

恶意代码分析与防治技术实验报告

Lab3

学号： 姓名： 专业： 信息安全

一、 实验环境

1. 已关闭病毒防护的 Windows10
2. VMware+Windows XP

二、 实验工具

Dependency Walker, PEid, Strings, ApateDNS, Process Monitor, Process Explorer, netcat, WireShark, Regshot

三、 实验内容

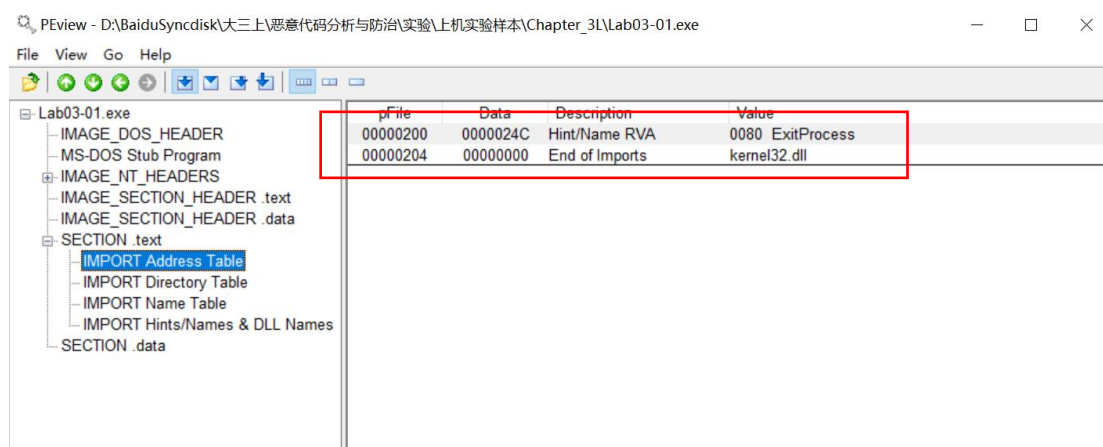
核心思想：先做基本的静态分析（PEvid -> Strings），再做动态分析

Lab 3-1

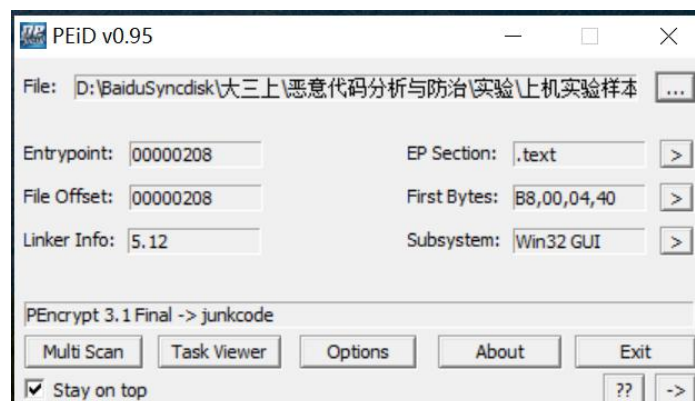
使用动态分析基础技术来分析在 Lab03-01.exe 文件中发现的恶意代码。

实验过程：

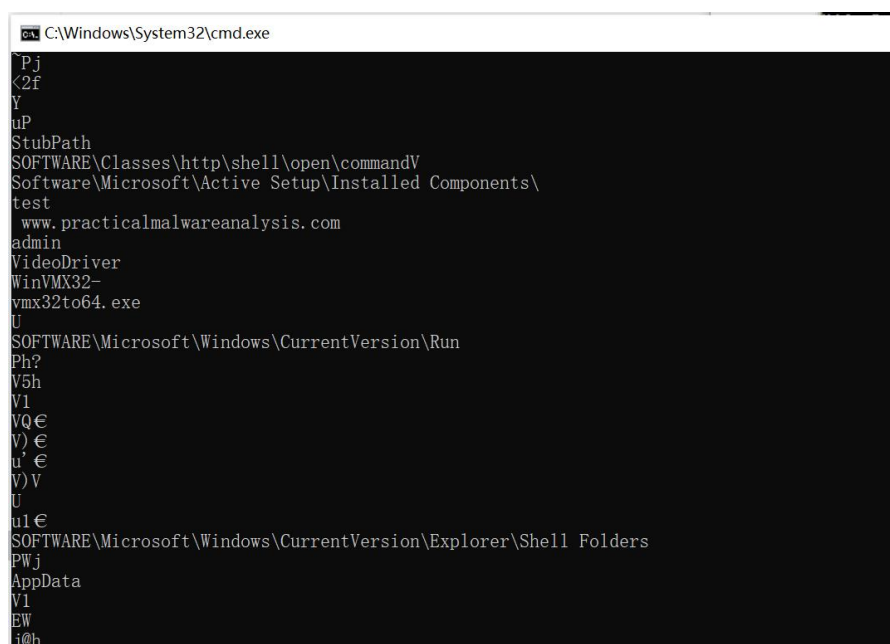
（1）使用 PEvent 打开 exe 文件，查看 SECTION .text 下的 IMPORT Address Table，这个恶意代码的导入函数内容异常少，仅有一个 ExitProcess，仅这么一个导入函数理论上无法使一个程序正常运行，怀疑出现了加壳或混淆。



