目录

[13. PE文件结构 1](#_Toc484161521)

[13.1 PE头结构 3](#_Toc484161522)

[13.1.1 DOS头 3](#_Toc484161523)

[13.1.2 虚拟地址和文件偏移地址的换算（RVA⬄RAW） 3](#_Toc484161524)

[13.2 IAT（导入地址表） 3](#_Toc484161525)

[13.3 notepad.exe 练习 4](#_Toc484161526)

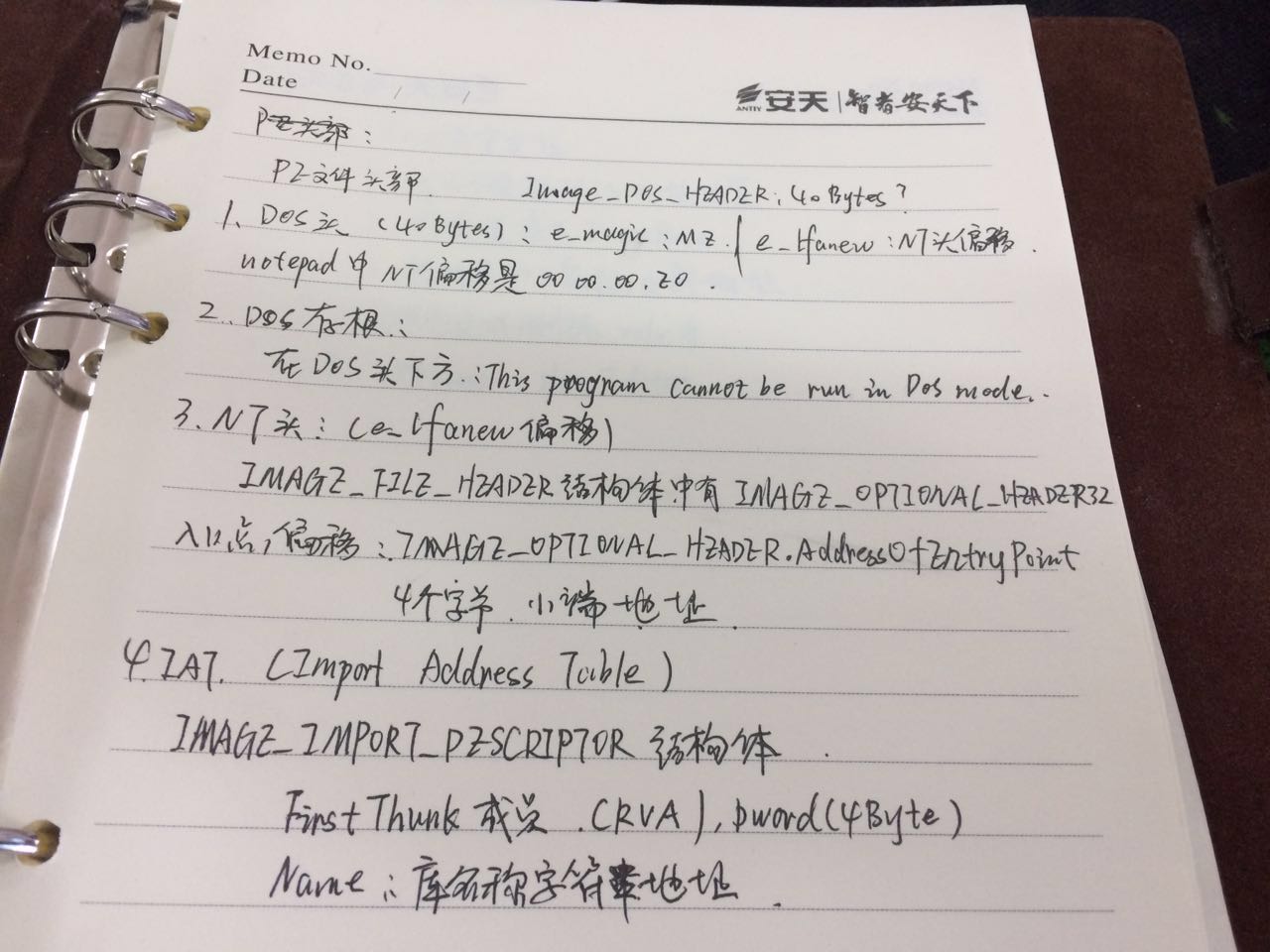
[14. 运行时压缩（加壳） 5](#_Toc484161527)

