

南开大学

汇编语言与逆向技术课程实验报告

实验六：PEViewer



学 院	<u>网络空间安全学院</u>
专 业	<u>信息安全</u>
学 号	<u>2313546</u>
姓 名	<u>蒋衲言</u>
班 级	<u>信息安全班</u>

一、实验目的

- 1.熟悉 PE 文件结构；
- 2.使用 Windows API 函数读取文件内容。

二、实验环境

Windows 操作系统，MASM32 编译环境。

三、程序的设计说明和控制流程图

1.设计说明

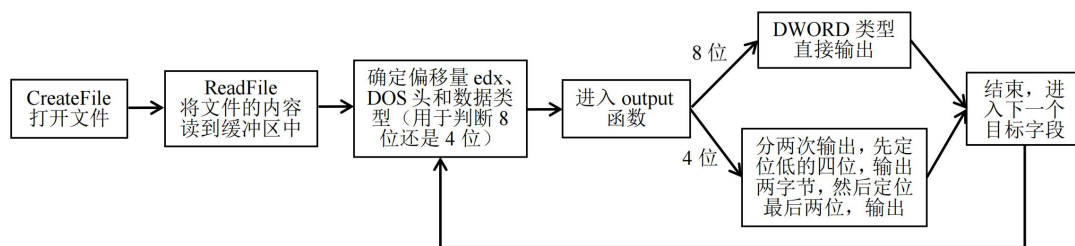
程序首先定义了一系列字符串用于输出提示信息和字段名称，以及缓冲区用于存储文件内容和转换后的十六进制数据。接着，程序通过标准输入获取用户指定的 PE 文件名，使用 `CreateFile` 函数打开该文件，并通过 `ReadFile` 函数读取文件的前 4000 字节到缓冲区中。

程序定义了一个 `Output` 过程，该过程根据传入的偏移量（`edx`）、数据类型大小（`ecx`，区分是 `DWORD` 还是 `WORD`）以及是否处理 DOS 头或 PE 头（通过 `temp1` 区分），从缓冲区中读取相应的数据，将其转换为十六进制字符串，然后输出到控制台。

主程序部分（`start` 标签下）依次输出 DOS 头和部分 PE 头的信息，包括 DOS 头的 `e_magic`、`e_lfanew`，PE 头的 `Signature`、`NumberOfSections`、`TimeDateStamp`、`Characteristics`、`AddressOfEntryPoint`、`ImageBase`、`SectionAlignment` 和 `FileAlignment` 等字段。

最后，程序关闭文件句柄并退出进程。

2.控制流程图



四、实验内容

- 1.输入 PE 文件的文件名，`peviewer` 程序调用 Windows API 函数，打开指定的 PE 文件；
- 2.从文件的头部开始，读取 `IMAGE_DOS_HEADER` 结构中的 `e_magic` 和 `e_lfanew` 字段的值，按照实验演示的方式输出到命令行窗口；
- 3.继续读取 PE 文件的 `IMAGE_NT_HEADER` 结构中的 `Signature` 字段的值，按照实验演示的方式输出到命令行窗口；
- 4.继续读取 `IMAGE_NT_HEADER` 结构中的 `IMAGE_FILE_HEADER` 结构，从中读取字段 `NumberOfSections`、`TimeDateStamp`、`Characteristics` 的值，按照实验演示的方式输出到命令行窗口；
- 5.继续读取 `IMAGE_NT_HEADER` 结构中的 `IMAGE_OPTIONAL_HEADER` 结构，从中读取字段 `AddressOfEntryPoint`、`ImageBase`、`SectionAlignment`、`FileAlignment` 的值，按照实验演示的方式输出到命令行窗口。

五、代码解释

```
.386
.model flat,stdcall
option casemap :none
include \masm32\include\windows.inc
include \masm32\include\masm32.inc
include \masm32\macros\macros.asm
include \masm32\include\kernel32.inc
```

```

includelib \masm32\lib\masm32.lib
includelib \masm32\lib\kernel32.lib

.data
;定义一些输出文本
str0 BYTE "Please input a PE file: ",0
str1 BYTE "IMAGE_DOS_HEADER",0Ah,0Dh,0
str2 BYTE "    e_magic:",0
str3 BYTE "    e_lfanew:",0
str4 BYTE "IMAGE_DOS_HEADER",0Ah,0Dh,0
str5 BYTE "    Signature:",0
str6 BYTE "IMAGE_DOS_HEADER",0Ah,0Dh,0
str7 BYTE "    NumberOfSections:",0
str8 BYTE "    TimeDateStamp:",0
str9 BYTE "    Charateristics:",0
str10 BYTE "IMAGE_DOS_HEADER",0Ah,0Dh,0
str11 BYTE "    e_magic:",0
str12 BYTE "    e_lfanew:",0
str13 BYTE "IMAGE_DOS_HEADER",0Ah,0Dh,0
str14 BYTE "    AddressOfEntryPoint:",0
str15 BYTE "    ImageBase:",0
str16 BYTE "    SectionAlignment:",0
str17 BYTE "    FileAlignment:",0

buf3 DWORD 4000 DUP(0)           ;缓冲区指针
buf4 DWORD 4000 DUP(0)           ;文件内容，转成十六进制存储
buf5 WORD 4000 DUP(0)            ;存储后四位
file BYTE 20 DUP(0),0            ;文件名
hfile DWORD 0,0                  ;文件句柄
endl BYTE 0Ah,0Dh,0             ;换行
temp DWORD 0,0                   ;储存 ecx 的值（是 4 位还是 8 位）
temp1 DWORD 0,0                  ;定位指针的初始位置，是 DOS 头还是 PE 头

.code
;Output 过程，用于输出
Output PROC
    mov esi,OFFSET buf3          ;读取相应位置的内容并转化成二进制
    add esi,edx                   ;把偏移量加上
    add esi,temp1                 ;DOS 头还是 NT 头
    mov eax,DWORD PTR[esi]        ;将最终定位的地址赋给 eax 寄存器
    mov ebx,eax
    invoke dw2hex,eax,addr buf4   ;将 eax 转换为 16 进制后存入 buf4 中
    mov ecx,temp
    .if ecx==8                    ;如果查表结果是 DWORD 型，直接输出
        invoke StdOut,addr buf4
    .else                        ;不是 DWORD 型，就是四位，按照分两次的方法进行输出
        mov ax,WORD PTR [buf4+4]
        mov buf5,ax
        invoke StdOut,addr buf5  ;先定位到低四位，输出两个字节
    .endif
Output ENDP

