

**汇编语言与逆向技术**

**课程实验报告**

**实验三：peviewer**

****

学 院 网络空间安全

专 业 信息安全

学 号 2212998

姓 名 胡博浩

班 级­­­­ 2022级信息安全

1. **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
2. **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，从中读取出字段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；
3. **peviewer程序的设计说明和控制流图**

1、设计说明：

（1）读一个文件用到的Windows API函数有CreateFile、SetFilePointer、ReadFile、CloseHandle：

①CreateFile：打开文件，返回一个新的文件句柄；

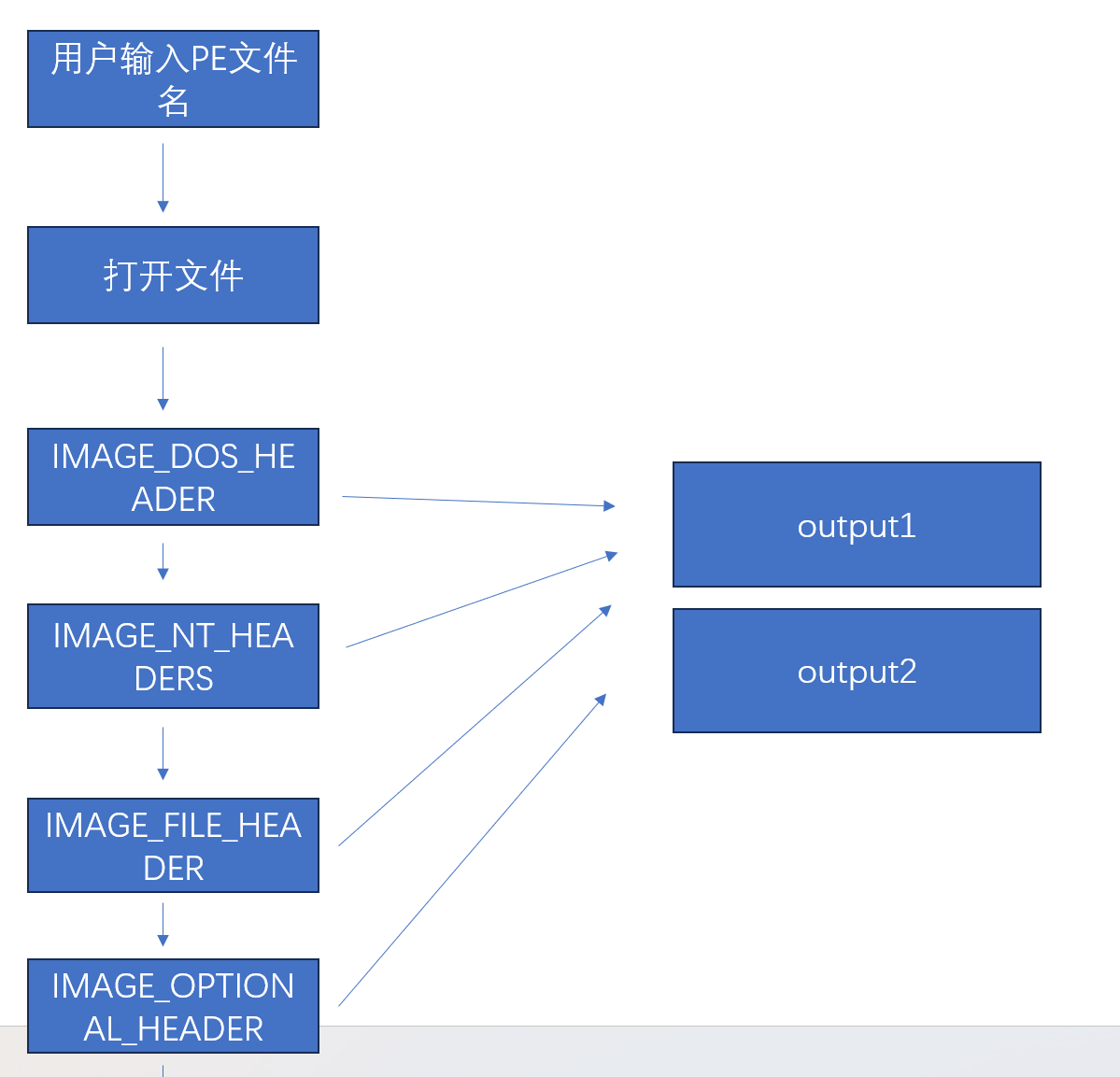
②SetFilePointer：在一个文件中设置当前的读写位置，用于设置文件指针，从文件头或文件尾开始读取文件，该实验是从文件头开始；

