Volumes & Partitions

Carrier: Chapter 4

Additional info

Overview

In this lecture we'll consider

- *Concept of a* **volume**
- *Concept of a* **partition**
- **Volumes** *and their relations to* **partitions**
- *Volume analysis*

Some Definitions
Volume

Volume: A group of addressable sectors on a disk that an OS or application can use for data storage.

- *Don't need to be physically contiguous sectors on a disk*
- *But need to appear to be contiguous*

A volume could be:

- *A single hard disk*
  - Physically contiguous sectors
- *Several hard disks that appear to an OS as a single volume*
  - Physically separate groups of contiguous sectors
- *A group of contiguous sectors within a hard disk* partition

Some Definitions
Partition

Partition: A group of contiguous sectors within a volume

Is a ***partition*** also a ***volume***?

Carrier writes

> *"By definition, a partition is also a volume, which is why the terms are frequently confused."*
>
> *He's right!*

But is a ***volume*** also a ***partition***?

Common Concepts
Different Terms

In a Hard Disk Drive, what is the smallest unit for read and write operations?

A sector

What about SSDs?

Carrier's book was published in 2005

SSDs were not yet on the scene

Carrier's book and these slides often refer to hard disk drives (HDDs) and sectors

The answer: ***A page***

You should be able to mentally map **sectors** ←→ **pages** or something similar in SSDs

HDD ***sectors*** are usually 512 bytes

But sometimes as large as 4096 bytes

SSD ***pages*** have been 512, 1024, 2048 or 4096 bytes

Some Definitions
Example: Volume & Partition

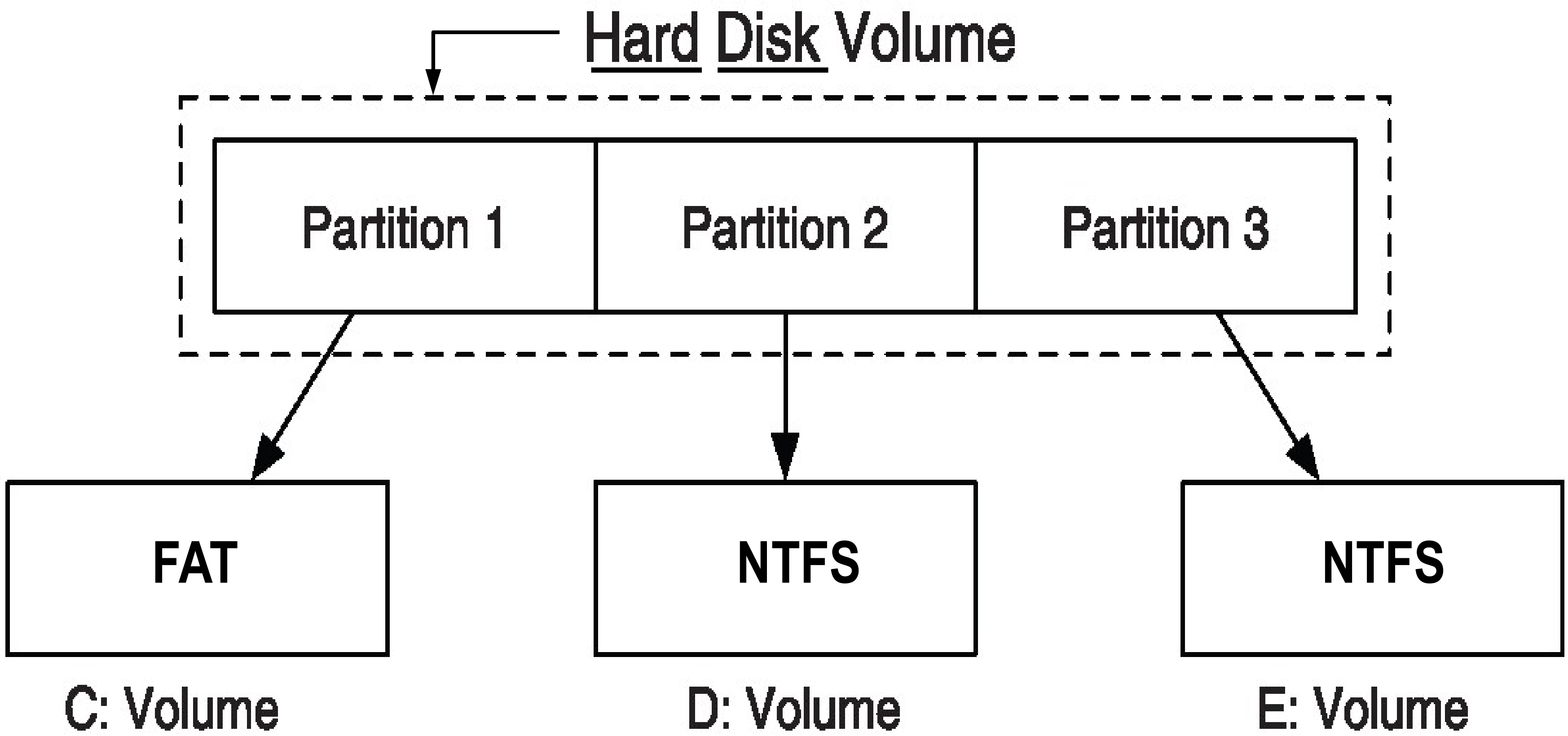
An **IA32** computer running Windows with a single hard disk has a single volume – the entire hard disk

But it can have one or more partitions within the volume

Computers are often arranged so that the drive has

- *A volume*
  - The hard disk
- *Several partitions*
  - Each consisting of contiguous sectors on the hard disk
  - Each of these partitions is also a volume

Some Definitions
Example: Volume & Partition



Why Have Partitions?

Often hard disks contain only one partition that encompasses the entire disk capacity

So why have partitions?

Analogy

- *Physical file cabinet analogy*
- *Suppose that a file cabinet has four drawers*
  - One drawer each for finance, equipment manuals, medical records and miscellaneous
- *What is analogous to a volume?*
- *What is analogous to a partition?*

Partitions
Pros and Cons Vs. One Large Partition*

Advantages

- *Improve privacy and security*
- *Allow different OSs to be installed independently*
- *Allow folders & files to be more easily organized & found*
- *Multiple users can be easily kept separate*
- *Can segregate data/software for specialized purpose (e.g., recovery)*

Disadvantages

- *Less flexible*
  - How are multiple partitions on a single volume less flexible?
- *Decreases efficiency of use of disk capacity*
  - How is efficiency decreased?
- *Reduced ability for data organization*
  - Lost opportunity for improved access times

Partitions
HDD vs. SSD

One advantage for partitioning is

- *Provides the ability to organize data*
- *Data that is accessed most frequently can be physically placed on disk for optimal read times*
  - E.g., Windows primary partition
  - Placing it on outside of platter improves read times

Another advantage

- *Helps guard against defragmentation*

Partitions

HDD vs. SSD

These advantages don't apply to SSD drives

- *Read access is the same regardless of where it is placed on SSD drive*
- *You don't need to defragment SSDs*
  - Would increase wear
  - Fragmented files can be read just as fast as defragmented files

However, you still need to segregate portions of the disk

- *System partition*
- *Primary partition*
- *Recovery partitions*

Some Definitions
Volume & Partition

The idea is

- *Start with disk hardware (one or more)*
- *Create one or more volumes on the disk(s)*
- *In each volume, create one or more partitions*
- *Each partition becomes itself a volume*

The words ***partition*** and ***volume*** are used imprecisely

- *Indeed, often either word is correct*

Different Partitioning Schemes

Hardware platforms use different partitioning schemes

- *IA32 systems*
  - A partitioning scheme with which most of us are familiar
  - OSs: Windows & Linux
- *Sun sparc servers*
  - Call their partitions "slices"
  - OS: Solaris
- *FreeBSD*
  - Also call their partitions "slices"
- *IA64 (Itanium) hardware*
  - Similar to IA32, but with less limitations

We'll discuss the different partitions later

Partition Tables

Partitioning schemes all work pretty much the same

Each volume contains a partition table

Might be called a "slice" table (sparc, FreeBSD, MacOS)

Each entry in the table describes a partition

The partition table organizes the volume layout

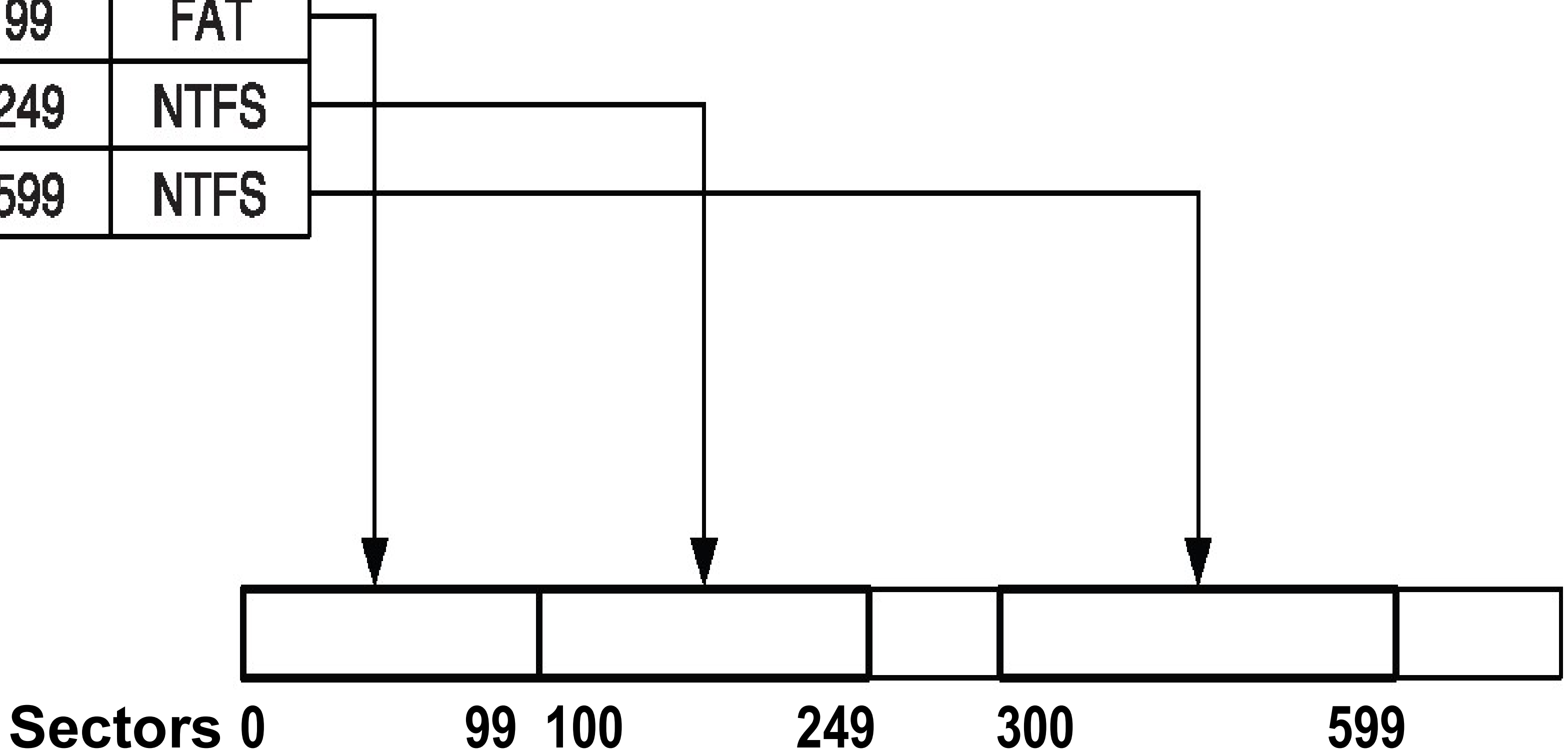
Partition tables contain two types of information

