

**《恶意代码分析与防治技术》课程实验报告**

**实验三**

****

学 院 网络空间安全学院

专 业 信息安全

学 号 2112060

姓 名 孙蕗

班 级 信息安全1班

《恶意代码分析与防治技术》课程Lab3实验报告

[一、 实验目的 3](#_Toc14879)

[二、 实验原理 3](#_Toc18024)

[三、 实验过程 3](#_Toc3969)

[（一） Lab3-1 3](#_Toc19596)

[（二） Lab3-2 8](#_Toc26069)

[（三） Lab3-3 13](#_Toc7279)

[（四） Lab3-4 16](#_Toc22067)

[四、 实验结论及心得体会 18](#_Toc8544)

1. **实验目的**

使用动态分析基础技术来分析恶意代码，包括查找恶意代码的导入函数和字符串列表，监视恶意代码的行为，识别感染迹象特征，以及尝试运行和分析恶意代码的行为。

1. **实验原理**

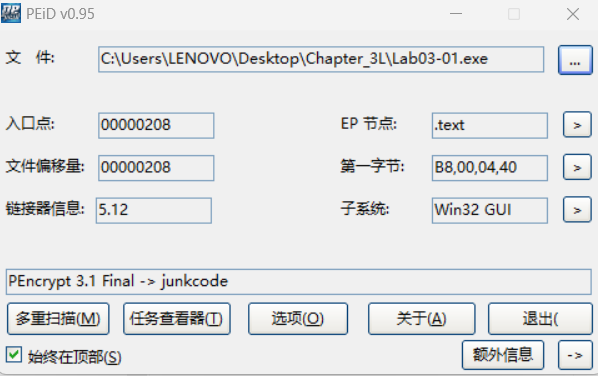
使用各种工具和技术，包括PEview、strings、Process Explorer、Procmon、Wireshark等，用于动态分析恶意代码的行为和特征。

通过监视进程、文件、注册表等的变化，可以识别恶意代码的行为和感染迹象。

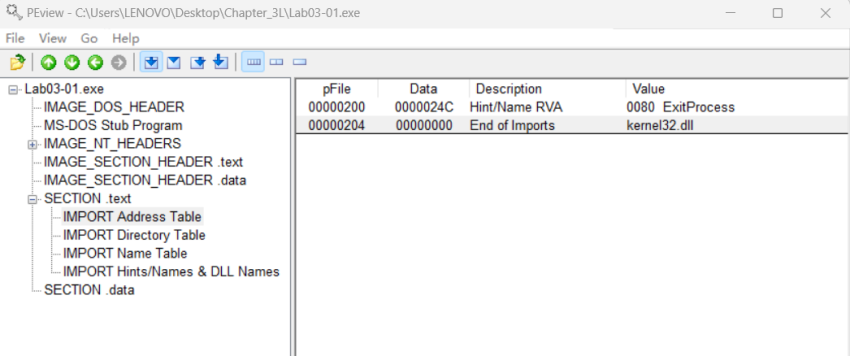
1. **实验过程**
2. **Lab3-1**

**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。**

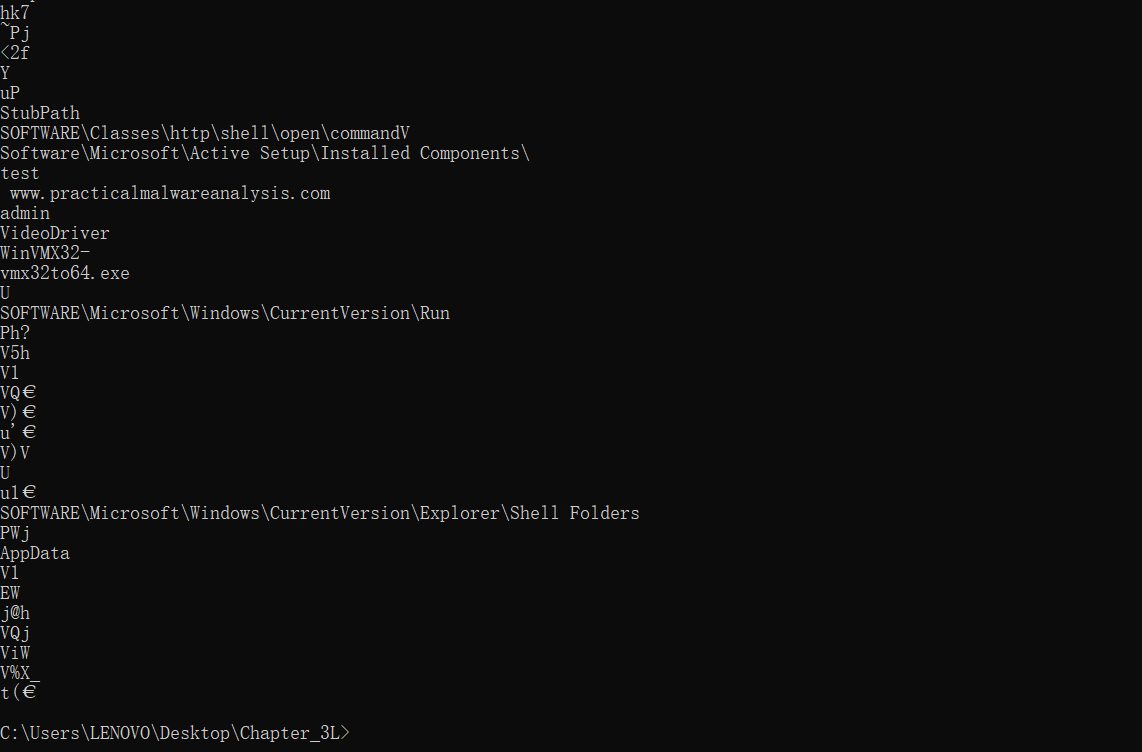
1. **找出这个恶意代码的导入函数与字符串列表。**
2. 用PEiD打开Lab03-01.exe，可以看到Lab03-01.exe是加壳的，是PEncrypt 3.1 Final -> junkcode的壳。



1. 使用PEview查看Lab03-01.exe的PE文件结构和字符串列表，PEview工具显示该程序导入函数只有ExitProcess，没有其他导入函数。



1. 使用strings.exe查看恶意代码中有哪些字符串。使用命令strings Lab03-01.exe，如下图所示。

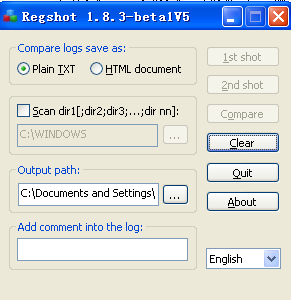


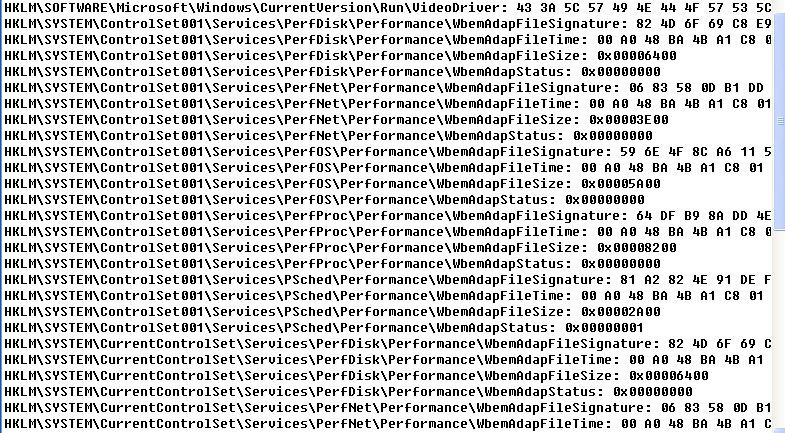
看到了很多的字符串，如注册表位置、域名、WinVMX32、VideoDriver、vmx32to64.exe等。其中方框标记的内容中包含很多注册表信息、一个域名信息和一个应用程序。

