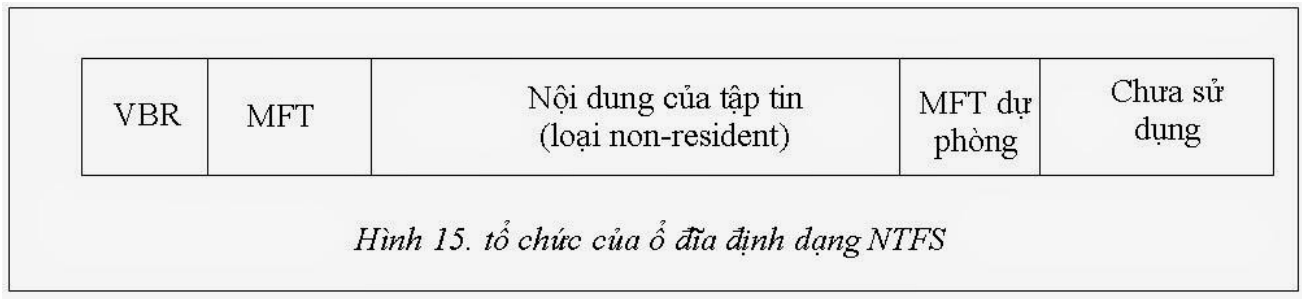
GIẢI ĐÁP THẮC MẮC PROJECT 1

Phùng Anh Khoa: *thầy ơi, 1 entry chính có thể có nhiều entry phụ hay chỉ 1 entry phụ*

Trả lời: 1 tập tin có 1 entry chính, nếu tên tập tin vượt quá 11 ký tự (kể cả phần mở rộng) sẽ xuất hiện thêm entry phụ (entry phụ chỉ dùng để lưu tên của tập tin). Mỗi entry phụ sẽ lưu được 26ký tự ascii hoặc 13 ký tự unicode (Tên càng dài càng tốn nhiều entry phụ)

Ngô Trường Tuyển : *thầy cho em hỏi là ở NTFS thì nó quy định là 16 sector đầu là partition boot sector, nên vị trí của MFT là 16 phải không thầy?*

Trả lời: Tổ chức của ổ đĩa logic định dạng NTFS được minh hoạn như sau



Hình 15. tổ chức của ổ đĩa định dạng NTFS

VBR (Volume Boot Record hay còn gọi là NTFS Partition Boot Sector) là bản ghi khởi động của ổ đĩa logic, nó luôn nằm ở vị trí đầu tiên của mỗi ổ đĩa logic. VBR chứa: mã khởi động, BPB, thông báo lỗi, và một số thông tin khác.

Trong đó BPB (Bios Parameter Block) chứa các thông tin quan trọng của phân vùng gồm 73 byte có cấu trúc như sau:

Địa chỉ (offset)	Kích thước (byte)	Mô tả
0Bh	2	Kích thước một sector. Đơn vị tính là byte.
0Dh	1	Số sector trong một cluster.
0Eh	2	Chưa sử dụng.
10h	1	Với hệ thống NTFS luôn mang giá trị 0.
11h	2	Với hệ thống NTFS luôn mang giá trị 0.
13h	2	Luôn mang giá trị 0, hệ thống NTFS không sử dụng tới trường này.
15h	1	Mã xác định loại đĩa.
16h	2	Với hệ thống NTFS luôn mang giá trị 0.
18h	2	Số sector/track.
1Ah	2	Số mặt đĩa (head hay side).
1Ch	4	Sector bắt đầu của ổ đĩa logic.
20h	4	Luôn mang giá trị 0, hệ thống NTFS không sử dụng tới trường này.
24h	4	Hệ thống NTFS luôn thiết lập giá trị này là “80008000”.
28h	8	Số sector của ổ đĩa logic.
30h	8	Cluster bắt đầu của MFT.
38h	8	Cluster bắt đầu của MFT dự phòng (MFTMirror).
40h	1	Kích thước của một bản ghi trong MFT (MFT entry), đơn vị tính là byte.
41h	3	Luôn mang giá trị 0, hệ thống NTFS không sử dụng tới trường này.

44h	1	Số cluster của Index Buffer.
45h	3	Luôn mang giá trị 0, hệ thống NTFS không sử dụng tới trường này.
48h	8	Số seri của ổ đĩa (volume serial number).
50h	4	Không được sử dụng bởi NTFS.

Như vậy dựa trên nội dung của BPB ta có thể xác định được vị trí cluster bắt đầu của MTF (Master File Table)

Nguyễn Đức Huy: 1 bytes có 8 bit sao mình đọc được ký tự Unicode trong file txt vậy thầy

Trả lời: Ký tự unicode được lưu trữ 2 byte do đó đọc 2 byte → đổi ra giá trị hệ 10 và tra bảng mã unicode để xác định ký tự là gì.

Trần Anh Túc: Thầy cho em hỏi kích thước RDET(FAT32) tính sao vậy thầy?

Trả lời: Trong FAT12/FAT16 RDET có kích thước cố định là 32 sector và lưu ở trước vùng DATA. Tuy nhiên, với FAT32 RDETcó kích thước không cố định và có thể được lưu bất kỳ vị trí nào trên vùng DATA. Do đó để xác định được các cluster của RDET ta sử dụng bảng FAT và dò ra được các cluster của RDET. Từ đó đọc dữ liệu các entry để xác định như bình thường.

9127535 : chỉ số cluster lưu trữ trên đĩa cứng (của tập tin / thư mục) là chỉ số cluster đầu tiên của cái tập tin / thư mục đó à thầy?

Trả lời: Trên hệ thống tập tin FAT, các cluster dữ liệu của một tập tin / thư mục được lưu ở dạng danh sách liên kết, do đó trên RDET chỉ lưu lại chỉ số cluster bắt đầu của tập tin / thư mục thôi, để truy cập tất cả các cluster của tập tin / thư mục ta dùng bảng FAT để dò ra các cluster còn lại.
Ví dụ: tập tin có chỉ số cluster bắt đầu là 2, ta tra bảng FAT phần tử thứ 2 để lấy giá trị (giá trị đó là chỉ số cluster kế tiếp của cluster 2) từ đó tiếp tục tra các phần tử tương ứng để ra cluster kế tiếp nữa … và cứ như thế cho đến khi gặp giá trị kết thúc EOF (FFFFFFFF)

19127535 : trong yêu cầu đề yêu cầu sector, vậy chuyển cluster thành sector à thầy?

Trả lời: Đúng rồi. Từ các cluster tìm được của tập tin / thư mục. Dựa trên giá trị "Số sector trên cluster" đọc được từ Boot Sector, và chỉ số sector bắt đầu của vùng Data ta có thể suy ra được chỉ số sector tương ứng của các cluster tìm được.

Nguyễn Hoàng Thông : Thầy cho em hỏi khi đọc phần BPB của NTFS, em xác định cluster bắt đầu của bảng MFT, vậy từ cluster bắt đầu ấy chuyển sang sector vật lý bắt đầu của bảng MFT làm ntn vậy ạ?

Trả lời: Giả sử Từ bảng thông tin của BPB trong VBR, xét 8 byte tại offset 30h, giá trị của 8 byte này cho biết MFT bắt đầu tại cluster thứ X
Ta đổi sang chỉ số sector như sau : (lưu ý, số sector của một cluster là Sc được lưu trong BPB)
Sector = X * Sc

Q: Cách đọc nội dung của BPB trong NTFS

Sau đây là phần minh họa các bước để đọc được nội dung BPB trên đĩa cứng thật.

