PE 재배치

SCP 20학번 박준형

목차 table of contents

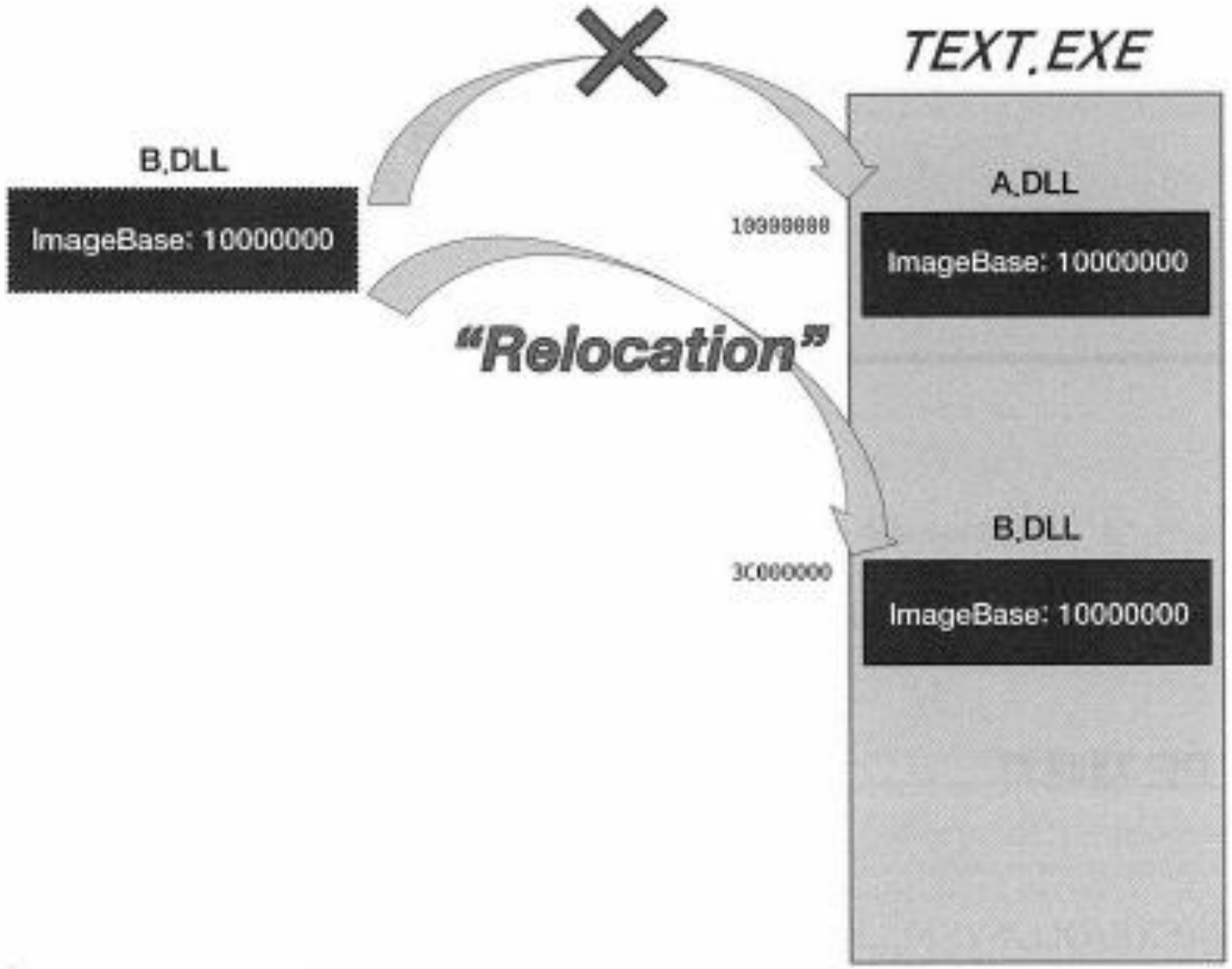
PE 재배치란?

Part 1 PE 재배치란?

PE 파일(EXE, DLL, SYS)이 프로세스 가상 메모리에 로딩(loading) 될 때 PE 헤더의 ImageBase 주소에 로딩됩니다. DLL(SYS) 파일의 경우 ImageBase 위치에 이미 다른 DLL(SYS) 파일이 로딩되어 있다면 다른 비어 있는 주소 공간에 로딩됩니다.

-> 즉, PE 재배치란 PE 파일이 ImageBase에 로딩되지 못하고 다른 주소에 로딩될 때 수행되는 작업들을 의미합니다.

