

# PE 파일 포맷

- PE 파일 포맷이란?

## ❖ PE 포맷

PE 포맷(Portable Executable)은 윈도우 운영 체제에서 사용되는 실행 파일, DLL, object 코드, FON 폰트 파일[1] 등을 위한 파일 형식이다. PE 포맷은 윈도우 로더가 실행 가능한 코드를 관리하는데 필요한 정보를 캡슐화한 데이터 구조체이다. 이것은 링킹을 위한 동적 라이브러리 참조, API 익스포트와 임포트 테이블, 자원 관리 데이터 그리고 TLS 데이터를 포함한다. 윈도우 NT 운영체제에서, PE 포맷은 EXE, DLL, SYS (디바이스 드라이버), 그리고 다른 파일 타입들에서 쓰인다. 통일 확장 펌웨어 인터페이스 (EFI) 설명서는 PE가 EFI 환경에서 표준 실행 파일 형식이라고 언급한다.

윈도우 NT 운영 체제에서, PE는 현재 IA-32, IA-64, x86-64 (AMD64/Intel64), 그리고 ARM instruction set architectures (ISAs)를 지원한다. 윈도우 2000 이전에서, 윈도우 NT는 (그리고 PE) MIPS, Alpha, 그리고 파워PC ISAs를 지원했다. PE가 윈도우 CE에서 사용되므로, 이것은 MIPS, ARM (Thumb 포함), 그리고 SuperH ISA의 변형들을 지원하고 있다.

\* 위키백과 사이트에서 2015년10월30일자로 변경된 내용입니다.

<출처10>

# PE 파일 포맷

- PE 파일의 종류

- PE 파일 포맷을 가지고 있는 주요 파일들

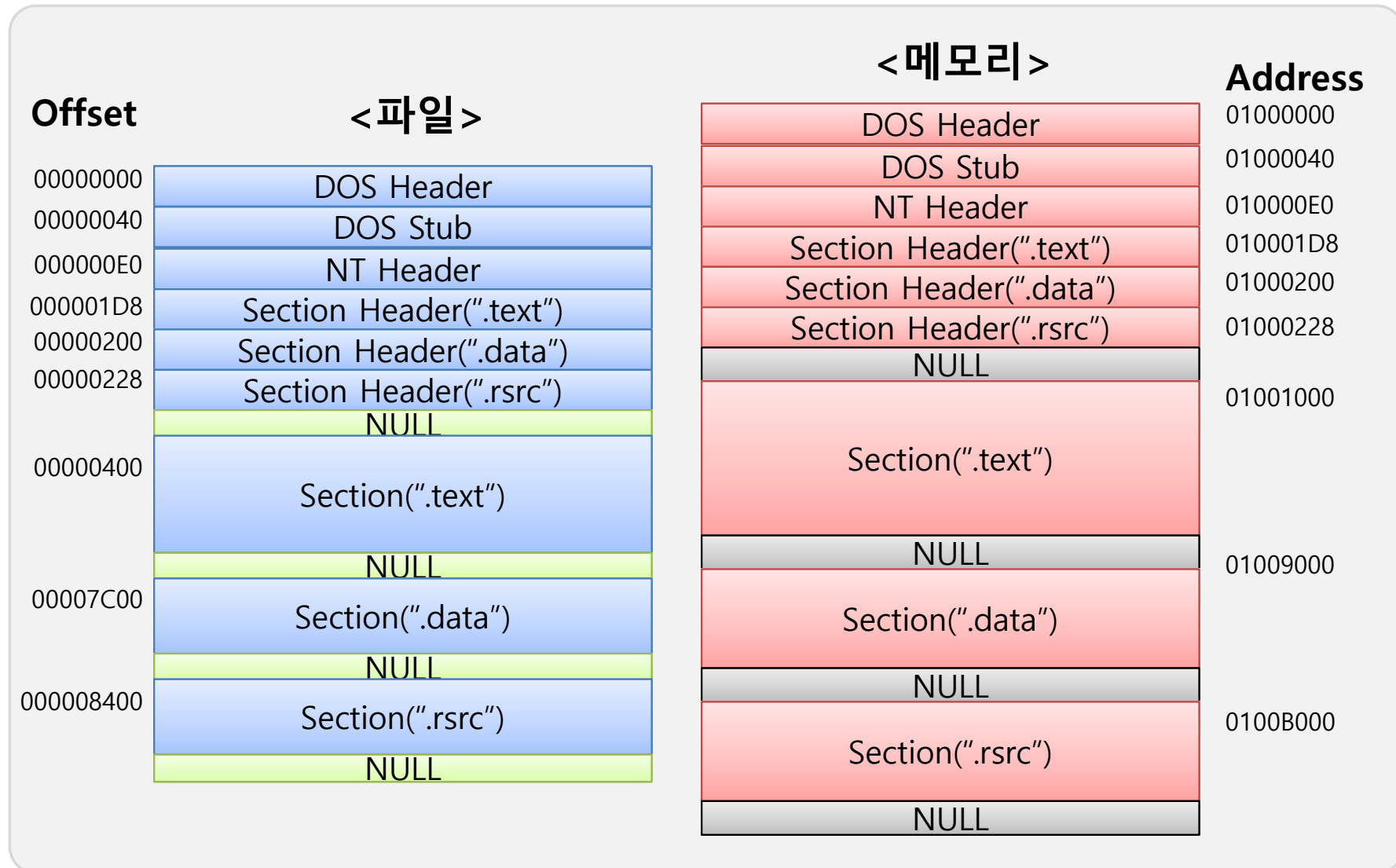
종류	확장자
실행 파일	exe, scr
라이브러리 파일	dll, ocx, cpl, drv
오브젝트 파일	obj
드라이버 파일	sys, vxd