- Sử dụng phần mềm Disk Editor (Trong project các bạn sử dụng code để đọc sector lên nhé), đọc MBR của đĩa cứng (sector 0). Định vị vùng partition table, bắt đầu tại địa chỉ (offset) 0x01BE, xem Hình 17.

```
Physical Sector: Absolute Sector 0
...
...
000001A0: 67 20 6F 70 65 72 61 74  - 69 6E 67 20 73 79 73 74   g operating syst
000001B0: 65 6D 00 00 00 63 7B 9A  - 3D 74 76 08 00 00 80 01   em...c{.=tv.....
000001C0: 01 00 07 FE FF FF 3F 00  - 00 00 01 3A 83 02 00 00   ......?....:....
000001D0: C1 FF 0F FE FF FF 40 3A  - 83 02 BD C5 7C 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.
```

Hình 17. entry mô tả thông tin cho ổ đĩa C:\ (phần tô vàng)

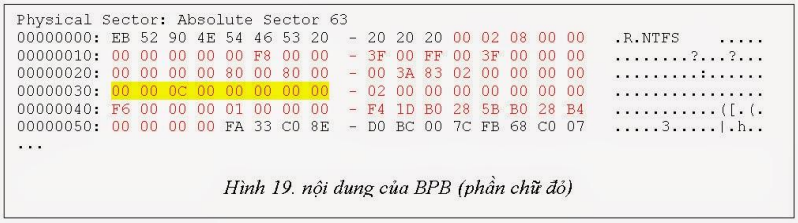
- Từ entry mô tả thông tin cho ổ đĩa C:\, xác định được địa chỉ bắt đầu của ổ đĩa C:\ là “3F 00 00 00” = 63.

- Đọc sector 63, đây là vị trí bắt đầu vùng VBR của ổ đĩa C:\. Kết quả được minh họa trong Hình 18.

```
Physical Sector: Absolute Sector 63
00000000: EB 52 90 4E 54 46 53 20  - 20 20 20 00 02 08 00 00   .R.NTFS    .....
00000010: 00 00 00 00 00 F8 00 00  - 3F 00 FF 00 3F 00 00 00   ........?...?...
00000020: 00 00 00 00 80 00 80 00  - 00 3A 83 02 00 00 00 00   .........:......
00000030: 00 00 0C 00 00 00 00 00  - 02 00 00 00 00 00 00 00   ................
00000040: F6 00 00 00 01 00 00 00  - F4 1D B0 28 5B B0 28 B4   ...........([.(.
00000050: 00 00 00 00 FA 33 C0 8E  - D0 BC 00 7C FB 68 C0 07   .....3.....|.h..
00000060: 1F 1E 68 66 00 CB 88 16  - 0E 00 66 81 3E 03 00 4E   .hf......f.>..N
00000070: 54 46 53 75 15 B4 41 BB  - AA 55 CD 13 72 0C 81 FB   TFSu..A..U..r...
00000080: 55 AA 75 06 F7 C1 01 00  - 75 03 E9 DD 00 1E 83 EC   U.u.....u.......
00000090: 18 68 1A 00 B4 48 8A 16  - 0E 00 8B F4 16 1F CD 13   .h...H..........
000000A0: 9F 83 C4 18 9E 58 1F 72  - E1 3B 06 0B 00 75 DB A3   .....Xr.;...u..
000000B0: 0F 00 C1 2E 0F 00 04 1E  - 5A 33 DB B9 00 20 2B C8   ........Z3... +.
000000C0: 66 FF 06 11 00 03 16 0F  - 00 8E C2 FF 06 16 00 E8   f...............
000000D0: 4B 00 2B C8 77 EF B8 00  - BB CD 1A 66 23 C0 75 2D   K.+.w......f#.u-
000000E0: 66 81 FB 54 43 50 41 75  - 24 81 F9 02 01 72 1E 16   f..TCPAu$....r..
000000F0: 68 07 BB 16 68 70 0E 16  - 68 09 00 66 53 66 53 66   h...hp..h..fSfSf
00000100: 55 16 16 16 68 B8 01 66  - 61 0E 07 CD 1A 33 C0 BF   U...h..fa....3..
00000110: 28 10 B9 D8 0F FC F3 AA  - E9 5F 01 90 90 66 60 1E   (........_...f`.
00000120: 06 66 A1 11 00 66 03 06  - 1C 00 1E 66 68 00 00 00   .f...f.....fh...
00000130: 00 66 50 06 53 68 01 00  - 68 10 00 B4 42 8A 16 0E   .fP.Sh..h...B...
00000140: 00 16 1F 8B F4 CD 13 66  - 59 5B 5A 66 59 66 59 1F   .......fY[ZfYfY
00000150: 0F 82 16 00 66 FF 06 11  - 00 03 16 0F 00 8E C2 FF   ....f...........
00000160: 0E 16 00 75 BC 07 1F 66  - 61 C3 A0 F8 01 E8 09 00   ...u...fa.......
00000170: A0 FB 01 E8 03 00 F4 EB  - FD B4 01 8B F0 AC 3C 00   ..............<.
00000180: 74 09 B4 0E BB 07 00 CD  - 10 EB F2 C3 0D 0A 41 20   t.............A
00000190: 64 69 73 6B 20 72 65 61  - 64 20 65 72 72 6F 72 20   disk read error
000001A0: 6F 63 63 75 72 72 65 64  - 00 0D 0A 42 4F 4F 54 4D   occurred...BOOTM
000001B0: 47 52 20 69 73 20 6D 69  - 73 73 69 6E 67 00 0D 0A   GR is missing...
000001C0: 42 4F 4F 54 4D 47 52 20  - 69 73 20 63 6F 6D 70 72   BOOTMGR is compr
000001D0: 65 73 73 65 64 00 0D 0A  - 50 72 65 73 73 20 43 74   essed...Press Ct
000001E0: 72 6C 2B 41 6C 74 2B 44  - 65 6C 20 74 6F 20 72 65   rl+Alt+Del to re
000001F0: 73 74 61 72 74 0D 0A 00  - 8C A9 BE D6 00 00 55 AA   start.........U.
```

Hình 18. nội dung VBR của ổ đĩa C:

- Từ VBR xác định vùng BPB, bắt đầu tại địa chỉ (offset) B đến 53, gồm 73 byte, xem Hình 19.



Hình 19. nội dung của BPB (phần chữ đỏ)

- Giá trị cụ thể của các trường trong BPB được minh họa trong bảng sau.

Địa chỉ (offset)	Kích thước (byte)	Giá trị	Giá trị hệ 10	Mô tả
0Bh	2	0x0200	512	Kích thước một sector là 512 byte.
0Dh	1	0x08	8	Số sector trong một cluster là 8, vậy kích thước một cluster là: 8 x 512 = 4096 B = 4 KB.
0Eh	2	“0000”	0	Chưa sử dụng.
10h	1	“00”	0	Với hệ thống NTFS, luôn mang giá trị 0.
11h	2	“0000”	0	Với hệ thống NTFS luôn mang giá trị 0.
13h	2	“0000”	0	Luôn mang giá trị 0, hệ thống NTFS không sử dụng tới trường này.
15h	1	0xF8	248	Mã xác định loại đĩa. F8 = Fixed Disk.
16h	2	“0000”	0	Với hệ thống NTFS, luôn mang giá trị 0.
18h	2	0x003F	63	Số sector/track là 63.
1Ah	2	0x00FF	255	Số mặt đĩa (head hay side) là 255.
1Ch	4	0x00003F	63	Sector bắt đầu của ổ đĩa logic C:\ là 63.
20h	4	“00000000”	0	Luôn mang giá trị 0, hệ thống NTFS không sử dụng tới trường này.
24h	4	“80008000”		Hệ thống NTFS luôn thiết lập giá trị này là “80008000”.
28h	8	0x0000 0000 0283 3A00	42 154 496	Số sector của ổ đĩa
30h	8	0x0000 0000 000C 0000	786 432	Cluster bắt đầu của MFT.
38h	8	0x0000 0000 0000 0002	2	Cluster bắt đầu của MFTMirror (MFT dự phòng)
40h	1 (số có dấu)	0xF6	-10	Kích thước của một bản ghi MFT (MFT entry). Đơn vị tính là byte.
0xF6 là một số có dấu, 0xF6 = 1111 0110 (dạng bù 2), tính ra được giá trị hệ thập phân = -10. Kích thước của một bản ghi MFT tính bằng 2 ^ |-10| = 1024 byte.				
41h	3	“000000”		Luôn mang giá trị 0, hệ thống NTFS không sử dụng tới trường này.
44h	1 (byte có dấu)	0x01	1	Số cluster của Index Buffer.
45h	3	“000000”		Luôn mang giá trị 0, hệ thống NTFS không sử dụng tới trường

				này.
48h	8	0xB428 B05B 28B0 1DF4		Số seri của ổ đĩa (volume serial number). Khi sử dụng lệnh dir trong cửa sổ dòng lệnh, hệ thống chỉ hiển thị giá trị của 4 byte cuối. Ví dụ: *C:\> dir* *Volume in drive C has no lable.* *Volume Serial Number is 28B0-1DF4*
50h	4	"0000 0000"	0	Không được sử dụng bởi NTFS.