***Essential** information*

***Non-essential** information*

Partition Tables

Essential		Not Essential
Start	**End**	**Type**
0	99	FAT
100	249	NTFS
300	599	NTFS



Master Boot Record (MBR)

The MBR can be thought of as the *partition table* for the volume

- *Created when you create the first partition on the hard disk*
- *The location of the MBR is <u>always</u> LBA sector 0*
  - For hard disks: *track 0 cylinder 0, side(head 0, and starts at the 1st sector)*

MBR contains

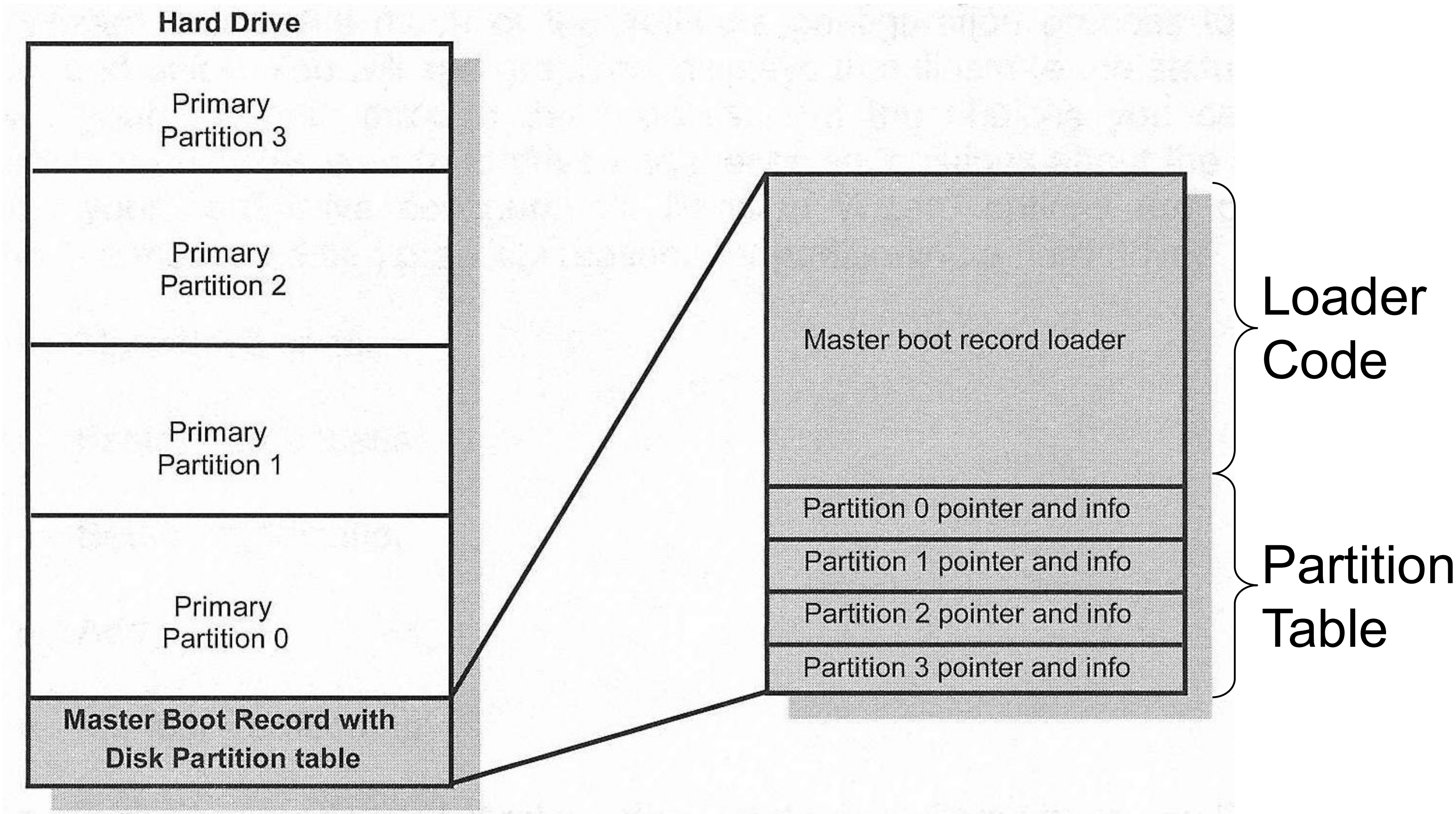
- *Partition Table for the hard disk volume*
- *Executable code (sometimes called* **MBR code** *or* **Loader Code***)*
  - Examines the Partition Table in the MBR
  - Identifies the system partition

Master Boot Record (MBR)

MBR code (What it does)

1. *Finds the **system partition's** starting sector on the drive*
2. *Loads a copy of its **Partition Boot Sector** into memory*
3. *Transfers execution to executable code in the **Partition Boot Sector***

Master Boot Record
For Drives



But We Will Be Seeing GPT

- We've now discussed basic **M**aster **B**oot **R**ecord (**MBR**) booting and **B**asic **MBR** (**BMBR**) partitions
  - *This booting & partition schema that has been very widely used since the early 1970s (half a century)*
  - *Companion software called BIOS*
    - Basic Input/Output System
  - *BMBR had limitations that had mostly been overcome*
- About the year 2000 another schema was proposed
  - *Overcame the BMBR limitations*
  - ***G**lobally **U**nique **ID**entifier **P**artition **T**able*
    - **GUID P**artition **T**able, **GPT**

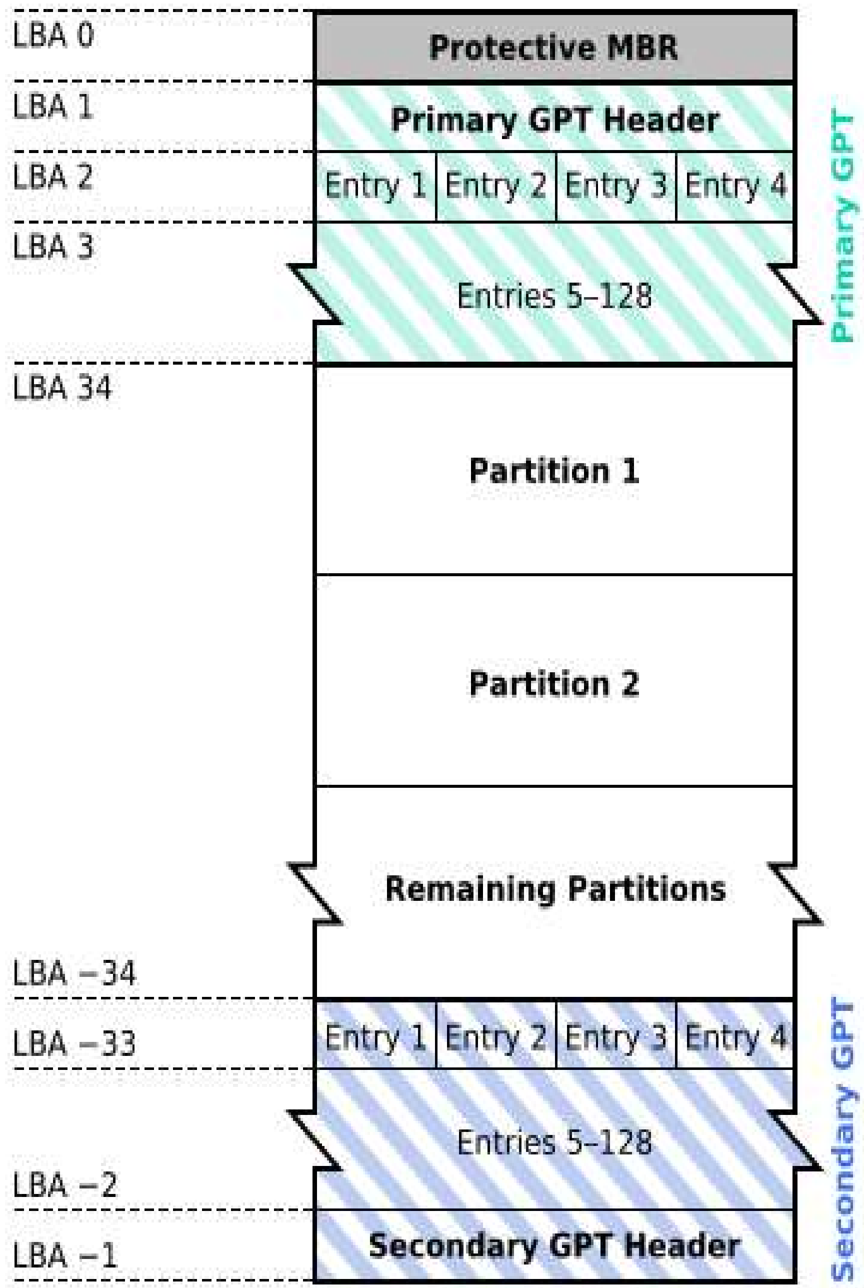
GPT Volume Layout

MBR *(in LBA 0)*
- Protective: *Prevents MBR-based software from overwriting GPT*
- Hybrid: *Allows GPT booting that starts in the MBR*

