

# 汇编语言程序设计

## 实验报告

实验题目： 期末大作业——病毒杀毒防毒程序

专 业： 网络空间安全

姓名	学号
CHARLES WIRANTO 李国龙	22920222205358
何唐璜	37220222203606
胡文涛	37220222203617
李涵	37220222203657
洪崧育	22920222202695

实验日期： 2024 年 06 月 01 日

## 一. 实验目的

1. 对学生汇编语言程序实验课编程能力整体考查
2. 考查学生团队合作编程的效率和团队学习能力
3. 具有一定的挑战性

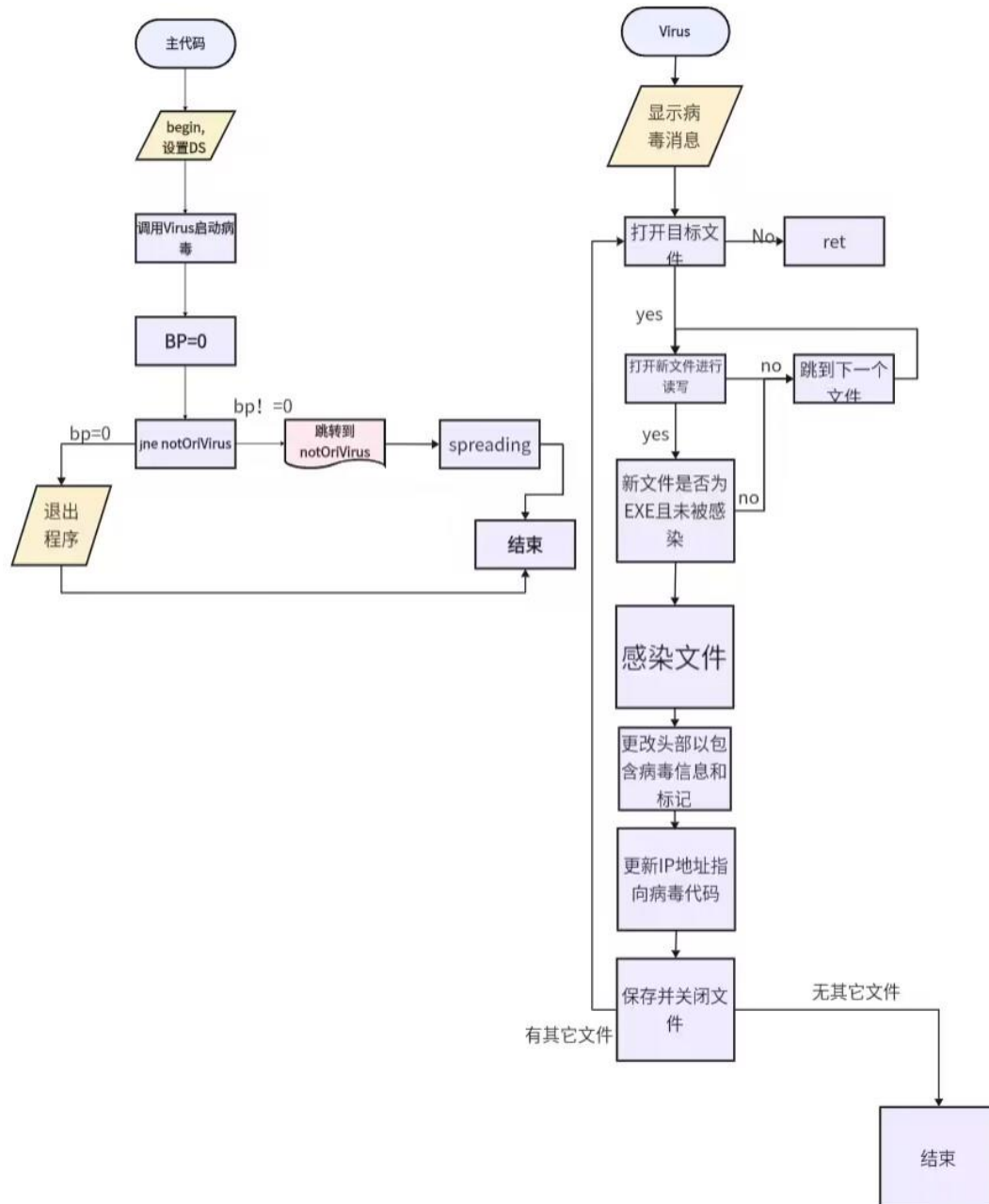
## 二. 实验内容

1. 编写一个病毒传染程序 virus.exe，运行 virus.exe 能将病毒传染给 DOS 当前文件夹所有 EXE 文件，已感染病毒 EXE 文件不重复传染。
2. 编写一个杀病毒程序 kill.exe，运行 kill.exe 能将 DOS 当前文件夹所有感染病毒 EXE 文件杀毒，未感染不杀。
3. 编写一个防病毒程序 defend.exe，运行 defend.exe 后再运行 virus.exe 或感染病毒的 EXE 文件，报警并自动杀毒。无毒文件正常运行。