Ngô Trường Tuyển : làm sao xác định kết thúc của MFT ạ

Trả lời: MFT được chia nhỏ thành các phần bằng nhau gọi là MFT entry. Kích thước của một MFT entry được quy định trong BPB, thường là 1024 byte.

MFT bản chất là một tập tin, do vậy cũng có một MFT entry mô tả cho chính nó, đó chính là MFT entry đầu tiên trong MFT, có tên là $MFT. $MFT mô tả về kích thước và tổ chức của MFT.

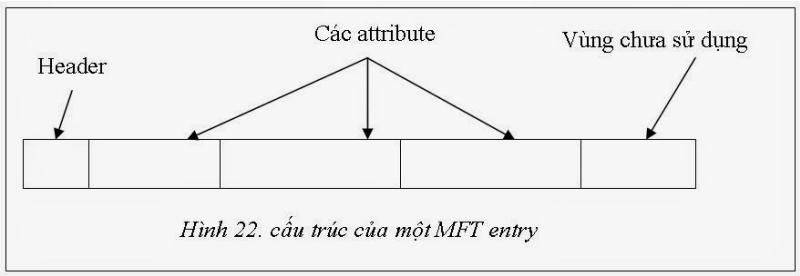
Cấu trúc của MFT entry

MFT entry gồm hai thành phần: header và các attribute.

Header gồm 42 byte đầu tiên được sử dụng để chứa một số thông tin mô tả cho MFT entry.

Phần còn lại của MFT entry được sử dụng để chứa các attribute. Nếu các attribute không sử dụng hết 1024 byte, hệ thống sẽ sử dụng giá trị 0xffffffff để đánh dấu kết thúc.

Hình 22 dưới đây minh họa tổ chức của một MFT entry, gồm header và ba attribute.



Hình 22. cấu trúc của một MFT entry

Các trường cụ thể của header được thể hiện trong bảng sau.

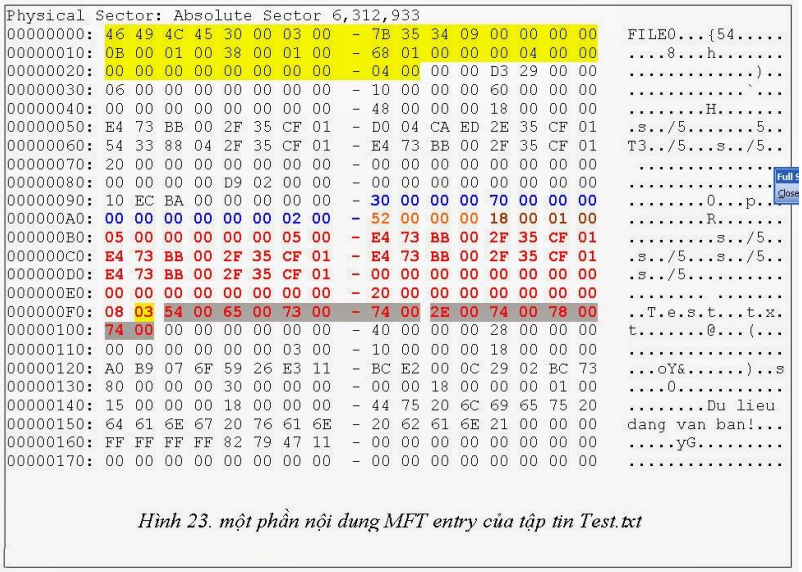
Offset	Số byte	Mô tả
0x0 – 0x03	4	Dấu hiệu nhận biết MFT entry.
0x04 – 0x05	2	Địa chỉ (offset) của Update sequence.
0x06 – 0x07	2	Số phần tử của mảng Fixup, mảng này chứa các giá trị bị thay thế trong quá trình thao tác với Update sequence.
0x08 – 0x0F	8	$LogFile Sequence Number (LSN): mã định danh

		MFT entry của file log (log record).
0x10 – 0x11	2	Sequence Number: cho biết số lần MFT entry này đã được sử dụng lại. Giá trị này được tăng lên một đơn vị sau mỗi lần tập tin tương ứng với MFT entry này bị xóa. Mang giá trị 0 nếu MFT entry này chưa được sử dụng.
0x12 – 0x13	2	Reference Count: cho biết số thư mục mà tập tin này được hiển thị trong đó, hay nói cách khác là số thư mục tham chiếu đến tập tin này. Trường này còn có tên gọi khác là hard link count.
0x14 – 0x15	2	Địa chỉ (offset) bắt đầu của các attribute.
0x16 – 0x17	2	Flags: - giá trị 0x01: MFT entry đã được sử dụng - giá trị 0x02: MFT entry của một thư mục - giá trị 0x04, 0x08: không xác định
0x18 – 0x1B	4	Số byte đã được sử dụng trong MFT entry.
0x1C – 0x1F	4	Kích thước vùng đĩa đã được cấp cho MFT entry.
0x20 – 0x27	8	Tham chiếu đến MFT entry cơ sở của nó (Base MFT Record). Mang giá trị 0 nếu là MFT entry cơ sở. MFT entry cơ sở dùng để chứa các thông tin về các MFT entry mở rộng (Extension Record).
0x28 – 0x29	2	Next attribute ID: mã định danh của attribute kế tiếp sẽ được thêm vào MFT entry.

Ví dụ

Tập tin được tạo trong tình huống ví dụ là Test.txt, như vậy sẽ có một MFT entry cho tập tin Test.txt được tạo ra trong MFT. Để xác định MFT entry này, cách đơn giản nhất là duyệt qua các khối 1024 byte (đây là kích thước của một MFT entry) bằng mắt thường, tại mỗi MFT entry quan sát giá trị tại offset 0x00F2, nếu thấy chuỗi Test.txt thì đó chính là MFT entry cho tập tin Test.txt. Ngoài ra, có thể sử dụng chức năng tìm kiếm của công cụ Disk Editor.

Hình 23 dưới đây là một phần nội dung MFT entry của tập tin Test.txt, nằm tại sector vật lý 6 321 933.



Hình 23. một phần nội dung MFT entry của tập tin Test.txt

Header của MFT entry này gồm 42 byte đầu tiên, phần tô vàng ở Hình 23. Giá trị của các trường được minh họa ở bảng sau.

