From Zero to Hero

Digital Forensics 1.0.1

Michael Hamm - CIRCL

TLP:CLEAR

75th TF-CSIRT Meeting 2025

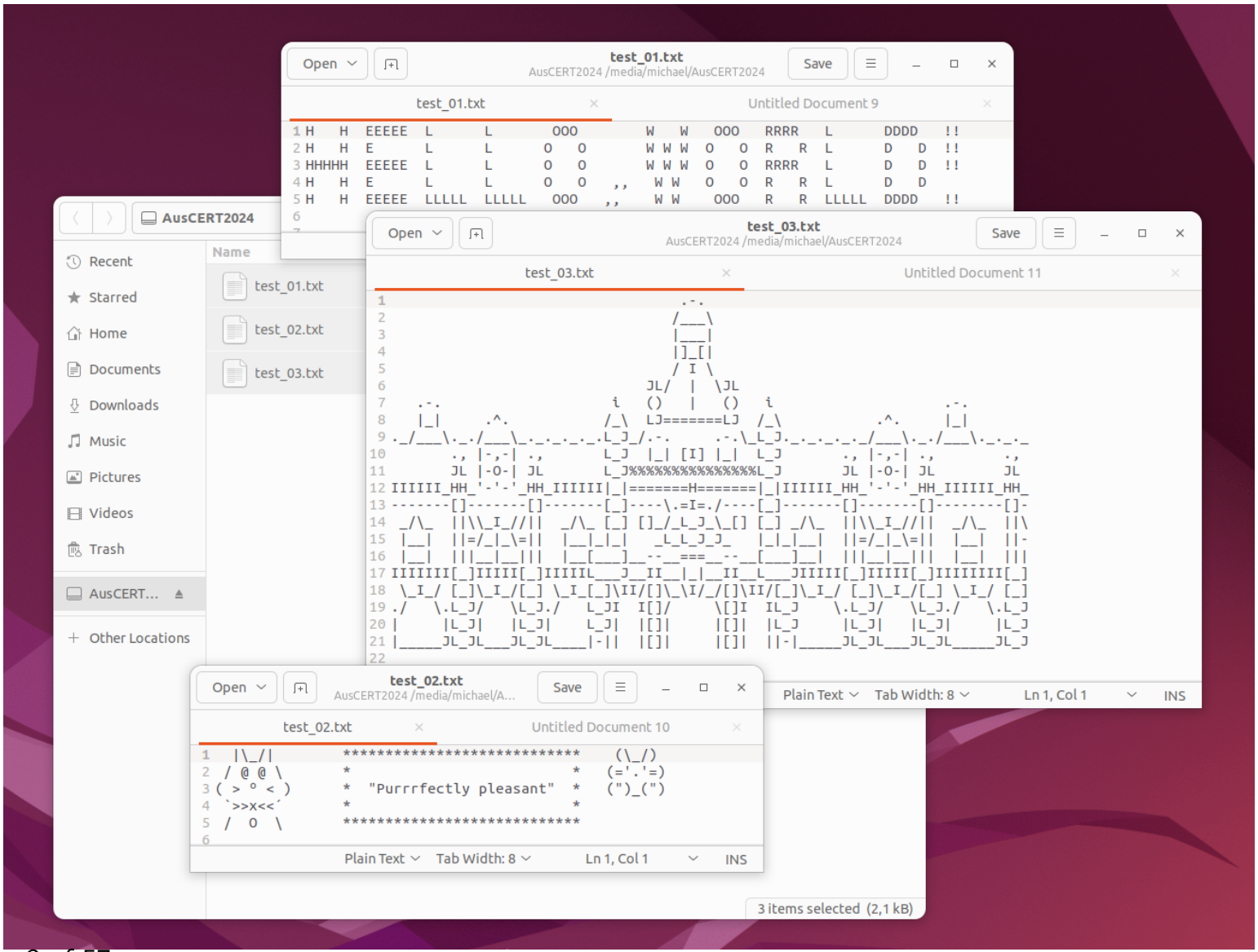
1.1 Motivation and Goals

- Motivation: The tool told so
  - Fancy tools delivering nice results
  - Do we understand the results
  - How the tools come to the results
  - What are the limits of the tools

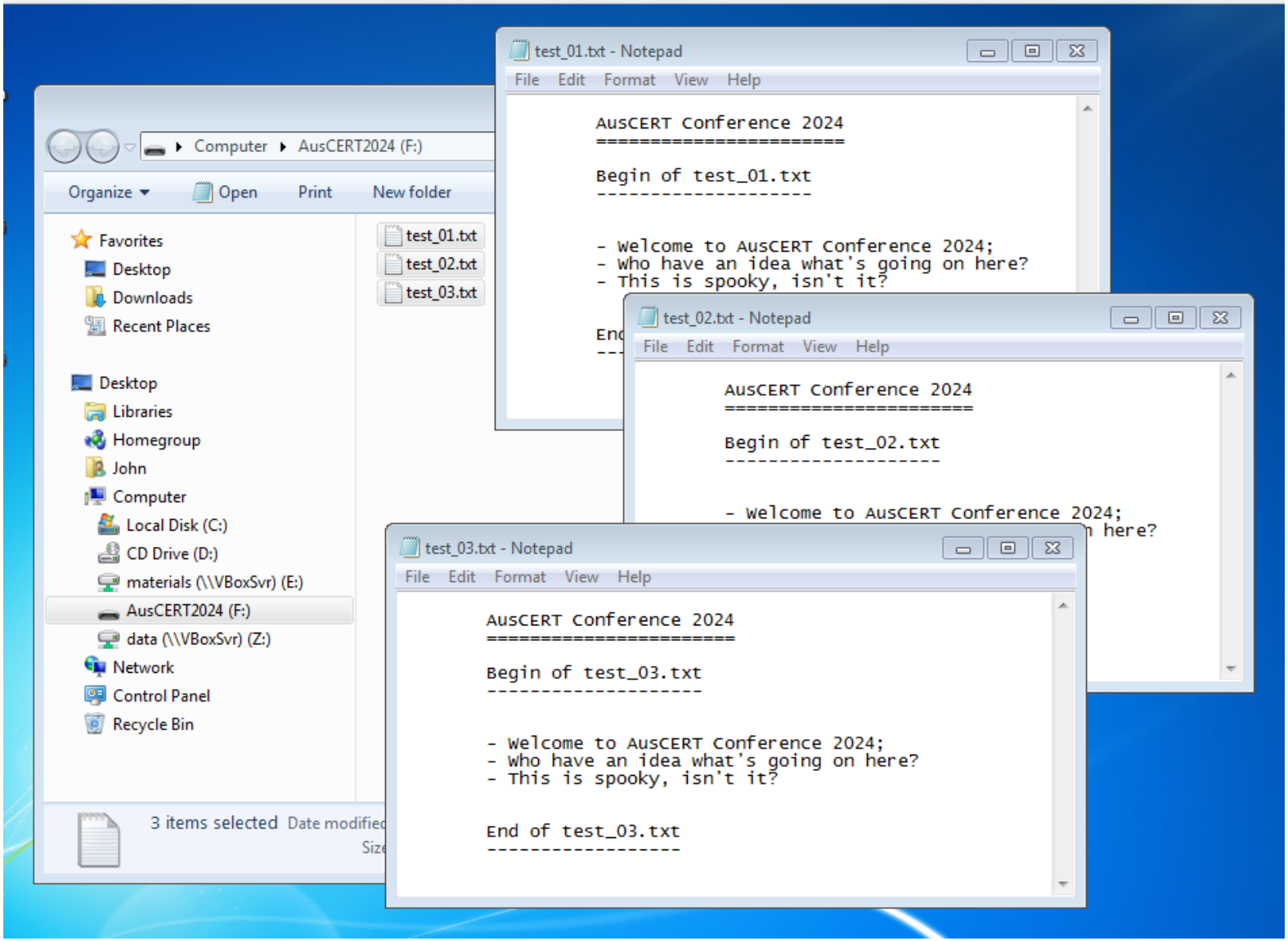
- Goals: The data told so
  - Read the data low level - Bits & Bytes
  - Understand how data is stored in structures
  - Be able to undrestand the data manually
  - Working with the low level tools

$\rightarrow$ Demo: Different tool producing different output

1.2 Demo


test_01.txt
AusCERT2024 /media/michael/AusCERT2024
Open
Save
test_01.txt
Untitled Document 9
H H EEEEE L L OOO W W OOO RRRR L DDDD !!
H H E L L O O W W W O O R R L D D !!
HHHHH EEEEE L L O O W W W O O RRRR L D D !!
H H E L L O O ,, W W O O R R L D D
H H EEEEE LLLLL LLLLL OOO ,, W W OOO R R LLLLL DDDD !!
test_03.txt
AusCERT2024 /media/michael/AusCERT2024
Open
Save
test_03.txt
Untitled Document 11
test_02.txt
AusCERT2024 /media/michael/A...
Open
Save
test_02.txt
Untitled Document 10
|_/|
/ @ @ \
(> º <)
`>>x<<´
/ O \
"Purrrfectly pleasant"
(_/)
(='.'=)
(")_(")
Plain Text
Tab Width: 8
Ln 1, Col 1
INS
AusCERT2024
Recent
Starred
Home
Documents
Downloads
Music
Pictures
Videos
Trash
AusCERT...
Other Locations
Name
test_01.txt
test_02.txt
test_03.txt
3 items selected (2,1 kB)

