



WORKING WITH WINDOWS AND DOS SYSTEMS

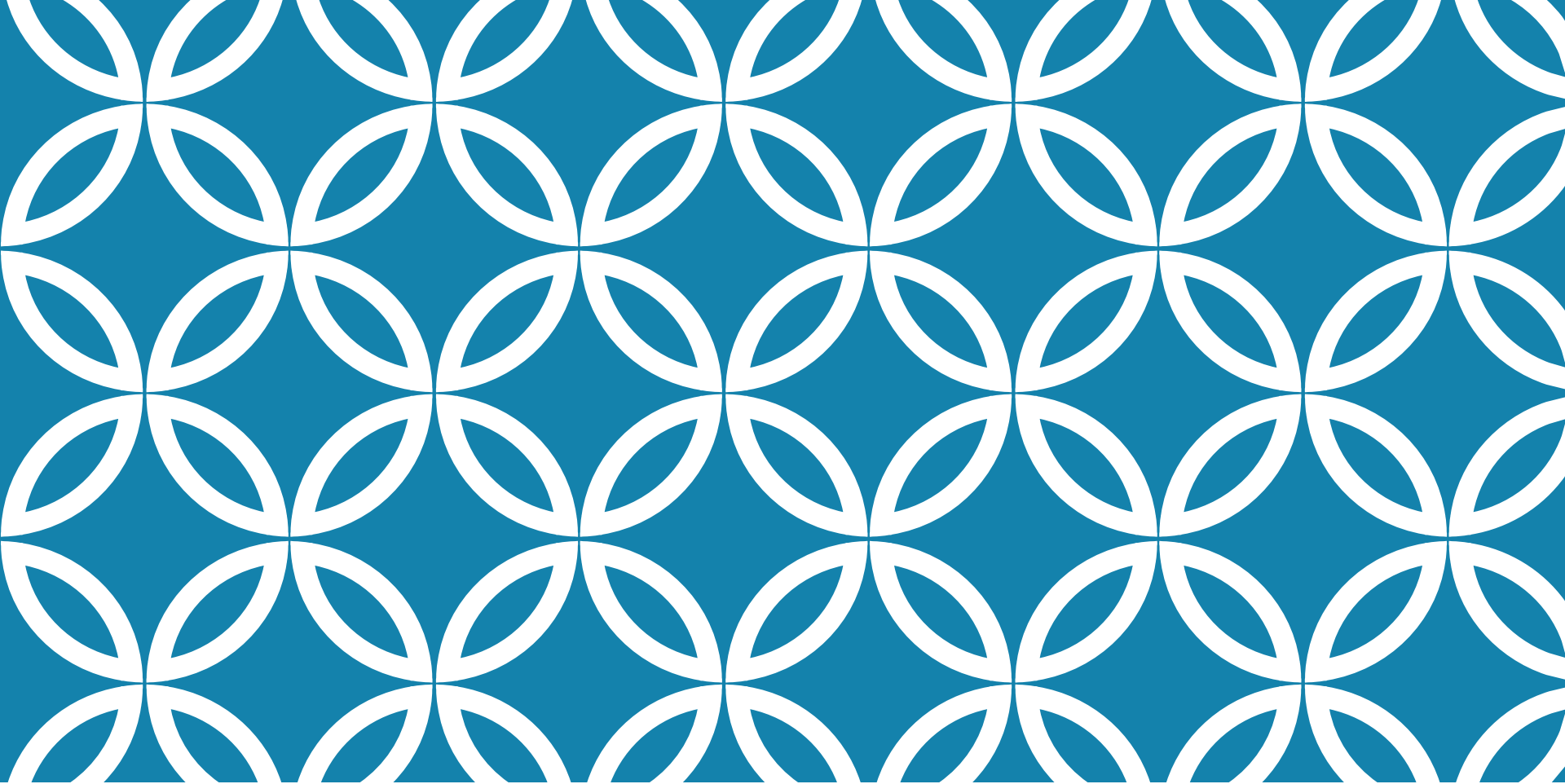
Chapter 7

OBJECTIVES

- ❑ Explain the purpose and structure of file systems
- ❑ Describe Microsoft file structures
- ❑ Explain the structure of New Technology File System (NTFS) disks
- ❑ List some options for decrypting drives encrypted with whole disk encryption

OBJECTIVES (CONTINUED)

- ☐ Explain how the Windows Registry works
- ☐ Describe Microsoft startup tasks
- ☐ Describe MS-DOS startup tasks
- ☐ Explain the purpose of a virtual machine



UNDERSTANDING FILE SYSTEMS

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
- ☐ A file system is usually directly related to an OS
- ☐ When you need to access a suspect's computer to acquire or inspect data
 - ☐ You should be familiar with the computer's platform

UNDERSTANDING THE BOOT SEQUENCE

Complementary Metal Oxide Semiconductor (CMOS)

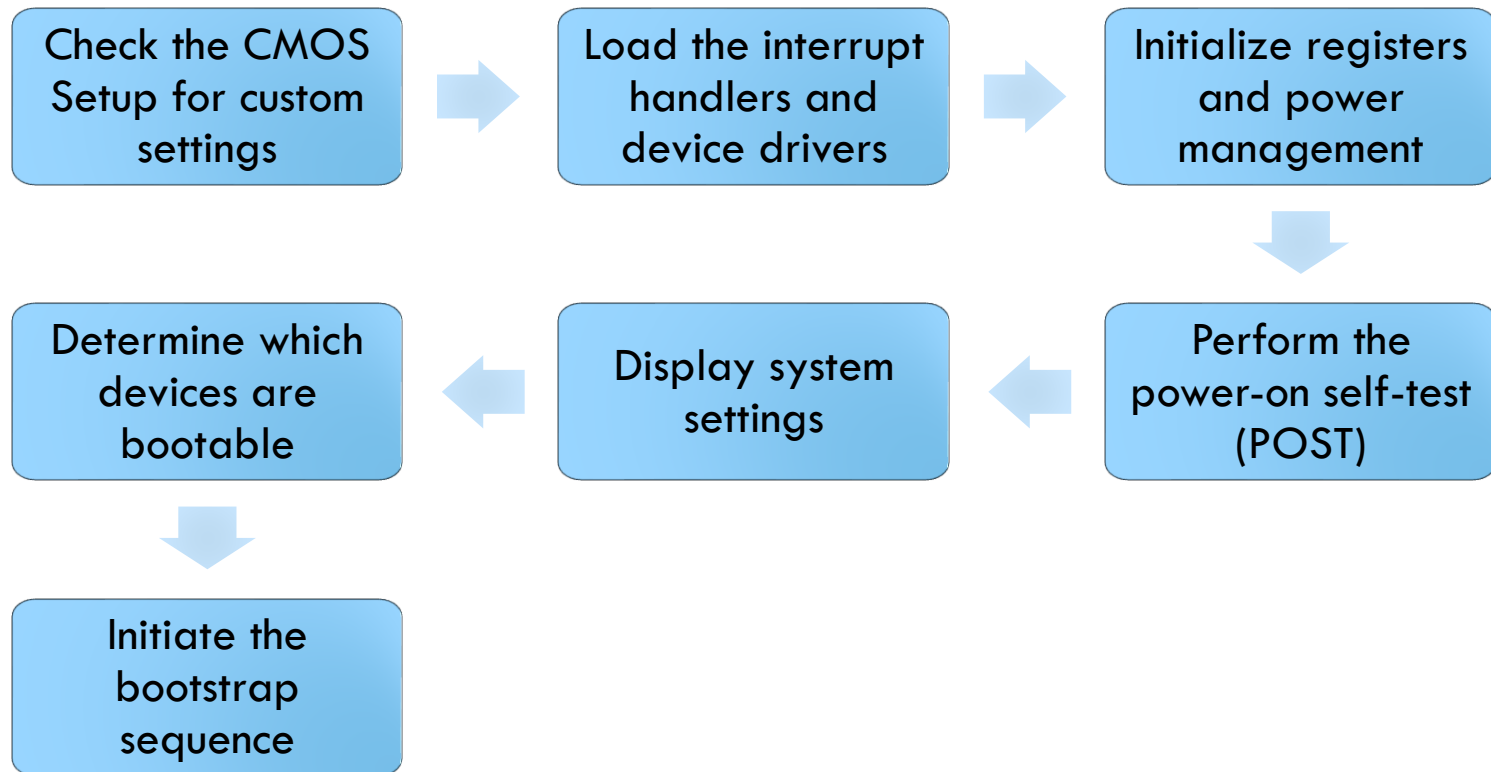
- Computer stores system configuration and date and time information in the CMOS
 - When power to the system is off

Basic Input/Output System (BIOS)

- Contains programs that perform input and output at the hardware level

UNDERSTANDING THE BOOT SEQUENCE

When you turn on your computer, the BIOS does several things. This is its usual sequence:



UNDERSTANDING THE BOOT SEQUENCE (CONTINUED)

Bootstrap process

- Contained in ROM, tells the computer how to proceed
- Displays the key or keys you press to open the CMOS setup screen
 - Could be Delete, F2, F10, Ctrl+Alt+Insert, Ctrl+A, Ctrl+S, Ctrl+F1, or something else

CMOS should be modified to boot from a forensic floppy disk or CD

BIOS SETUP UTILITY

PhoenixBIOS Setup Utility					
Main	Advanced	Security	Boot	Exit	
<div>+Removable Devices +Hard Drive CD-ROM Drive Network boot from AMD Am79C970A</div>				<div>Item Specific Help</div> <div>Keys used to view or configure devices: <Enter> expands or collapses devices with a + or - <Ctrl+Enter> expands all <+> and <-> moves the device up or down. <n> May move removable device between Hard Disk or Removable Disk <d> Remove a device that is not installed.</div>	
F1	Help	↑↓	Select Item	-/+	Change Values
Esc	Exit	↔	Select Menu	Enter	Select ► Sub-Menu
			F9	Setup Defaults	
			F10	Save and Exit	

UNDERSTANDING DISK DRIVES

Disk drives are made up of one or more platters coated with magnetic material.

Disk drive components:

- Geometry - refers to the disk's structure of platters, tracks and sectors
- Head - is the device that reads and writes data to a drive. There's one head per platter
- 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, Holds 512 bytes (you cannot read or write anything less than a sector)

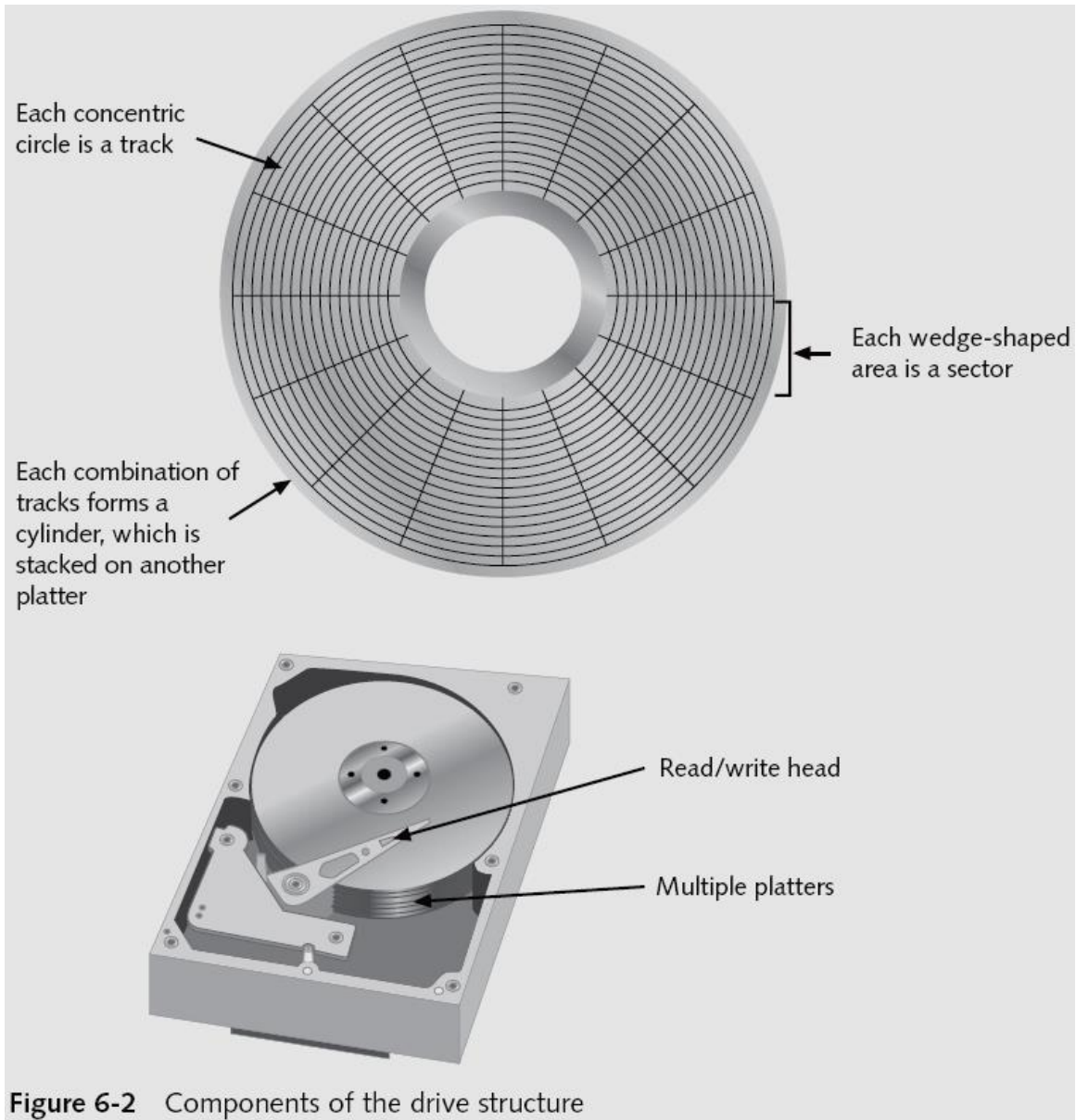
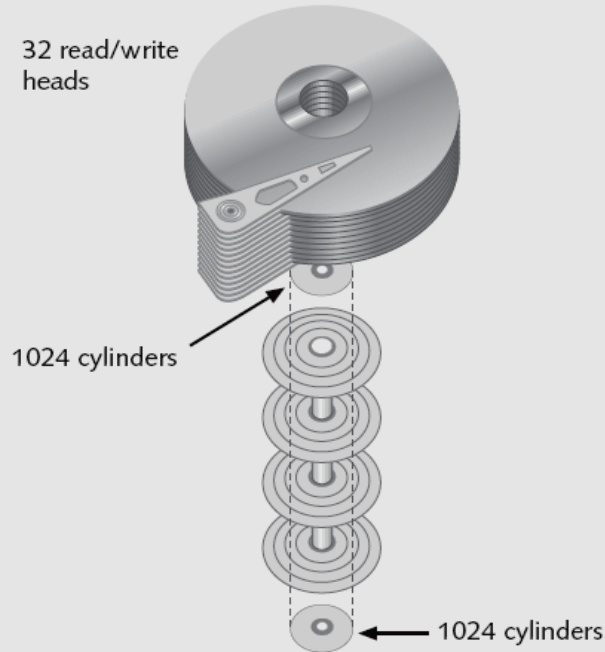
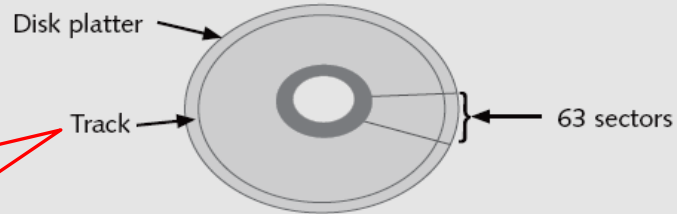
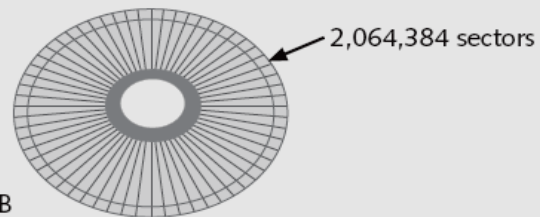


Figure 6-2 Components of the drive structure

Tracks following numbering scheme starting from 0
E.g.: if the disk lists 79 tracks, you actually have 80 tracks from 0 to 79



$$1024 \text{ cylinders} \times 32 \text{ heads} \times 63 \text{ sectors} = 2,064,384 \text{ sectors}$$



512 bytes per sector
1,056,964,608 or 1.056 GB

Figure 6-3 CHS calculation

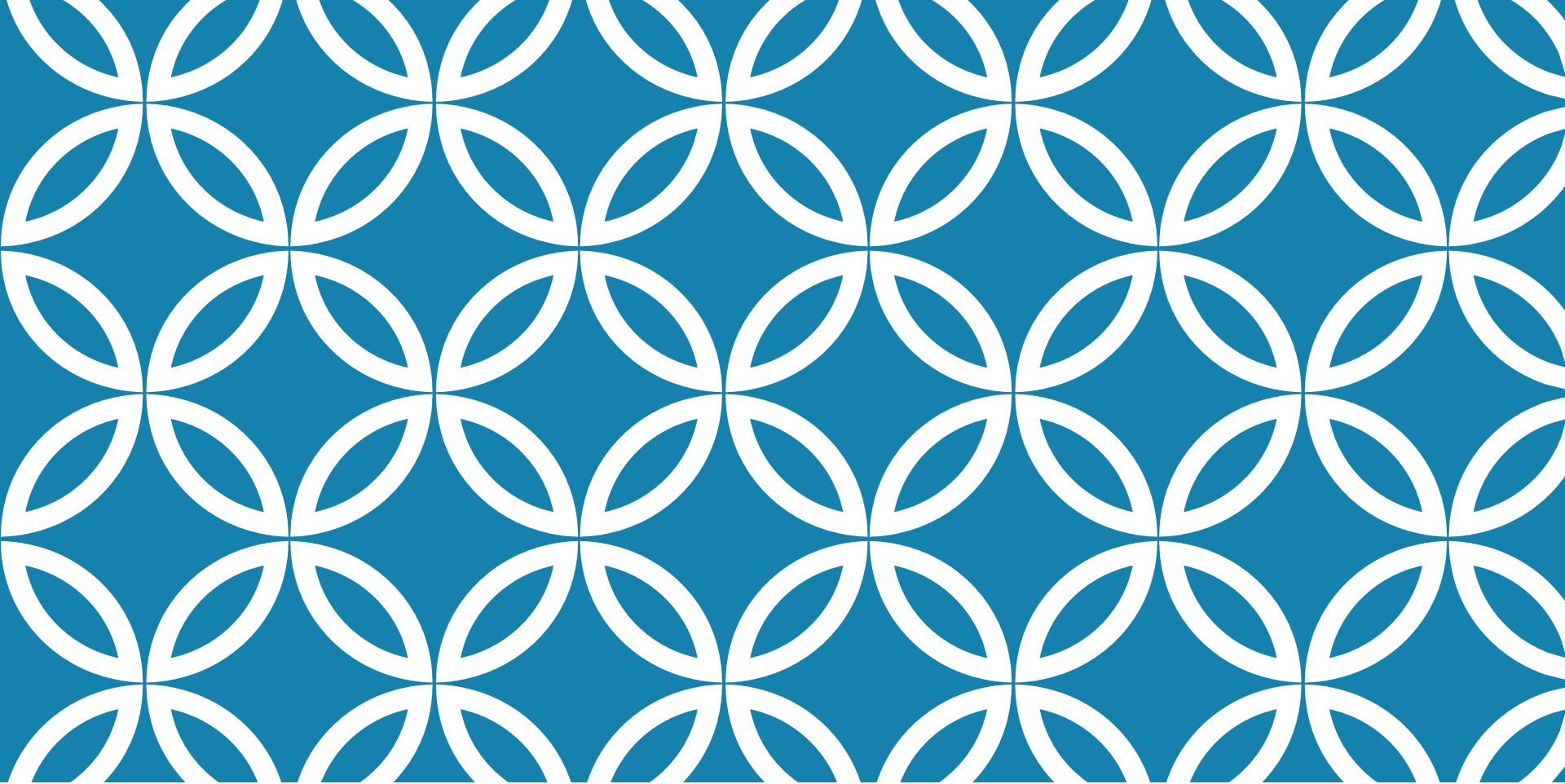
UNDERSTANDING DISK DRIVES (CONTINUED)

Properties handled at the drive's hardware or firmware level

- **Zoned bit recording (ZBR)** – is how most of manufacturers deal with a platter's inner tracks being shorter than its outer tracks. Grouping tracks by zones ensures all tracks hold the same amount of data.
- **Track density** - is the space between each track (the smaller the space between each track, the more tracks you can place on the platter.
- **Areal density** - refers to the number of bits in one square inch of a disk platter. This number includes the unused space between tracks.
- **Head and cylinder skew** - use to improve disk performance

NO NEED FOR MULTI-PATH ERASURE

- ❑ On older disks, the space between tracks was wider, which allowed heads to wander
- ❑ This made it possible for specialists to retrieve data from previous writes to a platter, even after erasure
 - ❑ Using an electron microscope
- ❑ On any IDE or SATA or later hard drive, this is impossible
- ❑ A single pass of zeroes erases all data on a disk so it cannot be recovered by any currently known technique



