Unit 8: File System Disk Structures

Prof. Li-Pin Chang
ESSLab@NCTU

Disk Layout

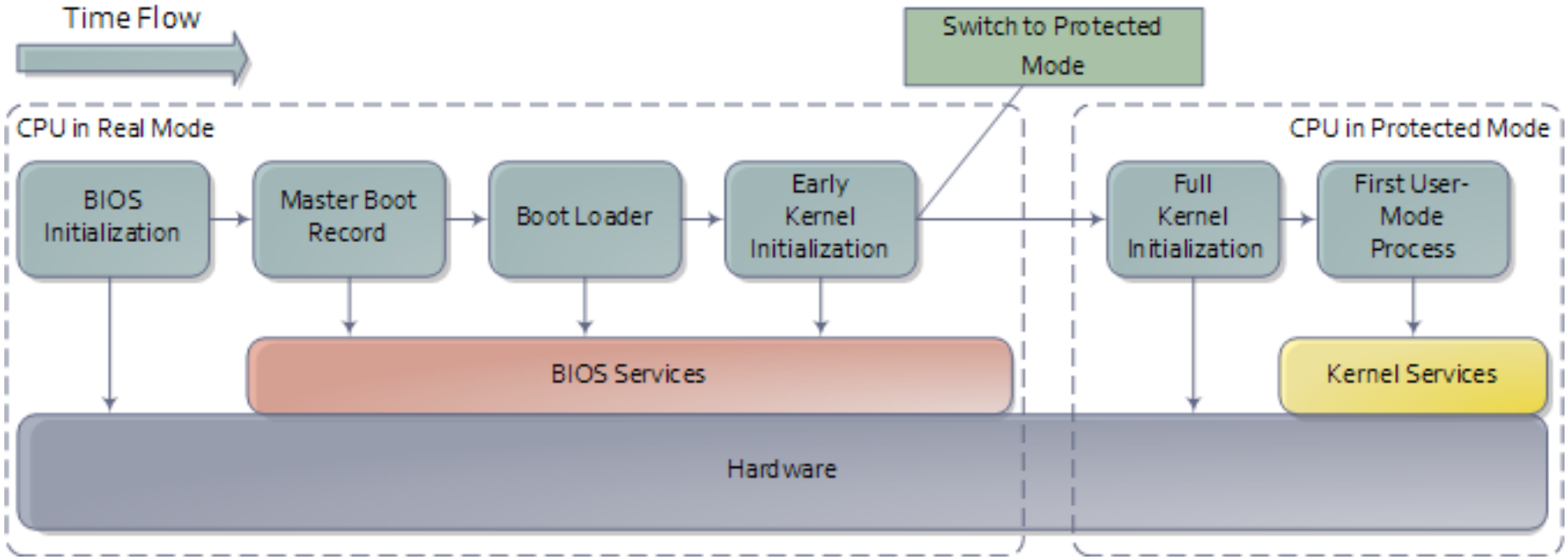
- The entire disk space is addressed by logical sectors, ordered from 0
  - C/H/S mode begins with 0/0/1
- Essential data structures (in order)
  - Boot sector (MBR, Master Boot Record)
    - Boot loader
    - Partition table
  - Partitions
    - formatted in different file systems
- MBR resides at the first sector (absolute sector 0)
  - A disk retires if its track 0 is damaged

Boot Sector Format

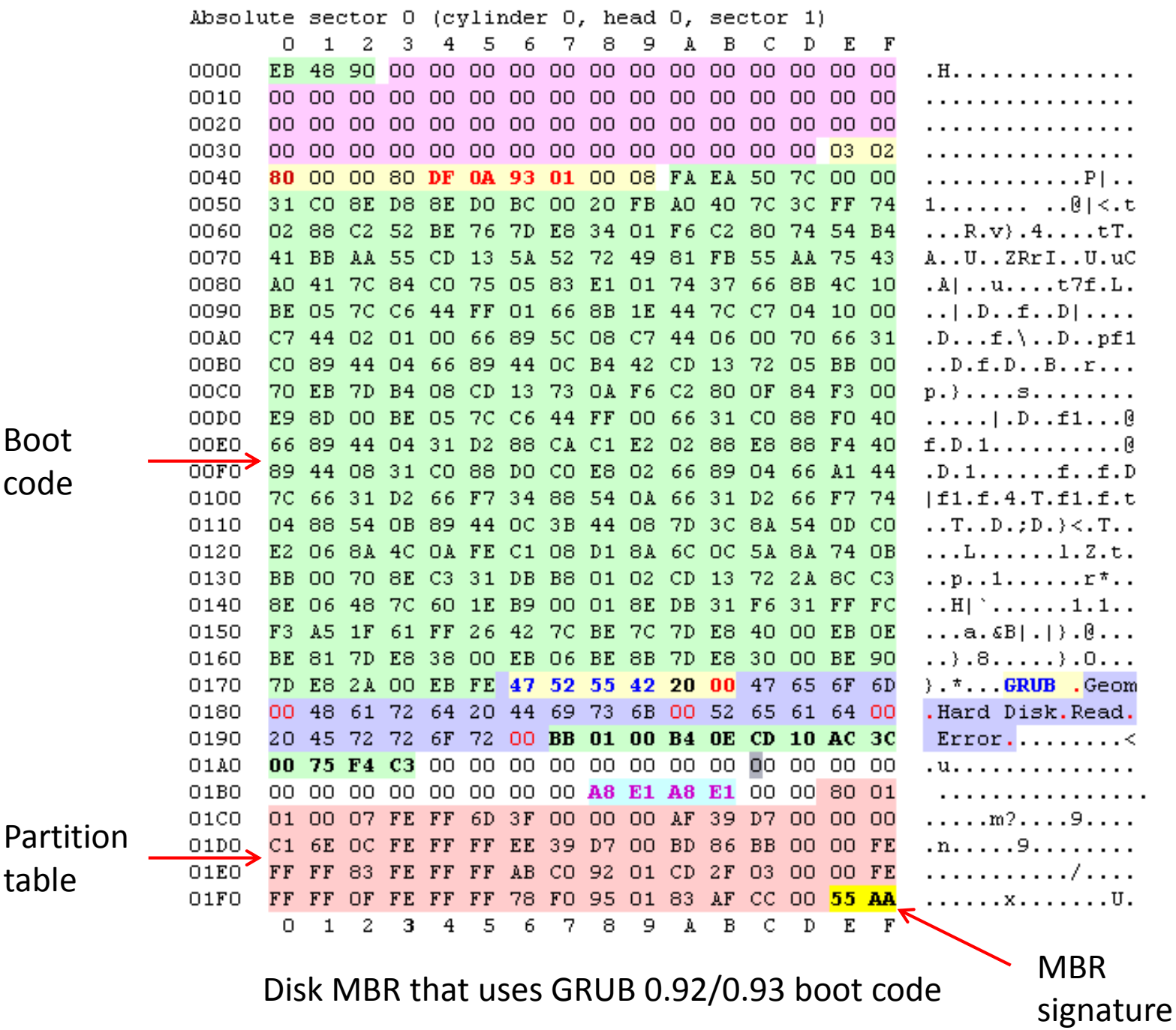
Basic Structure of the Master Boot Record Sector			
Offsets (within sector)		Length (in *bytes*)	**Description**
in Decimal	*in Hex*		
000 – 445	000 – 1BD	446	Code Area
446 – 509	1BE – 1FD	64	*Master* Partition Table
510 – 511	1FE – 1FF	2	Boot Record Signature

