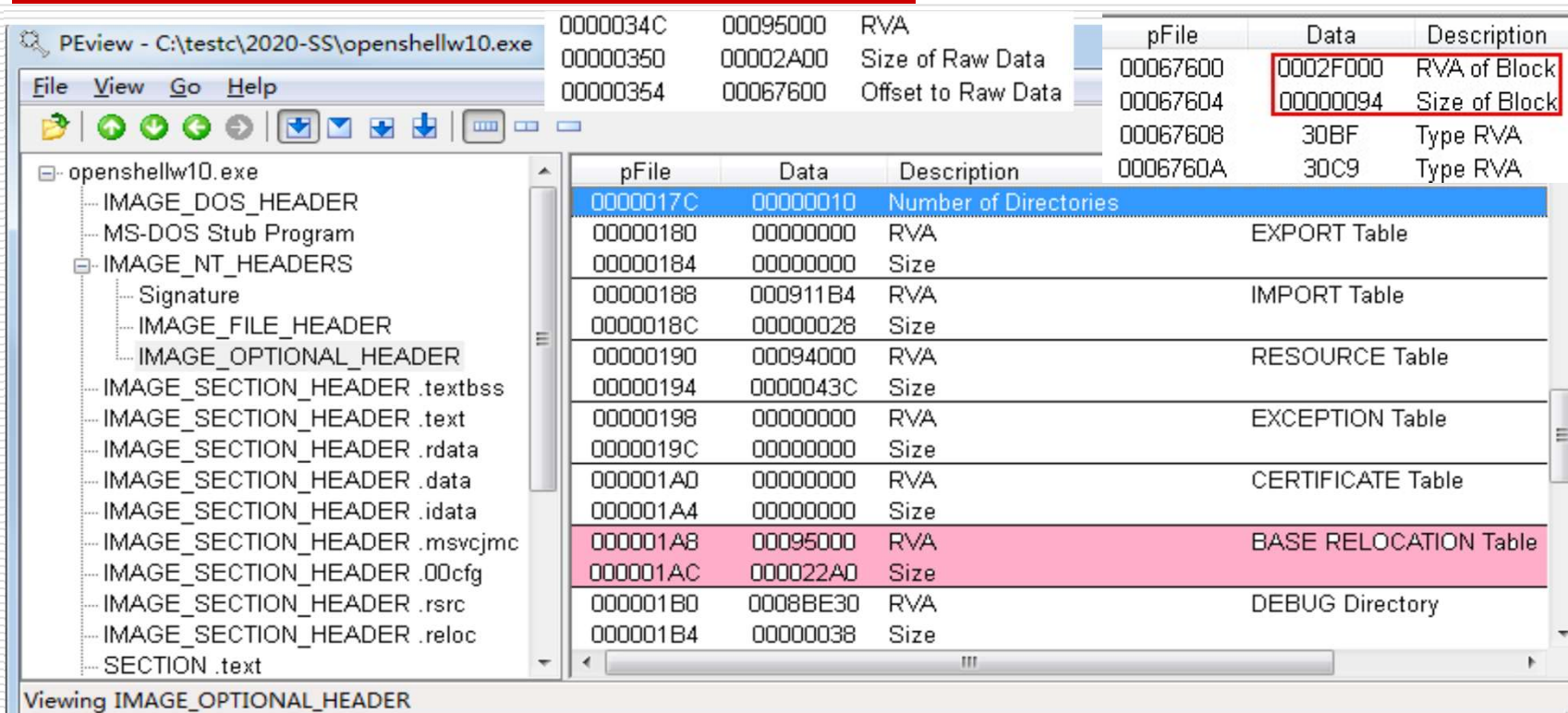


3.8 重定位表

- 重定位表在哪里
- 为什么需要重定位表
- 重定位表的解析
 - IMAGE_BASE_RELOCATION

重定位表在哪里



为什么需要重定位？

- ❑ PE文件中部分数据是以VA地址存储的，当PE文件无法加载到预期ImageBase时，这些地址需要修正。

OllyDbg - test.exe - [CPU - 主线程, 模块 - test]

