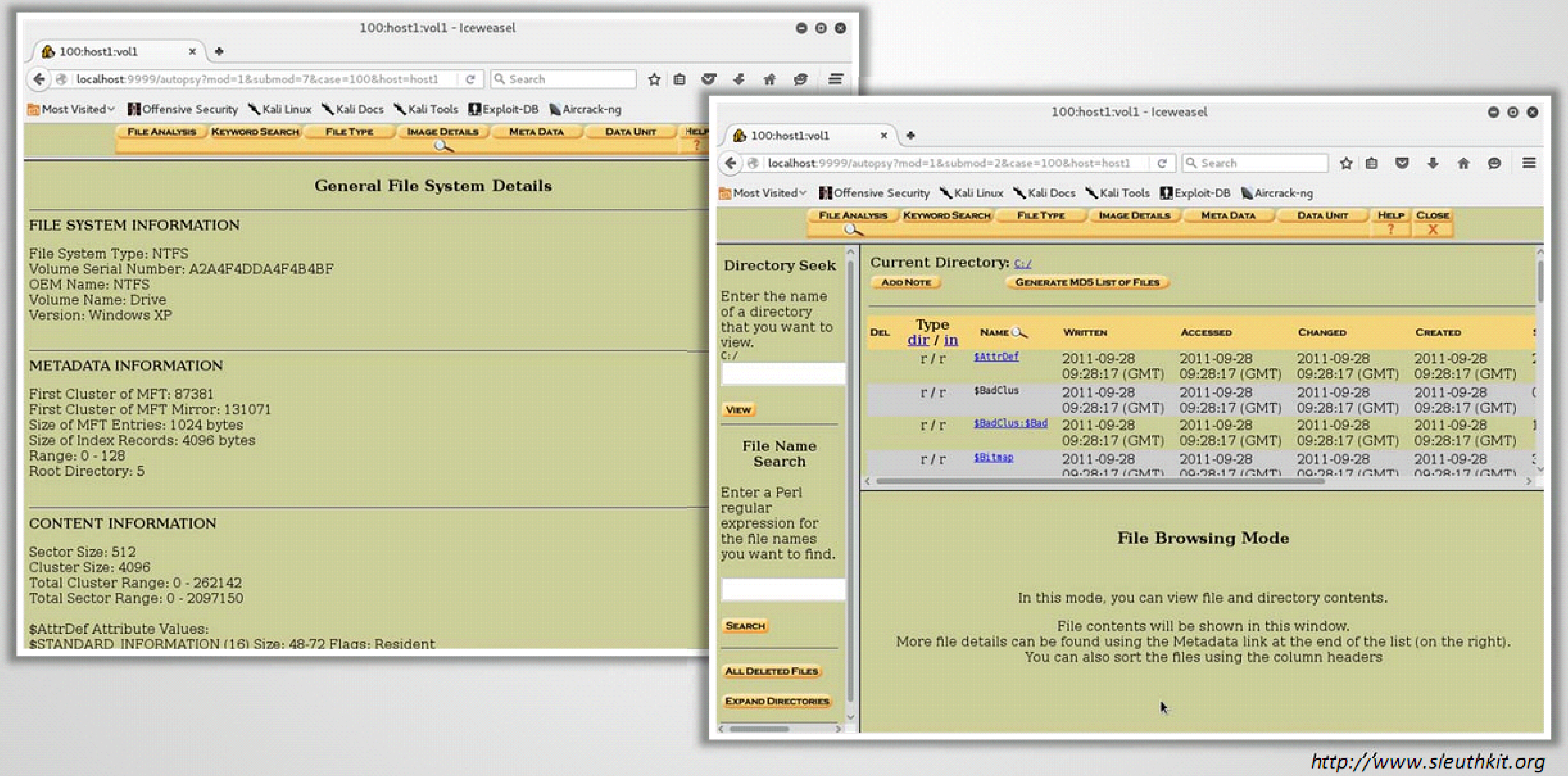
```
00000000  00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f

00000000  4f 67 67 53  00 02 00 00  00 00 00 00  00 00 b5 75   OggS..........µu
00000010  00 00 00 00  00 00 a9 cf  b4 b7 01 1e  01 76 6f 72   ......©Ï´·...vor
00000020  62 69 73 00  00 00     2  44 ac 00 00  ff ff ff ff   bis.....D¬..ÿÿÿÿ
00000030  e8 fd 00 00  ff                        67 53 00 00   èý..ÿÿÿÿ..OggS..
00000040  00 00 00 00  00                        01 00 00 00   ........µu......
00000050  da e5 1b 49  0f                        ff ff ff ff   Úå.I.Hÿÿÿÿÿÿÿÿÿÿ
00000060  ff ff ff 7a  03 76 6f 72  62 69 73 38  00 00 00 41   ÿÿÿz.vorbis8...A
00000070  4f 3b 20 61  6f 54 75 56  20 62 35 64  20 5b 32 30   O; aoTuV b5d [20
00000080  30 39 30 33  30 31 5d 20  28 62 61 73  65 64 20 6f   090301] (based o
00000090  6e 20 58 69  70 68 2e 4f  72 67 27 73  20 6c 69 62   n Xiph.Org's lib
000000a0  56 6f 72 62  69 73 29 00  00 00 00 01  05 76 6f 72   Vorbis)......vor
000000b0  62 69 73 21  42 43 56 01  00 00 01 00  18 63 54 29   bis!BCV......cT)
000000c0  46 99 52 d2  4a 89 19 73  94 31 46 99  62 92 4a 89   F™RÒJ‰.s"1F™b'J‰
000000d0  a5 84 16 42  48 9d 73 14  53 a9 39 d7  9c 6b ac b9   ¥„.BH.s.S©9×œk¬¹
000000e0  b5 20 84 10  1a 53 50 29  05 99 52 8e  52 69 19 63   µ „..SP).™RŽRi.c
000000f0  90 29 05 99  52 10 4b 49  25 74 12 3a  27 9d 63 10   .).™R.KI%t.:'.c.
```

Hex view of **OGG** file starts with "4f 67 67 53 00 02 00 00"

Ready | Offset: 00000000 (0) | Size: 0x0026cce4 (2,542,820): 2.43 MB | Hex bytes, 16, Default ANSI | OVR

File System Analysis Using Autopsy

- Autopsy is a digital forensics platform and **graphical interface to The Sleuth Kit** and other digital forensics tools that can be used to investigate activities which happened on a computer



General File System Details

FILE SYSTEM INFORMATION

File System Type: NTFS
Volume Serial Number: A2A4F4DDA4F4B4BF
OEM Name: NTFS
Volume Name: Drive
Version: Windows XP

METADATA INFORMATION

First Cluster of MFT: 87381
First Cluster of MFT Mirror: 131071
Size of MFT Entries: 1024 bytes
Size of Index Records: 4096 bytes
Range: 0 - 128
Root Directory: 5

CONTENT INFORMATION

Sector Size: 512
Cluster Size: 4096
Total Cluster Range: 0 - 262142
Total Sector Range: 0 - 2097150

$AttrDef Attribute Values:
$STANDARD_INFORMATION (16) Size: 48-72 Flags: Resident

Directory Seek

Enter the name of a directory that you want to view.
C:/

View

File Name Search

Enter a Perl regular expression for the file names you want to find.

Search

All Deleted Files

Expand Directories

Current Directory: C:/

Add Note   Generate MD5 List of Files

Del	Type dir / in	Name	Written	Accessed	Changed	Created
	r / r	$AttrDef	2011-09-28 09:28:17 (GMT)	2011-09-28 09:28:17 (GMT)	2011-09-28 09:28:17 (GMT)	2011-09-28 09:28:17 (GMT)
	r / r	$BadClus	2011-09-28 09:28:17 (GMT)	2011-09-28 09:28:17 (GMT)	2011-09-28 09:28:17 (GMT)	2011-09-28 09:28:17 (GMT)
	r / r	$BadClus:$Bad	2011-09-28 09:28:17 (GMT)	2011-09-28 09:28:17 (GMT)	2011-09-28 09:28:17 (GMT)	2011-09-28 09:28:17 (GMT)
	r / r	$Bitmap	2011-09-28 09:28:17 (GMT)	2011-09-28 09:28:17 (GMT)	2011-09-28 09:28:17 (GMT)	2011-09-28 09:28:17 (GMT)

File Browsing Mode

In this mode, you can view file and directory contents.