(2) 使用 PEiD 检测加壳情况信息，如下图所示，观察到壳为 PEncrypt 3.1 Final -> junkcode。



(3) 因为导入函数表和 PEiD 结果均表明该文件是被加过壳的，但是通过 Strings 工具可以观察到很多有价值的字符串，比如注册表位置、域名、WinVMX32、VideoDriver、vmx32to64.exe。接下来通过动态分析基础技术来检测这些字符串如何被使用。



(4) 监视进程运行情况之前先做一系列配置：

拍摄 VMWare 快照

ApateDNS: DNS Reply IP 设置为 127.0.0.1

Process Monitor: 添加三个过滤:

Process Name > is > Lab03-01.exe

Operation > is > WriteFile

Operation > is > RegSetValue

打开 Process Explorer

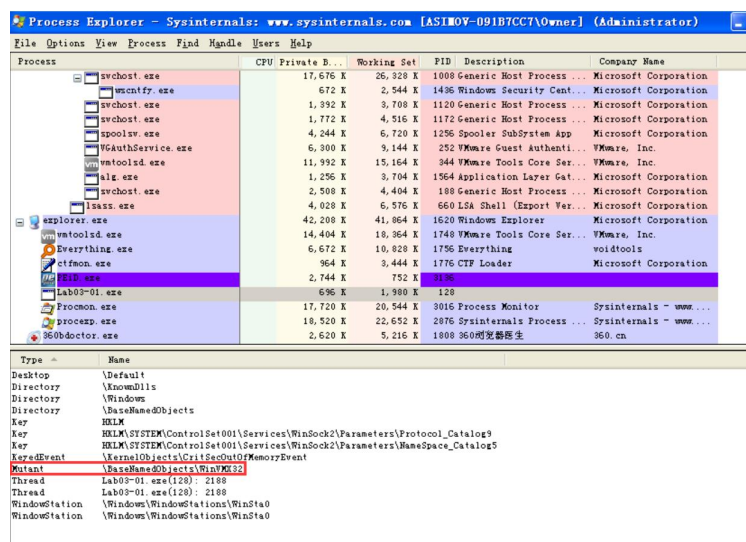
打开 netcat (nc 不支持同时监听多个端口, 可以开两个 cmd):

nc -l -p 403

nc -l -p 80

打开 WireShark

(5) Process Explorer 选中 Lab03-01.exe > View > Low Pane View > Handles, 可以看到创建了互斥量 WinVMX32:



(6) Process Explorer 选中 Lab03-01.exe > View > Low Pane View > DLL, 可以看到 ws2_32.dll 和 wshtcpip.dll 等联网的库:

unicode.nls	Windows XP USER API Client DLL	Microsoft Corporation	C:\WINDOWS\system32\unicode.nls
user32.dll	Unsubscribe Unicode script processor	Microsoft Corporation	C:\WINDOWS\system32\user32.dll
usp10.dll	Version Checking and File Installation Libraries	Microsoft Corporation	C:\WINDOWS\system32\usp10.dll
version.dll	LDAP RnK Provider DLL	Microsoft Corporation	C:\WINDOWS\system32\version.dll
winrnr.dll	Win32 LDAP API DLL	Microsoft Corporation	C:\WINDOWS\system32\winrnr.dll
wldap32.dll	Windows Socket 2.0 32-Bit DLL	Microsoft Corporation	C:\WINDOWS\system32\wldap32.dll
ws2_32.dll	Windows Socket 2.0 Helper for Windows NT	Microsoft Corporation	C:\WINDOWS\system32\ws2_32.dll
ws2help.dll	Windows Sockets Helper DLL	Microsoft Corporation	C:\WINDOWS\system32\ws2help.dll
wshtcpip.dll		Microsoft Corporation	C:\WINDOWS\system32\wshtcpip.dll

(7) 双击 WriteFile 一项, 可以看到该操作向系统路径写入了一个文件 C:\WINDOWS\system32\vmx32to64.exe。双击 RegSetValue 一项, 可以看到该操作往注册表写入了 HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\

Run\VideoDriver 一项。用 regedit 打开注册表，定位到对应位置，确实能够看到该注册表项，ApateDNS 监测到恶意代码向域名 www.practicalmalwareanalysis.com 发送了请求。并且该请求每隔 61 秒重新发送一次。

习题解答

1. 找出这个恶意代码的导入函数与字符串列表？

见上述实验过程

2. 这个恶意代码在主机上的感染迹象特征是什么？

该恶意代码创建了一个名为 WinVMX32 的互斥量，并复制自身到 C:\Windows\System32\vmx32to64.exe，并安装自己到系统自启动项中，通过创建注册表键值 HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\VideoDriver，并将其设置为复制副本的位置。