地址	HEX 数据	反汇编	注释
00401000	68 40100000	PUSH 1040	
00401005	68 00304000	PUSH test.00403000	Title = "数学测试"
00401008	68 09304000	PUSH test.00403009	Text = "PE入口点测试1: 进入第一入口位置401000H!"
0040100F	6A 00	PUSH 0	hOwner = NULL
00401011	E8 2A000000	CALL <JMP.>user32.MessageBoxA	MessageBox
00401016	68 40100000	PUSH 1040	Style = MB_OK MB_ICONASTERISK MB_SYSTEMMODAL
0040101B	68 00304000	PUSH test.00403000	Title = "数学测试"
00401020	68 31304000	PUSH test.00403031	Text = "PE入口点测试1: 进入第二入口位置401016H!"
00401025	6A 00	PUSH 0	hOwner = NULL
00401027	E8 14000000	CALL <JMP.>user32.MessageBoxA	MessageBox
0040102C	6A 00	PUSH 0	ExitCode = 0
0040102E	E8 01000000	CALL <JMP.>kernel32.ExitProcess	ExitProcess
00401033	CC	INT3	
00401034	- FF25 00204000	JMP DWORD PTR DS:[<kernel32.ExitProcess>	kernel32.ExitProcess
00401038	- FF25 0C204000	JMP DWORD PTR DS:[<user32.usprintfA>]	user32.usprintfA
00401040	- FF25 08204000	JMP DWORD PTR DS:[<user32.MessageBoxA>]	user32.MessageBoxA
00401046	00	DB 00	
00401047	00	DB 00	
00401048	00	DB 00	

地址	HEX 数据	ASCII
00403000	BD CC D1 A7 B2 E2 CA D4 00 50 45 C8 EB BF DA B5	数学测试.PE入口
00403010	E3 B2 E2 CA D4 31 A3 BA BD F8 C8 EB B5 DA D2 BB	点?; 进入第一
00403020	C8 EB BF DA CE BB D6 C3 34 30 31 30 30 48 21	入口位置401000H!
00403030	00 50 45 C8 EB BF DA B5 E3 B2 E2 CA D4 31 A3 BA	.PE入口点测试1
00403040	BD F8 C8 EB BF DA CE BB D6 C3 34 30 31 30 30 48 21	进入第二入口位置
00403050	34 30 31 30 31 36 40 21 00 68 6B 64 6F 6F 72 64	401016H!.hboord
00403060	6C 6C 2E 64 6C 6C 00 44 6C 6C 52 65 67 69 73 74	ll.dll.DllRegist
00403070	65 72 53 65 72 76 65 72 00 64 3A 5C 6D 61 73 6D	erServer.d't'naam
00403080	33 32 5C 68 65 6C 6F 74 2E 65 78 65 00 00 00	32\hellot.exe...

UltraEdit - [C:\Documents and Settings\Administrator\Desktop\工具\test\test.exe]

文件(F) 编辑(E) 搜索(S) 方案(P) 视图(V) 格式(T) 列(L) 宏(M) 脚本(S) 高级(A) 窗口(W) 帮助(H)

test.exe

资源管理器

列: 1 2 3 4 5 6 7 8 9 a b c d e f

000003C0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ;
000003D0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ;
000003E0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ;
000003F0h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ;
00000400h: 68 40 10 00 00 68 00 30 40 00 68 09 30 40 00 6A ; h0...h00.h.00.j
00000410h: 00 E8 2A 00 00 68 40 10 00 00 68 00 30 40 00 ; .?...h0...h.00.
00000420h: 68 31 30 40 00 6A 00 E8 14 00 00 6A 00 E8 01 ; h100.j?...j.?
00000430h: 00 00 00 CC FF 25 00 20 40 00 FF 25 0C 20 40 00 ; ...?.0. ?.0.
00000440h: FF 25 08 20 40 00 00 00 00 00 00 00 00 00 00 ; %.0.....
00000450h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ;
00000460h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ;
00000470h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ;
00000480h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ;

为什么需要重定位？

□ 打开《计算器》程序

1. `printf("a program will be opened\r\n");`
2. `strcpy(RunAppName,"calc.exe");`
3. `ret_EXE = WinExec((char*)RunAppName, SW_SHOWMAXIMIZED);`
4. `printf("%s is opened, ret code=%d\r\n",RunAppName,ret_EXE);`
5. `printf("hello world is closed\r\n");`

