Examining the FAT MBR and Partition Table

By Tim Conrad

Understanding drive partitioning is a very powerful tool.
Whether you are in the field of security/forensics or whether you are a hacker, this can be great information to know.

The following example contains a brief look at partition information for the standard MBR format:

Tools you need:

Boot into your favorite "good" linux distribution.
* Good is defined by whether it has the tools you need for examine a drive.
Tools:
fdisk
gdisk
hexeditor
programmers calculator - need to easily convert from hex to dec and back as needed
memorize the number 1,048,576 = The number of bytes in a Megabyte

Examining the MBR

fdisk -l to determine your target drive (If there is more than one drive attached make certain you know which drive you are wanting to examine. This can probably be accomplished by looking at the drive sizes.)

```
Disk /dev/sdf: 2004 MB, 2004877312 bytes
252 heads, 8 sectors/track, 1942 cylinders, total 3915776 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xa4b57300

   Device Boot      Start         End      Blocks   Id  System
/dev/sdf1   *          63     3903550     1951744    6  FAT16
```

Every drive looks different. It depends on what tool partitioned it and how it was partitioned.
Here is an MBR disk most likely formatted in Linux

hexedit /dev/sdf

```
00000000   EB 58 90 6D  6B 64 6F 73  66 73 00 00  02 08 20 00  .X.mkdosfs.... .
00000010   02 00 00 00  00 F8 00 00  3F 00 FF 00  00 00 00 00  ........?.......
00000020   00 C0 3B 00  E9 0E 00 00  00 00 00 00  02 00 00 00  ..;.............
00000030   01 00 06 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000040   00 01 29 DA  56 F6 62 20  20 20 20 20  20 20 20 20  ..).V.b
00000050   20 20 46 41  54 33 32 20  20 20 0E 1F  BE 77 7C AC    FAT32   ...w|.
00000060   22 C0 74 0B  56 B4 0E BB  07 00 CD 10  5E EB F0 32  ".t.V.......^..2
00000070   E4 CD 16 CD  19 EB FE 54  68 69 73 20  69 73 20 6E  .......This is n
00000080   6F 74 20 61  20 62 6F 6F  74 61 62 6C  65 20 64 69  ot a bootable di
00000090   73 6B 2E 20  20 50 6C 65  61 73 65 20  69 6E 73 65  sk.  Please inse
000000A0   72 74 20 61  20 62 6F 6F  74 61 62 6C  65 20 66 6C  rt a bootable fl
000000B0   6F 70 70 79  20 61 6E 64  0D 0A 70 72  65 73 73 20  oppy and..press
```

```
000000C0   61 6E 79 20  6B 65 79 20  74 6F 20 74  72 79 20 61  any key to try a
000000D0   67 61 69 6E  20 2E 2E 2E  20 0D 0A 00  00 00 00 00  gain ... .......
000000E0   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
000000F0   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000100   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000110   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000120   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000130   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000140   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000150   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000160   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
00000170   00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  ................
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 00 00 00  00 73 B5 A4  00 00 80 01  .........s......
000001C0   01 00 06 FB  08 F2 3F 00  00 00 00 90  3B 00 00 00  ......?.....;...
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  ..............U.
```

Looking at the ASCII information on the right side of the MBR infomation we can see that this drive was most likely formatted in linux due to the mkdosfs OEM stamp at the beginning of the partition information.
We can also see that at some point this drive was FAT32. Just because you see this in the ASCII side of the bootstrap code does not mean it is FAT 32 as we will see this momentarily.

Breaking down the MBR:

Location	Length	Value	Definition
00h	3 bytes	EB 58 90	Jump instruction
03h	8 bytes	6D 6B 64 6F 73 66 73 00 mkdosfs	OEM name in text
0Bh	25 bytes		Bios Parameter Block
24h	48 bytes		Extended BPB
54h	426 bytes		Bootstrap Code
1FEh	WORD		End of sector Marker

Looking at the MBR partition information:

The 1st and only partition shows up at 0x1BE in the bootstrap code.

```
000001B0   00 00 00 00  00 00 00 00  00 73 B5 A4  00 00 80 01  .........s......
000001C0   01 00 06 FB  08 F2 3F 00  00 00 00 90  3B 00 00 00  ......?.....;...
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  ..............U.
```

The 80 tells us that this is a bootable drive. It does not mean it has a bootable OS on it, but if it does not then at some point it probably did.