<출처10>

# PE 파일 포맷

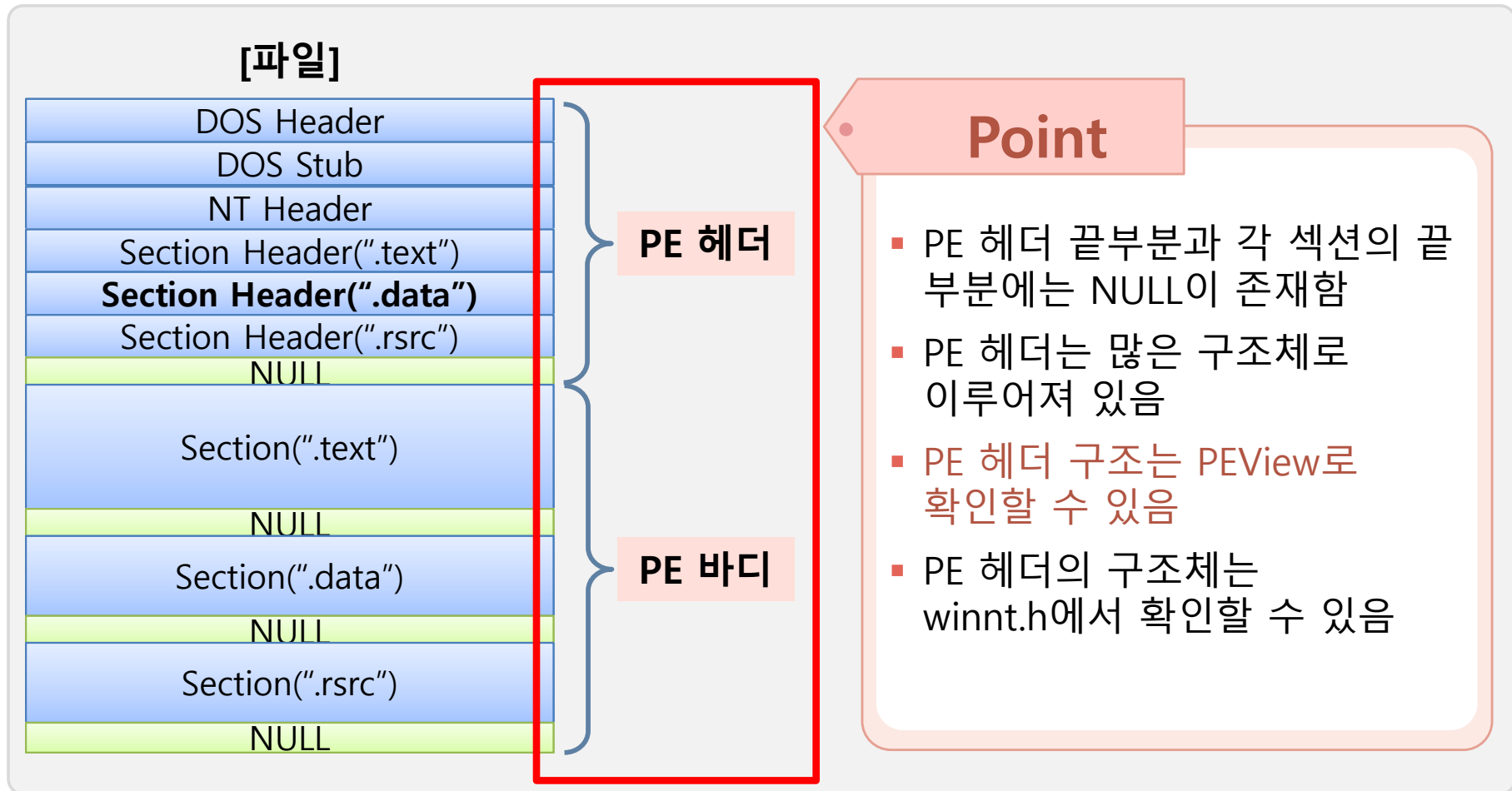
- PE 파일의 구조

[PE 파일이 메모리에 로딩되는 모습]