◆ 函数表

`printf`

`strcpy`

`winexec`

◆ 常量表

`a program will be opened\r\n`

`calc.exe`

`%s is opened, ret code=%d\r\n`

`hello world is closed\r\n`

为什么需要重定位？

□ 打开《计算器》程序

```
c7 04 24 24 30 40 00    mov     DWORD PTR [esp], 0x403024
e8 8b 08 00 00          call    401bf0 <_puts>
8d 85 f4 fb ff ff      lea     eax, [ebp-0x40c]
c7 00 63 61 6c 63      mov     DWORD PTR [eax], 0x636c6163
c7 40 04 2e 65 78 65    mov     DWORD PTR [eax+0x4], 0x6578652e
c6 40 08 00             mov     BYTE PTR [eax+0x8], 0x0
c7 44 24 04 03 00 00    mov     DWORD PTR [esp+0x4], 0x3
00
8d 85 f4 fb ff ff      lea     eax, [ebp-0x40c]
89 04 24               mov     DWORD PTR [esp], eax
e8 be 08 00 00          call    401c50 <_WinExec@8>
83 ec 08               sub     esp, 0x8
89 45 f4               mov     DWORD PTR [ebp-0xc], eax
8b 45 f4               mov     eax, DWORD PTR [ebp-0xc]
89 44 24 08            mov     DWORD PTR [esp+0x8], eax
8d 85 f4 fb ff ff      lea     eax, [ebp-0x40c]
89 44 24 04            mov     DWORD PTR [esp+0x4], eax
c7 04 24 4c 30 40 00    mov     DWORD PTR [esp], 0x40304c
e8 43 08 00 00          call    401bf8 <_printf>
c7 04 24 68 30 40 00    mov     DWORD PTR [esp], 0x403068
e8 37 08 00 00          call    401bf8 <_printf>
```

◆ 函数表

0x401bf8:printf

0x401bf0:put

0x401c50:winexec

strcpy

◆ 常量表

0x403024:a program will be
opened\r\n

calc.exe

0x40304c:%s is opened, ret
code=%d\r\n

0x404068:hello world is closed\r\n

为什么需要重定位？

□ 打开《计算器》程序

00401bf0 <_puts>:

401bf0:ff 25 34 61 40 00 jmp WORD PTR ds:0x406134

401bf6: 90 nop

401bf7: 90 nop

00401bf8 <_printf>:

401bf8:ff 25 30 61 40 00 jmp WORD PTR ds:0x406130

401bfe: 90 nop

401bff: 90 nop

00401c50 <_WinExec@8>:

401c50:ff 25 f0 60 40 00 jmp WORD PTR ds:0x4060f0

401c56: 90 nop

401c57: 90 nop

```
c7 04 24 24 30 40 00 mov     DWORD PTR [esp], 0x403024
e8 8b 08 00 00      call    401bf0 <_puts>
8d 85 f4 fb ff ff   lea     eax, [ebp-0x40c]
c7 00 63 61 6c 63   mov     DWORD PTR [eax], 0x636c6163
c7 40 04 2e 65 78 65 mov     DWORD PTR [eax+0x4], 0x6578652e
c6 40 08 00         mov     BYTE PTR [eax+0x8], 0x0
c7 44 24 04 03 00 00 mov     DWORD PTR [esp+0x4], 0x3
00
8d 85 f4 fb ff ff   lea     eax, [ebp-0x40c]
89 04 24           mov     DWORD PTR [esp], eax
e8 be 08 00 00      call    401c50 <_WinExec@8>
83 ec 08           sub     esp, 0x8
89 45 f4           mov     DWORD PTR [ebp-0xc], eax
8b 45 f4           mov     eax, DWORD PTR [ebp-0xc]
89 44 24 08        mov     DWORD PTR [esp+0x8], eax
8d 85 f4 fb ff ff   lea     eax, [ebp-0x40c]
89 44 24 04        mov     DWORD PTR [esp+0x4], eax
c7 04 24 4c 30 40 00 mov     DWORD PTR [esp], 0x40304c
e8 43 08 00 00      call    401bf8 <_printf>
c7 04 24 68 30 40 00 mov     DWORD PTR [esp], 0x403068
e8 37 08 00 00      call    401bf8 <_printf>
```

0x401bf8:printf

0x401bf0:put

0x401c50:winexec

为什么需要重定位？

❑ Imagebase

- 0x00600000

❑ 数据VA

- 0x403024、0x40304c
- 0x404068

❑ 函数VA

- 0x406134(puts)
- 0x406130(sprintf)
- 0x4060f0(WinExec)

```
c7 04 24 24 30 40 00 mov     DWORD PTR [esp], 0x403024
e8 8b 08 00 00 call    401bf0 <puts>
8d 85 f4 fb ff ff lea     eax, [ebp-0x40c]
c7 00 63 61 6c 63 mov     DWORD PTR [eax], 0x636c6163
c7 40 04 2e 65 78 65 mov     DWORD PTR [eax+0x4], 0x6578652e
c6 40 08 00 mov     BYTE PTR [eax+0x8], 0x0
c7 44 24 04 03 00 00 mov     DWORD PTR [esp+0x4], 0x3
00
8d 85 f4 fb ff ff lea     eax, [ebp-0x40c]
89 04 24 mov     DWORD PTR [esp], eax
e8 be 08 00 00 call    401c50 <_WinExec@8>
83 ec 08 sub     esp, 0x8
89 45 f4 mov     DWORD PTR [ebp-0xc], eax
8b 45 f4 mov     eax, DWORD PTR [ebp-0xc]
89 44 24 08 mov     DWORD PTR [esp+0x8], eax
8d 85 f4 fb ff ff lea     eax, [ebp-0x40c]
89 44 24 04 mov     DWORD PTR [esp+0x4], eax
c7 04 24 4c 30 40 00 mov     DWORD PTR [esp], 0x40304c
e8 43 08 00 00 call    401bf8 <_printf>
c7 04 24 68 30 40 00 mov     DWORD PTR [esp], 0x403068
e8 37 08 00 00 call    401bf8 <_printf>
```


