Guide to Computer Forensics and Investigations Fifth Edition

Chapter 5
Working with Windows and CLI Systems

Objectives

- Explain the purpose and structure of file systems
- Describe Microsoft file structures
- Explain the structure of NTFS disks
- List some options for decrypting drives encrypted with whole disk encryption
- Explain how the Windows Registry works
- Describe Microsoft startup tasks
- Explain the purpose of a virtual machine

Understanding File Systems

- **File system**
  - Gives OS a road map to data on a disk
- Type of file system an OS uses determines how data is stored on the disk
- When you need to access a suspect's computer to acquire or inspect data
  - You should be familiar with both the computer's OS and file systems

Understanding the Boot Sequence

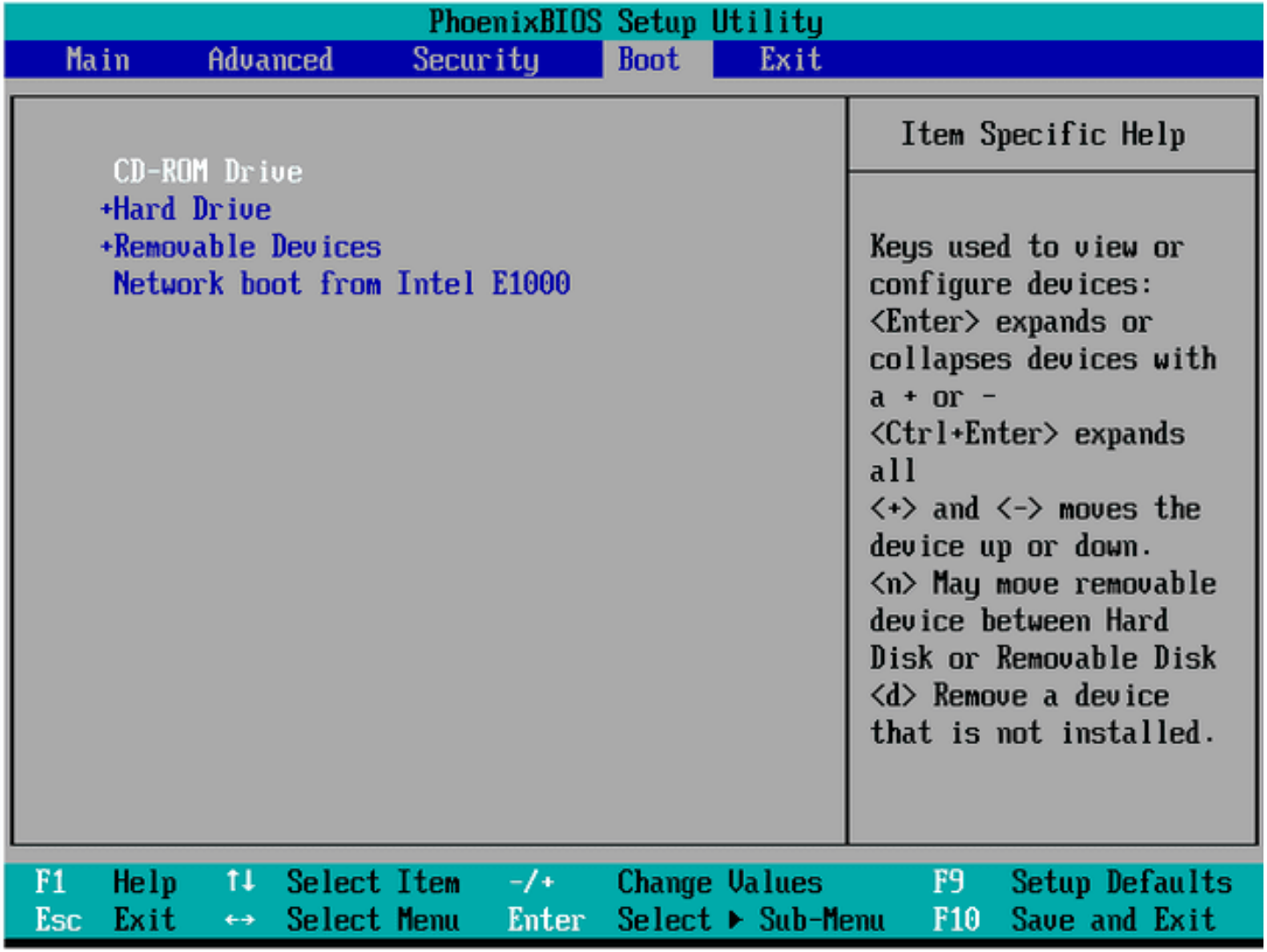
- Complementary Metal Oxide Semiconductor (CMOS)
  - Computer stores system configuration and date and time information in the CMOS
    - When power to the system is off
- Basic Input/Output System (BIOS) or Extensible Firmware Interface (EFI)
  - BIOS is designed for x86 computers and uses Master Boot Records (MBR)
  - EFI is designed for x64 computers and uses GUID Partition Table (GPT)
  - Contains programs that perform input and output at the hardware level