### 三. 实验步骤以及结果

#### 1. 病毒程序 virus

程序流程图：



## 实现思路及关键:

### 1. 代码布局与初始化

- 段定义与程序入口：通过 `code segment` 定义代码段，使用 `assume cs:code` 设定代码段寄存器，通过跳转避免执行数据段的指令直接进入程序主体。
- 数据段定义：定义了病毒信息字符串、病毒文件名、作者信息、临时存储用的缓冲区、文件查找模式、DOS 数据传输区 (DTA) 等。
- 程序起始点：在 `begin:` 标签处初始化数据段寄存器 `DS`，并调用 `virus` 过程，同时使用 `BP` 寄存器来区分是原始病毒还是被感染文件。

### 2. 病毒核心处理逻辑

- 准备阶段
  - 打印信息：通过调用 `printVirus` 子程序打印病毒信息字符串。
  - 头部备份：将真实文件头部信息备份到 `tmpHead`，以便在感染后恢复原始状态。
  - DOS 中断设置：通过 `INT 21h, AH=1Ah` 设置 DOS DTA，准备遍历文件。
- 文件遍历与感染
  - 文件查找与打开：使用 `INT 21h, AH=4Eh` 查找 `.exe` 文件并尝试打开。
  - 文件读取与检查：读取文件头，确认是否为 EXE 文件（通过“MZ”标识）及是否已被感染（偏移 `2Ah` 处的“CW”标记）。
- 感染操作：
  - 如果是未感染文件，将文件头部修改为已感染状态，同时记录被感染的标志。
  - 计算新的程序入口点，使病毒代码优先执行。
  - 将病毒代码复制到文件末尾，修改文件头以指向病毒代码的起始位置。
  - 保留原始头部信息在文件中特定位置（`2eh ~ 2eh+30h`），以供病毒运行后恢复原程序状态。
- 文件处理：一系列文件操作包括读写、定位、关闭等，通过 `INT 21h` 的各种功能调用来实现。

### 3. 循环与结束

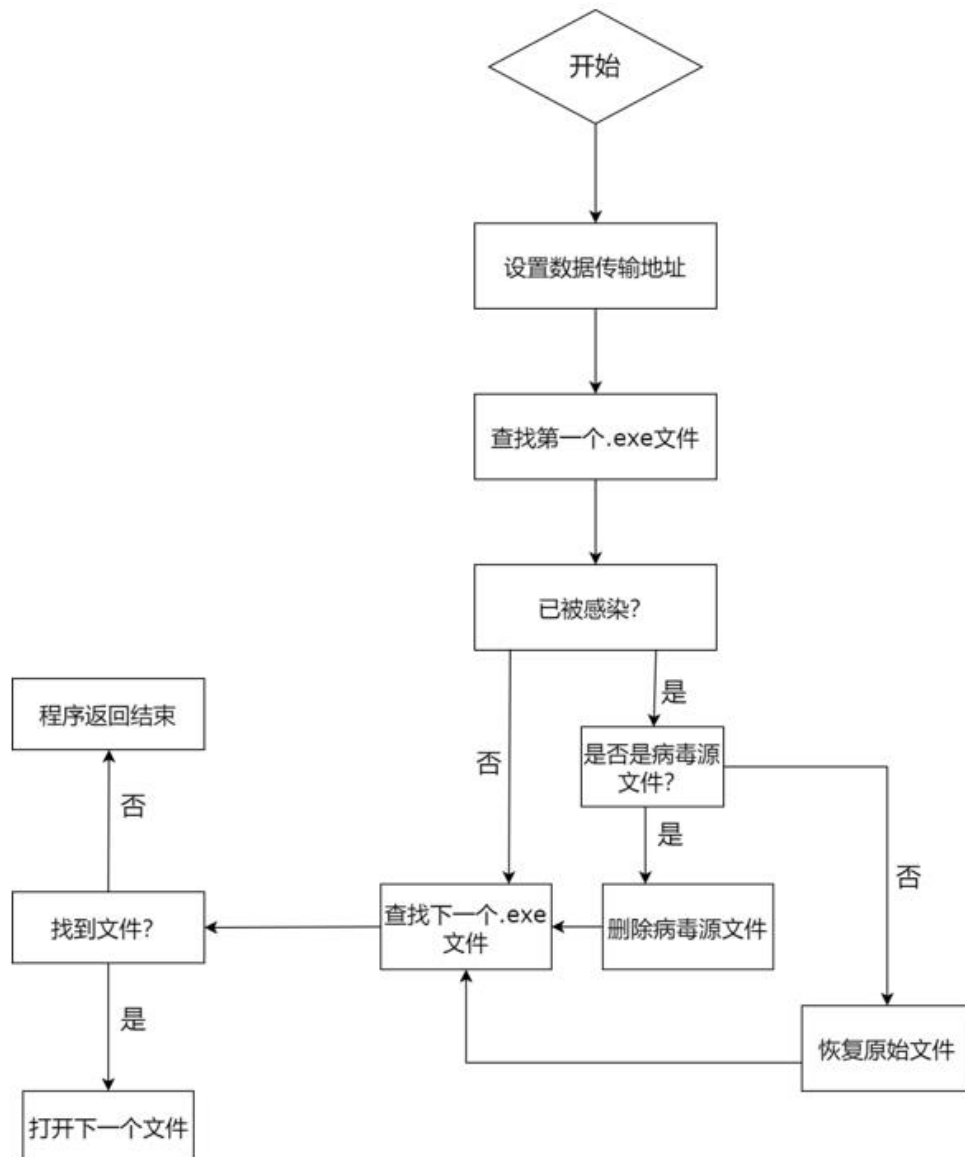
- 文件遍历循环：使用 `INT 21h, AH=4Fh` 查找下一个文件，重复感染过程，直到无更多文件。
- 退出条件：如果当前执行的不是原始病毒（`cmp bp, 0`），程序通过 `INT 21h, AH=4C00h` 正常退出。

