An example of Master Boot Record

Davy
21 Jan 2012

Sometimes when your computer refuses to boot, or your partition table is corrupt you have a "beautiful message" on screen saying your hard drive has an issue, a MBR error. Let's dive inside and decrypt an existing MBR in order to understand what can be found inside it.

Wikipedia, what's a MBR ?

The master boot record is a type of boot sector very popular (for instance Windows and Grub use it). It contains **512 bytes** stored at the first sector of your data storage device (HDD, USB stick).Inside this boot sector can be found:

- **the bootstrapping of your operation** system (described as "code area" in the table),
- an optional unique if for your data storage device (described as "disk signature" in the table)
- **the partition table** (described as "Table of primary partitions" in the table).

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	2
01FF	0777	511	AAh		
				MBR, total size: 446 + 64 + 2 =	512

source : wikipedia

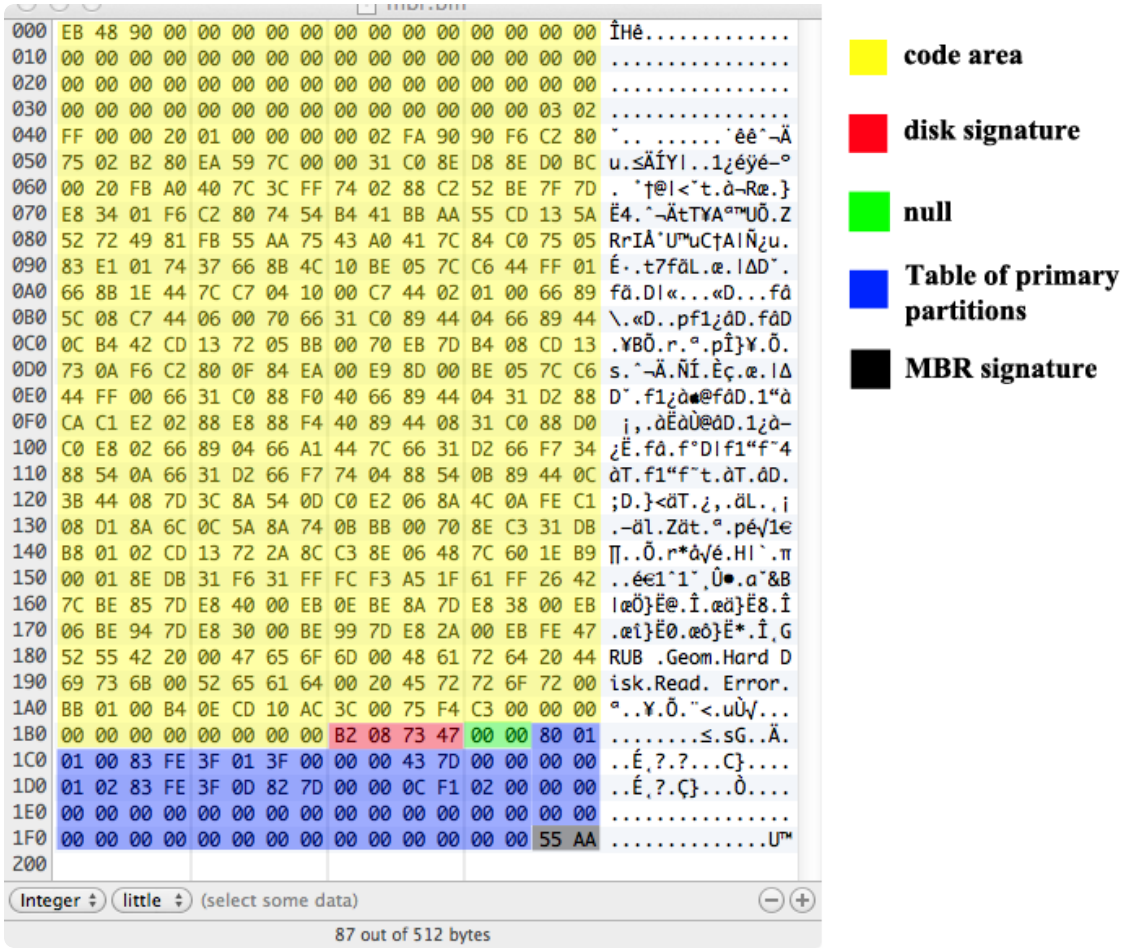
How to extract a Master Boot Record

We can simply use the command dd to copy the raw first 512 bytes of a media

```
$dd if=/dev/YOUR_DATA_STORAGE_DEVICE of=DESTINATION_FIL
E bs=512 count=1
```

An example of Master Boot Record

To provide a readable/hexa version of the MBR, I used an hexadecimal editor but you could use the command hexadump (to display hexa directly from your raw data storage device). And to understand what is displayed I added some sprinkles thanks photoshop =0)



Code area

In this example MBR, the code area contains shellcode and not only zeros padding. So, we can guess that this data storage device contains a bootstrap for an operating system. The goal of this tutorial isn't to understand how a OS bootstrap works, so nothing more to add... Maybe, we can cheat and discover the string GRUB in the plain text - not a reason to be sure it's GRUB but still a tips that confirms our supposition -

Disk signature

Yeah, an id...

Null

Too much said in the name

Table of primary partitions

Here we have four 16 bytes entries describing each primary partition. Now, you know why only 4 primary partitions can be defined for a data storage device. If we create bunch of 16 bytes, we have :

- Partition 1 : 80 01 01 00 83 FE 3F 01 3F 00 00 00 43 7D 00 00
- Partition 2 : 00 00 01 02 83 FE 3F 0D 82 7D 00 00 0C F1 02 00
- Partition 3 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
- Partition 4 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Even without knowing the meaning of those bytes we can say that there is only two partitions in the partition table described by this MBR because we have zero padding for the two last 16 bytes entries

MBR signature

It always end by 55 AA. It's a signature, so by extracting the 512 first bytes of a data storage device you could said if it's a MBR or something else by reading those two last bytes.

Let's decrypt a primary partition entry

Partition 1 : 80 01 01 00 83 FE 3F 01 3F 00 00 00 43 7D 00 00

- 80 : it's the status (80 for a bootable partition, 00 for a non bootable)
- 01 01 00 : Cylinder-head-sector address of the first absolute sector in partition (first byte is the head, 01, the second is "almost the sector" and the last one is the cylinder, 00)
- 83 : partition type in our case 83 means a native Linux file system (Ext2, Ext3 or others) - here to find each type
- FE 3F 01 : Cylinder-head-sector address of the last absolute sector in partition (same format)
- 3F 00 00 00 : Logical block addressing of first absolute sector in the partition
- 43 7D 00 00 : Number of sectors in partition (32067 sectors)

Guess what ?

We have enough information to guess about this data storage device. It contains a bootstrap (GRUB) and two partitions. The first partition is a Linux file system of 32067 sectors (- on a hard drive each sector is 512 bytes -, so 32067*512/1024=16mb) and it's a bootable partition (where the bootstrap, GRUB, will have to read). The second partition is a non bootable partition which contains a Linux file system of 192 780 sectors (192780*512/1024=96mb).

The right answer is ...