Understanding the Boot Sequence

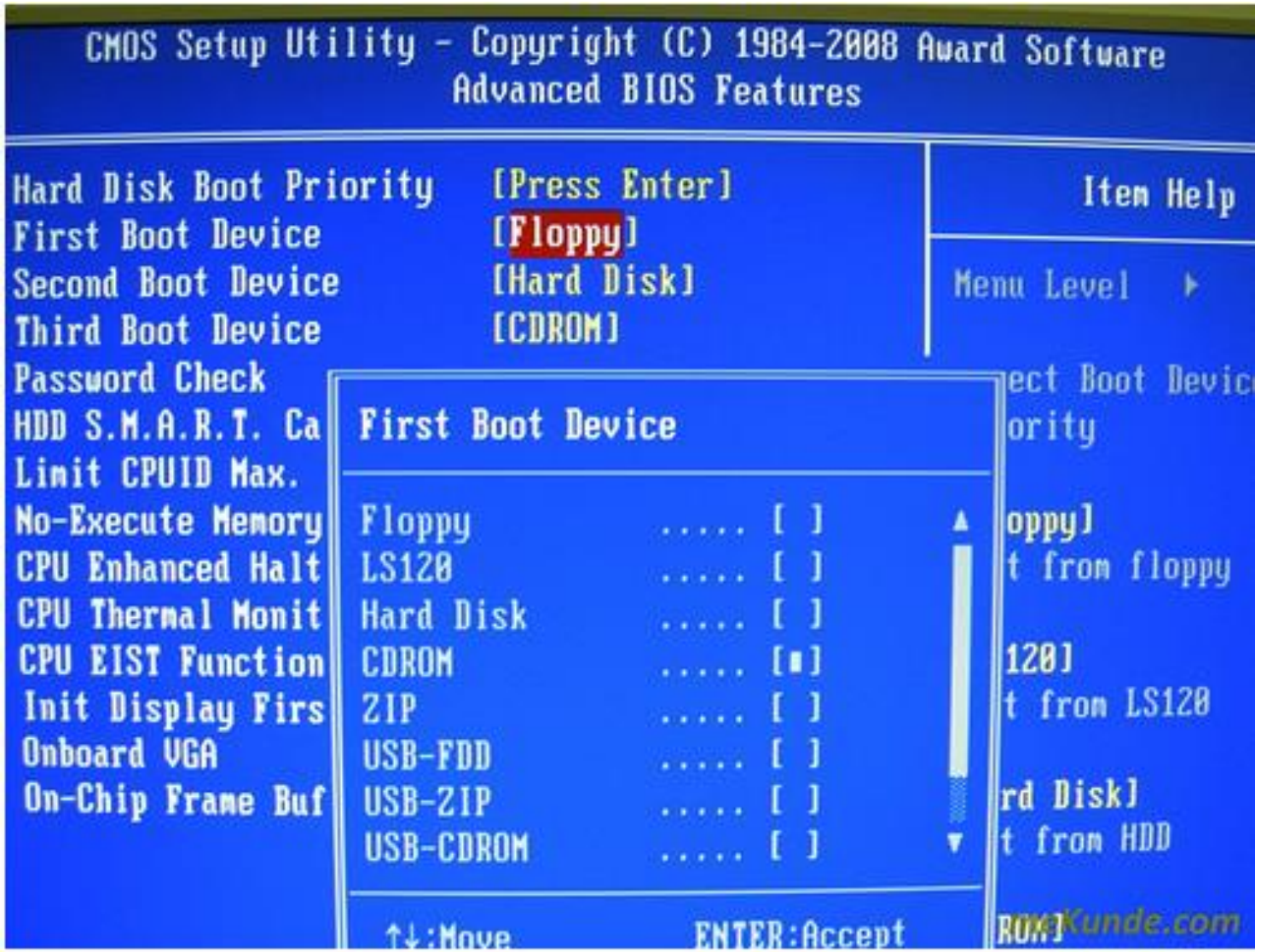
- **Bootstrap process**
  - Contained in ROM, tells the computer how to proceed
  - Displays the key or keys you press to open the CMOS setup screen
- CMOS should be modified to boot from a forensic floppy disk or CD