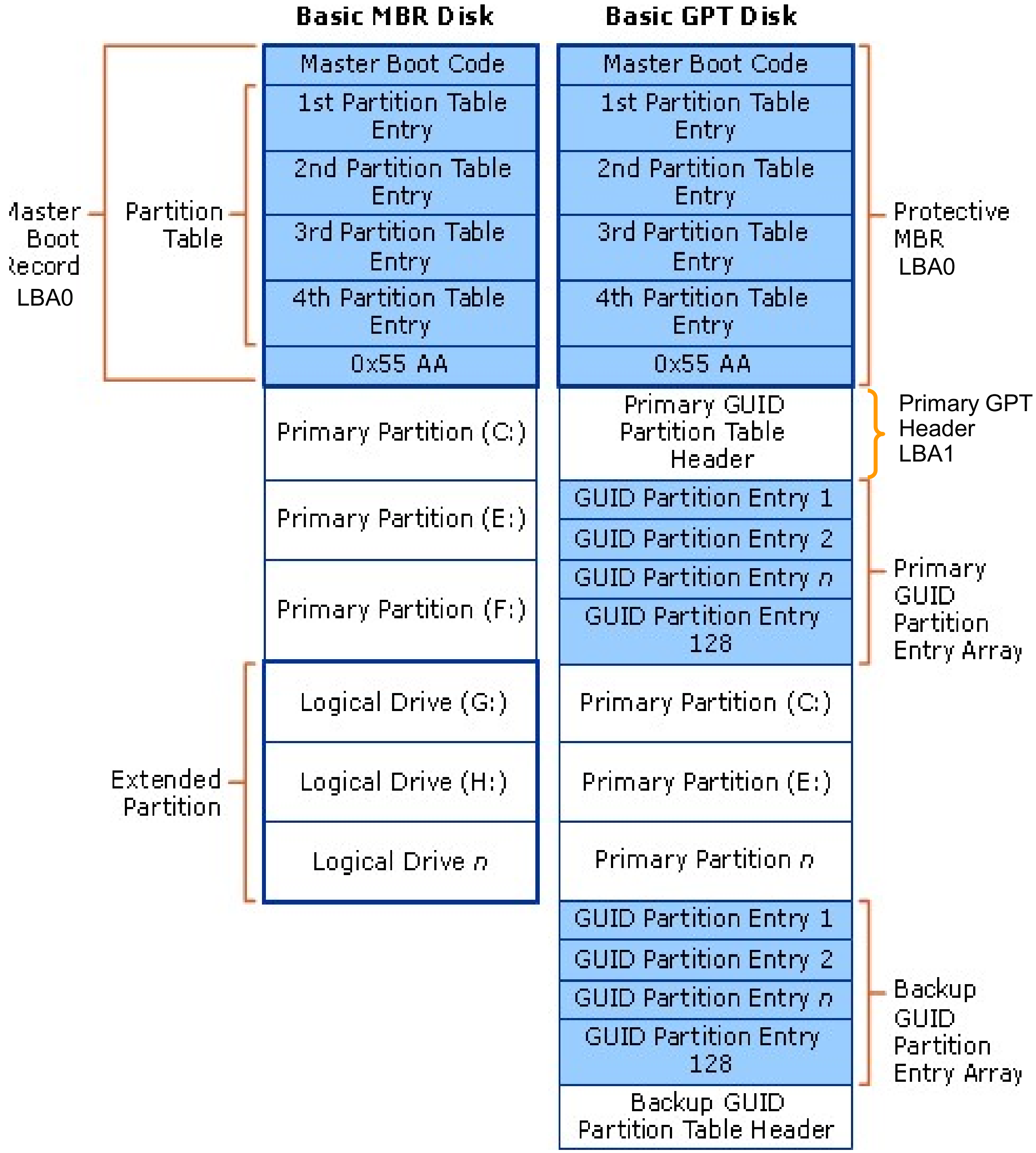
GPT Header (in LBA 1)
- *Identifies useable logical blocks*
- *Number of partitions for the drive*
- *Size of partition table* Entries
  - Reserves space for 128 entries each of 128 bytes
- *GUID (UUID) for entire volume*

Partition Table *(LBA3 - LBA33)*

Partitions *(LBA34-LBA$_{max}$ - 34)*



MBR & GPT Drive Layout Comparison



MBR Lab 6a-1

Initial Configuring of WinHex on RADISHng

Initial Configuration 1 of 2

Bring up your RADISHng Win10 VM

Open WinHex on RADISHng **<u>as Administrator</u>**

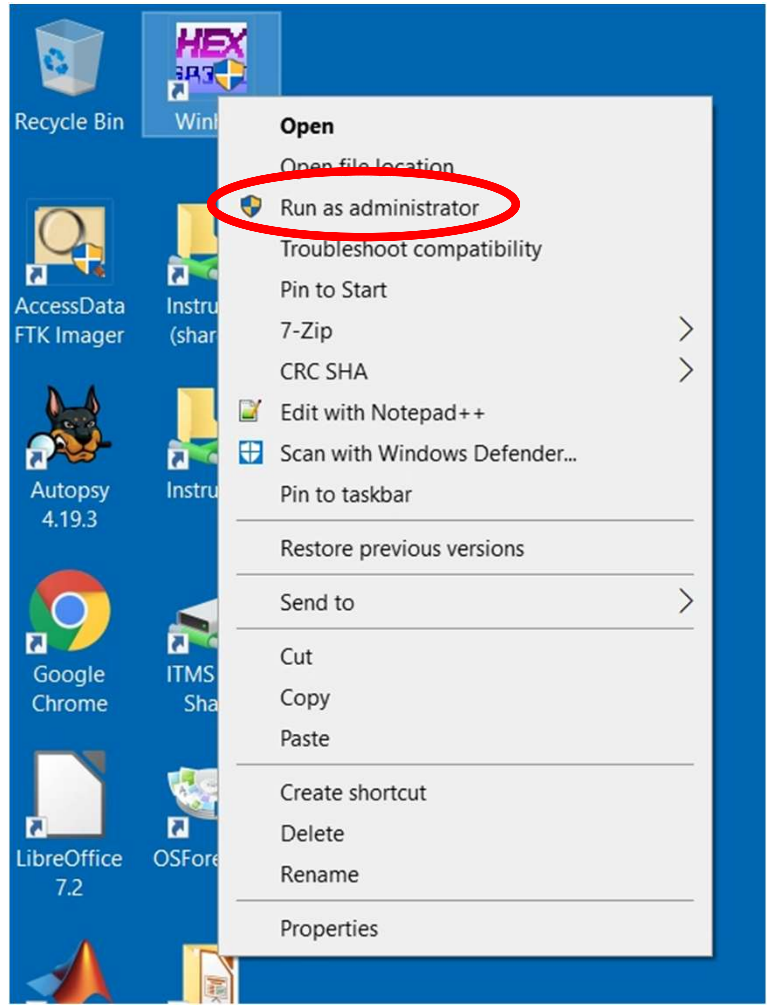
Right-click on desktop icon

In the WinHex window, if the Case Data area appears, remove it

View > Show > Case Data

Uncheck it

We do not have a WinHex Forensic License



Initial Configuration 2 of 2

Make sure that WinHex cannot change whatever file, logical or physical volume it has open

Options > Edit Mode…

Select: Read-only (write-protected)

Click: OK

MBR Lab 6a-2

Look at Physical Volume Using WinHex

Look at MBR

Viewing The MBR

We will look at the Master Boot Record using WinHex

Viewing a Physical Drive

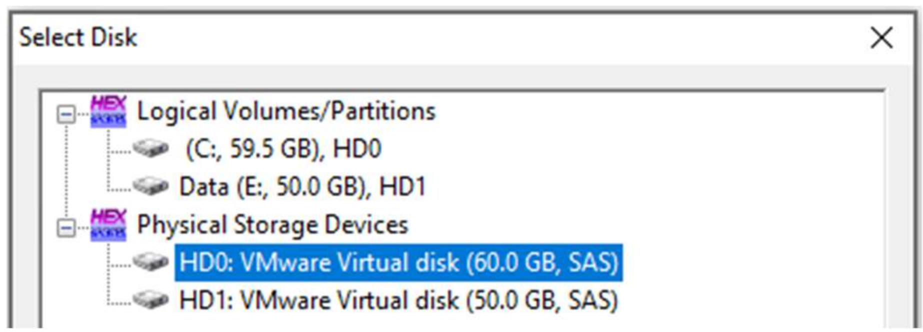
Using WinHex you can look at a computer's MBR

Menu: Tools | Open Disk..