- Code area
  - Binary code that loads the OS loader from disk (up to 446 bytes)
- Master partition table
  - Information about the partitions
- Boot record signature
  - 55h, AAh (or AA55h), indicating that this is a valid MBR

PC Boot Sequence



http://static.duartes.org/img/blogPosts/bootProcess.png



Disk MBR that uses GRUB 0.92/0.93 boot code

GRUB

- GRand Unified Bootloader
  - A boot manager that supports multi-boot
  - Actually a mini, self-contained OS (Linux)
  - Requires a dedicated partition
- Multi-stage boot process
  - Stage 1: load the larger stage 1.5 loader
    - This boot loader is in MBR
  - Stage 1.5: load the stage 2 loader
    - This loader is in the reserved space immediately after MBR
  - Stage 2: menu handling, parsing file system in the boot partition, loading the OS loader
    - This loader is in the dedicated partition

GRUB MBR Loader

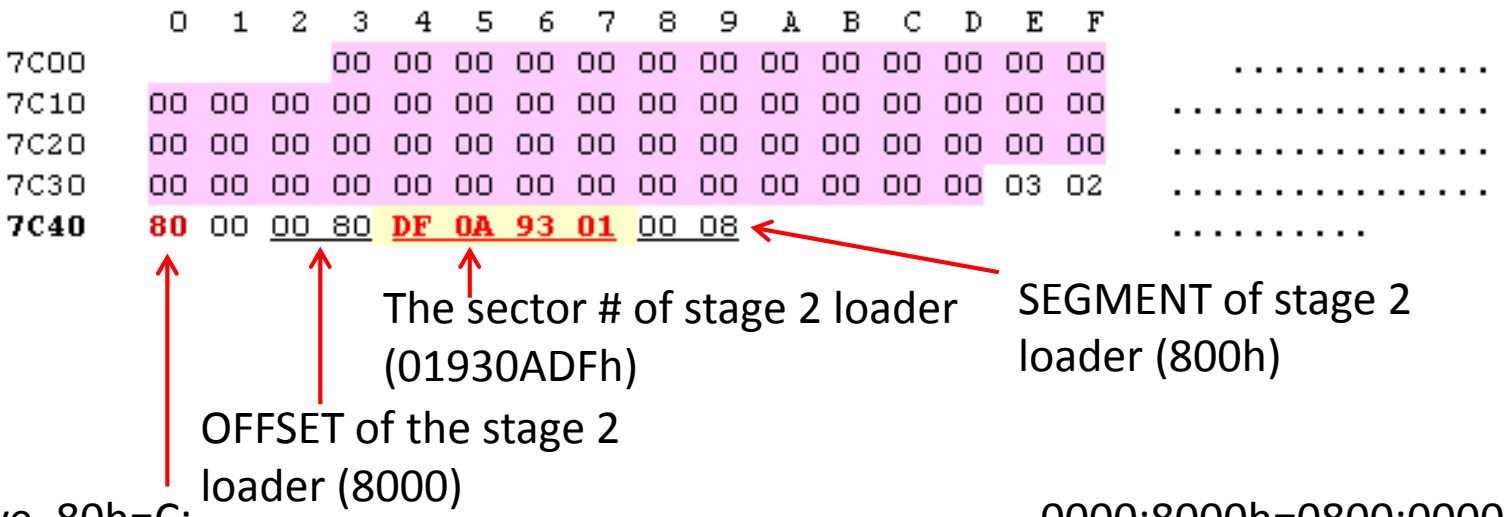
- A piece of code written in assembly
- Dump the MBR
  - dd if=/dev/sda of=mbr bs=512 count=1
- Disassembly MBR
  - objdump -D -b binary -mi386 -Maddr16,data16 mbr
  - Note that the source file is in NASM syntax
- You can also download GRUB and check the source code
  - grub-2.00\grub-core\boot\i386\pc\boot.S

GRUB MBR Loader

- The MBR loader changes dramatically in the recent revisions of GRUB, and we will focus on the GRUB 0.92/0.93 code in MASM syntax

GRUB MBR Loader

- The boot sector is loaded to 0000:7C00 by the BIOS loader
- 7C03~7C3D are BIOS Parameter Block (BPB), which is used by FAT MBR and NTFS MBR but not GRUB MBR
- 7C3E~7C49 are GRUB information



```
7C00 EB48          JMP     7C4A          ; Jump (short) over BPB data
7C02 90            NOP                   ; area to main body of code.
; 7C03~7C1E is BPB area and GRUB information
; ------------------ snipped ----------------
```

; Converting LBA in 7C44~7C47 to CHS

```
7D1F C0E206     *  SHL     DL,06         ; Drive #
7D22 8A4C0A        MOV     CL,[SI+0A]    ; Sector #
7D25 FEC1          INC     CL
7D27 08D1          OR      CL,DL
7D29 8A6C0C        MOV     CH,[SI+0C]    ; Track #
7D2C 5A            POP     DX
7D2D 8A740B        MOV     DH,[SI+0B]    ; Head #
7D30 BB0070        MOV     BX,7000
7D33 8EC3          MOV     ES,BX
7D35 31DB          XOR     BX,BX         ; ES:BX Buffer 7000:0000
7D37 B80102        MOV     AX,0201       ; Function 02 of INT13
7D3A CD13          INT     13            ; Read 1 sector into Memory
7D3C 722A          JB      7D68          ; There was a Read Error!
7D3E 8CC3          MOV     BX,ES
7D40 8E06487C      MOV     ES,[7C48]     ; <<<<<<<< WORD [0800 hex]
                                         ; Note: 800:0000 = 0000:8000

7D44 60         *  PUSHA
7D45 1E            PUSH    DS
7D46 B90001        MOV     CX,0100
7D49 8EDB          MOV     DS,BX         ; BX=7000
7D4B 31F6          XOR     SI,SI         ; DS:SI=7000:0000
7D4D 31FF          XOR     DI,DI         ; ES:DI=0800:0000
7D4F FC            CLD
7D50 F3A5          REP     MOVSW
7D52 1F            POP     DS
7D53 61         *  POPA
7D54 FF26427C      JMP     [7C42]        ; WORD    <<< 8000 hex.
                                         ; "stage2_address".
```

```
7D68 BE8B7D     MOV     SI,7D8B       ;
7D6B E83000     CALL    7D9E          ; Display "Read"
7D6E BE907D     MOV     SI,7D90       ;
7D71 E82A00     CALL    7D9E          ; Display "Error"
7D74 EBFE       JMP     7D74          ; Lock up the computer!
```