猜测分析可能会通过连接访问该网址下载某些木马文件或者通过vmx32to64.exe下载打开某些后门。

1. **这个恶意代码在主机上的感染迹象特征是什么？**
2. 使用RegShot工具分析注册表变化。

RegShot 是一种注册表比较工具，它通过两次抓取注册表而快速地比较出答案。它还可以将注册表以纯文本方式记录下来，便于浏览；还可以监察 Win.ini，System.ini 中的键值；还可以监察Windows目录和 System 目录中文件的变化，为手工卸载某些软件创造条件。通过扫描并保存注册表的“快照”，并对两次快照进行自动的对比，找出快照间存在的不同之处，结果保存成txt或者html文档。

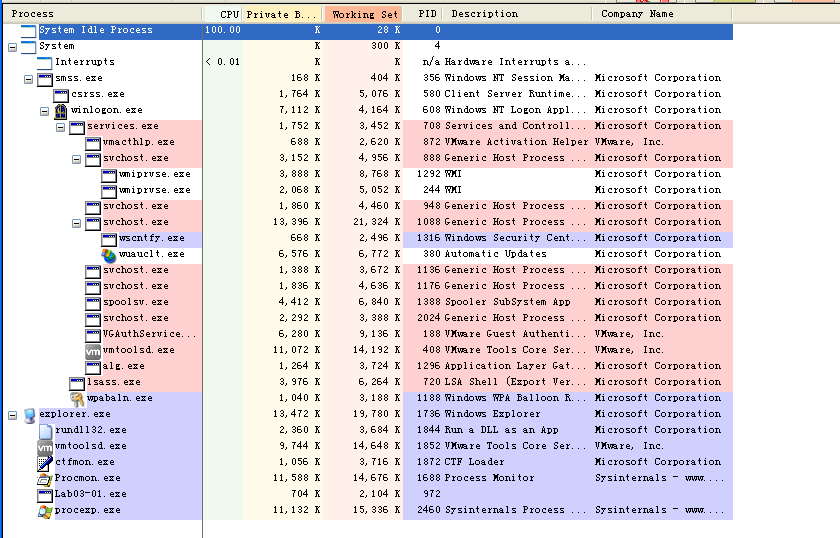




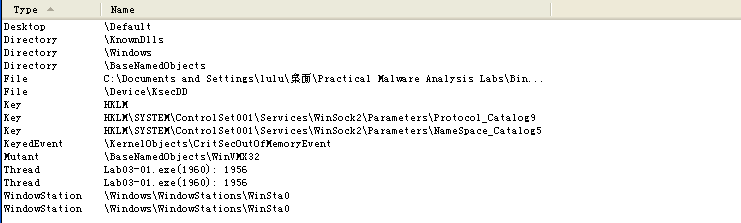
快照比较分析得出该恶意软件在HKEY\_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\VideoDriver下添加了一个名为-VideoDriver的自启动项：发现在自启动项VideoDriver中增加了键值43 3A 5C 57 49 4E 44 4F 57 53 5C 73 79 73 74 65 6D 33 32 5C 76 6D 78 33 32 74 6F 36 34 2E 65 78 65，将它换成字符为C:\WINDOWS\system32\vmx32to64.exe，说明VideoDriver自启动项就是指向system32目录下的vmx32to64.exe，程序vmx32to64.exe在开机时会自启动。

1. 使用Process explorer分析

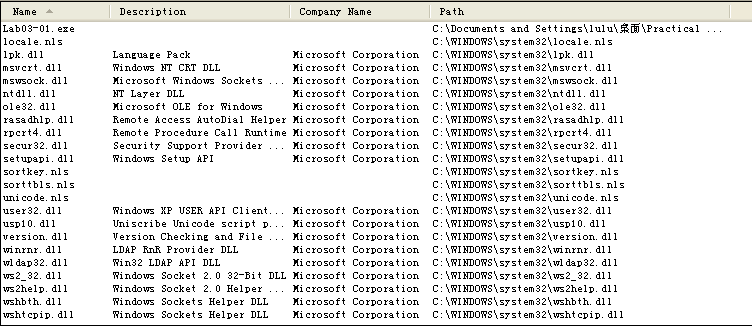
先运行procmon工具，并清除所有事件；启动Process Explorer，同时配置出一个虚拟网络，包括ApateDNS、Netcat(监听端口80和443),以及用于网络数据包捕获的Wireshark。



选择View→Lower Pane View→Handles，可以看到，恶意代码已经创建了一个名为WinVMX32的互斥量。



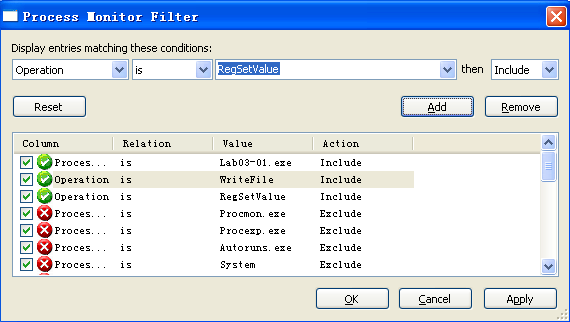
选择View→Lower Pane View→DLL,看到恶意代码动态装载的DLL文件。



ws2\_32.dll和wshtcpip.dll库文件的存在说明该样本存在网络方面的操作。

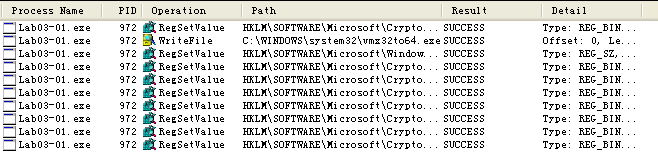
1. 使用Process Monitor分析

选择Filter→Filter,来呼出过滤器对话框，然后设置三个过滤器：一个是对进程名称的过滤(显示Lab03-01.exe对系统所做的)，和两个操作上的过滤，包含了RegSetValue和WriteFile，查看恶意代码对文件系统和注册表的修改操作。



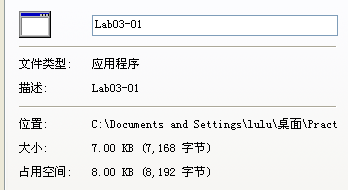
设置完过滤器之后，单击Apply按钮，便可看到过滤后的结果。显示的条目从数千减少到只有10条，其中只有1个WriteFile操作的条目，并有9个关于RegSetValue操作的条目。