Part 1 PE 재배치란? – DLL/SYS



Part 1 PE 재배치란? – EXE

ASLR(Address Space Layout Randomization) : EXE 파일이 실행될 때 마다 랜덤한 주소에 로딩하는 것

00000100	00022400	Size of Initialized Data
00000104	00000000	Size of Uninitialized Data
00000108	00003689	Address of Entry Point
0000010C	00001000	Base of Code
00000110	0000C000	Base of Data
00000114	01000000	Image Base
00000118	00001000	Section Alignment

첫 번째 실행

Address	Hex dump	Disassembly
00BC3689	$ E8 C5F9FFFF	CALL notepad.00BC3053
00BC368E	. 6A 58	PUSH 58

두 번째 실행

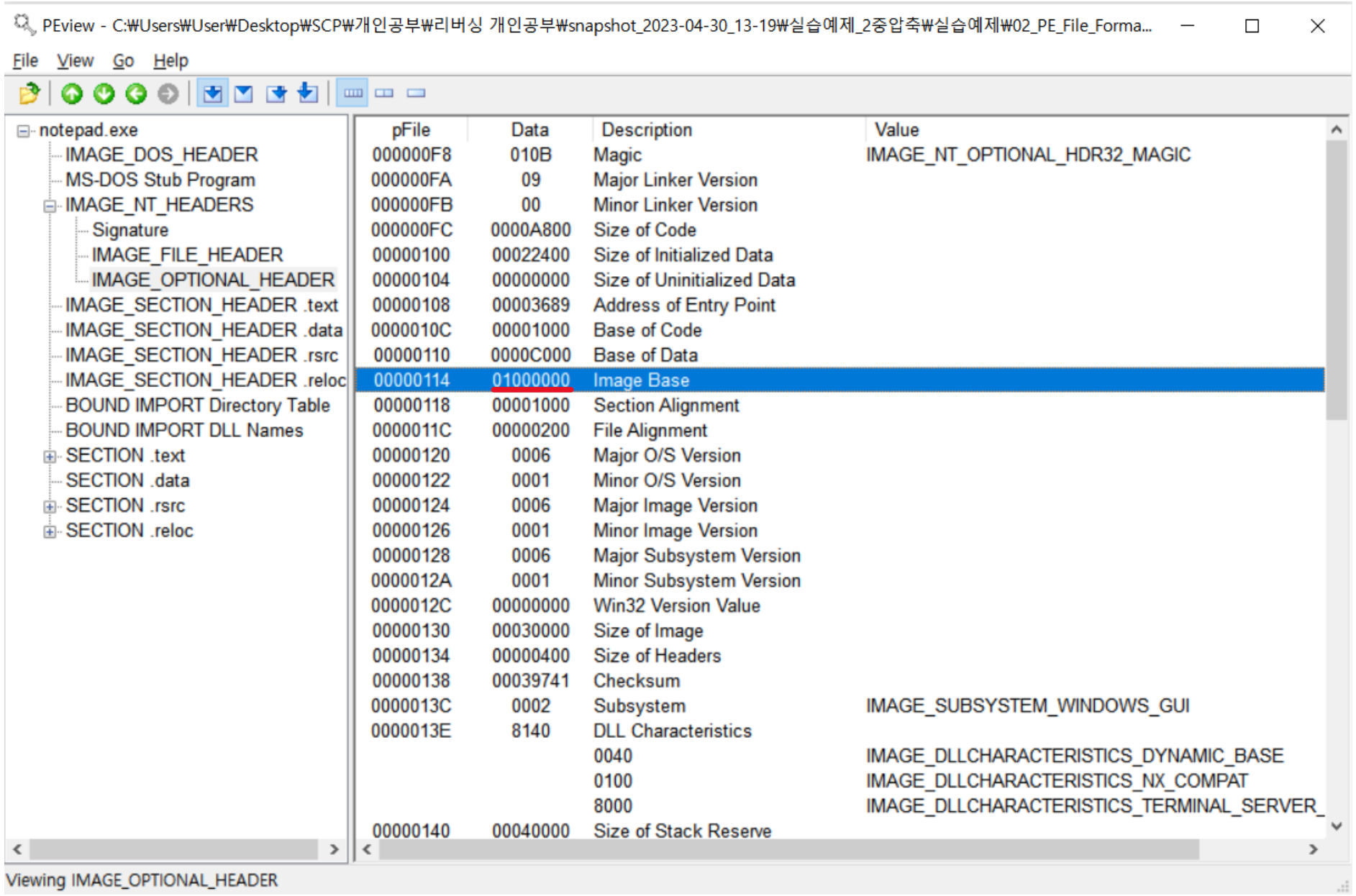
Address	Hex dump	Disassembly
008A3689	$ E8 C5F9FFFF	CALL notepad.008A3053
008A368E	. 6A 58	PUSH 58