Select: In popup window expand ***Physical Media***

Choose: HD0: VMware Virtual disk (60.0 GB SAS) or something similar

Click: OK



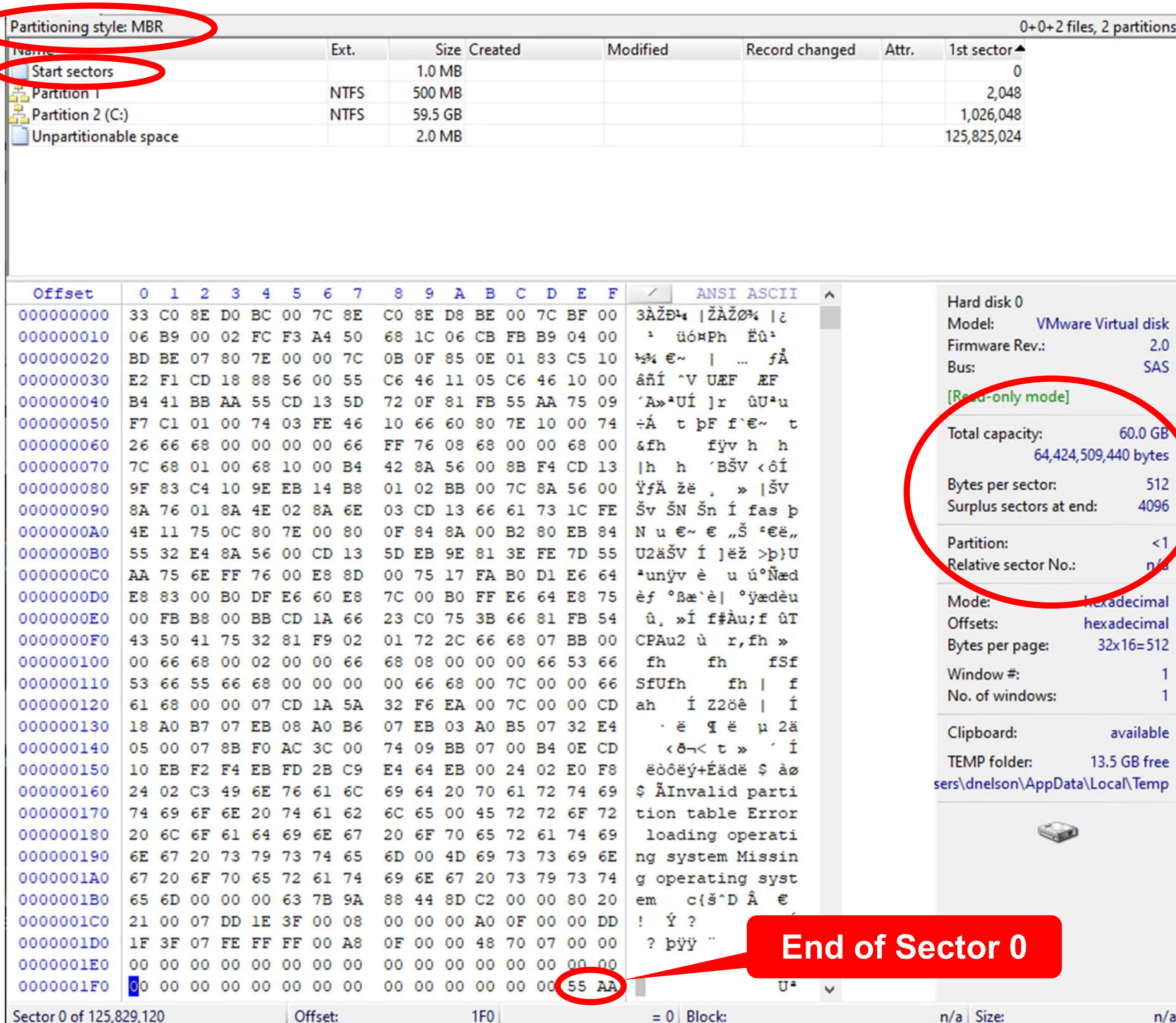
A hex display of the drive will come up

If it comes up with one or two small popup windows

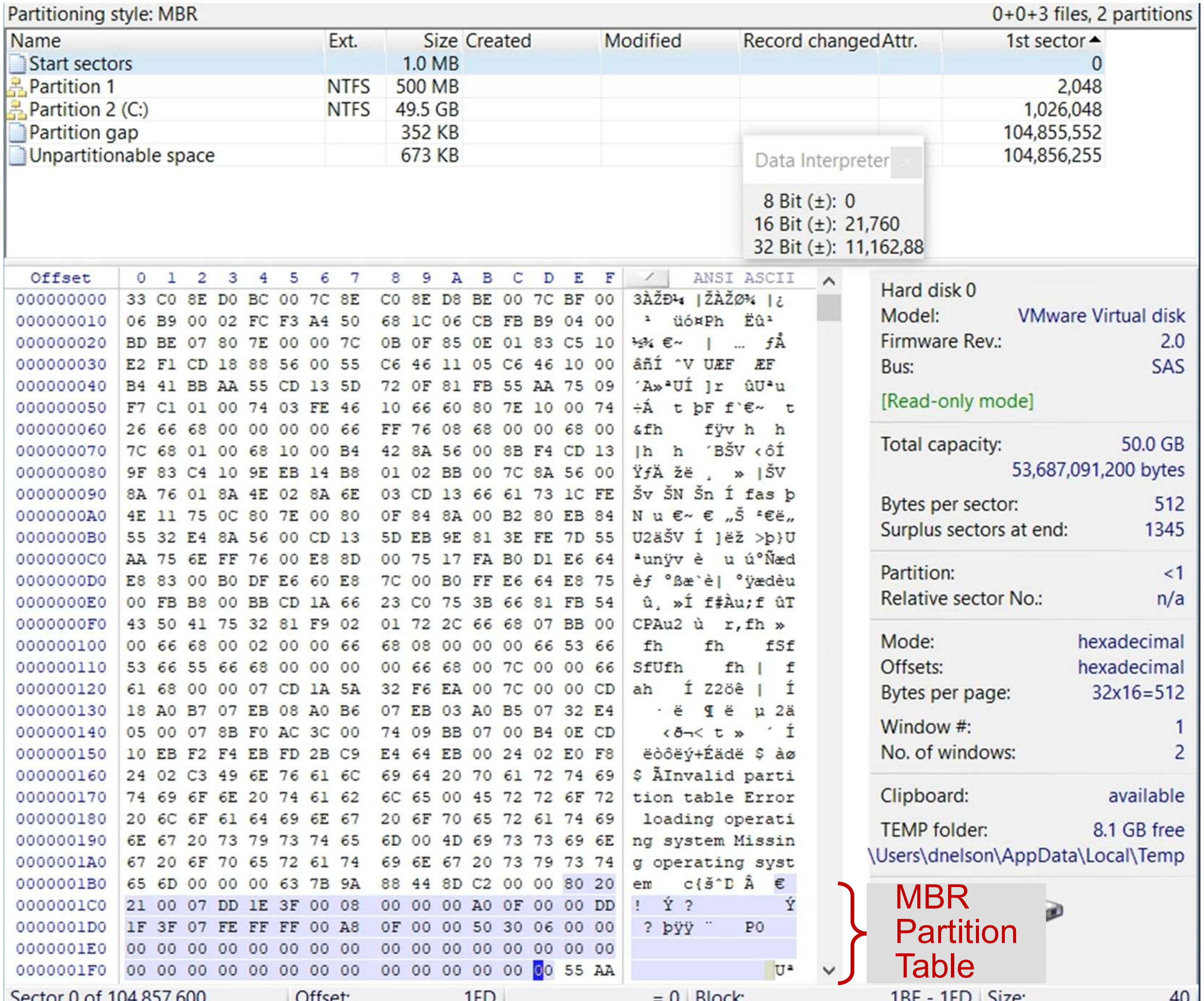
Stop! Do nothing.

I'll show you what to do on the next 2 slides

HD0: View of HD0 RADISHng Disk Volume



MBR Sector of My Rng HD0

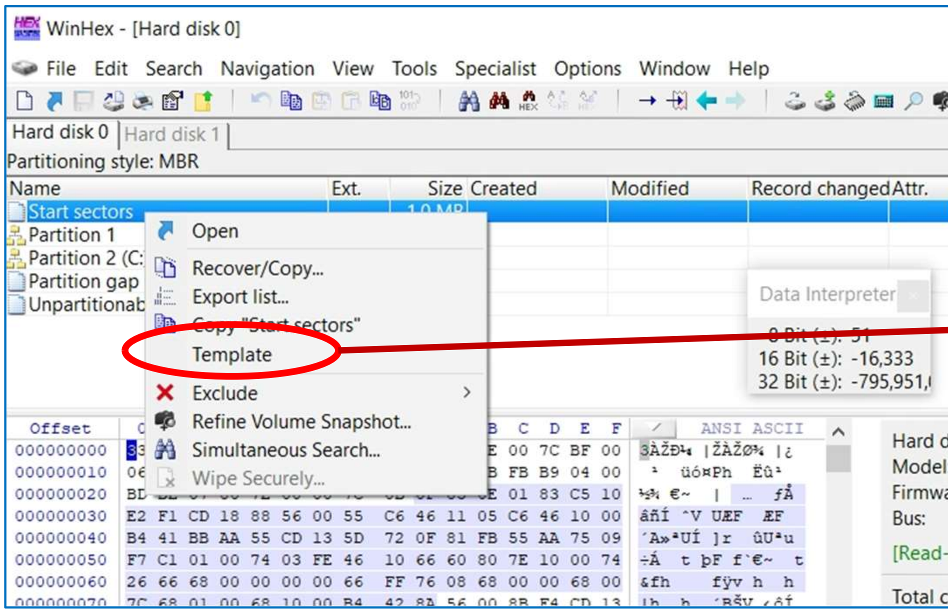


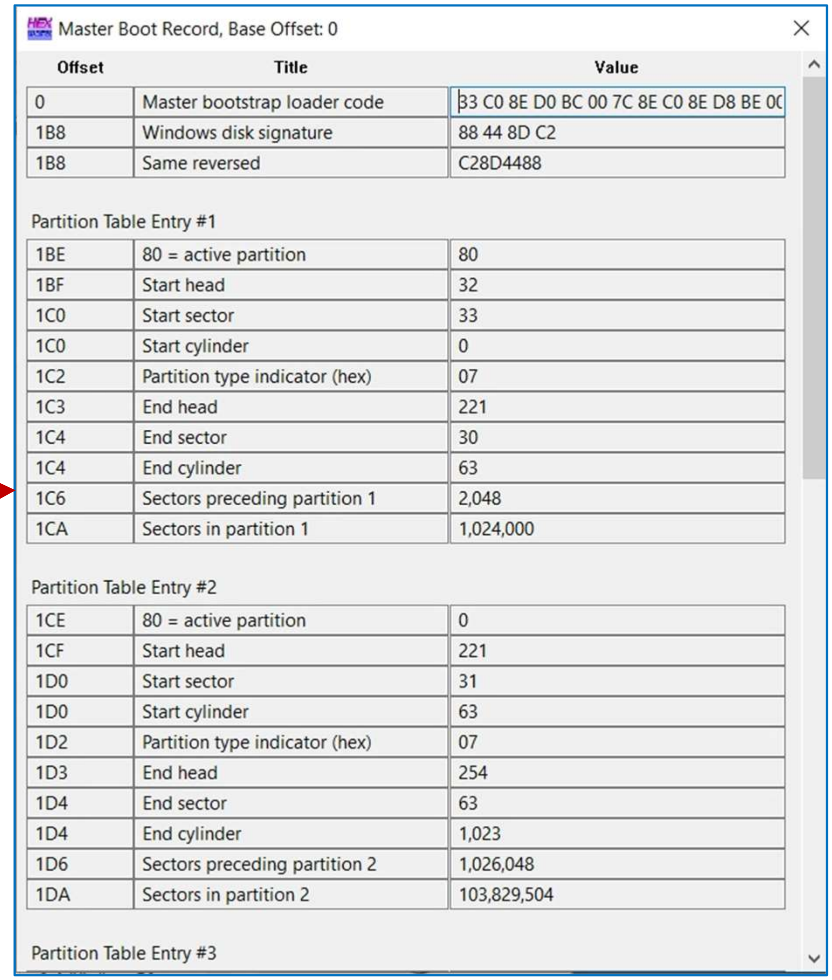
MBR Loader Code

MBR Partition Table

MBR Sector of HD0

Template Tool





GPT

The MBR can take occupy more than one sector

After sector # 0 (the 1st sector), the next 62 sectors (1-62 inclusive) are reserved on my personal computer drive

Sector #0:	MBR + some of the MBR Loader Code
Sector #1:	GPT (GUID) Partition Table
Sectors 2 - #62	Reserved *(maybe more loader code)*

Using WinHex, we would also look at the sectors after sector 0

We'll return to MBRs later

LINUX/UNIX Volumes

Volumes in UNIX-like OSs

In windows, each volume has a drive letter

C: *D:* *E:* ...

In Unix, Linux, FreeBSD…

Everything is a file

There are no separate things designated with drive letters

Every real or virtual device is represented as a file

Volumes in UNIX-like OSs

In Unix, Linux, FreeBSD…

Everything is organized as a topological tree of ***directories*** *(increasingly called* ***folders****)*

The directory tree begins at root (like all trees)

The root directory path is represented as `/`

Each directory in root is either

A subdirectory of the root volume, or

A mounting point for a new volume

Volumes in UNIX-like OSs

In windows, a CDROM is given a volume drive letter

Example: ***E:***

In UNIX, the CDROM is given a file name perhaps in the ***mnt*** or ***media*** directory

Example: ***/mnt/cdrom*** *or* ***/media/cdrom***

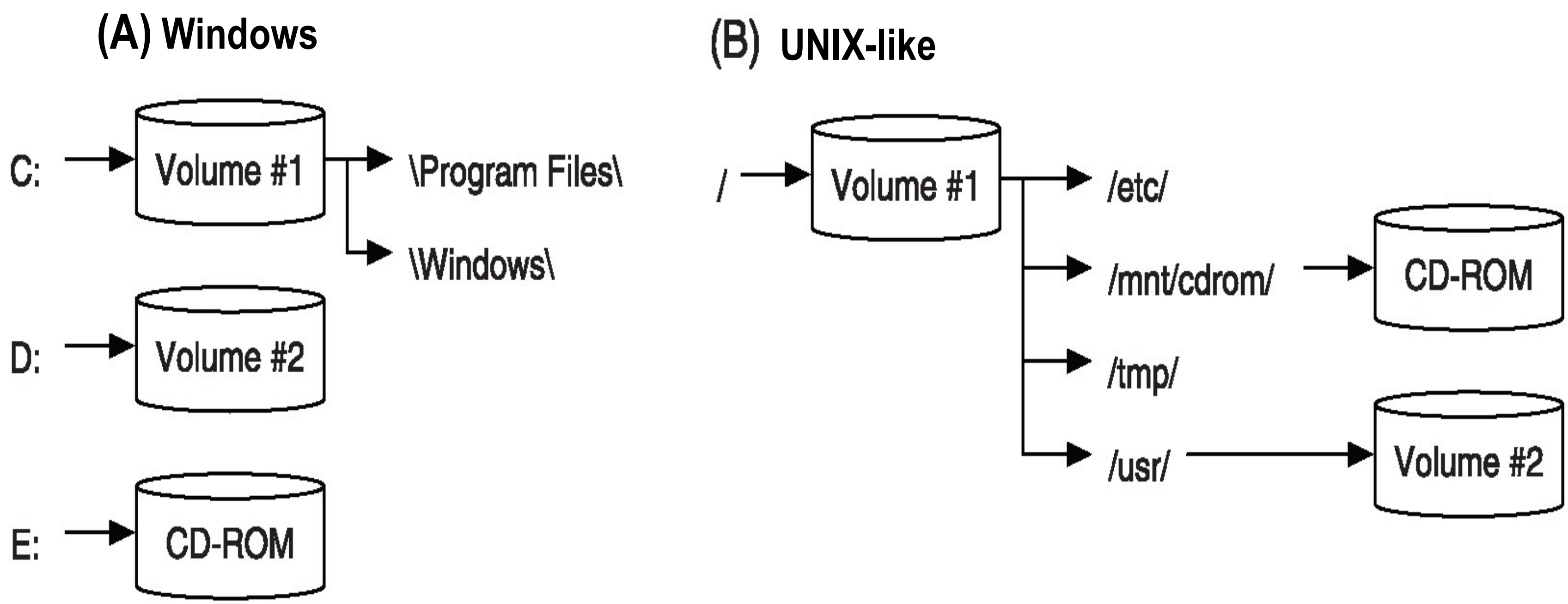
This is the mount point for the cdrom volume

Conventionally, UNIX-like OSs have different volumes for different things

/root *For common and basic configuration info*

/home *The beginning of users' home directories*

Volumes in UNIX-like OSs



Note: The figures that look like disks are not physical disks; they are **volumes**.