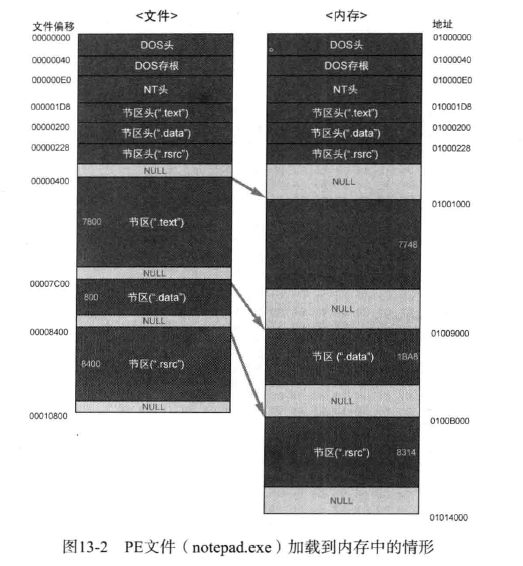
[15. UPX调试脱壳 5](#_Toc484161528)

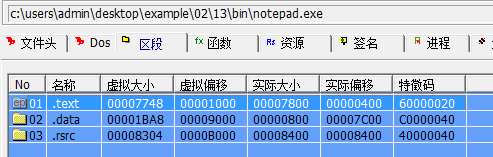
[15.1 跟踪代码和OEP定律 5](#_Toc484161529)

# PE文件结构

两图为notepad.exe的文件结构







名词解释：

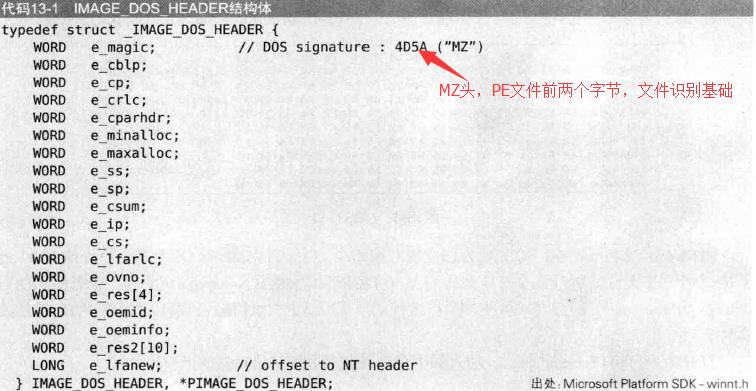
VA：进程虚拟内存的绝对地址

RVA：相对虚拟地址，从基准位置ImageBase开始的相对地址

VA = RVA + ImageBase

## 13.1 PE头结构

### 13.1.1 DOS头



DOS占用40个字节

### 13.1.2 虚拟地址和文件偏移地址的换算（RVA⬄RAW）



1. 查找RVA所在的区段

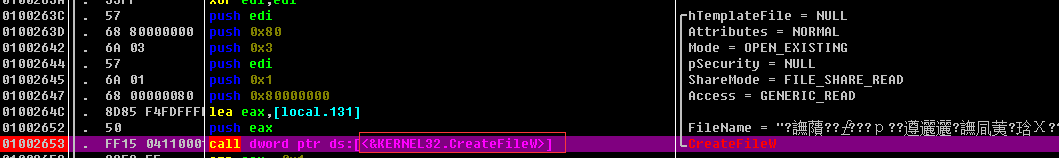
2. 利用公式计算RAW（文件偏移）

RAW – PointerToRawData = RVA – VirtualAddress（虚拟地址相对于虚拟偏移的位置等于文件偏移相当于当前节的文件偏移的位置）。

示例：RVA = 5000，求文件偏移

解： RVA = 5000，在1000和 9000之间，在text节，RAW = 5000 – 1000 + 400 = 4400.

## 13.2 IAT（导入地址表）

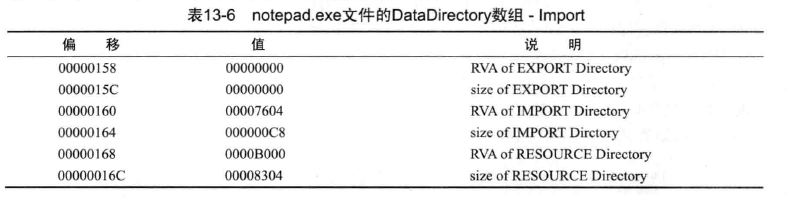


## 13.3 notepad.exe 练习

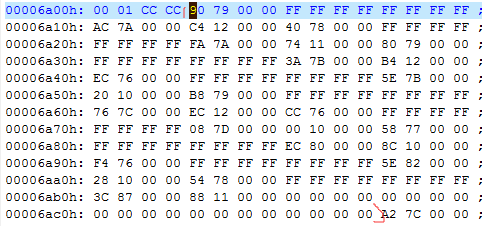
1. 从文件PE体中查找IMAGE\_OPTIONAL\_HEADER32.DataDirectory[1].