```

```

mov ax,WORD PTR [buf4+6]
mov buf5,ax
invoke StdOut,addr buf5           ;再定位到后两位，再输出两个字节
.endif
invoke StdOut,addr endl
ret

```

Output ENDP

start:

```

invoke StdOut,addr str0
invoke StdIn,addr file,20         ;输入 exe 文件名称

;打开文件
invoke CreateFile,addr file,\
                                GENERIC_READ,\
                                FILE_SHARE_READ,\
                                0,\
                                OPEN_EXISTING,\
                                FILE_ATTRIBUTE_ARCHIVE,\
                                0

;保存文件句柄
mov hfile,eax
invoke SetFilePointer,hfile,0,0,FILE_BEGIN
invoke ReadFile,hfile,addr buf3,4000,0,0
mov esi,OFFSET buf3

;文件入口
invoke StdOut,addr str1
invoke StdOut,addr str2

```

;下面就是根据表格定位指针，然后带入 Output 过程

```

mov edx,0
mov temp1,edx
mov ecx,4
mov temp,ecx
invoke Output

invoke StdOut,addr str3

mov edx,3ch
mov ecx,8
mov temp,ecx
invoke Output

invoke StdOut,addr str4
invoke StdOut,addr str5
mov temp1,ebx

mov edx,0

```

```
invoke Output

invoke StdOut,addr str6
invoke StdOut,addr str7

mov edx,6h
mov ecx,4
mov temp,ecx
invoke Output

invoke StdOut,addr str8

mov edx,8h
mov ecx,8
mov temp,ecx
invoke Output

invoke StdOut,addr str9

mov edx,16h
mov ecx,4
mov temp,ecx
invoke Output

invoke StdOut,addr str13
invoke StdOut,addr str14

mov edx,28h
mov ecx,8
mov temp,ecx
invoke Output

invoke StdOut,addr str15

mov edx,34h
invoke Output

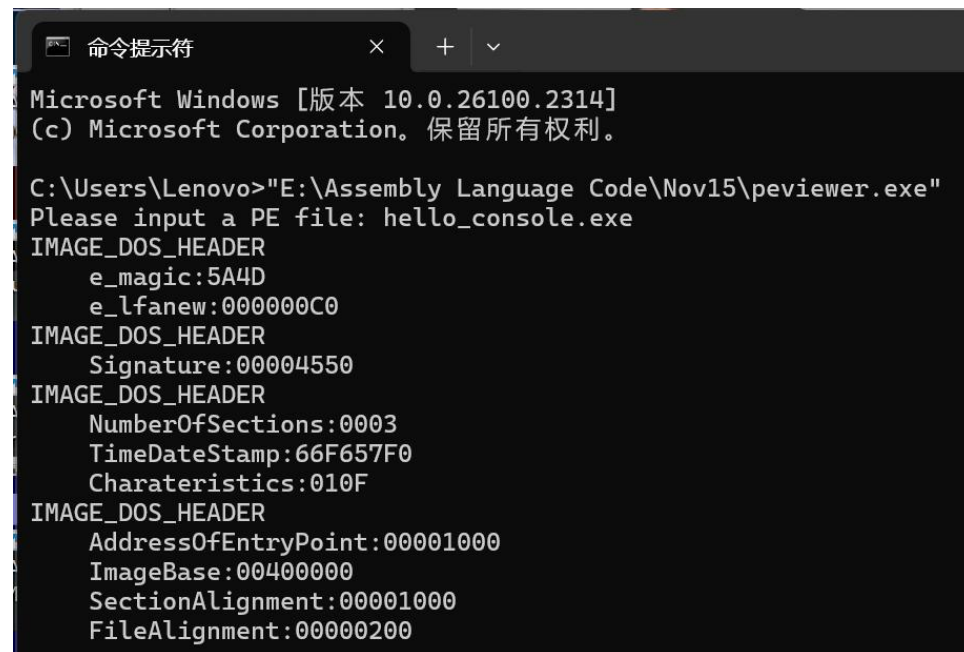
invoke StdOut,addr str16

mov edx,38h
invoke Output

invoke StdOut,addr str17

mov edx,3ch
invoke Output
invoke CloseHandle,hfile
invoke ExitProcess,0
end start
```

六、运行结果



```
命令提示符
Microsoft Windows [版本 10.0.26100.2314]
(c) Microsoft Corporation。保留所有权利。

C:\Users\Lenovo>"E:\Assembly Language Code\Nov15\peviewer.exe"
Please input a PE file: hello_console.exe
IMAGE_DOS_HEADER
    e_magic:5A4D
    e_lfanew:000000C0
IMAGE_DOS_HEADER
    Signature:00004550
IMAGE_DOS_HEADER
    NumberOfSections:0003
    TimeDateStamp:66F657F0
    Characteristics:010F
IMAGE_DOS_HEADER
    AddressOfEntryPoint:00001000
    ImageBase:00400000
    SectionAlignment:00001000
    FileAlignment:00000200
```