Offset	Số byte	Giá trị	Ý nghĩa
0x0 – 0x03	4	“FILE”	Dấu hiệu nhận biết MFT entry là “FILE”, nếu MFT entry bị lỗi giá trị của trường này sẽ là “BAAD”.
0x04 – 0x05	2	0x0030	Địa chỉ (offset) của Update sequence.
0x06 – 0x07	2	0x0003	Số phần tử của mảng Fixup, mảng này chứa các giá trị bị thay thế trong quá trình thao tác với Update sequence
0x08 – 0x0F	8	0x0000 0000 0934 357B	$LogFile Sequence Number (LSN): mã định danh MFT entry của file log (log record).
0x10 – 0x11	2	0x000B	Sequence Number: cho biết số lần MFT entry này đã được sử dụng lại. Giá trị này được tăng lên một đơn vị sau mỗi lần tập tin tương ứng với MFT entry này bị xóa. Mang giá trị 0 nếu MFT entry này chưa được sử dụng. MFT entry này đã được sử dụng 11 lần.
0x12 – 0x13	2	0x0001	Reference Count: cho biết số thư mục mà tập tin này được hiển thị trong đó, hay nói cách khác là số thư mục tham chiếu đến tập tin này. Trường này còn có tên gọi khác là hard link count.
0x14 – 0x15	2	0x0038	Địa chỉ (offset) bắt đầu của attribute đầu tiên, trong MFT entry này là byte thứ 56.
0x16 – 0x17	2	0x0001	Flags: - giá trị 0x01: MFT entry đã được sử dụng
0x18 – 0x1B	4	0x0000 0168	Số byte trong MFT entry đã được sử dụng. Ví dụ, trong trường hợp này đã sử dụng 0x0168 = 360 byte.
0x1C – 0x1F	4	0x0000 0400	Kích thước vùng đĩa đã được cấp cho MFT entry, Ví dụ: 0x0400 = 1024 byte.
0x20 – 0x27	8	0x0000 0000 0000 0000	Tham chiếu đến MFT entry cơ sở của nó (Base MFT Record). Mang giá trị 0 nếu là MFT entry cơ sở. MFT entry cơ sở dùng để chứa các thông tin về các MFT entry mở rộng (Extension Record).
0x28 – 0x29	2	0x0004	Next attribute ID: mã định danh của attribute kế tiếp sẽ được thêm vào MFT entry.

Attribute

Attribute là một cấu trúc dữ liệu, được sử dụng để chứa nội dung của tập tin, chứa các thông tin liên quan đến tập tin, thư mục,…v.v trong hệ thống NTFS.

Có nhiều loại attribute, mỗi loại có cấu trúc tổ chức riêng, có một mã loại (type ID) riêng. Mã loại là một số nguyên. Microsoft sắp xếp thứ tự các attribute trong mỗi MFT entry theo chiều tăng dần của mã loại, nghĩa là, attribute nào có mã loại nhỏ sẽ đứng trước, attribute nào có mã loại lớn sẽ đứng sau.

Các attribute quan trọng thường sử dụng mã loại mặc định, tuy nhiên mã này có thể được định nghĩa lại trong siêu tập tin $AttrDef.

Mỗi loại attribute cũng có một tên gọi riêng, tên gọi được viết hoa toàn bộ, bắt đầu bằng kí hiệu $.

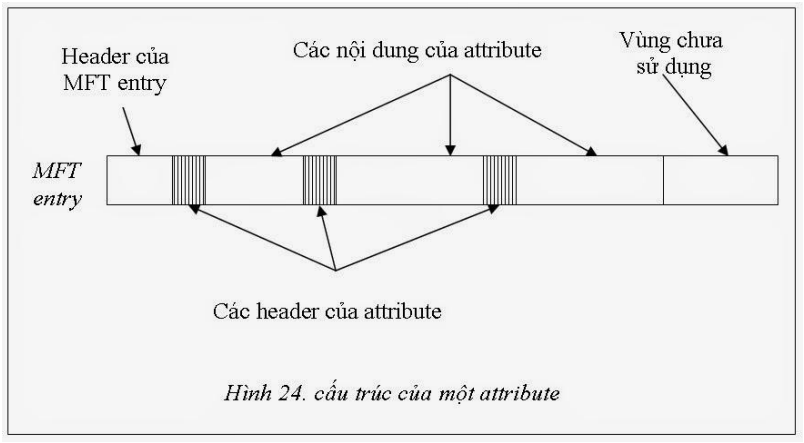
Bảng sau liệt kê một số loại attribute.

Mã loại (hệ 10)	Loại attribute	Mô tả

16	$STANDARD_INFORMATION	Chứa thông tin chung, ví dụ: các cờ, thời gian tạo, thời gian truy cập mới nhất, thời gian ghi mới nhất, người sở hữu, định danh bảo mật (security ID).
32	$ATTRIBUTE_LIST	Cho biết vị trí các attribute của một tập tin.
48	$FILE_NAME	Chứa tên tập tin (dạng Unicode), thời gian tạo, thời điểm ghi tập tin mới nhất, thời điểm truy cập mới nhất.
64	$VOLUME_VERSION	Chứa thông tin về ổ đĩa. Chỉ có ở phiên bản 1.2
64	$OBJECT_ID	Chứa định danh duy nhất của tập tin hoặc thư mục. Chỉ có ở phiên bản 3.0 trở về sau.
80	$SECURITY_DESCRIPTOR	Chứa thông tin về bảo mật và thông tin kiểm soát truy cập của tập tin.
96	$VOLUME_NAME	Chứa tên ổ đĩa logic.
112	$VOLUME_INFORMATION	Chứa thông tin về phiên bản của hệ thống quản lý tập tin và các cờ hiệu.
128	$DATA	Chứa nội dung của tập tin.
144	$INDEX_ROOT	Chứa nút gốc (root node) của cây chỉ mục (index tree).
160	$INDEX_ALLOCATION	Chứa các nút của cây chỉ mục (index tree) có gốc thuộc attribute $INDEX_ROOT.
176	$BITMAP	Chứa bitmap cho siêu tập tin $MFT và cho các chỉ mục.
192	$SYMBOLIC_LINK	Chứa thông tin liên kết mềm. Chỉ có ở phiên bản 1.2
192	$REPARSE_POINT	Chứa thông tin liên kết mềm. Có ở các phiên bản 3.0 về sau.
208	$EA_INFORMATION	Chứa thông tin đảm bảo việc tương thích với các ứng dụng trên nền OS/2.
224	$EA	Chứa thông tin đảm bảo việc tương thích với các ứng dụng trên nền OS/2.
256	$LOGGED_UTILITY_STREAM	Chứa khóa (key) và thông tin mã hóa attribute (encrypted attribute) trong các phiên bản từ 3.0 về sau.

Với hệ thống quản lý tập tin FAT32, có thao tác đọc và ghi nội dung tập tin. Tuy nhiên, đối với hệ thống NTFS, thao tác đọc và ghi nội dung tập tin được thay thế bằng thao tác đọc và ghi các attribute.

Cấu trúc của một attribute gồm hai phần: header của attribute và nội dung của attribute. Xem Hình 24 dưới đây.



Hình 24. cấu trúc của một attribute

Header của attribute

Header của attribute là phần đầu của mỗi attribute, có kích thước 16 byte. Header chứa thông tin về: mã loại, kích thước và tên của attribute. Header cũng chứa cờ báo cho biết attribute có được nén hay không? có được mã hóa hay không?

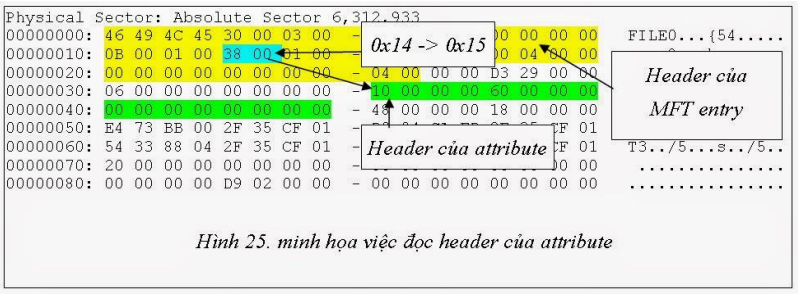
Cấu trúc cụ thể của header được minh họa trong bảng sau.

Byte thứ	Mô tả
0 – 3	Mã loại của attribute (type ID)
4 – 7	Kích thước của attribute
8 – 8	Cờ báo non-resident
9 – 9	Chiều dài của tên attribute
10 – 11	Vị trí (offset) chứa tên của attribute
12 – 13	Các cờ báo
14 – 15	Định danh của attribute (định danh này là duy nhất trong phạm vi một MFT entry)

Một MFT entry có thể chứa nhiều attribute cùng loại. Mỗi attribute có một mã định danh riêng (identifier) để phân biệt, mã định danh cần đảm bảo tính duy nhất trong phạm vi mỗi MFT entry.