Standard (MSDOS) Partition Table

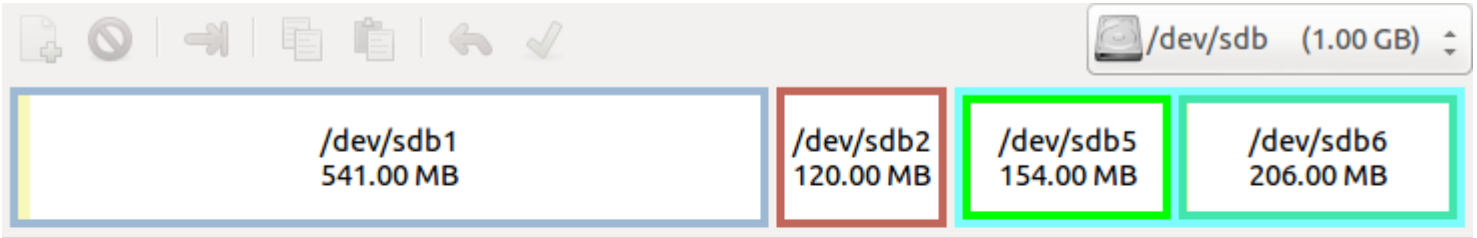
- A tiny 64 bytes table embedded in the boot sector
  - Each entry is of 16 bytes; so up to 4 entries
- Three types of partition: primary, logical, extended
  - A primary partition is a regular partition (active/bootable)
    - Up to 4 primary partitions (Linux); 1 primary partition (DOS)
  - An extended partition is a “container” of logical partitions
  - A logical partition is just like a primary partition, but it is not bootable
    - You can create as many logical partitions as your boot loader/OS support
    - The table entries for logical partitions are not in MBR
  - Example: 2 primary partition, 1 extended partition, in which there are two logical partitions

Partition Table Entry

- [Start CHS][End CHS][Start LBA][Partition ID][Size]
  - Partition ID: 82h Linux Swap, 83h Ext234, 0Eh FAT-16
  - Partition size up to 4G sectors; 2TB.
  - Your MBR and OS must support GUID Partition Table (GPT) to create partitions > 2TB.
    - Windows Vista and later versions of Windows do
    - Recent Linux distributions (2006-) support GPT

The *standard* 64-byte *Primary* Partition Table			
Offsets (within MBR sector)		Length (in *bytes*)	**Contents**
in Decimal	*in Hex*		
446 - 461	1BE - 1CD	16	Table Entry for Primary Partition **# 1**
462 - 477	1CE - 1DD	16	Table Entry for Primary Partition **# 2**
478 - 493	1DE - 1ED	16	Table Entry for Primary Partition **# 3**
494 - 509	1EE - 1FD	16	Table Entry for Primary Partition **# 4**

Structure of a *16-byte* Partition Table Entry		
Relative Offsets (*within entry*)	Length (*bytes*)	**Contents**
0	1	Boot Indicator (80h = *active*)
1 - 3	3	Starting CHS values
4	1	*Partition-type* Descriptor
5 - 7	3	Ending CHS values
8 - 11	4	Starting Sector
12 - 15	4	Partition Size (in sectors)