Many BIOS manufacturers use the Delete key to access CMOS; other manufacturers use Ctrl1Alt1Insert, Ctrl1A, Ctrl1S, or Ctrl1Fl, F2, or F10.

Understanding the Boot Sequence



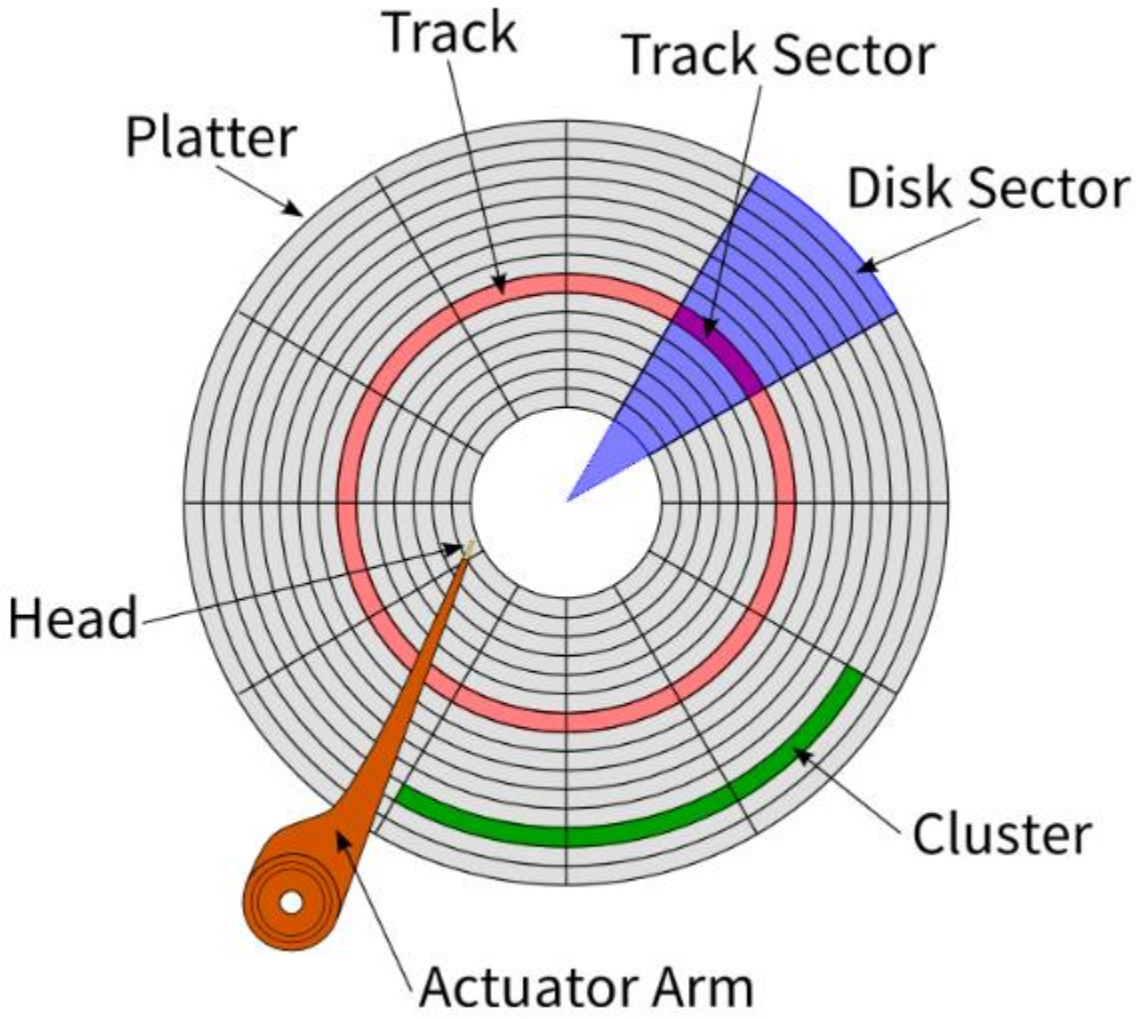
Understanding the Boot Sequence



Understanding Disk Drives

- Disk drives are made up of one or more platters coated with magnetic material
- Disk drive components
  - **Geometry—**refers to a disk's logical structure of platters, tracks, and sectors.
  - **Head—**The head is the device that reads and writes data to a drive. There are two heads per platter that read and write the top and bottom sides.
  - **Tracks—**Tracks are concentric circles on a disk platter where data is located.
  - **Cylinders—**A cylinder is a column of tracks on two or more disk platters. Typically, each platter has two surfaces: top and bottom.
  - **Sectors—**A sector is a section on a track, usually made up of 512 bytes.

