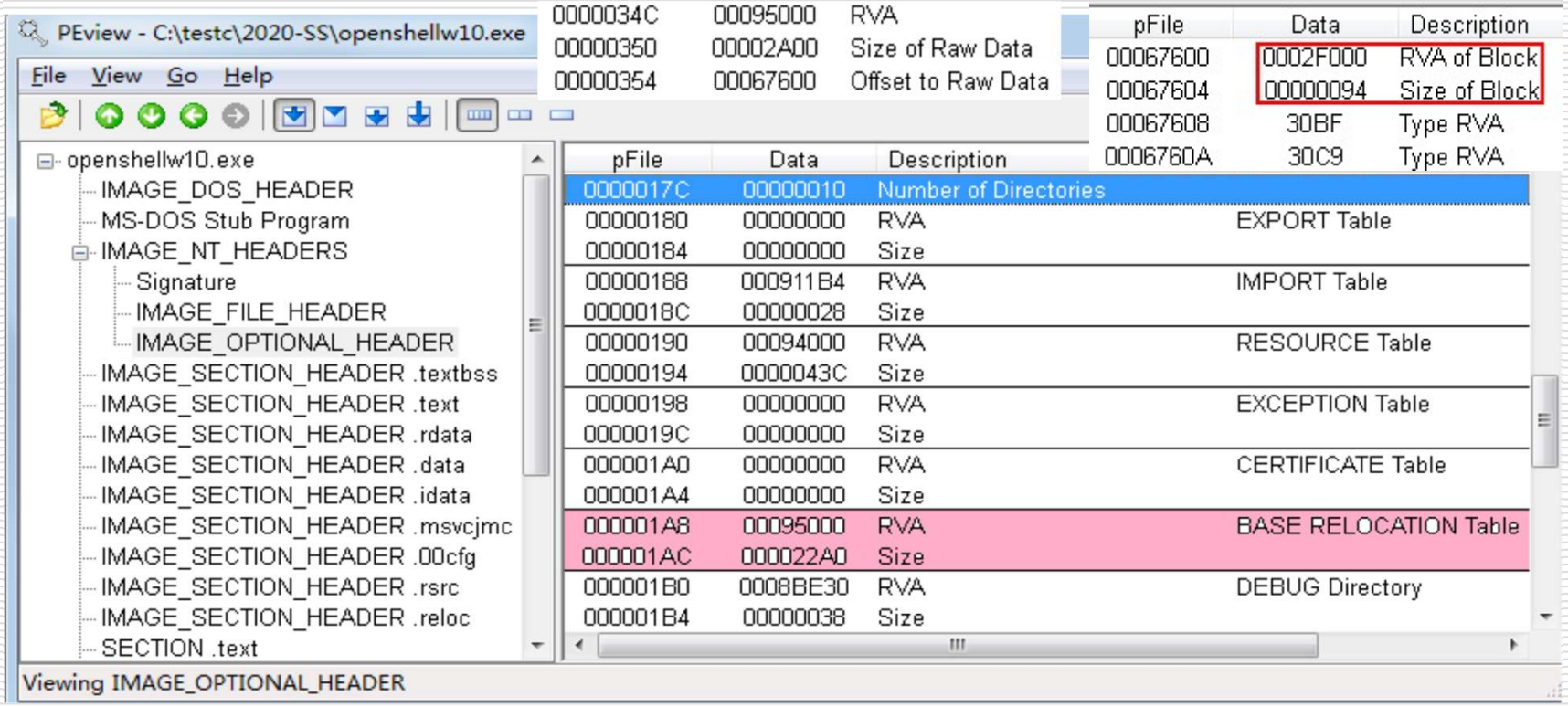
3.8 重定位表

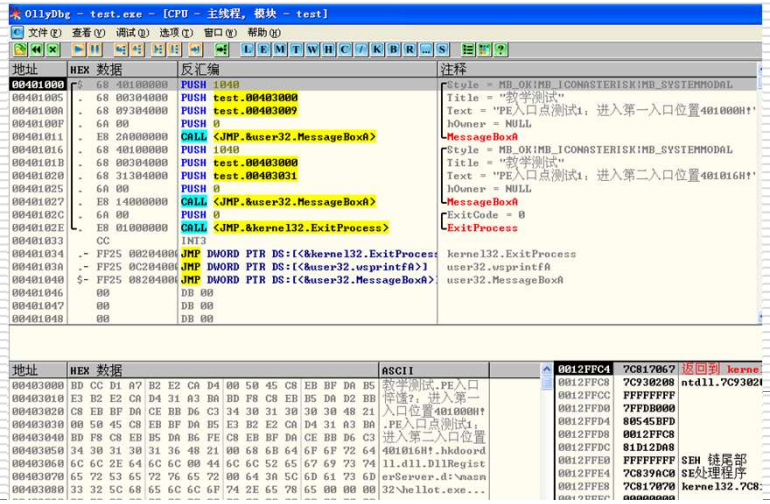
- 重定位表在哪里
- 为什么需要重定位表
- 重定位表的解析
  - IMAGE_BASE_RELOCATION

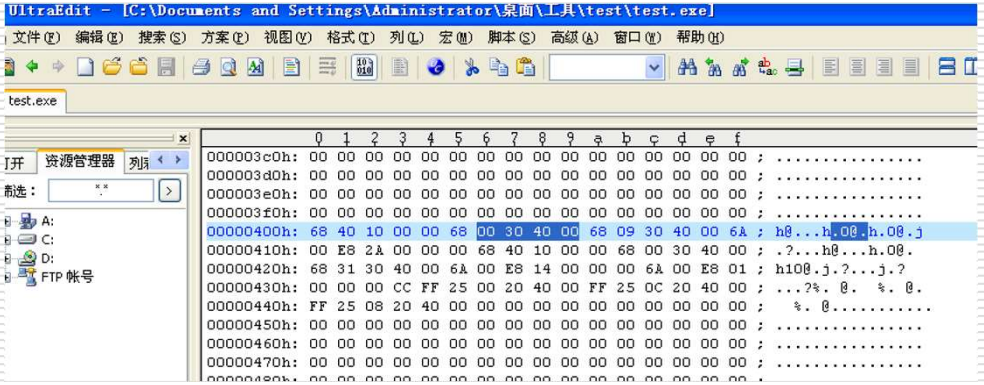
重定位表在哪里


PEview - C:\testc\2020-SS\openshellw10.exe
File View Go Help
openshellw10.exe
IMAGE_DOS_HEADER
MS-DOS Stub Program
IMAGE_NT_HEADERS
Signature
IMAGE_FILE_HEADER
IMAGE_OPTIONAL_HEADER
IMAGE_SECTION_HEADER .textbss
IMAGE_SECTION_HEADER .text
IMAGE_SECTION_HEADER .rdata
IMAGE_SECTION_HEADER .data
IMAGE_SECTION_HEADER .idata
IMAGE_SECTION_HEADER .msvcjmc
IMAGE_SECTION_HEADER .00cfg
IMAGE_SECTION_HEADER .rsrc
IMAGE_SECTION_HEADER .reloc
SECTION .text
0000034C 00095000 RVA
00000350 00002A00 Size of Raw Data
00000354 00067600 Offset to Raw Data
pFile Data Description
00067600 0002F000 RVA of Block
00067604 00000094 Size of Block
00067608 30BF Type RVA
0006760A 30C9 Type RVA
pFile Data Description
0000017C 00000010 Number of Directories
00000180 00000000 RVA EXPORT Table
00000184 00000000 Size
00000188 000911B4 RVA IMPORT Table
0000018C 00000028 Size
00000190 00094000 RVA RESOURCE Table
00000194 0000043C Size
00000198 00000000 RVA EXCEPTION Table
0000019C 00000000 Size
000001A0 00000000 RVA CERTIFICATE Table
000001A4 00000000 Size
000001A8 00095000 RVA BASE RELOCATION Table
000001AC 000022A0 Size
000001B0 0008BE30 RVA DEBUG Directory
000001B4 00000038 Size
Viewing IMAGE_OPTIONAL_HEADER

