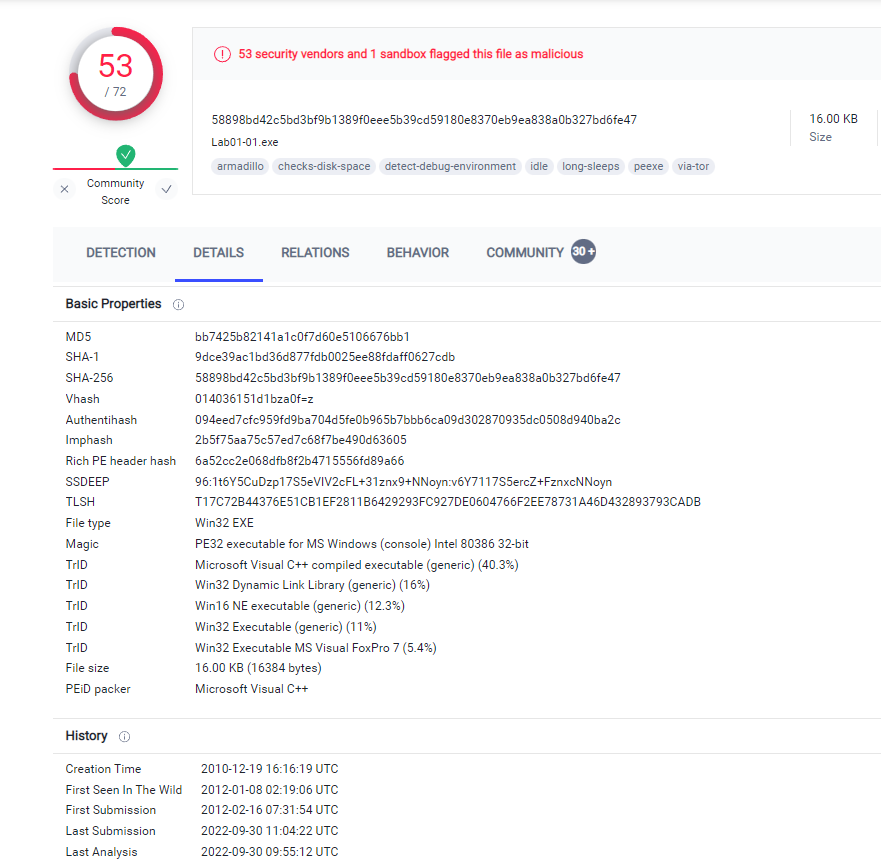
**恶意代码分析与防治技术实验报告**

**Lab1**

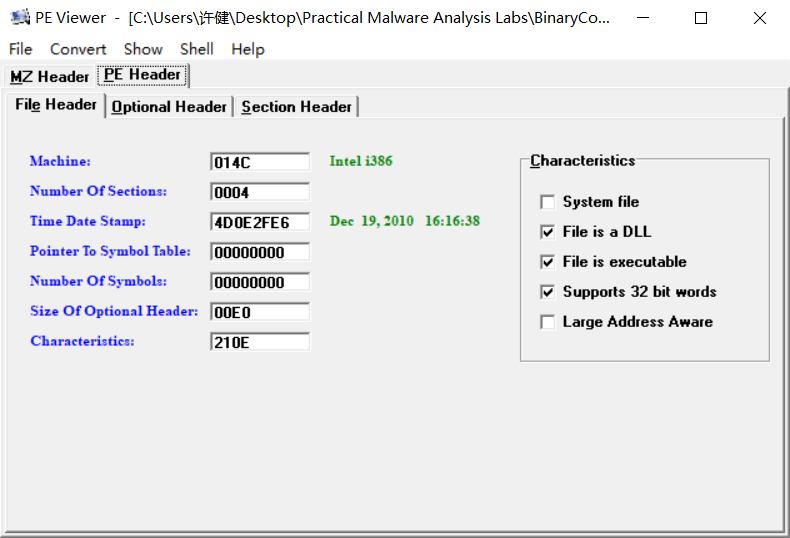
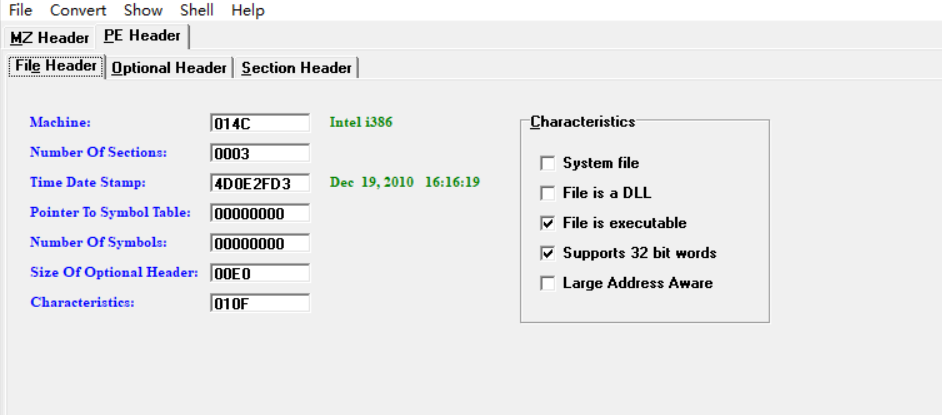
**学号：许健 姓名：2013018 专业：信息安全**