# PE 파일 포맷

## • PE 파일의 구조



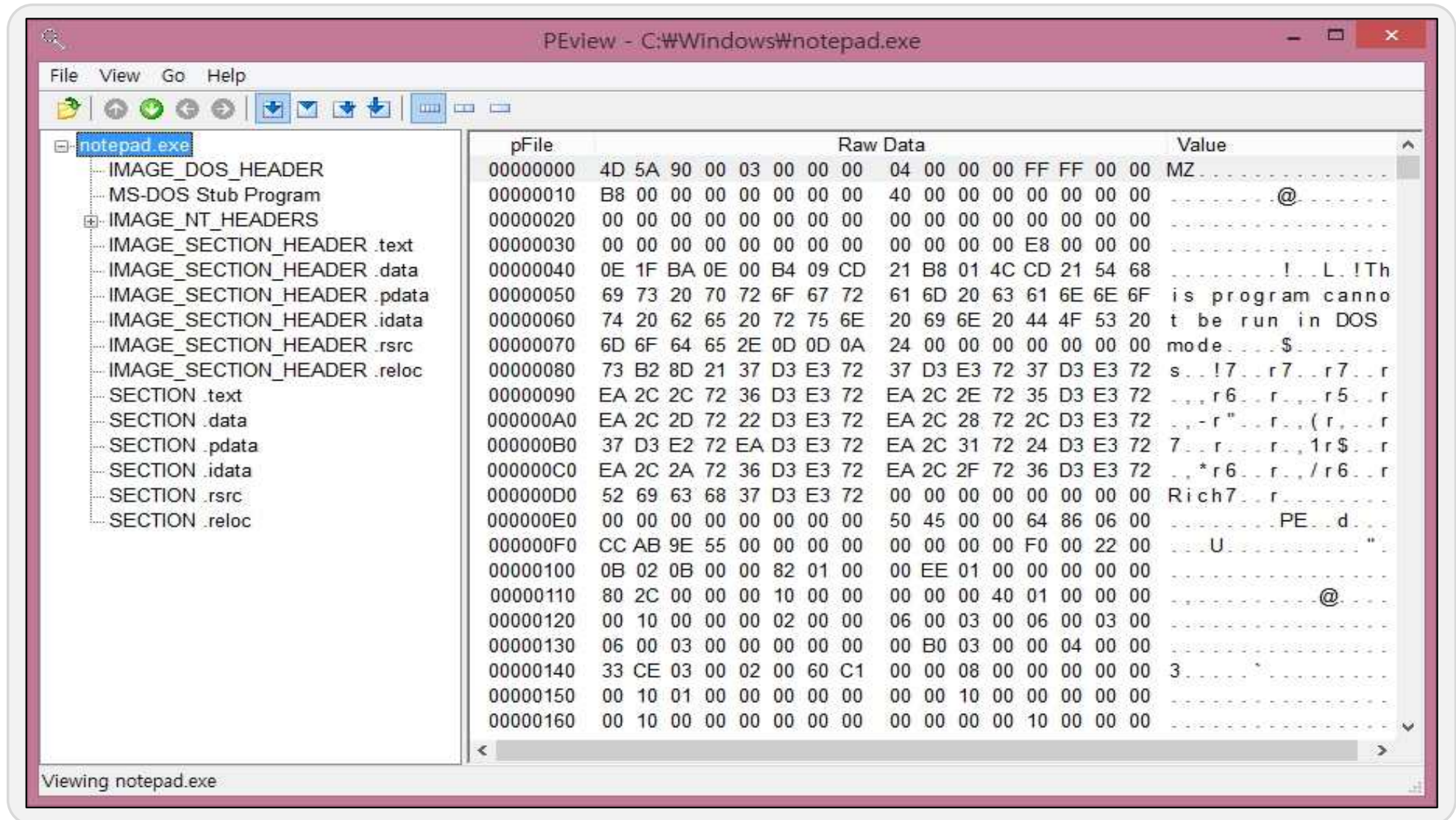
**TIP**

파일에서는 offset, 메모리에서는 VA(절대 주소)로 위치를 표현함

# PE 파일 포맷

- PE 파일의 구조

[PEView]



# PE 파일 포맷

- DOS Header

- PE 파일 포맷을 만들 때 그 당시에 쓰던 도스 파일에 대한 하위 호환성을 고려함
  - 중요한 멤버는 첫 번째 **e\_magic**(DOS Signature)와 마지막 멤버인 **e\_lfanew**(NT Header의 시작 주소) 두 가지임