EXPLORING MICROSOFT FILE STRUCTURES

EXPLORING MICROSOFT FILE STRUCTURES

- ❑ In Microsoft file structures, sectors are grouped to form **clusters**
 - ❑ Storage allocation units of one or more sectors
- ❑ Clusters are typically 512, 1024, 2048, 4096, or more bytes each
- ❑ Combining sectors minimizes the overhead of writing or reading files to a disk

EXPLORING MICROSOFT FILE STRUCTURES (CONTINUED)

- ❑ Clusters are numbered sequentially starting at 2
 - ❑ First sector of all disks contains a system area, the boot record, and a file structure database
- ❑ OS assigns these cluster numbers, called **logical addresses**
- ❑ 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
- ❑ FAT16 does not recognize disks larger than 2 GB
 - ❑ Large disks have to be partitioned
- ❑ Hidden partitions or voids
 - ❑ Large unused gaps between partitions on a disk
- ❑ **Partition gap**
 - ❑ Unused space between partitions

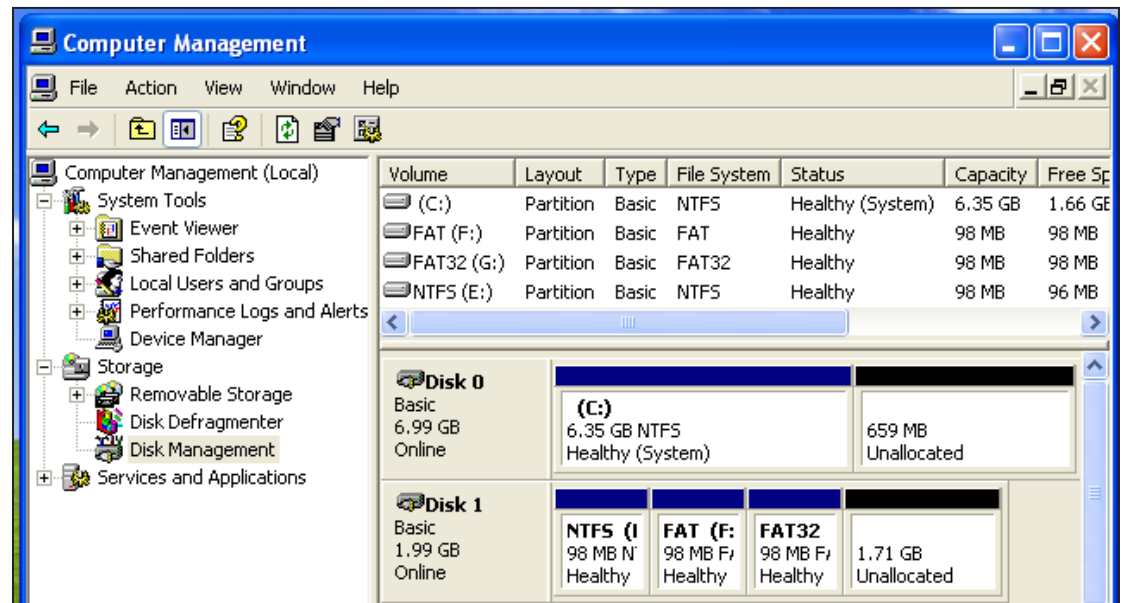
DISK PARTITIONS (CONTINUED)

- ❑ Disk editor utility can alter information in partition table
 - To hide a partition
- ❑ Can examine a partition's physical level with a disk editor:
 - HxD, Norton DiskEdit, WinHex, or Hex Workshop
- ❑ Analyze the key hexadecimal codes the OS uses to identify and maintain the file system

DEMO: VM WITH THREE PARTITIONS

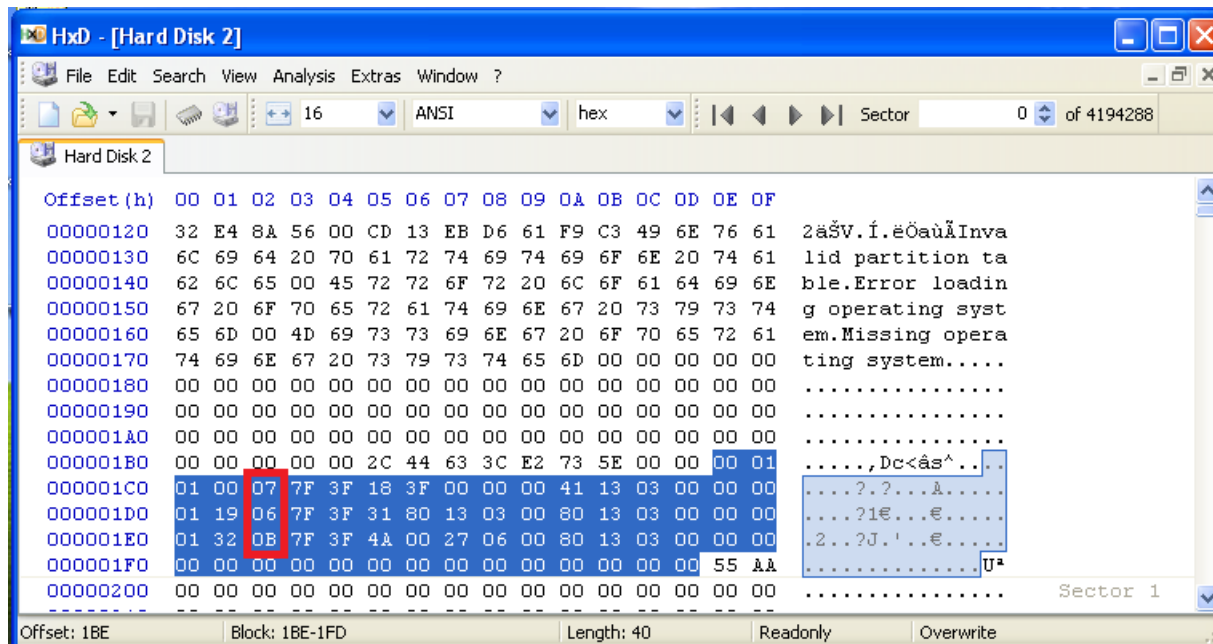
Partition Types

- NTFS: 07
- FAT: 06
- FAT32: 0B



VIEWING THE PARTITION TABLE HXD

- ❑ Start HxD, Extras, Open Disk, choose Physical Disk
- ❑ Partition Table starts at 0x1BE
- ❑ Partition Type field is at offset 0x04 in each record



MASTER BOOT RECORD STRUCTURE

From Wikipedia

Structure of a Master Boot Record

Address			Description		Size in bytes
Hex	Oct	Dec			
0000	0000	0	code area		440 (max. 446)
01B8	0670	440	disk signature (optional)		4
01BC	0674	444	Usually nulls; 0x0000		2
01BE	0676	446	Table of primary partitions (Four 16-byte entries, IBM partition table scheme)		64
01FE	0776	510	55h	MBR signature; 0xAA55 ^[1]	2
01FF	0777	511	AAh		
MBR, total size: 446 + 64 + 2 =					512

PARTITION TABLE STRUCTURE

Layout of one 16-byte partition record

Offset	Field length (bytes)	Description
0x00	1	status ^[7] (0x80 = bootable (<i>active</i>), 0x00 = non-bootable, other = invalid ^[8])
0x01	3	CHS address of first absolute sector in partition. ^[9] The format is described in the next 3 bytes.
0x01	1	head ^[10]
0x02	1	sector is in bits 5–0; ^[11] bits 9–8 of cylinder are in bits 7–6
0x03	1	bits 7–0 of cylinder ^[12]
0x04	1	partition type ^{[13][14]}
0x05	3	CHS address of last absolute sector in partition. ^[15] The format is described in the next 3 bytes.
0x05	1	head
0x06	1	sector is in bits 5–0; bits 9–8 of cylinder are in bits 7–6
0x07	1	bits 7–0 of cylinder
0x08	4	LBA of first absolute sector in the partition ^[16]
0x0C	4	number of sectors in partition, in little-endian format ^[16]

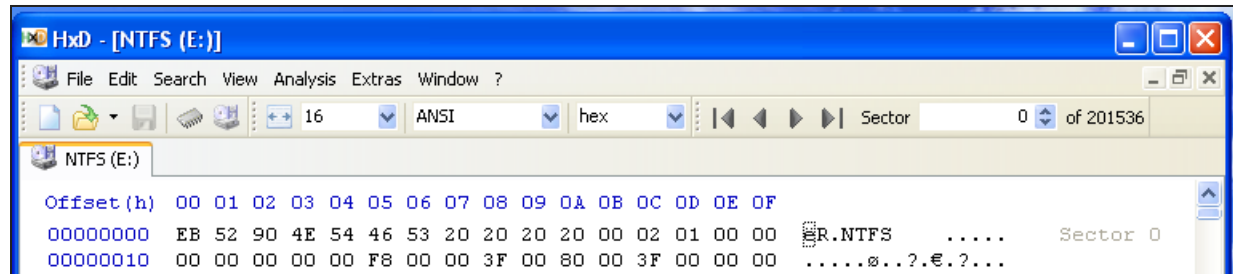
Table 6-1 Hexadecimal codes in the partition table

Hexadecimal code	File system
01	DOS 12-bit FAT
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
08	AIX bootable partition
09	AIX data partition
0B	DOS 32-bit FAT
0C	DOS 32-bit FAT for interrupt 13 support
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, 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

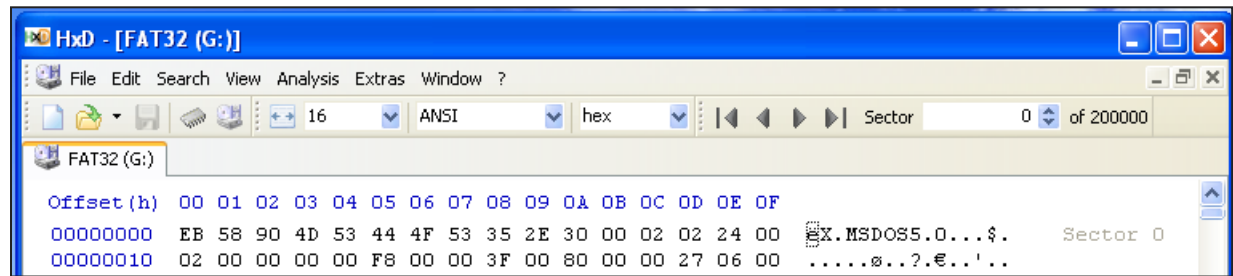
PARTITION MARK AT START OF VOLUME

Start HxD, Extras, Open Disk

NTFS



FAT32



.docx format is actually a Zip archive

MASTER BOOT RECORD

- ❑ On Windows and DOS computer systems
 - ❑ Boot disk contains a file called the **Master Boot Record (MBR)**
- ❑ MBR stores information about partitions on a disk and their locations, size, and other important items
- ❑ Several software products can modify the MBR, such as PartitionMagic's Boot Magic

EXAMINING FAT DISKS

☐ **File Allocation Table (FAT)**

- ☐ File structure database that Microsoft originally designed for floppy disks
- ☐ Used before Windows NT and 2000

☐ **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

☐ **FAT versions**

- ☐ FAT12, FAT16, FAT32, FATX (for Xbox), and VFAT

FAT VERSIONS

FAT12—for floppy disks, max size 16 MB

FAT16—allows hard disk sizes up to 2 GB

FAT32— allows hard disk sizes up to 2 TB

FATX—For Xbox media

- The date stamps start at the year 2000, unlike the other FAT formats that start at 1980

VFAT (Virtual File Allocation Table)

- Allows long file names on Windows (MS-DOS had 8.3 limitation)

EXAMINING FAT DISKS (CONTINUED)

Cluster sizes vary according to the hard disk size and file system

This table is for FAT-16

Table 6-2 Sectors and bytes per cluster

Drive size	Number of sectors per cluster	FAT16
0–32 MB	1	512 bytes
33–64 MB	2	1 KB
65–128 MB	4	2 KB
129–255 MB	8	4 KB
256–511 MB	16	8 KB
512–1023 MB	32	16 KB
1024–2047 MB	64	32 KB
2048–4095 MB	128	68 KB

EXAMINING FAT DISKS (CONTINUED)

- ❑ 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

EXAMINING FAT DISKS (CONTINUED)

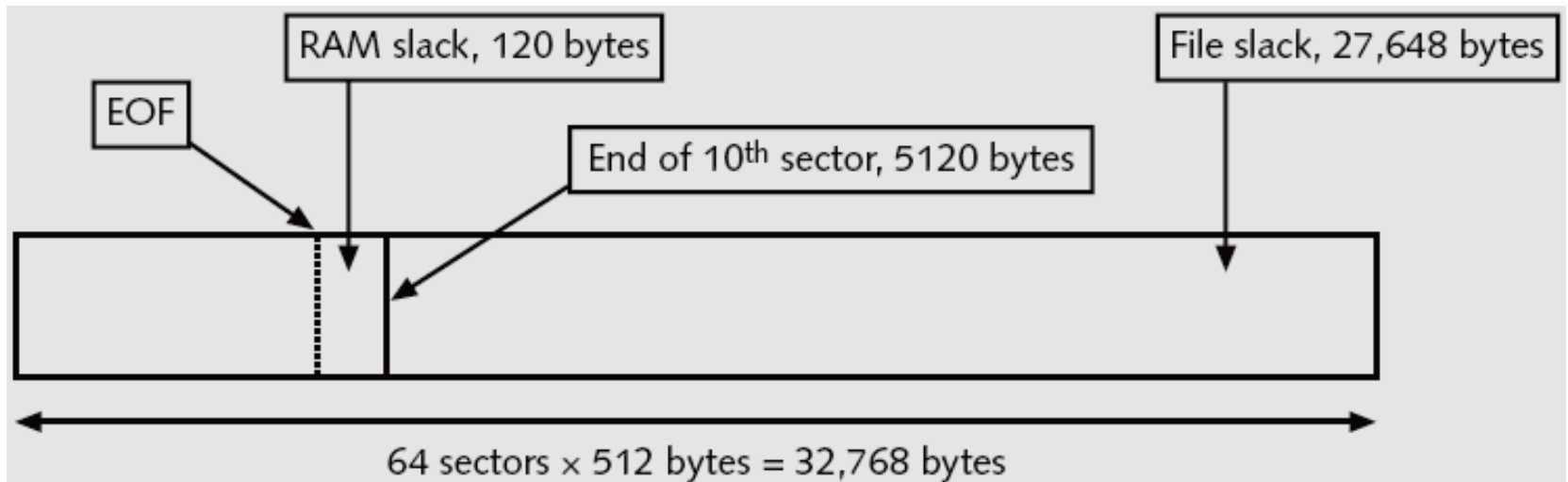


Figure 6-7 File slack space

EXAMINING FAT DISKS (CONTINUED)

- ❑ 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

PRODISCOVER SHOWING CLUSTER CHAIN



Figure 6-8 Chained sectors associated with clusters as a result of increasing file size

EXAMINING FAT DISKS (CONTINUED)

- ❑ 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
- ❑ 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 (σ) character replacing the first letter of the filename
 - ❑ FAT chain for that file is set to 0
- ❑ 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

EXAMINING NTFS DISKS

☐ **New Technology File System (NTFS)**

- ☐ Introduced with Windows NT
- ☐ Recommended file system for Windows 200 Pro, XP, and later versions through Windows 7 at least

☐ **Improvements over FAT file systems**

- ☐ NTFS provides more information about a file
- ☐ NTFS gives more control over files and folders

☐ **NTFS was Microsoft's move toward a journaling file system**

EXAMINING NTFS DISKS (CONTINUED)

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

EXAMINING NTFS DISKS (CONTINUED)

Table 6-3 Cluster sizes in an NTFS disk

Drive size	Sectors per cluster	Cluster size
0–512 MB	1	512 bytes
512 MB–1 GB	2	1024 bytes
1–2 GB	4	2048 bytes
2–4 GB	8	4096 bytes
4–8 GB	16	8192 bytes
8–16 GB	32	16,384 bytes
16–32 GB	64	32,768 bytes
More than 32 GB	128	65,536 bytes

NTFS FILE SYSTEM

- ❑ MFT contains information about all files on the disk
 - ❑ Including the system files the OS uses
- ❑ In the MFT, the first 15 records are reserved for system files
- ❑ Records in the MFT are called **metadata**

NTFS FILE SYSTEM (CONTINUED)

Table 6-4 Metadata records in the MFT

Filename	System file	Record position	Description
\$Mft	MFT	0	Base file record for each folder on the NTFS volume; other record positions in the MFT are allocated if more space is needed.
\$MftMirr	MFT 2	1	The first four records of the MFT are saved in this position. If a single sector fails in the first MFT, the records can be restored, allowing recovery of the MFT.
\$LogFile	Log file	2	Previous transactions are stored here to allow recovery after a system failure in the NTFS volume.
\$Volume	Volume	3	Information specific to the volume, such as label and version, is stored here.
\$AttrDef	Attribute definitions	4	A table listing attribute names, numbers, and definitions.
\$	Root file-name index	5	This is the root folder on the NTFS volume.

NTFS FILE SYSTEM (CONTINUED)

Table 6-4 Metadata records in the MFT (continued)

Filename	System file	Record position	Description
\$Bitmap	Boot sector	6	A map of the NTFS volume showing which clusters are in use and which are available.
\$Boot	Boot sector	7	Used to mount the NTFS volume during the bootstrap process; additional code is listed here if it's the boot drive for the system.
\$BadClus	Bad cluster file	8	For clusters that have unrecoverable errors, an entry of the cluster location is made in this file.
\$Secure	Security file	9	Unique security descriptors for the volume are listed in this file. It's where the access control list (ACL) is maintained for all files and folders on the NTFS volume.
\$Upcase	Upcase table	10	Converts all lowercase characters to uppercase Unicode characters for the NTFS volume.
\$Extend	NTFS extension file	11	Optional extensions are listed here, such as quotas, object identifiers, and reparse point data.
		12–15	Reserved for future use.

MFT AND FILE ATTRIBUTES

- ❑ In the NTFS MFT
 - ❑ All files and folders are stored in separate records of 1024 bytes each
 - ❑ Each record contains file or folder information
 - ❑ This information is divided into record fields containing metadata
 - ❑ A record field is referred to as an **attribute ID**
 - ❑ File or folder information is typically stored in one of two ways in an MFT record:
 - ❑ Resident and nonresident

MFT AND FILE ATTRIBUTES (CONTINUED)

- ❑ Files larger than 512 bytes are stored outside the MFT
 - ❑ MFT record provides cluster addresses where the file is stored on the drive's partition
 - ❑ Referred to as **data runs**
- ❑ Each MFT record starts with a header identifying it as a resident or nonresident attribute

Table 6-5 Attributes in the MFT

Attribute ID	Purpose
0x10	\$Standard Information This field contains data on file creation, alterations, MFT changes, read dates and times, and DOS file permissions.
0x20	\$Attribute_List Attributes that don't fit in the MFT (nonresident attributes) are listed here along with their locations.
0x30	\$File_Name The long and short names for a file are contained here. Up to 255 Unicode bytes are available for long filenames. For POSIX requirements, additional names or hard links can also be listed. Files with short filenames have only one attribute ID 0x30. Long filenames have two attribute ID 0x30s in the MFT record: one for the short name and one for the long name.
0x40	\$Object_ID (for Windows NT, it's named \$Volume_Version) Ownership and who has access rights to the file or folder are listed here. Every MFT record is assigned a unique GUID. Depending on your NTFS setup, some file records might not contain this attribute ID.
0x50	\$Security_Descriptor Contains the access control list (ACL) for the file.
0x60	\$Volume_Name The volume-unique file identifier is listed here. Not all files need this unique identifier.
0x70	\$Volume_Information This field indicates the version and state of the volume.
0x80	\$Data File data or data runs to nonresident files.
0x90	\$Index_Root Implemented for use of folders and indexes.
0xA0	\$Index_Allocation Implemented for use of folders and indexes.
0xB0	\$Bitmap Implemented for use of folders and indexes.
0xC0	\$Reparse_Point This field is used for volume mount points and Installable File System (IFS) filter drivers. For the IFS, it marks specific files used by drivers.
0xD0	\$EA_Information For use with OS2 HPFS file systems.
0xE0	\$EA For use with OS2 HPFS file systems.
0x100	\$Logged_Utility_Stream This field is used by Encrypting File System in Windows 2000 and XP.