File contents will be shown in this window.
More file details can be found using the Metadata link at the end of the list (on the right).
You can also sort the files using the column headers

http://www.sleuthkit.org

File System Analysis Using The Sleuth Kit (TSK)

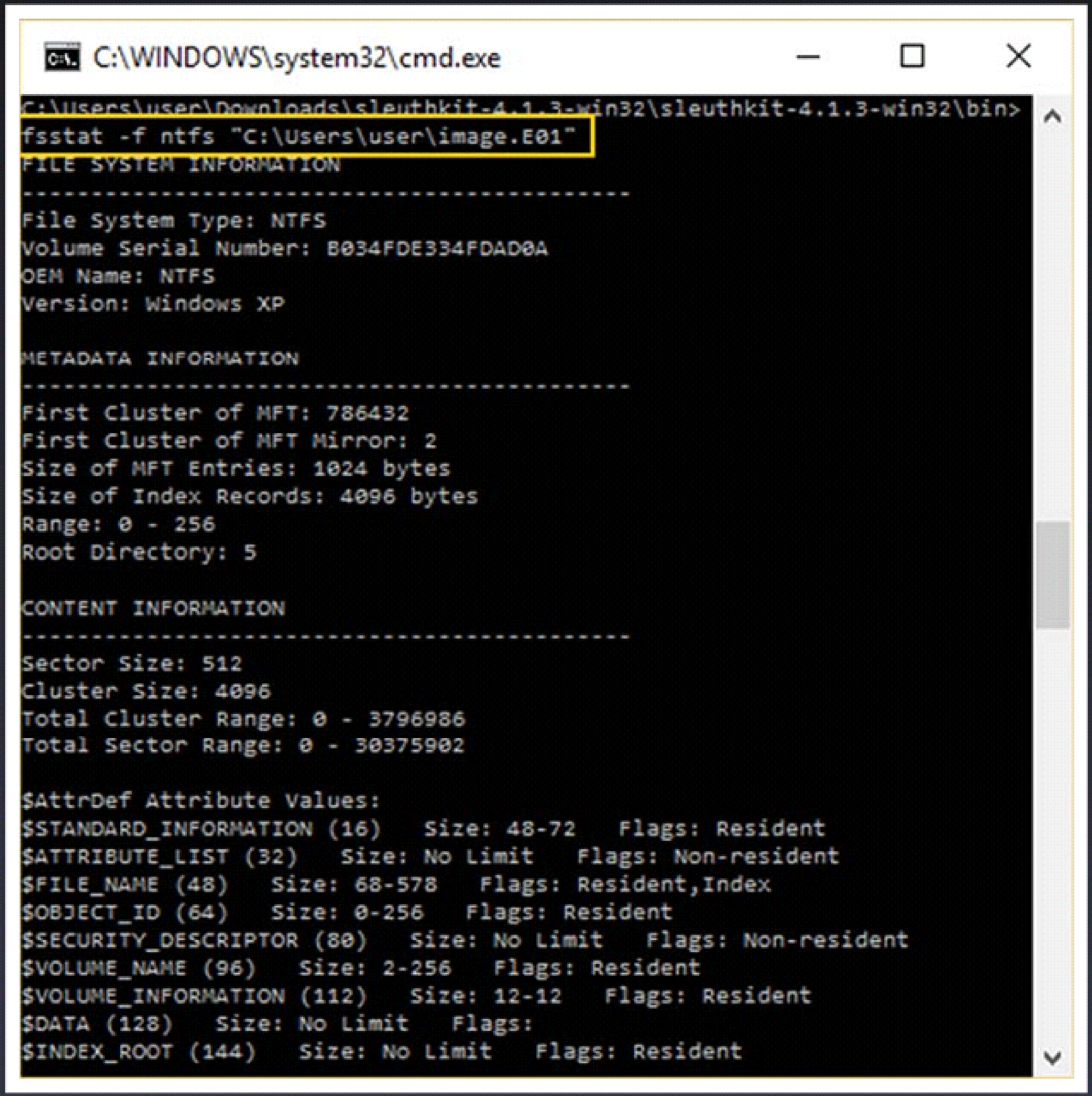
1. The Sleuth Kit (TSK) is a library and a collection of command line tools that allow to investigate volume and file system data
2. The file system tools allow to examine file systems of a suspect computer in a non-intrusive fashion
3. The volume system (media management) tools allow to examine the layout of disks and other media
4. It supports DOS partitions, BSD partitions (disk labels), Mac partitions, Sun slices (Volume Table of Contents), and GPT disks
5. It analyzes raw (i.e. dd), Expert Witness (i.e. EnCase) and AFF file systems and disk images
6. It supports the NTFS, FAT, ExFAT, UFS 1, UFS 2, EXT2FS, EXT3FS, EXT4, HFS, ISO 9660, and YAFFS2 file systems

http://www.sleuthkit.org

Note: To perform analysis, create a forensics image .dd or .E01 of hard disk or pen drive using disk imaging tools. Here, we have created forensics image of a pen drive (.E01 format) using AccessData FTK Imager.

The Sleuth Kit (TSK): fsstat

The **fsstat** tool displays the **file system category data** for a file system

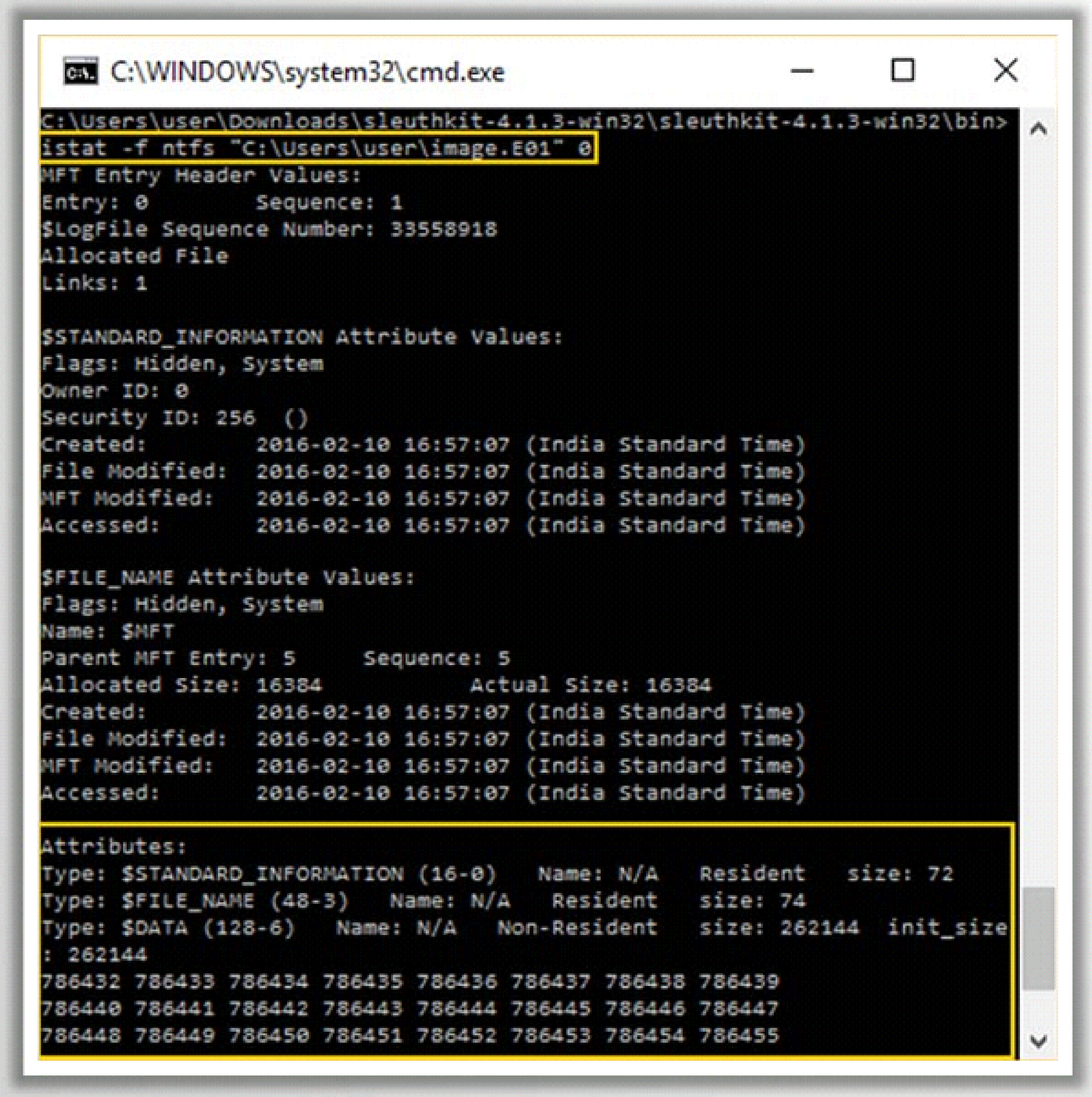


```
C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
fsstat -f ntfs "C:\Users\user\image.E01"
FILE SYSTEM INFORMATION
--------------------------------------------
File System Type: NTFS
Volume Serial Number: B034FDE334FDAD0A
OEM Name: NTFS
Version: Windows XP

METADATA INFORMATION
--------------------------------------------
First Cluster of MFT: 786432
First Cluster of MFT Mirror: 2
Size of MFT Entries: 1024 bytes
Size of Index Records: 4096 bytes
Range: 0 - 256
Root Directory: 5

CONTENT INFORMATION
--------------------------------------------
Sector Size: 512
Cluster Size: 4096
Total Cluster Range: 0 - 3796986
Total Sector Range: 0 - 30375902

$AttrDef Attribute Values:
$STANDARD_INFORMATION (16)   Size: 48-72   Flags: Resident
$ATTRIBUTE_LIST (32)   Size: No Limit   Flags: Non-resident
$FILE_NAME (48)   Size: 68-578   Flags: Resident,Index
$OBJECT_ID (64)   Size: 0-256   Flags: Resident
$SECURITY_DESCRIPTOR (80)   Size: No Limit   Flags: Non-resident
$VOLUME_NAME (96)   Size: 2-256   Flags: Resident
$VOLUME_INFORMATION (112)   Size: 12-12   Flags: Resident
$DATA (128)   Size: No Limit   Flags:
$INDEX_ROOT (144)   Size: No Limit   Flags: Resident
```