1. **Lab1-1**

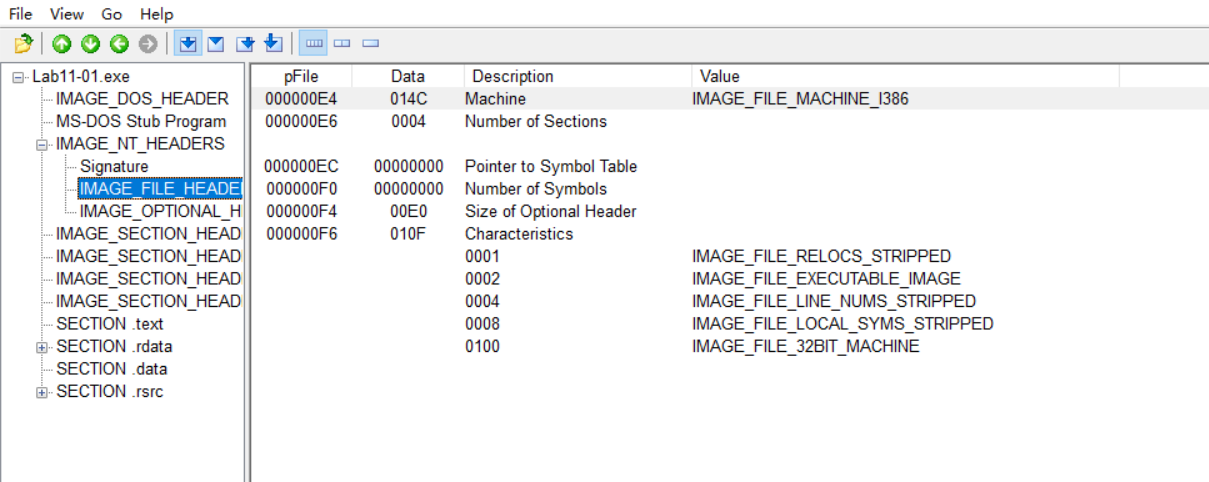
在VirusTotal.com上匹配到特征码，History中文件最早编译时间为2010.12.19



使用PE Viewer可以看到Time Data Stamp，两个文件编译的时间在一分钟之内

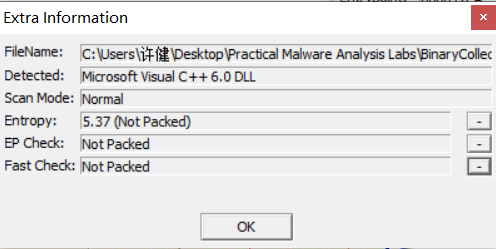
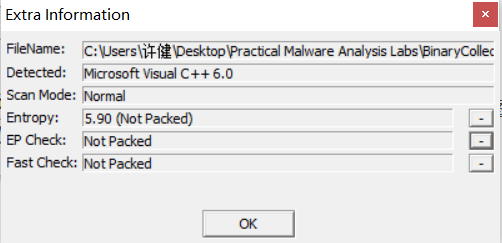