Understanding Disk Drives



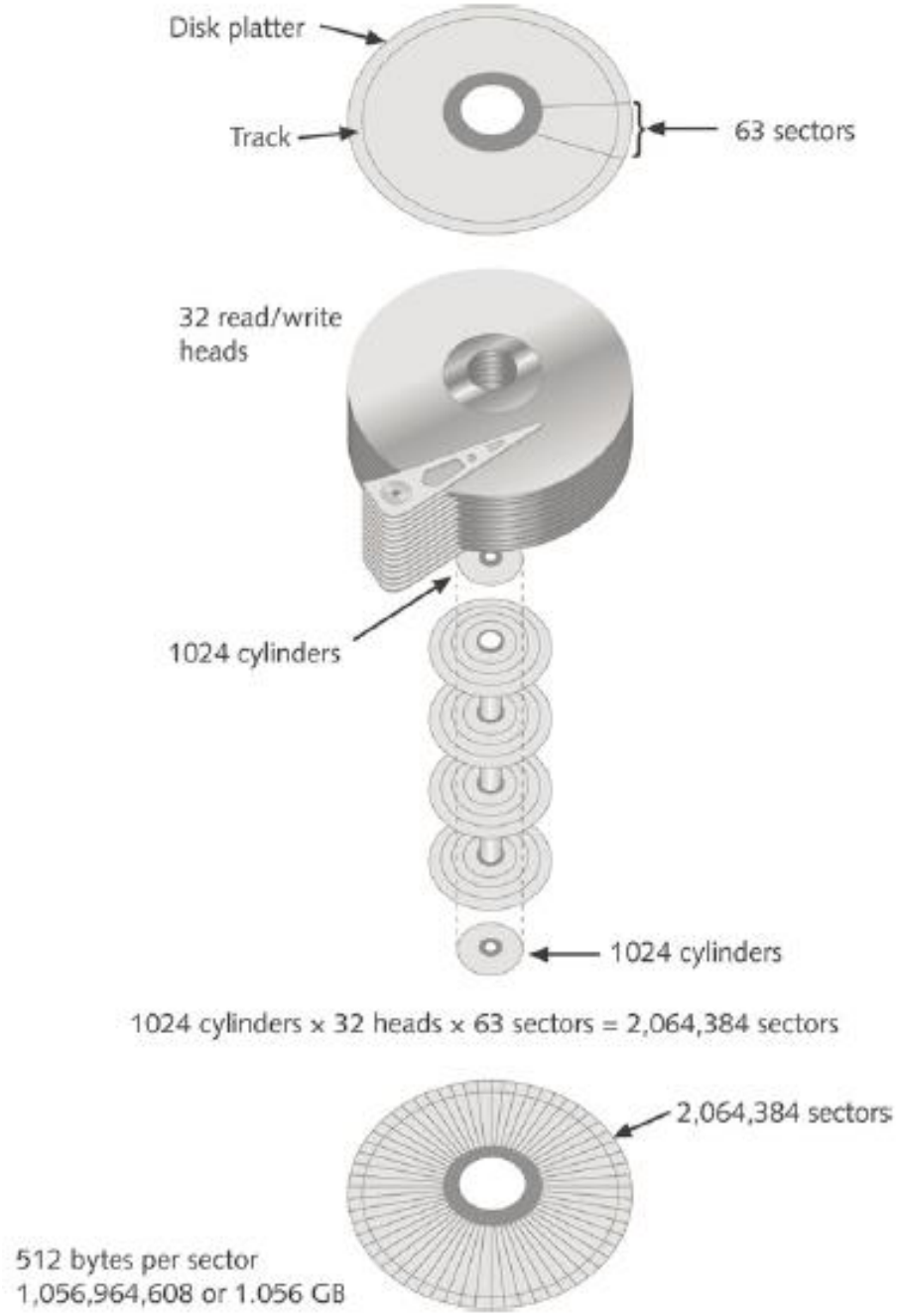


Figure 5-3 CHS calculation

Understanding Disk Drives

- Properties handled at the drive's hardware or firmware level
  - Zone bit recording (ZBR): is how most manufacturers deal with a platter's inner tracks having a smaller circumference
  - Track density: is the space between each track
  - Areal density: is the number of bits in one square inch of a disk platter.
  - Head and cylinder skew: are used to improve disk performance.

Solid-State Storage Devices

- All flash memory devices have a feature called **wear-leveling**
  - An internal firmware feature used in solid-state drives that ensures even wear of read/writes for all memory cells
- When dealing with solid-state devices, making a full forensic copy as soon as possible is crucial
  - In case you need to recover data from unallocated disk space

Wear leveling is a process that is designed to extend the life of solid-state storage devices.

Deleted data ….

- When data is deleted on a hard drive, only the references to it are removed, which leaves the original data in unallocated disk space.
  - With forensics recovery tools, recovering data from magnetic media is fairly easy by copying the unallocated space.
- USB drives are different, in that memory cells shift data at the physical level to other cells that have had fewer reads and writes continuously.
  - Memory cells are designed to perform only 10,000 to 100,000 reads/writes, depending on the manufacturer's design. When they reach their defined limits, they can no longer retain data.