The Sleuth Kit (TSK): istat (1 of 4)

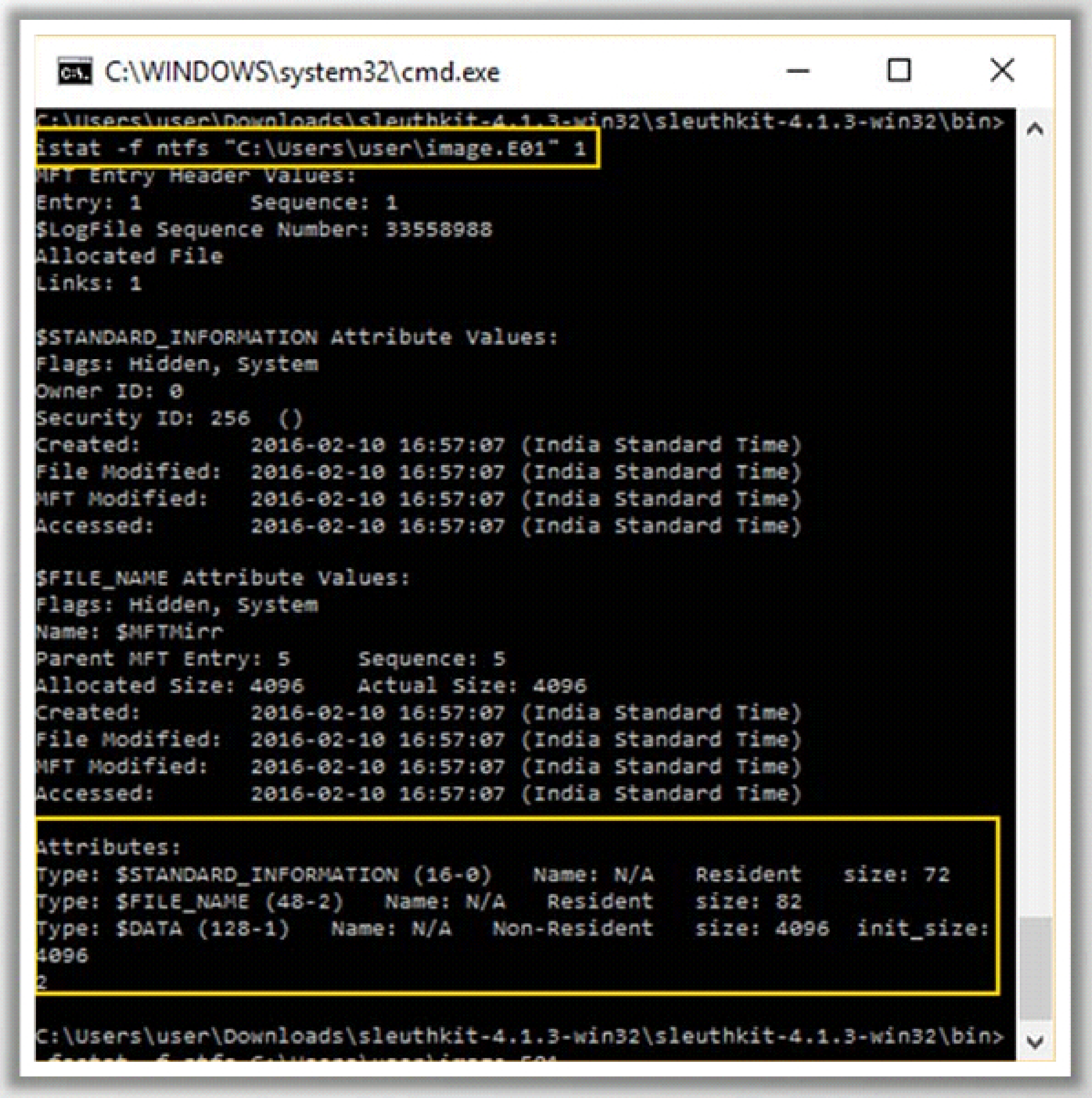
- The istat tool in TSK shows the **details of a directory entry** and its output for a given entry

MFT File Overview



```
C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
istat -f ntfs "C:\Users\user\image.E01" 0
MFT Entry Header Values:
Entry: 0        Sequence: 1
$LogFile Sequence Number: 33558918
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Hidden, System
Owner ID: 0
Security ID: 256  ()
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

$FILE_NAME Attribute Values:
Flags: Hidden, System
Name: $MFT
Parent MFT Entry: 5     Sequence: 5
Allocated Size: 16384           Actual Size: 16384
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

Attributes:
Type: $STANDARD_INFORMATION (16-0)   Name: N/A   Resident   size: 72
Type: $FILE_NAME (48-3)   Name: N/A   Resident   size: 74
Type: $DATA (128-6)   Name: N/A   Non-Resident   size: 262144  init_size
: 262144
786432 786433 786434 786435 786436 786437 786438 786439
786440 786441 786442 786443 786444 786445 786446 786447
786448 786449 786450 786451 786452 786453 786454 786455
```

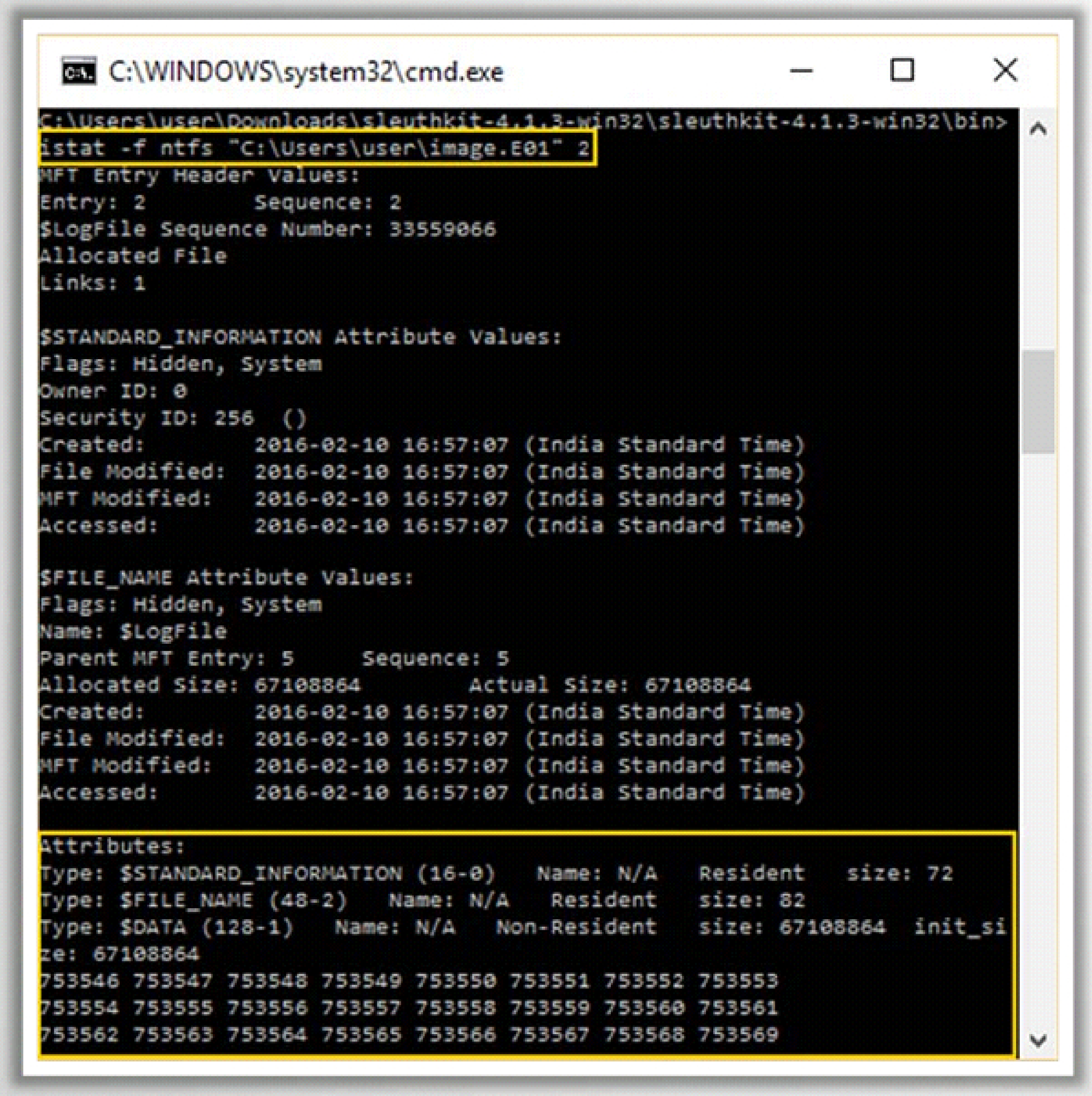
MFTMirr File Overview



```
C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
istat -f ntfs "C:\Users\user\image.E01" 1
MFT Entry Header Values:
Entry: 1        Sequence: 1
$LogFile Sequence Number: 33558988
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Hidden, System
Owner ID: 0
Security ID: 256  ()
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

$FILE_NAME Attribute Values:
Flags: Hidden, System
Name: $MFTMirr
Parent MFT Entry: 5     Sequence: 5
Allocated Size: 4096    Actual Size: 4096
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

Attributes:
Type: $STANDARD_INFORMATION (16-0)   Name: N/A   Resident   size: 72
Type: $FILE_NAME (48-2)   Name: N/A   Resident   size: 82
Type: $DATA (128-1)   Name: N/A   Non-Resident   size: 4096  init_size:
4096
2

C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
```