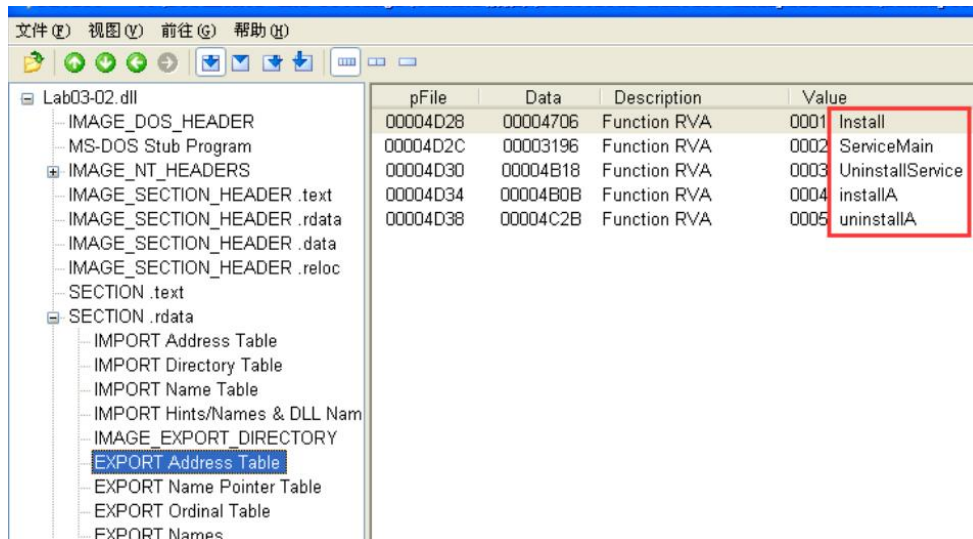
3. 这个恶意代码是否存在一些有用的网络特征码？如果存在，它们是什么？

恶意代码在进行 www.practicalmalwareanalysis.com 的域名解析后，持续地广播大小为 256 字节的数据包，其中包含看似随机的二进制数据。

Lab 3-2

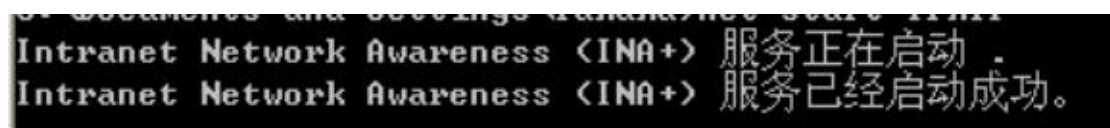
实验过程：

1. 使用 PEvent 查看导出函数。



2. 如上面看到的，执行了安装函数后，并不会返回什么信息，Lab03-02.dll 也并不会因此就变成了 Lab03-02.exe 供我们运行，因此我们需要分析下这个安装过程都干了什么。一般来说它大概率是将自己安装成了一个服务，对注册表会有操作，由于它还并不是个 exe，因此使用 Process Monitor 可能无法监视到其行为，这里就只用 Regshot 来查看安装前后的注册表变化，先利用虚拟机快照恢复功能恢复到安装前的状态，Regshot 建立第一张快照。

3. 执行命令，拍摄快照 B，点击比较快照并确定。其他新加的键值、改变的键值没那么重要了，知道添加了这几个键即可，它们告诉我们运行这个 dll 的安装函数后创建了一个服务“IPRIP”，使用命令 net start IPRIP 运行服务。服务启动成功，即让恶意代码运行起来了。



4. 在“运行”中输入“services.msc”查看服务。



名称	描述	状态	启动类型	登录为
IMAPI CD-Burning COM Service	用 ...		手动	本地系统
Indexing Service	本		手动	本地系统
Intranet Network Awareness (INA+)	Dep...	已启动	自动	本地系统
Internet Explorer	管	已启动	自动	本地系统

5. 打开 Process Explorer, 点击 “Find” - “Find Handle or DLL...”。查找 “Lab03-02.dll”。
6. 该恶意代码依附在 svchost.exe 下运行, 也就是父进程为 svchost.exe, 2.3 间中可以看到父进程 PID 为 1060。Process Monitor 中打开过滤器, 选择 “Parent PID”, 填入 “1060”。(先开启 Process Monitor 再 net start 开启服务)
7. 在开启 IPRIP 服务前后又拍摄了注册表快照来分析键值对变化, 主要还是对 IPRIP 键进行了一些增改, 没有什么太有标志性的变化, 因此该恶意代码在主机上的感染迹象特征仍是: 相关目录下出现 “IPRIP” 键; 新增了一个 IPRIP 服务, 且该服务被设置为开机自动启动。

新添加键 (2) 快照 B

```
[HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\IPRIP\Enum]
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\IPRIP\Enum]
```