1.2 Demo


test_01.txt - Notepad
File Edit Format View Help
AusCERT Conference 2024
Begin of test_01.txt
- Welcome to AusCERT Conference 2024;
- Who have an idea what's going on here?
- This is spooky, isn't it?
test_02.txt - Notepad
File Edit Format View Help
AusCERT Conference 2024
Begin of test_02.txt
- Welcome to AusCERT Conference 2024;
test_03.txt - Notepad
File Edit Format View Help
AusCERT Conference 2024
Begin of test_03.txt
- Welcome to AusCERT Conference 2024;
- Who have an idea what's going on here?
- This is spooky, isn't it?
End of test_03.txt
Computer ▸ AusCERT2024 (F:)
Organize Open Print New folder
Favorites
Desktop
Downloads
Recent Places
Libraries
Homegroup
John
Computer
Local Disk (C:)
CD Drive (D:)
materials (\\VBoxSvr) (E:)
AusCERT2024 (F:)
data (\\VBoxSvr) (Z:)
Network
Control Panel
Recycle Bin
test_01.txt
test_02.txt
test_03.txt
3 items selected

1.2 Demo

```
$ sudo dmesg -T -W
     usb 3-9: new high-speed USB device number 8 using xhci_hcd
     usb 3-9: Product: USB Flash Disk
     usb-storage 3-9:1.0: USB Mass Storage device detected
     sd 0:0:0:0: [sda] 15722496 512-byte logical blocks: (8.05 GB/7.50 GiB)
     sda: sda1


$ sudo fdisk -l /dev/sda
     Device     Boot  Start    End Sectors  Size Id Type
     /dev/sda1        144000 262143  118144 57,7M  7 HPFS/NTFS/exFAT


$ sudo mmls /dev/sda



$ mount | grep sda
     /dev/sda1 on /media/michael/AusCERT2024 type ntfs3 (rw,nosuid, .....


$ sudo fsstat /dev/sda1
     File System Type: NTFS
     Volume Serial Number: 7B702E352BD29E15
     OEM Name: NTFS
     Volume Name: AusCERT2024
     Version: Windows XP
```

2.1 Data in a binary system

- BIT $\rightarrow$ Binary digit
- Data stored in binary form

  `x Bits --> 01010000011010010110111001100111 --> y Bits`

  Bit x + 2 = 1

  Bit x + 3 = 0

  $\rightarrow$ What information is stored within this data?

- *"..... information is data arranged in a meaningful way for some perceived purpose ....."* $\rightarrow$ Interpretative rules

- Grouping, addressing and interpreting

```
-->   01010000   01101001   01101110   01100111   -->
      --------   --------   --------   --------
-->   Byte 117   Byte 118   Byte 119   Byte 120   -->
```

2.1 Data in a binary system

- Grouping examples:
  - Nibble: `0101 0000 0110 1001 0110 1110 0110 0111`
  - Byte: `01010000 01101001 01101110 01100111`
  - Word: `0101000001101001 0110111001100111`
  - Double Word: `01010000011010010110111001100111`

- Interpreting:
  - Integer: (Signed, Unsigned)
  - Endian: (Big, Little)
  - Floating Point
  - Binary Coded Decimal, Packed BCD
  - Encoding: (ASCII, ISO8859, Unicode 16L, 16B, 32L, 32B)
  - Binary: (ELF, MZ, PE, GIF, JPEG, ZIP, PDF, OLE, ...)
  - ...

2.2 Number Systems

- Decimal:

```
2145
||||_   5 * 10^0 =      5
|||__   4 * 10^1 =     40
||___   1 * 10^2 =    100
|____   2 * 10^3 =  2,000
                   -------
                   2,145
```

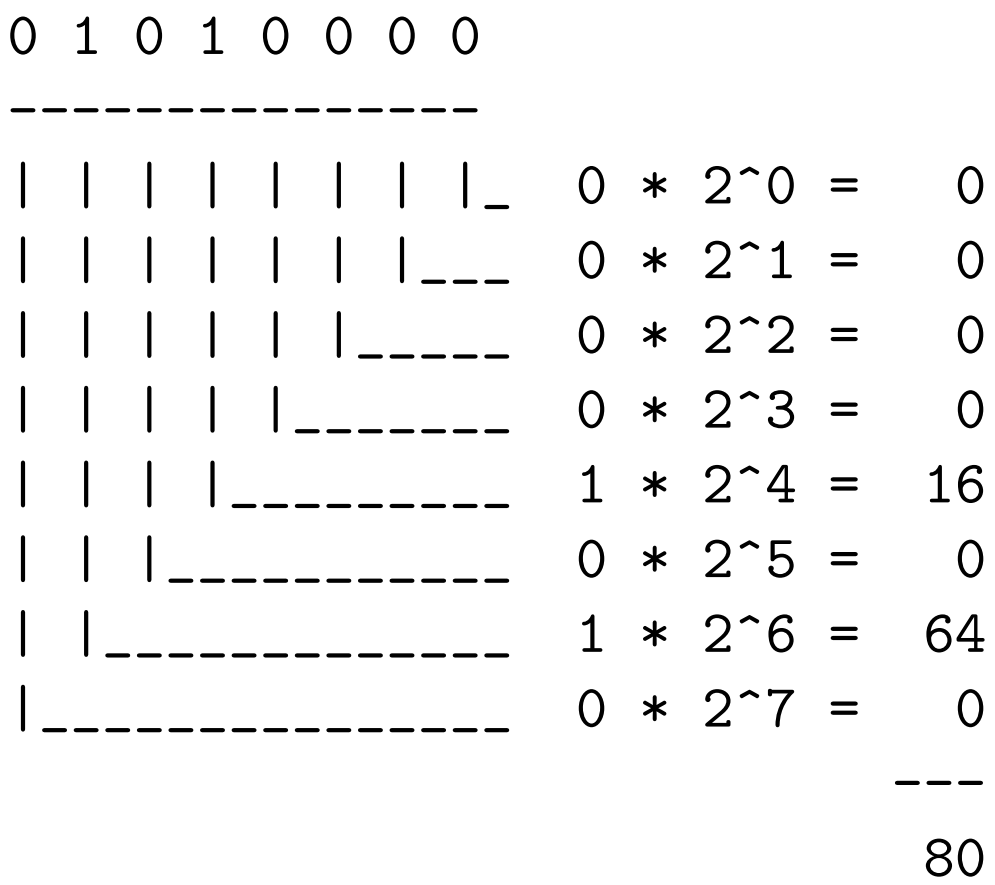
- Binary:

```
1111
||||_   1 *  2^0 =      1
|||__   1 *  2^1 =      2
||___   1 *  2^2 =      4
|____   1 *  2^3 =      8
                   -------
                        15 = 1111
```

- Hexadecimal:

```
2A9F
||||_  15 * 16^0 =     15
|||__  09 * 16^1 =    144
||___  10 * 16^2 =  2,560
|____  02 * 16^3 =  8,192
                   -------
                   10,911 = 0x2A9F
```

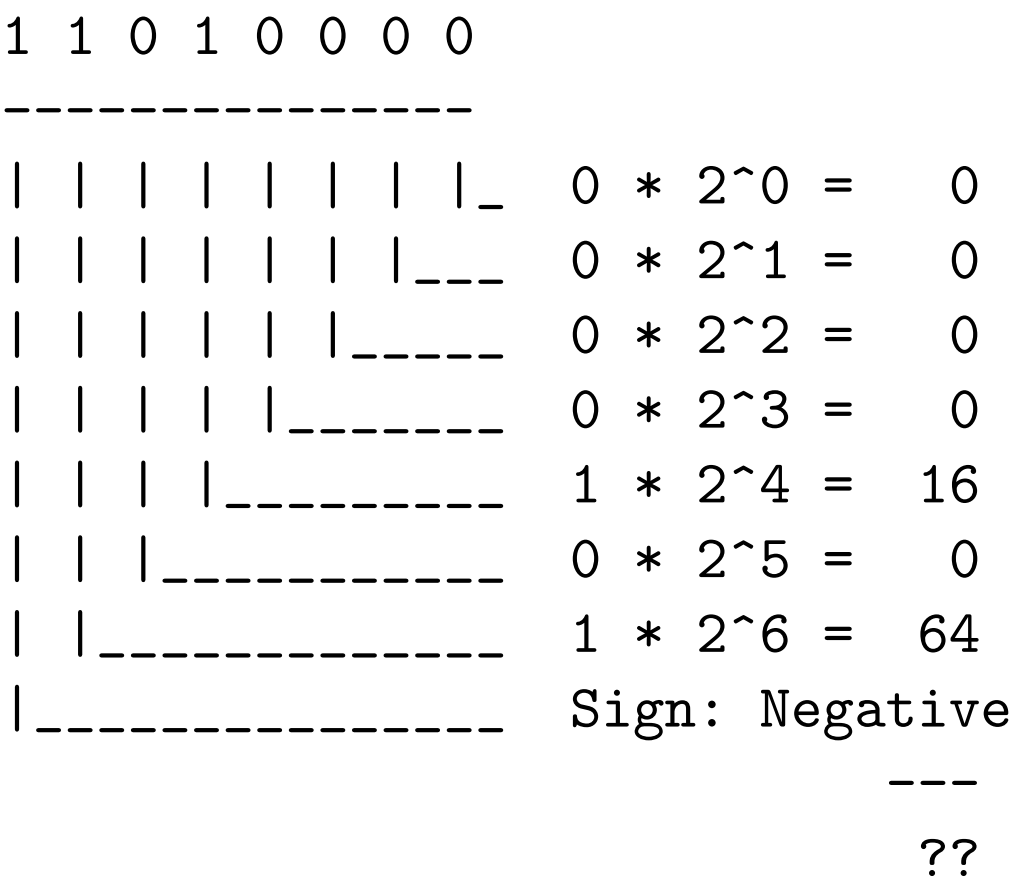
2.3 Interpreting binary data: Integer



2.3 Interpreting binary data: Signed Integer

```
0 1 0 1 0 0 0 0
---------------
| | | | | | | |_  0 * 2^0 =   0
| | | | | | |___  0 * 2^1 =   0
| | | | | |_____  0 * 2^2 =   0
| | | | |_______  0 * 2^3 =   0
| | | |_________  1 * 2^4 =  16
| | |___________  0 * 2^5 =   0
| |_____________  1 * 2^6 =  64
|_______________  Sign: Positive
                             ---
                              80
```