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。





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    DWORD PTR ds:0x406134
401bf6:    90                      nop
401bf7:    90                      nop
00401bf8 <_printf>:
401bf8:ff 25 30 61 40 00  jmp    DWORD PTR ds:0x406130
401bfe:    90                      nop
401bff:    90                      nop
00401c50 <_WinExec@8>:
401c50:ff 25 f0 60 40 00  jmp    DWORD 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(printf)
  - 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	重定项位数组，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。

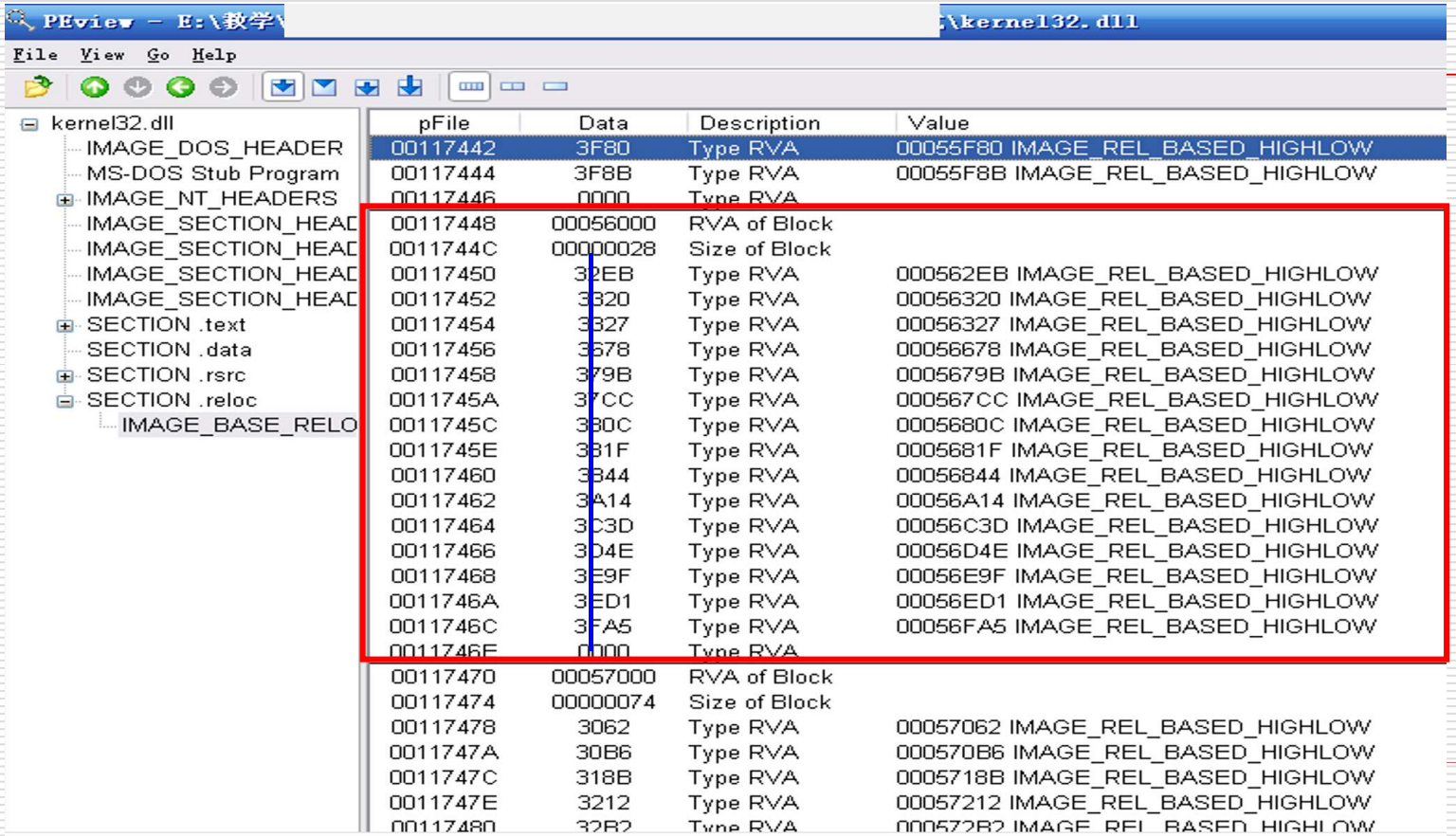
重定位表的解析

- 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



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？