## 特殊逻辑与注意事项

- BP 的使用：BP 寄存器，用于判断当前执行环境是原始病毒还是被感染文件，影响程序流程。

## 2. 杀毒程序 killing

程序流程图：



实现思路及关键:

### 1. 初始化

- 段定义与程序入口: `code segment` 定义代码段, `assume cs: code` 设置代码段寄存器。程序从 `start:` 标签开始, 通过短跳转 `jmp short begin` 绕过数据段直接执行程序主体。
- 数据段定义: 包含了程序名称、作者信息、消息字符串、文件查找模式、数据传输区 (DTA)、文件头缓冲区、以及其他用于比较的标记字符串。
- 程序启动: 设置 `DS` 为代码段, 调用 `kill` 过程进行病毒扫描工作。
- 程序退出: 调用 `DOS` 中断 `21h`, `AH=4C00h` 结束程序。

### 2. kill 过程

- 打印信息: 使用 `INT 21h`, `AH=09h` 打印消息 "Killing virus!"。
- 设置 DTA: 通过 `INT 21h`, `AH=1Ah` 准备 `DOS` 用于文件操作的数据传输区。
- 查找文件: 利用 `INT 21h`, `AH=4Eh` 查找第一个匹配 `*.exe` 的文件。如果找到文件, 则跳转至 `kill` 过程进行详细检查和处理。否则, 直接返回, 继续查找下一个文件。

### 3. killing 子过程

- 打开文件: 尝试使用 `INT 21h`, `AH=3Dh` 打开文件。
- 读取文件头: 通过 `INT 21h`, `AH=42h` 定位文件开头, 接着使用 `INT 21h`, `AH=3Fh` 读取前 `140h` 字节的文件头信息到 `head` 缓冲区。
- 检查文件类型和状态:
  - 验证是否为 `EXE` 文件 (检查文件头是否以 "MZ" 开始)。
  - 检查特定偏移处的标记 (`2Ah` 处的 "CW"、`6Ch` 处的 "DF"、`2Ch` 处的 "GL") 以判断文件是否被感染、是否已有防御措施或是否为原始病毒。
- 根据不同状态执行操作:
  - 未感染或已防御 ("DF" 标记): 不做处理, 直接查找下一个文件。
  - 原始病毒 ("GL" 标记): 尝试删除该文件。
  - 被感染 ("CW" 标记): 尝试恢复文件头 (清除病毒标记), 清空被感染区域, 并根据文件头信息调整文件大小以还原原始文件内容。