Đọc nội dung header của attribute

Hình 25 dưới đây minh họa việc đọc header của attribute.



Hình 25. minh họa việc đọc header của attribute

Từ header của MFT entry, địa chỉ (offset) 0x14 -> 0x15 cho biết nơi bắt đầu của các attribute (đơn vị tính là byte). Giá trị của trường này là 0x0038, đổi sang hệ thập phân là 56. Nghĩa là, trong MFT entry này, các attribute sẽ được bắt đầu từ byte thứ 56.

Đọc 16 byte, bắt đầu từ byte thứ 56 sẽ là header của attribute đầu tiên, phần tô màu xanh lá cây trong Hình 25.

Giá trị cụ thể các trường của header được thể hiện trong bảng sau.

Byte thứ	Giá trị (Hệ 16 – Hệ 10)	Mô tả
0 – 3	0x00000010 – 16	Mã loại là 16: $STANDARD_INFORMATION
4 – 7	0x00000060 – 96	Kích thước của attribute là 96 byte
8 – 8	0x00 – 0	Attribute thuộc kiểu resident
9 – 9	0x00 – 0	Attribute này không được đặt tên, nên không có giá trị chiều dài của tên.
10 – 11	0x0000 – 0	Attribute này không được đặt tên, nên không có thông tin về vị trí của tên.
12 – 13	0x0000 – 0	Giá trị cờ báo
14 – 15	0x0000 – 0	Định danh của attribute (attribute ID) là 0.

Nội dung của attribute

Phần nội dung của attribute được sử dụng để chứa dữ liệu ở định dạng bất kì, với kích thước bất kì. Ví dụ, attribute chứa nội dung của một tập tin có thể có kích thước từ vài MB tới hàng GB. Tuy nhiên, kích thước của một MFT entry chỉ là 1024 byte, nên việc chứa toàn bộ nội dung của attribute trong MFT entry là không thực tế.

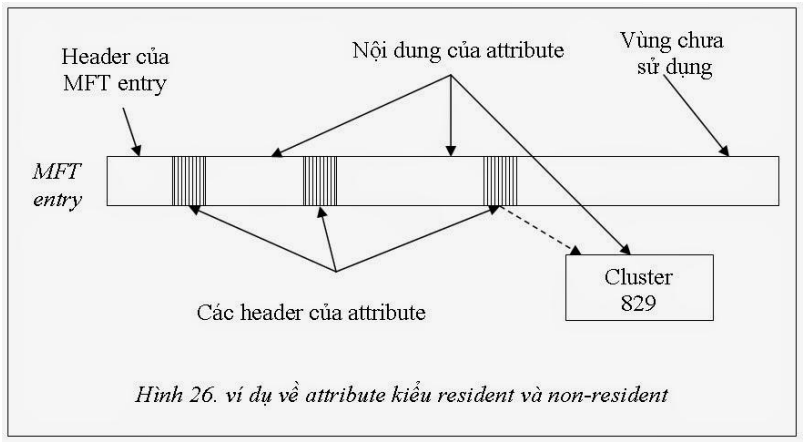
Để giải quyết vấn đề này, hệ thống NTFS cung cấp hai tùy chọn để lưu nội dung của attribute:

- lưu trực tiếp trong MFT entry,
- và lưu ở ngoài MFT entry.

Attribute có phần nội dung được lưu ngay trong MFT entry được gọi là resident attribute (attribute thường trú), thường áp dụng với các attribute có kích thước phần nội dung nhỏ.

Attribute lưu phần nội dung ở các cluster bên ngoài MFT entry được gọi là non-resident attribute (attribute không thường trú).

Trong header của attribute có trường cho biết attribute đó là resident hay non-resident. Nếu attribute thuộc loại resident, phần nội dung sẽ được đặt ngay sau header của attribute, ngược lại, nếu attribute thuộc loại non-resident, header sẽ cung cấp địa chỉ của cluster. Xem hình minh họa sau đây, Hình 25: attribute thứ nhất, thứ hai thuộc loại resident, attribute thứ ba thuộc loại non-resident.



Hình 26. ví dụ về attribute kiểu resident và non-resident

Tổ chức cụ thể của một attribute kiểu non-resident sẽ được trình bày sau.

Phần này sẽ trình bày về tổ chức của một attribute kiểu resident. Cấu trúc của một attribute kiểu resident được minh họa trong bảng sau.

Byte thứ	Mô tả
0 - 15	Cấu trúc header chuẩn (có trong tất cả các loại attribute – Hình 25).
16 – 19	Cho biết kích thước phần nội dung của attribute.
20 - 21	Cho biết nơi bắt đầu (offset) của phần nội dung.

Tùy thuộc vào mỗi loại attribute, phần nội dung của attribute sẽ có cấu trúc tổ chức khác nhau.

Để hiểu rõ hơn về cấu trúc của attribute, phần tiếp theo sẽ trình bày chi tiết về các attribute: $STANDARD_INFORMATION, $FILE_NAME, $DATA.

Tài liệu đã tham khảo:
Brian Carrie, *File System Forensic Analysis*, Addison Wesley Professional, 2005

He thong quan ly tap tin NTFS - 8 - Attribute $STANDARD_INFORMATION

(Tiếp theo của He thong quan ly tap tin NTFS - 7 - Attribute)

Attribute $STANDARD_INFORMATION

Attribute $STANDARD_INFORMATION có trong tất cả các tập tin và thư mục, attribute này chứa một số thông tin quan trọng như: thời gian, ngày tháng, quyền sở hữu (ownership), phân quyền sử dụng (security), hạn ngạch đĩa (quota). Những thông tin này không cần thiết đối với việc lưu trữ tập tin, tuy nhiên, Windows rất cần đến nó trong các ứng dụng.

Mã loại (type ID) mặc định của attribute này là 16. Trong Windows 2000 và XP, attribute này có kích thước 72 byte, trong Windows NT là 48 byte, trong Windows 7 là 96 byte.

Hệ thống Windows luôn sắp xếp các attribute trong một MFT entry theo thứ tự tăng dần của mã loại. Do mã loại của attribute $STANDARD_INFORMATION có giá trị nhỏ nhất nên nó luôn nằm ở vị trí đầu tiên, ngay sau header của MFT entry.

Attribute này có bốn thông tin về thời gian, gồm:

- Thời gian tạo tập tin (created).
- Thời gian thay đổi mới nhất nội dung hai attribute $DATA hoặc $INDEX (Modified Time).
- Thời gian thay đổi mới nhất thông tin mô tả tập tin (metadata). Thời gian này không được hiển thị cho người dùng (MFT Modified Time).
- Thời gian truy cập nội dung tập tin mới nhất (Accessed Time).

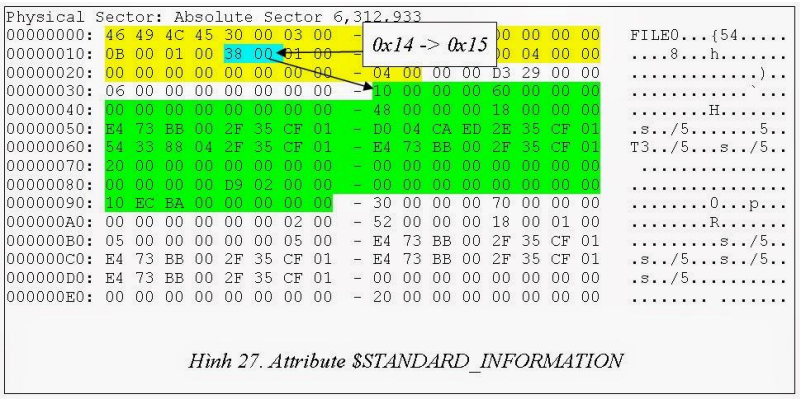
Attribute này cũng chứa “cờ báo” (flag) cho biết thông tin về kiểu của tập tin: chỉ đọc (read only), tập tin hệ thống (system), thông tin liên quan đến việc lưu dự phòng (archive).