• **Point**

- e\_magic은 DOS 시절 실행파일을 설계한 Mark Zbikowski의 이니셜로 만든 것임
  - e\_lfanew는 NT Header의 시작 주소를 나타냄
- 
- DOS Header의 e\_magic, e\_lfanew 값 중 하나라도 변경된다면 프로그램이 정상적으로 실행되지 않음

# PE 파일 포맷

- DOS Header

[PE파일 구조 중 DOS Header]

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
00000000	4D	5A	90	00	03	00	00	00	04	00	00	00	FF	FF	00	00	MZ.....ÿÿ..
00000010	B8	00	00	00	00	00	00	00	40	00	00	00	00	00	00	00	,.....@.....
00000020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000030	00	00	00	00	00	00	00	00	00	00	00	00	E0	00	00	00	.....à...
00000040	0E	1F	BA	0E	00	B4	09	CD	21	B8	01	4C	CD	21	54	68	..°..'.'Í!..LÍ!Th
00000050	69	73	20	70	72	6F	67	72	61	6D	20	63	61	6E	6E	6F	is program canno
00000060	74	20	62	65	20	72	75	6E	20	69	6E	20	44	4F	53	20	t be run in DOS
00000070	6D	6F	64	65	2E	0D	0D	0A	24	00	00	00	00	00	00	00	mode....\$.....
00000080	EC	85	5B	A1	A8	E4	35	F2	A8	E4	35	F2	A8	E4	35	F2	ì...[;`ä5ò`ä5ò`ä5ò
00000090	6B	EB	3A	F2	A9	E4	35	F2	6B	EB	55	F2	A9	E4	35	F2	kë:ò@ä5òkëUò@ä5ò
000000A0	6B	EB	68	F2	BB	E4	35	F2	A8	E4	34	F2	63	E4	35	F2	këhò»ä5ò`ä4òcä5ò
000000B0	6B	EB	6B	F2	A9	E4	35	F2	6B	EB	6A	F2	BF	E4	35	F2	këkò@ä5òkëjòçä5ò
000000C0	6B	EB	6F	F2	A9	E4	35	F2	52	69	63	68	A8	E4	35	F2	këoò@ä5òRich`ä5ò
000000D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000E0	50	45	00	00	4C	01	03	00	87	52	02	48	00	00	00	00	PE..L...#R.H....
000000F0	00	00	00	00	E0	00	0F	01	0B	01	07	0A	00	78	00	00	....à.....x..

# PE 파일 포맷

- DOS Stub

- DOS Stub의 존재 여부는 옵션이며 크기도 일정하지 않음
  - 있어도 되고 없어도 되는 부분임
  - 크기가 정해져 있는 것이 아니기 때문에 DOS stub 부분에 악성코드를 심기도 함
- DOS Stub에는 어셈블리어 코드와 This program cannot be run in DOS mode라는 문자열이 있음
- DOS 모드에서 해당 파일을 실행하면 위의 문자열을 출력하고 종료함

[PE파일 구조 중 DOS Stub]

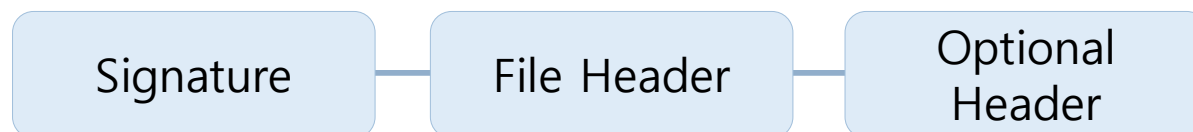
00000030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....à...
00000040	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	..°..'í!„.Lí!Th
00000050	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is program canno
00000060	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t be run in DOS
00000070	6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00	mode....\$......
00000080	EC 85 5B A1 A8 E4 35 F2 A8 E4 35 F2 A8 E4 35 F2	ì...[;`ä5ò`ä5ò`ä5ò
00000090	6B EB 3A F2 A9 E4 35 F2 6B EB 55 F2 A9 E4 35 F2	kë:ò@ä5òkëUò@ä5ò
000000A0	6B EB 68 F2 BB E4 35 F2 A8 E4 34 F2 63 E4 35 F2	këhò»ä5ò`ä4òcä5ò
000000B0	6B EB 6B F2 A9 E4 35 F2 6B EB 6A F2 BF E4 35 F2	këkò@ä5òkëjòçä5ò
000000C0	6B EB 6F F2 A9 E4 35 F2 52 69 63 68 A8 E4 35 F2	këoò@ä5òRich`ä5ò
000000D0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
000000E0	50 45 00 00 4C 01 03 00 87 52 02 48 00 00 00 00	PF..I...+R.H....