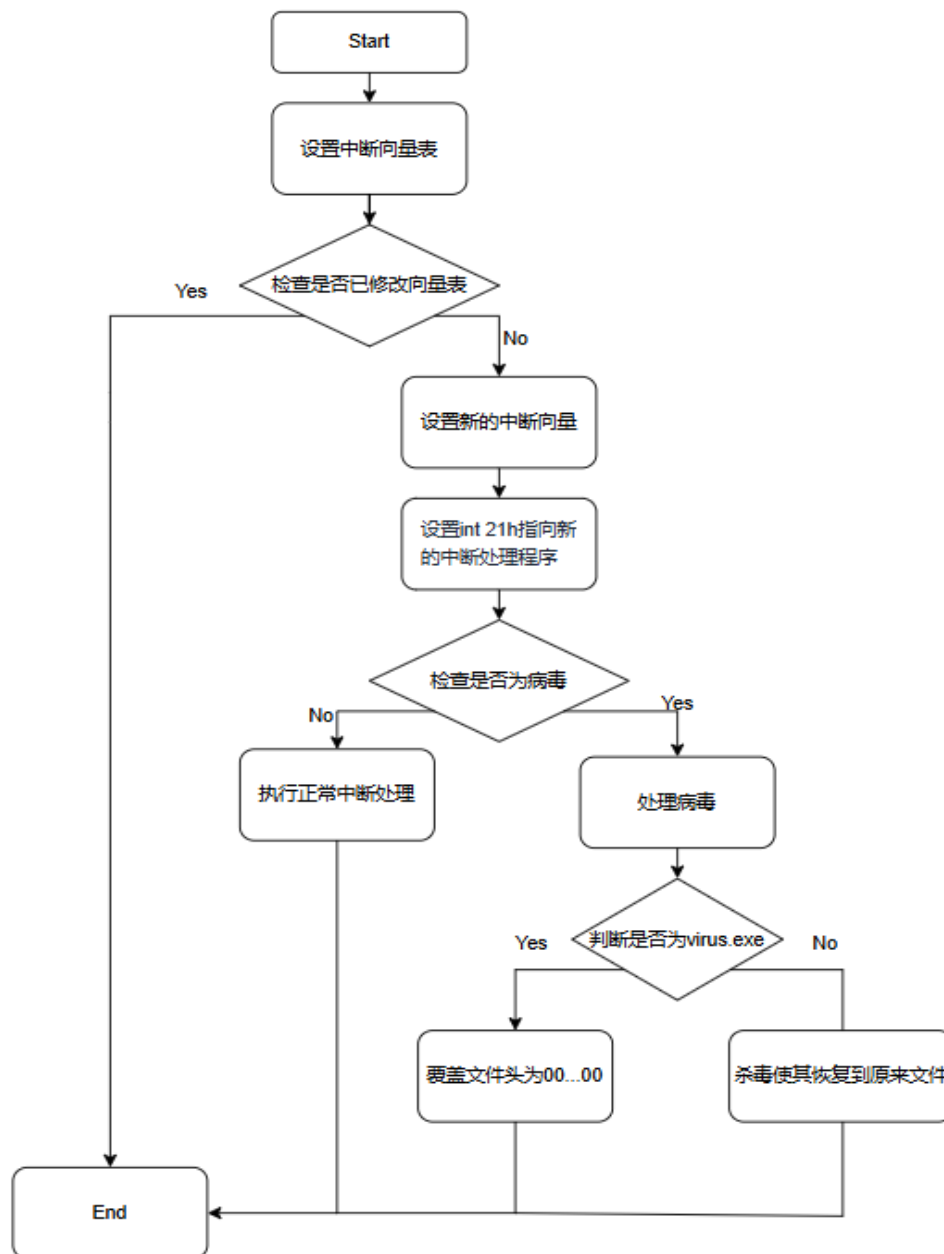
### 4. 辅助功能

- 文件处理后: 关闭当前文件 (`INT 21h`, `AH=3Eh`), 继续查找下一个文件 (`INT 21h`, `AH=4Fh`), 直到没有更多文件为止。

- 计算原始文件大小：根据文件头信息（如偏移 72h、74h 处的值）计算未被病毒感染前的文件大小，以便正确截断文件。

### 3. 防毒程序 defend

程序流程图：



实现思路及关键：

#### 1. 初始化与入口

- 代码段定义：codesg segment 定义了代码段。

- 入口点: start:从 start 标签开始执行, 设置 DS 段寄存器指向代码段, 将新中断处理程序复制到内存低地址, 然后检查和修改中断向量以接管 INT 21h。

## 2. 中断处理接管

- 中断 21h 接管逻辑: 代码复制了新中断处理程序到内存中的低地址 (通常为 0x20:0000h 开始), 并检查当前的 INT 21h 是否已经被修改 (通过检查内存地址 0:000h 的值), 如果未修改则替换 INT 21h 的中断向量为指向新中断处理程序