Exploring Microsoft File Structures

- In Microsoft file structures, sectors are grouped to form **clusters**
  - Storage allocation units of one or more sectors
- Clusters range from 512 bytes up to 32,000 bytes each
- Combining sectors minimizes the overhead of writing or reading files to a disk

Exploring Microsoft File Structures

- Clusters are numbered sequentially starting at 0 in NTFS and 2 in FAT
  - First sector of all disks contains a system area, the boot record, and a file structure database
- OS assigns these numbers, called **logical addresses**cluster
- Sector numbers are called **physical addresses**
- Clusters and their addresses are specific to a logical disk drive, which is a disk partition

Disk Partitions

- A **partition** is a logical drive
- Windows OSs can have three primary partitions followed by an extended partition that can contain one or more logical drives
- Hidden partitions or voids
  - Large unused gaps between partitions on a disk
- **Partition gap**
  - Unused space between partitions

Disk Partitions

Table 5-1 Hexadecimal codes in the partition table

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

Disk Partitions

- The partition table is in the **Master Boot Record (MBR)**
  - Located at sector 0 of the disk drive
- MBR stores information about partitions on a disk and their locations, size, and other important items
- In a hexadecimal editor, such as WinHex, you can find the first partition at offset 0x1BE
  - The file system's hexadecimal code is offset 3 bytes from 0x1BE for the first partition