Volume Spanning

Overview

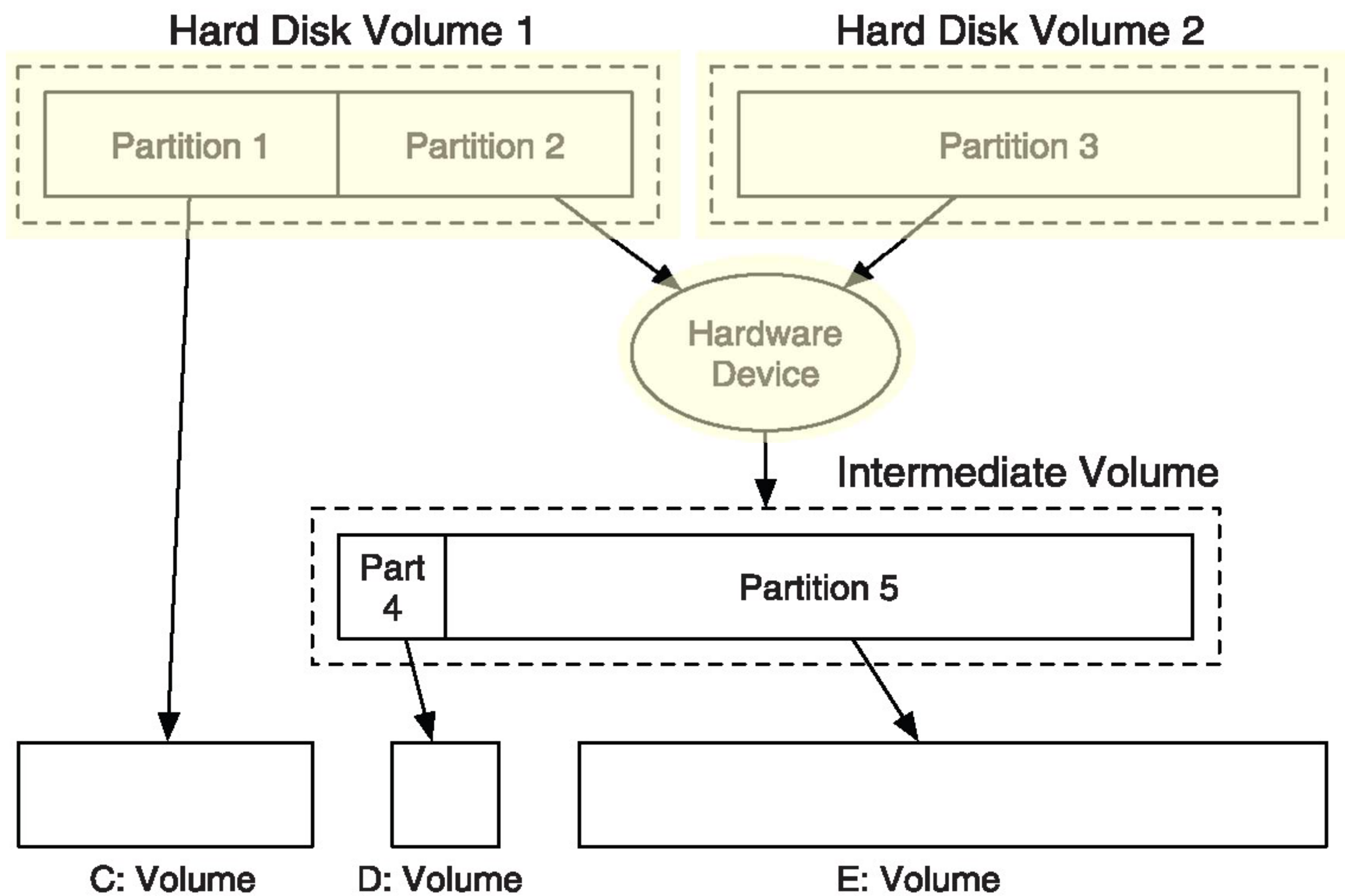
A volume can exist across two or more physical disks

- *Called **volume spanning***

Why should we want this?

- *Redundancy*
- *Economy*
  - Need for a very large volume that can be economically created using smaller physical disks
- *Speed*
  - Sometimes it's possible to increase disk throughput by writing to several physical disks concurrently

Volume Spanning Example



Sector Addresses

Three Types of Sector Addresses

Physical sector address

This is the number of the sector on a physical drive relative to the beginning of the physical disk

Logical partition sector address

The number of a sector in a partition relative to the beginning of the partition

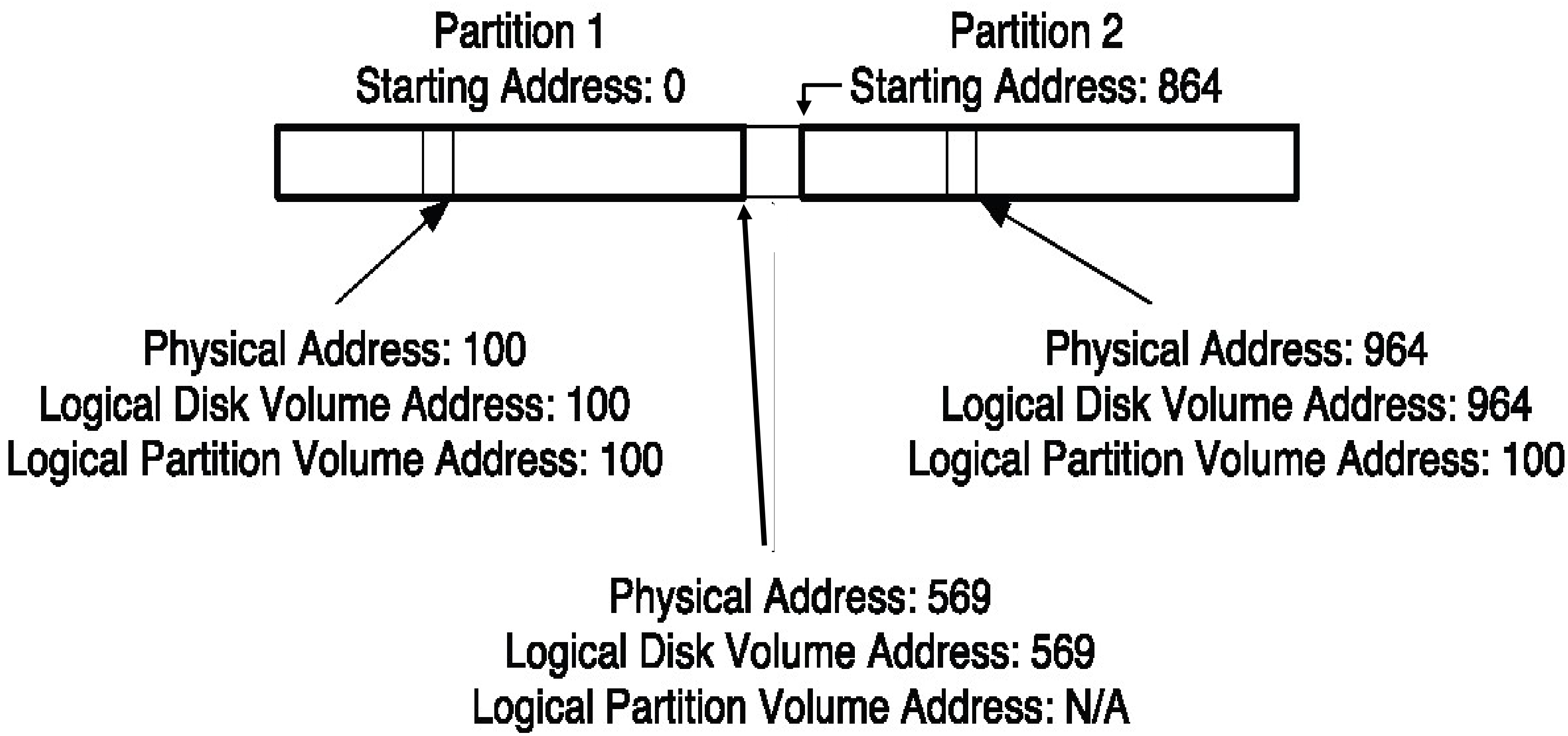
Logical volume sector address

The number of a sector in a drive volume relative to the beginning of the volume

The disk volume may be part of a physical drive or be parts of several physical drives

Sector Address Example

Single physical drive with single volume and two partitions



Volume Analysis

Volume Analysis

- Volume analysis investigates the layout and organization of a disk volume
  - *A volume's beginning and ending sectors*
  - *Each partition's beginning and ending sectors*
    - May need to investigate partitions
    - Usually, each partition contains a file system
  - *Unallocated sectors in a volume*
- Basically, to do this analysis, we need to find the partition tables
  - *Identify the essential and non-essential data*
  - *Use the essential data to determine the volume organization*

Volume Analysis

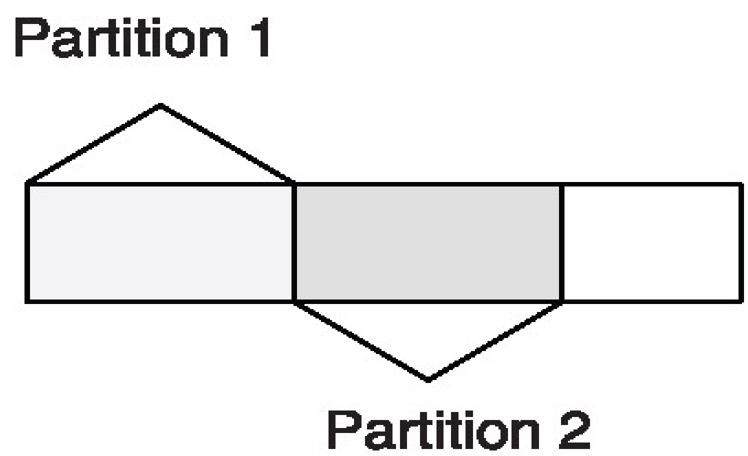
The *essential data* in the partition table defines the arrangement of the volume