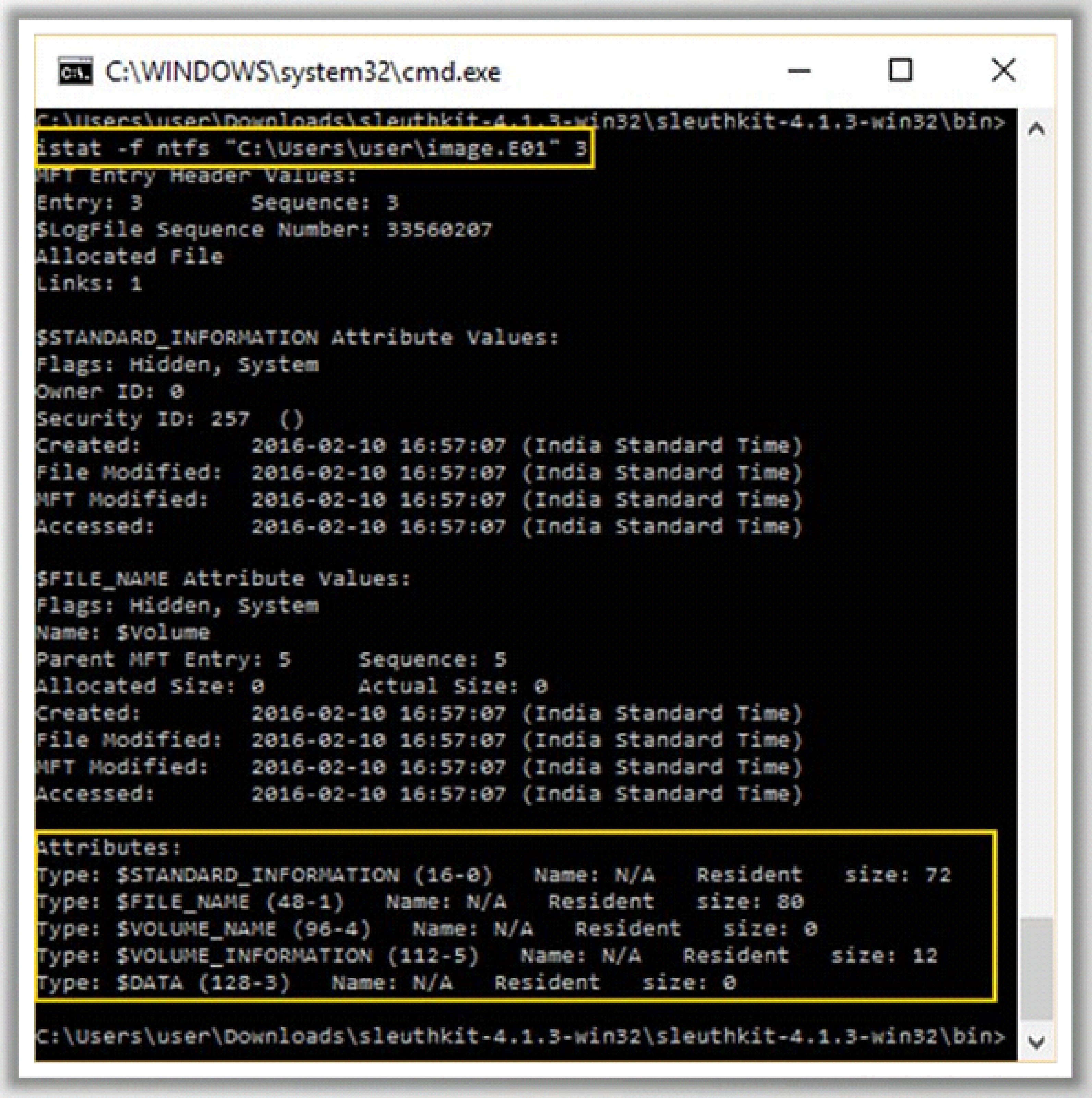
The Sleuth Kit (TSK): istat (2 of 4)

LogFile Overview



```
C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
istat -f ntfs "C:\Users\user\image.E01" 2
MFT Entry Header Values:
Entry: 2        Sequence: 2
$LogFile Sequence Number: 33559066
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Hidden, System
Owner ID: 0
Security ID: 256  ()
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

$FILE_NAME Attribute Values:
Flags: Hidden, System
Name: $LogFile
Parent MFT Entry: 5     Sequence: 5
Allocated Size: 67108864        Actual Size: 67108864
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

Attributes:
Type: $STANDARD_INFORMATION (16-0)   Name: N/A   Resident   size: 72
Type: $FILE_NAME (48-2)   Name: N/A   Resident   size: 82
Type: $DATA (128-1)   Name: N/A   Non-Resident   size: 67108864  init_si
ze: 67108864
753546 753547 753548 753549 753550 753551 753552 753553
753554 753555 753556 753557 753558 753559 753560 753561
753562 753563 753564 753565 753566 753567 753568 753569
```

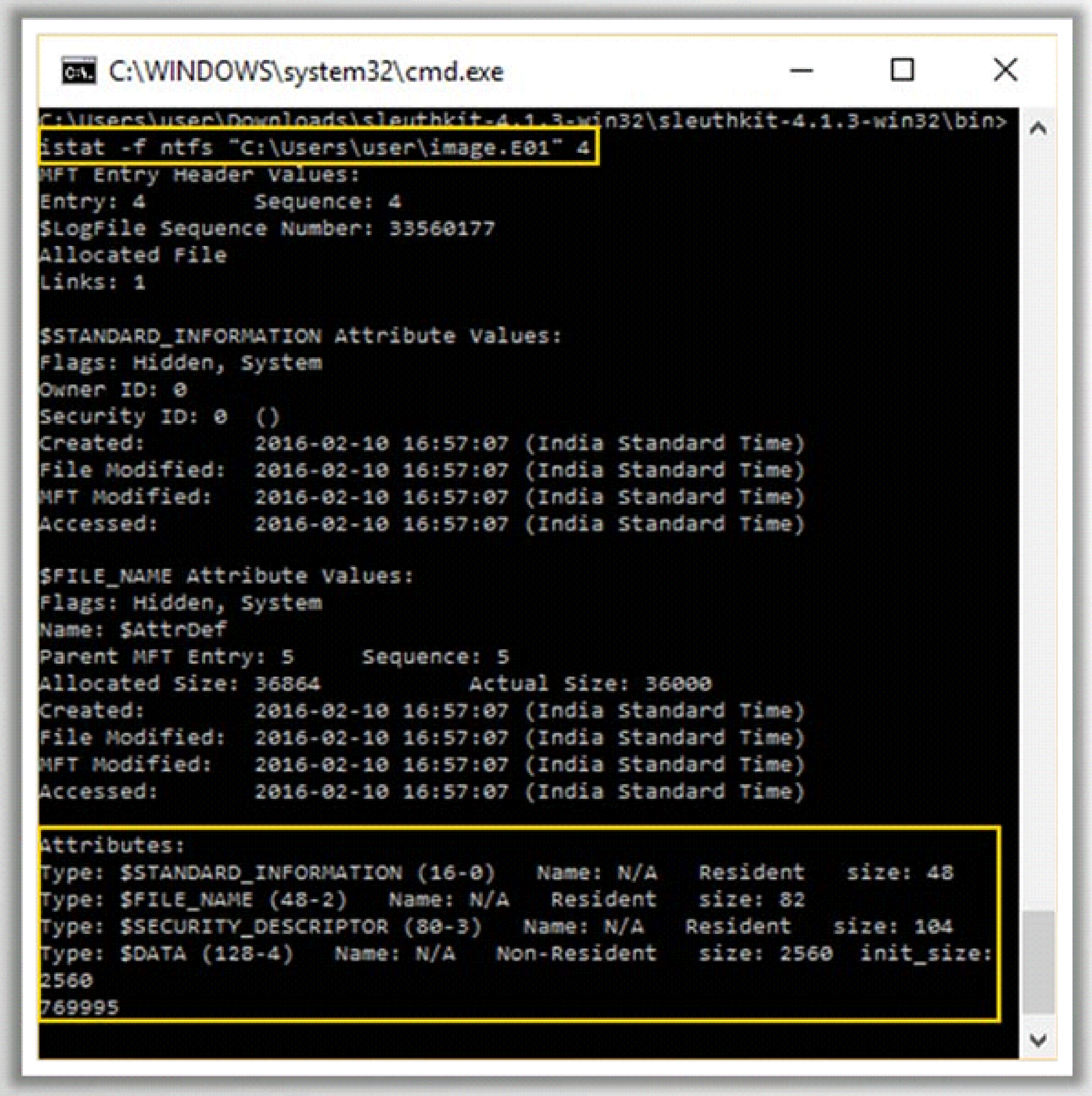
Volume File Overview



```
C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
istat -f ntfs "C:\Users\user\image.E01" 3
MFT Entry Header Values:
Entry: 3        Sequence: 3
$LogFile Sequence Number: 33560207
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Hidden, System
Owner ID: 0
Security ID: 257  ()
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

$FILE_NAME Attribute Values:
Flags: Hidden, System
Name: $Volume
Parent MFT Entry: 5     Sequence: 5
Allocated Size: 0       Actual Size: 0
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

Attributes:
Type: $STANDARD_INFORMATION (16-0)   Name: N/A   Resident   size: 72
Type: $FILE_NAME (48-1)   Name: N/A   Resident   size: 80
Type: $VOLUME_NAME (96-4)   Name: N/A   Resident   size: 0
Type: $VOLUME_INFORMATION (112-5)   Name: N/A   Resident   size: 12
Type: $DATA (128-3)   Name: N/A   Resident   size: 0

C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
```