Với các MFT entry không phải là của tập tin hoặc thư mục, attribute $STANDARD_INFORMATION còn cung cấp thông tin về: nén, tập tin “thưa” (sparse) hoặc mã hóa (encrypted). Nếu là MFT entry của tập tin hoặc thư mục, các thông tin này sẽ được lưu trong header của MFT entry.

Trong các hệ thống NTFS phiên bản 3.0 về sau (Windows 2000, XP…v.v), attribute này có chứa thêm bốn thông tin gồm:

- Thông tin về sở hữu (owner identity), được sử dụng trong việc tính hạn ngạch đĩa của mỗi người dùng.
- Thông tin cho biết dung lượng của tập tin được tính vào hạn ngạch đĩa của người dùng.
- Định danh bảo mật (Security ID), định danh này được sử dụng trong tập tin $Secure để xác định quyền truy cập tập tin.
- Giá trị USN (update sequence number), hỗ trợ trong việc tìm kiếm hàng loạt các tập tin đã có thay đổi trong một khoảng thời gian nhất định.

Hình 27 là attribute $STANDARD_INFORMATION trong MFT entry của tập tin Test.txt.



Hình 27. Attribute $STANDARD_INFORMATION

Quan sát Hình 27, trong phần header của MFT entry, offset 0x14 -> 0x15 cho biết vị trí bắt đầu của các attribute, có giá trị là 0x0038 = 56.

Trong một MFT entry, các attribute được sắp xếp theo thứ tự tăng dần của mã loại (type ID). Attribute này có mã loại nhỏ nhất nên luôn luôn nằm ở vị trí đầu tiên.

Cấu trúc của attribute $STANDARD_INFORMATION được thể hiện ở bảng sau.

Byte thứ	Mô tả
0 – 15	Header của attribute $STANDARD_INFORMATION.
16 – 19	Kích thước phần nội dung của attribute $STANDARD_INFORMATION. 0x00000048 = 72 (byte).
20 – 21	Nơi bắt đầu (offset) của phần nội dung attribute $STANDARD_INFORMATION. 0x0018 = 24 (byte thứ 24 tính từ đầu attribute).

Bảng sau là nội dung header của attribute $STANDARD_INFORMATION.

Byte thứ	Giá trị (Hệ 16 – Hệ 10)	Mô tả
0 – 3	0x00000010 – 16	Mã loại là 16.
4 – 7	0x00000060 – 96	Kích thước của attribute $STANDARD_INFORMATION là 96 byte.
8 – 8	0x00 – 0	Attribute thuộc kiểu resident.
9 – 9	0x00 – 0	Attribute này không được đặt tên, nên không có giá trị chiều dài của tên.
10 – 11	0x0000 – 0	Attribute này không được đặt tên, nên không có thông tin về vị trí của tên.
12 – 13	0x0000 – 0	Giá trị cờ báo.
14 – 15	0x0000 – 0	Định danh của attribute (attribute ID) $STANDARD_INFORMATION là 0.

Phần nội dung của attribute $STANDARD_INFORMATION bắt đầu tại byte 24 (tính từ đầu attribute $STANDARD_INFORMATION), kích thước của phần nội dung là 72 byte.

Cấu trúc của phần nội dung attribute $STANDARD_INFORMATION được thể hiện ở bảng sau.

Byte thứ	Giá trị hệ 16 – Hệ 10	Mô tả
0 – 7	0x01CF 352F 00BB 73E4 - 130381390209053668	Thời gian tạo tập tin: Saturday, March 1, 2014 4:17:01AM UTC.

8 – 15	0x01CF 352E EDCA 04D0 – 130381389891241168	Thời gian thay đổi mới nhất nội dung hai attribute $DATA hoặc $INDEX: Saturday, March 1, 2014 4:16:29AM UTC.
16 – 23	0x01CF 352F 0488 3354 – 130381390272803668	Thời gian thay đổi mới nhất thông tin mô tả tập tin: Saturday, March 1, 2014 4:17:07AM UTC.
24 – 31	0x01CF 352F 00BB 73E4 – 130381390209053668	Thời gian truy cập nội dung tập tin mới nhất: Saturday, March 1, 2014 4:17:01AM UTC.
32 – 35	0x00000020	Giá trị cờ báo, tập tin được đánh dấu là archive. (xem bảng về các giá trị của cờ ở bên dưới).
36 – 39	0x00000000	Maximum number of versions
40 – 43	0x00000000	Version number
44 – 47	0x00000000	Class ID
48 – 51	0x00000000	Định danh sở hữu - Owner ID (từ phiên bản 3.0 về sau).
52 – 55	0x000002D9	Định danh bảo mật - Security ID (từ phiên bản 3.0 về sau). Lưu ý: đây không phải là SID trong Windows.
56 – 63	0x00000000 00000000	Thông tin về hạn ngạch - Quota charged (từ phiên bản 3.0 về sau)
64 – 71	0x0000 0000 00BA EC10	Giá trị của USN (update sequence number) (từ phiên bản 3.0 về sau)

Giá trị về thời gian được biểu diễn bằng một số 64 bit, đây là kết quả của: (số nano giây tính từ thời điểm 1/1/1601 UTC)/100, hay nói cách khác, đây là số 100 nano giây tính từ thời điểm 1/1/1601 UTC. UTC là một thỏa hiệp viết tắt của Coordinated Universal Time, là giờ chuẩn quốc tế, tạm dịch là Giờ phối hợp quốc tế (wikipedia).

Ví dụ, thời gian tạo tập tin là: Saturday, March 1, 2014 4:17:01AM UTC sẽ được biểu diễn là: 0x01CF 352F 00BB 73E4, đổi ra hệ 10 là 130381390209053668 (trăm nano giây). Có thể sử dụng công cụ đổi tại website: www.silisoftware.com/tools/date.php.

Bảng sau là giá trị và ý nghĩa của cờ tại offset 32 –> 35.

Giá trị cờ	Ý nghĩa
0x0001	Chỉ đọc (read only).
0x0002	Ẩn (hidden).
0x0004	Thuộc hệ thống (system).
0x0020	Thông tin phục vụ việc lưu dự phòng. Tập tin được đánh dấu là archive.
0x0040	Thuộc thiết bị (Device).
0x0080	#Normal.
0x0100	Temporary.
0x0200	Tập tin ‘thưa’ - Spares file.
0x0400	Reparse point.
0x0800	Nén (compressed).
0x1000	Offline
0x2000	Nội dung không được tạo chỉ mục để tăng tốc độ tìm kiếm.
0x4000	Mã hóa (encrypted).

He thong quan ly tap tin NTFS - 9 - Attribute $FILE_NAME & $DATA

(tiếp theo của He thong quan ly tap tin NTFS - 8 - Attribute $STANDARD_INFORMATION)

Attribute $FILE_NAME

Mỗi tập tin hoặc thư mục (từ đây gọi tắt là tập tin) luôn có ít nhất một attribute $FILE_NAME trong MFT entry của nó. Một bản sao của attribute $FILE_NAME cũng được lưu trong index của thư mục cha (chứa nó), hai phiên bản này không nhất thiết phải giống nhau hoàn toàn về nội dung. Nội dung của attribute $FILE_NAME trong index của thư mục cha sẽ được đề cập sau, phần này chỉ xem xét attribute $FILE_NAME trong MFT entry.

Mã loại của attribute này là 48. Kích thước của attribute này không cố định, tùy thuộc vào chiều dài của tên tập tin. Cụ thể, kích thước của attribute là: 66 + chiều dài của tên tập tin.