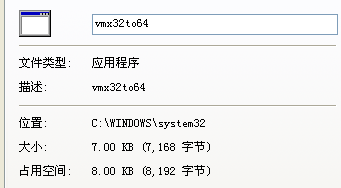
HKLM\SOFTWARE\Microsoft\Cryptography\RNG\Seed键值上的RegSetValue操作是典型的噪声。第四条往后都有Seed，Seed说明该程序用了随机数，随机数发生器的种子会有软件在注册表中不停地更新。



双击WriteFile操作条目，显示记录会告诉我们，恶意代码往C:\WINDOWS\System32\vmx32to64.exe中写了7168字节,恰好是Lab03-01.exe文件的大小。

打开Windows资源管理器浏览到这个位置，使用WinMD5查看vmx32to64.exe与Lab03-01.exe的MD5值，可以看到这个新创建的文件有着与Lab03-01.exe相同的MD5哈希值，这告诉恶意代码已经复制本身到这个文件系统位置上。这是一个非常有用的感染主机迹象特征，因为它使用了一个硬编码的文件名。



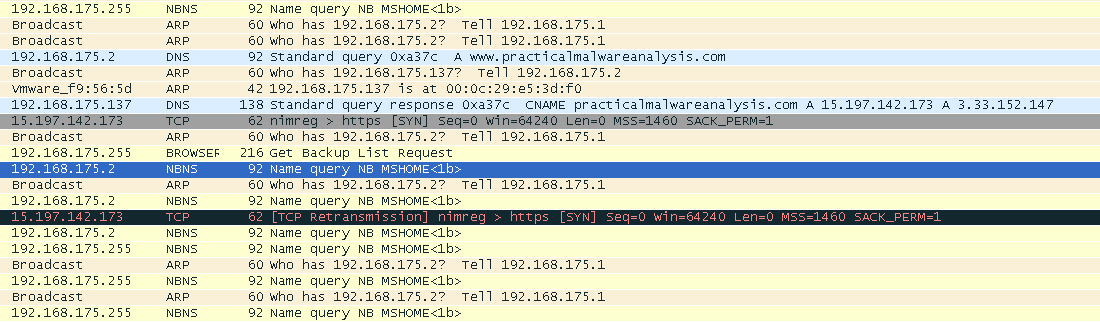


双击红框中的第二行，发现恶意代码在Windows\System32路径下创建写入来vmx32to64.exe文件并且在CurrentVersion\Run下创建了VideoDriver自启动项键值，用于在系统启动时自动运行vmx32to64.exe。

综上所述，经过初步分析，可以认为在主机上创建互斥量、尝试复制自身并加入系统自启动项，创建注册表键值等，这些都可以作为其在主机上感染的迹象特征。

1. **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？**

使用wireshark进行抓包分析，可以看到如下结果：



出现了网址www.practicalmalwareanalysis.com。

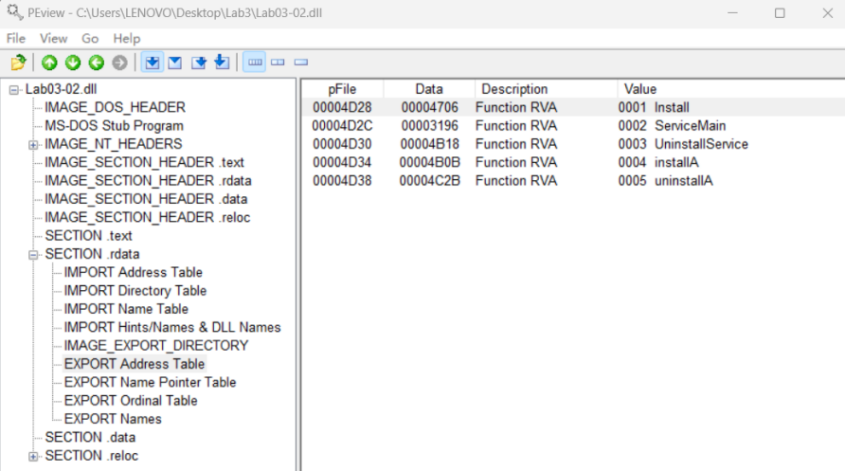
综上所述，Lab03-01.exe访问了网址[www.practicalmalwareanalysis.com，可以作为网络依据。](http://www.practicalmalwareanalysis.com可以作为网络依据。)

1. **Lab3-2**

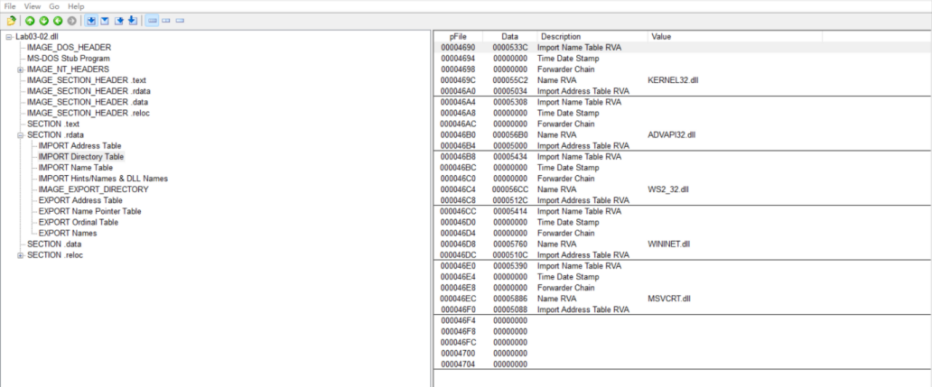
**使用动态分析基础技术来分析在Lab03-02.dll文件中发现的恶意代码。**

1. **你怎样才能让这个恶意代码自行安装？**

使用PEview查看Lab03-02.dll的导出函数



查看directory table,发现了一些依赖导入函数动态链接库



1. ADVAPI32.dll：

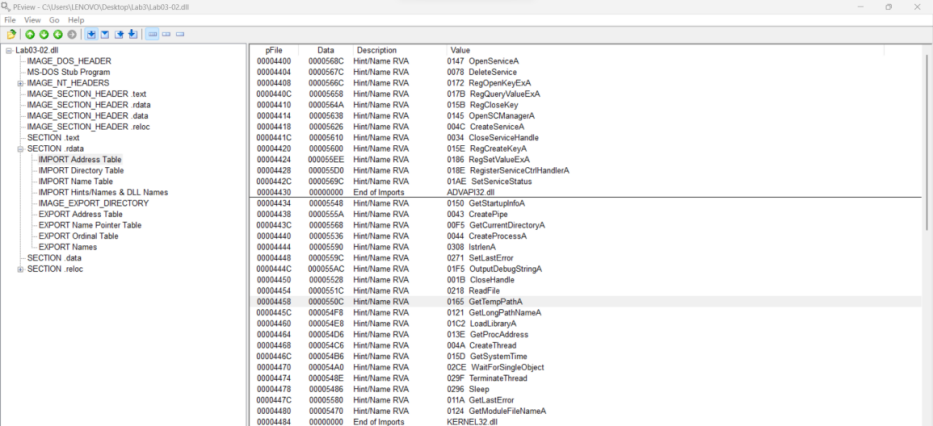
advapi32.dll是Windows操作系统的一个核心动态链接库文件，包含了很多与系统安全、身份验证、访问控制和注册表操作等相关的函数。该库提供了一些功能，例如创建和管理用户帐户、验证用户凭据、创建和管理安全上下文、操纵Windows服务、管理系统注册表等。