# PE 파일 포맷

- NT Header

- NT Header 구조체
  - DOS Header의 마지막 멤버가 가리키는 주소에서 시작함
  - 3개의 멤버로 구성되어 있음



⇒ Signature는 50450000h (PE..)의 값을 가지고 나머지 두 멤버는 또 다른 구조체임

[PE파일 구조 중 NT Header]

00000000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
000000E0	50 45 00 00 4C 01 03 00 87 52 02 48 00 00 00 00	PE..L...#R.H...
000000F0	00 00 00 00 E0 00 0F 01 0B 01 07 0A 00 78 00 00	....à.....x..
00000100	00 8C 00 00 00 00 00 00 9D 73 00 00 00 10 00 00	.@.....s.....
00000110	00 90 00 00 00 00 00 01 00 10 00 00 00 02 00 00	.....
00000120	05 00 01 00 05 00 01 00 04 00 00 00 00 00 00 00	.....
00000130	00 40 01 00 00 04 00 00 CE 26 01 00 02 00 00 80	.@.....î&.....€
00000140	00 00 04 00 00 10 01 00 00 00 10 00 00 10 00 00	.....
00000150	00 00 00 00 10 00 00 00 00 00 00 00 00 00 00 00	.....
00000160	04 76 00 00 C8 00 00 00 00 B0 00 00 04 83 00 00	.v..È....°...f..
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 50 13 00 00 1C 00 00 00	.....P.....
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 A8 18 00 00 40 00 00 00	....."....@...
000001B0	50 02 00 00 D0 00 00 00 00 10 00 00 48 03 00 00	P...Ð.....H...
000001C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
000001D0	00 00 00 00 00 00 00 00 2E 74 65 78 74 00 00 00	.....text...

# PE 파일 포맷

- **NT Header**

- File Header

- 주요 멤버는 다음 4가지임

- 1 Machine ①

CPU별 고유한 값, Intel x86=0x014C, Intel x64=0x0200의 값을 가짐

- 2 NumberOfSections ②

섹션의 개수를 나타냄

- 3 SizeOfOptionalHeader ③

Optional Header의 크기를 나타냄

# PE 파일 포맷

- NT Header

- File Header

- 주요 멤버는 다음 4가지임

4

Characteristics ④

파일의 속성을 나타냄

실행이 가능한지 또는 DLL파일인지 등의 정보가 있음

[NT Header 중 File Header]

00000000	00 00 00 00	1	00 00 00 00	2	00 00 00 00 00 00 00 00	.....
000000E0	50 45 00 00		4C 01 03 00		52 02 48 00 00 00 00 00	PE..L...#R.H....
000000F0	00 00 00 00	3	EO 00 0F 01	4	01 07 0A 00 78 00 00 00	....à.....x..
00000100	00 00 00 00		00 00 00 00		00 00 00 00 00 00 00 00	.....

# PE 파일 포맷

- NT Header

- Optional Header

- 주요 멤버는 다음 10가지임

[NT Header 중 Optional Header]

00000000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
000000E0	50 45 00 00 4C 01 03 00 07 52 02 48 00 00 00 00	PE..L...#R.H...
000000F0	00 00 00 00 E0 00 0F 01 0B 01 07 0A 78 00 00 00	....à.....x..
00000100	00 8C 00 00 00 00 00 00 9D 73 00 00 10 00 00 00	.....s.....
00000110	00 90 00 00 00 00 00 01 00 10 00 00 00 02 00 00	.....
00000120	05 00 01 00 05 00 01 00 00 00 00 00 00 00 00 00	.....
00000130	00 40 01 00 00 04 00 00 26 01 00 02 00 80 00 00	.@.....î&.....€
00000140	00 00 04 00 00 10 01 00 00 00 10 00 00 10 00 00	.....
00000150	00 00 00 00 10 00 00 00 00 00 00 00 00 00 00 00	.....
00000160	04 76 00 00 C8 00 00 00 00 B0 00 00 04 83 00 00	.v..È....°...f..
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 50 13 00 00 1C 00 00 00	.....P.....
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 A8 18 00 00 40 00 00 00	....."....@...
000001B0	50 02 00 00 D0 00 00 00 00 10 00 00 48 03 00 00	P...D.....H...
000001C0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
000001D0	00 00 00 00 00 00 00 00 25 65 78 74 00 00 00 00	.....text...
000001E0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....

※ 각 번호별 설명은 다음 페이지에 있습니다.