分割區	檔案系統	大小	已使用	未使用	旗標
/dev/sdb1	ext2	541.00 MB	8.93 MB	532.07 MB	
/dev/sdb2	linux-swap	120.00 MB	---	---	
▼ /dev/sdb3	extended	362.00 MB	---	---	
/dev/sdb5	fat16	154.00 MB	180.00 KB	153.82 MB	
/dev/sdb6	ntfs	206.00 MB	1.48 MB	204.52 MB	

```
80 20 21 00 83 18 13 45 00 08 00 00 00 E8 10 00
00 18 14 45 82 64 10 54 00 F0 10 00 00 C0 03 00
00 64 11 54 05 8A 08 82 00 B0 14 00 00 50 0B 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Bootable?

Starting CHS

Partition type
5=extended

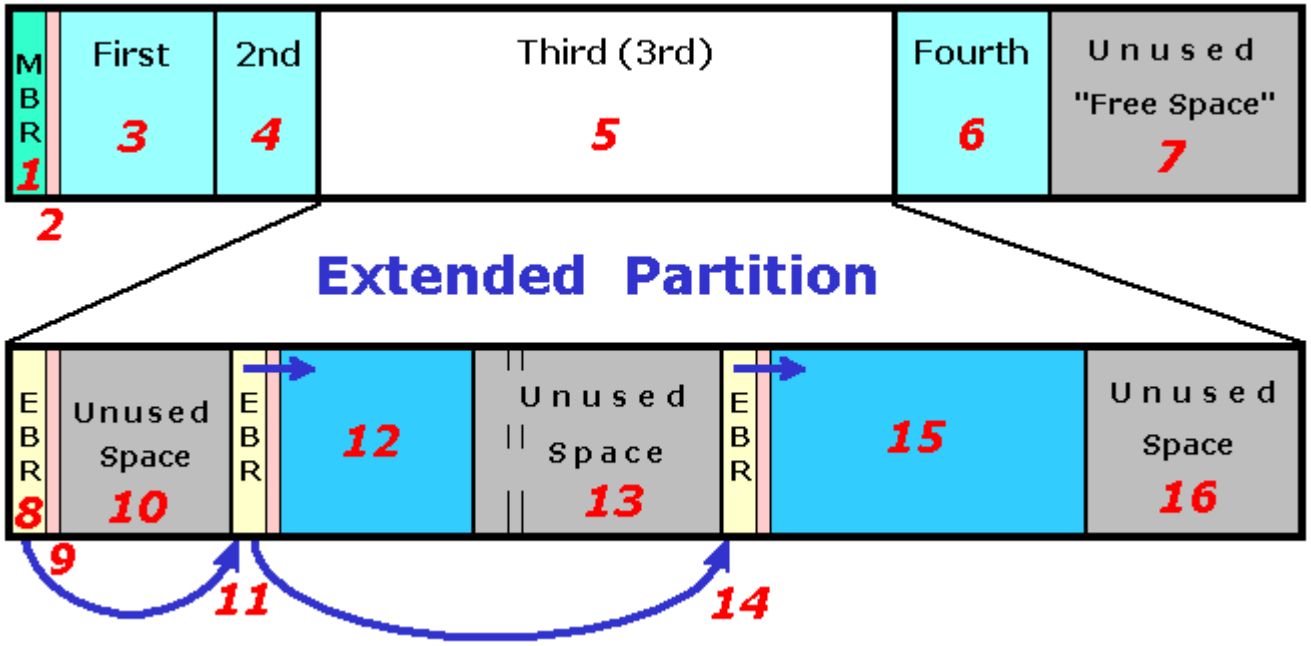
Ending CHS

Starting Sector #

Partition Size (sectors)

Extended Boot Record

- A logical partition is preceded by its EBR, and EBRs of logical partitions are chained in a list
- EBR's format is similar to MBR, but the code region is not used



For EBRs describing logical partitions in the extended partition, see http://thestarman.pcministry.com/asm/mbr/PartTables2.htm

Extended Boot Record

Structure of an ***Extended Boot Record*** (EBR) Sector				
Offsets (within sector)		**Description**		**Length (in *bytes*)**
in Decimal	***in Hex***			
000 - 445	000 - 1BD	**Normally Unused** [5]		**446**
446 - 509	1BE - 1FD	EBR Partition Table		**64**
510	1FE	**55**	**Boot Record Signature**	2
511	1FF	**AA**		

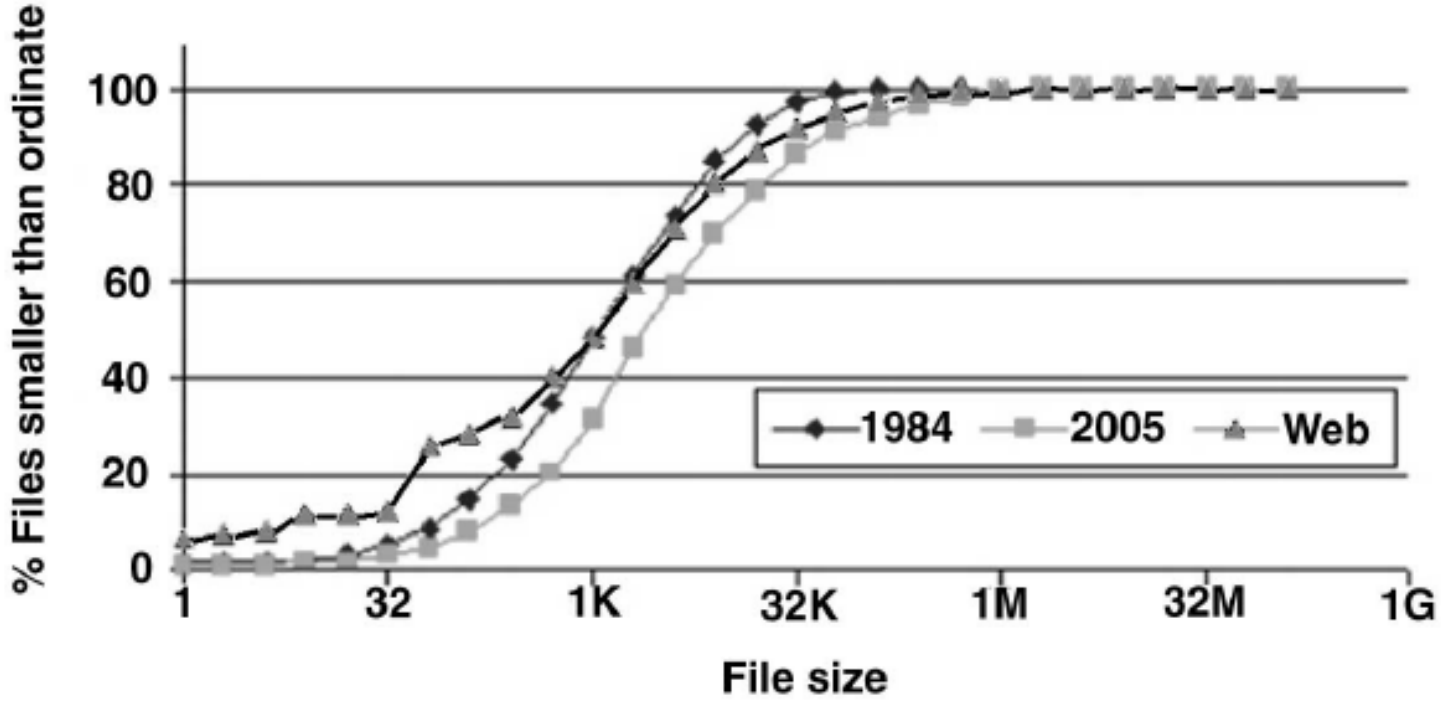
An ***EBR*** Partition Table			
Offsets (within sector)		**Length (in *bytes*)**	**Contents**
in Decimal	***in Hex***		
446 - 461	1BE - 1CD	16	**Table Entry for this Logical Partition**
462 - 477	1CE - 1DD	16	If present, **link** to **next Logical partition**
478 - 509	1DE - 1FD	32	**Normally Unused**

FAT File System

- FAT-12 developed for 3.5” or 5.25” floppy disks
- FAT-16 for hard drives
- FAT-32 for large hard drivers (>=1GB)
- The number postfix *x* means the disk space is divided into 2^x clusters
  - Large internal fragmentation when clusters are large
  - An FAT-16 cluster size is 16 KB in a 1 GB disk

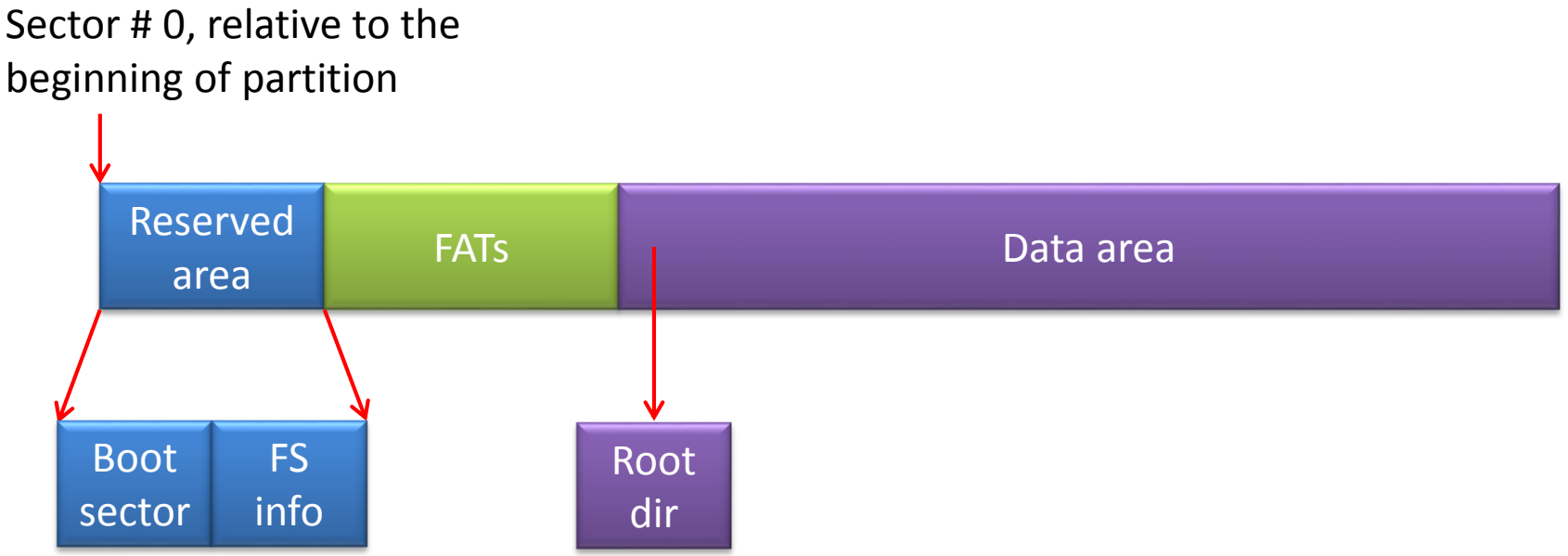
Attribute	FAT12	FAT16	FAT32
Size of Each FAT Entry	12 bits	16 bits	28 bits
Maximum Number of Clusters	4,086 (4,096 theoretical)	65,526	268,435,456
Cluster Size Used	0.5 KB to 4 KB	2 KB to 32 KB	4 KB to 32 KB
Maximum Volume Size	16,736,256	2,147,123,200 ≒2GB	about 2^41 (8 TB (with 32KB clusters))

UNIX File Size Distribution



Andrew S. Tanenbaum, Jorrit N. Herder, and Herbert Bos. 2006. File size distribution on UNIX systems: then and now. *SIGOPS Oper. Syst. Rev.* 40, 1 (January 2006), 100-104.

Disk Layout