## 3. 病常规检测与处理

- 检测病毒标记: truestart 标签处, 检测当前执行流中的程序头信息是否包含病毒标记 (在偏移位 2Ah 处为 53H, 即 "CW" "CW"), 如果是则跳转到 virus:处理病毒`分支。
- 正常程序: 如果检测未发现病毒标记, 直接通过 pop 指令恢复寄存器栈, 执行 int 80h 返回。

## 4. 病毒处理逻辑

- virus: 处理: 病毒分支开始, 首先通过 dx 获取 BP 寄存器判断是否指向了有效地址, 用于决定是否打印信息。然后跳转到 kill 标签。
- kill: 移动态修改文件头: 通过 mov ax, 3d0h 等指令尝试打开文件, mov ah, 40h 读取文件头信息到 head 缓冲区, 修改头信息以防止重新感染, mov ah, 40h 写回文件。
- 打印信息: printmes 标签处, 根据状态打印发现病毒信息或已清除信息, 通过 mov ah, 09h 调用 INT 80h 输出。
- 结束: 最后 mov ax, 4c0h 调用 INT 80h 结束程序。



四. 实验结果与分析

完整代码请见附件。

1. 病毒程序 virus

A. 对于正常程序 aa，病毒程序可对其进行感染

名称	修改日期
 AA.EXE	2024/6/5 10:41
 VIRUS.EXE	2024/6/4 16:26

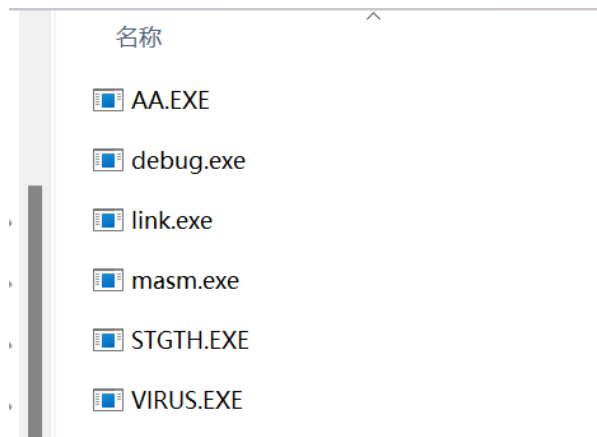
```
C:\VIRUS>cd virus1
C:\VIRUS\VIRUS1>aa
hello, world!
C:\VIRUS\VIRUS1>virus
I am a virus
C:\VIRUS\VIRUS1>aa
I am a virus
hello, world!
```

B. 被感染后的 aa 文件可以传染别的文件 bb

名称	修改日期	类型
 BB.EXE	2024/6/3 20:48	应用程序
 AA.EXE	2024/6/5 10:41	应用程序

```
C:\VIRUS\VIRUS2>bb
hello, world!
C:\VIRUS\VIRUS2>aa
I am a virus
hello, world!
C:\VIRUS\VIRUS2>bb
I am a virus
hello, world!
C:\VIRUS\VIRUS2>
```

### C. 病毒也可以感染 debug 等各种大文件



```
hello, world!

C:\VIRUS\VIRUS3>debug
-q

C:\VIRUS\VIRUS3>link

Microsoft (R) Overlay Linker  Version 3.60
Copyright (C) Microsoft Corp 1983-1987.  All rights reserved.

Object Modules [.OBJ]:
Object Modules [.OBJ]:
LINK : fatal error L1020: no object modules specified

C:\VIRUS\VIRUS3>virus
I am a virus

C:\VIRUS\VIRUS3>bb
Illegal command: bb.

C:\VIRUS\VIRUS3>aa
I am a virus
hello, world!

C:\VIRUS\VIRUS3>_
```

```
C:\VIRUS\VIRUS3>aa
I am a virus
hello, world!

C:\VIRUS\VIRUS3>debug
I am a virus
-q



C:\VIRUS\VIRUS3>link
I am a virus

Microsoft (R) Overlay Linker  Version 3.60
Copyright (C) Microsoft Corp 1983-1987.  All rights reserved.

Run File [!<!FU~.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : fatal error L1093: !<!FU~.OBJ : object not found

C:\VIRUS\VIRUS3>_
```

D. 被感染后的大文件同样可以感染其他文件。

名称	修改日期	类型
 BB.EXE	2024/6/3 20:48	应用程序
 link.exe	2024/6/5 10:51	应用程序

```
C:\VIRUS>cd virus4

C:\VIRUS\VIRUS4>bb
hello, world!

C:\VIRUS\VIRUS4>link
I am a virus

Microsoft (R) Overlay Linker Version 3.60
Copyright (C) Microsoft Corp 1983-1987. All rights reserved.

Run File [!<!FU~.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : fatal error L1093: !<!FU~.OBJ : object not found

C:\VIRUS\VIRUS4>bb
I am a virus
hello, world!
```