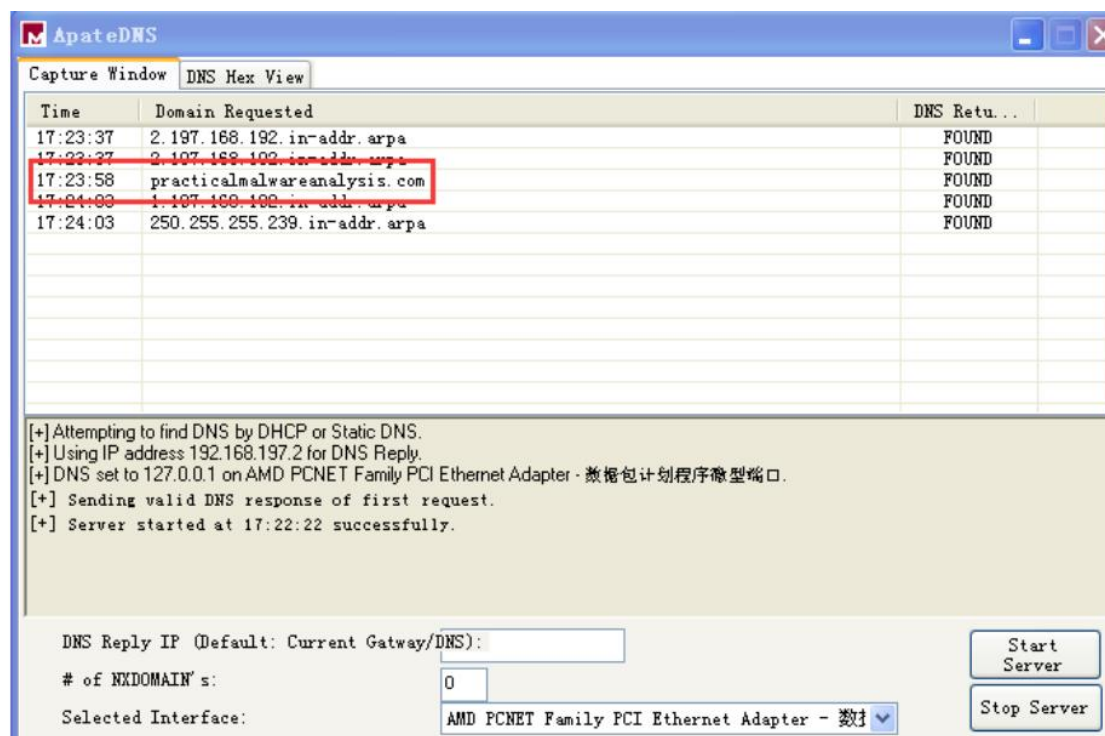
已删除值 (0) 快照 A

新添加值 (8) 快照 B

```
[HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\IPRIP\Enum]
"0"="Root\\LEGACY_IPRIP\\0000"
"Count"=dword:00000001
"NextInstance"=dword:00000001
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\IPRIP\Enum]
"0"="Root\\LEGACY_IPRIP\\0000"
"Count"=dword:00000001
```

8. 在开启服务前使用 ApateDNS 监视网络访问行为。然后使用 net start 命令开

启服务。对网站“practicalmalwareanalysis.com”进行了访问，该网址即是这个恶意代码的网络特征码。



习题解答

1. 你怎样才能让这个恶意代码自行安装？

执行 `rundll32.exe Lab03-02.dll,installA` 指令

2. 在安装之后，你如何让这个恶意代码运行起来？

执行 `net start IPRIP` 指令

3. 你怎么能找到这个恶意代码是在哪个进程下运行的？

使用 Process Explorer 来确定哪个进程正在运行服务。由于恶意代码将会运行在一个系统上的 `svchost.exe` 进程中，因此查看每个进程，直到看到该服务名，或者使用 Process Explorer 的 Find DLL 功能来搜索 Lab03-02.dll。

4. 你可以在 **procmon** 工具中设置什么样的过滤器,才能收集这个恶意代码的信息?

在 **procmon** 工具中,可以使用在 **Process Explorer** 中发现的 **PID** 进行过滤。

5. 这个恶意代码在主机上的感染迹象特征是什么?

默认情况下,恶意代码将安装为 **IPRIP** 服务,显示的服务名称为 **Intranet Network Awareness \(\INA+\)**,描述为 “**Depends INA+, Collects and stores network configuration and location information, and notifies applications when this information changes**”。

它将自身持久地安装在注册表中 **HKLM\SYSTEM\CurrentControlSet\Services\IPRIP\Parameters\ServiceDll:%Currentdirectory%\Lab03-02.dll**。

如果将 **Lab03-02.dll** 重命名为其他文件名,如 **malware.dll**,那么该恶意代码就会把 **malware.dll** 写入到注册表项中,而不是使用名称 **Lab03-02.dll**。

6. 这个恶意代码是否存在一些有用的网络特征码?

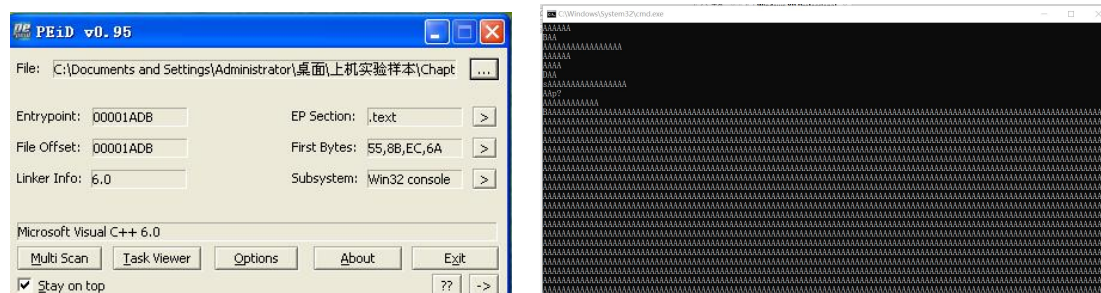
恶意代码申请解析域名 **practicalmalwareanalysis.com**,然后通过 **80** 端口连接到这台主机,使用的协议看起来似乎是 **HTTP** 协议。它在做一个 **GET** 请求 **serve.html**,使用的用户代理为 **%ComputerName% Windows XP 6.11**。