- Reserved area
  - A boot sector
  - An optional file system information sector (FAT-32)
- FATs
  - File allocation tables. # of copies is specified in the boot sector
- Data area
  - A root directory and data clusters.
  - The root dir size is fixed in FAT-12/16, and is variable in FAT-32

Boot Sector

Offset	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	0123456789ABCDEF
000000000	EB	3C	90	6D	6B	64	6F	73	66	73	00	00	02	08	08	00	.<.mkdosfs.....
000000016	02	00	02	00	00	F8	A0	00	3F	00	FF	00	00	00	00	00	........?.....
000000032	00	D0	04	00	00	00	29	73	AC	67	4A	20	20	20	20	20	......)s.gJ
000000048	20	20	20	20	20	20	46	41	54	31	36	20	20	20	0E	1F	FAT16 ..
000000064	BE	5B	7C	AC	22	C0	74	0B	56	B4	0E	BB	07	00	CD	10	.[|.".t.V.....
000000080	5E	EB	F0	32	E4	CD	16	CD	19	EB	FE	54	68	69	73	20	^..2.....This
000000096	69	73	20	6E	6F	74	20	61	20	62	6F	6F	74	61	62	6C	is not a bootabl
000000112	65	20	64	69	73	6B	2E	20	20	50	6C	65	61	73	65	20	e disk. Please
000000128	69	6E	73	65	72	74	20	61	20	62	6F	6F	74	61	62	6C	insert a bootabl
000000144	65	20	66	6C	6F	70	70	79	20	61	6E	64	0D	0A	70	72	e floppy and..pr
000000160	65	73	73	20	61	6E	79	20	6B	65	79	20	74	6F	20	74	ess any key to t
000000176	72	79	20	61	67	61	69	6E	20	2E	2E	2E	20	0D	0A	00	ry again ... ...
000000192	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000208	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000224	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000240	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000256	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000272	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000288	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000304	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000320	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000336	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000352	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000368	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000384	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000400	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000416	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000432	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000448	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000464	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000480	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000496	00	00	00	00	00	00	00	00	00	00	00	00	00	00	55	AA	.....U.