세 번째 실행

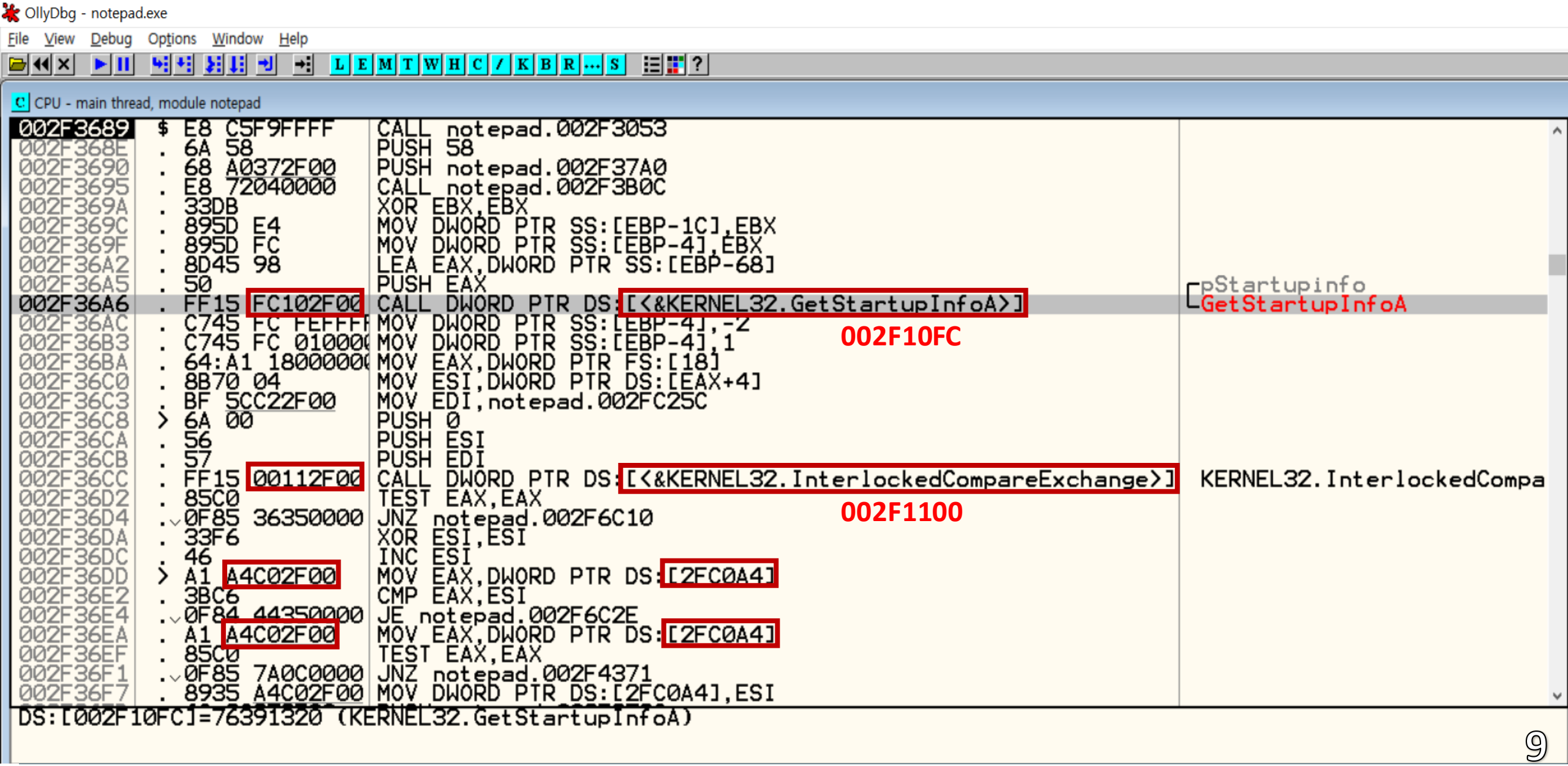
Address	Hex dump	Disassembly
008A3689	$ E8 C5F9FFFF	CALL notepad.008A3053
008A368E	. 6A 58	PUSH 58