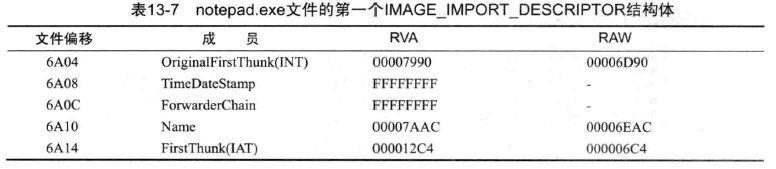
VirtualAddress（结构体数组的起始地址）。



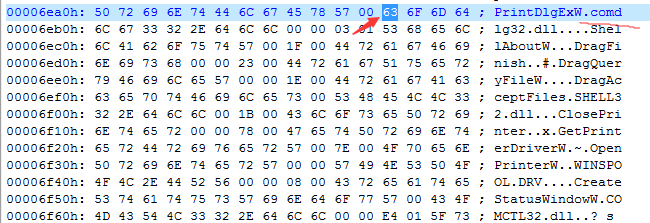


第一个字节是RVA，第二个字节是字节数。RVA = 00007604，size = 0xc8 。RAW = RVA – 1000 + 400（详见之前的公式） = 6A04





Name是导入函数的名称。 RVA ：7AAC =》RAW ：6EAC



# 运行时压缩（加壳）

UPX加壳特点：

1. PE头大小不变
2. 节名称有变化：.text => UPX0 | .data => UPX1
3. UPX0节的RawDataSize = 0
4. EP在第二个节区
5. .rsrc节大小几乎没有变化

# UPX调试脱壳

PUSHAD指令：

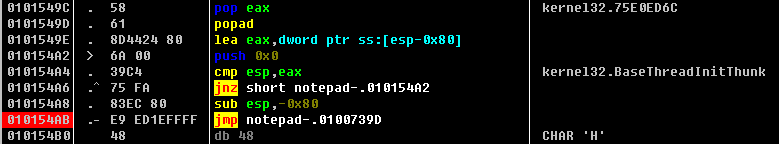
将EAX-EDI寄存器的值保存到栈中。

## 跟踪代码和OEP定律



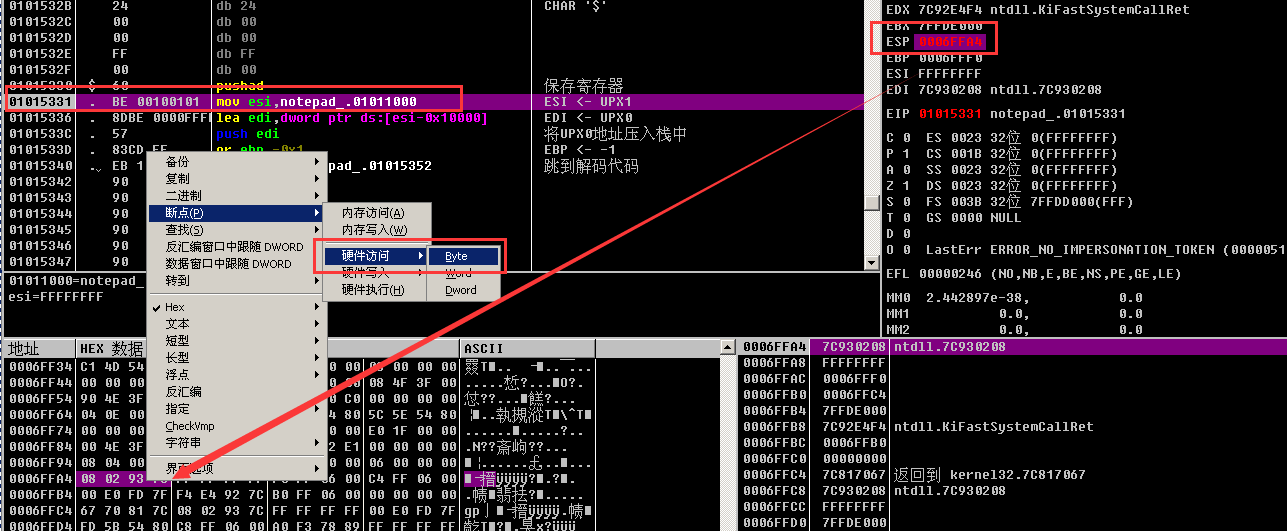
UPX快速脱壳方法：

1. popad 后面的jmp指令下断点



1. 硬件断点（但是没有看明白）

在数据窗口找到执行完pushad后在esp对应地址，然后选择该地址开始的1字节，右键设置硬件断点（或者按书p133那样设置硬件断点均可）。



当程序执行popad时会首先从这个地址上读取数据到edi(pop edi)，此时会触发硬件断点，程序断下。其下方即是跳转到OEP的JMP指令。



不过和软件断点(INT 3)断点不同的是，触发断点的指令执行（popad）完成后程序才会停止运行，即程序会断在popad下一条指令。

OD的硬件断点可以在调试->硬件断点里看到。

## 修复IAT

ImportRCE工具（但是没用明白）。

# 基地址重定位

16、17章好好看

# 删除.reloc节

# Upack PE文件头详细分析

好好看这章

# Upack 调试 查找OEP