Lab 3-3

在一个安全的环境中执行 **Lab03-03.exe** 文件中发现的恶意代码,同时使用基础的动态行为分析工具监视它的行为。

实验过程

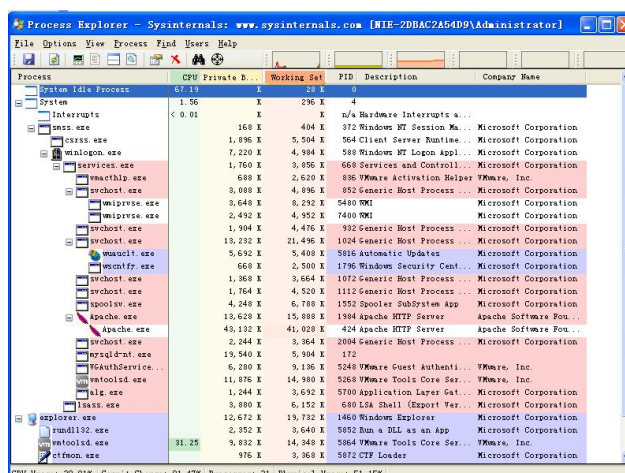
(1) 先通过 PEiD 和 Strings 做基本静态分析,可以发现未加壳(Visual C++ 6.0 编译), 字符串中发现大量 A



(2) 动态分析前准备:

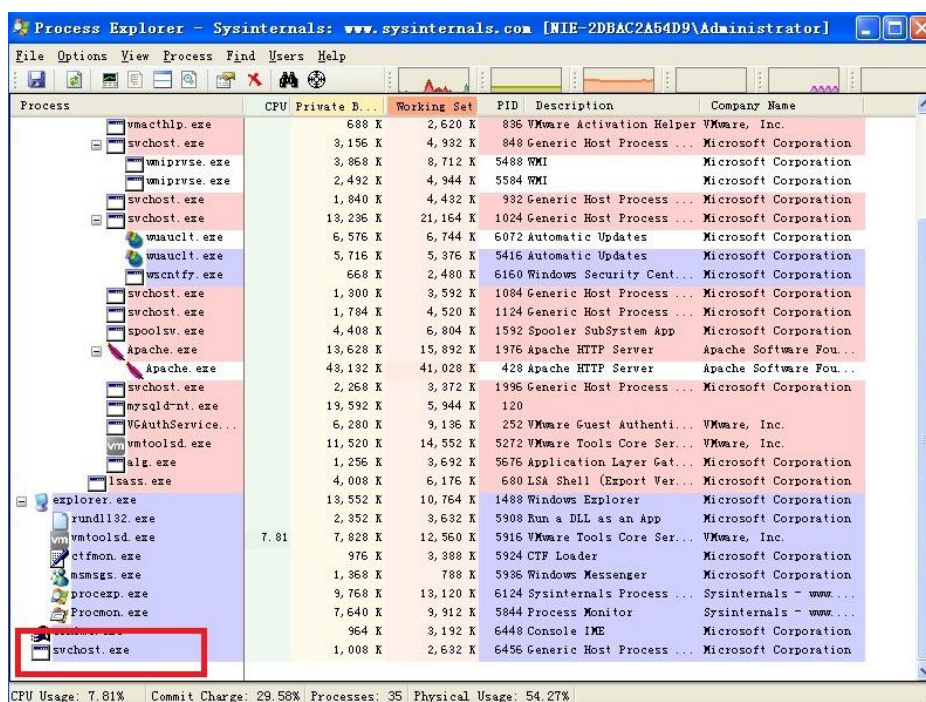
拍摄 VMWare 快照

打开 Process Explorer

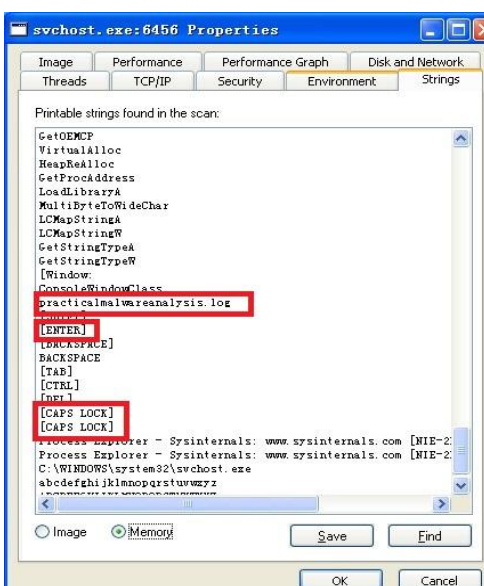
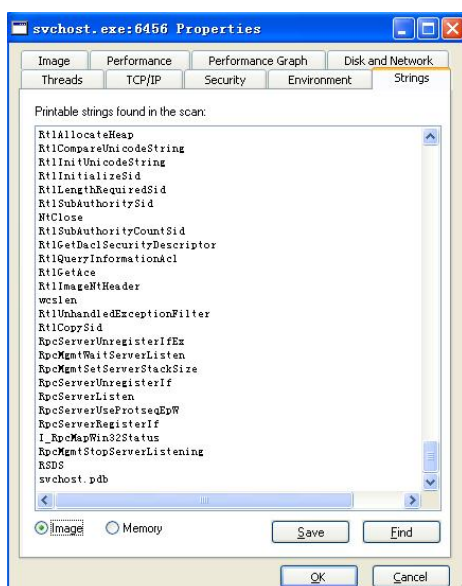