2.3 Interpreting binary data: Signed Integer



2.3 Interpreting binary data: Signed Integer

```
  1 1 0 1 0 1 1 0          Two's complement of 214:
  ---------------          ------------------------

  0 0 1 0 1 0 0 1          1. Invert all single bits
+ 0 0 0 0 0 0 0 1          2. Add the value 00000001
-----------------

  0 0 1 0 1 0 1 0

      |   |   |

     32   8   2            3. Convert to Decimal: -42
```

2.3 Interpreting binary data: Signed Integer

```
  1 1 0 1 0 0 0 0
  ---------------

+
-----------------
```

Two's complement of 108:

1. Invert all single bits
2. Add the value 00000001

3. Convert to Decimal:

2.3 Interpreting binary data: Signed Integer

```
  1 1 0 1 0 0 0 0
  ---------------

  0 0 1 0 1 1 1 1
+ 0 0 0 0 0 0 0 1
-----------------

  0 0 1 1 0 0 0 0

      | |

     32 16
```

Two's complement of 108:

1. Invert all single bits
2. Add the value 00000001
3. Convert to Decimal: -48

2.4 Challenge: 1 byte signed Integer

- Biggest possible positive number

  $\longrightarrow$

- Smalest possible positive number

  $\longrightarrow$

- Biggest possible negative number

  _____

  _____ $\longrightarrow$

- Smalest possible negative number

  _____

  _____ $\longrightarrow$

2.4 Challenge: 1 byte signed Integer

- Biggest possible positive number

  `0111 1111` —>

- Smalest possible positive number

  `0000 0000` —>

- Biggest possible negative number

  _____

  _____ —>

- Smalest possible negative number

  _____

  _____ —>

2.4 Challenge: 1 byte signed Integer

- Biggest possible positive number

  `0111 1111` —> `127`

- Smalest possible positive number

  `0000 0000` —> `0`

- Biggest possible negative number

  _____

  _____ —>

- Smalest possible negative number

  _____

  _____ —>

2.4 Challenge: 1 byte signed Integer

- Biggest possible positive number

  0111 1111 —> 127

- Smalest possible positive number

  0000 0000 —> 0

- Biggest possible negative number

  ```
  1111 1111
  ---------
  0000 0000
  0000 0001      —>
  ---------
  0000 0001
  ```

- Smalest possible negative number

  ```
  1000 0000
  ---------
  0111 1111
  0000 0001      —>
  ---------
  1000 0000
  ```

2.4 Challenge: 1 byte signed Integer

- Biggest possible positive number

  0111 1111 —> 127

- Smalest possible positive number

  0000 0000 —> 0

- Biggest possible negative number

  1111 1111

  0000 0000
  0000 0001 —> −1

  0000 0001

- Smalest possible negative number

  1000 0000

  0111 1111
  0000 0001 —> −128

  1000 0000

2.5 From Bin to Hex

Example:

0001 1000	0101 0101	0000 1111	1010 0110
1 8	5 5	0 F	A 6

Exercise:

1001 0110	1010 0101	0000 1111	1100 0011

2.5 From Bin to Hex

Example:

0001 1000	0101 0101	0000 1111	1010 0110
1 8	5 5	0 F	A 6

Exercise:

1001 0110	1010 0101	0000 1111	1100 0011
9 6	A 5	0 F	C 3

2.6 Big Endian and Little Endian

Multibyte words:

Example: 256 in Big Endian representation:

2^:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data:	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
Address:	10.000								10.001							

Multibyte words:

Example: 256 in Little Endian representation:

2^:	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8
Data:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Address:	10.000								10.001							

2.6 Exercise: Little Endian

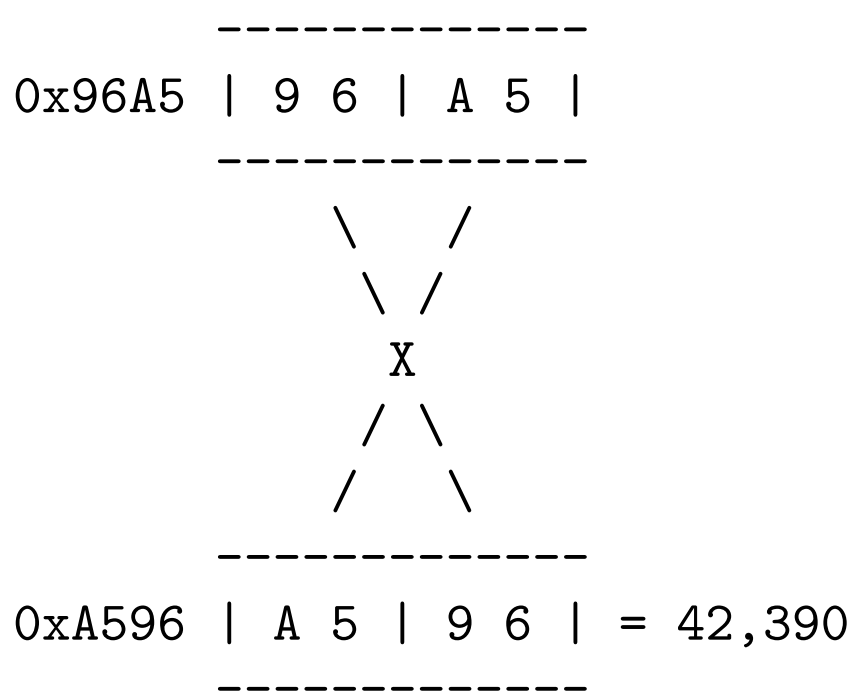
Read and interpret this little endian 2 byte 'word'

```
          -------------
0x96A5 | 9 6 | A 5 |
          -------------
```

```
          -------------
0x        |     |     | =
          -------------
```

2.6 Exercise: Little Endian

Read and interpret this little endian 2 byte 'word'



2.6 Demo: Little Endian

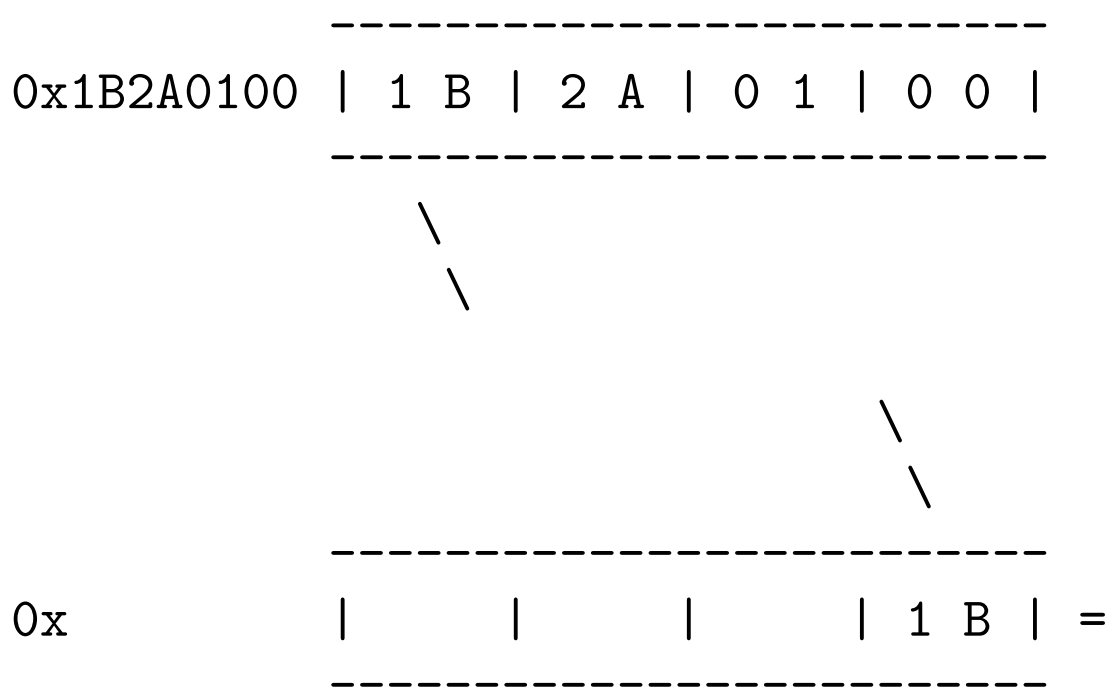
Read and interpret this little endian 'double word'

```
                ------------------------
0x1B2A0100 | 1 B | 2 A | 0 1 | 0 0 |
                ------------------------
```

```
                ------------------------
0x              |     |     |     |     | =
                ------------------------
```