遇到的问题：使用PEView似乎看不到时间戳，可能是软件的bug



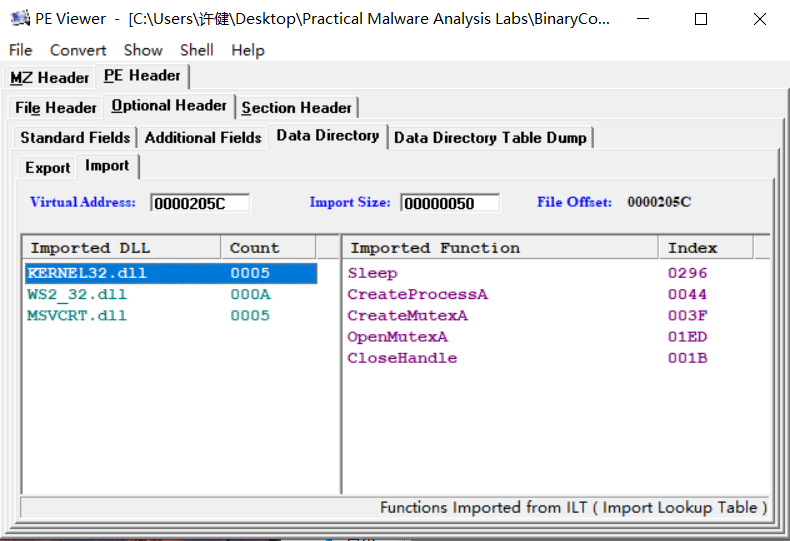
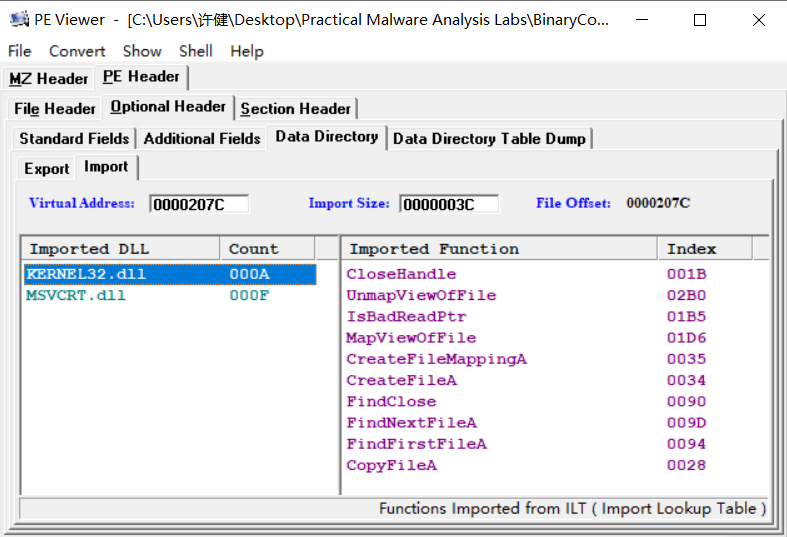
使用PEiD查看，没有找到加壳或混淆过的迹象

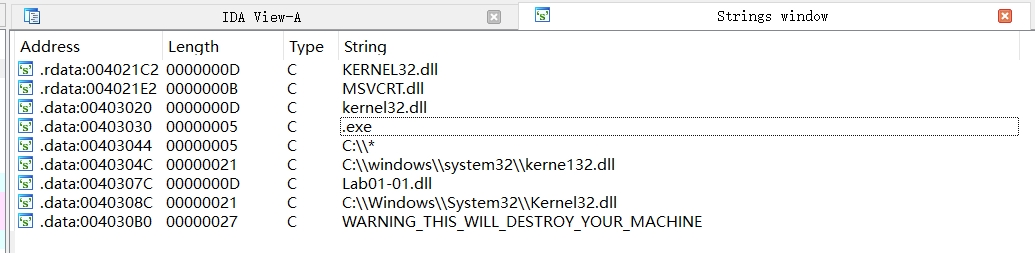
还可以看到.exe文件是由Microsoft Visual C++ 6.0编译的