The next 3 bytes 01 01 00 = the starting sector in Cylinder Head Sector values or CHS

The Next byte 06 = the partition type. 06 is FAT16. Here is where we see that the drive is currently FAT16 and not FAT32.

If the partition type is an extended partition type, like 05h, you will need to work your way through the MBR data and then to the extended partition table to see how it is configured. The MBR should be thought of as a "guide" to the actual partition location.

The next three bytes, FB 08 F2 represent the ending CHS

The next 4 bytes of data will tell us where the first sector of the partition table starts
3F 00 00 00
This information is in little-endian format, so we need to flip it. 0000003f
Using our programmers calculator we do the following:
convert 3fh to dec. = 63 sectors
Multiply the decimal value 63 x 512 (our bytes per sector) = 32,256
Convert back to hex
32256d = 7E00h
Remember this 7E00 for later.

The last four bytes lets us know the size of the partition
00 90 3B 00
Remember this is in little-endian format
003B9000
Convert this
003B9000h = 3903488d
3903488 x 512 = 1998585856 / the number you memorized earlier 1048576 = 1906MB or roughly 2GB

Now using the hex editor lets take a look at the actual partition containing our data

hexedit /dev/sdf
Now using the number we found earlier we will search for 7E00

```
00007E00   EB 3C 90 28  68 63 6D 2F  49 48 43 00  02 40 02 00  .<.(hcm/IHC..@..
00007E10   02 00 02 00  00 F8 EF 00  3F 00 FF 00  3F 00 00 00  ........?...?...
00007E20   00 90 3B 00  80 01 29 59  25 DC 56 4E  4F 4E 45 20  ..;...)Y%.VNONE
00007E30   20 20 20 20  20 20 46 41  54 31 36 20  20 20 33 C9        FAT16   3.
00007E40   8E D1 BC F0  7B 8E D9 B8  00 20 8E C0  FC BD 00 7C  ....{.... .....|
00007E50   38 4E 24 7D  24 8B C1 99  E8 3C 01 72  1C 83 EB 3A  8N$}$....<.r...:
00007E60   66 A1 1C 7C  26 66 3B 07  26 8A 57 FC  75 06 80 CA  f..|&f;.&.W.u...
00007E70   02 88 56 02  80 C3 10 73  EB 33 C9 8A  46 10 98 F7  ..V....s.3..F...
00007E80   66 16 03 46  1C 13 56 1E  03 46 0E 13  D1 8B 76 11  f..F..V..F....v.
00007E90   60 89 46 FC  89 56 FE B8  20 00 F7 E6  8B 5E 0B 03  `.F..V.. ....^..
00007EA0   C3 48 F7 F3  01 46 FC 11  4E FE 61 BF  00 00 E8 E6  .H...F..N.a.....
00007EB0   00 72 39 26  38 2D 74 17  60 B1 0B BE  A1 7D F3 A6  .r9&8-t.`....}..
00007EC0   61 74 32 4E  74 09 83 C7  20 3B FB 72  E6 EB DC A0  at2Nt... ;.r....
00007ED0   FB 7D B4 7D  8B F0 AC 98  40 74 0C 48  74 13 B4 0E  .}.}....@t.Ht...
00007EE0   BB 07 00 CD  10 EB EF A0  FD 7D EB E6  A0 FC 7D EB  .........}....}.
00007EF0   E1 CD 16 CD  19 26 8B 55  1A 52 B0 01  BB 00 00 E8  .....&.U.R......
00007F00   3B 00 72 E8  5B 8A 56 24  BE 0B 7C 8B  FC C7 46 F0  ;.r.[.V$..|...F.
00007F10   3D 7D C7 46  F4 29 7D 8C  D9 89 4E F2  89 4E F6 C6  =}.F.)}...N..N..
00007F20   06 96 7D CB  EA 03 00 00  20 0F B6 C8  66 8B 46 F8  ..}..... ...f.F.
00007F30   66 03 46 1C  66 8B D0 66  C1 EA 10 EB  5E 0F B6 C8  f.F.f..f....^...
00007F40   4A 4A 8A 46  0D 32 E4 F7  E2 03 46 FC  13 56 FE EB  JJ.F.2....F..V..
```

```
00007F50   4A 52 50 06  53 6A 01 6A  10 91 8B 46  18 96 92 33  JRP.Sj.j...F...3
00007F60   D2 F7 F6 91  F7 F6 42 87  CA F7 76 1A  8A F2 8A E8  ......B...v.....
00007F70   C0 CC 02 0A  CC B8 01 02  80 7E 02 0E  75 04 B4 42  .........~..u..B
00007F80   8B F4 8A 56  24 CD 13 61  61 72 0B 40  75 01 42 03  ...V$..aar.@u.B.
00007F90   5E 0B 49 75  06 F8 C3 41  BB 00 00 60  66 6A 00 EB  ^.Iu...A...`fj..
00007FA0   B0 42 4F 4F  54 4D 47 52  20 20 20 20  0D 0A 52 65  .BOOTMGR    ..Re
00007FB0   6D 6F 76 65  20 64 69 73  6B 73 20 6F  72 20 6F 74  move disks or ot
00007FC0   68 65 72 20  6D 65 64 69  61 2E FF 0D  0A 44 69 73  her media....Dis
00007FD0   6B 20 65 72  72 6F 72 FF  0D 0A 50 72  65 73 73 20  k error...Press
00007FE0   61 6E 79 20  6B 65 79 20  74 6F 20 72  65 73 74 61  any key to resta
00007FF0   72 74 0D 0A  00 00 00 00  00 00 00 AC  CB D8 55 AA  rt............U.
```