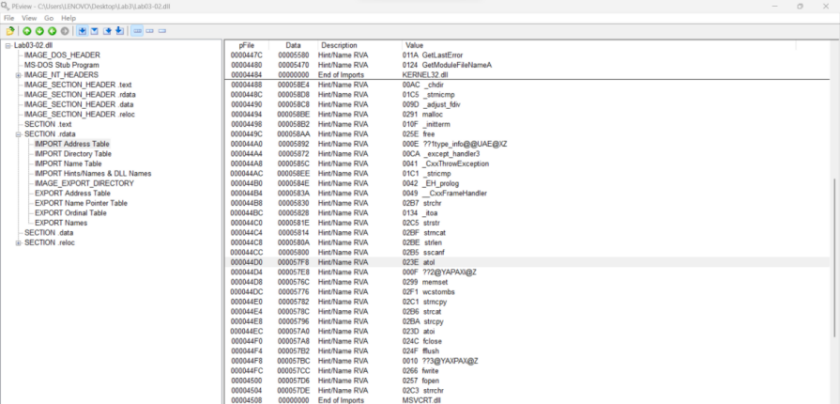
1. MSVCRT.dll:

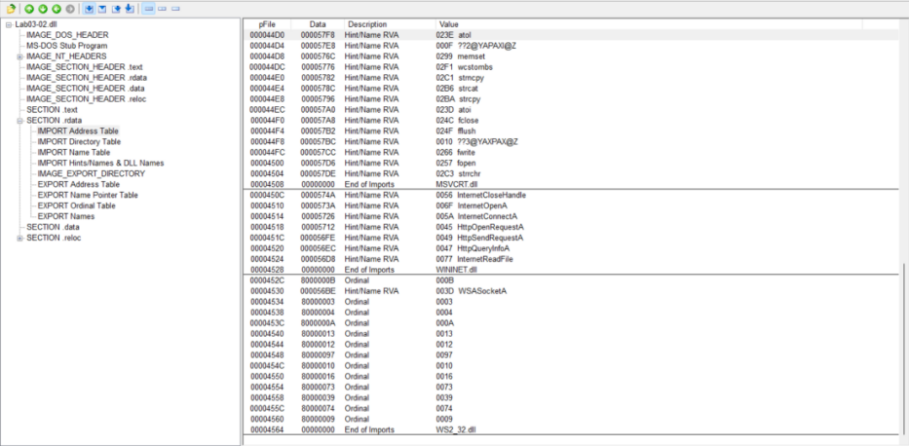
msvcrt.dll是windows操作系统中提供的C语言运行库执行文件(Microsoft Visual C Runtime Library)，其中提供了printf, malloc, strcpy等C语言库函数的具体运行实现，并且为使用C/C++(Vc)编译的程序提供了初始化（如获取命令行参数）以及退出等功能。

观察ADVAPI32.dll依赖的导入函数，发现该恶意代码的功能实现中涉及了开启服务；开启注册表、注册表查询、创建注册表键、设置注册表键值、注册服务控制处理、设置服务状态等。这显然是为了实现ServiceMain的正常运行。

查看导入函数，发现程序可以创建进程（CreateProcessA）和线程（CreateThread）服务，创建操控服务（CreateServiceA、OpenServiceA）、操控注册表（RegCreateKeyA、RegSetValueExA、RegisterServiceCtrlHandlerA、RegOpenKeyExA、RegQueryValueExA、RegCloseKey等），对网络（InternetOpenA、InternetReadFile、InternetConnectA、InternetCloseHandle、HttpOpenRequest、HttpSendRequestA、HttpQueryInfoA等）进行操作等。

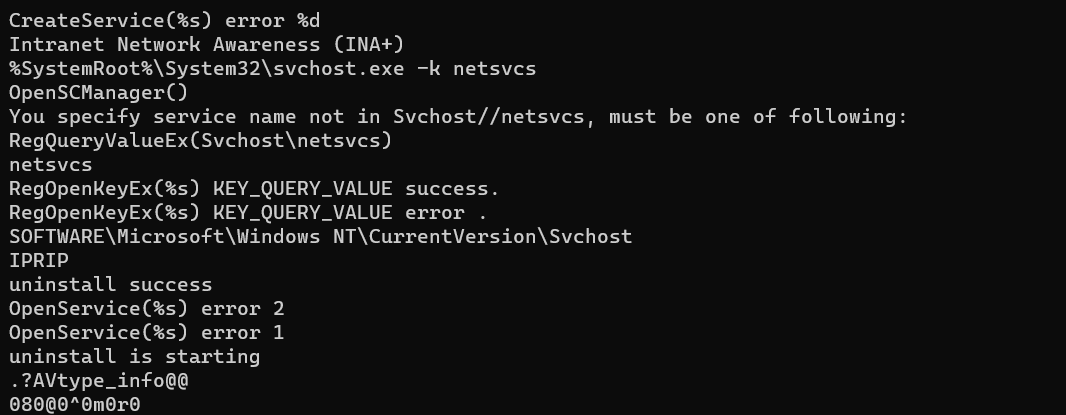






使用strings Lab03-02.dll命令进行分析，除去刚才已分析的部分，发现





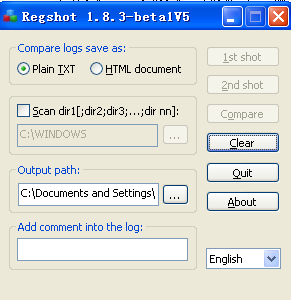
观察可打印字符串中发现，除了同上发现的一下导入导出函数外，还发现了practicalmalwareanalysis.com等一些域名信息、一些注册表信息、IPRIP、serve.html等字符串，此外还发现了svchost.exe的可执行程序。

综上所述，该程序可能通过访问 practicalmalwareanalysis.com网址，来下载某些程序，将自己下载为某个服务，而dll文件在Windows系统中一般不可以直接执行，所以必然存在某个可执行程序来运行Lab03-02.dll文件，而在可打印字符串中发现的svchost很有可能便是运行dll文件的exe文件。推测这个恶意代码需要使用导出函数installA将自身注册为恶意代码分析实战一个服务，我们将使用这个函数来试图安装恶意代码。

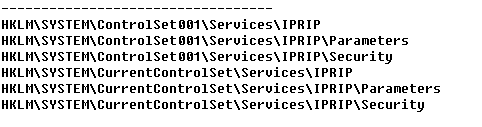
根据上述分析可知，可以利用rundl32.exe工具，使用命令rundll32.exe Lab03-02.dll,installA，运行恶意代码导出的installA函数，便可将恶意代码安装为一个服务。