PE 재배치 발생시 수행되는 작업

Part 2 PE 재배치 발생시 수행되는 작업



Part 2 PE 재배치 발생시 수행되는 작업



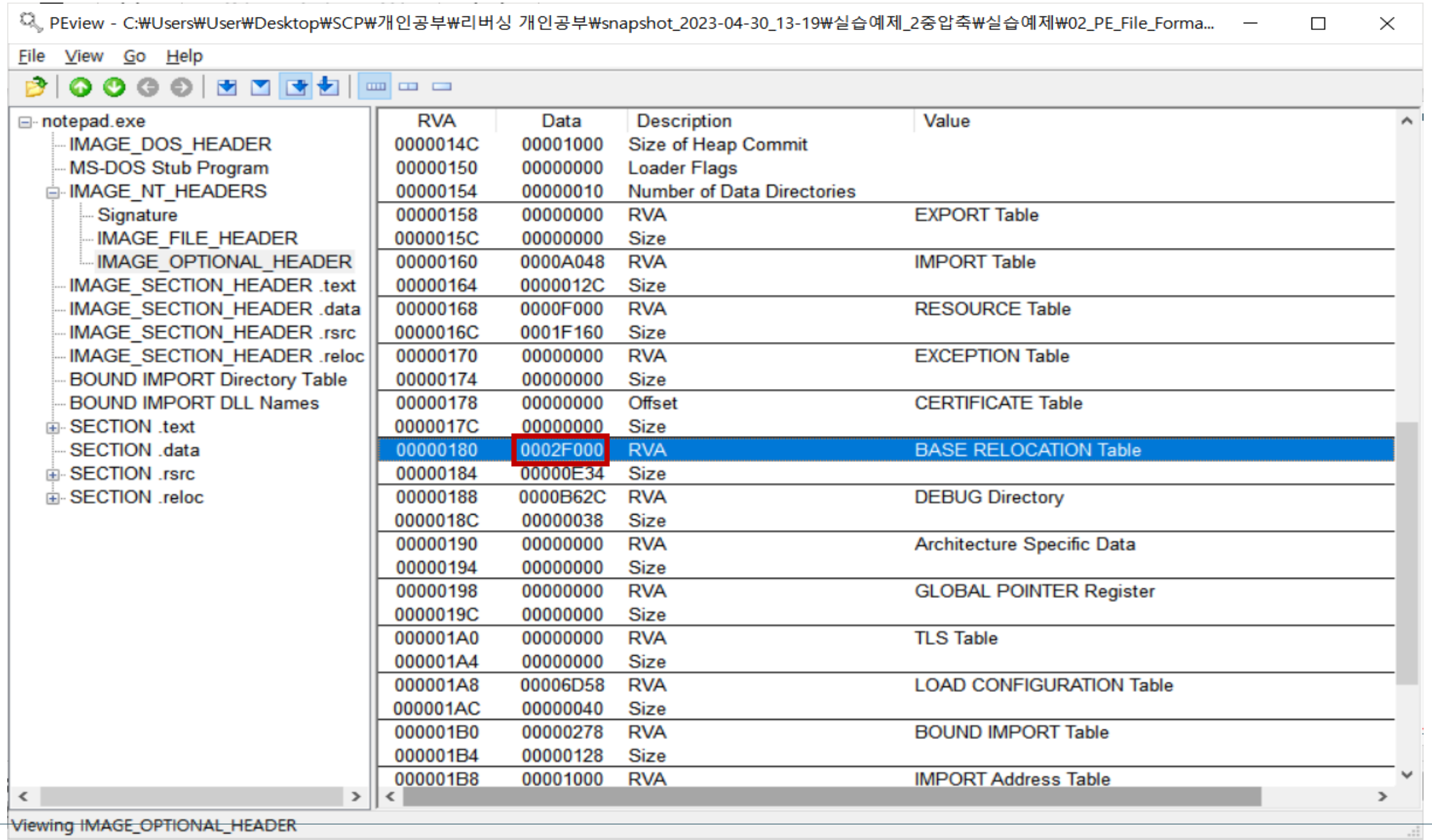
PE 재배치 동작 원리

1) 프로그램에서 하드코딩된 주소 위치를 찾는다.