打开 Process Monitor: 关闭捕捉, 清除所有事件, 开启捕捉

(3) 运行 Lab03-03.exe, Process Explorer 中显示 Lab03-03.exe 创建了一个 svchost.exe 的进程, 然后退出, 保留 svchost.exe 作为孤儿进程运行(图中 PID 为 1732 的进程)。而正常情况下, svchost.exe 应当是 services.exe 的子进程。这一点很可疑。



(4) 右键选择 Properties->Strings, 对比磁盘和内存中的字符串, 发现磁盘镜像和内存镜像中可执行文件的字符串列表差异很大。内存中的字符串多出了 practicalmalwareanalysis.log 和 [SHIFT]、[ENTER]、[BACKSPACE]这样的字符串, 而这些通常在正常的 svchost.exe 的磁盘镜像中不应当出现, 推测是一个击键的记录器。



(5) 在 Process Explorer 中我们可看到 svchost.exe 的 PID 为 6456，因此在 Process Monitor 中过滤，添加 PID is 6456 规则。打开记事本，敲击几个字符。可以看到 Process Monitor 中的事件数立刻增加，主要是将 practicalmalwareanalysis.log 文件 WriteFile 到 Lab03-03.exe 所在路径下，打开 practicalmalwareanalysis.log，注意选择编码 Western (Windows 1252)，可以发现果然是击键的记录。

习题解答：

1. 当你使用 Process Explorer 工具进行监视时，你注意到了什么？

恶意代码执行了对 svchost.exe 文件的替换。

2. 你可以找出任何的内存修改行为吗？

对比内存映像与磁盘映像中的 svchost.exe，显示它们并不是一样的。内存映像拥有如 practicalmalwareanalysis.log 和 [ENTER] 这样的字符串，而磁盘镜像中却没有。

3. 这个恶意代码在主机上的感染迹象特征是什么？

这个恶意代码创建了一个 practicalmalwareanalysis.log 日志文件。

4. 这个恶意代码的目的是什么？

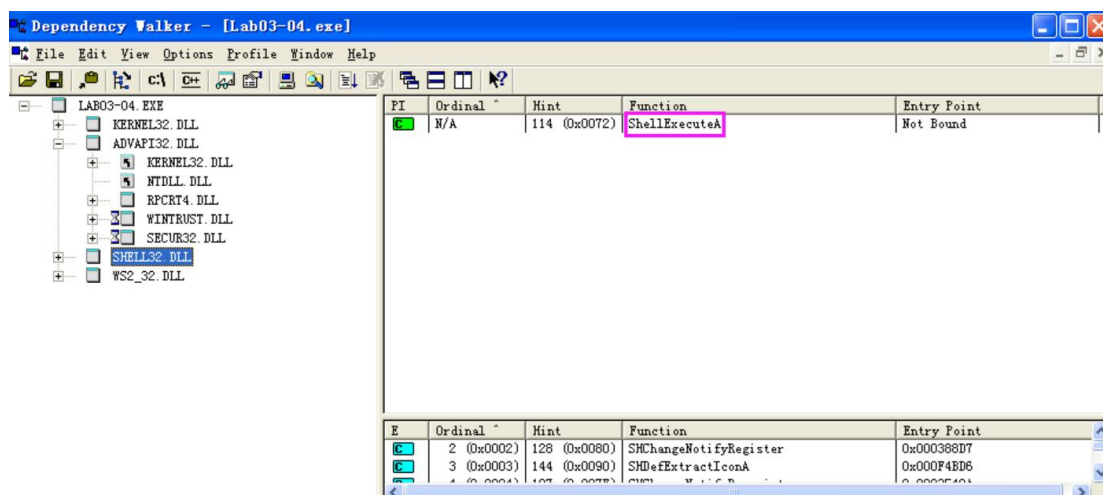
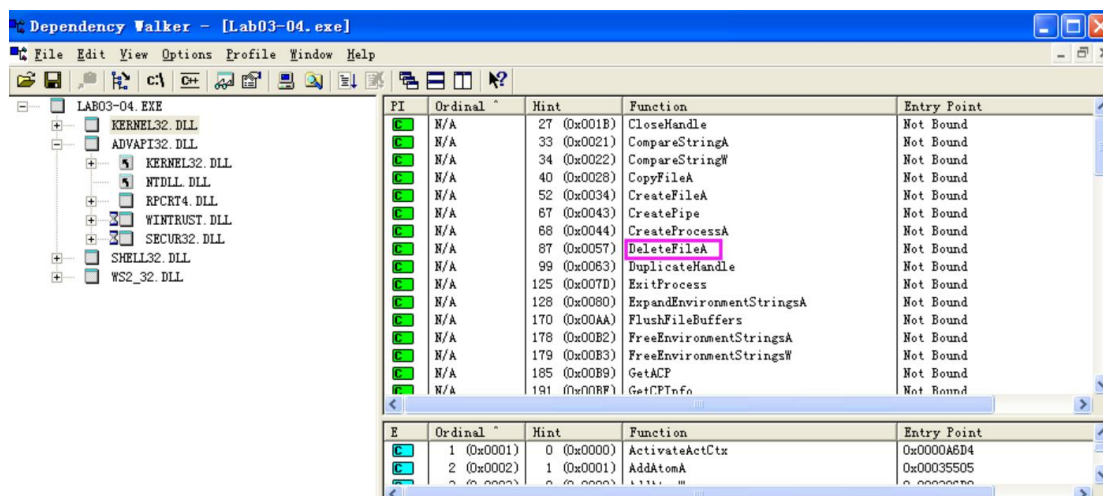
这个程序在 svchost.exe 进程上执行了进程替换，来启动一个击键记录器。