③ReadFile：从文件中读出数据（读多少数据，读到什么位置）；

④CloseHandle：文件读取完毕后，用于关闭文件；

（2）通过过程dw2hex,可以将寄存器中的值转化为字符串；

2、控制流图



1. **实验步骤**

1. peviewer.asm的源代码和注释

.386

.model flat, stdcall

option casemap: none

include \masm32\include\windows.inc

include \masm32\include\kernel32.inc

include \masm32\include\masm32.inc

includelib \masm32\lib\kernel32.lib

includelib \masm32\lib\masm32.lib

.data

FileName BYTE 100 DUP(0), 0 ; 输入的文件名

str1 BYTE "Please input a PE file: ", 0 ; 用户提示输入PE文件名

str2 BYTE "IMAGE\_DOS\_HEADER", 0Ah, 0Dh, 0 ; DOS头信息，换行

str3 BYTE " e\_magic:", 0 ; e\_magic字段说明

str4 BYTE " e\_lfanew:", 0 ; e\_lfanew字段说明

str5 BYTE "IMAGE\_NT\_HEADERS", 0Ah, 0Dh, 0 ; NT头信息，换行

str6 BYTE " Signature:", 0 ; Signature字段说明

str7 BYTE "IMAGE\_FILE\_HEADER", 0Ah, 0Dh, 0 ; 文件头信息，换行

str8 BYTE " NumberOfSections:", 0 ; NumberOfSections字段说明

str9 BYTE " TimeDateStamp:", 0 ; TimeDateStamp字段说明

str10 BYTE " Characteristics:", 0 ; Characteristics字段说明

str11 BYTE "IMAGE\_OPTINAL\_HEADER", 0Ah, 0Dh, 0 ; 可选头信息，换行

str12 BYTE " AddressOfEntryPoint:", 0 ; AddressOfEntryPoint字段说明

str13 BYTE " ImageBase:", 0 ; ImageBase字段说明

str14 BYTE " SectionAlignment:", 0 ; SectionAlignment字段说明

str15 BYTE " FileAlignment:", 0 ; FileAlignment字段说明

fileHandle DWORD 0 ; 文件句柄

data1 WORD 1 DUP(0) ; 用于存储WORD数据

data2 DWORD 1 DUP(0) ; 用于存储DWORD数据

data3 BYTE 4 DUP(0), 0 ; 用于存储DWORD数据的字符串表示

data4 BYTE 8 DUP(0), 0 ; 用于存储DWORD数据的字符串表示

data5 BYTE 0Dh, 0Ah ; 换行符

.code

start:

invoke StdOut, addr str1 ; 输出提示信息

invoke StdIn, addr FileName, 20 ; 接收文件名

invoke CreateFile, addr FileName, GENERIC\_READ, 0, 0,

OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, 0 ; 调用CreateFile打开文件

mov fileHandle, eax ; 将eax中得到的文件句柄给fileHandle

;IMAGE\_DOS\_HEADER

invoke StdOut, addr str2

invoke StdOut, addr str3

invoke SetFilePointer, fileHandle, 0, 0, FILE\_BEGIN ; 将文件指针初始化

call output2 ; 获取并输出e\_magic

invoke StdOut, addr str4

invoke SetFilePointer, fileHandle, 58, 0, FILE\_CURRENT ; 调整文件指针

call output1 ; 获取并输出e\_lfanew

;IMAGE\_NT\_HEADERS

invoke StdOut, addr str5

invoke StdOut, addr str6 ; 读取Signature，此时ebx中存储了e\_lfanew的值

invoke SetFilePointer, fileHandle, EBX, 0, FILE\_BEGIN ; 调整文件指针

call output1

;IMAGE\_FILE\_HEADER

invoke StdOut, addr str7

invoke StdOut, addr str8

invoke SetFilePointer, fileHandle, 2, 0, FILE\_CURRENT ; 调整文件指针

call output2 ; 读取NumberOfSections

invoke StdOut, addr str9

call output1 ; 读取TimeDateStamp

invoke StdOut, addr str10

invoke SetFilePointer, fileHandle, 10, 0, FILE\_CURRENT; 调整文件指针

call output2 ; 读取Characteristics

;IMAGE\_OPTIONAL\_HEADER

invoke StdOut, addr str11

invoke StdOut, addr str12

invoke SetFilePointer, fileHandle, 16, 0, FILE\_CURRENT; 调整文件指针

call output1 ; 读取AddressOfEntryPoint

invoke StdOut, addr str13

invoke SetFilePointer, fileHandle, 8, 0, FILE\_CURRENT ; 调整文件指针

call output1 ; 读取ImageBase

invoke StdOut, addr str14

call output1 ; 读取SectionAlignment

invoke StdOut, addr str15

call output1 ; 读取FileAlignment

invoke CloseHandle, fileHandle ; 关闭文件

invoke ExitProcess, 0

; 获取数据的两个过程

output1 PROC

invoke ReadFile, fileHandle, addr data2, 4, 0, 0

mov ebx, data2

invoke dw2hex, EBX, addr data4 ; 转化为字符串

invoke StdOut, addr data4

invoke StdOut, addr data5

ret

output1 ENDP

output2 PROC

invoke ReadFile, fileHandle, addr data1, 2, 0, 0

movzx ebx, data1

invoke dw2hex, EBX, addr data3 ; 转化为字符串输出

invoke StdOut, addr data3+4

invoke StdOut, addr data5

ret

output2 ENDP

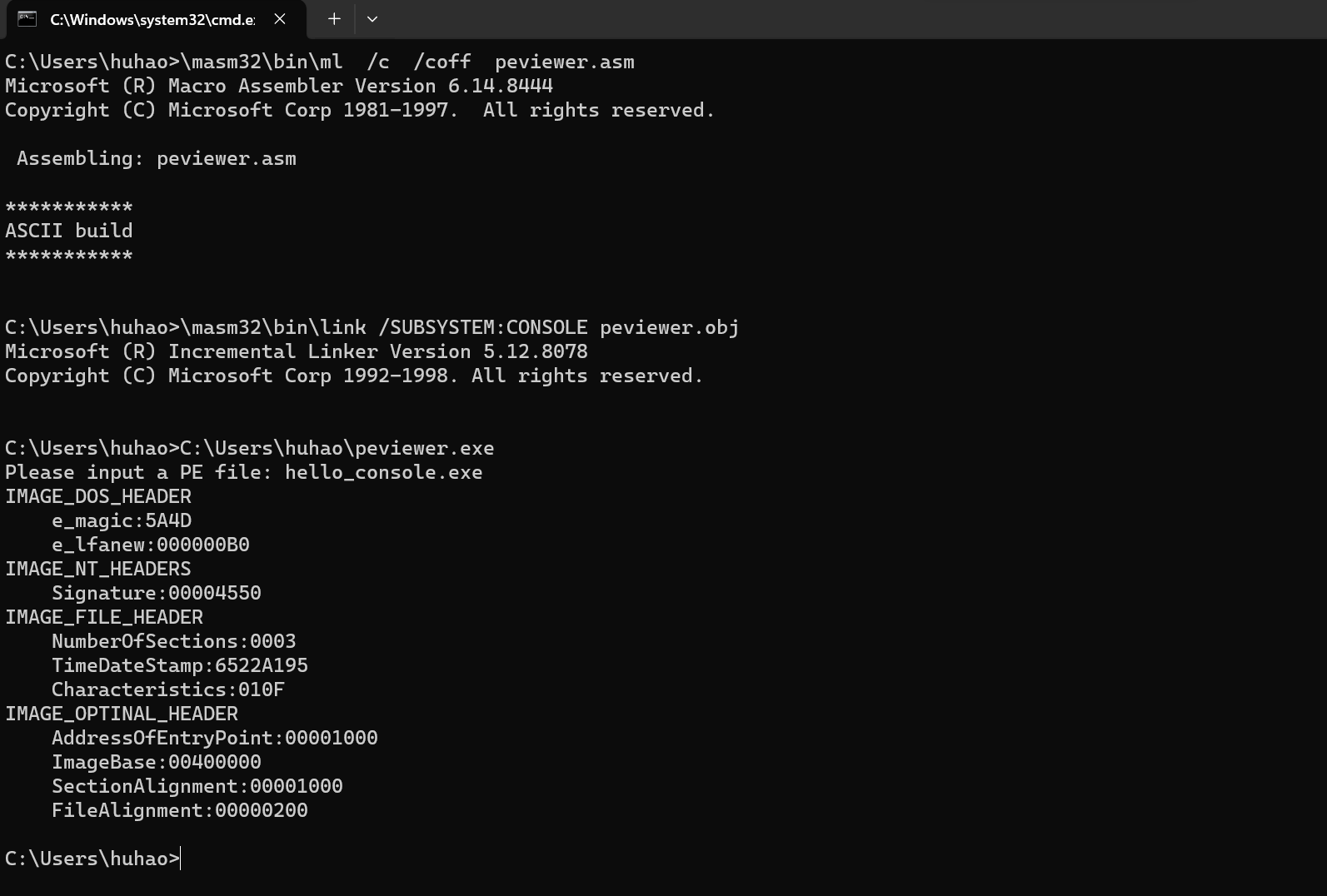
END start

2. 源代码的编译和链接过程说明

1）使用ml将bubble\_sort.asm文件汇编到dec2hex.obj目标文件，编译命令：“\masm32\bin\ml /c /coff peviewer.asm”

2）使用link将目标文件dec2hex.obj链接成dec2hex.exe可执行文件，链接命令：“\masm32\bin\link /SUBSYSTEM:CONSOLE peviewer.obj”

3. peviewer.exe运行截图

****