1. **在安装之后，你如何让这个恶意代码运行起来？**

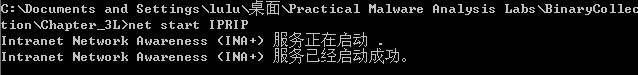
使用Process Explorer监控系统上运行的进程。在使用命令rundll32.exe Lab03-02.dll,installA运行Lab03-02.dll文件前后，分别用regshot记录一下注册表信息，点击compare对比运行Lab03-02.dll的变化。



观察编译后的.txt文件，在Keys added一节中，显示了恶意代码将自身安装为IPRIP服务，由于这个恶意代码是个DLL文件，它依赖于一个可执行文件来执行它；看到ImagePath被设置为svchost.exe意味着这个恶意代码将会在一个swchost.exe进程中启动。

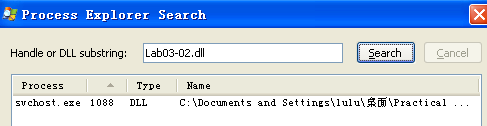


可以看到恶意代码将为自身安装IPRIP服务。使用net start IPRIP启动服务，就可以成功运行了。



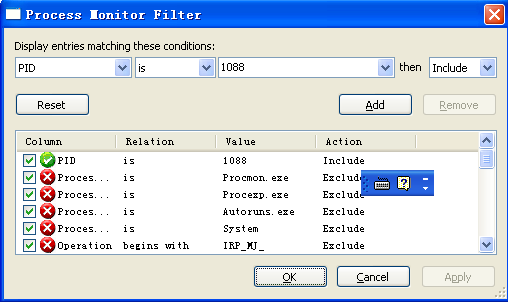
1. **你怎么能找到这个恶意代码是哪个进程下运行的？**

打开Process Explorer，在里面查找Lab03-02.dll，发现Lab03-02.dll附着在svchost.exe上运行，其pid为1088。

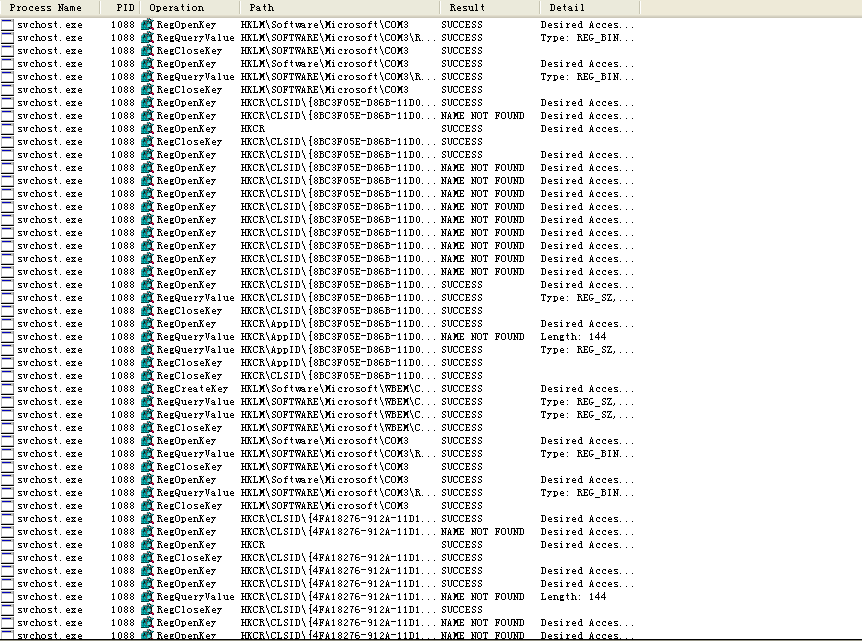


1. **你可以在procmon工具中设置什么样的过滤器，才能收集这个恶意代码的信息？**

已知该程序是由svchost.exe运行的，并且已经得知了其PID为1088，所以在Process Monitor的filter使用Process Explorer中发现的PID作为过滤器。



下面是搜索到的部分结果



1. **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+)，它将自身持久地安装在注册表中HKLM\SYSTEM\CurrentControlSet\Services\IPRIP\Parameters\ServiceDll:%CurrentDirectory%\Lab03-02.d11，在Process Explorer中搜索DLL即可查看到Intranet Network Awareness这一服务。

如果将Lab03-02.dll重命名为其他文件名，如malware.dll,然后这个恶意代码就会把malware.dll写入到注册表项中。

综上所述，创建IPRIP服务可以是这个代码在主机上的感染迹象特征。

1. **这个恶意代码是否存在一些有用的网络特征码？**

查看网络情况，发现了1088的端口的网络迹象。

对Wireshark的结果进行进一步分析



反映了该程序申请解析域名practicalmalwareanalysis.com，通过80端口连接到这台主机，使用HTTP协议，做一个GET请求serve.html,可以作为网络上的一种判断迹象。

1. **Lab3-3**

**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。**

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

首先使用Process Explorer和procmon工具。使用File→Capture Events将事件捕获先关闭，直到所有动态分析程序都就位，准备好执行恶意代码的时候，再打开开关。用Filter→Filter呼出过滤器对话框，然后按一下Reset按钮，确保只有默认过滤器被启用。

从命令提示符或双击图标运行Lab03-03.exe，在Process Explorer中观察进程变化。

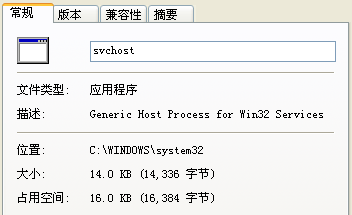
执行恶意代码，观察 Process Explorer发现，Lab03-03.exe 刚开始运行几秒钟便退出了，并创建了一个子进程svchost.exe。过滤procmon的结果，分析 Lab03-03.exe的行为，发现它可以创建进程、加载映像、创建文件、关闭文件等，通过进一步对operation的过滤，我们可以发现它确实创建了一个子进程svchost.exe，PID为2768。

注意到存在大量的svchost进程，但瞬间会变为红色，也就是创建子进程swchost.exe之后便自动退出，将svchost.exe进程继续作为一个“孤儿”进程执行。swchost.exe作为“孤儿”进程是极不寻常而且是高度可疑的。