IMAGE_BASE_RELOCATION结构

顺序	名字	大小（字节）	描述
1	VirtualAddress	4	重定位数据开始的RVA地址
2	SizeofBlock	4	本结构的大小
3	TypeOffset[]	不定	重定位数组，数组每项占两字节

❑ **VirtualAddress:** 是一个4KB（一页）的边界。该值加上后面TypeOffset数组的成员便得到了需要重定位数据的地址。

❑ **SizeBlock:** 为这一结构块的大小。该大小减去前两项的字节数8便得到第3项的大小，再除2即得到定位项的个数。

重定位表的解析

□ 页面的定位项:

```
struct TypeOffset
```

```
{
```

```
    WORD Offset : 12; //代表了页面中重定位地址的偏移量。
```

```
    WORD Type : 4; //代表了所需要的重定位类型
```

```
};
```

□ 重定位的类型:

- **MAGE_REL_BASED_HIGHLOW (3)**。将delta添加到偏移位置的32位字段上。

实例：kernel32.dll

PEview - E:\教学\kernel32.dll

File View Go Help

kernel32.dll

- IMAGE_DOS_HEADER
- MS-DOS Stub Program
- IMAGE_NT_HEADERS
- IMAGE_SECTION_HEADER
- IMAGE_SECTION_HEADER
- IMAGE_SECTION_HEADER
- SECTION .text
- SECTION .data
- SECTION .rsrc
- SECTION .reloc
 - IMAGE_BASE_RELOCATION

pFile	Data	Description	Value
00117442	3F80	Type RVA	00055F80 IMAGE_REL_BASED_HIGHLOW
00117444	3F8B	Type RVA	00055F8B IMAGE_REL_BASED_HIGHLOW
00117446	0000	Type RVA	
00117448	00056000	RVA of Block	
0011744C	00000028	Size of Block	
00117450	32EB	Type RVA	000562EB IMAGE_REL_BASED_HIGHLOW
00117452	3320	Type RVA	00056320 IMAGE_REL_BASED_HIGHLOW
00117454	3327	Type RVA	00056327 IMAGE_REL_BASED_HIGHLOW
00117456	3378	Type RVA	00056678 IMAGE_REL_BASED_HIGHLOW
00117458	379B	Type RVA	0005679B IMAGE_REL_BASED_HIGHLOW
0011745A	37CC	Type RVA	000567CC IMAGE_REL_BASED_HIGHLOW
0011745C	380C	Type RVA	0005680C IMAGE_REL_BASED_HIGHLOW
0011745E	381F	Type RVA	0005681F IMAGE_REL_BASED_HIGHLOW
00117460	3844	Type RVA	00056844 IMAGE_REL_BASED_HIGHLOW
00117462	3A14	Type RVA	00056A14 IMAGE_REL_BASED_HIGHLOW
00117464	3C3D	Type RVA	00056C3D IMAGE_REL_BASED_HIGHLOW
00117466	3D4E	Type RVA	00056D4E IMAGE_REL_BASED_HIGHLOW
00117468	3E9F	Type RVA	00056E9F IMAGE_REL_BASED_HIGHLOW
0011746A	3ED1	Type RVA	00056ED1 IMAGE_REL_BASED_HIGHLOW
0011746C	3FA5	Type RVA	00056FA5 IMAGE_REL_BASED_HIGHLOW
0011746E	0000	Type RVA	
00117470	00057000	RVA of Block	
00117474	00000074	Size of Block	
00117478	3062	Type RVA	00057062 IMAGE_REL_BASED_HIGHLOW
0011747A	30B6	Type RVA	000570B6 IMAGE_REL_BASED_HIGHLOW
0011747C	318B	Type RVA	0005718B IMAGE_REL_BASED_HIGHLOW
0011747E	3212	Type RVA	00057212 IMAGE_REL_BASED_HIGHLOW
00117480	32B2	Type RVA	000572B2 IMAGE_REL_BASED_HIGHLOW

思考题

- 如果手工修改了PE头部的ImageBase值，该程序会产生什么错误？
- 希望程序能继续运行，如何修改该程序的PE文件？