Lab 3-4

实验过程

1. 先通过 PEiD 和 Dependency Walker 做基本静态分析，可以发现未加壳(Visual

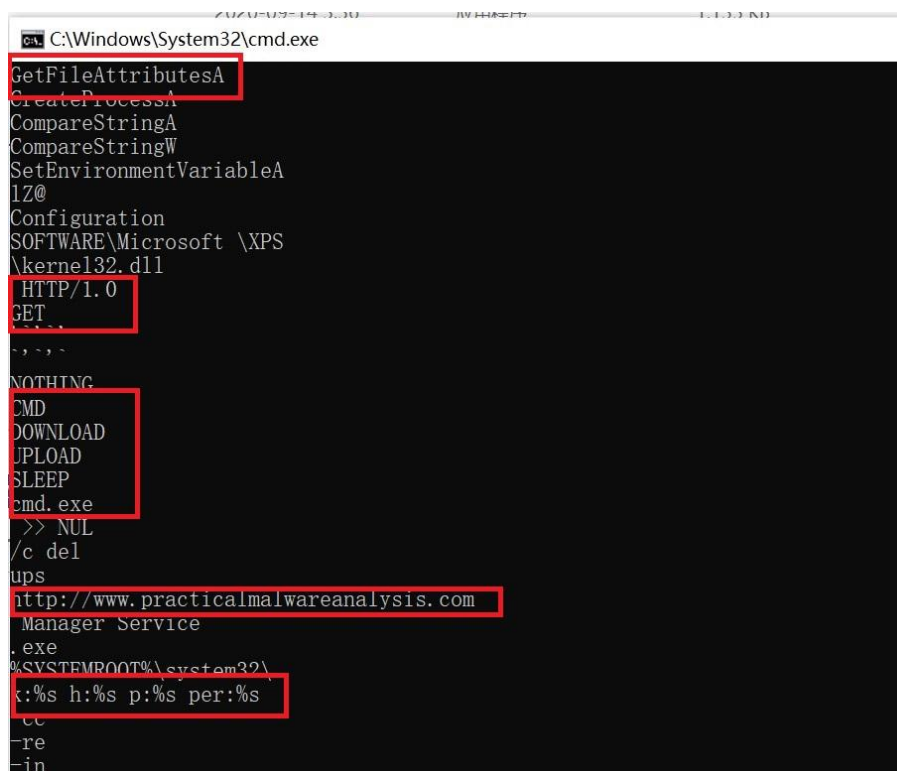
C++ 6.0 编译)



2. 通过 Strings 查看 Lab03-04.exe 字符串

根据下图，可以猜测文件和环境的函数名：GetFileAttributes、GetEnvironmentStrings，系统命令：cmd.exe、/c del、CMD、SLEEP、DOWNLOAD、

UPLOAD, 疑似命令行参数: -cc、-re、-in、k:%s h:%s p:%s per:%s, HTTP 命令: HTTP/1.0、GET, 域名: http://www.practicalmalwareanalysis.com, 系统文件: %SYSTEMROOT%\system32\



```
C:\Windows\System32\cmd.exe
GetFileAttributesA
CreateProcessA
CompareStringA
CompareStringW
SetEnvironmentVariableA
1Z@
Configuration
SOFTWARE\Microsoft \XPS
\kernel32.dll
HTTP/1.0
GET
...
NOTHING
CMD
DOWNLOAD
UPLOAD
SLEEP
cmd.exe
>> NUL
/c del
ups
http://www.practicalmalwareanalysis.com
Manager Service
.exe
%SYSTEMROOT%\system32\
k:%s h:%s p:%s per:%s
cc
-re
-in
```

3. 运行 Lab03-04.exe。可以看到几秒后该文件从文件夹中消失。Process Monitor 添加过滤规则 Process Name is Lab03-04.exe。记录了很多与注册表和文件相关的操作(RegQueryValue、RegOpenKey、RegCloseKey、ReadFile、CreateFile、CloseFile)。

习题解答

1. 当你运行这个文件时, 会发生什么呢?

双击运行该恶意代码时, 它会立刻将自身删除掉。

2. 是什么原因造成动态分析无法有效实施?

有可能需要提供一个命令行参数, 或者是这个程序缺失某个部件。

3. 是否有其他方式来运行这个程序？

可以尝试使用在字符串列表中显示的一些命令行参数，比如 `-in`，但这样做没有得到有效结果，所以需要更深入的分析。（我们将在第 9 章的实验作业中进一步分析这个恶意代码。）