RESIDENT FILE IN A MFT RECORD

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
035B3400	46	49	4C	45	30	00	03	00	9B	99	98	00	00	00	00	00
035B3410	02	00	01	00	30	00	01	00	A8	01	00	00	00	04	00	00
035B3420	00	00	00	00	00	00	00	00	04	00	00	00	A0	17	00	00
035B3430	03	00	00	00	00	00	00	00	10	00	00	00	60	00	00	00
035B3440	00	00	00	00	00	00	00	00	48	00	00	00	18	00	00	00
035B3450	62	16	9B	68	0A	7C	C9	01	BC	78	9D	68	0A	7C	C9	01
035B3460	BC	78	9D	68	0A	7C	C9	01	BC	78	9D	68	0A	7C	C9	01
035B3470	20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
035B3480	00	00	00	00	09	01	00	00	00	00	00	00	00	00	00	00
035B3490	00	00	00	00	00	00	00	00	30	00	00	00	70	00	00	00
035B34A0	00	00	00	00	00	00	02	00	52	00	00	00	18	00	01	00
035B34B0	8A	00	00	00	00	00	01	00	62	16	9B	68	0A	7C	C9	01
035B34C0	BC	78	9D	68	0A	7C	C9	01	BC	78	9D	68	0A	7C	C9	01
035B34D0	BC	78	9D	68	0A	7C	C9	01	00	00	00	00	00	00	00	00
035B34E0	00	00	00	00	00	00	00	00	20	00	00	00	00	00	00	00
035B34F0	08	03	42	00	65	00	6E	00	31	00	2E	00	74	00	78	00
035B3500	74	00	00	00	00	00	00	00	40	00	00	00	28	00	00	00
035B3510	00	00	00	00	00	00	03	00	10	00	00	00	10	00	00	00
035B3520	F4	7C	F1	27	DF	E7	DD	11	A8	3F	00	22	18	D5	88	06
035B3530	80	00	00	00	70	00	00	00	00	00	18	00	00	00	01	00
035B3540	34	00	00	00	12	00	00	00	41	20	63	6F	75	6E	74	72
035B3550	79	6D	61	6E	20	62	65	74	77	69	65	6E	20	74	77	6F
035B3560	20	6C	61	77	79	65	72	73	20	69	73	20	6C	69	6B	65
035B3570	20	61	20	66	69	73	68	20	62	65	74	77	65	65	6E	20
035B3580	74	77	6E	20	63	61	74	73	2E	0D	0A	42	65	6E	6A	61
035B3590	6D	69	6E	20	46	72	61	6E	6B	6C	69	6E	00	00	00	00
035B35A0	FF	FF	FF	FF	82	79	47	11	00	00	00	00	00	00	00	00

FILE0 ... ||| ...
 ...8...
 ...
 ...H...
 b|h|É.Mx.h|É.
 Mx.h|É.Mx.h|É.
 ...
 ...0...p...
 ...R...
 |...b|h|É.
 Mx.h|É.Mx.h|É.
 Mx.h|É.
 ...
 ..Ben.i...t.x.
 t...@...(
 ...
 ó|ñ'8çÿ."?."Ö|
 |...P...
 T...A countr
 yman between two
 lawyers is like
 a fish between
 two cats...Benja
 min Franklin...
 yyyyyyG...

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

RESIDENT FILE DATA IN THE MFT

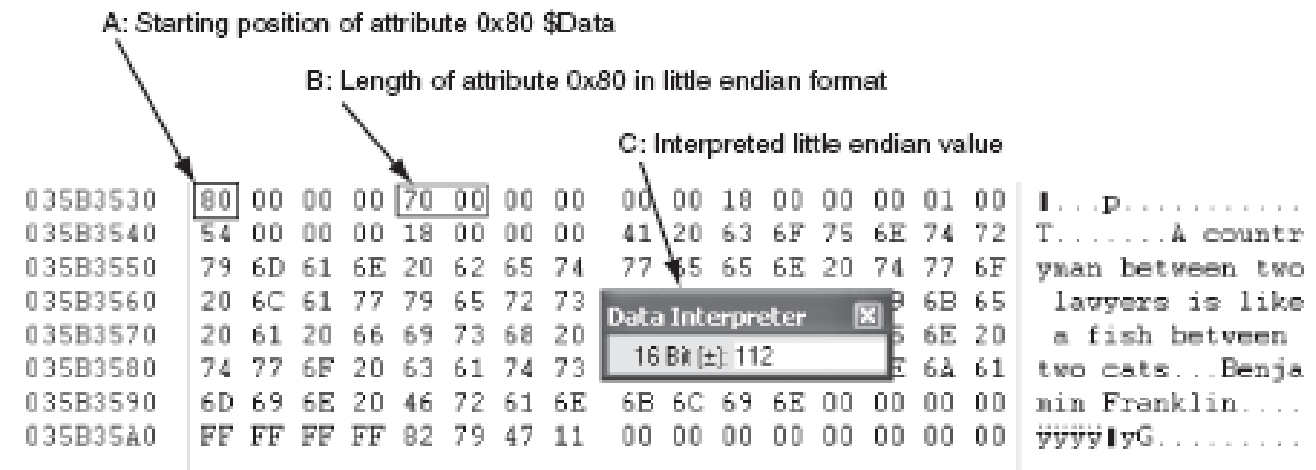


Figure 6-10 File data for a resident file

This figure is a repeat of a portion of the previous one

NONRESIDENT FILE'S MFT RECORD

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

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

Figure 6-11 Nonresident file in an MFT record

MFT AND FILE ATTRIBUTES (CONTINUED)

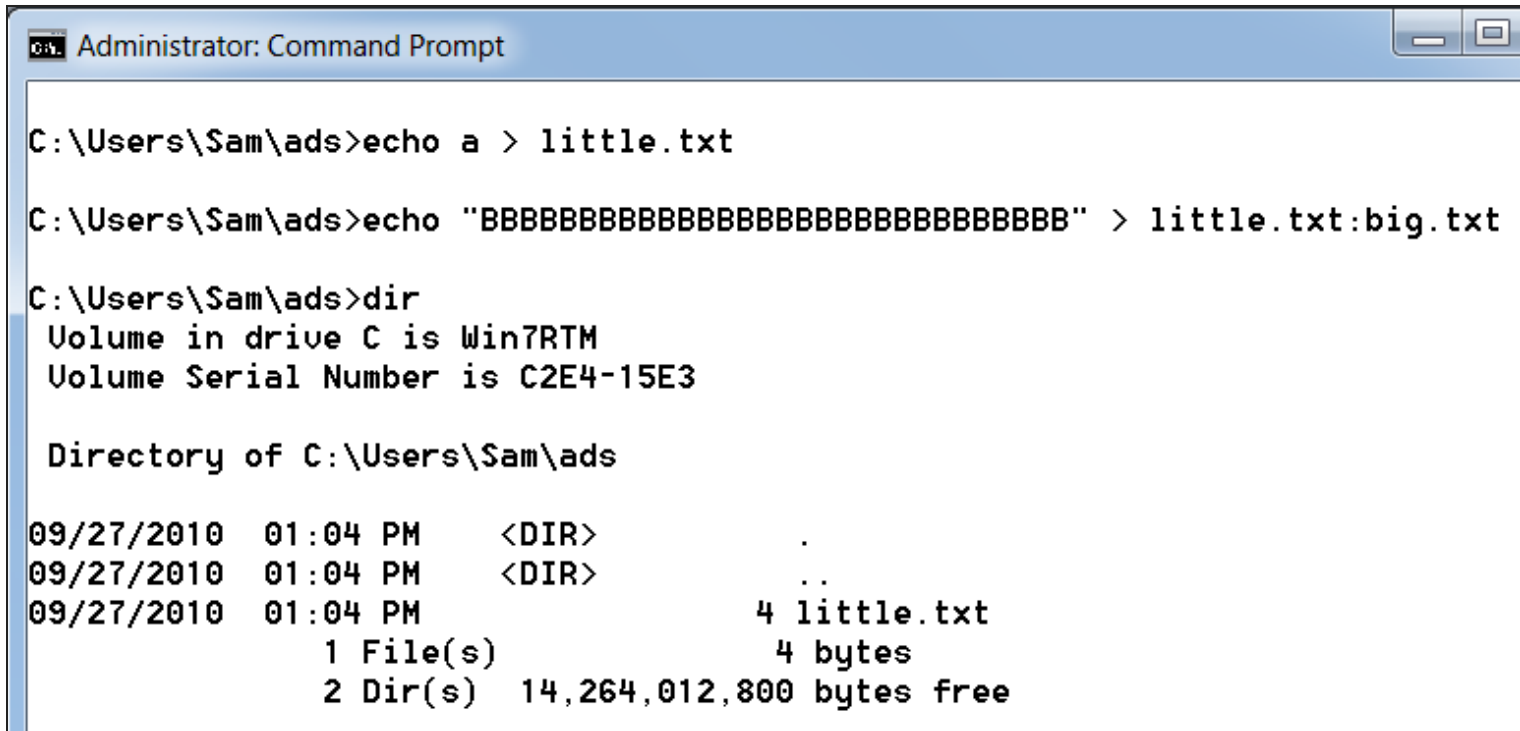
- ❑ When a disk is created as an NTFS file structure
 - ❑ OS assigns logical clusters to the entire disk partition
- ❑ These assigned clusters are called **logical cluster numbers (LCNs)**
 - ❑ Become the addresses that allow the MFT to link to nonresident files on the disk's partition

NTFS DATA STREAMS

☐ **Data streams**

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

ALTERNATE DATA STREAMS DEMONSTRATION



```
Administrator: Command Prompt

C:\Users\Sam\ads>echo a > little.txt

C:\Users\Sam\ads>echo "BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB" > little.txt:big.txt

C:\Users\Sam\ads>dir
Volume in drive C is Win7RTM
Volume Serial Number is C2E4-15E3

Directory of C:\Users\Sam\ads

09/27/2010  01:04 PM    <DIR>          .
09/27/2010  01:04 PM    <DIR>          ..
09/27/2010  01:04 PM                4 little.txt
               1 File(s)                4 bytes
               2 Dir(s)  14,264,012,800 bytes free
```

NTFS COMPRESSED FILES

- ❑ NTFS provides compression similar to FAT DriveSpace 3
- ❑ Under NTFS, files, folders, or entire volumes can be compressed
- ❑ Most computer forensics tools can uncompress and analyze compressed Windows data

NTFS ENCRYPTING FILE SYSTEM (EFS)

❑ Encrypting File System (EFS)

- ❑ Introduced with Windows 2000
- ❑ Implements a **public key** and **private key** method of encrypting files, folders, or disk volumes

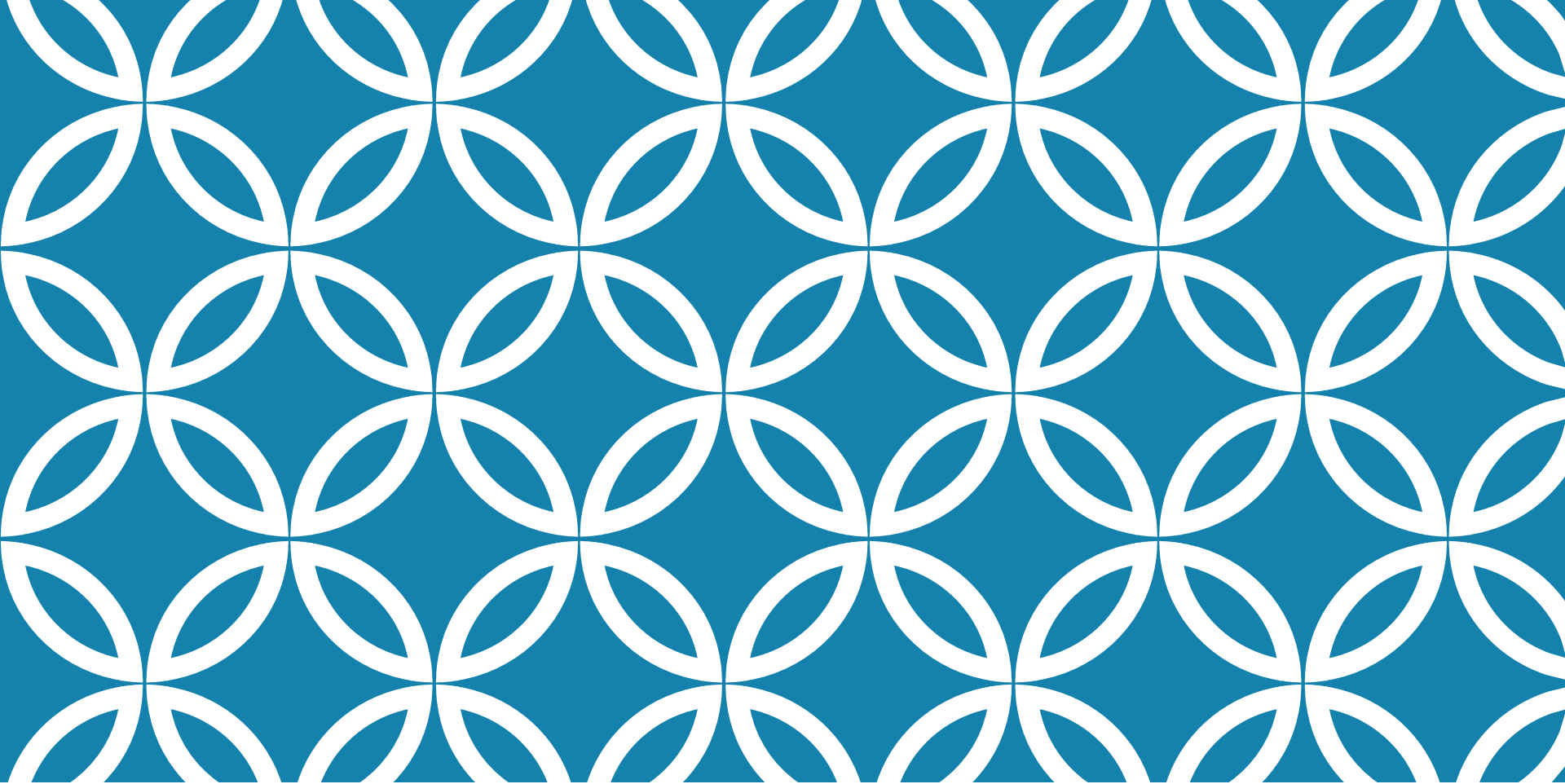
❑ When EFS is used in Windows 2000

- ❑ A **recovery certificate** is generated and sent to the local Windows administrator account

❑ Users can apply EFS to files stored on their local workstations or a remote server

DELETING NTFS FILES

- ☐ When a file is deleted in Windows XP, 2000, or NT
 - ☐ The OS renames it and moves it to the Recycle Bin
- ☐ Can use the Del (delete) MS-DOS command
 - ☐ Eliminates the file from the MFT listing in the same way FAT does



UNDERSTANDING WHOLE DISK ENCRYPTION



UNDERSTANDING WHOLE DISK ENCRYPTION

- ❑ In recent years, there has been more concern about loss of
 - ❑ **Personal identity information (PII)** and trade secrets caused by computer theft
- ❑ Of particular concern is the theft of laptop computers and other handheld devices
- ❑ To help prevent loss of information, software vendors now provide whole disk encryption

UNDERSTANDING WHOLE DISK ENCRYPTION (CONTINUED)

- ❑ Current whole disk encryption tools offer the following features:
 - ❑ Preboot authentication (e.g. single sig-on password, fingerprint scan or token)
 - ❑ Full or partial disk encryption with secure hibernation (e.g. activating a password-protected screen saver)
 - ❑ Advanced encryption algorithms (e.g. AES and IDEA)
 - ❑ Key management function (uses a challenge-and-response method to reset passwords or passphrases)
 - ❑ A **Trusted Platform Module (TPM)** microchip to generate encryption keys and authenticate logins

UNDERSTANDING WHOLE DISK ENCRYPTION (CONTINUED)

- ❑ Whole disk encryption tools encrypt each sector of a drive separately
- ❑ Many of these tools encrypt the drive's boot sector
 - ❑ To prevent any efforts to bypass the secured drive's partition
- ❑ To examine an encrypted drive, decrypt it first
 - ❑ Run a vendor-specific program to decrypt the drive

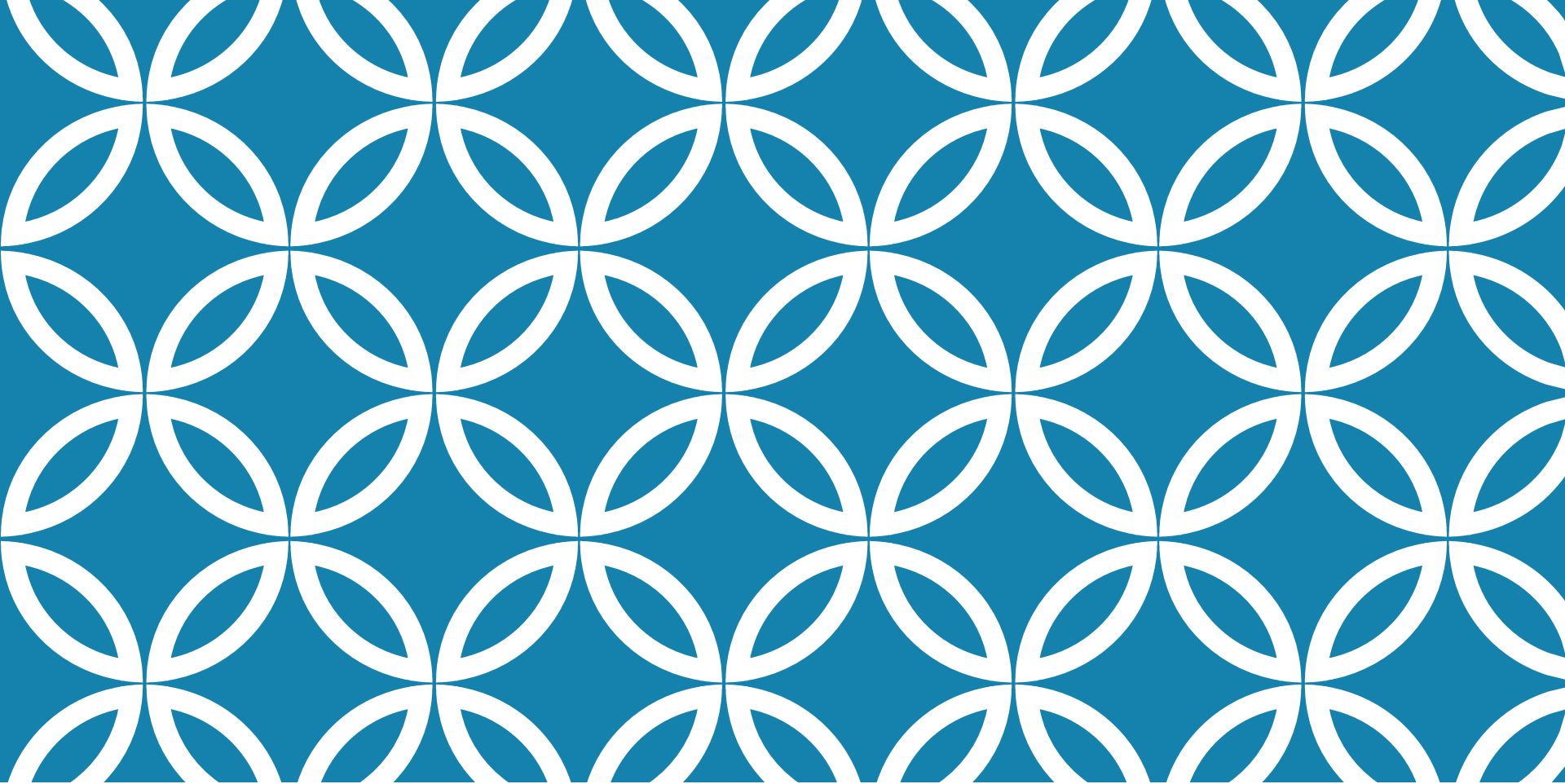
EXAMINING THIRD-PARTY DISK ENCRYPTION TOOLS

☐ Some available third-party WDE utilities:

- ☐ PGP Whole Disk Encryption
- ☐ Voltage SecureDisk
- ☐ Utimaco SafeGuard Easy
- ☐ Jetico BestCrypt Volume Encryption
- ☐ SoftWinter Sentry 2020 for Windows XP

☐ Some available open-source encryption tools:

- ☐ TrueCrypt
- ☐ CrossCrypt
- ☐ FreeOTFE



UNDERSTANDING THE WINDOWS REGISTRY



UNDERSTANDING THE WINDOWS REGISTRY

☐ Registry

- ☐ A database that stores hardware and software configuration information, network connections, user preferences, and setup information
- ☐ For investigative purposes, the Registry can contain valuable evidence
- ☐ To view the Registry, you can use:
 - ☐ Regedit (Registry Editor) program for Windows 9x systems
 - ☐ Regedt32 for Windows 2000 and XP

EXPLORING THE ORGANIZATION OF THE WINDOWS REGISTRY

☐ Registry terminology:

- ☐ Registry – collection of files containing system and user information
- ☐ Registry Editor – windows utility for viewing and modifying data in Registry
- ☐ HKEY – categories of registry
- ☐ Key – each HKEY contains folders referred as keys. Keys can contain other key folders or values
- ☐ Subkey – a key displayed under another key

EXPLORING THE ORGANIZATION OF THE WINDOWS REGISTRY

☐ Registry terminology:

- ☐ Branch – a key and its contents
- ☐ Value – a name and value in the key; its similar to a file and its data content
- ☐ Default value
- ☐ Hives – specific branches in HKEY_

EXPLORING THE ORGANIZATION OF THE WINDOWS REGISTRY (CONTINUED)