Tên của tập tin là một chuỗi kí tự kiểu UTF-16 Unicode, được định dạng theo kiểu DOS 8.3, Win32 hoặc POSIX. Windows thường yêu cầu một tập tin ít nhất phải có định dạng tên kiểu DOS 8.3, do đó trong attribute $FILE_NAME sẽ có cả hai loại định dạng là: DOS và dạng tên đầy đủ. Tùy thuộc vào kiểu định dạng tên, sẽ có quy định những kí tự nào gọi là hợp lệ khi đặt tên cho tập tin.

Attribute $FILE_NAME có chứa địa chỉ (file reference) MFT entry của thư mục cha chứa nó. Attribute $FILE_NAME cũng chứa bốn thông tin về thời gian, tương tự như trong attribute $STANDARD_INFORMATION. Tuy nhiên, hệ thống Windows không cập nhật thông tin về thời gian thường xuyên như trong attribute $STANDARD_INFORMATION, thông thường nó chỉ được cập nhật khi tập tin được tạo ra, được di chuyển hoặc khi đổi tên.

Trong attribute $FILE_NAME có trường cho biết kích thước của tập tin, tuy nhiên, giá trị trường này thường là 0.

Cuối cùng, attribute $FILE_NAME có chứa “cờ báo” cho biết một số thông tin liên quan đến tập tin này, ví dụ như: là thư mục, là tập tin chỉ đọc, là tập tin hệ thống, được nén, được mã hóa…v.v.

Nói chung, attribute $FILE_NAME chứa rất nhiều các thông tin tương tự như trong attribute $STANDARD_INFORMATION. Trong đó có hai thông tin quan trọng là tên của tập tin, tên này cũng được sử dụng để tạo chỉ mục trong thư mục và địa chỉ của thư mục cha, địa chỉ này giúp xác định đường dẫn.

Trong MFT entry, thông thường attribute $FILE_NAME nằm ở vị trí thứ hai và là attribute kiểu resident. Tuy nhiên, nếu một tập tin cần nhiều hơn một MFT entry thì sẽ có attribute $ATTRIBUTE_LIST nằm giữa attribute $STANDARD_INFORMATION và attribute $FILE_NAME.

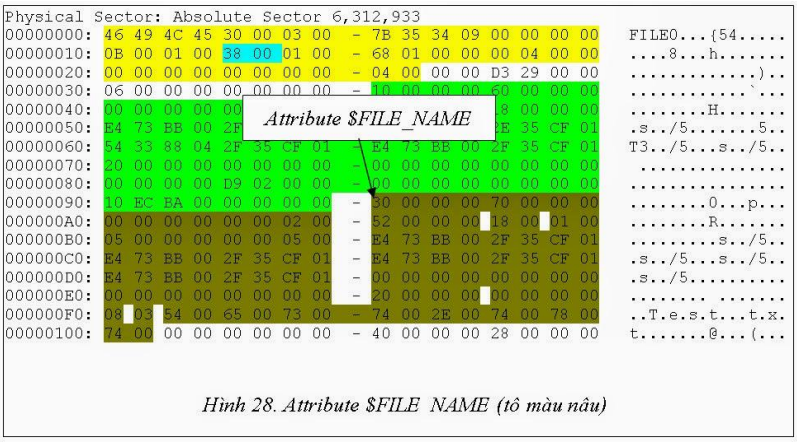
Cũng như tất cả các attribute khác, cấu trúc của attribute $FILE_NAME gồm các phần sau.

Byte thứ	Mô tả
0 – 15	Cấu trúc header chuẩn của attribute $FILE_NAME.
16 – 19	Kích thước phần nội dung của attribute $FILE_NAME.
20 – 21	Nơi bắt đầu (offset) của phần nội dung attribute $FILE_NAME.

Để đọc được nội dung của attribute $FILE_NAME, trước hết cần tính byte bắt đầu của attribute này.

Attribute $FILE_NAME nằm ngay sau attribute $STANDARD_INFORMATION. Attribute $STANDARD_INFORMATION bắt đầu tại offset 56, kích thước của $STANDARD_INFORMATION là 96 byte, vậy attribute $FILE_NAME sẽ bắt đầu tại vị trí (byte): 56 + 96 = 152.

Từ offset 152, đọc 16 byte sẽ là nội dung header của attribute $FILE_NAME, xem Hình 28.



Hình 28. Attribute $FILE_NAME (tô màu nâu)

Bảng sau là nội dung header của attribute $FILE_NAME.

Byte thứ	Giá trị (Hệ 16 – Hệ 10)	Mô tả
0 – 3	0x00000030 – 48	Mã loại là 48.
4 – 7	0x00000070 – 112	Kích thước của attribute $FILE_NAME là 112 byte.
8 – 8	0x00 – 0	Attribute thuộc kiểu resident.
9 – 9	0x00 – 0	Attribute này không được đặt tên, nên không có giá trị chiều dài của tên.
10 – 11	0x0000 – 0	Attribute này không được đặt tên, nên không có thông tin về vị trí của tên.
12 – 13	0x0000 – 0	Giá trị cờ báo.
14 – 15	0x0002 – 2	Định danh của attribute (attribute ID) $FILE_NAME là 2.

Đọc tiếp byte thứ 16 -> 19 để biết kích thước phần nội dung, 0x00000052 = 82 byte.

Đọc byte thứ 20 -> 21 để biết vị trí bắt đầu của phần nội dung, 0x0018 = 24, vậy phần nội dung sẽ được lưu bắt đầu từ byte thứ 24 tính từ đầu attribute.

Cấu trúc các trường của phần nội dung attribute $FILE_NAME được mô tả trong bảng sau.

Byte thứ	Giá trị hệ 16 – hệ 10	Mô tả
0 – 7	“05 00 00 00 00 00 05 00”	Địa chỉ MFT entry của thư mục cha (file reference).
8 – 15	0x01CF352F00BB73E4 – 130381390209053668	Thời gian tạo tập tin: Saturday, March 1, 2014 4:17:01AM UTC.
16 – 23	0x01CF352F00BB73E4 – 130381390209053668	Thời gian tập tin có thay đổi: Saturday, March 1, 2014 4:17:01AM UTC.
24 – 31	0x01CF352F00BB73E4 –	Thời gian MFT entry có thay đổi:

	130381390209053668	Saturday, March 1, 2014 4:17:01AM UTC.
32 – 39	0x01CF352F00BB73E4 – 130381390209053668	Thời gian truy cập tập tin mới nhất: Saturday, March 1, 2014 4:17:01AM UTC.
40 - 47	0x0000000000000000	Kích thước cấp phát cho tập tin. NTFS không sử dụng đến trường này, giá trị luôn là 0.
48 – 55	0x0000000000000000	Kích thước thật của tập tin. NTFS không sử dụng đến trường này, giá trị luôn là 0. Giá trị thật được lưu trong attribute $DATA.
56 – 59	0x00000020	Giá trị cờ báo, tập tin được đánh dấu là archive. (Xem thêm về giá trị và ý nghĩa của cờ đã đề cập trong phần attribute $STANDARD_INFORMATION).
60 – 63	“00 00 00 00”	Giá trị Reparse
64 – 64	0x08 – 8	Chiều dài của tên tập tin: 8 kí tự.
65 – 65	0x03 – 3	Giá trị cho biết định dạng tên tập tin (Namespace). Xem thêm bên dưới.
66+ (8)	“54 00 65 00 73 00 74 00”	Tên của tập tin: Test.txt

Bảng sau mô tả một số kiểu định dạng tên tập tin.

Giá trị	Kiểu định dạng tên	Mô tả
0	POSIX	Tên có phân biệt chữ hoa, chữ thường. Cho phép sử dụng tất cả các kí tự Unicode ngoại trừ ‘/’ và NULL.
1	Win32	Tên có phân biệt chữ hoa, chữ thường. Cho phép sử dụng tất cả các kí tự Unicode ngoại trừ ‘/’, ‘\’, ‘ :’, ‘ >’, ‘<’ và ‘?’.
2	DOS	Tên không phân biệt chữ hoa, chữ thường, tất cả đều được chuyển sang dạng chữ hoa. Số kí tự của phần tên phải ít hơn hoặc bằng tám kí tự, phần mở rộng phải ít hơn hoặc bằng ba kí tự.
3	Win32 & DOS	Định dạng kép Win32 và DOS. Khi định dạng DOS có thể lưu đầy đủ tên tập tin rồi, thì mặc định hiểu là nó cũng được lưu ở định dạng Win32 mà không cần lưu ở hai định dạng riêng biệt.