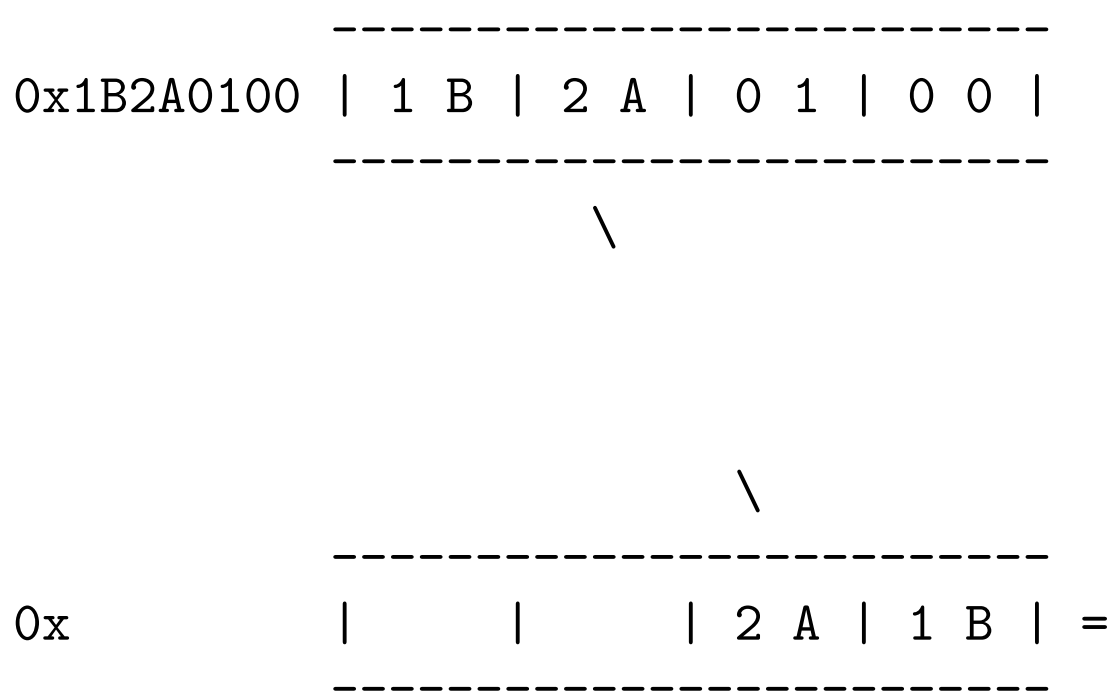
2.6 Demo: Little Endian

Read and interpret this little endian 'double word'



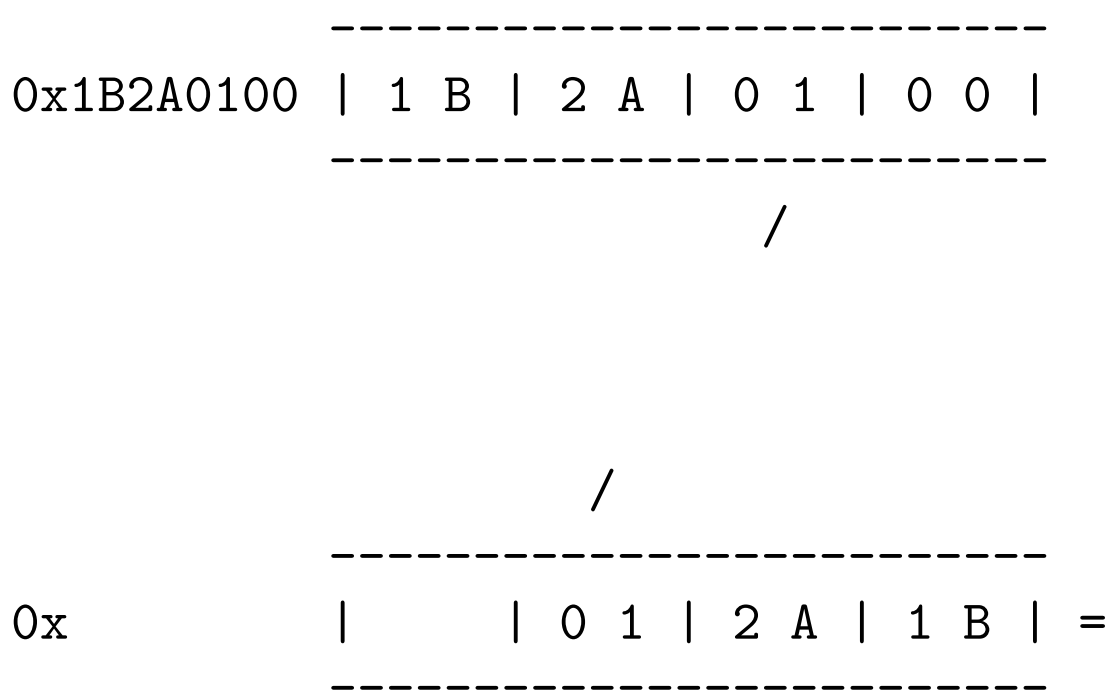
2.6 Demo: Little Endian

Read and interpret this little endian 'double word'



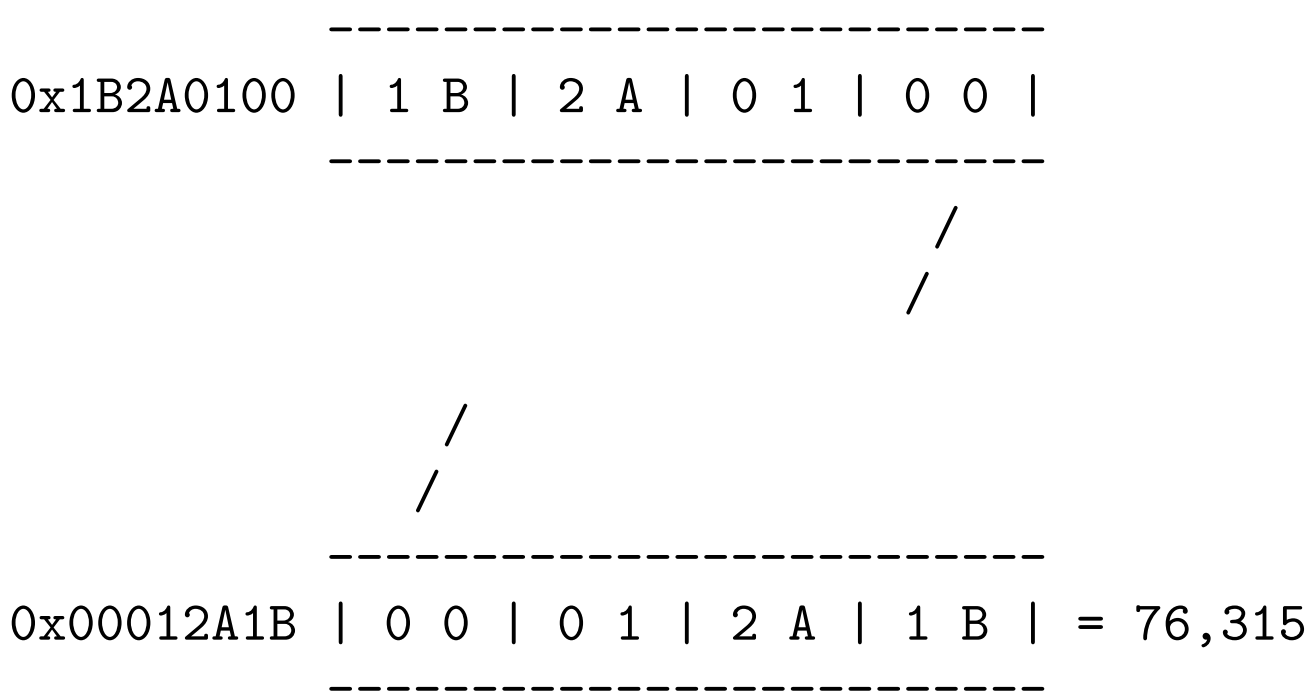
2.6 Demo: Little Endian

Read and interpret this little endian 'double word'



2.6 Demo: Little Endian

Read and interpret this little endian 'double word'