We can see where there are sectors not allocated to a partition

We can see if there are inconsistencies such as overlapping partitions

Volume Analysis

Example A: Unallocated Space at End



Final partition ends before end of its parent volume

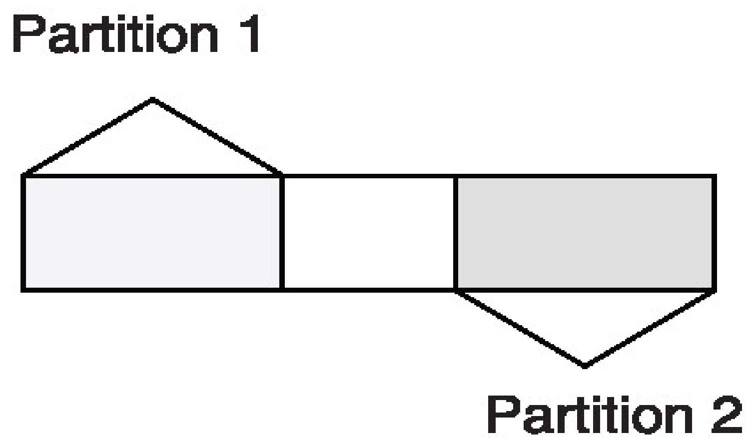
Could hide information in the unallocated sectors

Look for end of volume and end of last sector & compare

Valid configuration

Volume Analysis

Example B: Unallocated Space Between Partitions



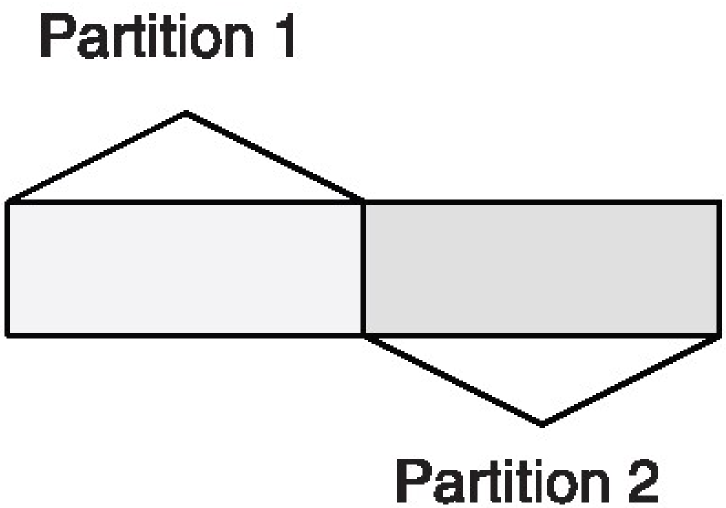
Unallocated sectors exist between the partitions

Could hide information in the unallocated sectors

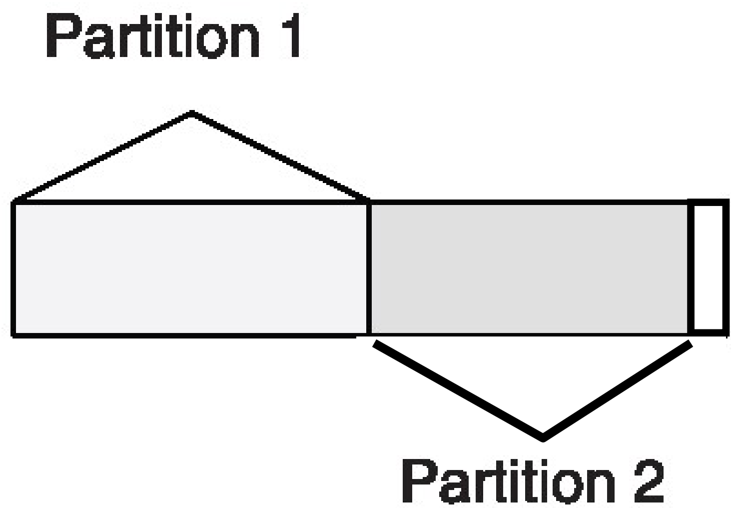
Valid configuration

Volume Analysis

Example C: Normal Partitioning



The partitions completely fill up the volume

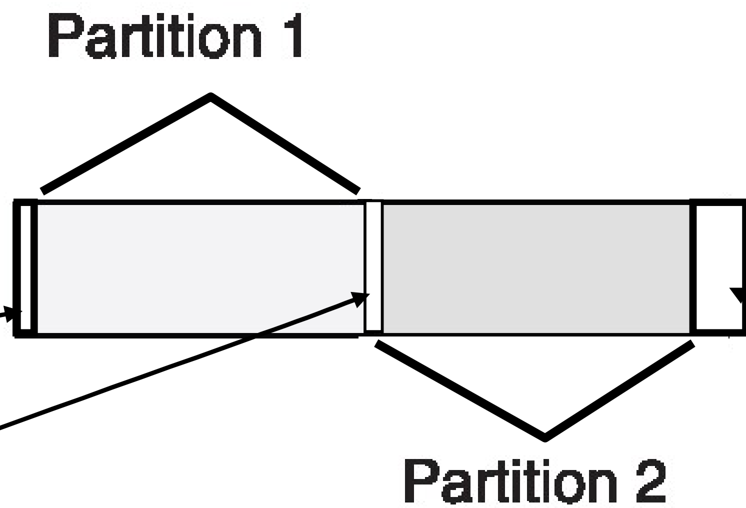


The partitions almost fill up the volume

"Almost" may mean that a 100 MB is unallocated

Some partitioning software cannot end a partition at the end of a volume or disk

Volume Analysis
Example C: Realistic Normal Partitioning



The 1st 63 sectors (outer disk track) is reserved for the partition table and boot code *(63 x 512 = 32256 bytes)*

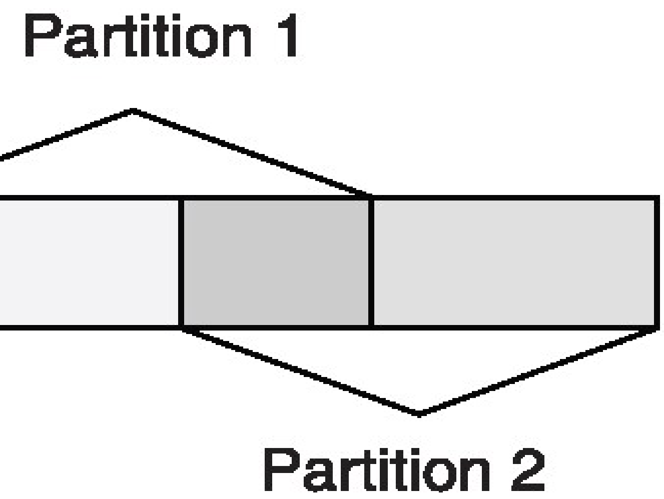
The partitions fill up most of the disk

There may be gaps between the partitions

There are a few Mbytes left at the end of the disk

Volume Analysis

Example D: Overlapping Partitions



Partition 2's beginning sector is smaller than Partition 1's ending sector

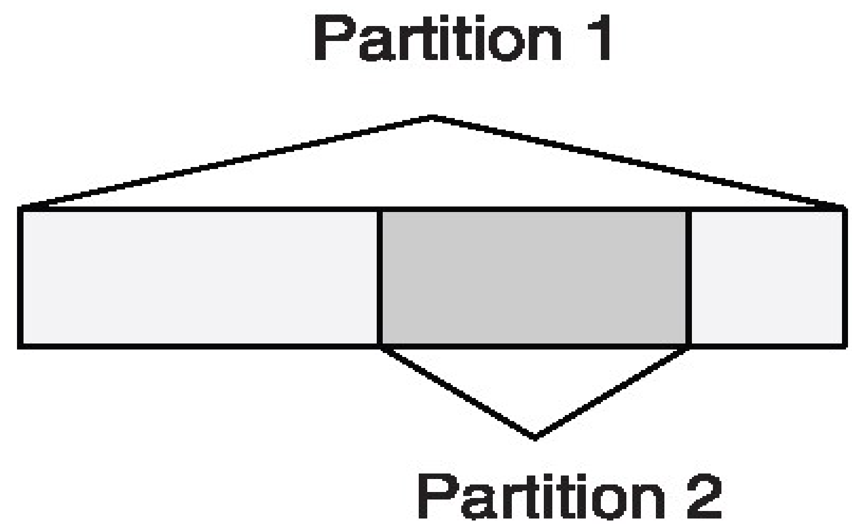
Not a valid configuration

May indicate an error in the partition table

Must analyze the partitions themselves

Volume Analysis

Example E: Nested Partitions



Partition 2 is totally contained in Partition 1

Not a valid configuration

May indicate an error in the partition table

Must analyze the partitions themselves

Separating & Analyzing the Parts of a Volume

Overview

Suppose that we have a volume (often the disk itself)

A way to analyze the volume is to:

- *Identify the parts*
- *Separate them*
- *Then analyze each piece*

Some commercial tools do this semi automatically

We'll do it manually here

- *So that we understand what's going on*
- *We can fall back on this if other tools don't work*

Process

Steps involved in identifying, separating and analyzing the parts of a volume

1. ***Identify*** *the volume disk sector addresses*
   - The starting of each part
   - The end of each part
2. ***Extract*** *each part into a separate file*
3. ***Analyze*** *each part separately*

1. Identify

A tool in The SleuthKit (TSK) looks at the partition table and reports the details

Essential

Non essential

The tool that we will use is ***mmls***

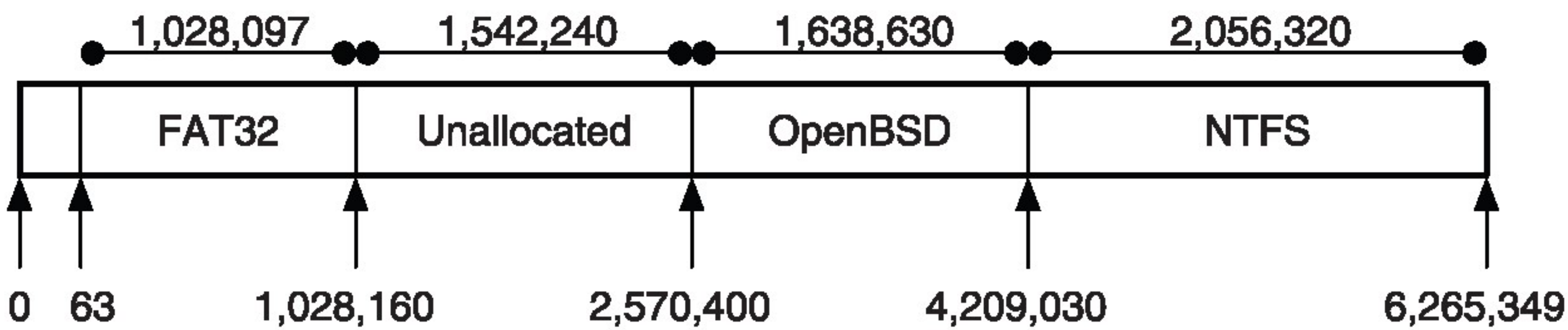
Similar to `fdisk -lu` *in Linux distros*