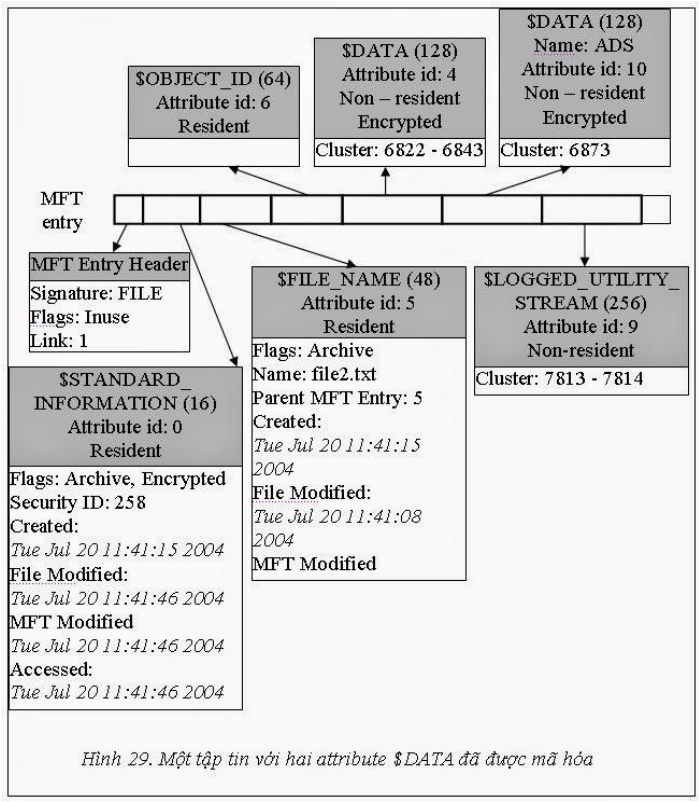
Attribute $DATA

Attribute $DATA được sử dụng để lưu trữ tất cả các loại dữ liệu. Mã loại của attribute này là 128. Attribute $DATA không có kích thước cố định. Mỗi tập tin có thể gồm một hoặc nhiều attribute $DATA. Attribute $DATA đầu tiên không có tên, các attribute $DATA tiếp sau phải được đặt tên cụ thể.

Trong hệ thống Windows, attribute $DATA cũng được tạo thêm khi người dùng nhập thông tin vào mục “Summary” (chuột phải vào tập tin\ chọn properties\ Summary), được tạo thêm do trình anti-virus, hoặc được tạo thêm do chương trình sao lưu dự phòng…v.v.

Attribute $DATA có thể được mã hóa để đảm bảo an toàn thông tin. Khi được mã hóa, “cờ báo” trong header của attribute sẽ được thiết lập, “khóa” của quá trình mã hóa được lưu trong attribute $LOGGED_UTILITY_STREAM.

Hình 29 minh họa một tập tin với hai attribute $DATA đã được mã hóa.



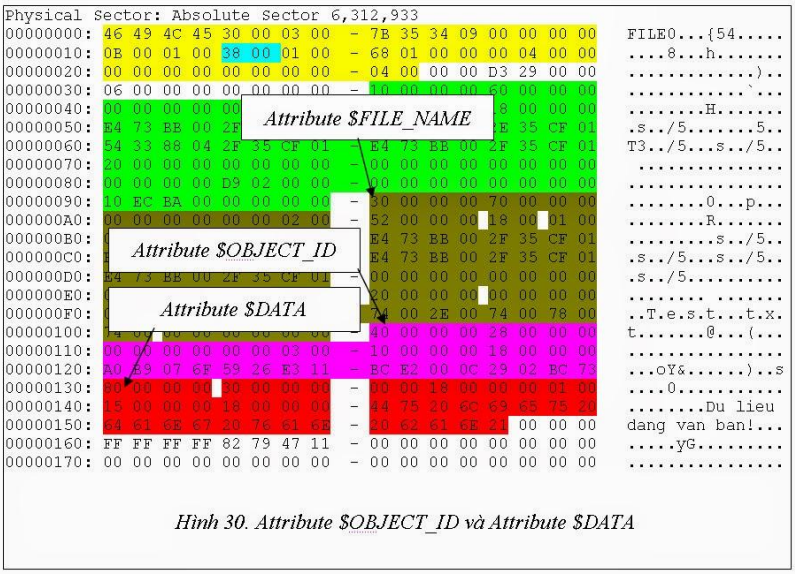
Hình 29. Một tập tin với hai attribute $DATA đã được mã hóa

Cấu trúc của attribute $DATA khá đơn giản. Sau phần header của attribute là phần nội dung, đây là dữ liệu ở dạng thô của một tập tin. Attribute $DATA không có kích thước tối thiểu và tối đa. Nếu kích thước phần nội dung vượt quá 700 byte, attribute sẽ được chuyển từ loại resident sang loại non-resident. Trong hầu hết các tập tin, attribute $DATA luôn nằm ở vị trí sau cùng trong MFT entry.

Phần sau đây trình bày quá trình đọc attribute $DATA, xem Hình 30.

Đọc header của attribute nằm kế sau attribute $FILE_NAME, gồm 16 byte. Xét mã loại, byte 0 -> 3 có giá trị là 0x00000040 = 64, kết luận: đây là attribute $OBJECT_ID. Tính kích thước của attribute $OBJECT_ID để nhảy qua attribute kế tiếp, byte 4 –> 7 có giá trị là 0x00000028 = 40 byte.

Vậy ta sẽ bỏ qua 40 byte của attribute $OBJECT_ID, để đọc header của attribute kế tiếp.



Hình 30. Attribute $OBJECT_ID và Attribute $DATA

Xét mã loại, byte 0 -> 3 có giá trị 0x00000080 = 128, kết luận: đây là attribute $DATA.

Bảng sau là nội dung header của attribute $DATA.

Byte thứ	Giá trị (Hệ 16 – Hệ 10)	Mô tả
0 – 3	0x00000080 – 128	Mã loại là 128
4 – 7	0x00000030 – 48	Kích thước của attribute $DATA là 48 byte
8 – 8	0x00 – 0	Attribute thuộc kiểu resident
9 – 9	0x00 – 0	Attribute này không được đặt tên, nên không có giá trị chiều dài của tên.
10 – 11	0x0018 – 24	Attribute này không được đặt tên, nhưng vẫn có thông tin về vị trí của tên?
12 – 13	0x0000 – 0	Giá trị cờ báo.
14 – 15	0x0001 – 1	Định danh của attribute (attribute ID) $DATA là 1.

Đọc tiếp byte thứ 16 -> 19 để biết kích thước phần nội dung, 0x00000015 = 21 byte.

Đọc byte thứ 20 -> 21 để biết vị trí bắt đầu của phần nội dung, 0x0018 = 24, tức là bắt đầu từ byte thứ 24 tính từ đầu attribute.

Nội dung của tập tin văn bản là “Du lieu dang van ban!”.

Cuối cùng, do chưa sử dụng hết 1024 byte của MTF entry, hệ thống sử dụng giá trị 0xFFFFFFFF để đánh dấu kết thúc phần nội dung của MFT entry.

Tài liệu đã tham khảo:

Brian Carrie, *File System Forensic Analysis*, Addison Wesley Professional, 2005

Tài liệu được trích từ website: http://legiacong.blogspot.com

Nếu còn các thắc mắc khác các em gởi mail: lvlong@fit.hcmus.edu.vn hoặc facebook: levietlong.teaching nhé.