2.6 Exercise: Little Endian

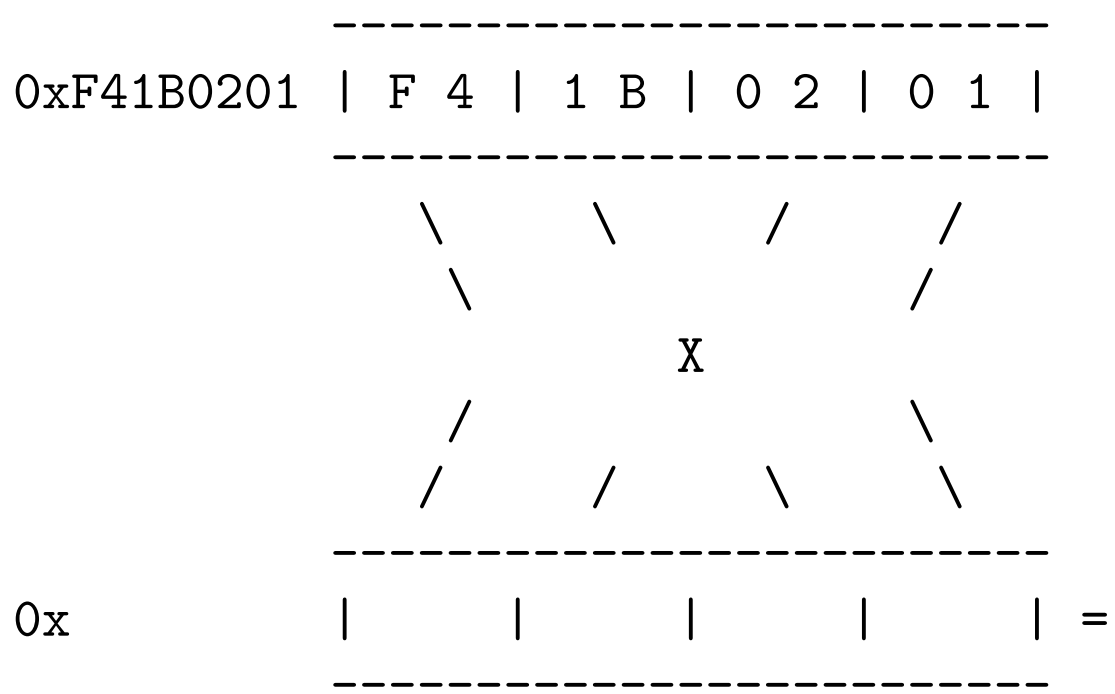
Read and interpret this little endian 'double word'

```
                 -------------------------
0xF41B0201 | F 4 | 1 B | 0 2 | 0 1 |
                 -------------------------
```

```
                 -------------------------
0x               |     |     |     |     | =
                 -------------------------
```

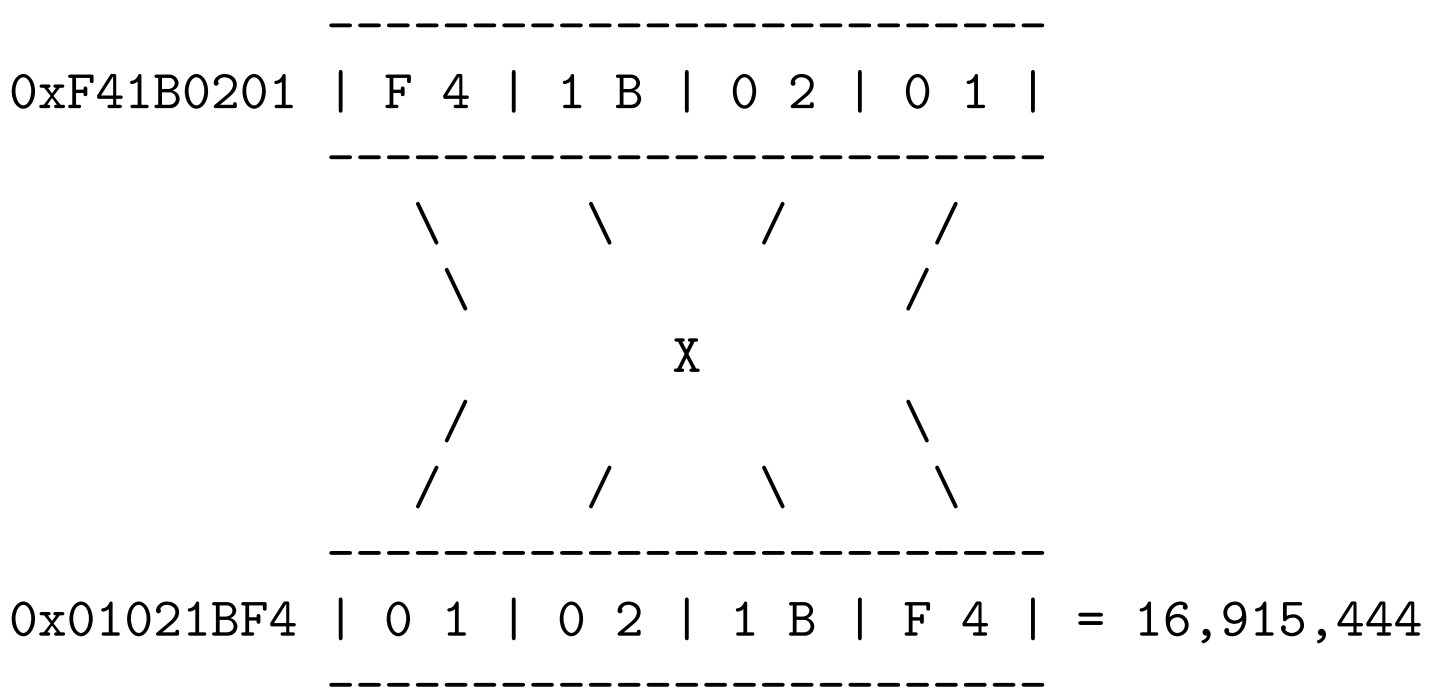
2.6 Exercise: Little Endian

Read and interpret this little endian 'double word'



2.6 Exercise: Little Endian

Read and interpret this little endian 'double word'



2.7 Other interpretation of binary data

ASCII

```
01110000  01101001  01101110  01100111
--------  --------  --------  --------
  0x70      0x65      0x6E      0x67
   112       105       110       103
    p         i         n         g
```

BCD / PBCD

```
   2         9         1        |     6  na     0  9
-------- -------- --------      |   -------- --------
00000010 00001001 00000001      |   01101010 00001001
```

(Double) Float, Unicode, Boolean Flags, Assembler Instruction, ...

2.8 Data structures: Exercise

- Can you read this data?
- Can you extract information out of this data?
- Can you generate knowledge out of this data?

```
01000100010001100100010101000001000010000000111000000000011111111
10111010001100101011100110111010000101110011101000111100001110 1
00001000100100100001100101011011000110110001101111001000000101 0
11101101111011100100110110001100100001000100000110101000100010 0
01100100010101000001000001110000000100000001000000000011001000 11
00110011010010111001000101110001101100011010001010010010001010 1
01101001001010010101010110100101000010011110010110010001010111 0
11110000110110001100101011010010011010001001011000010101111111 1
11111111111111111111111
```

2.8 Data structures: Organizing data

0	8	16
44 46 45 41 08 0E 00 FF	74 65 73 74 2E 74 78 74	22 48 65 6C 6C 6F 20 57

24	32	40
6F 72 6C 64 22 0D 44 46	45 41 07 11 00 00 64 66	69 72 2E 36 34 52 45 5A

48	56	64
4A 55 69 42 79 64 57 78	6C 65 69 34 4B 0A FF FF	FF FF

2.8 Data structures: Definition of the structure

0	8	16
|44|46|45|41|08|0E|00|FF|74|65|73|74|2E|74|78|74|22|48|65|6C|6C|6F|20|57|		

24	32	40
|6F|72|6C|64|22|0D|44|46|45|41|07|11|00|00|64|66|69|72|2E|36|34|52|45|5A|		

48	56	64
|4A|55|69|42|79|64|57|78|6C|65|69|34|4B|0A|FF|FF|FF|FF| | | | | | |		

Offset	Size	Description
0	4	Header signature (ASCII: DFEA – Digital Forensics EDU Archive)
4	1	Lenght of file name (Integer)
5	2	Lenght of data (Little Endian)
7	1	Type of data (Signed Integer) (–1 = ASCII; 0 = base64 encoded)
8	—	Variable file name (ASCII)
9++	—	Data (Binary)
EOF	4	EOF signature (Binary: FF FF FF FF)

2.8 Data structures: Apply structure

0	8	16
44 46 45 41 08 0E 00 FF	74 65 73 74 2E 74 78 74	22 48 65 6C 6C 6F 20 57

24	32	40
6F 72 6C 64 22 0D 44 46	45 41 07 11 00 00 64 66	69 72 2E 36 34 52 45 5A

48	56	64
4A 55 69 42 79 64 57 78	6C 65 69 34 4B 0A FF FF	FF FF

Offset	Size	Description
0	4	Header signature (ASCII: DFEA – Digital Forensics EDU Archive)
4	1	Lenght of file name (Integer)
5	2	Lenght of data (Little Endian)
7	1	Type of data (Signed Integer) (–1 = ASCII; 0 = base64 encoded)
8	—	Variable file name (ASCII)
9++	—	Data (Binary)
EOF	4	EOF signature (Binary: FF FF FF FF)

2.8 Data structures: Read information

0	8	16
|44|46|45|41|08|0E|00|FF|74|65|73|74|2E|74|78|74|22|48|65|6C|6C|6F|20|57|		
| D F E A| | | |		

24	32	40
|6F|72|6C|64|22|0D|44|46|45|41|07|11|00|00|64|66|69|72|2E|36|34|52|45|5A|		

48	56	64
|4A|55|69|42|79|64|57|78|6C|65|69|34|4B|0A|FF|FF|FF|FF| | | | | | |		

Offset	Size	Description
0	4	Header signature (ASCII: DFEA – Digital Forensics EDU Archive)
4	1	Lenght of file name (Integer)
5	2	Lenght of data (Little Endian)
7	1	Type of data (Signed Integer) (–1 = ASCII; 0 = base64 encoded)
8	—	Variable file name (ASCII)
9++	—	Data (Binary)
EOF	4	EOF signature (Binary: FF FF FF FF)