```
Offset    00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 0123456789ABCDEF
000000000 EB 3C 90 6D 6B 64 6F 73 66 73 00 00 02 08 08 00 .<.mkdosfs......
000000016 02 00 02 00 00 F8 A0 00 3F 00 FF 00 00 00 00 00 ........?.......
000000032 00 D0 04 00 00 00 29 73 AC 67 4A 20 20 20 20 20 ......)s.gJ
```

Bytes	Purpose	
0-2	Assembly code instructions to jump to boot code (mandatory in bootable partition)	**JMP 003E, NOP**
3-10	OEM name in ASCII	**mkdosfs**
11-12	Bytes per sector (512, 1024, 2048, or 4096)	**0200=512**
13	Sectors per cluster (Must be a power of 2 and cluster size must be <=32 KB)	**08 (4 KB)**
14-15	Size of reserved area, in sectors	**0008**
16	Number of FATs (usually 2)	**2**
17-18	Maximum number of files in the root directory (FAT12/16; 0 for FAT32)	**0200=512**
19-20	Number of sectors in the file system; if 2 B is not large enough, set to 0 and use 4 B value in bytes 32-35 below	**0000**
21	Media type (0xf0=removable disk, 0xf8=fixed disk)	**F8**
22-23	Size of each FAT, in sectors, for FAT12/16; 0 for FAT32	**00A0=160**
24-25	Sectors per track in storage device	
26-27	Number of heads in storage device	
28-31	Number of sectors before the start partition	
32-35	Number of sectors in the file system; this field will be 0 if the 2B field above (bytes 19-20) is non-zero	**0004D000=315392**

```
000000032 00 D0 04 00 00 00 29 73 AC 67 4A 20 20 20 20 20 ......)s.gJ
000000048 20 20 20 20 20 20 46 41 54 31 36 20 20 20 0E 1F      FAT16   ..
000000064 BE 5B 7C AC 22 C0 74 0B 56 B4 0E BB 07 00 CD 10 .[|."t.V.......
000000080 5E EB F0 32 E4 CD 16 CD 19 EB FE 54 68 69 73 20 ^..2.......This
000000096 69 73 20 6E 6F 74 20 61 20 62 6F 6F 74 61 62 6C is not a bootabl
000000112 65 20 64 69 73 6B 2E 20 20 50 6C 65 61 73 65 20 e disk.  Please
000000128 69 6E 73 65 72 74 20 61 20 62 6F 6F 74 61 62 6C insert a bootabl
000000144 65 20 66 6C 6F 70 70 79 20 61 6E 64 0D 0A 70 72 e floppy and..pr
000000160 65 73 73 20 61 6E 79 20 6B 65 79 20 74 6F 20 74 ess any key to t
000000176 72 79 20 61 67 61 69 6E 20 2E 2E 2E 20 0D 0A 00 ry again ... ...
                              ⋮
000000496 00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 AA ..............U.
```

Bytes	Purpose
0-35	(See previous table)
36	BIOS INT 13h (low level disk services) drive number
37	Not used
38	Extended boot signature to validate next three fields (0x29)
39-42	Volume serial number
43-53	Volume label, in ASCII
54-61	File system type level, in ASCII. (Generally "FAT", "FAT12", or "FAT16")
62-509	Boot code (loaded by MBR loader)
510-511	Signature value (0xaa55)

00 (C:)

00

29

4A67AC73

An empty string

FAT16

Showing “insert a bootable…”

AA55

FAT Table

- The entire partition is divided into clusters, and each cluster has a parallel entry in the FAT table
- An FAT entry has one of the following meanings
  - the cluster number of the next cluster in a chain
  - a special *end of cluster-chain* (*EOC*) entry that indicates the end of a chain
  - a special entry to mark a bad cluster
  - a zero to note that the cluster is unused
- Only clusters allocated by files are in cluster chains

FAT Table

- Special FAT entry values
  - 0000h: free cluster
  - 0001h: reserved
  - 0002h~FFEFh: the next cluster # in chain
  - FFF0h~FFF6h: reserved
  - FFF7h: bad cluster
  - FFF8h~FFFFh: end of cluster chain

FAT Example