Yara 规则

```
rule Lab03_01 {  
  strings:  
    $s1 = "vmx32to64.exe" fullword ascii  
    $s2 = "SOFTWARE\\Classes\\http\\shell\\open\\commandV" fullword ascii  
    $s3 = " www.practicalmalwareanalysis.com" fullword ascii  
    $s4 = "CONNECT %s:%i HTTP/1.0" fullword ascii  
    $s5 = "advpack" fullword ascii  
    $s6 = "VideoDriver" fullword ascii  
    $s7 = "AppData" fullword ascii /* Goodware String - occurred 110 times */  
    $s8 = "6I*h<8" fullword ascii /* Goodware String - occurred 1 times */  
    $s9 = "WinVMX32-" fullword ascii  
    $s10 = "^-m-m<|<|<|M" fullword ascii  
    $s11 = "Software\\Microsoft\\Active Setup\\Installed Components\\" fullword ascii /*  
Goodware String - occurred 4 times */  
  condition:  
    uint16(0) == 0x5a4d and filesize < 20KB and  
    8 of them  
}
```

```
rule Lab03_02 {  
  strings:  
    $x1 = "%SystemRoot%\\System32\\svchost.exe -k " fullword ascii
```

```

    $x2 = "cmd.exe /c " fullword ascii

    $s3 = "RegOpenKeyEx(%s) KEY_QUERY_VALUE error ." fullword ascii

    $s4 = "Lab03-02.dll" fullword ascii

    $s5 = "practicalmalwareanalysis.com" fullword ascii

    $s6 = "RegOpenKeyEx(%s) KEY_QUERY_VALUE success." fullword ascii

    $s7 = "Y29ubmVjdA==" fullword ascii /* base64 encoded string 'connect' */

    $s8 = "dW5zdXBwb3J0" fullword ascii /* base64 encoded string 'unsupport' */

    $s9 = "GetModuleFileName() get dll path" fullword ascii

    $s10 = "CreateService(%s) error %d" fullword ascii

    $s11 = "OpenService(%s) error 2" fullword ascii

    $s12 = "OpenService(%s) error 1" fullword ascii

    $s13 = "You specify service name not in Svchost\\netsvcs, must be one of following:" fullword
ascii

    $s14 = "RegQueryValueEx(Svchost\\netsvcs)" fullword ascii

    $s15 = "netsvcs" fullword ascii

    $s16 = "::$2:K:U:\\:l:" fullword ascii

    $s17 = "serve.html" fullword ascii

    $s18 = "uninstall success" fullword ascii

    $s19 = "uninstall is starting" fullword ascii

    $s20 = "Depends INA+, Collects and stores network configuration and location information,
and notifies applications when this informatio" ascii

    condition:

        uint16(0) == 0x5a4d and filesize < 70KB and

        1 of ($x*) and 4 of them

    }

```

rule Lab03_03 {

strings:

$$s_1 =$$
$$s_2 =$$
$$s_3 =$$

\$s4 =

```
$s5 = "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA" ascii
```

```
$s7 = "\\svchost.exe" fullword ascii
```

[illegible]

```
$s9 = "AAAAAABAAAAA" ascii
```

[illegible]

```
$s11 = "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA" ascii
```

```
$s13 = "AAAgAAApAAAsAAArAAAUAAAtAAAwAAAvAAAyAAAxAAA" fullword ascii
```

```

/* base64 encoded string ' * ) , + . - 0 / 2 1 ' */

$s14 =

"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAq" fullword ascii

$s15 =

"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAA" ascii

$s16 = "wqpwlKla/.5a$/.4&)a21 \" $a'.3a5)3$ %a% 5 LKALK #/.3, -a13.&3 ,a5$3,(/
5(/LKAAAA" fullword ascii

$s17 = "- 22AA13 \"5(\" -, -6 3$ / -82(2o-.&AAAAaAAA" fullword ascii

$s18 = "+A+A+A+A" fullword ascii /* reversed goodwill string 'A+A+A+A+' */

$s19 = "\"3.2.'5a" fullword ascii /* hex encoded string '2Z' */

$s20 = "wqsvLKla/.5a$/.4&)a21 \" $a'.3a-6(.a/(5( -(; 5(/LKAAAA" fullword ascii

condition:

uint16(0) == 0x5a4d and filesize < 200KB and

8 of them

}

```

rule Lab03_04 {

strings:

```

$s1 = "http://www.practicalmalwareanalysis.com" fullword ascii

$s2 = "%SYSTEMROOT%\\system32\\" fullword ascii

$s3 = " HTTP/1.0" fullword ascii

$s4 = " Manager Service" fullword ascii

$s5 = "UPLOAD" fullword ascii /* Goodware String - occurred 2 times */

$s6 = "DOWNLOAD" fullword ascii /* Goodware String - occurred 29 times */

$s7 = "command.com" fullword ascii /* Goodware String - occurred 91 times */

$s8 = "COMSPEC" fullword ascii /* Goodware String - occurred 247 times */

$s9 = "SOFTWARE\\Microsoft \\XPS" fullword ascii

```

```
$s10 = "k:%s h:%s p:%s per:%s" fullword ascii
$s11 = "\"WWSH(" fullword ascii /* Goodware String - occurred 2 times */
$s12 = "6KRich" fullword ascii
$s13 = "/c del " fullword ascii
$s14 = ">> NUL" fullword ascii

condition:

uint16(0) == 0x5a4d and filesize < 200KB and

8 of them

}
```

动态分析的优缺点

优点：可以检测复杂的内存处理错误并且精度更高。

缺点：速度慢、效率低、复杂度更高，不能保证完整的代码覆盖率，可伸缩性差，难以进行大规模测试。

四、 实验心得

通过此次实验，将静态分析与动态分析相结合进行恶意代码分析，对于第一次实验中加壳代码不知道怎么分析的问题有了初步了解。