E. 被感染后的小文件同样可以感染其他文件包括大文件

```
C:\VIRUS5>aa
hello, world!

C:\VIRUS5>link

Microsoft (R) Overlay Linker Version 3.60
Copyright (C) Microsoft Corp 1983-1987. All rights reserved.

Object Modules [.OBJ]:
Object Modules [.OBJ]:
LINK : fatal error L1020: no object modules specified

C:\VIRUS5>debug
-q

C:\VIRUS5>bb
I am a virus
hello, world!

C:\VIRUS5>debug
I am a virus
-q
```

```
C:\VIRUS5>debug
I am a virus
-q

C:\VIRUS5>link
I am a virus

Microsoft (R) Overlay Linker Version 3.60
Copyright (C) Microsoft Corp 1983-1987. All rights reserved.

Run File [!<!FU~.EXE]:
List File [NUL.MAP]:
Libraries [.LIB]:
LINK : fatal error L1093: !<!FU~.OBJ : object not found

C:\VIRUS5>aa
I am a virus
hello, world!
```

F. 感染性没有重复性（virus.exe 也不会感染自己）

```
C:\VIRUS1>virus
I am a virus

C:\VIRUS1>virus
I am a virus

C:\VIRUS1>aa
I am a virus
hello, world!

C:\VIRUS1>virus
I am a virus

C:\VIRUS1>aa
I am a virus
hello, world!
```



观察已感染病毒文件信息

名称	修改日期	类型
AA.EXE	2024/6/5 10:51	应用程序
debug.exe	2024/6/5 10:51	应用程序
KILLING.EXE	2024/6/5 7:40	应用程序
link.exe	2024/6/5 10:51	应用程序
masm.exe	2024/6/5 10:51	应用程序
STGTH.EXE	2024/6/5 10:51	应用程序
VIRUS.EXE	2024/6/4 16:26	应用程序

```
C:\VIRUS>cd kill1
C:\VIRUS\KILL1>dir
Directory of C:\VIRUS\KILL1\
.                <DIR>                05-06-2024  11:00
..               <DIR>                05-06-2024  10:57
AA              EXE                  1,371 05-06-2024  10:51
DEBUG          EXE                 21,442 05-06-2024  10:51
KILLING        EXE                  1,426 05-06-2024   7:40
LINK           EXE                 65,790 05-06-2024  10:51
MASM           EXE                103,983 05-06-2024  10:51
STGTH          EXE                  1,398 05-06-2024  10:51
VIRUS          EXE                  1,320 04-06-2024  16:26
       7 File(s)                196,730 Bytes.
       2 Dir(s)                 262,111,744 Bytes free.
C:\VIRUS\KILL1>
```

观察经过杀毒后，各文件信息恢复正常

```
C:\VIRUS\KILL1>killing
I am a virus
Killing virus!
Virus detected in file : AA.exe

Virus detected in file : DEBUG.exe

Virus detected in file : KILLING.exe

Virus detected in file : LINK.exe

Virus detected in file : MASM.exe

Virus detected in file : STGTH.exe

virus.exe detected! Deleting!
C:\VIRUS\KILL1>a
```

```
Virus detected in file : MASM.exe

Virus detected in file : STGTH.exe

virus.exe detected! Deleting!

C:\VIRUS\KILL1>dir
Directory of C:\VIRUS\KILL1\
.                <DIR>                05-06-2024 11:03
..               <DIR>                05-06-2024 10:57
AA              EXE                    563 05-06-2024 11:03
DEBUG           EXE                   20,634 05-06-2024 11:03
KILLING         EXE                    1,426 05-06-2024 11:03
LINK            EXE                   64,982 05-06-2024 11:03
MASM            EXE                  103,175 05-06-2024 11:03
STGTH           EXE                     590 05-06-2024 11:03
        6 File(s)                191,370 Bytes.
        2 Dir(s)                262,111,744 Bytes free.

C:\VIRUS\KILL1>
```