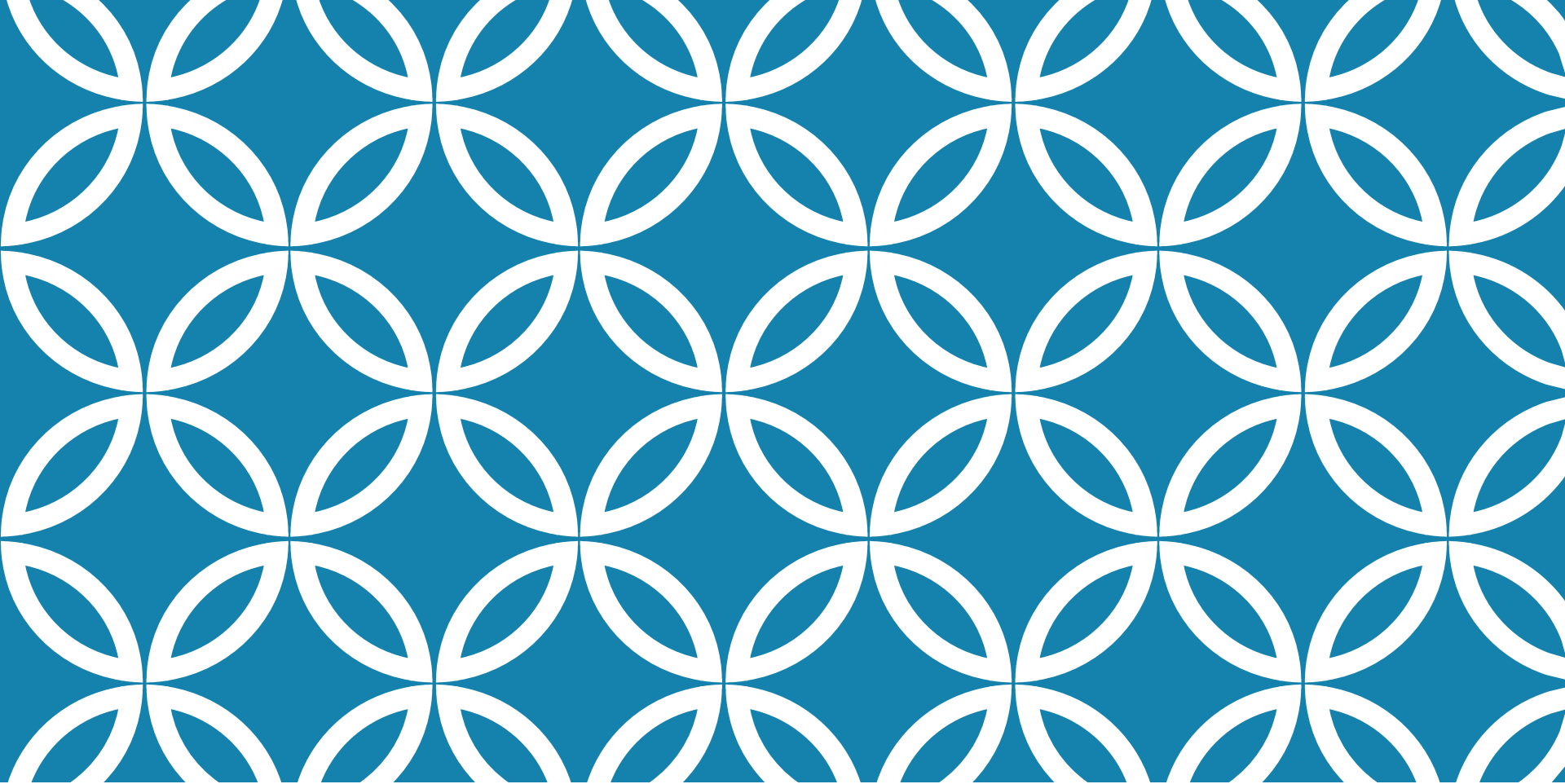
Table 6-6 Registry file locations and purposes

Filename and location	Purpose of file
Windows 9x/Me	
Windows\System.dat	User-protected storage area; contains installed program settings, usernames and passwords associated with installed programs, and system settings
Windows\User.dat Windows\profile\user-account	Contains the most recently used (MRU) files list and desktop configuration settings; every user account created on the system has its own user data file
Windows NT, 2000, XP, and Vista	
Documents and Settings\ user-account\Ntuser.dat	User-protected storage area; contains the MRU files list and desktop configuration settings
Winnt\system32\config\Default	Contains the computer's system settings
Winnt\system32\config\SAM	Contains user account management and security settings
Winnt\system32\config\Security	Contains the computer's security settings
Winnt\system32\config\Software	Contains installed programs settings and associated usernames and passwords
Winnt\system32\config\System	Contains additional computer system settings

EXPLORING THE ORGANIZATION OF THE WINDOWS REGISTRY (CONTINUED)

Table 6-7 Registry HKEYs and their functions

HKEY	Function
HKEY_CLASS_ROOT	A symbolic link to HKEY_LOCAL_MACHINE\SOFTWARE\Classes; provides file type and file extension information, URL protocol prefixes, and so forth
HKEY_CURRENT_USER	A symbolic link to HKEY_USERS; stores settings for the currently logged-on user
HKEY_LOCAL_MACHINE	Contains information about installed hardware and software
HKEY_USERS	Stores information for the currently logged-on user; only one key in this HKEY is linked to HKEY_CURRENT_USER
HKEY_CURRENT_CONFIG	A symbolic link to HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Hardware Profile\xxxx (with xxxx representing the current hardware profile); contains hardware configuration settings
HKEY_DYN_DATA	Used only in Windows 9x/Me systems; stores hardware configuration settings



UNDERSTANDING MICROSOFT STARTUP TASKS



UNDERSTANDING MICROSOFT STARTUP TASKS

- ❑ Learn what files are accessed when Windows starts
- ❑ This information helps you determine when a suspect's computer was last accessed
 - ❑ Important with computers that might have been used after an incident was reported

STARTUP IN WINDOWS NT AND LATER

□ All Windows NT computers perform the following steps when the computer is turned on:

- Power-on self test (POST)
- Initial startup
- Boot loader
- Hardware detection and configuration
- Kernel loading
- User logon

STARTUP IN WINDOWS NT AND LATER (CONTINUED)

Windows XP System Files

Table 6-8 Windows XP system files

Filename	Description
Ntoskrnl.exe	The XP executable and kernel
Ntkrnlpa.exe	The physical address support program for accessing more than 4 GB of physical RAM
Hal.dll	The Hardware Abstraction Layer (described earlier)
Win32k.sys	The kernel-mode portion of the Win32subsystem
Ntdll.dll	System service dispatch stubs to executable functions and internal support functions
Kernel32.dll	Core Win32 subsystem DLL file
Advapi32.dll	Core Win32 subsystem DLL file
User32.dll	Core Win32 subsystem DLL file
Gdi32.dll	Core Win32 subsystem DLL file

STARTUP IN WINDOWS NT AND LATER (CONTINUED)

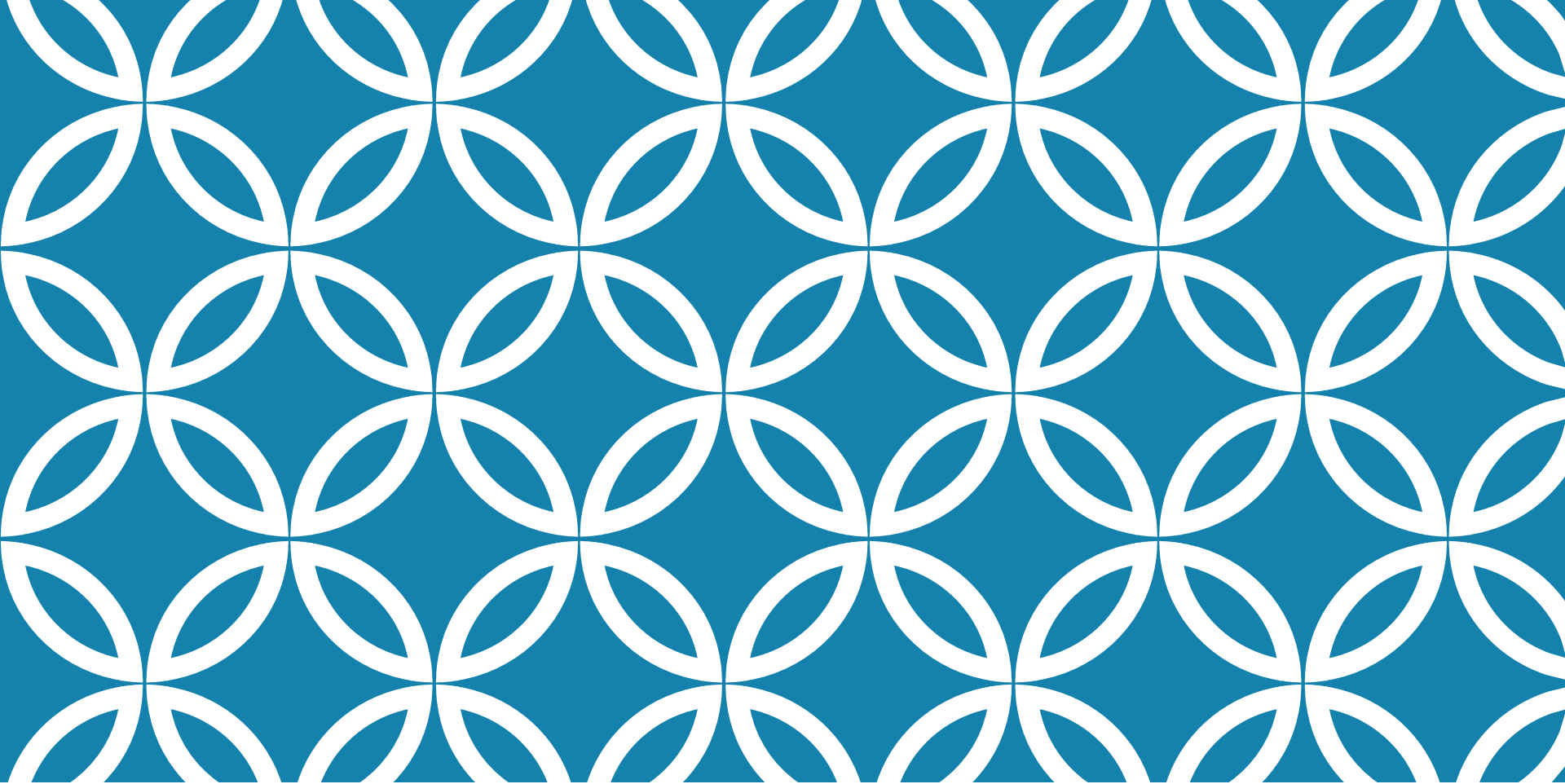
☐ Contamination Concerns with Windows XP

- ☐ When you start a Windows XP NTFS workstation, several files are accessed immediately

- ☐ The last access date and time stamp for the files change to the current date and time

- ☐ Destroys any potential evidence

- ☐ That shows when a Windows XP workstation was last used



UNDERSTANDING MS-DOS STARTUP TASKS

UNDERSTANDING MS-DOS STARTUP TASKS

- ❑ Two files are used to configure MS-DOS at startup:

- ❑ **Config.sys**

- ❑ A text file containing commands that typically run only at system startup to enhance the computer's DOS configuration

- ❑ **Autoexec.bat**

- ❑ A batch file containing customized settings for MS-DOS that runs automatically

- ❑ **io.sys** is the first file loaded after the ROM bootstrap loader finds the disk drive

UNDERSTANDING MS-DOS STARTUP TASKS (CONTINUED)

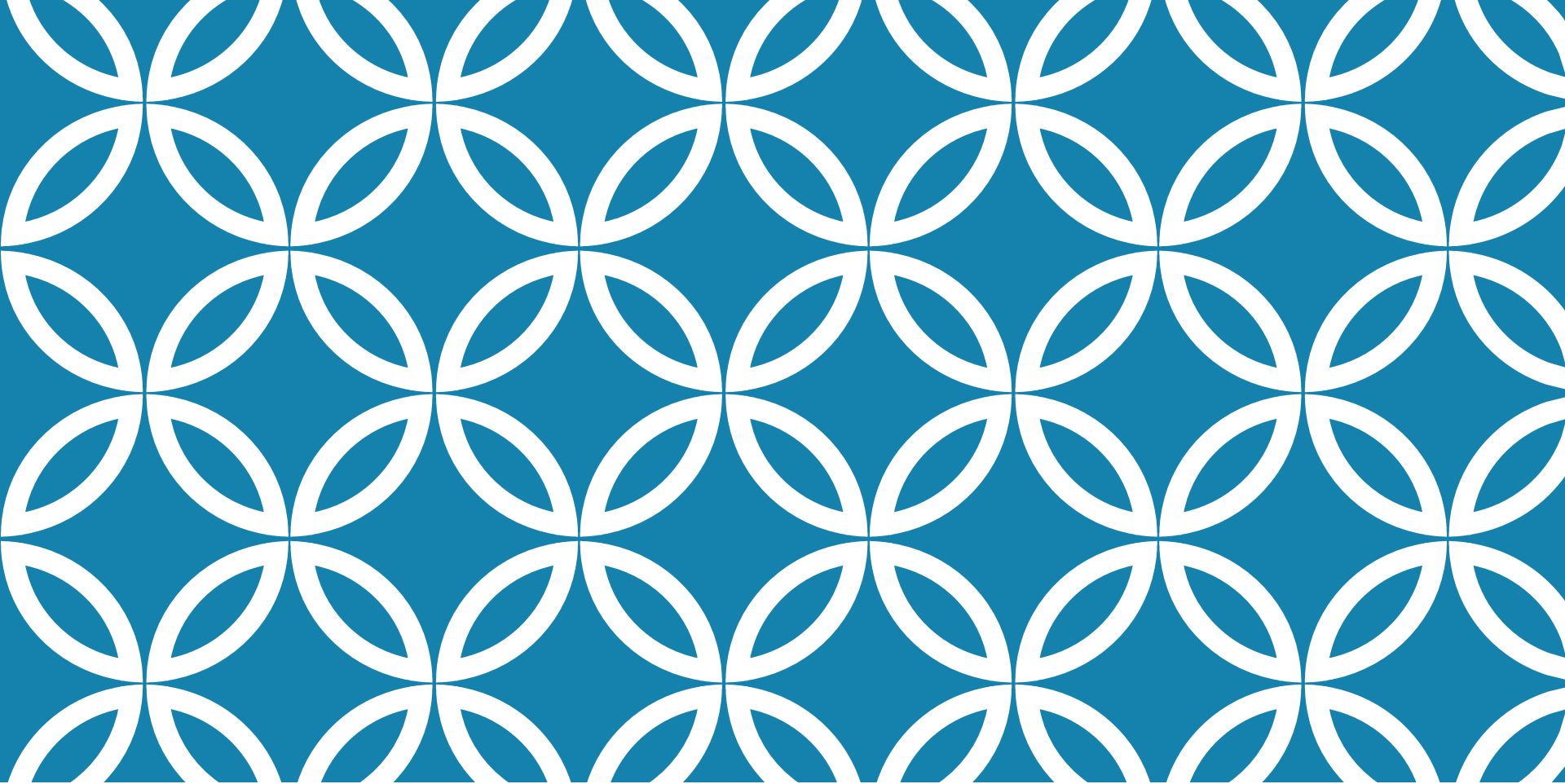
- ❑ Msdos.sys is the second program to load into RAM immediately after io.sys
 - ❑ It looks for the Config.sys file to configure device drivers and other settings
- ❑ Msdos.sys then loads Command.com
- ❑ As the loading of Command.com nears completion, Msdos.sys looks for and loads Autoexec.bat

OTHER DISK OPERATING SYSTEMS

- ❑ Control Program for Microprocessors (CP/M)
 - ❑ First nonspecific microcomputer OS
 - ❑ Created by Digital Research in 1970
 - ❑ 8-inch floppy drives; no support for hard drives
- ❑ Digital Research Disk Operating System (DR-DOS)
 - ❑ Developed in 1988 to compete with MS-DOS
 - ❑ Used FAT12 and FAT16 and had a richer command environment

OTHER DISK OPERATING SYSTEMS (CONTINUED)

- ❑ Personal Computer Disk Operating System (PC-DOS)
 - ❑ Created by Microsoft under contract for IBM
 - ❑ PC-DOS works much like MS-DOS



UNDERSTANDING VIRTUAL MACHINES



UNDERSTANDING VIRTUAL MACHINES

☐ **Virtual machine**

- ☐ Allows you to create a representation of another computer on an existing physical computer
- ☐ A virtual machine is just a few files on your hard drive
 - ☐ Must allocate space to it
- ☐ A virtual machine recognizes components of the physical machine it's loaded on
 - ☐ Virtual OS is limited by the physical machine's OS

UNDERSTANDING VIRTUAL MACHINES (CONTINUED)

- ❑ In computer forensics

- ❑ Virtual machines make it possible to restore a suspect drive on your virtual machine
 - ❑ And run nonstandard software the suspect might have loaded

- ❑ From a network forensics standpoint, you need to be aware of some potential issues, such as:

- ❑ A virtual machine used to attack another system or network

CREATING A VIRTUAL MACHINE

- ❑ Popular applications for creating virtual machines
 - ❑ VMware, Microsoft Virtual PC, Oracle VirtualBox
- ❑ Using Virtual PC
 - ❑ You must download and install Virtual PC first

CREATING A VIRTUAL MACHINE (CONTINUED)

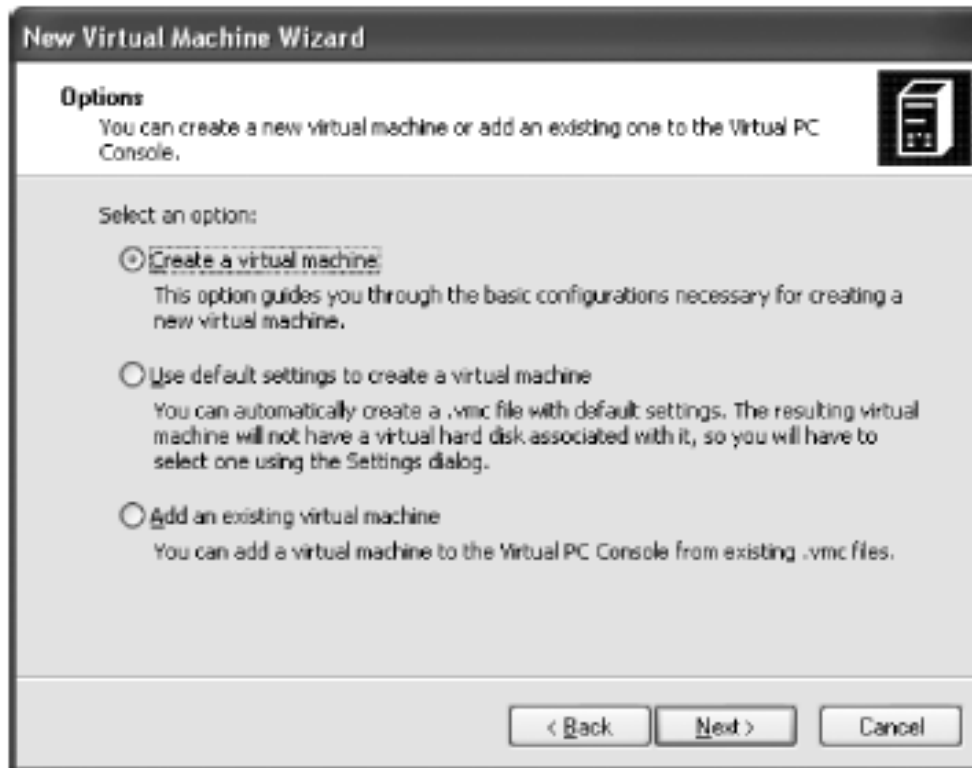


Figure 6-33 Creating a new virtual machine

CREATING A VIRTUAL MACHINE (CONTINUED)

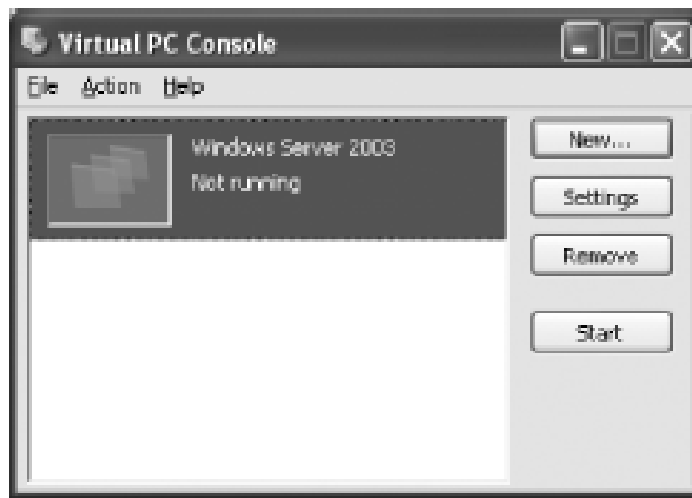


Figure 6-34 The Virtual PC Console with a virtual machine available

CREATING A VIRTUAL MACHINE (CONTINUED)