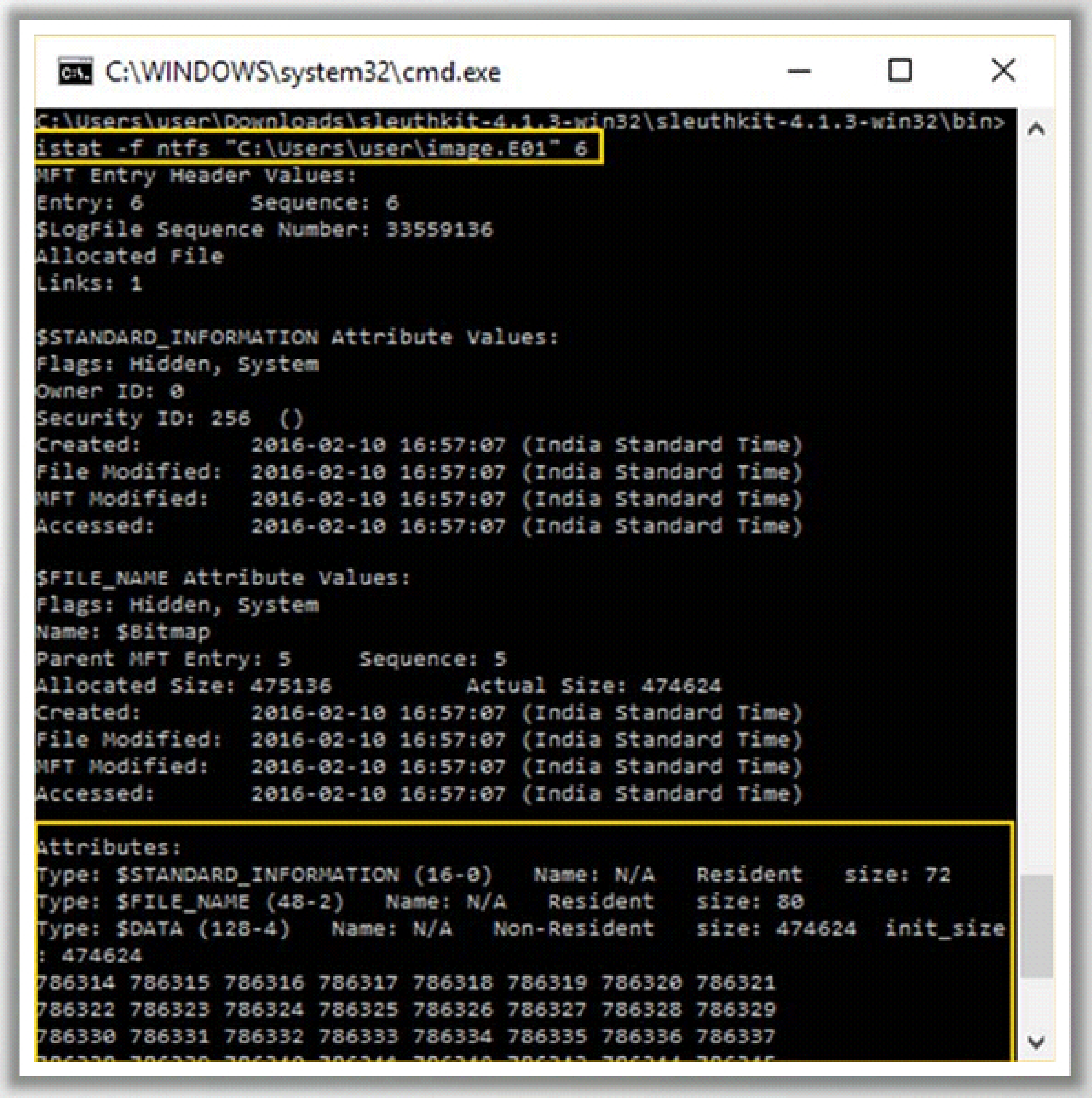
The Sleuth Kit (TSK): istat (3 of 4)

AttrDef File Overview



```
C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
istat -f ntfs "C:\Users\user\image.E01" 4
MFT Entry Header Values:
Entry: 4        Sequence: 4
$LogFile Sequence Number: 33560177
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Hidden, System
Owner ID: 0
Security ID: 0  ()
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

$FILE_NAME Attribute Values:
Flags: Hidden, System
Name: $AttrDef
Parent MFT Entry: 5     Sequence: 5
Allocated Size: 36864           Actual Size: 36000
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

Attributes:
Type: $STANDARD_INFORMATION (16-0)   Name: N/A   Resident   size: 48
Type: $FILE_NAME (48-2)   Name: N/A   Resident   size: 82
Type: $SECURITY_DESCRIPTOR (80-3)   Name: N/A   Resident   size: 104
Type: $DATA (128-4)   Name: N/A   Non-Resident   size: 2560  init_size:
2560
769995
```

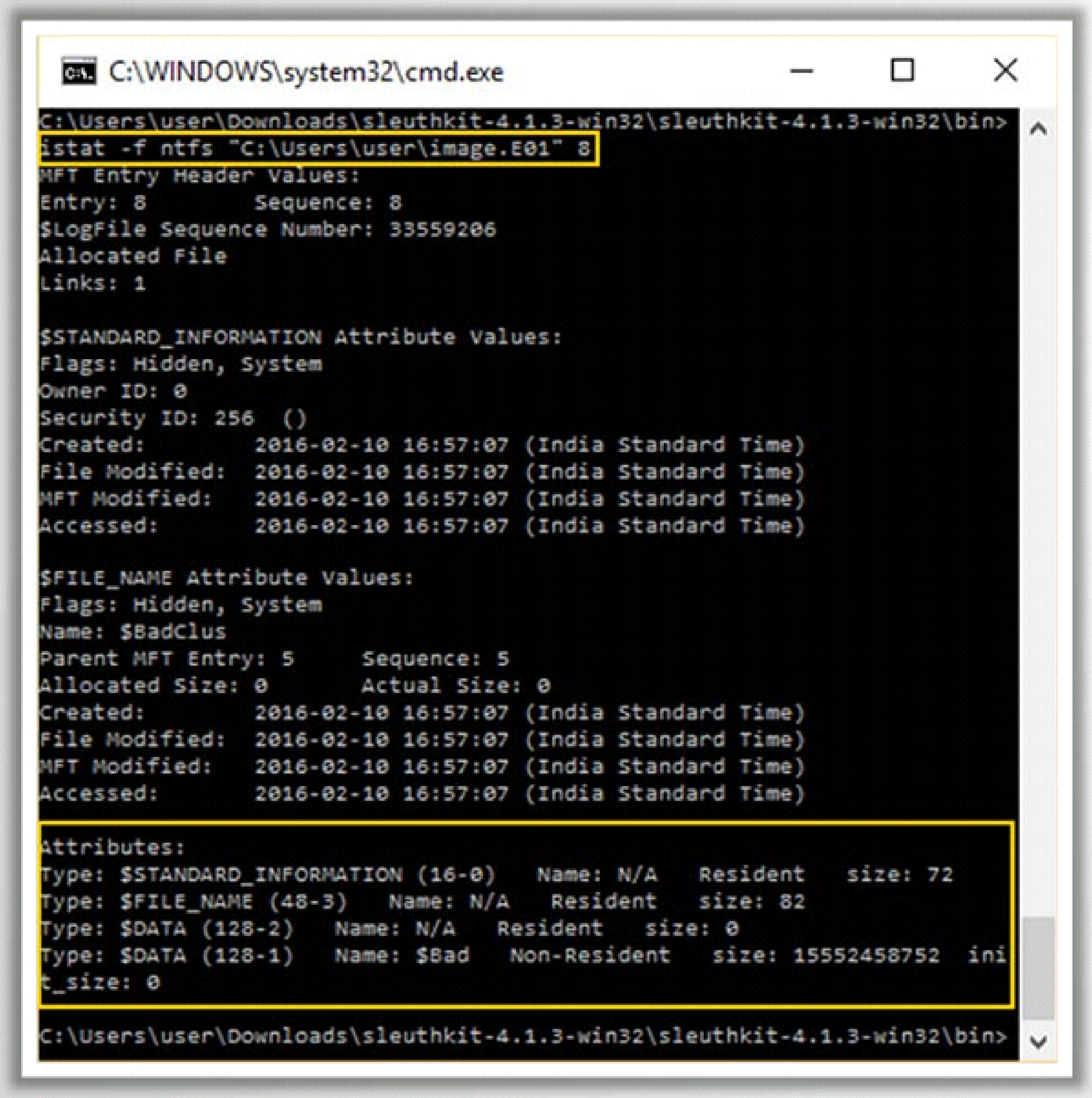
Bitmap File Overview



```
C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
istat -f ntfs "C:\Users\user\image.E01" 6
MFT Entry Header Values:
Entry: 6        Sequence: 6
$LogFile Sequence Number: 33559136
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Hidden, System
Owner ID: 0
Security ID: 256  ()
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

$FILE_NAME Attribute Values:
Flags: Hidden, System
Name: $Bitmap
Parent MFT Entry: 5     Sequence: 5
Allocated Size: 475136          Actual Size: 474624
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

Attributes:
Type: $STANDARD_INFORMATION (16-0)   Name: N/A   Resident   size: 72
Type: $FILE_NAME (48-2)   Name: N/A   Resident   size: 80
Type: $DATA (128-4)   Name: N/A   Non-Resident   size: 474624  init_size
: 474624
786314 786315 786316 786317 786318 786319 786320 786321
786322 786323 786324 786325 786326 786327 786328 786329
786330 786331 786332 786333 786334 786335 786336 786337
```