上面证明了杀毒不会破坏源文件，杀毒之后文件也会恢复到感染之前的情况。  
下面利用 WinHex 来分析头文件的改变。

00000000	4D 5A D6 01 7F 00 11 00	20 00 6D 03 FF FF 2A 11	MZÖ	m ŷŷ*	MZ	□ □ □ □ □
00000010	00 20 A4 7C 5E C7 00 00	1E 00 00 00 01 00 8F 35	□   ^ Ç	5	□ □ □ □ □ □ □ □	
00000020	00 00 14 00 AB 0F 12 00	AB 0F 10 00 AB 0F 0E 00	« « «		□ □ □ □	
00000030	AB 0F 0C 00 AB 0F 0A 00	AB 0F 08 00 AB 0F 06 00	« « « «		□ □ □ □	
00000040	AB 0F 04 00 AB 0F 04 0D	B6 0E 75 C7 00 00 96 0D	« « ŷ u Ç -		□ □ □ □ □ □	
00000050	B6 0E C4 CC 00 00 66 CD	00 00 6B CF 00 00 C0 D5	ŷ Ä ì f í k ĩ Ä Ö		□ f □ □ □	
00000060	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				
00000070	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				
00000080	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				
00000090	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				
000000A0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				
000000B0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				
000000C0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				
000000D0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				
000000E0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				
000000F0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				
00000100	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				
00000110	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				
00000120	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				
00000130	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				
00000140	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00				

Link.exe 感染之前

00000000	4D 5A F8 00 81 00 11 00	20 00 6D 03 FF FF 2A 11	MZø	m ýý*	MZø	øø
----------	-------------------------	-------------------------	-----	-------	-----	--

## Link.exe 感染之后

00000000	4D 5A D6 01 7F 00 11 00	20 00 6D 03 FF FF 2A 11	MZö	m ýý*	MZ	□ □ □m□□	
00000010	00 20 A4 7C 5E C7 00 00	1E 00 00 00 01 00 8F 35	α ^Ç	5	□ □ □ □□□□□		
00000020	00 00 14 00 AB 0F 12 00	AB 0F 10 00 AB 0F 0E 00	«	«	«	«	□□□ □ □
00000030	AB 0F 0C 00 AB 0F 0A 00	AB 0F 08 00 AB 0F 06 00	«	«	«	«	□ □ □ □
00000040	AB 0F 04 00 AB 0F 04 0D	B6 0E 75 C7 00 00 96 0D	«	«	¶ uÇ	-	□ □ □ □□□
00000050	B6 0E C4 CC 00 00 66 CD	00 00 6B CF 00 00 C0 D5	¶ Äî	fí	kî	Äö	□ f□ □ □
00000060	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					
00000070	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					
00000080	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					
00000090	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					
000000A0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					
000000B0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					
000000C0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					
000000D0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					
000000E0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					
000000F0	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					
00000100	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					
00000110	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					
00000120	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					
00000130	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					
00000140	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00					

## Link.exe 杀毒之后

上面能够证明连头文件不会被破坏，尤其是 Link.exe 文件，因为它本身有 70H 的头部份。



3. 防毒程序 defend

A. defend 不会对正常程序相应。

名称	修改日期	类型
AA.EXE	2024/6/3 20:48	应用程序
BB.EXE	2024/6/3 20:48	应用程序
debug.exe	2024/6/3 20:48	应用程序
DEFEND.EXE	2024/6/5 8:55	应用程序
link.exe	2024/6/3 20:48	应用程序
masm.exe	2024/6/3 20:48	应用程序
VIRUS.EXE	2024/6/4 16:26	应用程序

```
C:\VIRUS\DEFEND1>aa
hello, world!

C:\VIRUS\DEFEND1>defend

C:\VIRUS\DEFEND1>aa
hello, world!

C:\VIRUS\DEFEND1>_
```

B. 可以对已感染文件进行反应，进行防毒杀毒，且防毒后 virus 不会再感染其他文件

名称	修改日期	类型
AA.EXE	2024/6/5 11:09	应用程序
BB.EXE	2024/6/5 11:09	应用程序
debug.exe	2024/6/5 11:10	应用程序
DEFEND.EXE	2024/6/5 11:09	应用程序
link.exe	2024/6/5 11:09	应用程序
masm.exe	2024/6/5 11:09	应用程序
VIRUS.EXE	2024/6/4 16:26	应用程序

```
C:\VIRUS\DEFEND2>aa
hello, world!

C:\VIRUS\DEFEND2>virus
I am a virus

C:\VIRUS\DEFEND2>aa
I am a virus
hello, world!

C:\VIRUS\DEFEND2>debug
I am a virus
-q

C:\VIRUS\DEFEND2>defend
I am a virus

C:\VIRUS\DEFEND2>defend
Virus detected!
Virus has been killed!
C:\VIRUS\DEFEND2>
```

```
C:\VIRUS\DEFEND2>defend
Virus detected!
Virus has been killed!
C:\VIRUS\DEFEND2>defend

C:\VIRUS\DEFEND2>aa
Virus detected!
Virus has been killed!
C:\VIRUS\DEFEND2>aa
hello, world!

C:\VIRUS\DEFEND2>debug
Virus detected!
Virus has been killed!
C:\VIRUS\DEFEND2>debug
-q

C:\VIRUS\DEFEND2>virus
Virus detected!

C:\VIRUS\DEFEND2>virus
Virus detected!
```