- ❑ You need an ISO image of an OS
 - ❑ Because no OSs are provided with Virtual PC
- ❑ Virtual PC creates two files for each virtual machine:
 - ❑ A .vhd file, which is the actual virtual hard disk
 - ❑ A .vmc file, which keeps track of configurations you make to that disk
- ❑ See what type of physical machine your virtual machine thinks it's running
 - ❑ Open the Virtual PC Console, and click Settings

CREATING A VIRTUAL MACHINE (CONTINUED)

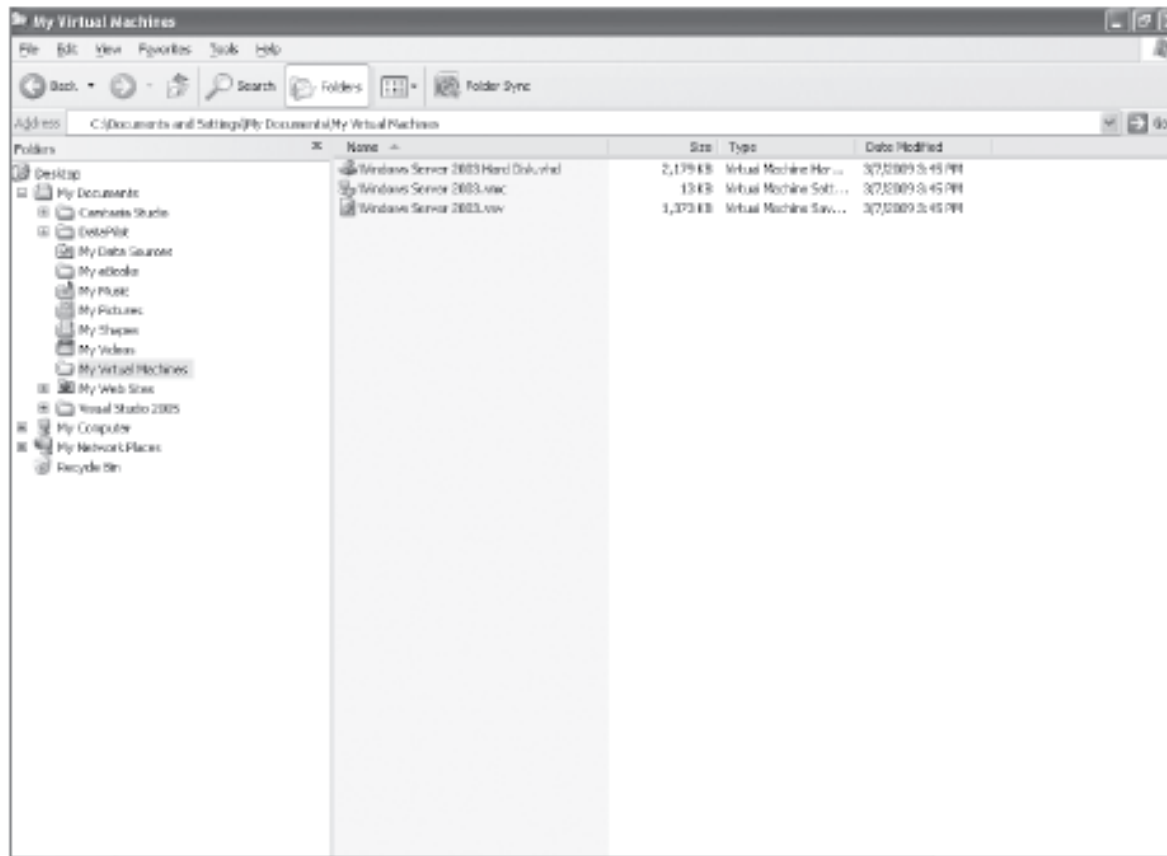
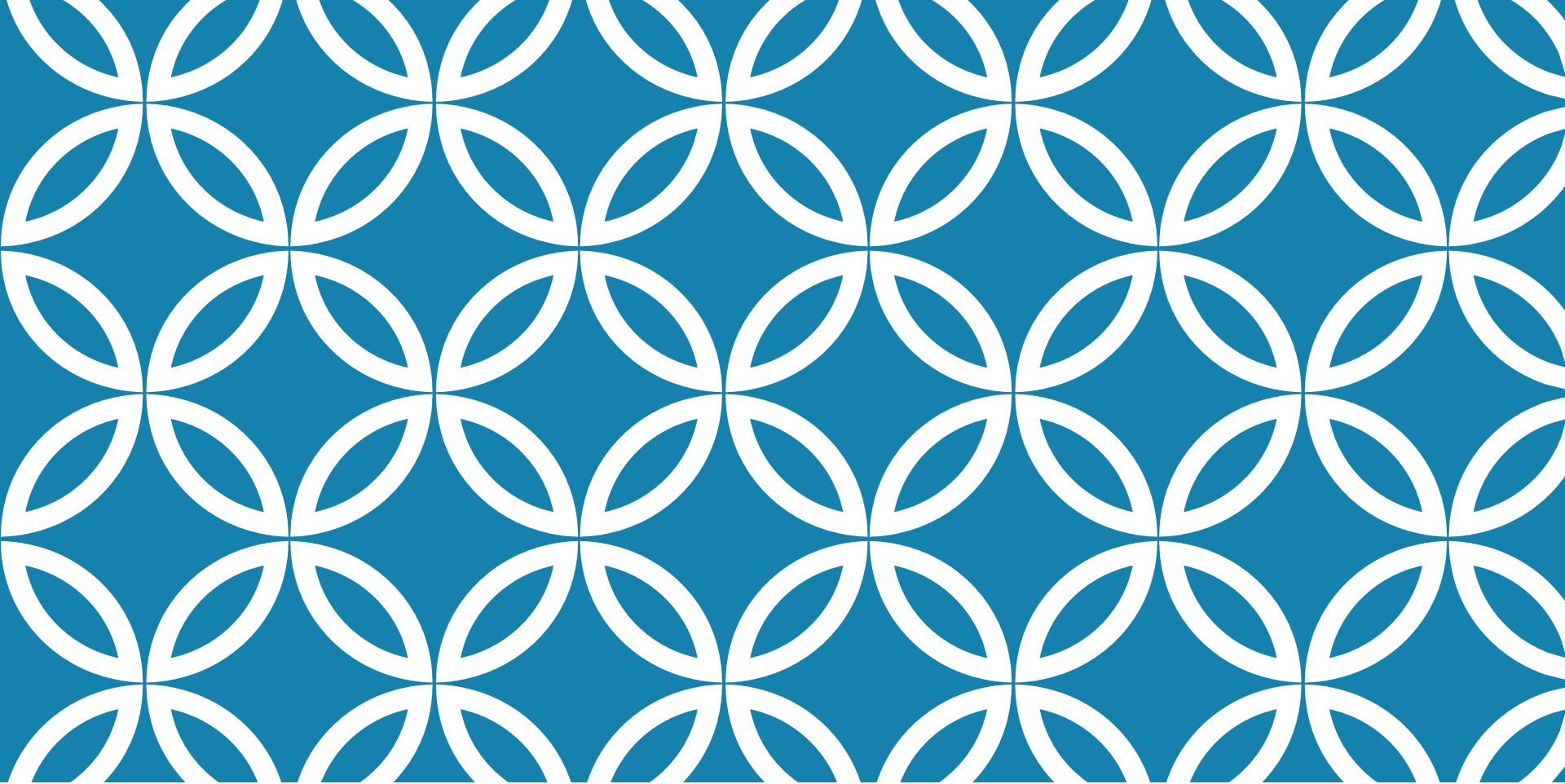


Figure 6-35 Virtual machine configuration files

CREATING A VIRTUAL MACHINE (CONTINUED)



Figure 6-36 Properties of a virtual machine



MACINTOSH AND LINUX BOOT PROCESSES AND FILE SYSTEMS

Chapter 7

OBJECTIVES

- ❑ Explain Macintosh file structures and the boot process
- ❑ Explain UNIX and Linux disk structures and boot processes
- ❑ Describe other disk structures



UNDERSTANDING THE MACINTOSH FILE STRUCTURE AND BOOT PROCESS

UNDERSTANDING THE MACINTOSH FILE STRUCTURE AND BOOT PROCESS

- ❑ Mac OS X version 10.5 - Leopard
 - ❑ Darwin core
 - ❑ **BSD UNIX** application layer

UNDERSTANDING THE MACINTOSH FILE STRUCTURE AND BOOT PROCESS

- ❑ Mac OS 9 & earlier used:

- ❑ **Hierarchical File System (HFS)**

- ❑ Files stored in nested directories (folders)

- ❑ **Extended Format File System (HFS+)**

- ❑ Introduced with Mac OS 8.1

- ❑ Supports smaller file sizes on larger volumes, resulting in more efficient disk use

UNDERSTANDING THE MACINTOSH FILE STRUCTURE AND BOOT PROCESS (CONTINUED)

- ❑ **File Manager** utility

- ❑ Reading, writing, and storing data to physical media

- ❑ **Finder**

- ❑ Keeps track of files and maintain users' desktops

- ❑ In older Mac OSs, a file consists of two parts:

- ❑ **Data fork** and **resource fork**

- ❑ Stores file metadata and application information

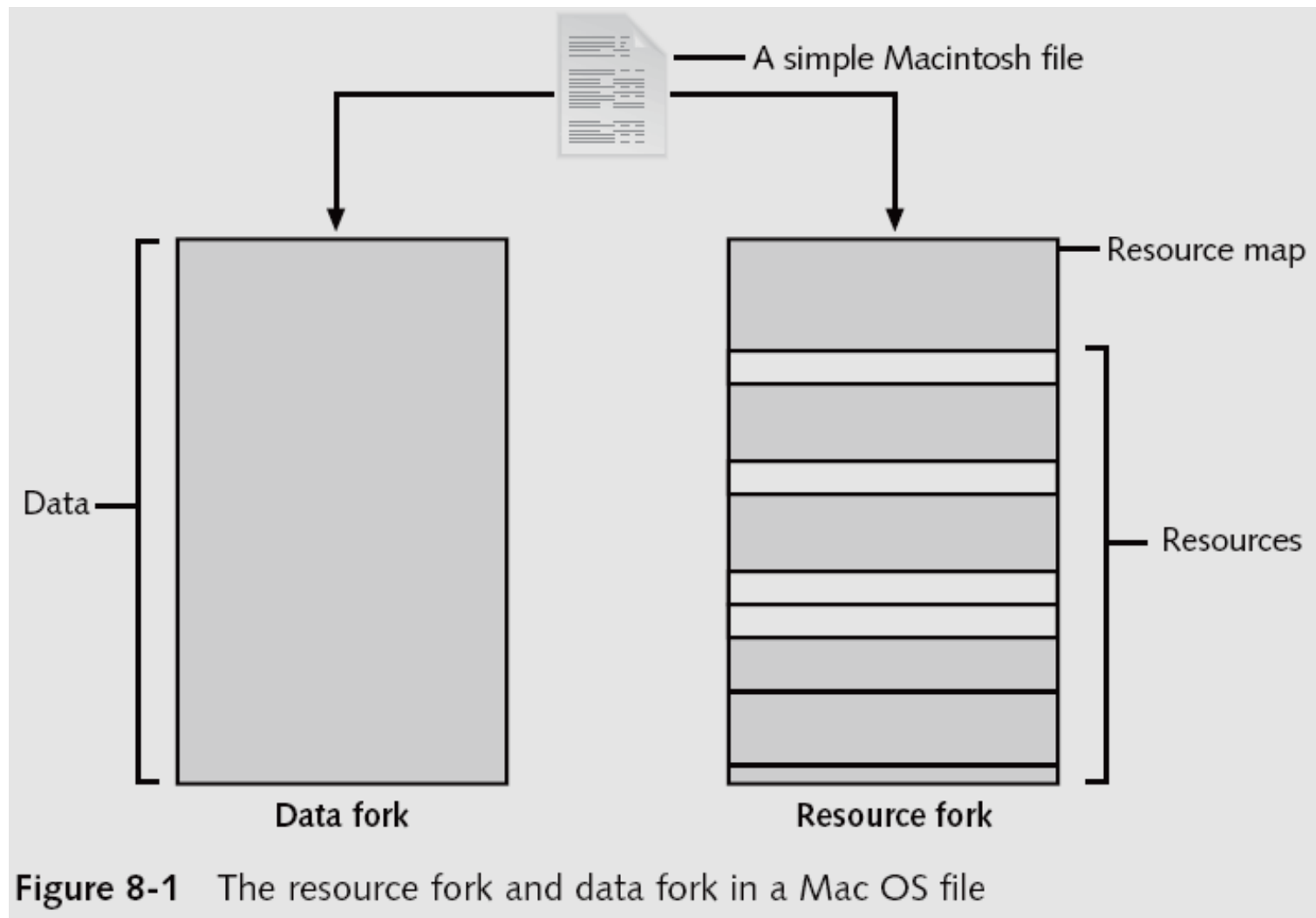


Figure 8-1 The resource fork and data fork in a Mac OS file

UNDERSTANDING MACINTOSH OS 9 VOLUMES

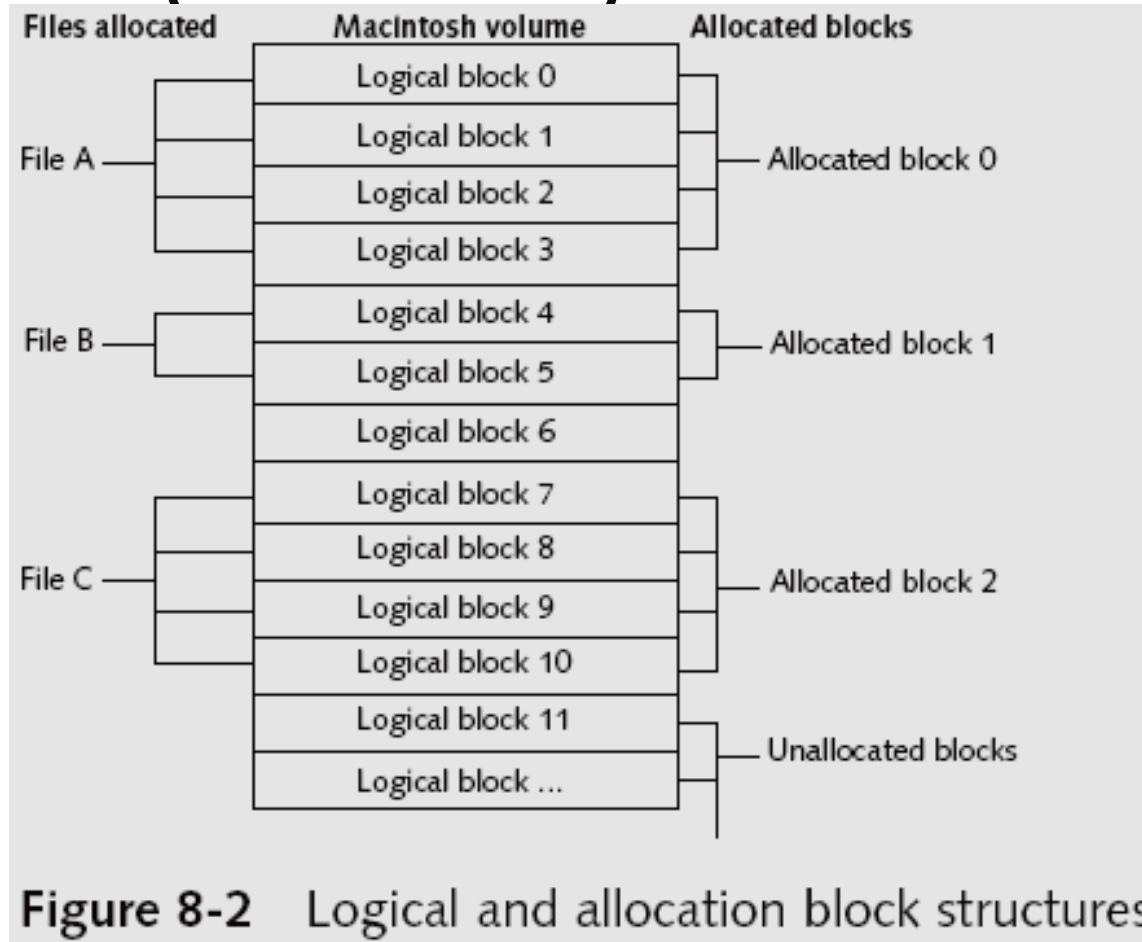
- ❑ A volume is any storage medium used to store files

- ❑ Can be all or part of a hard disk
- ❑ On a floppy disk is always the entire disk

❑ **Allocation and logical blocks**

- ❑ Logical blocks cannot exceed 512 bytes
- ❑ Allocation blocks are a set of consecutive logical blocks

UNDERSTANDING MACINTOSH OS 9 VOLUMES (CONTINUED)



UNDERSTANDING MACINTOSH OS 9 VOLUMES (CONTINUED)

☐ Two EOF descriptors

☐ Logical EOF

- ☐ Actual size of the file

☐ Physical EOF

- ☐ The number of allocation blocks for that file

☐ Clumps

- ☐ Groups of contiguous allocation blocks

- ☐ Used to reduce file fragmentation

- ☐ Volume fragmentation is kept to a minimum by adding more clumps to larger files

UNDERSTANDING MACINTOSH OS 9 VOLUMES (CONTINUED)

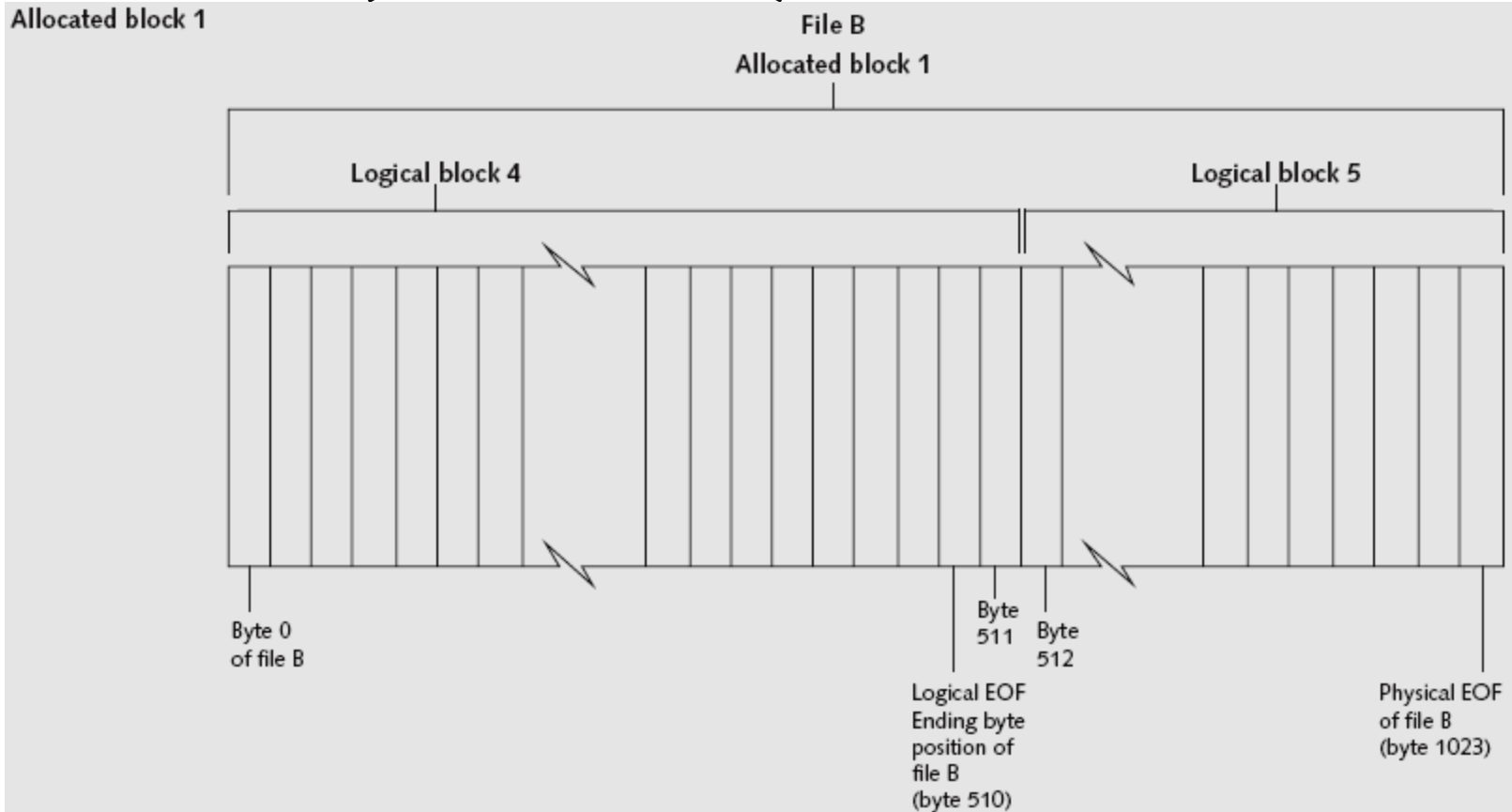


Figure 8-3 Logical EOF and physical EOF

EXPLORING MACINTOSH BOOT TASKS

- ❑ Use Open Firmware instead of BIOS
 - ❑ Processor- and system-independent firmware
 - ❑ Controls microprocessor after hardware initialization
- ❑ The boot process for OS 9 is as follows:
 1. Power on the computer
 2. Hardware self-test and Open Firmware run
 3. Macintosh OS starts
 4. The startup disk is located
 5. System files are opened
 6. System extensions are loaded
 7. OS 9 Finder starts

Table 8-1 HFS system files

HFS block position	HFS structure	Purpose of structure
0 1	Boot block	Startup volume containing boot instructions. Also stores system files and Finder information.
2	Master Directory Block (MDB)	Contains volume creation date and time and location of other system files, such as Volume Bitmap. A duplicate of this file called the Alternate MDB is located at the second-to-last block on the volume. Its purpose is to provide information to the OS disk utilities.
3	Volume Bitmap	Tracks used and unused blocks on the volume.
	Catalog	Lists all files and directories on the volume. It's a B*-tree file that uses the extents overflow file to coordinate all file allocations to the volume.
	Extents overflow file	Lists the extra extents, which are the allocated blocks used to store data files. It's a B*-tree file.