# PE 파일 포맷

- NT Header

- Optional Header

- 주요 멤버는 다음 10가지임

Magic ①	<ul style="list-style-type: none"><li>▪ Magic 넘버는 32비트면 10B, 64비트면 20B를 가짐</li></ul>
AddressOfEntryPoint ②	<ul style="list-style-type: none"><li>▪ EP의 RVA(Relative Virtual Address) 값을 가지고 있음</li></ul>
ImageBase ③	<ul style="list-style-type: none"><li>▪ PE 파일이 로딩되는 시작 주소임</li><li>▪ 파일을 메모리에 로딩 후 EIP 레지스터의 값 = ImageBase + AddressOfEntryPoint</li></ul>
SectionAlignment ④	<ul style="list-style-type: none"><li>▪ 메모리에서 섹션의 최소단위</li></ul>
FileAlignment ⑤	<ul style="list-style-type: none"><li>▪ 파일에서 섹션의 최소단위</li></ul>
SizeOfImage ⑥	<ul style="list-style-type: none"><li>▪ PE 파일이 메모리에 로딩되었을 때의 크기</li></ul>

# PE 파일 포맷

- **NT Header**

- Optional Header
  - 주요 멤버는 다음 10가지임

SizeOfHeader ⑦	PE헤더의 전체 크기
Subsystem ⑧	GUI, CUI 버전인지 드라이브파일인지 나타냄 1 = 드라이버, 2 = GUI, 3 = CUI
NumberOfRvaAndSizes ⑨	DataDirectory 배열의 개수를 나타냄
DataDirectory ⑩	IMAGE_DATA_DIRECTORY 구조체의 배열, 각 항목은 정해진 값을 가지고 각 항목 table이 어디에 위치해 있는지 RVA 값과 크기를 나타냄

(2/2)

# PE 파일 포맷

- Section Header



[PE파일 구조 중 Section Header]

000001D0	00 00 00 00 00 00 00 00	2E 75 78 74 00 00 00 00	.....text...
000001E0	48 77 00 00 00 10 00 00	00 78 00 00 00 04 00 00	Hw.....x.....
000001F0	00 00 00 00 00 00 00 00	00 00 00 00 20 00 00 60	.....
00000200	2E 64 61 74 61 00 00 00	38 1B 00 00 00 00 00 00	data ..

# PE 파일 포맷

- Section Header

- 종류

.text 섹션	컴파일된 코드, 코드, 실행, 읽기 속성, 컴파일 후의 결과가 저장되는 섹션
.data 섹션	읽기/쓰기가 가능하고 초기화된 전역변수, 상수가 존재하는 섹션
.rdata 섹션	읽기만 가능하고 문자열 상수나 const로 선언된 변수처럼 읽기만 가능한 읽기 전용데이터 섹션
.idata 섹션	읽기/쓰기 모두 가능하고 IAT와 관련된 정보가 들어있는 섹션
.edata 섹션	읽기만 가능하고 EAT와 관련된 정보가 들어 있는 섹션
.bss 섹션	읽기/쓰기가 가능하고 초기화 되지 않은 전역변수
.rsrc 섹션	읽기만 가능하고 리소스가(아이콘 등)이 저장되는 섹션



# 패킹과 난독화된 악성코드

- 패킹이란?

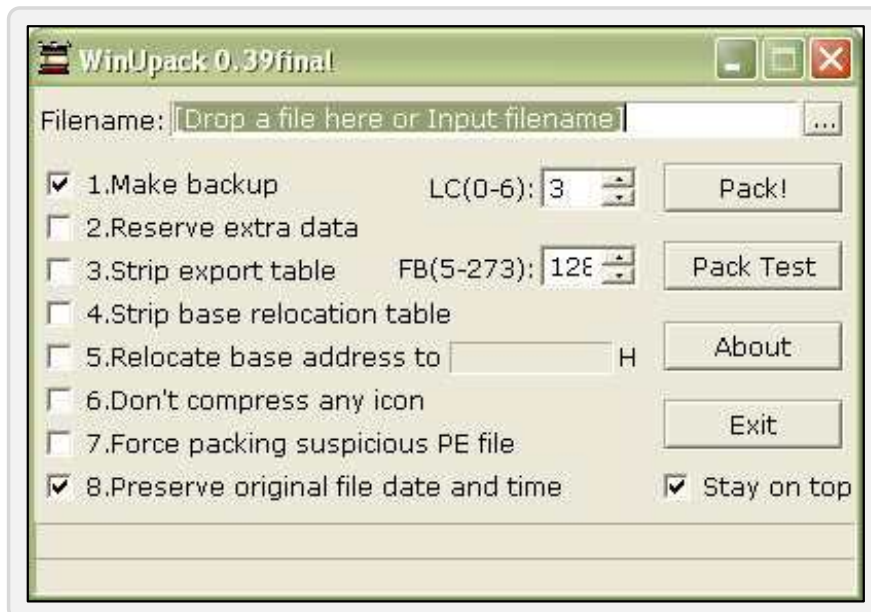
## 패킹

PE 파일의 크기를 줄이거나 내부 코드와 리소스를 감추기 위해 사용하는 기술

## 예 대표적인 예

UPX, ASPack, Upack, PESpin 등이 있음

[WinUpack 0.39final]