2.8 Data structures: Read information

0	8	16
|44|46|45|41|08|0E|00|FF|74|65|73|74|2E|74|78|74|22|48|65|6C|6C|6F|20|57|		
| D F E A| 8| | | |		

24	32	40
|6F|72|6C|64|22|0D|44|46|45|41|07|11|00|00|64|66|69|72|2E|36|34|52|45|5A|		

48	56	64
|4A|55|69|42|79|64|57|78|6C|65|69|34|4B|0A|FF|FF|FF|FF| | | | | | |		

Offset	Size	Description
0	4	Header signature (ASCII: DFEA – Digital Forensics EDU Archive)
4	1	Lenght of file name (Integer)
5	2	Lenght of data (Little Endian)
7	1	Type of data (Signed Integer) (–1 = ASCII; 0 = base64 encoded)
8	—	Variable file name (ASCII)
9++	—	Data (Binary)
EOF	4	EOF signature (Binary: FF FF FF FF)

2.8 Data structures: Read information

```
 0                               8                               16
|44|46|45|41|08|0E|00|FF|74|65|73|74|2E|74|78|74|22|48|65|6C|6C|6F|20|57|
| D  F  E  A| 8|  14 |  |                                |

 24                              32                              40
|6F|72|6C|64|22|0D|44|46|45|41|07|11|00|00|64|66|69|72|2E|36|34|52|45|5A|
                     |

 48                              56                              64
|4A|55|69|42|79|64|57|78|6C|65|69|34|4B|0A|FF|FF|FF|FF|  |  |  |  |  |  |
```

Offset	Size	Description
0	4	Header signature (ASCII: DFEA – Digital Forensics EDU Archive)
4	1	Lenght of file name (Integer)
5	2	Lenght of data (Little Endian)
7	1	Type of data (Signed Integer) (–1 = ASCII; 0 = base64 encoded)
8	—	Variable file name (ASCII)
9++	—	Data (Binary)
EOF	4	EOF signature (Binary: FF FF FF FF)

2.8 Data structures: Read information

```
 0                           8                           16
|44|46|45|41|08|0E|00|FF|74|65|73|74|2E|74|78|74|22|48|65|6C|6C|6F|20|57|
| D  F  E  A| 8|  14 |-1|                          |

 24                          32                          40
|6F|72|6C|64|22|0D|44|46|45|41|07|11|00|00|64|66|69|72|2E|36|34|52|45|5A|
                     |

 48                          56                          64
|4A|55|69|42|79|64|57|78|6C|65|69|34|4B|0A|FF|FF|FF|FF|  |  |  |  |  |  |
```

Offset	Size	Description
0	4	Header signature (ASCII: DFEA – Digital Forensics EDU Archive)
4	1	Lenght of file name (Integer)
5	2	Lenght of data (Little Endian)
7	1	Type of data (Signed Integer) (–1 = ASCII; 0 = base64 encoded)
8	—	Variable file name (ASCII)
9++	—	Data (Binary)
EOF	4	EOF signature (Binary: FF FF FF FF)

2.8 Data structures: Apply informtion

0	8	16
44 46 45 41 08 0E 00 FF 74 65 73 74 2E 74 78 74 22 48 65 6C 6C 6F 20 57		
D F E A 8 14 −1		

24	32	40
6F 72 6C 64 22 0D 44 46 45 41 07 11 00 00 64 66 69 72 2E 36 34 52 45 5A		
|		

48	56	64
4A 55 69 42 79 64 57 78 6C 65 69 34 4B 0A FF FF FF FF		

Offset	Size	Description
0	4	Header signature (ASCII: DFEA – Digital Forensics EDU Archive)
4	1	Lenght of file name (Integer)
5	2	Lenght of data (Little Endian)
7	1	Type of data (Signed Integer) (−1 = ASCII; 0 = base64 encoded)
8	—	Variable file name (ASCII)
9++	—	Data (Binary)
EOF	4	EOF signature (Binary: FF FF FF FF)

2.8 Data structures: Interprete bytes

```
 0                       8                       16
|44|46|45|41|08|0E|00|FF|74|65|73|74|2E|74|78|74|22|48|65|6C|6C|6F|20|57|
| D  F  E  A| 8|  14 |-1| t  e  s  t  .  t  x  t |

 24                      32                      40
|6F|72|6C|64|22|0D|44|46|45|41|07|11|00|00|64|66|69|72|2E|36|34|52|45|5A|
                     |

 48                      56                      64
|4A|55|69|42|79|64|57|78|6C|65|69|34|4B|0A|FF|FF|FF|FF|  |  |  |  |  |  |
```

Offset	Size	Description
0	4	Header signature (ASCII: DFEA – Digital Forensics EDU Archive)
4	1	Lenght of file name (Integer)
5	2	Lenght of data (Little Endian)
7	1	Type of data (Signed Integer) (–1 = ASCII; 0 = base64 encoded)
8	—	Variable file name (ASCII)
9++	—	Data (Binary)
EOF	4	EOF signature (Binary: FF FF FF FF)

2.8 Data structures: Interprete bytes

```
 0                               8                               16
|44|46|45|41|08|0E|00|FF|74|65|73|74|2E|74|78|74|22|48|65|6C|6C|6F|20|57|
| D  F  E  A| 8|  14 |-1| t  e  s  t  .  t  x  t |"  H  e  l  l  o     W

 24                              32                              40
|6F|72|6C|64|22|0D|44|46|45|41|07|11|00|00|64|66|69|72|2E|36|34|52|45|5A|
  o  r  l  d  " CR|

 48                              56                              64
|4A|55|69|42|79|64|57|78|6C|65|69|34|4B|0A|FF|FF|FF|FF|  |  |  |  |  |  |
```

Offset	Size	Description
0	4	Header signature (ASCII: DFEA – Digital Forensics EDU Archive)
4	1	Lenght of file name (Integer)
5	2	Lenght of data (Little Endian)
7	1	Type of data (Signed Integer) (–1 = ASCII; 0 = base64 encoded)
8	—	Variable file name (ASCII)
9++	—	Data (Binary)
EOF	4	EOF signature (Binary: FF FF FF FF)

2.8 Data structures: Exercise: Your turn

```
 0                               8                               16
|44|46|45|41|08|0E|00|FF|74|65|73|74|2E|74|78|74|22|48|65|6C|6C|6F|20|57|
| D  F  E  A| 8|  14 |-1| t  e  s  t  .  t  x  t |"  H  e  l  l  o     W

 24                              32                              40
|6F|72|6C|64|22|0D|44|46|45|41|07|11|00|00|64|66|69|72|2E|36|34|52|45|5A|
  o  r  l  d  " CR|

 48                              56                              64
|4A|55|69|42|79|64|57|78|6C|65|69|34|4B|0A|FF|FF|FF|FF|  |  |  |  |  |  |
```

```
Offset  Size  Description
     0     4  Header signature (ASCII: DFEA - Digital Forensics EDU Archive)
     4     1  Lenght of file name (Integer)
     5     2  Lenght of data (Little Endian)
     7     1  Type of data (Signed Integer)    (-1 = ASCII; 0 = base64 encoded)
     8   --   Variable file name (ASCII)
    9++  --   Data (Binary)
_    EOF   4  EOF signature (Binary: FF FF FF FF)
```

2.8 Data structures: Exercise: Solution

```
 0                               8                               16
|44|46|45|41|08|0E|00|FF|74|65|73|74|2E|74|78|74|22|48|65|6C|6C|6F|20|57|
| D  F  E  A| 8|  14 |-1| t  e  s  t  .  t  x  t |"  H  e  l  l  o     W

 24                              32                              40
|6F|72|6C|64|22|0D|44|46|45|41|07|11|00|00|64|66|69|72|2E|36|34|52|45|5A|
  o  r  l  d "CR| D  F  E  A| 7|  17 | 0| d  f  i  r  .  6  4| R  E  Z

 48                              56                              64
|4A|55|69|42|79|64|57|78|6C|65|69|34|4B|0A|FF|FF|FF|FF|  |  |  |  |  |  |
  J  U  i  B  y  d  W  x  l  e  i  4  K NL|FF FF FF FF|
```

Offset	Size	Description
0	4	Header signature (ASCII: DFEA – Digital Forensics EDU Archive)
4	1	Lenght of file name (Integer)
5	2	Lenght of data (Little Endian)
7	1	Type of data (Signed Integer) (–1 = ASCII; 0 = base64 encoded)
8	—	Variable file name (ASCII)
9++	—	Data (Binary)
EOF	4	EOF signature (Binary: FF FF FF FF)

2.9 From Data to Knowledge

- Bit/Bytes + Addressing + Structur + Interpretation
  $\rightarrow$ Information
- Where do you find this information?
  - Binary inside TEMP folder
  - Autorun folder
  - Registry
  - Browser history
  - Command line history