Table 8-2 HFS+ system files

HFS+ byte offset (fixed starting position)	HFS+ structure	Purpose of structure
0	Boot blocks	No change from HFS.
1024	Volume Information Block (VIB)	Replaces the MDB used in HFS.
Not fixed	Allocation file	Tracks available free blocks on the volume; replaces the HFS Volume Bitmap.
Not fixed	Extents overflow file	For files with more than eight extents, additional extents are recorded and managed through this B*-tree system file.
Not fixed	Catalog	Similar to an HFS catalog, this improved version allows up to eight extents for each file's forks. It's a B*-tree file.
Not fixed	Attributes file	Stores new file attribute information that isn't available in HFS. The new attributes are inline data attribute records, fork data attribute records, and extension attribute records.
Not fixed	Startup file	New to HFS+, this file can boot non-HFS and HFS+ volumes.
Not fixed	Alternate VIB	Same file as the HFS Alternate MDB.
	Reserved (512 bytes)	Last sector of the volume; used by Apple during manufacturing.

EXPLORING MACINTOSH BOOT TASKS (CONTINUED)

- ❑ Older Macintosh OSs use
 - ❑ First two logical blocks as boot blocks
 - ❑ **Master Directory Block (MDB)** or **Volume Information Block (VIB)**
 - ❑ Stores all information about a volume
 - ❑ **Volume Control Block (VCB)**
 - ❑ Stores information from the MDB when OS mounts
- ❑ **Extents overflow file**
 - ❑ Stores any file information not in the MDB or a VCB

EXPLORING MACINTOSH BOOT TASKS (CONTINUED)

☐ Catalog

- ☐ Listing of all files and directories on the volume
- ☐ Maintains relationships between files and directories

☐ Volume Bitmap

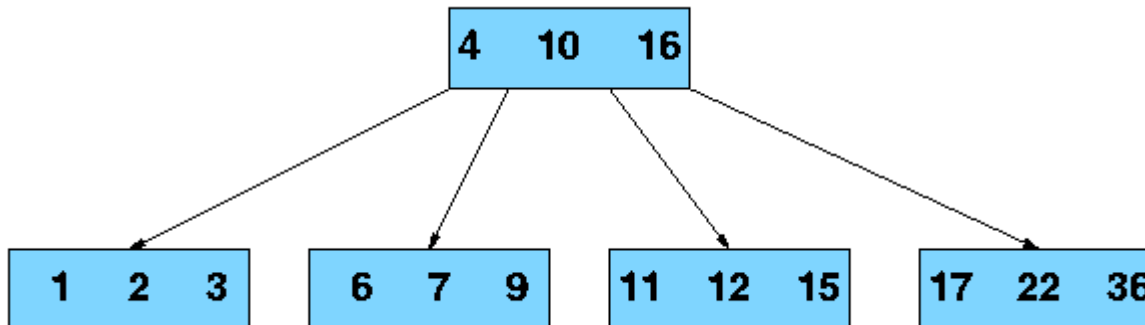
- ☐ Tracks used and unused blocks on a volume

☐ Mac OS 9 uses the **B*-tree** file system for File Manager

- ☐ Actual file data is stored on the **leaf nodes**
- ☐ B*-tree also uses header, index, and map nodes

B-TREE

- ❑ A way of storing records so they can be found rapidly
- ❑ Each node can only hold a few records, if more are added the node splits and the tree grows taller



USING MACINTOSH FORENSIC SOFTWARE

☐ Tools and vendors

- ☐ BlackBag Technologies
- ☐ SubRosaSoft MacForensicsLab
- ☐ Guidance EnCase
- ☐ X-Ways Forensics
- ☐ ProDiscover Forensic Edition
- ☐ Sleuth Kit and Autopsy

USING MACINTOSH FORENSIC SOFTWARE (CONTINUED)

☐ Macintosh Acquisition Methods

- ☐ Make an image of the drive
- ☐ Static acquisition of the suspect drive is preferable to a live acquisition
- ☐ Removing the drive from a Macintosh Mini's CPU case is difficult
 - ☐ Attempting to do so without Apple factory training could damage the computer
- ☐ Use a Macintosh-compatible forensic boot CD (or FireWire boot drive) to make an image on an external USB or FireWire drive

USING MACINTOSH FORENSIC SOFTWARE (CONTINUED)

- ❑ Macintosh Acquisition Methods (continued)
 - ❑ BlackBag Technologies sells acquisition products specifically designed for OS 9 and earlier
 - ❑ As well as OS X
 - ❑ MacQuisition is a forensic boot CD that makes an image of a Macintosh drive
 - ❑ After making an acquisition, examine the image of the file system
 - ❑ The tool you use depends on the image file format

USING MACINTOSH FORENSIC SOFTWARE (CONTINUED)

- ❑ Macintosh Acquisition Methods (continued)
 - ❑ BlackBag Technologies Macintosh Forensic Software and SubRosaSoft MacForensicsLab
 - ❑ Can disable/enable **Disk Arbitration**—which mounts drives
 - ❑ Being able to turn off the mount function in OS X
 - ❑ Allows you to connect a suspect drive to a Macintosh without a write-blocking device

USING MACINTOSH FORENSIC SOFTWARE (CONTINUED)

- ❑ Examining OS 9 Data Structures with BlackBag
 - ❑ Activities in this section assume you have a Macintosh running OS X
 - ❑ All data acquisitions (image files) must be configured as Disk Images
 - ❑ With the correct filename and extensions
 - ❑ To keep the correct order of each segment
 - ❑ Numbers need to be inserted between the filename and the extension

Table 8-3 Requirements for renaming Disk Image files

Original filenames for image file and segments	Macintosh Disk Image filenames
GCFI-OS9.001	GCFI-OS9.dmg
GCFI-OS9.002	GCFI-OS9.002.dmgpart
GCFI-OS9.003	GCFI-OS9.003.dmgpart
GCFI-OS9.004	GCFI-OS9.004.dmgpart

USING MACINTOSH FORENSIC SOFTWARE (CONTINUED)

Examining OS 9 Data Structures with BlackBag (continued)

- Load images as a virtual disk image double-clicking the files in Finder (Figure 8-4)
- OS X loads and displays an icon of the virtual mounted disk with the name “untitled” on the desktop
 - You can rename it with your case name
 - See Figure 8-5



Figure 8-4 OS X Finder showing the renamed raw files as .dmg files

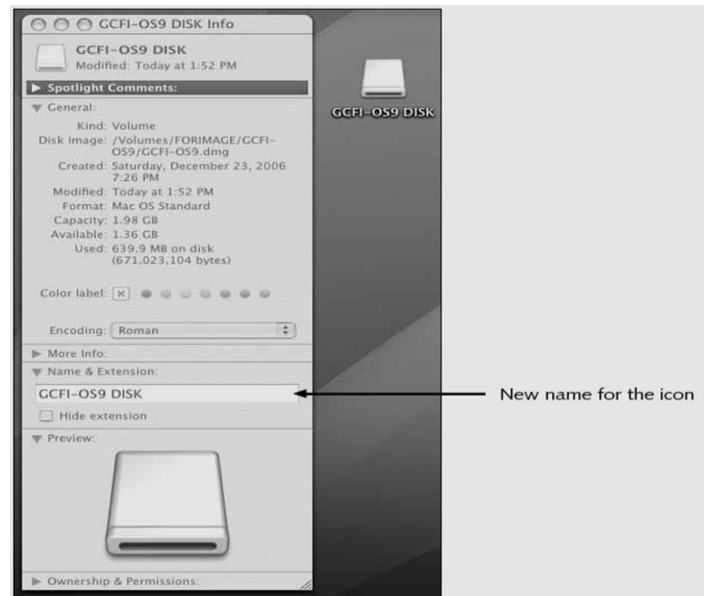


Figure 8-5 Changing the icon name

USING MACINTOSH FORENSIC SOFTWARE (CONTINUED)

- ❑ Examining OS 9 Data Structures with BlackBag (continued)
 - ❑ Start BlackBag from Finder
 - ❑ See Figure 8-6
 - ❑ BlackBag includes several utilities for conducting a full analysis of evidence, including
 - ❑ PDISKInfo, PMAPIInfo, DirectoryScan, FileSearch, MacCarver, and FileSpy



Figure 8-6 Starting BlackBag from Finder

USING MACINTOSH FORENSIC SOFTWARE (CONTINUED)

□ Examining OS 9 Data Structures with BlackBag (continued)

□ Activity 1:

- Use the BlackBag DirectoryScan utility, which lists all folders and files, visible and hidden, in the image loaded as a .dmg file

- See Figure 8-8

□ Activity 2:

- Use the FileSearcher utility to locate files by a specific extension

- See Figure 8-9

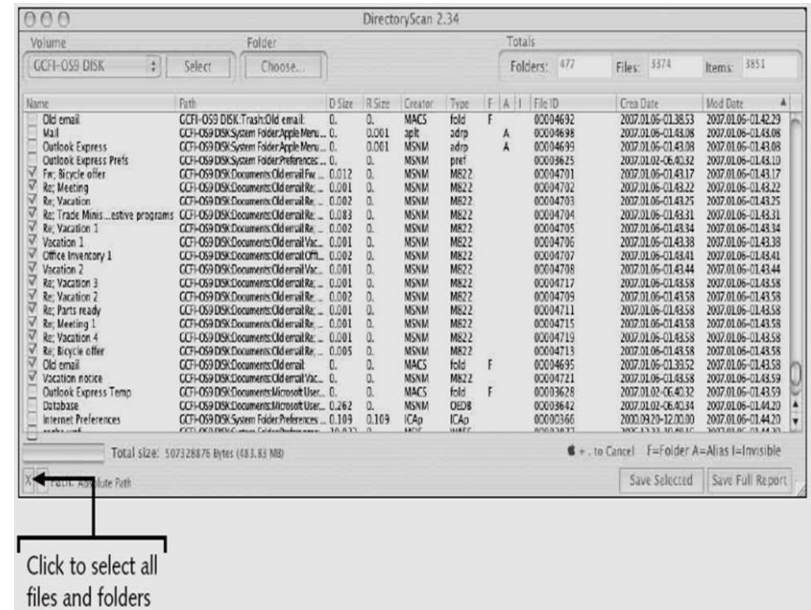


Figure 8-8 Selecting the entire GCFI-OS9 DISK volume in the DirectoryScan window

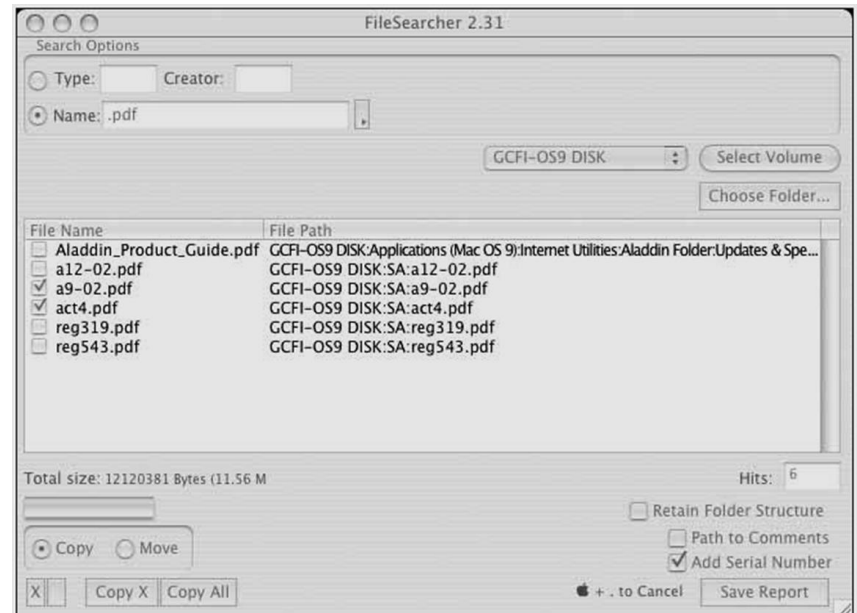
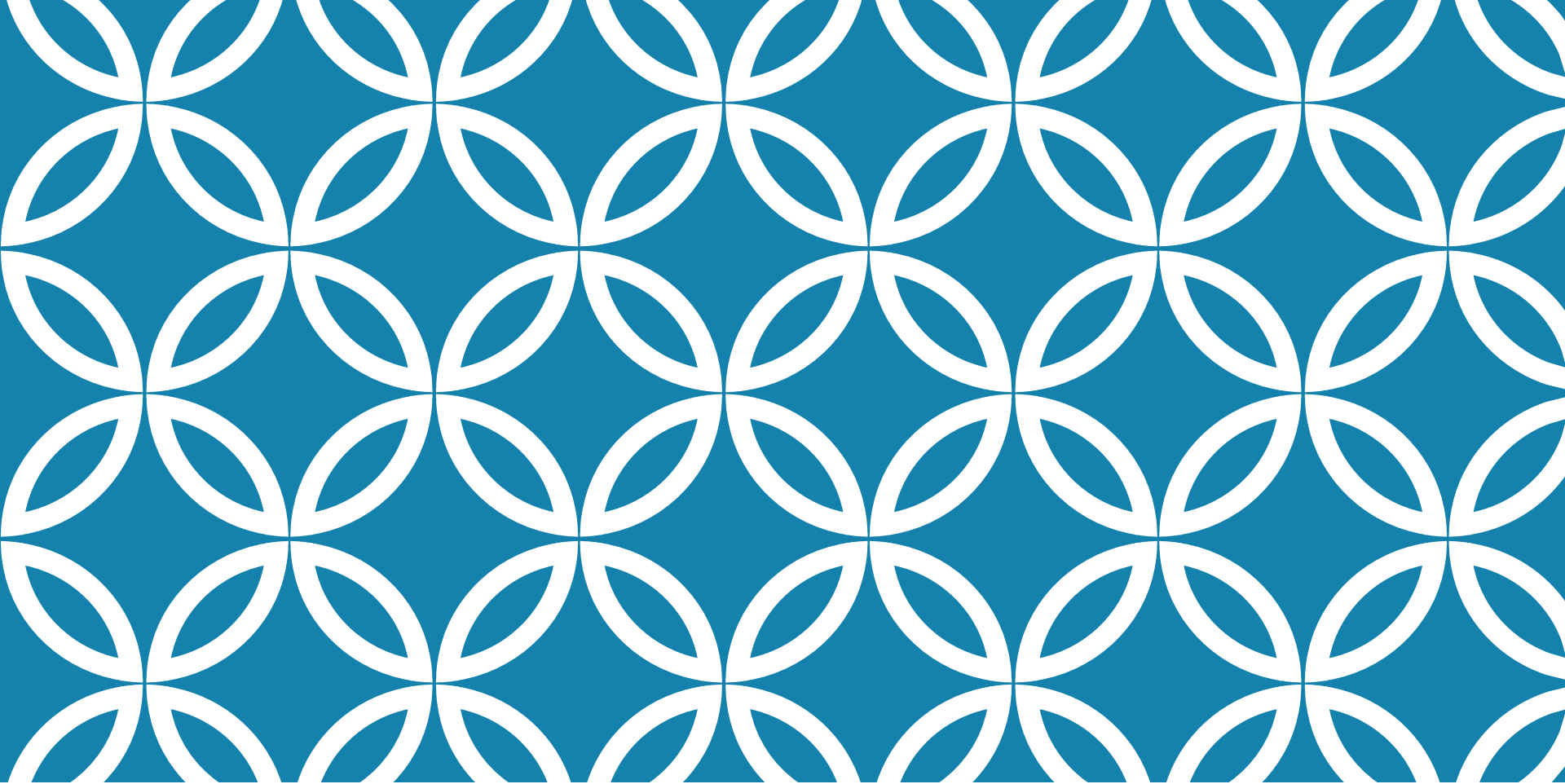


Figure 8-9 FileSearcher listing all .pdf documents in the search results



EXAMINING UNIX AND LINUX DISK STRUCTURES AND BOOT PROCESSES



EXAMINING UNIX AND LINUX DISK STRUCTURES AND BOOT PROCESSES

☐ UNIX flavors

- ☐ System V variants, Sun Solaris, IBM AIX, and HP-UX
- ☐ BSD, FreeBSD, OpenBSD, and NetBSD

☐ Linux distributions

- ☐ Red Hat, Fedora, Ubuntu, and Debian
- ☐ Most consistent UNIX-like OSs

☐ Linux kernel is regulated under the **GNU General Public License (GPL)** agreement

EXAMINING UNIX AND LINUX DISK STRUCTURES AND BOOT PROCESSES (CONTINUED)

- ❑ BSD license is similar to the GPL
 - ❑ But makes no requirements for derivative works
- ❑ Some useful Linux commands to find information about your Linux system
 - ❑ `uname -a` (displays the name of the system)
 - ❑ `ls -l` (list all files)
 - ❑ `ls -ul filename`
 - ❑ `netstat -s` (identifies other systems connected the network to a UNIX or Linux system)

Table 8-4 UNIX system files

OS	System files	Purpose
AIX	/etc/exports	Configuration file
	/etc/filesystems	File system table of devices and mount points
	/etc/utmp	Current user's logon information
	/var/adm/wtmp	Logon and logoff history information
	/etc/security/lastlog	User's last logon information
	/var/adm/sulog	Substitute user attempt information
	/etc/group	Group memberships for the local system
	/var/log/syslog	System messages log
	/etc/security/passwd	Master password file for the local system
	/etc/security/failedlogin	Failed logon attempt information
HP-UX	/etc/utmp and /etc/utmpx	Current user's logon information
	/var/adm/wtmp and /var/adm/wtmpx	Logon and logoff history information
	/var/adm/btmp	Failed logon attempt information
	/etc/fstab	File system table of devices and mount points
	/etc/checklist	File system table information (version 9.x)
	/etc/exports	Configuration files
	/etc/passwd	Master password file for the local system
	/etc/group	Group memberships for the local system
	/var/adm/syslog.log	System messages log
	syslog	System log files
	/var/adm/sulog	Substitute user attempt information

Table 8-4 UNIX system files (continued)

OS	System files	Purpose
IRIX	/var/adm/syslog	System log files
	/etc/exports	Configuration files
	/etc/fstab	File system table of devices and mount points
	/var/adm/btmp	Failed logon information
	/var/adm/lastlog	User's last logon information
	/var/adm/wtmp and /var/adm/wtmpx	Logon and logoff history information
	/var/adm/sulog	Substitute user attempt information
	/etc/shadow	Master password file for the local system
	/etc/group	Group memberships for the local system
	/var/adm/utmp and /var/adm/utmpx	Current user's logon information
Linux	/etc/exports	Configuration files
	/etc/fstab	File system table of devices and mount points
	/var/log/lastlog	User's last logon
	/var/log/wtmp	Logon and logoff history information
	/var/run/utmp	Current user's logon information
	/var/log/messages	System messages log
	/etc/shadow	Master password file for the local system
	/etc/group	Group memberships for the local system
Solaris	/etc/passwd	Account information for local system
	/etc/group	Group information for local system
	/var/adm/sulog	Switch user log data
	/var/adm/utmp	Logon information
	/var/adm/wtmp, /var/adm/wtmpx, and /var/adm/lastlog	Logon history information
	/var/adm/loginlog	Failed logon information
	/var/adm/messages	System log files
	/etc/vfstab	Static file system information
	/etc/dfs/dfstab and /etc/vfstab	Configuration files

EXAMINING UNIX AND LINUX DISK STRUCTURES AND BOOT PROCESSES (CONTINUED)

- ❑ Linux file systems
 - ❑ **Second Extended File System (Ext2fs)**
 - ❑ Ext3fs, journaling version of Ext2fs
- ❑ Employs **inodes**
 - ❑ Contain information about each file or directory
 - ❑ Pointer to other inodes or blocks
 - ❑ Keep internal link count
 - ❑ Deleted inodes have count value 0