But has information in a better format

Let's look at an example from the book

Example Disk Volume*

Here is a disk that is in itself a volume




```
# mmls -t dos disk1.dd
Units are in 512-byte sectors
     Slot  Start       End         Length      Description
00:  ----- 0000000000  0000000000  0000000001  Table #0
01:  ----- 0000000001  0000000062  0000000062  Unallocated
02:  00:00 0000000063  0001028159  0001028097  Win95 FAT32 (0x0B)
03:  ----- 0001028160  0002570399  0001542240  Unallocated
04:  00:03 0002570400  0004209029  0001638630  OpenBSD (0xA6)
05:  00:01 0004209030  0006265349  0002056320  NTFS (0x07)
```

`mmls` Interpretation

```
# mmls -t dos disk1.dd
Units are in 512-byte sectors
     Slot  Start       End         Length      Description
00:  ----- 0000000000  0000000000  0000000001  Table #0
01:  ----- 0000000001  0000000062  0000000062  Unallocated
02:  00:00 0000000063  0001028159  0001028097  Win95 FAT32 (0x0B)
03:  ----- 0001028160  0002570399  0001542240  Unallocated
04:  00:03 0002570400  0004209029  0001638630  OpenBSD (0xA6)
05:  00:01 0004209030  0006265349  0002056320  NTFS (0x07)
```

Location of structure where information came from on disk

`00:01` means 1st structure, 2nd entry

`02:03` means 3rd structure, 4th entry

`-----` indicates either a MBR or partition table or unallocated sectors

Index of mmls entries

Essential & Non-Essential

```
# mmls –t dos disk1.dd
Units are in 512-byte sectors
     Slot  Start       End         Length      Description
00:  ----- 0000000000  0000000000  0000000001  Table #0
01:  ----- 0000000001  0000000062  0000000062  Unallocated
02:  00:00 0000000063  0001028159  0001028097  Win95 FAT32 (0x0B)
03:  ----- 0001028160  0002570399  0001542240  Unallocated
04:  00:03 0002570400  0004209029  0001638630  OpenBSD (0xA6)
05:  00:01 0004209030  0006265349  0002056320  NTFS (0x07)
```

What is the essential data?

The starting and ending location for each partition

Why?

The purpose of a partition system is to organize the layout of the volume and this is the only information required for this task

What is non-essential

All others, including type and description

Why?

They serve to provide additional information about the partitions but are not necessary to determine the partition layout.

2. Extract

Now that we know what the parts are we separate them

To do this we can use good old `dd`

`dd` allows us to specify what we need with these parameters

- ***if***: *The disk image to read from*
- ***of***: *The output file to save to*
- ***bs***: *The size of the block to read each time, 512 bytes is the default*
- ***skip***: *The number of blocks to skip before reading, each of size **bs***
- ***count***: *Number of blocks to copy from input to output, each of size **bs***

2. Extract

```
# mmls -t dos disk1.dd
Units are in 512-byte sectors
      Slot   Start        End          Length       Description
00:   -----  0000000000   0000000000   0000000001   Table #0
01:   -----  0000000001   0000000062   0000000062   Unallocated
02:   00:00  0000000063   0001028159   0001028097   Win95 FAT32 (0x0B)
03:   -----  0001028160   0002570399   0001542240   Unallocated
04:   00:03  0002570400   0004209029   0001638630   OpenBSD (0xA6)
05:   00:01  0004209030   0006265349   0002056320   NTFS (0x07)
```

 # dd if=disk1.dd of=part1.dd bs=512 skip=63 count=1028097

 # dd if=disk1.dd of=part2.dd bs=512 skip=2570400 count=1638630

 # dd if=disk1.dd of=part3.dd bs=512 skip=4209030 count=2056320

 # dd if=disk1.dd of=unalloc0.dd bs=512 skip=1 count=62

 # dd if=disk1.dd of=unalloc1.dd bs=512 skip=1028160 count=1542240

3. Analyze

Once we have the parts of the volume separated, we can analyze each one separately

- *Partitions can be analyzed using tools designed for analyzing partitions*
- *We'll discuss these later*
- *Unallocated parts can be analyzed using tools such as hex viewers*

Deleted or Corrupted Partitions

Deleted or Corrupted Partitions*

What if there **was** a partition in unallocated space?

It has been deleted, or

The disk has been repartitioned, or

The partition table is corrupted

What if the partition cannot be analyzed because

The partition is messed up

The above are often used to purposely hide information

But they can also occur for benign reasons

Think about how you might create a hidden partition in unallocated space

Tools exist that try to deal with these situations

Deleted or Corrupted Partitions

Partition recovery tools work by assuming that there was a file system

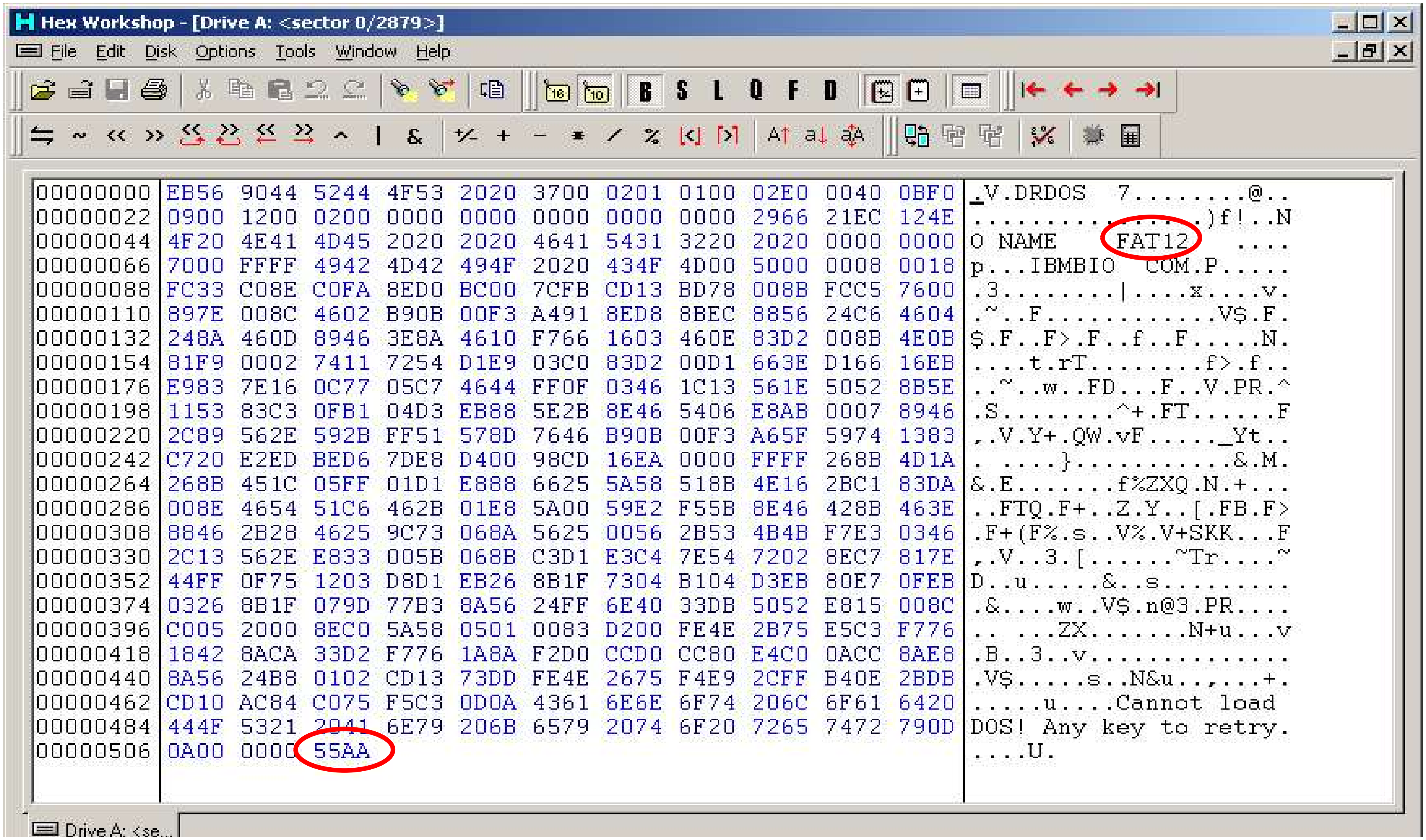
Many file systems start with a data structure that has a known signature

FAT and most NTFS partitions have the value 0x55AA in bytes 510 and 511 of the 1st sector of the partition

$510_{10} = 1FE_{16} = 0x1FE$

What about removable storage?