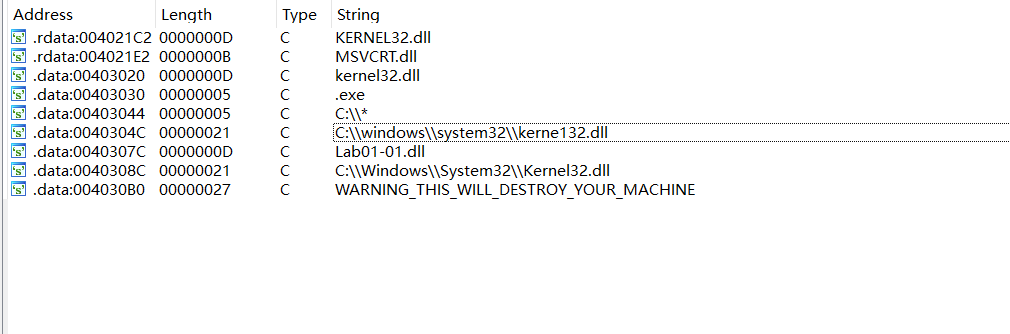
使用PE Viewer查看导入表，Lab01-01.exe中包含了FindFirstFileA、FindNextFileA、CopyFile函数，猜测程序是在搜索文件系统和复制文件。Lab01-01.dll中导入CreateProcessA和Sleep函数（这两个函数普遍在后门程序中使用），并且导入了WS2\_32.dll用于提供联网功能。

使用IDA Pro查看Strings window，看到.exe字符串，因此该程序的功能很可能是搜索目标系统上的可执行文件。





可以检查C:\Windows\System32\kerne1.32来发现额外的恶意活动，这个文件试图伪装成系统文件，可以用来在主机作为恶意代码感染的迹象进行搜索。



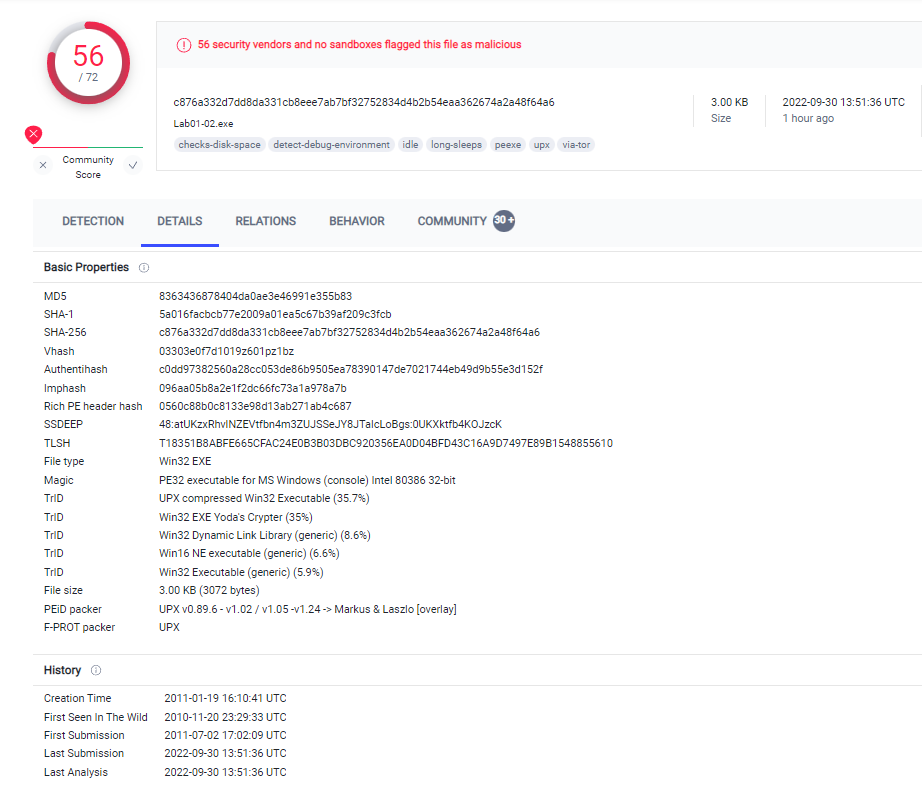
查看.dll的导入表和字符串列表，可以看到文件中包含了一个私有子网ip地址127.26.152.13的字符串（用于教学而不是恶意破坏，否则就是可路由的公网ip地址），这是一个很好的基于网络的恶意代码感染迹象。