UNIX AND LINUX OVERVIEW

- ❑ Everything is a file
 - ❑ Including disks, monitors, NIC, RAM
 - ❑ Files are objects with properties and methods
- ❑ UNIX consists of four components
 - ❑ Boot block
 - ❑ Superblock
 - ❑ inode block
 - ❑ Data block

BOOT BLOCK AND SUPERBLOCK

❑ Boot block

- ❑ Block is a disk allocation unit of at least 512 bytes
- ❑ Contains the bootstrap code
- ❑ UNIX/Linux computer has only one boot block, located on the main hard disk

❑ Superblock

- ❑ Indicates disk geometry, available space, location of the first inode, and free inode list
- ❑ Manages the file system
- ❑ Multiple copies of the superblock are kept

INODE BLOCKS AND DATA BLOCKS

☐ inode blocks

- ☐ First data after the superblock
- ☐ An inode is assigned to every file allocation unit

☐ Data blocks

- ☐ Where directories and files are stored
- ☐ This location is linked directly to inodes
- ☐ Each sector contains 512 bytes
- ☐ Each data block contains 1024-4096 bytes
- ☐ Analogous to a cluster on a FAT or NTFS volume

UNIX AND LINUX OVERVIEW (CONTINUED)

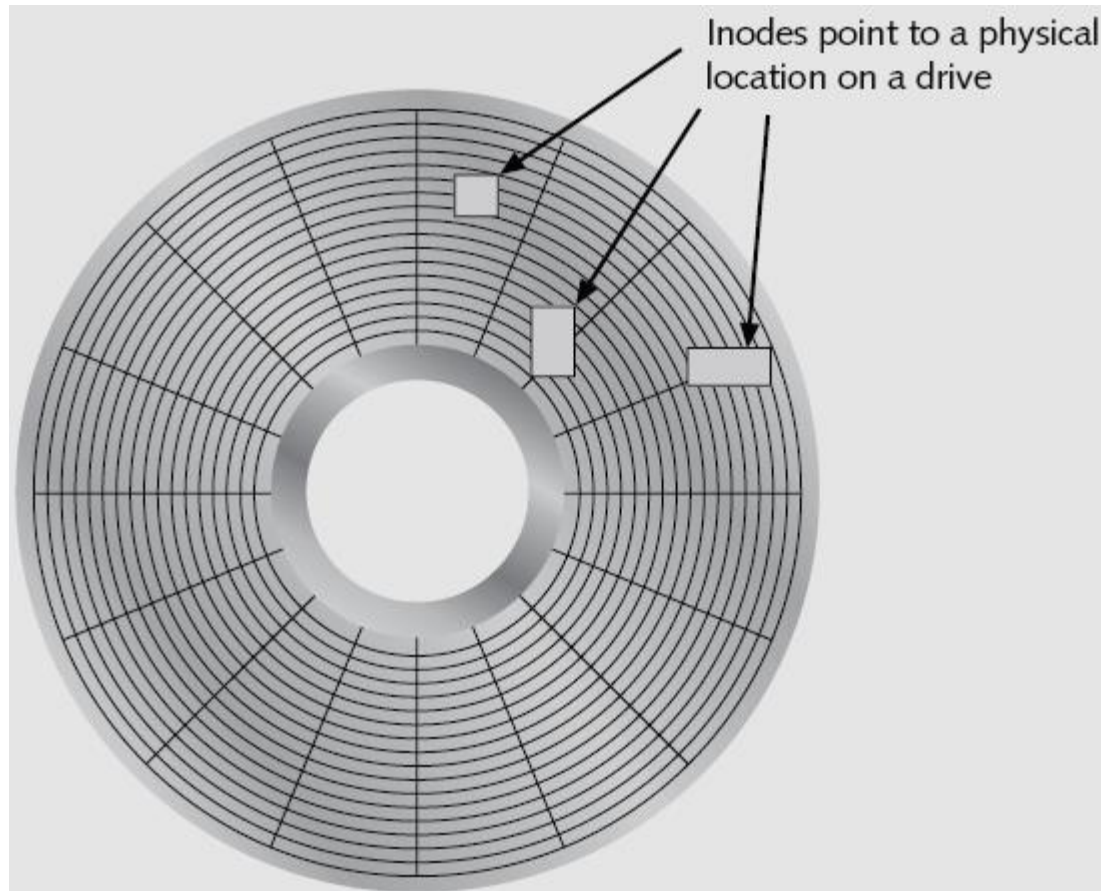


Figure 8-10 Clustering data blocks to save a file in Linux

UNIX AND LINUX OVERVIEW (CONTINUED)

☐ **Bad block inode**

- ☐ Keeps track of disk's bad sectors
- ☐ Commands: badblocks, mke2fs, and e2fsck

☐ **Linux ls command displays information about files and directories**

- ☐ lowercase LS

☐ **For details, use the `ls -l` command**

- ☐ lowercase LS -L

UNIX AND LINUX OVERVIEW (CONTINUED)

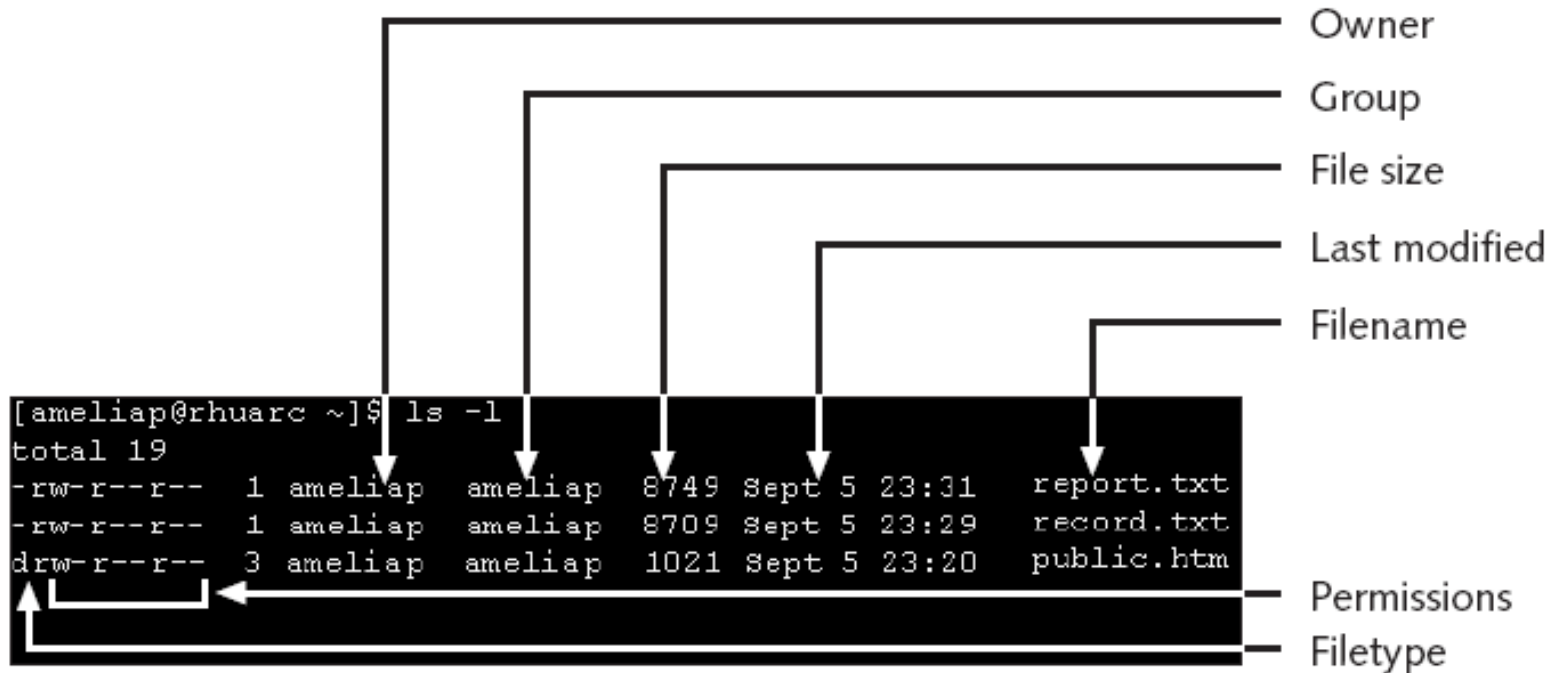


Figure 8-11 Finding information about a file

CONTINUATION INODE

☐ Continuation inode

- ☐ Provides information about a file or directory
 - ☐ Mode and file type, the quantity of links in the file or directory, the file or directory status flag
- ☐ Sticky bit
 - ☐ Used in some old Unix versions to make programs load faster by keeping parts of the program in RAM
 - ☐ Used in modern Unix systems to prevent users from deleting files owned by others

UNIX AND LINUX OVERVIEW (CONTINUED)

Table 8-5 Code values for an inode

Code values	Description
4000	UID on execution—set
2000	GID on execution—set
1000	Sticky bit—set
0400	Read by owner—allowed
0200	Write by owner—allowed
0100	Execution/search by owner—allowed
0040	Read by group—allowed
0020	Write by group—allowed
0010	Execution/search by group—allowed
0004	Read by others—allowed
0002	Write by others—allowed
0001	Execution/search by others—allowed

UNDERSTANDING INODES

- ❑ Link data stored in data blocks (usually 1024 bytes)
- ❑ Ext2fs and Ext3fs are improvements over Ext
 - ❑ Data recovery easier on Ext3fs than on Ext2fs
- ❑ First inode has 13 pointers
 - ❑ Pointers 1 to 10 are direct pointers to data storage blocks
 - ❑ Pointer 11 is an **indirect pointer**
 - ❑ Pointer 12 is a **double-indirect pointer**
 - ❑ Pointer 13 is a **triple-indirect pointer**
 - ❑ Pointers 11-13 are needed for large files



Figure 8-12 Inode pointers in the Linux file system

UNDERSTANDING INODES (CONTINUED)

Table 8-6 UNIX and Linux shell commands

Shell command	Associated options	Purpose
<code>cat file</code> <code>more file</code>		Displays the contents of a file (similar to the MS-DOS Type command)
<code>dd</code>	Refer to man pages for available options	Copies a disk drive by blocks, which is the same as creating an image of a disk drive
<code>df</code> <code>bdf (HP-UX)</code>	-k (Solaris)	Displays partition information for local or NFS mounted partitions
<code>find</code>	Refer to man pages for available options	Locates files matching a specific attribute, such as name, last modification time, or owner
<code>netstat</code>	-a	Identifies other systems connected via the network to a UNIX or Linux system
<code>ps</code>	ax (BSD) -ef (System V)	Displays the status of OS processes
<code>uname</code>	-a	Displays the name of the system

UNDERSTANDING UNIX AND LINUX BOOT PROCESSES

- ❑ Instruction code in firmware is loaded into RAM

- ❑ This is called **memory-resident** code because it is stored in ROM

- ❑ Instruction code then:

- ❑ Checks the hardware

- ❑ Load the boot program

- ❑ Boot program

- ❑ Loads kernel

- ❑ Transfers control to kernel

- ❑ Kernel's first task is to identify all devices

UNDERSTANDING UNIX AND LINUX BOOT PROCESSES (CONTINUED)

☐ Kernel

- ☐ Boots system on single-user mode
- ☐ Runs startup scripts
- ☐ Changes to multiuser mode, then user logs on
- ☐ Identifies root directory, swap, and dump files
- ☐ Sets hostname and time zone
- ☐ Runs consistency checks on the file system and mounts partitions
- ☐ Starts services and sets up the NIC
- ☐ Establishes user and system accounting and quotas

UNDERSTANDING LINUX LOADER AND GRUB

☐ Linux Loader (LILO)

- ☐ Old boot manager
- ☐ Can start two or more OSs
- ☐ Uses configuration file `/etc/lilo.conf`

☐ Grand Unified Boot Loader (GRUB)

- ☐ More powerful than LILO
- ☐ As LILO, it resides on MBR
- ☐ Command line or menu driven

UNDERSTANDING UNIX AND LINUX DRIVES AND PARTITION SCHEMES

- ❑ Labeled as path starting at root (/) directory
 - ❑ Primary master disk (/dev/hda)
 - ❑ First partition is /dev/hda1
 - ❑ Second partition is /dev/hda2
 - ❑ Primary slave or secondary master or slave (/dev/hdb, /dev/hdc, or /dev/hdd)
 - ❑ First partition is /dev/hdb2
- ❑ SCSI controllers
 - ❑ /dev/sda with first partition /dev/sda1
 - ❑ Linux treats SATA, USB, and FireWire devices the same way as SCSI devices

EXAMINING UNIX AND LINUX DISK STRUCTURES

- ❑ Most commercial computer forensics tools can analyze UNIX UFS and UFS2
 - ❑ And Linux Ext2, Ext3, ReiserFS, and Reiser4 file systems
- ❑ Freeware tools include Sleuth Kit and its Web browser interface, Autopsy Browser
- ❑ Foremost
 - ❑ A freeware carving tool that can read many image file formats
 - ❑ Configuration file: foremost.conf

EXAMINING UNIX AND LINUX DISK STRUCTURES (CONTINUED)

☐ Tarball

- ☐ A data file containing one or more files or whole directories and their contents

☐ Installing Sleuth Kit and Autopsy

- ☐ Requires downloading and installing the most recent updates of these tools
- ☐ Download the most current source code from www.sleuthkit.org
- ☐ To run Sleuth Kit and Autopsy Browser, you need to have root privileges

EXAMINING UNIX AND LINUX DISK STRUCTURES (CONTINUED)

```
[joe@fridaypi ~]$ cd /usr/local/autopsy-2.08
[joe@fridaypi autopsy-2.08]$ su
Password: *****
[joe@fridaypi autopsy-2.08]$ ./autopsy

=====

                Autopsy Forensic Browser
            http://www.sleuthkit.org/autopsy/
                ver 2.08

=====

Evidence Locker: /home/joe/work
Start Time: Mon Jan 22 07:55:33 2007
Remote Host: localhost
Local Port: 9999

Open an HTML browser on the remote host and paste this URL in it

    http://localhost:9999/autopsy

Keep this process running and use <ctrl-c> to exit
|
```

Figure 8-13 Starting Autopsy from a Linux terminal window

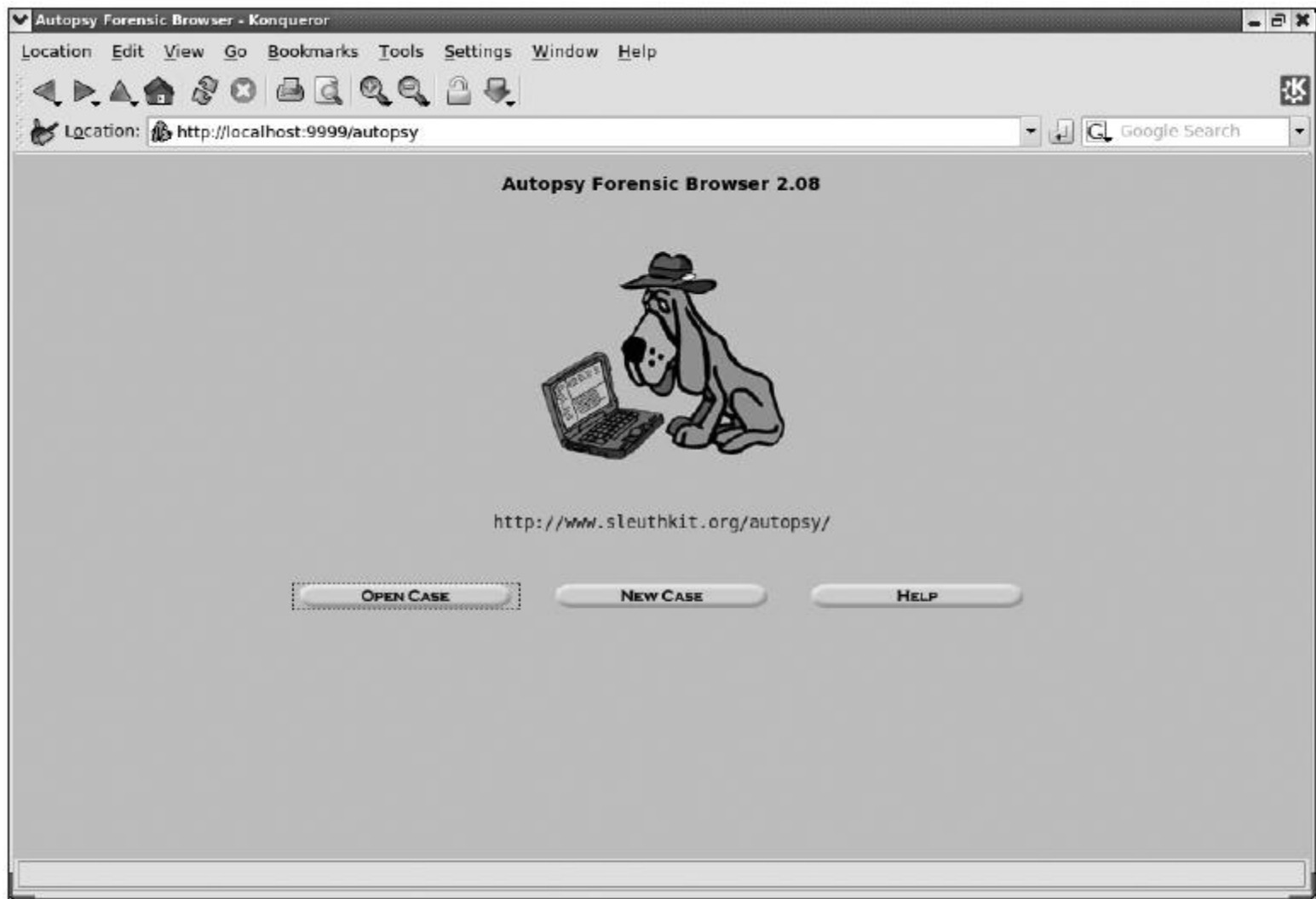
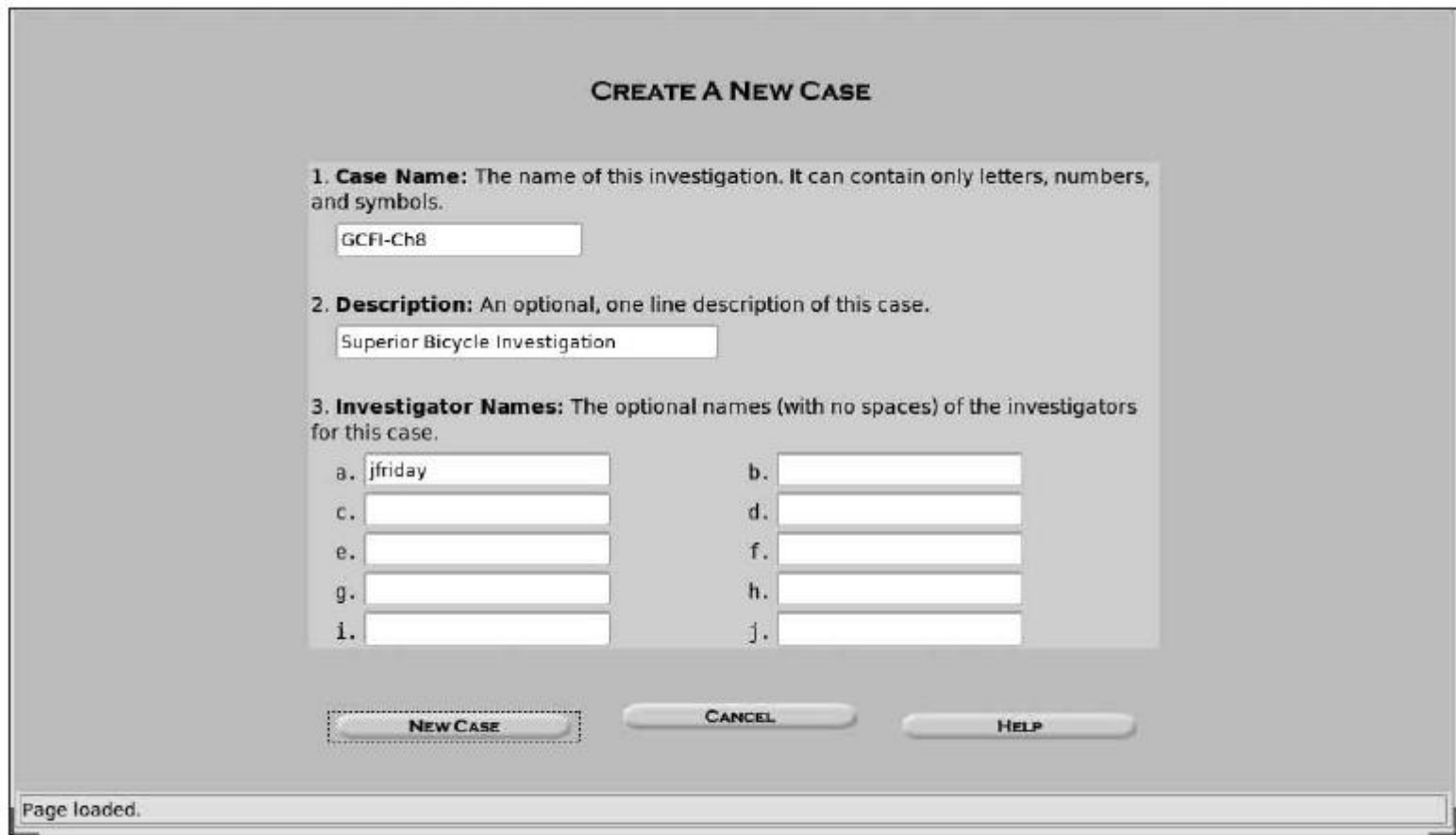


Figure 8-14 The Autopsy main window

EXAMINING UNIX AND LINUX DISK STRUCTURES (CONTINUED)

- Examining a case with Sleuth Kit and Autopsy
 - Use Sleuth Kit and Autopsy Browser to analyze a Linux Ext2 and Ext3 file system
 - See Figures 8-15 through 8-18

EXAMINING UNIX AND LINUX DISK STRUCTURES (CONTINUED)



CREATE A NEW CASE

1. Case Name: The name of this investigation. It can contain only letters, numbers, and symbols.

2. Description: An optional, one line description of this case.

3. Investigator Names: The optional names (with no spaces) of the investigators for this case.

a. <input type="text" value="jfriday"/>	b. <input type="text"/>
c. <input type="text"/>	d. <input type="text"/>
e. <input type="text"/>	f. <input type="text"/>
g. <input type="text"/>	h. <input type="text"/>
i. <input type="text"/>	j. <input type="text"/>

Page loaded.