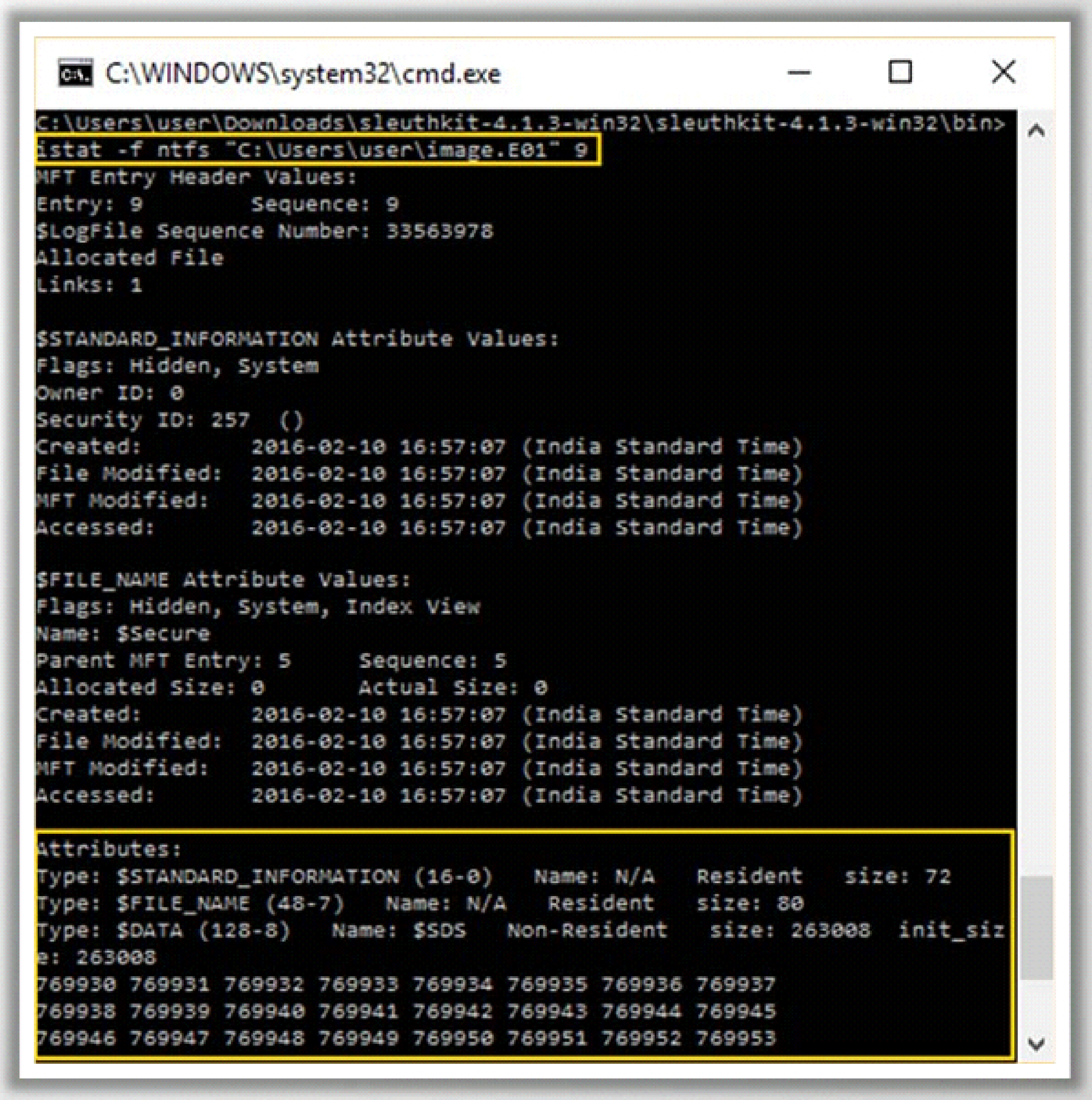
The Sleuth Kit (TSK): istat (4 of 4)

BadClus File Overview



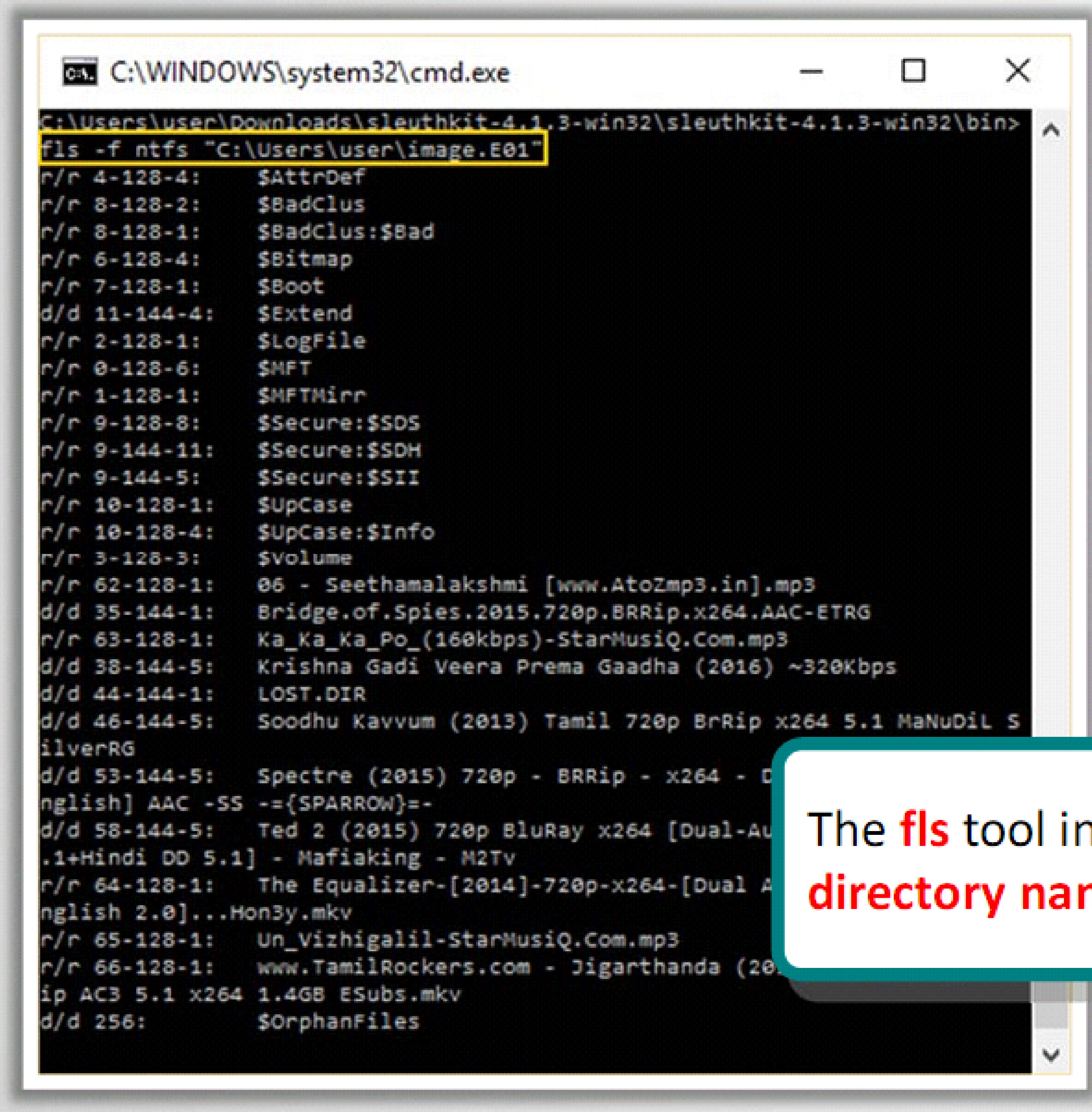
```
C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
istat -f ntfs "C:\Users\user\image.E01" 8
MFT Entry Header Values:
Entry: 8        Sequence: 8
$LogFile Sequence Number: 33559206
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Hidden, System
Owner ID: 0
Security ID: 256  ()
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

$FILE_NAME Attribute Values:
Flags: Hidden, System
Name: $BadClus
Parent MFT Entry: 5     Sequence: 5
Allocated Size: 0       Actual Size: 0
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

Attributes:
Type: $STANDARD_INFORMATION (16-0)   Name: N/A   Resident   size: 72
Type: $FILE_NAME (48-3)   Name: N/A   Resident   size: 82
Type: $DATA (128-2)   Name: N/A   Resident   size: 0
Type: $DATA (128-1)   Name: $Bad   Non-Resident   size: 15552458752  ini
t_size: 0

C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
```