上面运行了 defend 三次表示被感染的 defend.exe 运行（包括 defend.exe 源文件的代码）、开始防毒以及杀毒自己的病毒、正常运行 defend.exe。其他部分是为了证明 defend.exe 能够防毒、杀毒。最后运行了 virus.exe，因为前面运行了 defend.exe，所以 virus.exe 不会感染其他文件。

## 五. 实验总结

### 1. 知识点总结:

#### (1)virus:

- 内存段的使用和设置(代码段、数据段)。
- 中断调用(如 int 21h)的用法,用于执行各种 DOS 功能。
- 文件操作,如打开、读取、写入、关闭文件。
- 使用指针(偏移地址)访问内存数据。
- 循环结构的实现。
- 条件判断(比较和跳转)的实现。
- 堆栈的使用,用于保存和恢复数据。
- 使用调用约定(CALL/RET)实现过程。
- 学习它的编写方式对于深入理解 x86 汇编语言、DOS 编程和病毒工作原理很有帮助。

#### (2)Killing:

- 文件操作
  - 打开、读取、写入和关闭文件。
  - 使用 DOS 中断调用执行文件操作。
- 内存操作
  - 使用指针访问内存数据。
  - 复制内存数据。
  - 条件判断和循环结构
- 使用比较和跳转指令实现条件判断。
- 使用循环遍历目录中的文件。
- 程序控制流
  - 使用过程调用(CALL/RET)实现代码模块化。
  - 使用跳转指令控制程序执行流程。
- 数据结构
  - 使用缓冲区存储文件头和其他数据。
  - 使用标记字符串标识不同的文件状态。
- 算术运算
  - 使用乘法和加法计算文件大小。
  - 使用减法和借位处理截断文件大小。

#### (3)Defend:

- 中断向量表的修改:
  - 将新的中断处理程序(newint21start)复制到内存地址 210h 处。
  - 将原始的 INT 21h 中断向量保存到 INT 80h 中断向量。
  - 将 INT 21h 中断向量修改为指向新的中断处理程序(210h)。
- 文件操作:
  - 使用 INT 80h 中断进行文件操作,如打开文件(AH=3Dh)、读取文件

- (AH=3Fh)、写入文件 (AH=40h)、关闭文件 (AH=3Eh) 等。
- 使用文件句柄 (存储在 BX 寄存器) 来识别打开的文件。
- 使用文件指针移动函数 (AH=42h) 来定位文件中的特定位置。
- 内存操作:
  - 使用 DS 寄存器和偏移量来访问内存中的数据。
  - 使用 SI 寄存器作为源数据的指针, DI 寄存器作为目标数据的指针。
  - 使用 REP MOVSB 指令进行内存块的复制。
- 中断和标志位的处理:
  - 使用 STI 和 CLI 指令来开启和关闭中断。
  - 使用 IRET 指令从中断处理程序返回。

## 2. 过程错误总结:

### (1)Virus:

- 内存分配错误: 病毒使用固定的缓冲区大小来存储文件头部和其他数据。如果实际数据大小超过缓冲区大小, 可能会导致缓冲区溢出和不可预测的行为。
- 计算错误: 病毒在计算新的入口点 (IP) 和文件大小时使用了硬编码的值。如果这些值不适用于某些文件, 可能会导致计算错误和感染失败, 文件大小算错了, 导致 overflow, 文件变得很大。

### (2)Killing:

- 删除病毒代码的方式不完整:
  - 在 infectedExe 过程中, 这只删除了文件头部分的病毒代码, 而没有删除附加在文件末尾的病毒代码。
- 缺少必要的出错处理:
  - 代码中缺少一些必要的出错处理, 例如在打开文件失败时, 代码直接跳转到 nextfile, 而没有关闭文件句柄或执行其他清理操作
- 代码结构和可读性有待改进:
  - 虽然代码使用了过程和标签来组织代码, 但是代码的结构和命名可以进一步改进, 以提高可读性和可维护性。

### (3)Defend:

- 没有对文件操作的返回值进行检查, 忽略了可能的错误情况, 如文件不存在、访问被拒绝等。
- 假设被感染的文件具有特定的文件头结构, 但是不同的可执行文件格式可能具有不同的头部结构。
- 没有对可能的异常情况进行适当的处理, 如内存分配失败、磁盘空间不足等。