exec字符串可能是通过网络来给后门程序传送命令，让它通过CreateProcess函数运行一个程序，sleep字符串可能用于命令后门程序进行休眠模式。（恶意代码较复杂，参考了附录C的解析）

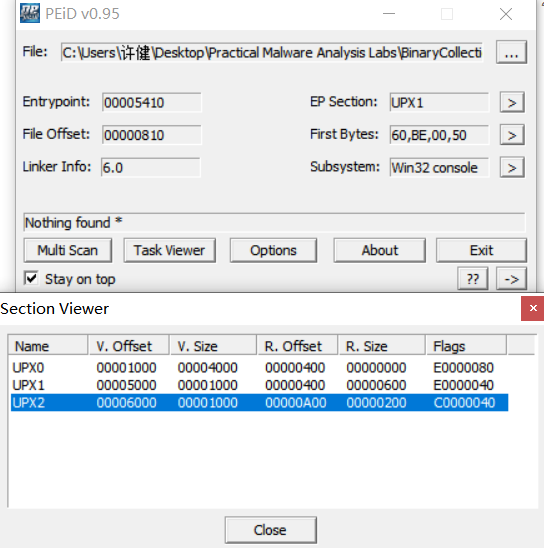


1. **Lab1-2**

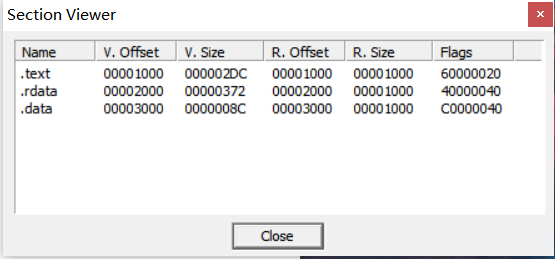
在VirusTotal.com上匹配到72个反病毒引擎中的56个病毒特征码，VirusTotal.com