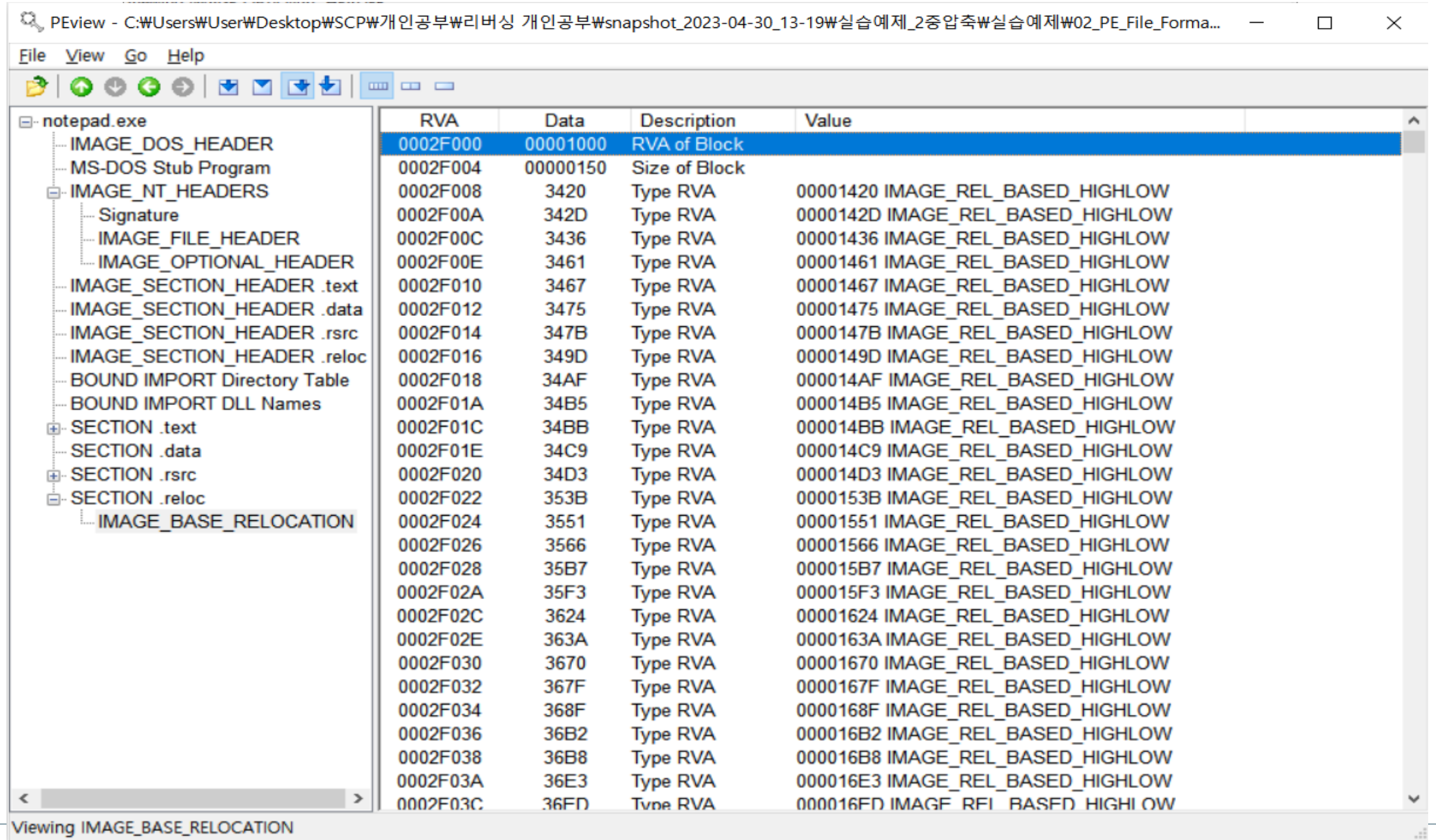
2) 값을 읽은 후 ImageBase를 뺀다. (VA -> RVA)

3) 실제로 로딩된 주소를 더한다. (RVA -> VA)