1st Sector of a Floppy



1st Sector of a 2 GByte Flash Drive

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
0000000000	EB	3C	90	4D	53	57	49	4E	34	2E	31	00	02	40	08	00	ë<▮MSWIN4.1..@..
0000000016	02	00	02	00	00	F8	F0	00	20	00	80	00	20	00	00	00	.....øð. .▮. ...
0000000032	E0	FF	3B	00	00	00	29	10	6E	3A	24	4E	4F	20	4E	41	àÿ;...).n:$NO NA
0000000048	4D	45	20	20	20	20	46	41	54	31	36	20	20	20	FA	33	ME FAT16 ú3
0000000064	C9	8E	D1	BC	FC	7B	16	07	BD	78	00	C5	76	00	1E	56	É▮Ñ¼ü{..½x.Åv..V
0000000080	16	55	BF	22	05	89	7E	00	89	4E	02	B1	0B	FC	F3	A4	.U¿".▮~.▮N.±.üó¤
0000000096	06	1F	BD	00	7C	C6	45	FE	0F	8B	46	18	88	45	F9	38	..½.|ÆEþ.▮F.▮Eù8
0000000112	4E	24	7D	22	8B	C1	99	E8	77	01	72	1A	83	EB	3A	66	N$}"▮Á▮èw.r.▮ë:f
0000000128	A1	1C	7C	66	3B	07	8A	57	FC	75	06	80	CA	02	88	56	¡.|f;.▮Wüu.▮Ê.▮V
0000000144	02	80	C3	10	73	ED	33	C9	8A	46	10	98	F7	66	16	03	.▮Ã.sí3É▮F.▮÷f..
0000000160	46	1C	13	56	1E	03	46	0E	13	D1	8B	76	11	60	89	46	F..V..F..Ñ▮v.`▮F
0000000176	FC	89	56	FE	B8	20	00	F7	E6	8B	5E	0B	03	C3	48	F7	ü▮Vþ¸ .÷æ▮^..ÃH÷
0000000192	F3	01	46	FC	11	4E	FE	61	BF	00	07	E8	23	01	72	39	ó.Fü.Nþa¿..è#.r9
0000000208	38	2D	74	17	60	B1	0B	BE	D8	7D	F3	A6	61	74	39	4E	8-t.`±.¾Ø}ó¦at9N
0000000224	74	09	83	C7	20	3B	FB	72	E7	EB	DD	BE	7F	7D	AC	98	t.▮Ç ;ûrçëÝ¾▮}¬▮
0000000240	03	F0	AC	84	C0	74	17	3C	FF	74	09	B4	0E	BB	07	00	.ð¬▮Àt.<ÿt.´.»..
0000000256	CD	10	EB	EE	BE	82	7D	EB	E5	BE	80	7D	EB	E0	98	CD	Í.ëî¾▮}ëå¾▮}ëà▮Í
0000000272	16	5E	1F	66	8F	04	CD	19	BE	81	7D	8B	7D	1A	8D	45	.^.f▮.Í.¾▮}▮}.▮E
0000000288	FE	8A	4E	0D	F7	E1	03	46	FC	13	56	FE	B1	04	E8	C1	þ▮N.÷á.Fü.Vþ±.èÁ
0000000304	00	72	D6	EA	00	02	70	00	B4	42	EB	2D	60	66	6A	00	.rÖê..p.´Bë-`fj.
0000000320	52	50	06	53	6A	01	6A	10	8B	F4	74	EC	91	92	33	D2	RP.Sj.j.▮ôtì''3Ò
0000000336	F7	76	18	91	F7	76	18	42	87	CA	F7	76	1A	8A	F2	8A	÷v.'÷v.B▮Ê÷v.▮ò▮
0000000352	E8	C0	CC	02	0A	CC	B8	01	02	8A	56	24	CD	13	8D	64	èÀÌ..Ì¸..▮V$Í.▮d
0000000368	10	61	72	0A	40	75	01	42	03	5E	0B	49	75	77	C3	03	.ar.@u.B.^.IuwÃ.
0000000384	18	01	27	0D	0A	49	6E	76	61	6C	69	64	20	73	79	73	..'..Invalid sys
0000000400	74	65	6D	20	64	69	73	6B	FF	0D	0A	44	69	73	6B	20	tem diskÿ..Disk
0000000416	49	2F	4F	20	65	72	72	6F	72	FF	0D	0A	52	65	70	6C	I/O errorÿ..Repl
0000000432	61	63	65	20	74	68	65	20	64	69	73	6B	2C	20	61	6E	ace the disk, an
0000000448	64	20	74	68	65	6E	20	70	72	65	73	73	20	61	6E	79	d then press any
0000000464	20	6B	65	79	0D	0A	00	00	49	4F	20	20	20	20	20	20	key....IO
0000000480	53	59	53	4D	53	44	4F	53	20	20	20	53	59	53	7F	01	SYSMSDOS SYS▮.
0000000496	00	41	BB	00	07	80	7E	02	0E	E9	40	FF	00	00	55	AA	.A»..▮~..é@ÿ..Uª
0000000512	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....

Sector 0 of 3932128 | Offset: 0 | = 235 | Block: n/a | Size:

1st Sector of Another 2 GByte Flash Drive

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	33	C0	8E	D0	BC	00	7C	FB	50	07	50	1F	FC	BE	1B	7C	3À▌Ð¼.|ûP.P.ü¾.|
00000010	BF	1B	06	50	57	B9	E5	01	F3	A4	CB	BD	BE	07	B1	04	¿..PW¹å.ó¤Ë½¾.±.
00000020	38	6E	00	7C	09	75	13	83	C5	10	E2	F4	CD	18	8B	F5	8n.|.u.▌Å.âôÍ.▌õ
00000030	83	C6	10	49	74	19	38	2C	74	F6	A0	B5	07	B4	07	8B	▌Æ.It.8,tö µ.´.▌
00000040	F0	AC	3C	00	74	FC	BB	07	00	B4	0E	CD	10	EB	F2	88	ð¬<.tü»..´.Í.ëò▌
00000050	4E	10	E8	46	00	73	2A	FE	46	10	80	7E	04	0B	74	0B	N.èF.s*þF.▌~..t.
00000060	80	7E	04	0C	74	05	A0	B6	07	75	D2	80	46	02	06	83	▌~..t. ¶.uÒ▌F..▌
00000070	46	08	06	83	56	0A	00	E8	21	00	73	05	A0	B6	07	EB	F..▌V..è!.s. ¶.ë
00000080	BC	81	3E	FE	7D	55	AA	74	0B	80	7E	10	00	74	C8	A0	¼▌>þ}Uªt.▌~..tÈ
00000090	B7	07	EB	A9	8B	FC	1E	57	8B	F5	CB	BF	05	00	8A	56	·.ë©▌ü.W▌õË¿..▌V
000000A0	00	B4	08	CD	13	72	23	8A	C1	24	3F	98	8A	DE	8A	FC	.´.Í.r#▌Á$?▌▌Þ▌ü
000000B0	43	F7	E3	8B	D1	86	D6	B1	06	D2	EE	42	F7	E2	39	56	C÷ã▌Ñ▌Ö±.ÒîB÷â9V
000000C0	0A	77	23	72	05	39	46	08	73	1C	B8	01	02	BB	00	7C	.w#r.9F.s.¸..».|
000000D0	8B	4E	02	8B	56	00	CD	13	73	51	4F	74	4E	32	E4	8A	▌N.▌V.Í.sQOtN2ä▌
000000E0	56	00	CD	13	EB	E4	8A	56	00	60	BB	AA	55	B4	41	CD	V.Í.ëä▌V.`»ªU´AÍ
000000F0	13	72	36	81	FB	55	AA	75	30	F6	C1	01	74	2B	61	60	.r6▌ûUªu0öÁ.t+a`
00000100	6A	00	6A	00	FF	76	0A	FF	76	08	6A	00	68	00	7C	6A	j.j.ÿv.ÿv.j.h.|j
00000110	01	6A	10	B4	42	8B	F4	CD	13	61	61	73	0E	4F	74	0B	.j.´B▌ôÍ.aas.Ot.
00000120	32	E4	8A	56	00	CD	13	EB	D6	61	F9	C3	49	6E	76	61	2ä▌V.Í.ëÖaùÃInva
00000130	6C	69	64	20	70	61	72	74	69	74	69	6F	6E	20	74	61	lid partition ta
00000140	62	6C	65	00	45	72	72	6F	72	20	6C	6F	61	64	69	6E	ble.Error loadin
00000150	67	20	6F	70	65	72	61	74	69	6E	67	20	73	79	73	74	g operating syst
00000160	65	6D	00	4D	69	73	73	69	6E	67	20	6F	70	65	72	61	em.Missing opera
00000170	74	69	6E	67	20	73	79	73	74	65	6D	00	00	00	00	00	ting system.....
00000180	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000190	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
000001A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
000001B0	00	00	00	00	00	2C	44	63	18	2E	07	C3	00	00	80	01	.....,Dc...Ã..▌.
000001C0	01	00	06	80	E0	C0	20	00	00	00	E0	FF	3B	00	00	00	...▌àÀ ...àÿ;...
000001D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
000001E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
000001F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	55	AA	..............Uª

Further Analysis Possibilities

Note that you can't always assume that the text "FAT" will occur

- *The 2GB flash drive didn't have it*

Signatures such as 0x55AA are used to determine where a partition may have started

- *Work backward from byte 511 (0x1FF) relative to the start of the partition*

Try At Home

Search a flash drive and locate the 0x55AA signature

Using WinHex search your ***RADISHng*** hard disk ***C*** partition to locate a 0x55AA signature.

- *Is it at the end of a sector?*
- *Then it is probably the end of the MBR or a partition table sector*

Try the above on your personal computer when you have time.

- *But it might be a GPT partition*

Further Confirmation

If the signature suggests a specific file system, then you can look further for some confirmation

For instance, since FAT allocates sectors in clusters of 2^x, where x is an integer

Tools can search for these numbers

Suppose that there are many numbers in what should be the location of the File Allocation Table and its duplicate that are not integer powers of two.

What might you suspect?

The FAT file system signature may be bogus

Ways Tools Work

Searching mechanism of tools vary

Some tools examine each sector and compare it to known signatures

- *These are sometimes proprietary*

Some search cylinder boundaries

- *Partitions are often created on cylinder boundaries on HDDs*

gpart
A Free Partition Recovery Tool

- ***gpart*** : Free Linux & FreeBSD tool that can be used for partition recovery
  - *Tests sectors to determine likely file system*
  - ***http://www.stud.uni-hannover.de/user/76201/gpart/***
    - This web site is restricted
  - *In Kali Linux, if you type* **'gpart'**, *you will be told how to install it*
    - Command: **sudo apt install gpart**

gpart Output from Disk Image File *disk2.dd*

```
# gpart -v disk2.dd
* Warning: strange partition table magic 0x0000.
[REMOVED]
Begin scan...
Possible partition(DOS FAT), size(800mb), offset(0mb)
   type: 006(0x06)(Primary 'big' DOS (> 32MB))
   size: 800mb #s(1638566) s(63-1638628)
   chs:  (0/1/1)-(101/254/62)d (0/1/1)-(101/254/62)r
   hex:  00 01 01 00 06 FE 3E 65 3F 00 00 00 A6 00 19 00

Possible partition(DOS FAT), size(917mb), offset(800mb)
   type: 006(0x06)(Primary 'big' DOS (> 32MB))
   size: 917mb #s(1879604) s(1638630-3518233)
   chs:  (102/0/1)-(218/254/62)d (102/0/1)-(218/254/62)r
   hex:  00 00 01 66 06 FE 3E DA E6 00 19 00 34 AE 1C 00

Possible partition(Linux ext2), size(502mb), offset(1874mb)
   type: 131(0x83)(Linux ext2 filesystem)
   size: 502mb #s(1028160) s(3839535-4867694)
   chs:  (239/0/1)-(302/254/63)d (239/0/1)-(302/254/63)r
   hex:  00 00 01 EF 83 FE 7F 2E 2F 96 3A 00 40 B0 0F 00
```

gparted
Not the Same as gpart

gparted is used to analyze, add, remove and resize partitions

- *It is similar to the Disk Management utility in Windows*
- *It has a nice GUI*

Let's now run it on our Kali Linux VM

`testdisk`
Another Free Partition Recovery Tool

`testdisk` : Free tool that runs on several platforms

- *Recovers partitions and make disks bootable again*
- *Does sector analysis and recreates "correct" partition table*
- *Runs on MSDOS, Win32, Linux, FreeBSD & MacOS*
- *Works only for "basic wiping" or bad partition tables*
- *In Kali,* **`testdisk`** *is already installed*

WinHex
Scan for Lost Partitions

In WinHex, there is a menu option:

Tools > Disk Tools > Scan for Lost Partitions

This command searches for the signature of master boot records, partition table sectors, FAT and NTFS boot sectors via the 0x55 0xAA signature

It also supports Linux file systems (Ext2/Ext3/Ext4)

Works with sector size 512 bytes only

Lab 06a-4
Slide 1 of 1

Use ***hdparm*** & ***mmls*** to analyze your hard drive in your Kali Linux VM

Commands:

```
sudo hdparm -I /dev/sda
sudo mmls /dev/sda
```

What do the tool outputs tell you?

Note: ***hdparm*** is a Linux-only SleuthKit tool that uses ATA commands to query an ATA disk to determine if it has a Host Protected Area (HPA). It bypasses the OS and BIOS and gives ATA commands directly to the disk.