Figure 8-15 The Create A New Case dialog box

EXAMINING UNIX AND LINUX DISK STRUCTURES (CONTINUED)

ADD A NEW HOST

1. **Host Name:** The name of the computer being investigated. It can contain only letters, numbers, and symbols.

2. **Description:** An optional one-line description or note about this computer.

3. **Time zone:** An optional timezone value (i.e. EST5EDT). If not given, it defaults to the local setting. A list of time zones can be found in the help files.

4. **Timeskew Adjustment:** An optional value to describe how many seconds this computer's clock was out of sync. For example, if the computer was 10 seconds fast, then enter -10 to compensate.

5. **Path of Alert Hash Database:** An optional hash database of known bad files.

6. **Path of Ignore Hash Database:** An optional hash database of known good files.

Figure 8-16 The Add A New Host dialog box

EXAMINING UNIX AND LINUX DISK STRUCTURES (CONTINUED)



Figure 8-17 The Keyword Search dialog box

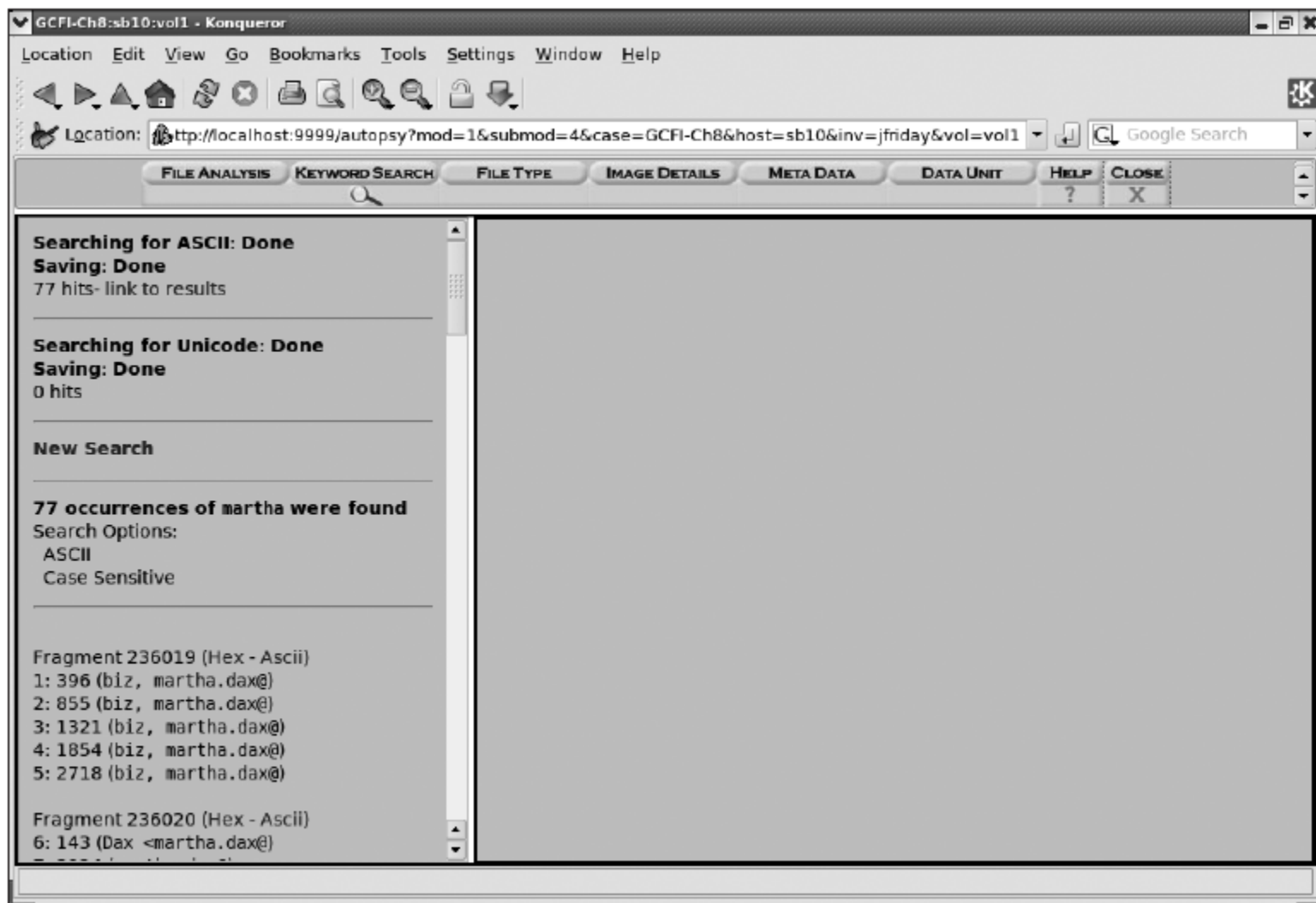


Figure 8-18 Summary of search results

EXAMINING UNIX AND LINUX DISK STRUCTURES (CONTINUED)

- Examining a case with Sleuth Kit and Autopsy (continued)

- Use the File Activity Time Lines function

- Identifies what files were active at a specific time

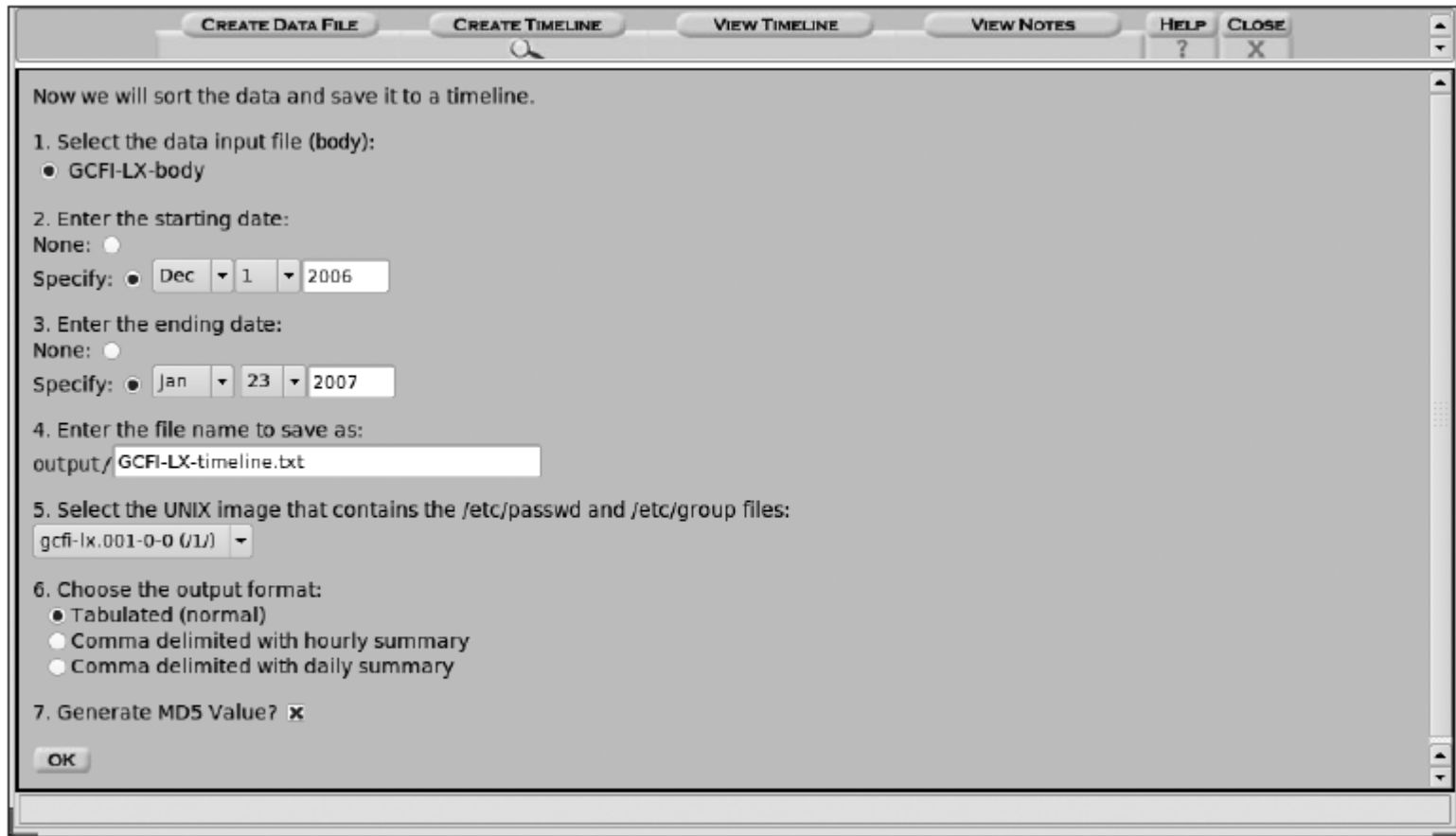
- See Figures 8-19 and 8-20

EXAMINING UNIX AND LINUX DISK STRUCTURES (CONTINUED)



Figure 8-19 The Select a volume to analyze or add a new image file dialog box

EXAMINING UNIX AND LINUX DISK STRUCTURES (CONTINUED)



The screenshot shows a software window with a title bar containing buttons for 'CREATE DATA FILE', 'CREATE TIMELINE', 'VIEW TIMELINE', 'VIEW NOTES', 'HELP', and 'CLOSE'. The main area of the window contains the following text and form elements:

Now we will sort the data and save it to a timeline.

1. Select the data input file (body):
☒ GCFI-LX-body

2. Enter the starting date:
None: ☐
Specify: ☒ Dec 1 2006

3. Enter the ending date:
None: ☐
Specify: ☒ Jan 23 2007

4. Enter the file name to save as:
output/ GCFI-LX-timeline.txt

5. Select the UNIX image that contains the /etc/passwd and /etc/group files:
gcfi-lx.001-0-0 (1/1)

6. Choose the output format:
☒ Tabulated (normal)
☐ Comma delimited with hourly summary
☐ Comma delimited with daily summary

7. Generate MD5 Value? ☒

OK

Figure 8-20 Entering timeline options



UNDERSTANDING OTHER DISK STRUCTURES

UNDERSTANDING OTHER DISK STRUCTURES

- ❑ CDs and DVDs
- ❑ SCSI disks
- ❑ IDE/EIDE disks
- ❑ SATA drives

EXAMINING CD DATA STRUCTURES

- ❑ Laser burns flat areas (lands)
- ❑ Lower areas are called pits
- ❑ Transitions
 - ❑ From lands to pits have binary value 1 (on)
 - ❑ No transition has binary value 0 (off)
- ❑ **International Organization of Standards (ISO)**
 - ❑ ISO 9660 for CD, CD-R and CD-RW
 - ❑ ISO 13346 for DVDs

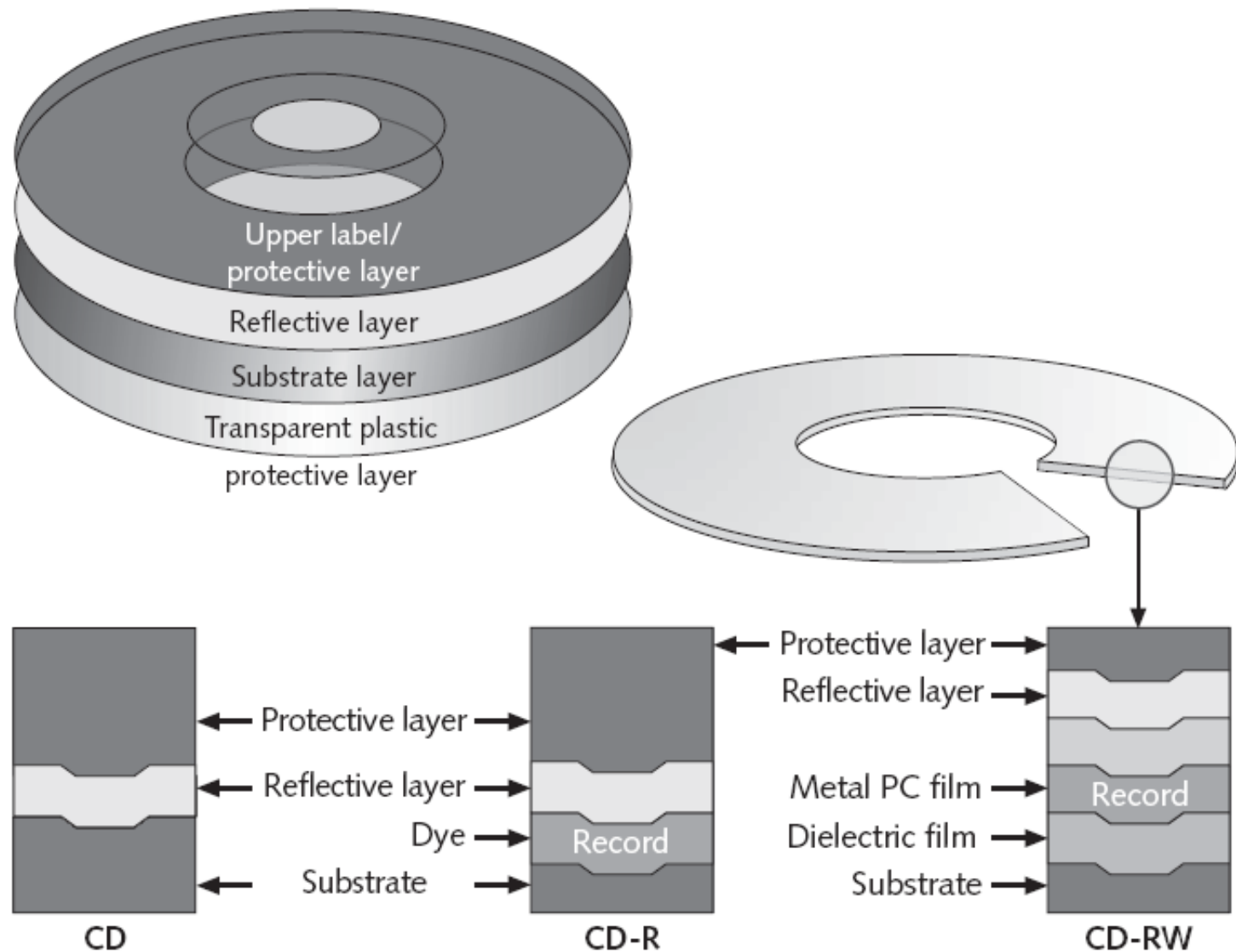


Figure 8-21 Physical makeup of a CD

EXAMINING CD DATA STRUCTURES (CONTINUED)

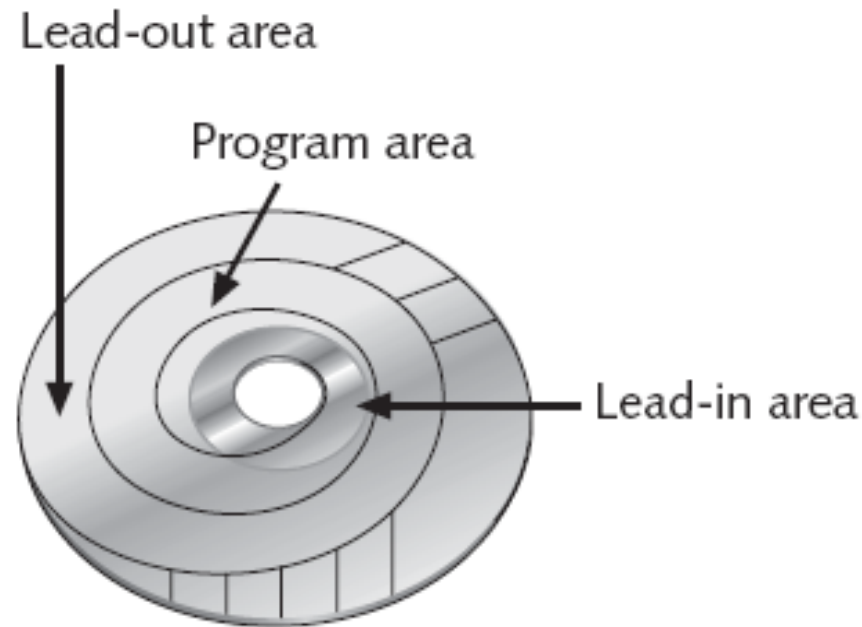


Figure 8-22 Logical layout of a CD

EXAMINING CD DATA STRUCTURES (CONTINUED)

- ❑ 99 tracks available in the lead-in area, for the table of contents
- ❑ Program area also has 99 tracks available for data

EXAMINING CD DATA STRUCTURES (CONTINUED)

- ❑ Frame is the unit storage
 - ❑ Contains 24 17-bits symbols
- ❑ Frames are combined into blocks
- ❑ Blocks are combined into sectors
 - ❑ 2352 bytes for CD-DA (music CDs)
 - ❑ 2048 bytes for CD (data CDs)
- ❑ **Constant Linear Velocity** ($< 12X$)
- ❑ **Constant Angular Velocity** ($\geq 12X$)

EXAMINING CD DATA STRUCTURES (CONTINUED)

- ❑ DVD disk file structures use a Universal Disk Format (UDF)
 - ❑ Called Micro-UDF (M-UDF)
- ❑ For backward compatibility, some DVDs have integrated ISO 9660
 - ❑ To allow compatibility with current OSs

EXAMINING SCSI DISKS

☐ **Small Computer System Interface (SCSI)**

- ☐ Provides a common bus communication device

☐ **During investigation**

- ☐ Check if the device is internal or external
- ☐ Check if card, cables, adapters, terminators, and drivers are available

☐ **Advance SCSI Programming Interface (ASPI)**

- ☐ Provides several software drivers for communication between the OS and SCSI component

EXAMINING SCSI DISKS (CONTINUED)

- ❑ Might need to adjust settings
 - ❑ Port numbers and terminators
- ❑ Newer SCSI devices typically use an integrated self-terminator
- ❑ One problem with older SCSI drives is identifying which jumper group terminates and assigns a port number

EXAMINING IDE/EIDE AND SATA DEVICES

- ❑ Most forensic disk examinations involve EIDE and SATA drives
- ❑ SATA drives from ATA-33 to ATA-133
 - ❑ Standard 40-pin ribbon or shielded cable
 - ❑ 40-pin/80-wire cable for ATA-66, 100, and 133
- ❑ CMOS identifies proper disk settings using:
 - ❑ Logical block addressing (LBA)
 - ❑ Enhanced CHS configurations
- ❑ Can be a problem during an investigation

EXAMINING IDE/EIDE AND SATA DEVICES (CONTINUED)

☐ Solutions

- ☐ Use disk imaging tools

- ☐ Use an old PC

- ☐ Cards and adapters

 - ☐ ISA SCSI card

 - ☐ A-Card IDE adapter

 - ☐ SCSI-to-IDE adapter

 - ☐ EISA FireWire card

 - ☐ FireWire-to-EIDE adapter

EXAMINING IDE/EIDE AND SATA DEVICES (CONTINUED)

- ❑ Examining the IDE host protected area
 - ❑ ATAPI-5 AT introduced in 1998 reserved and protected areas on IDE devices
 - ❑ Protected Area Run Time Interface Extension Service (PARTIES)
 - ❑ Data stored by diagnostic and restore programs
 - ❑ Tools
 - ❑ X-Ways Replica
 - ❑ HPA is also referred to as a BIOS Engineering Extension Record (BEER) data structure

EXAMINING IDE/EIDE AND SATA DEVICES (CONTINUED)

- ❑ Exploring hidden partitions
 - ❑ Suspects try to conceal evidence by hiding disk partitions
 - ❑ Norton Disk Edit can change the disk partition table
 - ❑ Leaving no indication that the deactivated partition exists in Windows Explorer
 - ❑ Use imaging tools that can access un-partitioned areas of a drive