  Information $\rightarrow$ Knowledge

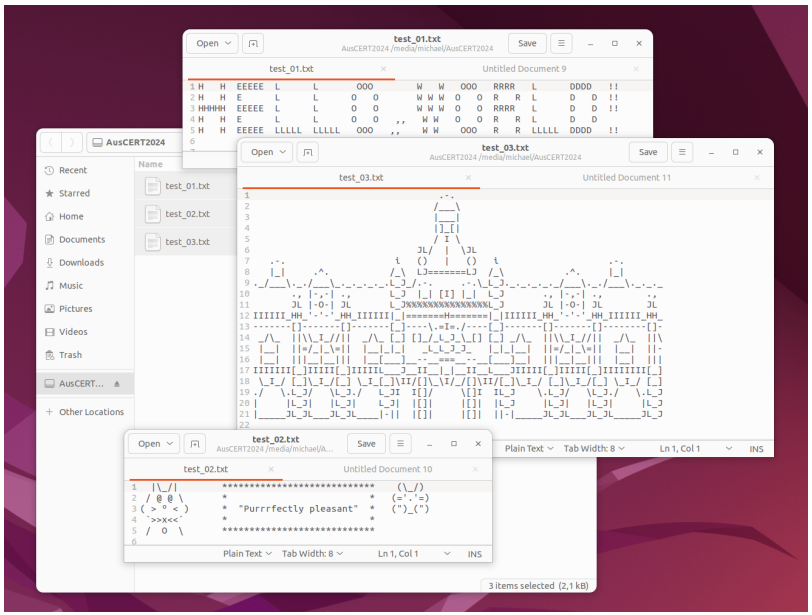
- Information $\rightarrow$ Stored in files
- Files $\rightarrow$ Contains data
- Files $\rightarrow$ Data organized in data structures
- Files $\rightarrow$ Meta data describe files
- Files $\rightarrow$ File systems organize files and meta data

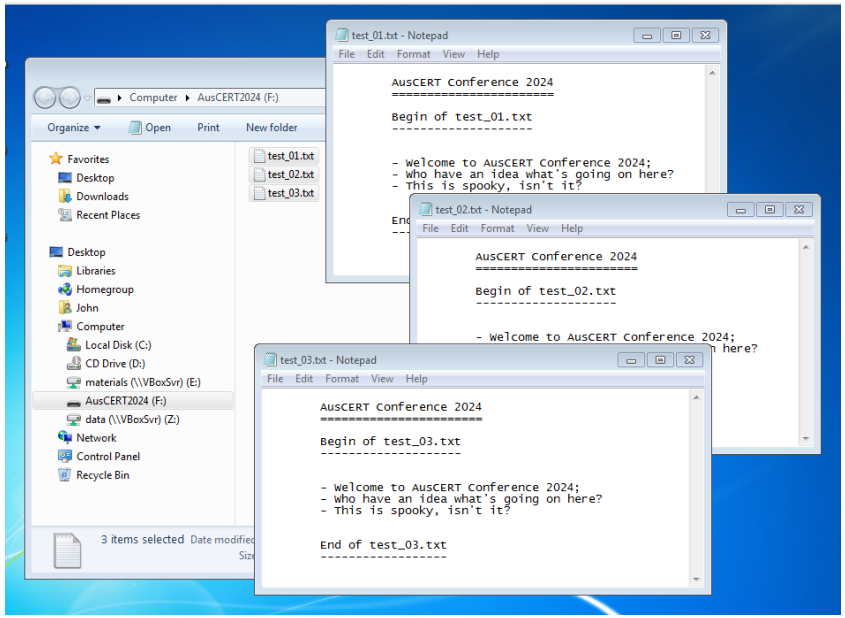
3 Solution

Remember: We still like to understand what's going on here

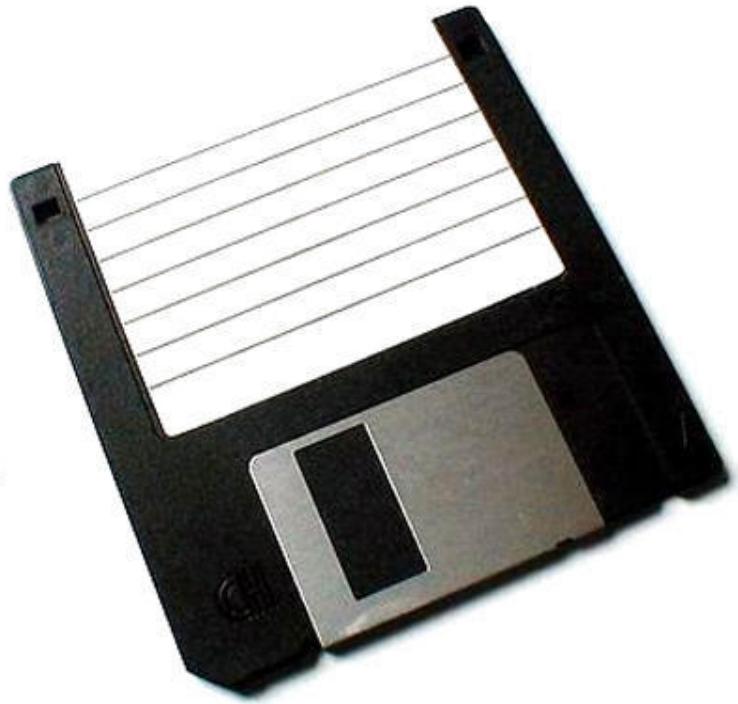
Download disk image:

`https://www.circl.lu/assets/files/AusCERT24_materials.7z`





3.1 Volume Boot Record - Boot Sector



```
-------------------------------------------------------------------------------
||V|  test_01.txt                                                             |
||B|           test_02.txt                                                    |
||R|                    test_03.txt                                           |
-------------------------------------------------------------------------------
  ^
  | Boot Sector
```

3.1 Volume Boot Record - Boot Sector

```
00000000: eb52 904e 5446 5320 2020 2000 0208 0000  .R.NTFS    .....
00000010: 0000 0000 00f8 0000 0000 0000 0000 0000  ................
00000020: 0000 0000 8000 8000 ffff 0300 0000 0000  ................
00000030: 0400 0000 0000 0000 ff3f 0000 0000 0000  .........?......
00000040: f600 0000 0100 0000 3c91 9a52 e282 f91a  ........<..R....
00000050: 0000 0000 0e1f be71 7cac 22c0 740b 56b4  .......q|.".t.V.
00000060: 0ebb 0700 cd10 5eeb f032 e4cd 16cd 19eb  ......^..2......
00000070: fe54 6869 7320 6973 206e 6f74 2061 2062  .This is not a b
00000080: 6f6f 7461 626c 6520 6469 736b 2e20 506c  ootable disk. Pl
00000090: 6561 7365 2069 6e73 6572 7420 6120 626f  ease insert a bo
000000a0: 6f74 6162 6c65 2066 6c6f 7070 7920 616e  otable floppy an
000000b0: 640d 0a70 7265 7373 2061 6e79 206b 6579  d..press any key
000000c0: 2074 6f20 7472 7920 6167 6169 6e20 2e2e   to try again ..
000000d0: 2e20 0d0a 0000 0000 0000 0000 0000 0000  . ..............
...
000001e0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000001f0: 0000 0000 0000 0000 0000 0000 0000 55aa  ..............U.
```

```
00 - 02          Size:  3    Jump to bootstrap code
03 - 0A          Size:  8    OEM-ID: NTFS
0B - 0C          Size:  2    Bytes per sector: 0x0002 -> 0x0200 (little endian)-> 512
0D               Size:  1    Sectors per cluster: 0x08 -> 4096 bytes per cluster
...
54                           Bootstrap code
1FE              Size:  2    Signature: 0x55AA
```