Let's break down this partition table
7E00h = EB 3C 90 = Jump instruction
7E03h = 28 68 63 6D 2F 49 48 43 or (hcm/IHC = OEM Name
0x7E0B = 00 02 or 200h = 512d Bytes per sector = Bytes per sector
0x7E0D = 40h or 64d = sectors per cluster
0x7E0E = 02 00 = 02h or 2d = reserved sectors = Anything larger than 1 indicates the bootstrap code is larger than the partition sector
0x7E10 = 02h or 2d = Number of file allocation tables (in FAT not NTFS)
0x7E11 = 00 02 = 2h or 2d = Root Entries
0X7E13 = 00 00 = Small sectors - 0 = Large sectors are used instead -
0x7E15 = F8 = Media Type - F8 means Hard disk
0x7E16 = EF 00 = Sectors per FAT(File Allocation Table)
0x7E18 = 3F 00 = Sectors per track
0x7E1A = FF 00 = Number of heads
0x7E1C = 3F 00 00 00 = Hidden Sectors
0x7E20 = 00 90 3B 00 = Large Sectors
0x7E24 = 80 = Physical disk number - 80h stands for physical disk - Value only relevant to startup disk so this will often times be 80h
0x7E25 = 01 = Current Head - N/A to Fat partitions
0x7E26 = 29 = Signature
0x7E27 = 59 25 DC 56 = Volume serial number - Unique number to each time the drive is formatted
0x72B = 4E 4F 4E 45 20 20 20 20 20 20 20 = None = Volume Name in text
0x73 = 46 41 54 31 36 20 20 20 = FAT16 = Volume ID in text

As you can see, from a forensics, hacking, or general knowledge perspective, there is a ton of information that can be discovered when examining a partition table.

Sources for guidance came from past experiences and references from the company Active Data Recovery Software