Secure File Overview



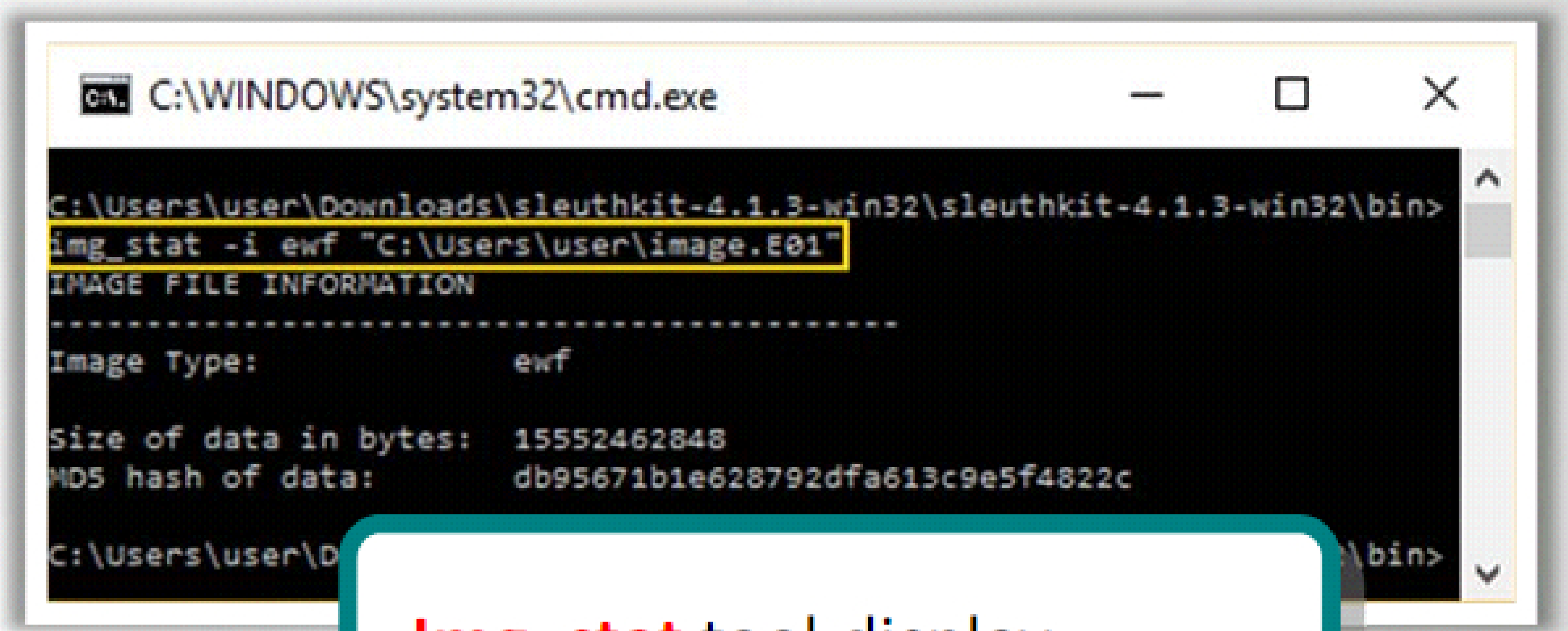
```
C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
istat -f ntfs "C:\Users\user\image.E01" 9
MFT Entry Header Values:
Entry: 9        Sequence: 9
$LogFile Sequence Number: 33563978
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Hidden, System
Owner ID: 0
Security ID: 257  ()
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

$FILE_NAME Attribute Values:
Flags: Hidden, System, Index View
Name: $Secure
Parent MFT Entry: 5     Sequence: 5
Allocated Size: 0       Actual Size: 0
Created:        2016-02-10 16:57:07 (India Standard Time)
File Modified:  2016-02-10 16:57:07 (India Standard Time)
MFT Modified:   2016-02-10 16:57:07 (India Standard Time)
Accessed:       2016-02-10 16:57:07 (India Standard Time)

Attributes:
Type: $STANDARD_INFORMATION (16-0)   Name: N/A   Resident   size: 72
Type: $FILE_NAME (48-7)   Name: N/A   Resident   size: 80
Type: $DATA (128-8)   Name: $SDS   Non-Resident   size: 263008  init_siz
e: 263008
769930 769931 769932 769933 769934 769935 769936 769937
769938 769939 769940 769941 769942 769943 769944 769945
769946 769947 769948 769949 769950 769951 769952 769953
```

The Sleuth Kit (TSK): fls and img_stat



```
C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
fls -f ntfs "C:\Users\user\image.E01"
r/r 4-128-4:    $AttrDef
r/r 8-128-2:    $BadClus
r/r 8-128-1:    $BadClus:$Bad
r/r 6-128-4:    $Bitmap
r/r 7-128-1:    $Boot
d/d 11-144-4:   $Extend
r/r 2-128-1:    $LogFile
r/r 0-128-6:    $MFT
r/r 1-128-1:    $MFTMirr
r/r 9-128-8:    $Secure:$SDS
r/r 9-144-11:   $Secure:$SDH
r/r 9-144-5:    $Secure:$SII
r/r 10-128-1:   $UpCase
r/r 10-128-4:   $UpCase:$Info
r/r 3-128-3:    $Volume
r/r 62-128-1:   06 - Seethamalakshmi [www.AtoZmp3.in].mp3
d/d 35-144-1:   Bridge.of.Spies.2015.720p.BRRip.x264.AAC-ETRG
r/r 63-128-1:   Ka_Ka_Ka_Po_(160kbps)-StarMusiQ.Com.mp3
d/d 38-144-5:   Krishna Gadi Veera Prema Gaadha (2016) ~320Kbps
d/d 44-144-1:   LOST.DIR
d/d 46-144-5:   Soodhu Kavvum (2013) Tamil 720p BrRip x264 5.1 MaNuDiL S
ilverRG
d/d 53-144-5:   Spectre (2015) 720p - BRRip - x264 - D
nglish] AAC -SS -={SPARROW}=-
d/d 58-144-5:   Ted 2 (2015) 720p BluRay x264 [Dual-Au
.1+Hindi DD 5.1] - Mafiaking - M2Tv
r/r 64-128-1:   The Equalizer-[2014]-720p-x264-[Dual A
nglish 2.0]...Hon3y.mkv
r/r 65-128-1:   Un_Vizhigalil-StarMusiQ.Com.mp3
r/r 66-128-1:   www.TamilRockers.com - Jigarthanda (20
ip AC3 5.1 x264 1.4GB ESubs.mkv
d/d 256:        $OrphanFiles
```