3.2 Master Boot Record & Partitioned Disk

Sector, Head, Track, Cylinder / Cluster, Block / CHS, LBA

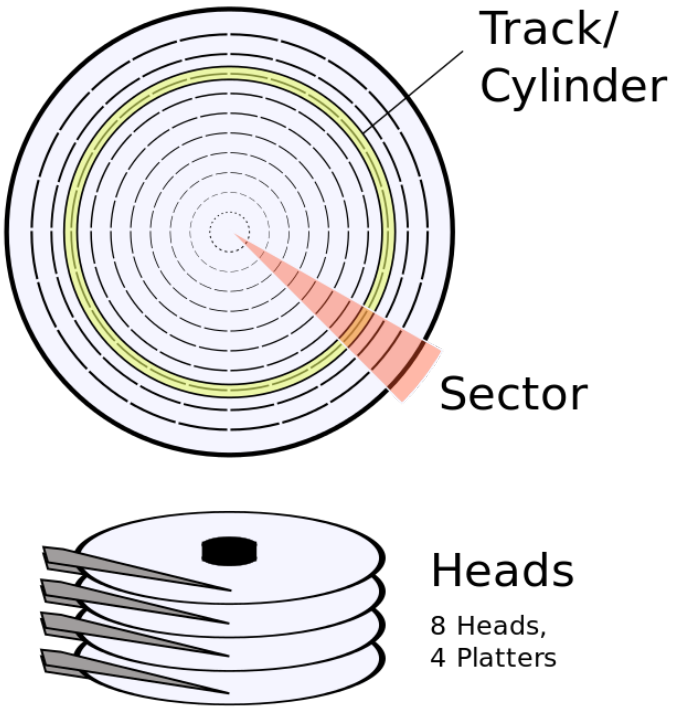
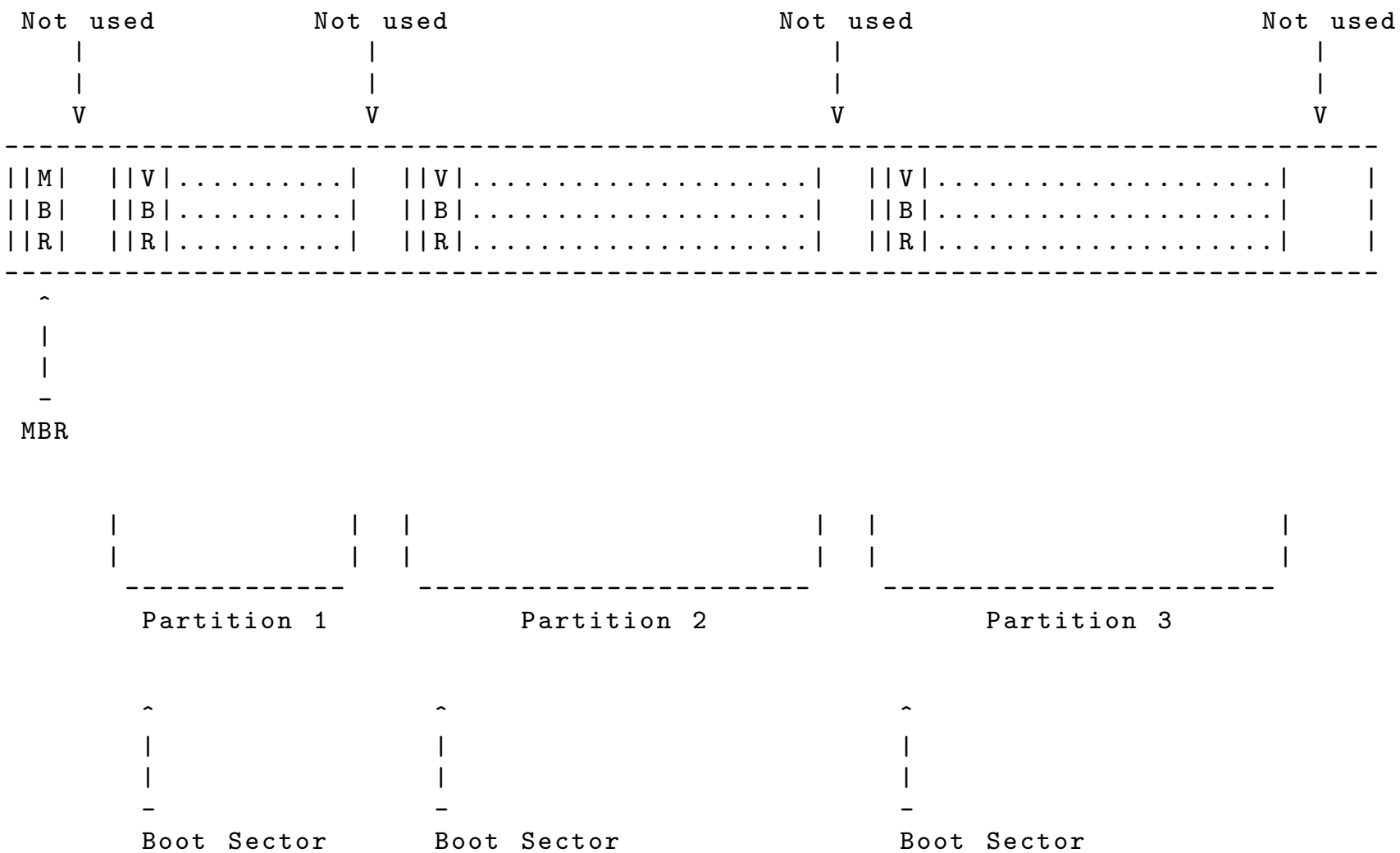


Image (c) wikipedia.org - Image used solely for illustration purposes

3.2 Master Boot Record & Partitioned Disk



3.2 Master Boot Record & Partitioned Disk

```
00000000: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000010: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000020: 0000 0000 0000 0000 0000 0000 0000 0000  ................
...
000001a0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000001b0: 0000 0000 0000 0000 ce69 d7db 0000 00f5  .........i......
000001c0: 2e08 0751 0110 8032 0200 80cd 0100 0000  ...Q...2........
000001d0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000001e0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000001f0: 0000 0000 0000 0000 0000 0000 0000 55aa  ..............U.
```

```
000             Size: 439       Boot code
1B8             Size:   4       Disc signature
1BC             Size:   2       Reserved
1BE             Size   16       Partitiontable entry 1
1CE             Size   16       Partitiontable entry 2
1DE             Size   16       Partitiontable entry 3
1EE             Size   16       Partitiontable entry 4
1FE             Size    2       0x55AA
```

3.2 Master Boot Record & Partitioned Disk

```
00000000: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000010: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000020: 0000 0000 0000 0000 0000 0000 0000 0000  ................
...
000001a0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000001b0: 0000 0000 0000 0000 ce69 d7db 0000 00f5  .........i......
000001c0: 2e08 0751 0110  8032  0200 80cd  0100 0000  ...Q...2........
000001d0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000001e0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000001f0: 0000 0000 0000 0000 0000 0000 0000 55aa  ..............U.
```

```
Partitiontable:
  Offset: 0     Size: 1 Value: 0x80     --> Bootable
  Offset: 1     Size: 3 Value:          --> Starting CHS address
  Offset: 4     Size: 1 Value: 0x0b     --> FAT32
                               0x07     --> NTFS
  Offset: 5     Size: 3 Value:          --> Ending CHS address
  Offset: 8     Size: 4 Value:          --> Starting LBA address (Little Endian)
  Offset:12     Size: 4 Value:          --> LBA size in sectors  (Little Endian)
```

Exercise: Calculate partition Start and Size:

3.2 Master Boot Record & Partitioned Disk

```
00000000: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000010: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000020: 0000 0000 0000 0000 0000 0000 0000 0000  ................
...
000001a0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000001b0: 0000 0000 0000 0000 ce69 d7db 0000 00f5  .........i......
000001c0: 2e08 0751 0110 8032 0200 80cd 0100 0000  ...Q...2........
000001d0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000001e0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000001f0: 0000 0000 0000 0000 0000 0000 0000 55aa  ..............U.
```

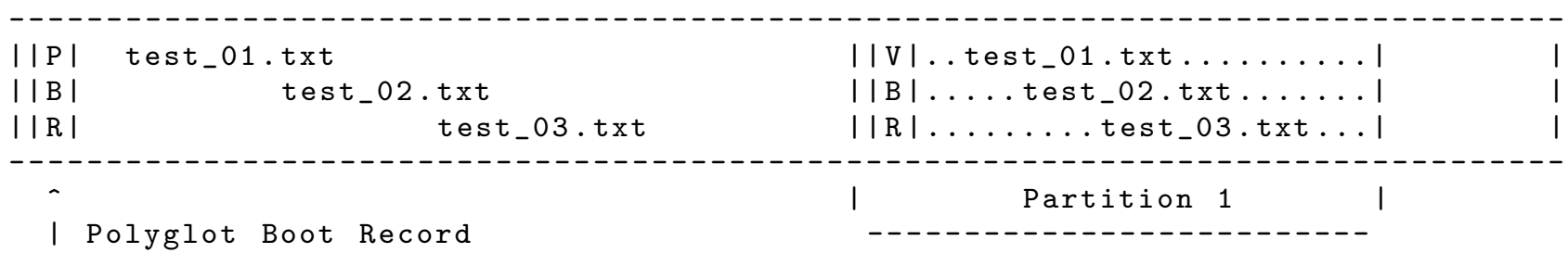
```
Partitiontable:
  Offset: 0      Size: 1 Value: 0x80      --> Bootable
  Offset: 1      Size: 3 Value:           --> Starting CHS address
  Offset: 4      Size: 1 Value: 0x0b      --> FAT32
                                0x07      --> NTFS
  Offset: 5      Size: 3 Value:           --> Ending CHS address
  Offset: 8      Size: 4 Value:           --> Starting LBA address (Little Endian)
  Offset:12      Size: 4 Value:           --> LBA size in sectors  (Little Endian)
```

Exercise: Calculate partition Start and Size: 144000, 118114

3.3 Polyglot Boot Record

```
00000000: eb52 904e 5446 5320 2020 2000 0208 0000  .R.NTFS    .....
00000010: 0000 0000 00f8 0000 0000 0000 0000 0000  ................
00000020: 0000 0000 8000 8000 ffff 0300 0000 0000  ................
00000030: 0400 0000 0000 0000 ff3f 0000 0000 0000  .........?......
00000040: f600 0000 0100 0000 af3c 3363 b6f7 8833  .........<3c...3
00000050: 0000 0000 0e1f be71 7cac 22c0 740b 56b4  .......q|.".t.V.
00000060: 0ebb 0700 cd10 5eeb f032 e4cd 16cd 19eb  ......^..2......
00000070: fe54 6869 7320 6973 206e 6f74 2061 2062  .This is not a b
00000080: 6f6f 7461 626c 6520 6469 736b 2e20 506c  ootable disk. Pl
00000090: 6561 7365 2069 6e73 6572 7420 6120 626f  ease insert a bo
000000a0: 6f74 6162 6c65 2066 6c6f 7070 7920 616e  otable floppy an

000001b0: 0000 0000 0000 0000 ce69 d7db 0000 00f5  .........i......
000001c0: 2e08 0751 0110 8032 0200 80cd 0100 0000  ...Q...2........
000001d0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000001e0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000001f0: 0000 0000 0000 0000 0000 0000 0000 55aa  ..............U.
```



From Zero to Hero

Digital Forensics 1.0.1

Michael Hamm - CIRCL

TLP:CLEAR

75th TF-CSIRT Meeting 2025