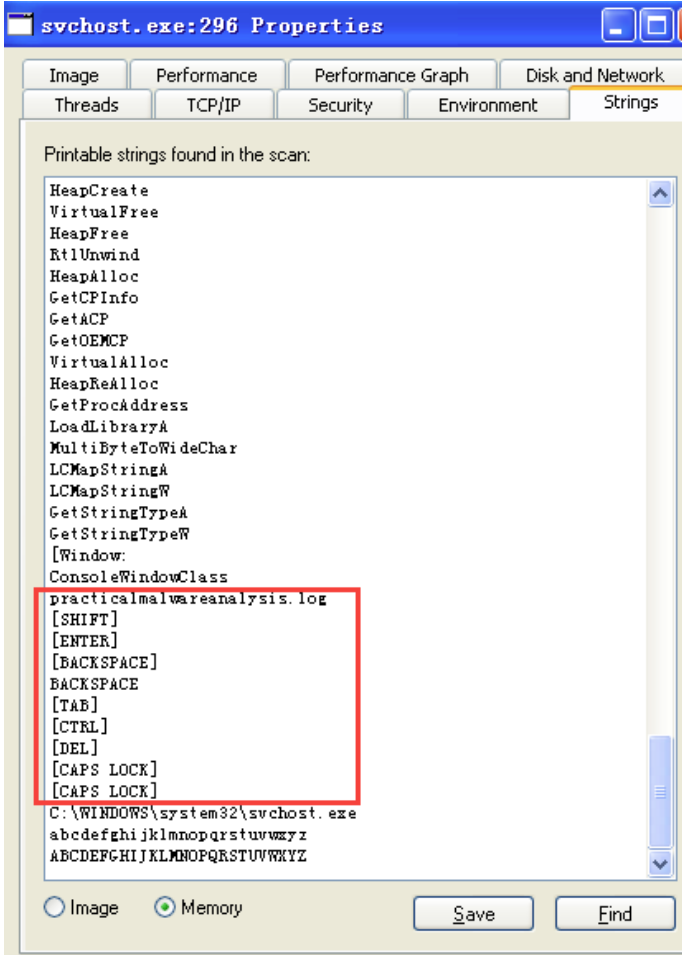
1. **你可以找出任何的内存修改行为嘛？**

对svchost.exe"孤儿"进程右击选择Properties进行进一步分析，查看它的特征。svchost.exe文件大小只有14KB，但在process explorer中大小远大于14KB。

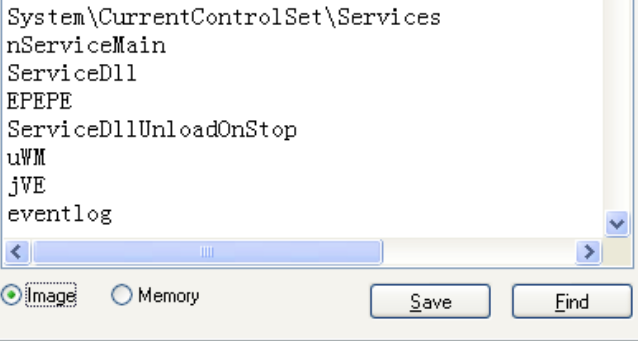


进程看起来像是一个PID为2768的合法svchost.exe进程，但这个svchost.exe是很可疑的，因为svchost.exe通常是services.exe的子进程。

svchost.exe 在 Image 和 Memory 上是非常不同的，Memory拥有 practicalmalwareanalysis.log、[SHIFT]、[ENTER]等字符，而 Image 中没有。



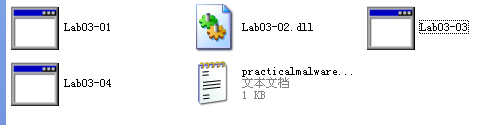
（Memory镜像）



（Image镜像）

从这个相同的属性页面中，选择Strings同时显示在磁盘镜像中和内存镜像中可执行文件的字符串列表。通过在Image和Memory单选按钮之间切换，可以看出两者镜像中的显著差异。

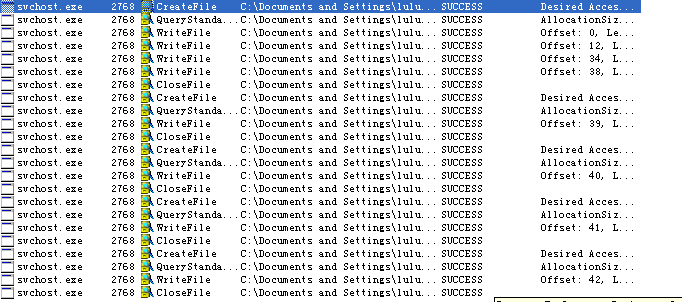
观察Memory镜像框中标出的内容。最上方是一个写入的文件名，然后是一系列键盘符。分析可知，这个程序在尝试创建practicalmalwareanalysis.log文件，这就是尝试修改内存的行为。Memory镜像中蓝框的内容一般不会在磁盘镜像中一个典型的svchost.exe文件中出现。磁盘中镜像，确实也没有这样的指令，所以可以判定是恶意代码修改的。



1. **这个恶意代码在主机上的感染迹象特征是什么？**

该恶意软件在Windows\System32目录下创建svchost.exe，且在 Lab03-03.exe的同一目录下创建了一个practicalmalwareanalysis.log日志文件。

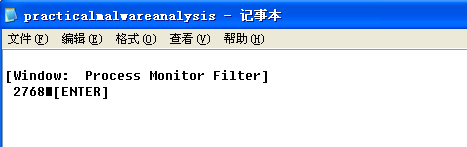
practicalmalwareanalysis.log字符串的存在，再加上出现了[ENTER]、[CAPS LOCK]、[BACKSPACE]等字符串，表明这个程序很可能是一个击键记录器。使用进程svchost.exe的在可Process Explorer中发现的PID（2768）在procmon工具中创建一个过滤器，并使用过滤器进行过滤，只显示从PID(2768)进程的事件。



发现swchostexe的CreateFile和WriteFile事件在频繁的写一个名为practicalmalwareanalysis.log的文件。它在不断地尝试对自己创建的日志文件practicalmalwareanalysis.log进行编辑，这可以作为感染的迹象特征。

1. **这个恶意代码的目的是什么？**

用记事本打开practicalmalwareanalysis.log,便显示出了刚刚在记事本中输入的击键记录。



这个程序首先创建了一个svchost.exe进程，然后就自动退出，可以认为这是尝试在伪装。而其创建的进程试图对原有的进程svchost.exe进行替换，实现一个击键记录器。创建practicalmalwareanalysis.log文件，内存映像中的字符串信息中会存在大量键盘指令，该文件是为了记录用户对键盘的操作。

1. **Lab3-4**

**使用基础的动态行为分析工具来分析在Lab03-04.exe文件中发现的恶意代码。**

**(这个程序还会在第9章的实验作业中进一步分析)**

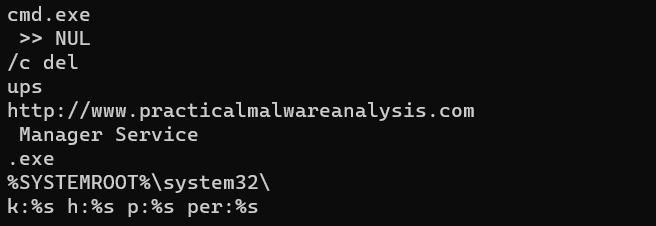
1. **当你运行这个文件时，会发生什么呢?**

当尝试双击打开或者是命令行运行该程序时，发现该程序会打开一个命令行窗口，迅速关掉，最后该程序都会进行自我删除。

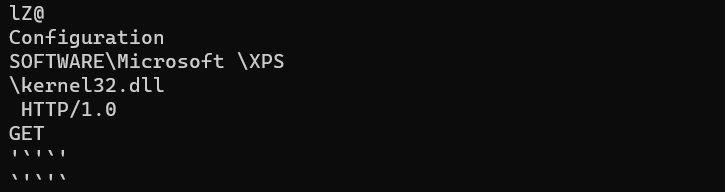
在Process Explorer中可以看到，快速运行了Lab03-04.exe，打开了cmd.exe，然后进程自己消失。