```
root@leslie-virtual-machine:/mnt# ls -l --sort=time -r
總計 600
drwxr-xr-x 2 root root   4096  5月  6 22:11 mydir1
drwxr-xr-x 2 root root   4096  5月  6 22:12 mydir2
-rwxr-xr-x 1 root root     15  5月  6 22:12 myfile1.txt
-rwxr-xr-x 1 root root 598050  5月  7 17:27 lde_261.tgz
```

- 2 FATs, starting at sector 8
- In the root directory, there are
  - 2 directories
  - 2 files; one small and one large
    - The large file requires 147 clusters
    - 1 cluster= 4 KB

The primary FAT

Offset	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	0123456789ABCDEF
000004096	F8	FF	FF	FF	00	00	FF	FF	FF	FF	FF	FF	00	00	08	00	.....
000004112	09	00	0A	00	0B	00	0C	00	0D	00	0E	00	0F	00	10	00	.....
000004128	11	00	12	00	13	00	14	00	15	00	16	00	17	00	18	00	.....
000004144	19	00	1A	00	1B	00	1C	00	1D	00	1E	00	1F	00	20	00	..... .
000004160	21	00	22	00	23	00	24	00	25	00	26	00	27	00	28	00	!.".#.$.%.&.'.(.
000004176	29	00	2A	00	2B	00	2C	00	2D	00	2E	00	2F	00	30	00	).*.+.,.-.../.0.
000004192	31	00	32	00	33	00	34	00	35	00	36	00	37	00	38	00	1.2.3.4.5.6.7.8.
000004208	39	00	3A	00	3B	00	3C	00	3D	00	3E	00	3F	00	40	00	9.:.;.<.=.>.?.@.
000004224	41	00	42	00	43	00	44	00	45	00	46	00	47	00	48	00	A.B.C.D.E.F.G.H.
000004240	49	00	4A	00	4B	00	4C	00	4D	00	4E	00	4F	00	50	00	I.J.K.L.M.N.O.P.
000004256	51	00	52	00	53	00	54	00	55	00	56	00	57	00	58	00	Q.R.S.T.U.V.W.X.
000004272	59	00	5A	00	5B	00	5C	00	5D	00	5E	00	5F	00	60	00	Y.Z.[.\.].^._.`.
000004288	61	00	62	00	63	00	64	00	65	00	66	00	67	00	68	00	a.b.c.d.e.f.g.h.
000004304	69	00	6A	00	6B	00	6C	00	6D	00	6E	00	6F	00	70	00	i.j.k.l.m.n.o.p.
000004320	71	00	72	00	73	00	74	00	75	00	76	00	77	00	78	00	q.r.s.t.u.v.w.x.
000004336	79	00	7A	00	7B	00	7C	00	7D	00	7E	00	7F	00	80	00	y.z.{.|.}.~.....
000004352	81	00	82	00	83	00	84	00	85	00	86	00	87	00	88	00	.....
000004368	89	00	8A	00	8B	00	8C	00	8D	00	8E	00	8F	00	90	00	.....
000004384	91	00	92	00	93	00	94	00	95	00	96	00	97	00	98	00	.....
000004400	99	00	FF	FF	00	00	00	00	00	00	00	00	00	00	00	00	.....
000004416	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000004432	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000004448	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000004464	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000004480	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000004496	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000004512	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000004528	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000004544	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000004560	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000004576	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000004592	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....

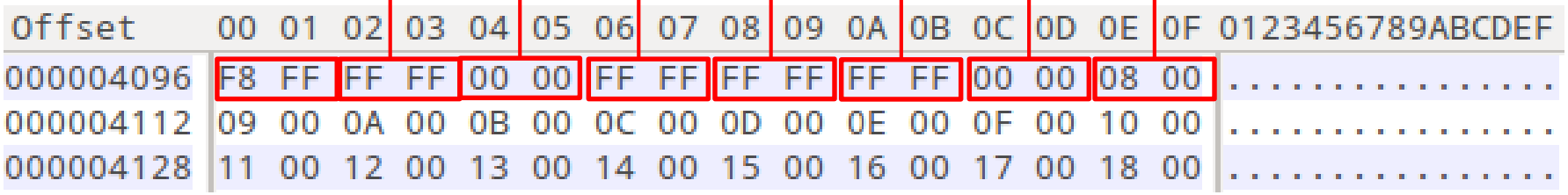
The first 2 entries are media descriptor and reserved fields

EOF. These are three independent chains but their length are just 1.

Beginning of the 4th cluster chain

A free cluster

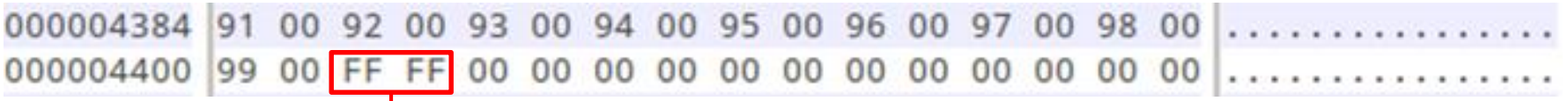
A free cluster



Offset	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	0123456789ABCDEF
000004096	F8	FF	FF	FF	00	00	FF	FF	FF	FF	FF	FF	00	00	08	00	.....
000004112	09	00	0A	00	0B	00	0C	00	0D	00	0E	00	0F	00	10	00	.....
000004128	11	00	12	00	13	00	14	00	15	00	16	00	17	00	18	00	.....

From 0E: A cluster chain of the big file lde_261.tgz

0008→0009→000A→000B→000C→000D0→000E→000F→0010→……



000004384	91	00	92	00	93	00	94	00	95	00	96	00	97	00	98	00	.....
000004400	99	00	FF	FF	00	00	00	00	00	00	00	00	00	00	00	00	.....

End of the 4th cluster chain

Root Directory

- The size and location of a FAT-12/16 root directory is statically specified in the boot sector
  - Location=reserved area size+2*fat size
  - Size=512 entries, 32 bytes each (32 sectors)
  - Non-root directories have not such restrictions
- Standard fat-16 file names are in the 8.3 format
  - vfat supports long file names by writing Long File Names (LFN) records

Root directory

Offset	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	0123456789ABCDEF
000167936	41	6D	00	79	00	64	00	69	00	72	00	0F	00	E6	32	00	Am.y.d.i.r....2.
000167952	00	00	FF	FF	FF	FF	FF	FF	FF	FF	00	00	FF	FF	FF	FF	.....
000167968	4D	59	44	49	52	32	20	20	20	20	20	10	00	00	90	B1	MYDIR2 .....
000167984	A6	42	A6	42	00	00	90	B1	A6	42	04	00	00	00	00	00	.B.B.....B.....
000168000	41	6D	00	79	00	64	00	69	00	72	00	0F	00	DE	31	00	Am.y.d.i.r....1.
000168016	00	00	FF	FF	FF	FF	FF	FF	FF	FF	00	00	FF	FF	FF	FF	.....
000168032	4D	59	44	49	52	31	20	20	20	20	20	10	00	64	6A	B1	MYDIR1 ..dj.
000168048	A6	42	A6	42	00	00	6A	B1	A6	42	03	00	00	00	00	00	.B.B..j..B.....
000168064	41	6D	00	79	00	66	00	69	00	6C	00	0F	00	8B	65	00	Am.y.f.i.l....e.
000168080	31	00	2E	00	74	00	78	00	74	00	00	00	00	00	FF	FF	1...t.x.t.....
000168096	4D	59	46	49	4C	45	31	20	54	58	54	20	00	64	99	B1	MYFILE1 TXT .d..
000168112	A6	42	A6	42	00	00	99	B1	A6	42	05	00	0F	00	00	00	.B.B.....B.....
000168128	E5	6D	00	79	00	66	00	69	00	6C	00	0F	00	5B	65	00	.m.y.f.i.l...[e.
000168144	32	00	2E	00	74	00	78	00	74	00	00	00	00	00	FF	FF	2...t.x.t.....
000168160	E5	59	46	49	4C	45	32	20	54	58	54	20	00	64	77	8B	.YFILE2 TXT .dw.
000168176	A7	42	A6	42	00	00	77	8B	A7	42	07	00	22	20	09	00	.B.B..w..B.." ..
000168192	41	6C	00	64	00	65	00	5F	00	32	00	0F	00	5D	36	00	Al.d.e._.2...]6.
000168208	31	00	2E	00	74	00	67	00	7A	00	00	00	00	00	FF	FF	1...t.g.z.....
000168224	4C	44	45	5F	32	36	31	20	54	47	5A	20	00	64	77	8B	LDE_261 TGZ .dw.
000168240	A7	42	A6	42	00	00	77	8B	A7	42	07	00	22	20	09	00	.B.B..w..B.." ..
000168256	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000168272	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000168288	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000168304	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000168320	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000168336	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000168352	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000168368	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000168384	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000168400	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000168416	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000168432	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....

A deleted file