Disk Partitions

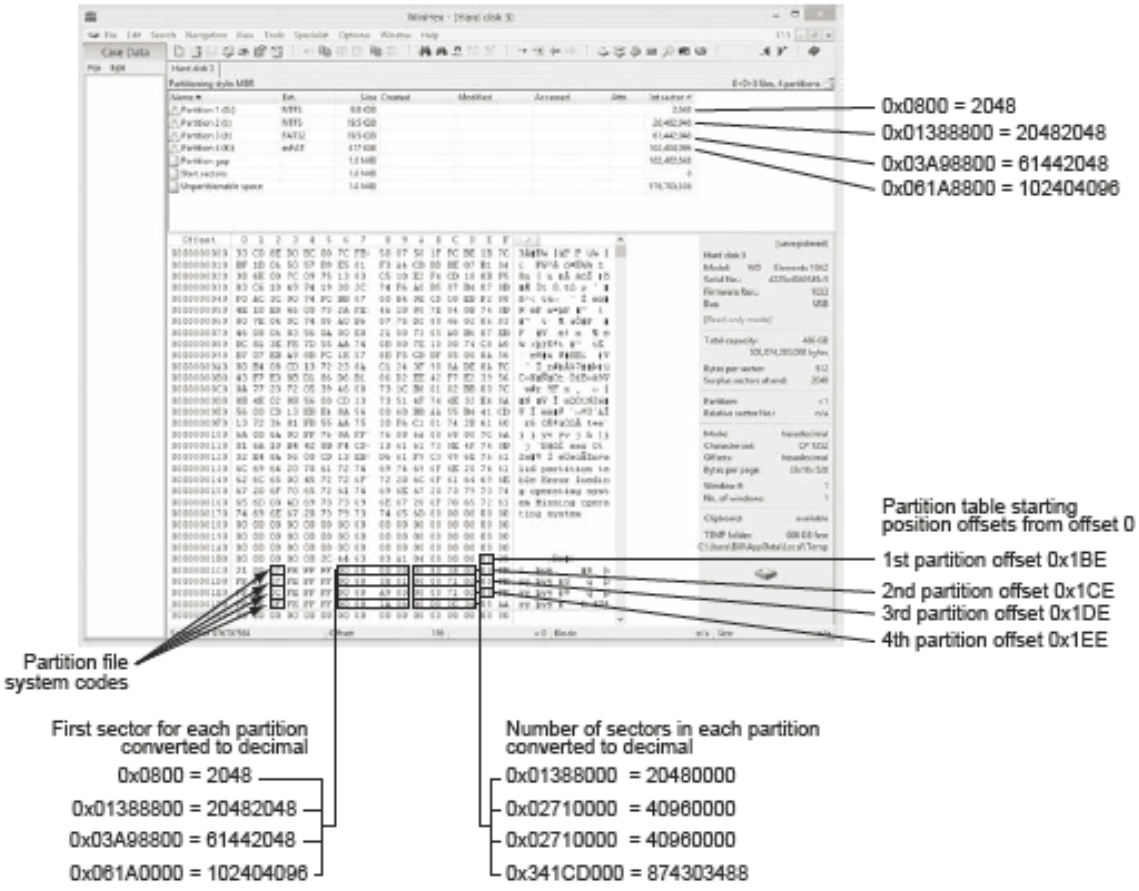


Figure 5-4 The partition table in a hexidecimal editor

Examining FAT Disks

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

Examining FAT Disks

- Cluster sizes vary according to the hard disk size and file system
- Table 5-2 lists the number of sectors and bytes assigned to a cluster on FAT16 disk according to hard disk size.