[ASPack 2.2]



# 패킹과 난독화된 악성코드

- UPX

- UPX를 이용해서 notepad.exe를 패킹함

```
C:\>c:\wpx\wpx.exe c:\w\notepad.exe
                                Ultimate Packer for eXecutables
                                Copyright (C) 1996 - 2011
UPX 3.08w      Markus Oberhumer, Laszlo Molnar & John Reiser   Dec 12th 2011

      File size      Ratio      Format      Name
      -----
      67584 -> 48128  71.21%  win32/pe  notepad.exe

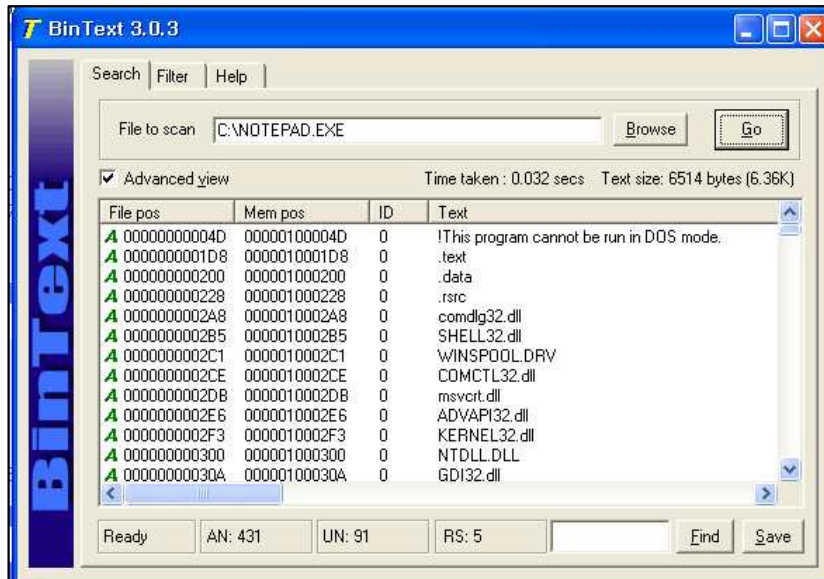
Packed 1 file.
```

# 패킹과 난독화된 악성코드

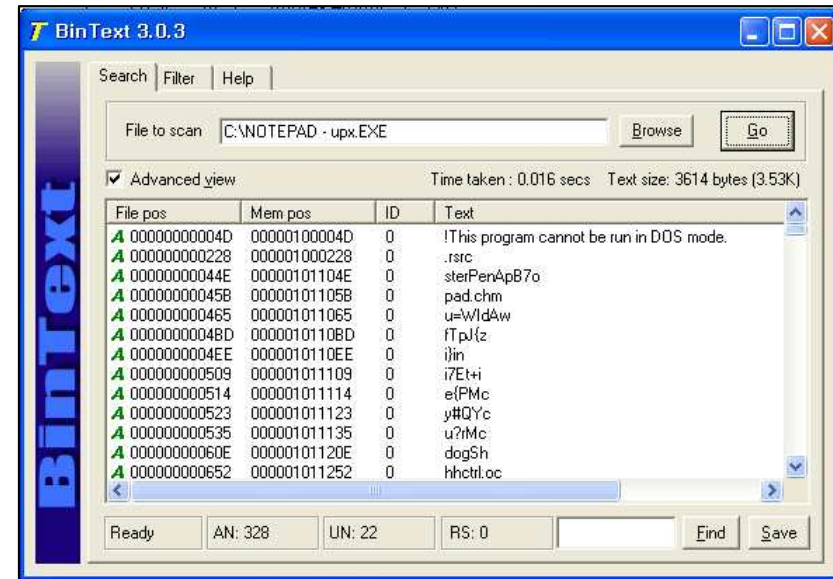
- UPX

- UPX를 이용해서 notepad.exe를 패킹한 결과

[패킹 전]



[패킹 후]



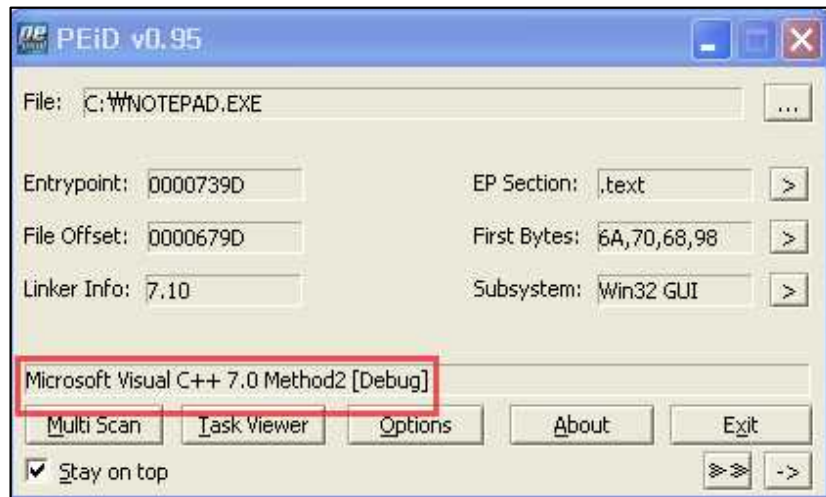
(1/3)