```
000168160 E5 59 46 49 4C 45 32 20 54 58 54 20 00 64 77 8B .YFILE2 TXT .dw.
000168176 A7 42 A6 42 00 00 77 8B A7 42 07 00 22 20 09 00 .B.B..w..B.." ..
```

A sub directory

```
000168032 4D 59 44 49 52 31 20 20 20 20 20 10 00 64 6A B1 MYDIR1     ..dj.
000168048 A6 42 A6 42 00 00 6A B1 A6 42 03 00 00 00 00 00 .B.B..j..B......
```

Long file name (LFN) record for “MYDIR1”

```
000168000 41 6D 00 79 00 64 00 69 00 72 00 0F 00 DE 31 00 Am.y.d.i.r....1.
000168016 00 00 FF FF FF FF FF FF FF FF 00 00 FF FF FF FF ................
```

A regular file “LDE_261.TGZ”

```
000168224 4C 44 45 5F 32 36 31 20 54 47 5A 20 00 64 77 8B LDE_261 TGZ .dw.
000168240 A7 42 A6 42 00 00 77 8B A7 42 07 00 22 20 09 00 .B.B..w..B.." ..
```

Bytes	Purpose
0	First character of file name (ASCII) *or* allocation status (0x00=unallocated, 0xe5=deleted)
1-10	Characters 2-11 of the file name (ASCII); the "." is implied between bytes 7 and 8
11	File attributes (see File Attributes table)
12	Reserved
13	File creation time (in tenths of seconds)*
14-15	Creation time (hours, minutes, seconds)*
16-17	Creation date*
18-19	Access date*
20-21	High-order 2 bytes of address of first cluster (0 for FAT12/16)*
22-23	Modified time (hours, minutes, seconds)
24-25	Modified date
26-27	Low-order 2 bytes of address of first cluster
28-31	File size (0 for directories)

4C, allocated

LDE_261 TGZ

0000

0007

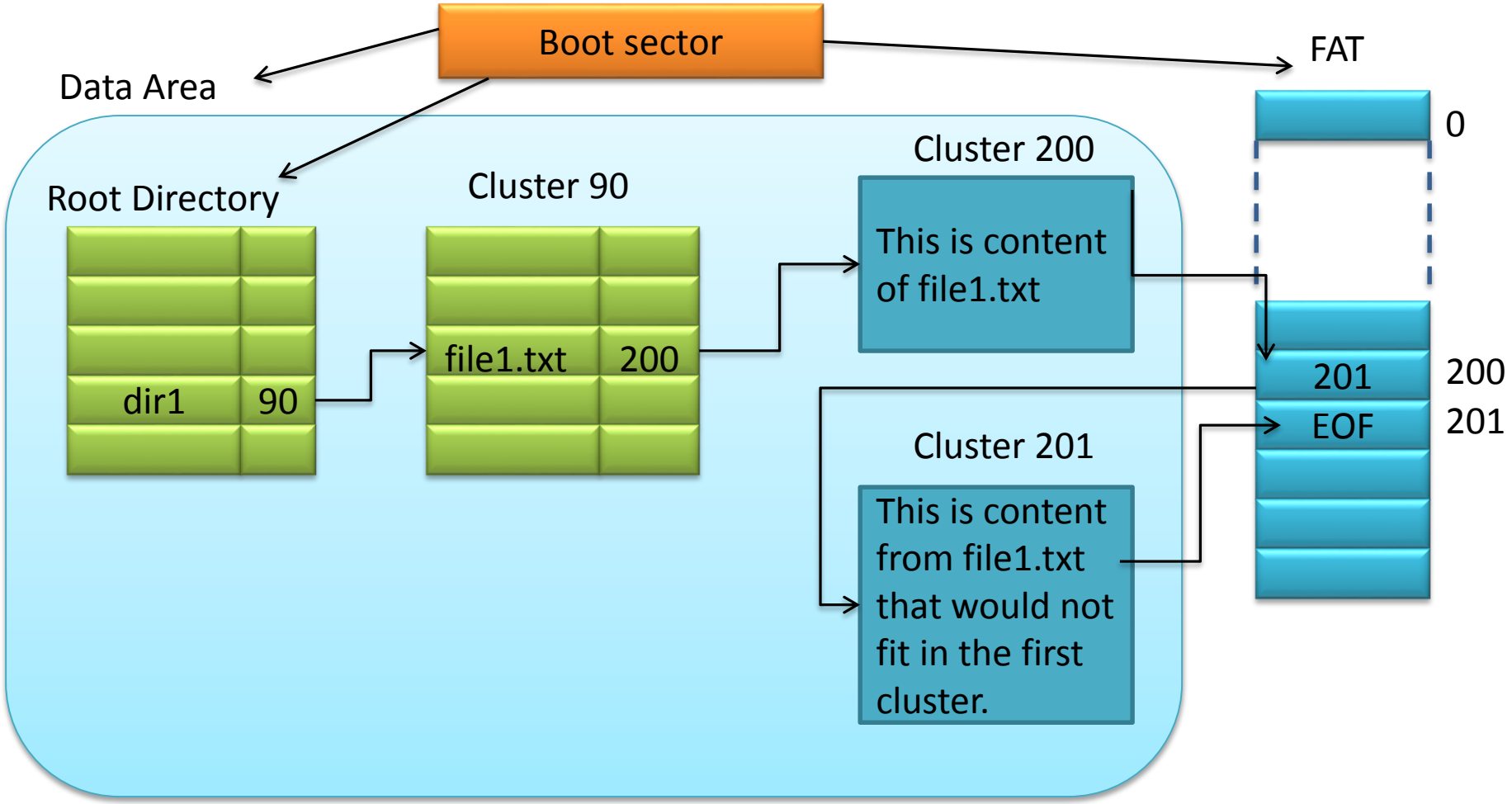
00092022=598050

Directory Entry Allocation

- Windows 98: first free
- Windows XP: next free
  - Better for un-deleting files

Cluster size: 4 ,096 bytes
file size: 6,000 bytes

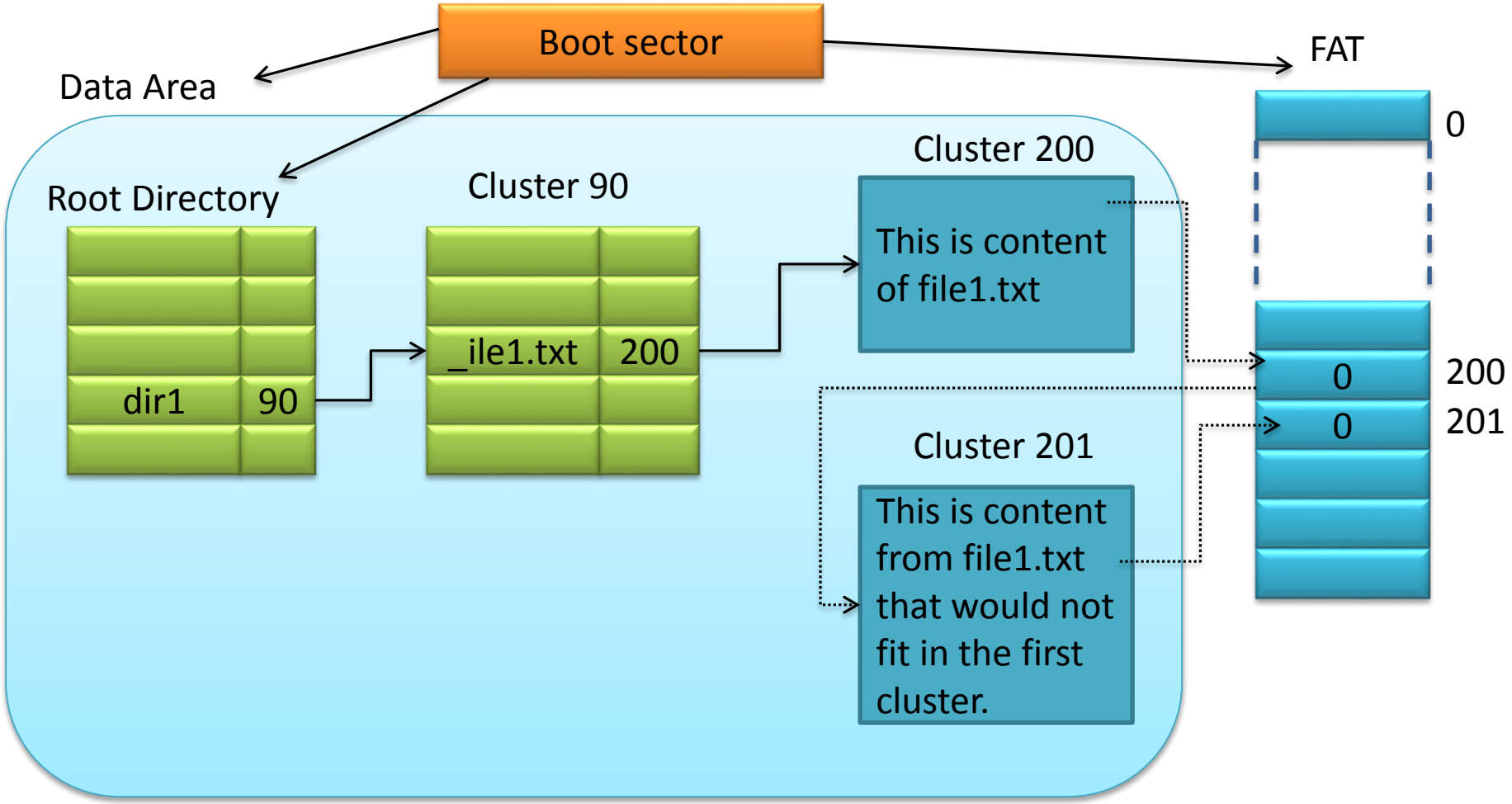
File Allocation Example: create the *dir1\file1.txt file*



Credit: Chen-Yi Wen

Cluster size: 4 ,096 bytes
file size: 6,000 bytes

File Allocation Example: delete the *dir1\file1.txt file*



Lab 10: FAT Disk Editing

FAT Disk Editing

- Use a disk editor to manually manipulate FAT disk structures
- Part 1: installing a disk editor
- Part 2: creating a file using the disk editor
- Part 3: un-deleting a deleted file using the disk editor

References

- Brian Carrier, ”File System Forensic Analysis,” 2005
- The Starman, “The GRUB MBR,” 2009
  - http://thestarman.pcministry.com/asm/mbr/GRUB.htm
- Allan Gottlieb, “The FAT File System”
  - http://cs.nyu.edu/~gottlieb/courses/os/kholodov-fat.html#F01_0010_overview
- Daniel B. Sedory, “MBR/EBR Partition Tables,” 2013
  - http://thestarman.pcministry.com/asm/mbr/PartTables.htm
- Andrew S. Tanenbaum, Jorrit N. Herder, and Herbert Bos. 2006. File size distribution on UNIX systems: then and now. *SIGOPS Oper. Syst. Rev.* 40, 1 (January 2006), 100-104.