Part 3 PE 재배치 동작 원리 – Base Relocation Table



Part 3 PE 재배치 동작 원리 – Base Relocation Table



Part 3 PE 재배치 동작원리 -IMAGE BASE RELOCATION

```
typedef struct _IMAGE_BASE_RELOCATION {
        DWORD VirtualAddress; 각 블록들의 기준 주소. RVA값.
    DWORD SizeOfBlock;각 단위 블록의 크기.
//  WORD TypeOffset[1]; 이 구조체 밑으로 WORD 타입의 배열이 따라온다는 뜻. 프로그램에 하드코딩된 주소들의 offset.
} IMAGE_BASE_RELOCATION;
typedef IMAGE_BASE_RELOCATION UNALIGNED * PIMAGE_BASE_RELOCATION;

#define IMAGE_REL_BASED_ABSOLUTE              0
#define IMAGE_REL_BASED_HIGH                  1
#define IMAGE_REL_BASED_LOW                   2
#define IMAGE_REL_BASED_HIGHLOW               3
#define IMAGE_REL_BASED_HIGHADJ               4
#define IMAGE_REL_BASED_MIPS_JMPADDR          5
#define IMAGE_REL_BASED_MIPS_JMPADDR16        9
#define IMAGE_REL_BASED_IA64_IMM64            9
#define IMAGE_REL_BASED_DIR64                 10
```

Part 3 PE 재배치 동작원리-Base Relocation Table의 해석 방법

RVA	Data	Description	Value
0002F000	00001000	RVA of Block	기준 주소
0002F004	00000150	Size of Block	블록의 사이즈
0002F008	3420	Type RVA	00001420 IMAGE_REL_BASED_HIGHLOW
0002F00A	342D	Type RVA	0000142D IMAGE_REL_BASED_HIGHLOW
0002F00C	3436	Type RVA	00001436 IMAGE_REL_BASED_HIGHLOW
0002F00E	3461	Type RVA	00001461 IMAGE_REL_BASED_HIGHLOW

Typeoffset 값

Type(4비트)	Offset(12비트)
3	420

```
#define IMAGE_REL_BASED_HIGHLOW        3
```

PE 재배치 동작 실습

Part 4 PE 재배치 동작 실습

```
002F369F   . 895D FC        MOV DWORD PTR SS:[EBP-4],EBX
002F36A2   . 8D45 98        LEA EAX,DWORD PTR SS:[EBP-68]
002F36A5   . 50             PUSH EAX                                 ; pStartupinfo
002F36A6   . FF15 FC102F00  CALL DWORD PTR DS:[<&KERNEL32.GetStartupInfoA>]  ; GetStartupInfoA
002F36AC   . C745 FC FEFFFF MOV DWORD PTR SS:[EBP-4],-2
002F36B3   . C745 FC 010000 MOV DWORD PTR SS:[EBP-4],1
002F36BA   . 64:A1 18000000 MOV EAX,DWORD PTR FS:[18]
002F36C0   . 8B70 04        MOV ESI,DWORD PTR DS:[EAX+4]
```

Part 4 PE 재배치 동작 실습

RVA	Data	Description	Value
0002F362	3691	Type RVA	00003691 IMAGE_REL_BASED_HIGHLOW
0002F364	36A8	Type RVA	000036A8 IMAGE_REL_BASED_HIGHLOW
0002F366	36C4	Type RVA	000036C4 IMAGE_REL_BASED_HIGHLOW
0002F368	36CE	Type RVA	000036CE IMAGE_REL_BASED_HIGHLOW
0002F36A	36DE	Type RVA	000036DE IMAGE_REL_BASED_HIGHLOW
0002F36C	36EB	Type RVA	000036EB IMAGE_REL_BASED_HIGHLOW

Part 4 PE 재배치 동작 실습

RVA	Raw Data	Value
00003680	EA 5E 5D C3 90 90 90 90 90 E8 C5 F9 FF FF 6A 58	. ^] . . . . . j X
00003690	68 A0 37 00 01 E8 72 04 00 00 33 DB 89 5D E4 89	h . 7 . . . r . . . 3 . .] . .
000036A0	5D FC 8D 45 98 50 FF 15 FC 10 00 01 C7 45 FC FE	] . . E . P . . . . . E . .
000036B0	FF FF FF C7 45 FC 01 00 00 00 64 A1 18 00 00 00	. . . . E . . . . . d . . . . .
000036C0	8B 70 04 BF 5C C2 00 01 6A 00 56 57 FF 15 00 11	. p . . \ . . . j . V W . . . .
000036D0	00 01 85 C0 0F 85 36 35 00 00 33 F6 46 A1 A4 C0	. . . . . 6 5 . . 3 . F . . .
000036E0	00 01 3B C6 0F 84 44 35 00 00 A1 A4 C0 00 01 85	. . ; . . . D 5 . . . .

36A6 36A8

010010FC – 01000000 + 002F0000 =

(하드코딩된 주소 – ImageBase + 실제 로딩 주소)

```
002F36A6  . FF15 FC102F00  CALL DWORD PTR DS:[<&KERNEL32.GetStartupInfoA>]   GetStartupInfoA
```

Q&A