# 패킹과 난독화된 악성코드

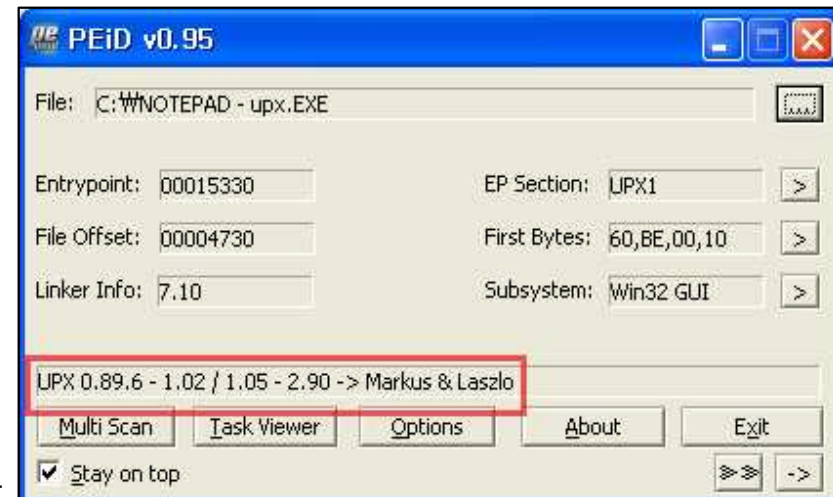
- UPX

- UPX를 이용해서 notepad.exe를 패킹한 결과

[패킹 전]



[패킹 후]



있음

(2/3)

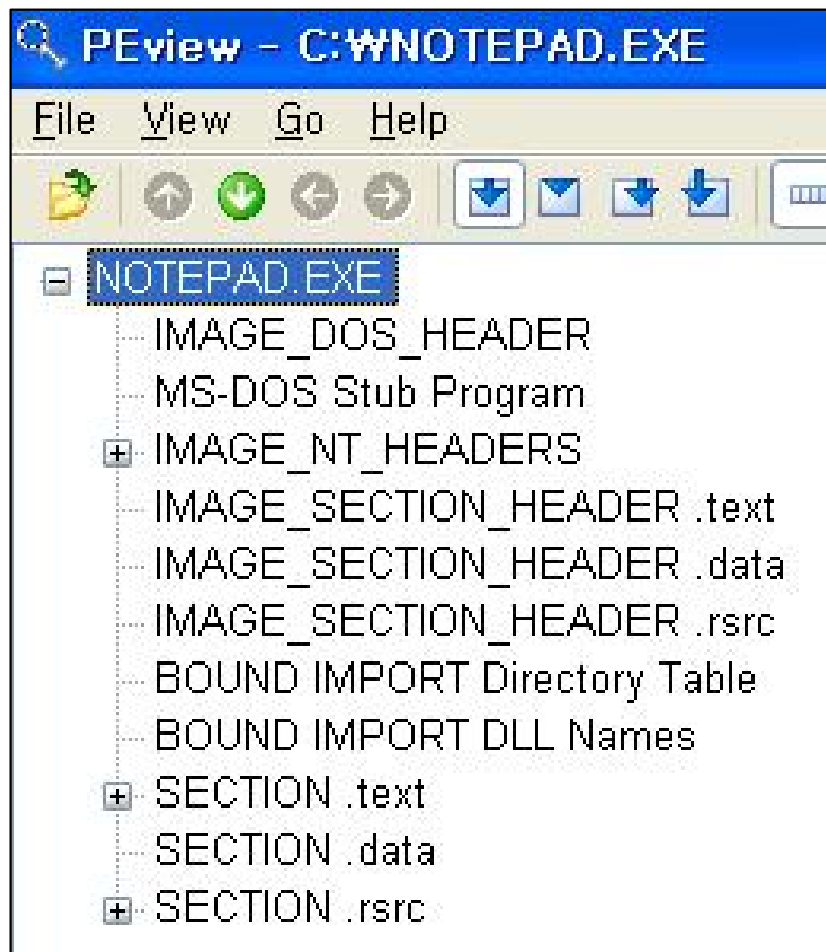


# 패킹과 난독화된 악성코드

- UPX

- UPX를 이용해서 notepad.exe를 패킹한 결과

[패킹 전]



[패킹 후]