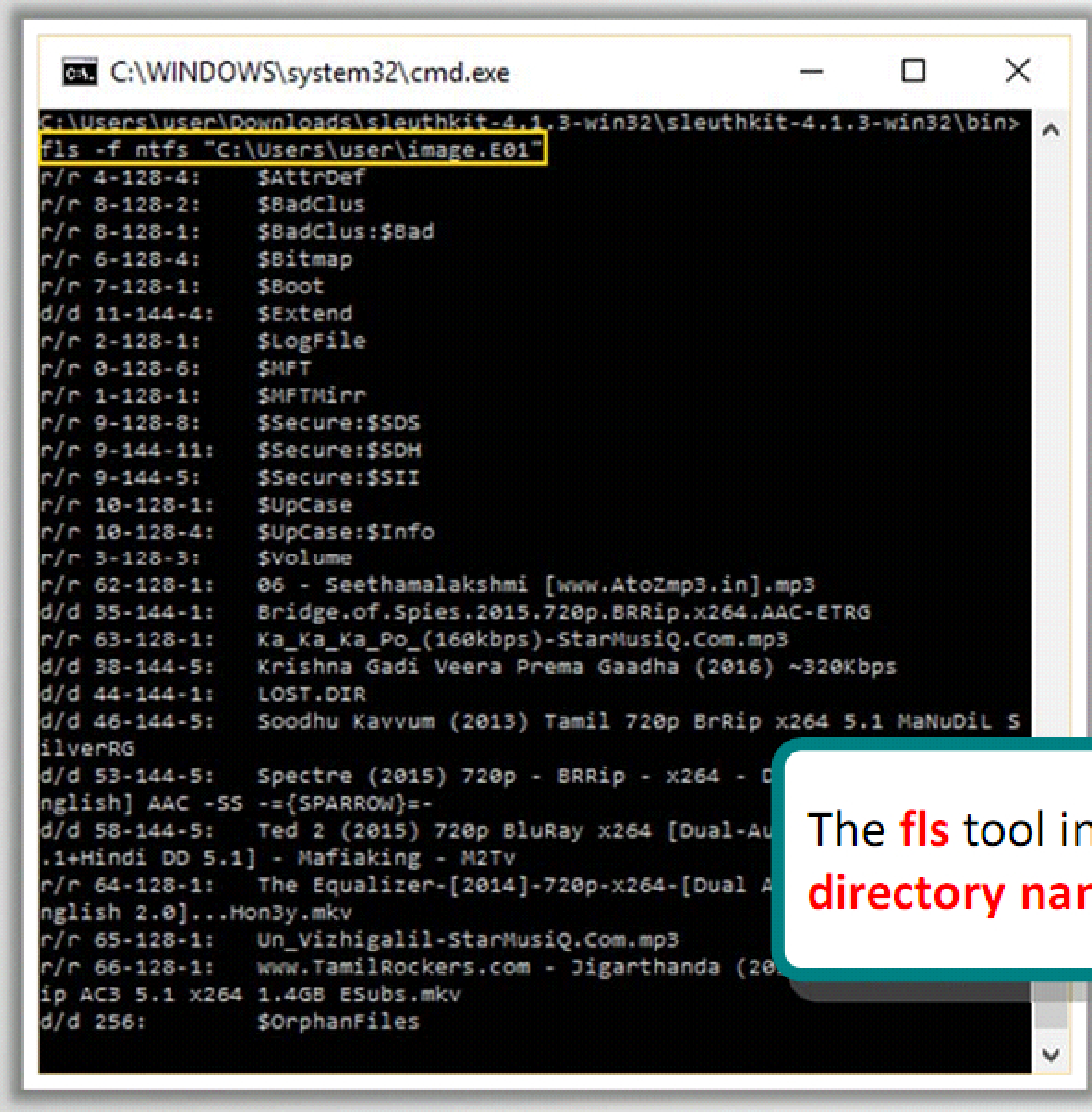
The **fls** tool in TSK **list file and directory names** in a disk image



```
C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
img_stat -i ewf "C:\Users\user\image.E01"
IMAGE FILE INFORMATION
--------------------------------------------
Image Type:             ewf

Size of data in bytes:  15552462848
MD5 hash of data:       db95671b1e628792dfa613c9e5f4822c

C:\Users\user\D                                              \bin>
```

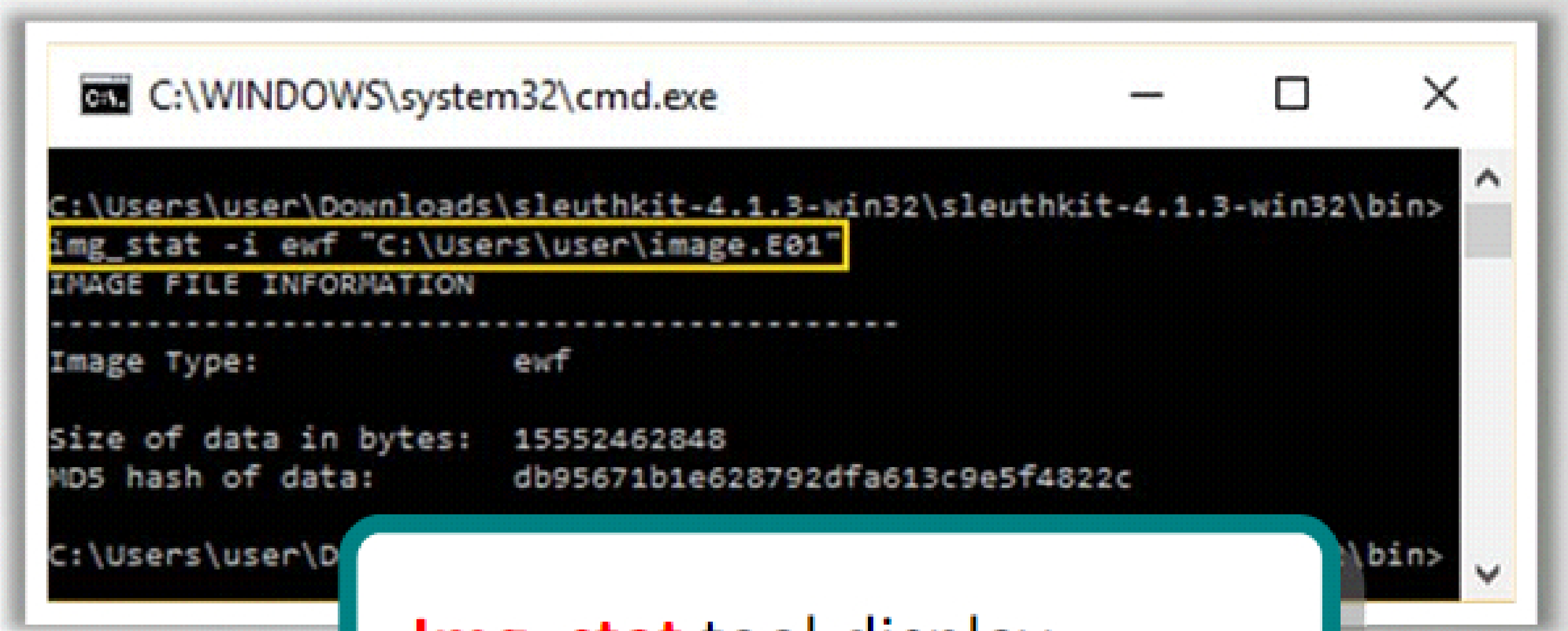
Img_stat tool display details of an **image file**

The Sleuth Kit (TSK): fls and img_stat



```
C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
fls -f ntfs "C:\Users\user\image.E01"
r/r 4-128-4:    $AttrDef
r/r 8-128-2:    $BadClus
r/r 8-128-1:    $BadClus:$Bad
r/r 6-128-4:    $Bitmap
r/r 7-128-1:    $Boot
d/d 11-144-4:   $Extend
r/r 2-128-1:    $LogFile
r/r 0-128-6:    $MFT
r/r 1-128-1:    $MFTMirr
r/r 9-128-8:    $Secure:$SDS
r/r 9-144-11:   $Secure:$SDH
r/r 9-144-5:    $Secure:$SII
r/r 10-128-1:   $UpCase
r/r 10-128-4:   $UpCase:$Info
r/r 3-128-3:    $Volume
r/r 62-128-1:   06 - Seethamalakshmi [www.AtoZmp3.in].mp3
d/d 35-144-1:   Bridge.of.Spies.2015.720p.BRRip.x264.AAC-ETRG
r/r 63-128-1:   Ka_Ka_Ka_Po_(160kbps)-StarMusiQ.Com.mp3
d/d 38-144-5:   Krishna Gadi Veera Prema Gaadha (2016) ~320Kbps
d/d 44-144-1:   LOST.DIR
d/d 46-144-5:   Soodhu Kavvum (2013) Tamil 720p BrRip x264 5.1 MaNuDiL S
ilverRG
d/d 53-144-5:   Spectre (2015) 720p - BRRip - x264 - D
nglish] AAC -SS -={SPARROW}=-
d/d 58-144-5:   Ted 2 (2015) 720p BluRay x264 [Dual-Au
.1+Hindi DD 5.1] - Mafiaking - M2Tv
r/r 64-128-1:   The Equalizer-[2014]-720p-x264-[Dual A
nglish 2.0]...Hon3y.mkv
r/r 65-128-1:   Un_Vizhigalil-StarMusiQ.Com.mp3
r/r 66-128-1:   www.TamilRockers.com - Jigarthanda (20
ip AC3 5.1 x264 1.4GB ESubs.mkv
d/d 256:        $OrphanFiles
```

The fls tool in TSK list file and directory names in a disk image



```
C:\Users\user\Downloads\sleuthkit-4.1.3-win32\sleuthkit-4.1.3-win32\bin>
img_stat -i ewf "C:\Users\user\image.E01"
IMAGE FILE INFORMATION
--------------------------------------------
Image Type:             ewf

Size of data in bytes:  15552462848
MD5 hash of data:       db95671b1e628792dfa613c9e5f4822c

C:\Users\user\D                                                    \bin>
```

Img_stat tool display details of an image file