1. **是什么原因造成动态分析无法有效实施?**

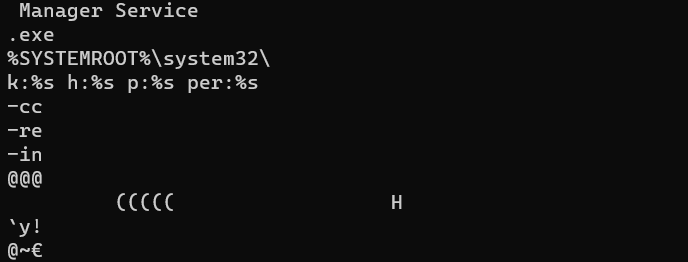
使用strings Lab03-04.exe查看字符串，进行静态分析



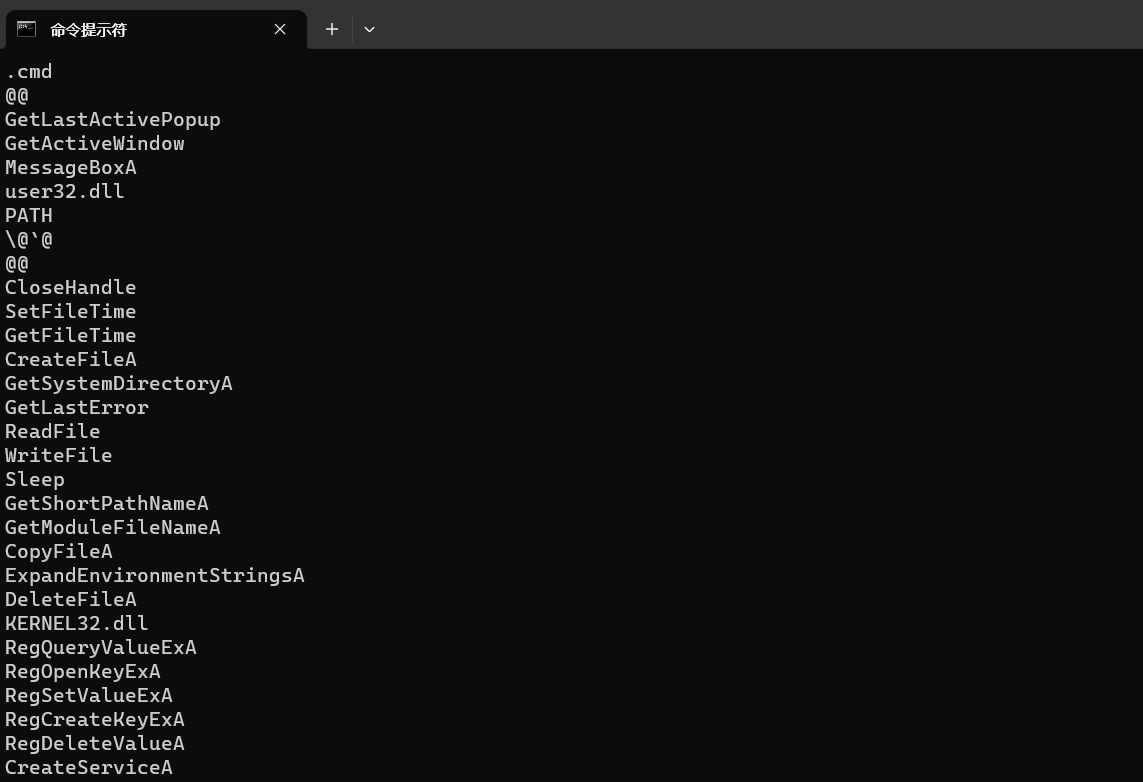
观察到cmd命令和https://www.practicalmalwareanalysis.com网址，还看到了/c del删除操作。

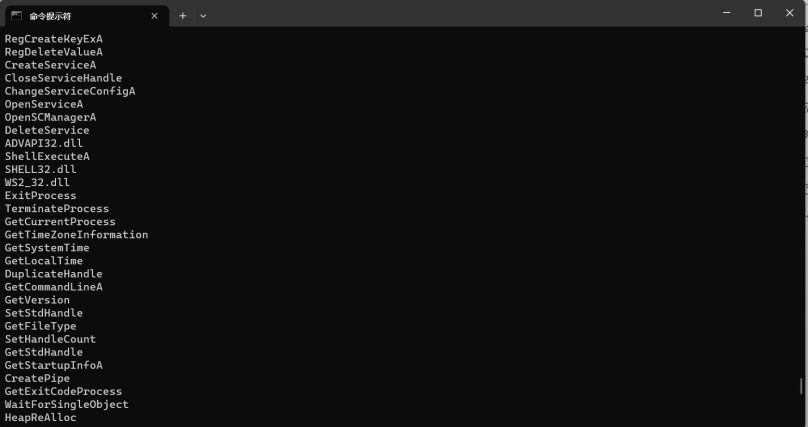


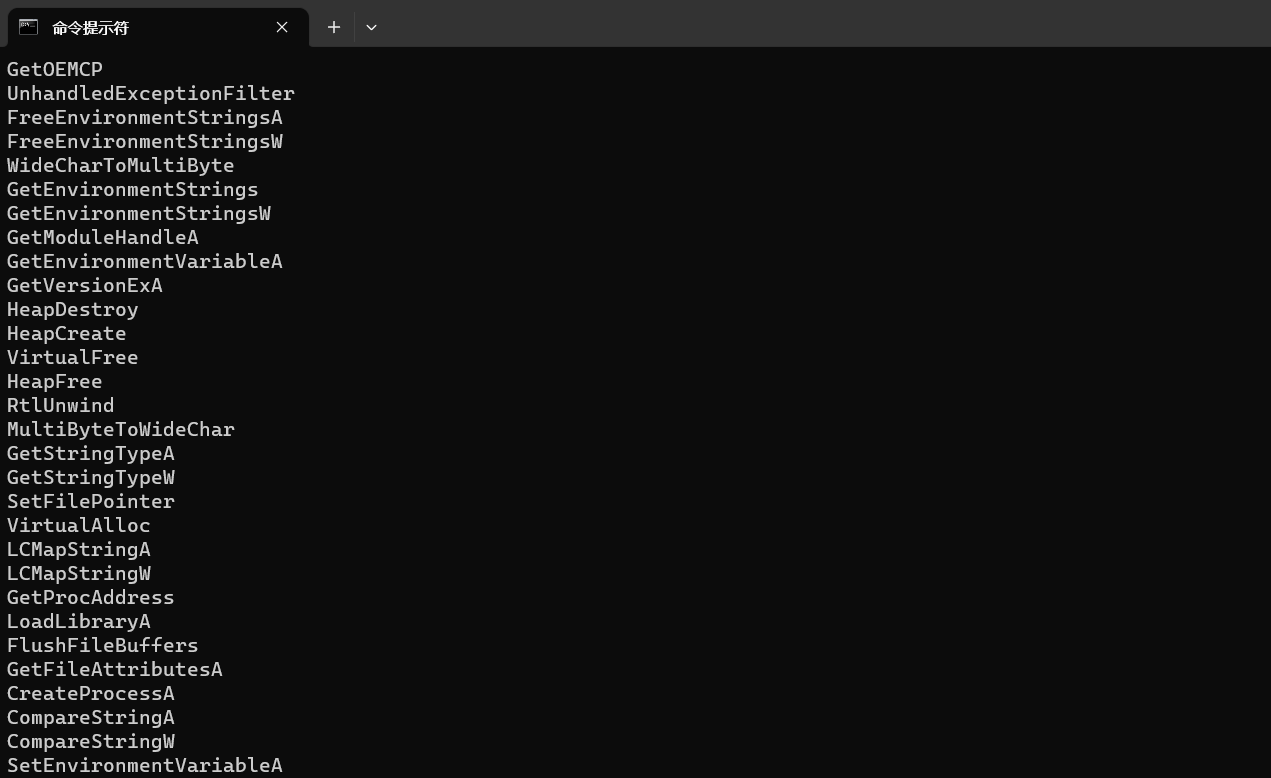
看到域名、注册表位置，如字符串SOFTWARE\Microsoft \XPS;DOWNLOAD、UPLOAD这样的命令字符串，以及HTTP/1.0字符串等。推测可能是在命令行中运行某些指令，来修改注册表、联网等。这些表明恶意代码可能是一个HTTP后门程序。