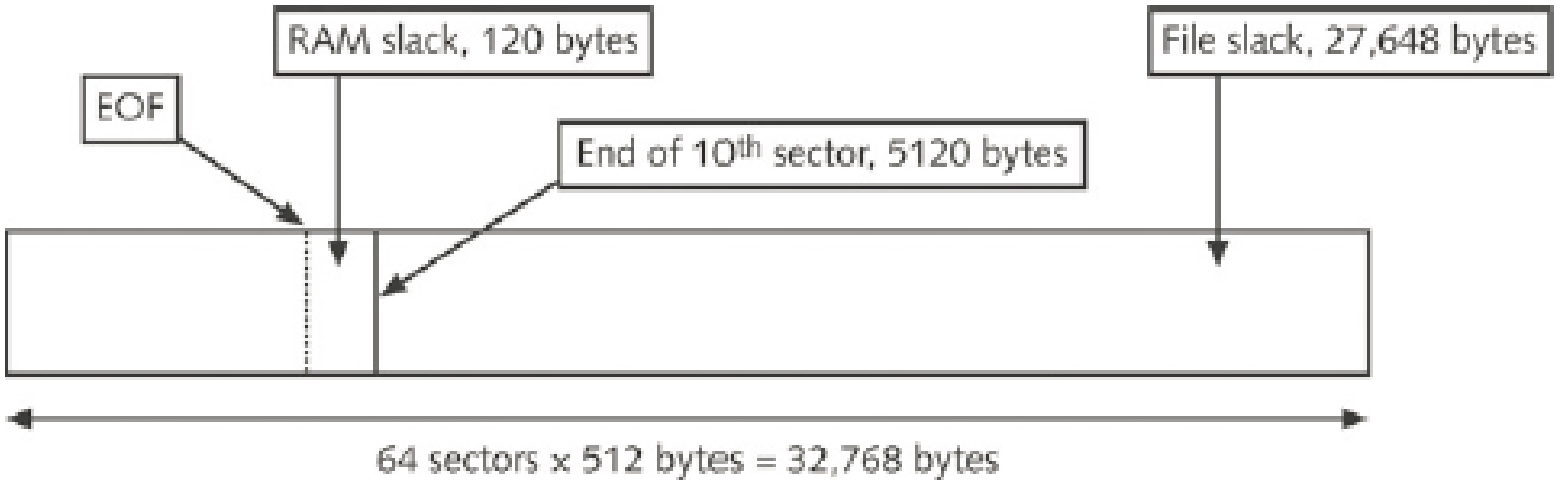
Table 5-2 Sectors and bytes per cluster

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

Examining FAT Disks

- Microsoft OSs allocate disk space for files by clusters
  - Results in **drive slack**
    - Unused space in a cluster between the end of an active file and the end of the cluster
- Drive slack includes:
  - **RAM slack** and **file slack**
- An unintentional side effect of FAT16 having large clusters was that it reduced fragmentation
  - As cluster size increased

For example, suppose you create a text document containing 5000 characters—that is, 5000 bytes of data. If you save this file on a FAT16 1.6 GB disk, a Microsoft OS reserves one cluster for it automatically. For a 1.6 GB disk, the OS allocates about 32,000 bytes, or 64 sectors (512 bytes per sector), for your file. The unused space, 27,000 bytes, is the file slack (see Figure 5-8). That is, RAM slack is the portion of the last sector used in the last assigned cluster, and the remaining sectors are referred to as "file slack." The 5000-byte text document uses up 10 sectors, or 5120 bytes, so 120 bytes of a sector aren't used; however, DOS must write in full 512-byte chunks of data (sectors). The data to fill the 120-byte void is pulled from RAM and placed in the area between the end of the file (EOF) and the end of the last sector used by the active file in the cluster.



Guic **Figure 5-8** File slack space

Examining FAT Disks

- When you run out of room for an allocated cluster
  - OS allocates another cluster for your file, which creates more slack space on the disk
- As files grow and require more disk space, assigned clusters are chained together
  - The chain can be broken or fragmented
- When the OS stores data in a FAT file system, it assigns a starting cluster position to a file
  - Data for the file is written to the first sector of the first assigned cluster

Examining FAT Disks

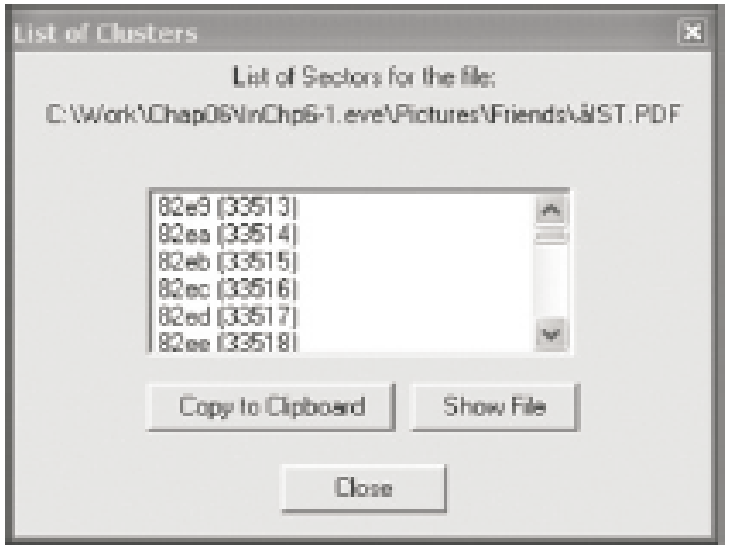


Figure 5-9 Chained sectors associated with clusters as a result of increasing file size
Courtesy of Technology Pathways, LLC

Examining FAT Disks

- When this first assigned cluster is filled and runs out of room
  - FAT assigns the next available cluster to the file
- If the next available cluster isn't contiguous to the current cluster
  - File becomes fragmented

Deleting FAT Files

- In Microsoft OSs, when a file is deleted
  - Directory entry is marked as a deleted file
    - With the HEX E5 character replacing the first letter of the filename
    - FAT chain for that file is set to 0 (0xE5)
- Data in the file remains on the disk drive
- Area of the disk where the deleted file resides becomes **unallocated disk space**
  - Available to receive new data from newly created files or other files needing more space