字符串-cc、-re、-in应该是一些命令行参数(例如-in可能是install的缩写)。







观察导入函数可知，很多函数与之前的相同，但然没有如WriteFile、RegSetValue之类的事件。但是发现了一个CreateProcessA及与其相关的条目。分析可知，该恶意代码是通过使用"C:\WIND0WS\system32\cmd.exe"/c del Z:\Lab03-04.exe >> NUL来从系统中删去自身的。

综上所述，猜测动态分析无法有效实施的原因可能是需要提供一个命令行参数，或者这个程序的某个部件缺失了。

1. **是否有其他方式来运行这个程序？**

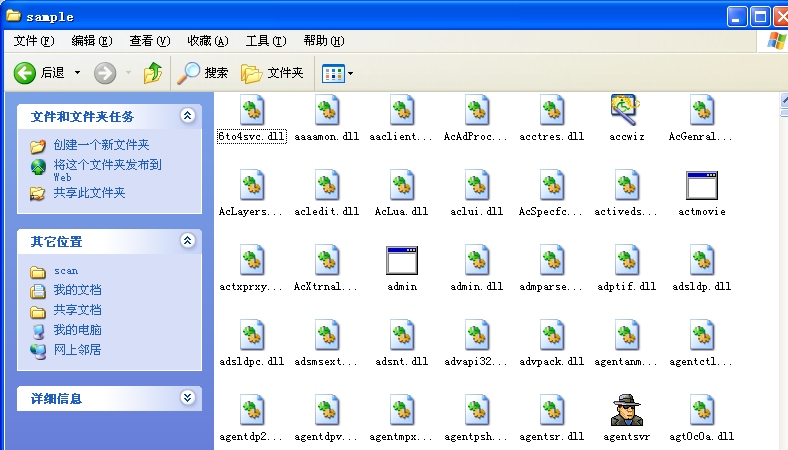
尝试使用在字符串列表中显示的一些命令行参数，比如-in、-re、-cc，但这样做却没有得到有效的结果，程序还是会删除自身，需要更深入的分析。

1. **实验结论及心得体会**
2. **实验结论**
3. 恶意代码可以通过修改注册表、创建自启动项、创建互斥量等方式在主机上感染迹象特征。
4. 恶意代码可能会通过访问特定的域名或URL来与远程服务器通信，这可以作为网络特征码。
5. 恶意代码可以创建服务或进程，以隐藏自身并运行在系统中。
6. 恶意代码可以修改内存中的内容，如键盘记录器可以记录用户的击键信息。
7. 有些恶意代码可能会自我删除以避免被检测。
8. **心得体会**

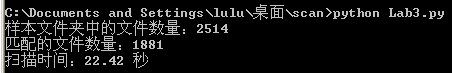
通过本次实验了解了恶意代码分析和感染迹象特征的识别方法。动态分析是一种有力的工具，可以帮助我们理解恶意代码的行为和特征，从而加强网络安全防护和恶意代码防治技术。实验中的工具和技术对于安全专业人员来说是非常重要的，能够帮助我们更好地应对潜在的安全威胁。

1. **写yara规则进行扫描**

使用scan.py将所有PE结构的文件保存到scan.py文件所在目录的sample文件夹中。



运行结果：



Lab03.py：

|  |
| --- |
| import os |
| import yara |
| import time |
| # 定义YARA规则 |
| rules = yara.compile("Lab03.yara") |
| # 扫描的样本文件夹路径 |
| sample\_folder = "sample" |
| # 初始化变量 |
| matched\_files = 0 |
| start\_time = time.time() |
| # 遍历样本文件夹中的文件 |
| for root, dirs, files in os.walk(sample\_folder): |
| for file\_name in files: |
| file\_path = os.path.join(root, file\_name) |
| try: |
| # 使用YARA规则扫描文件 |
| matches = rules.match(file\_path) |
| # 如果有匹配的规则，记录匹配的文件数量 |
| if matches: |
| matched\_files += 1 |
| except Exception as e: |
| # 处理可能的异常 |
| pass # 这里可以加入处理异常的代码 |
| # 计算扫描时间 |
| end\_time = time.time() |
| scan\_time = end\_time - start\_time |
| # 输出结果 |
| print("样本文件夹中的文件数量：{}".format(len(files))) |
| print("匹配的文件数量：{}".format(matched\_files)) |
| print("扫描时间：{:.2f} 秒".format(scan\_time)) |

Lab03.yara

|  |
| --- |
| rule Detect\_Lab03 { |
| meta: |
| description = "YARA rule to detect Lab03 malicious code" |
| strings: |
| $url = "www.practicalmalwareanalysis.com" |
| $registry\_key = "HKEY\_MACHINE\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run\\VideoDriver" |
| $mutex = "WinVMX32" |
| $file\_creation = "C:\\WINDOWS\\system32\\vmx32to64.exe" |
| $export\_function = "installA" |
| $svchost\_exe = "svchost.exe" |
| $practicalmalwareanalysis\_log = "practicalmalwareanalysis.log" |
| $keyboard\_strings = /\\[CTRL]|\\[ENTER\\]|\\[BACKSPACE\\]|\\[TAB\\]|\\[DEL\\]|\\[CAPS LOCK]/ |
| $cmd = "cmd.exe" |
| $del = "/c del" |
| $http\_string = "HTTP/1.0" |
| $command\_strings = /DOWNLOAD|UPLOAD/ |
| $regedit = "SOFTWARE\\Microsoft \\XPS" |
| condition: |
| any of ($url, $registry\_key, $mutex, $file\_creation,$export\_function,$svchost\_exe,$practicalmalwareanalysis\_log,$keyboard\_strings,$cmd,$del,$http\_string,$command\_strings,$regedit) |
| } |