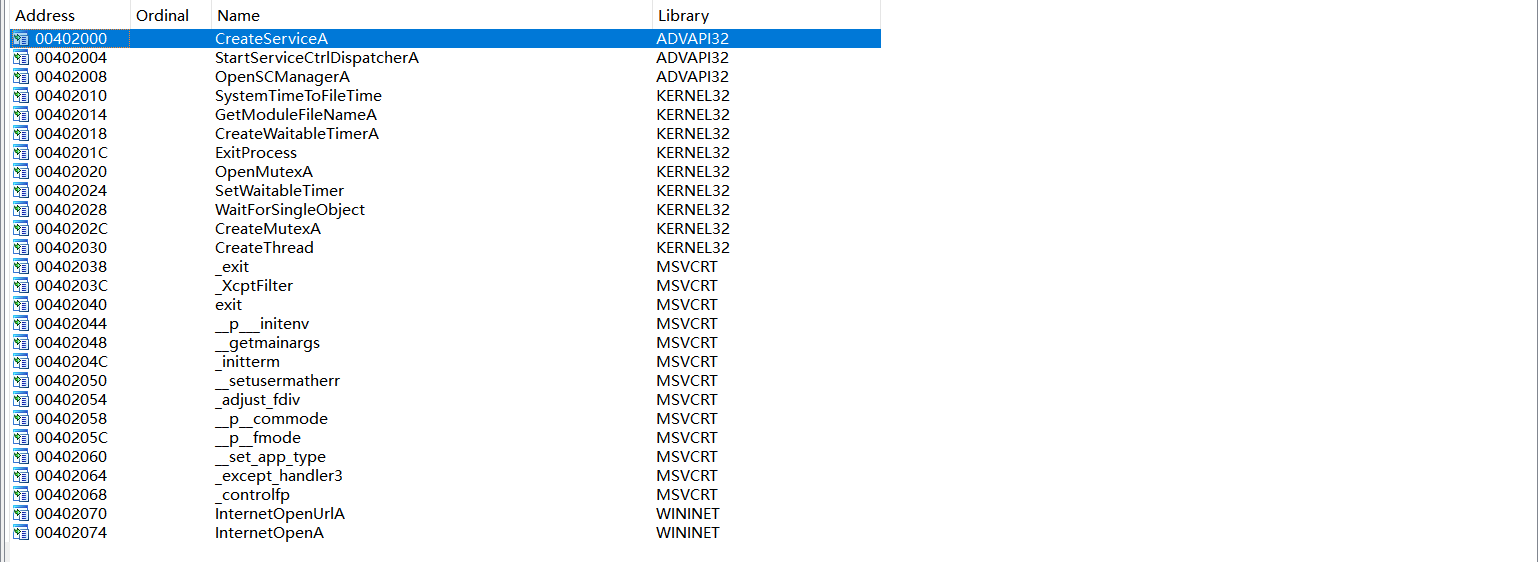
使用PEiD检测，有迹象表明文件是加壳的，比如名为UPX0、UPX1、UPX2的节，可见加壳类型为UPX。



下载UPX工具并尝试进行脱壳，可以看到正常的节信息，说明已脱壳



使用IDA Pro查看导入表和字符串列表，试图寻找基于主机或基于网络的迹象来确定被恶意代码感染的机器。





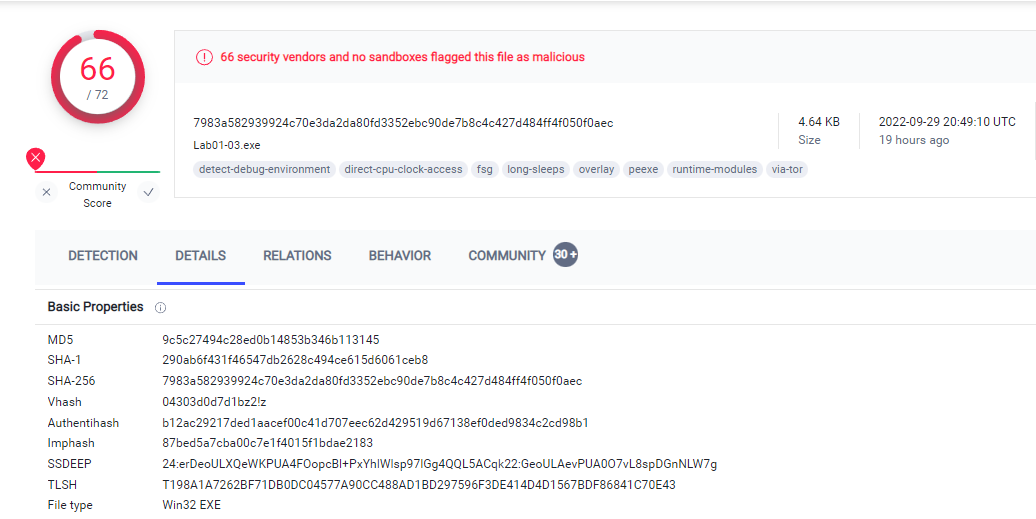
从wininet.dll导入的函数说明恶意代码会进行联网操作(InternetOpen和IntelnetOpenURL)，从advapi32.dll的导入函数(CreateService)说明这个代码会创建一个服务。

查看字符串列表，可以看到http://www.malwareanalysisbook.com，这可能是IntelnetOpenURL函数所打开的URL，还有Malservice字符串可能是创建的服务名称。

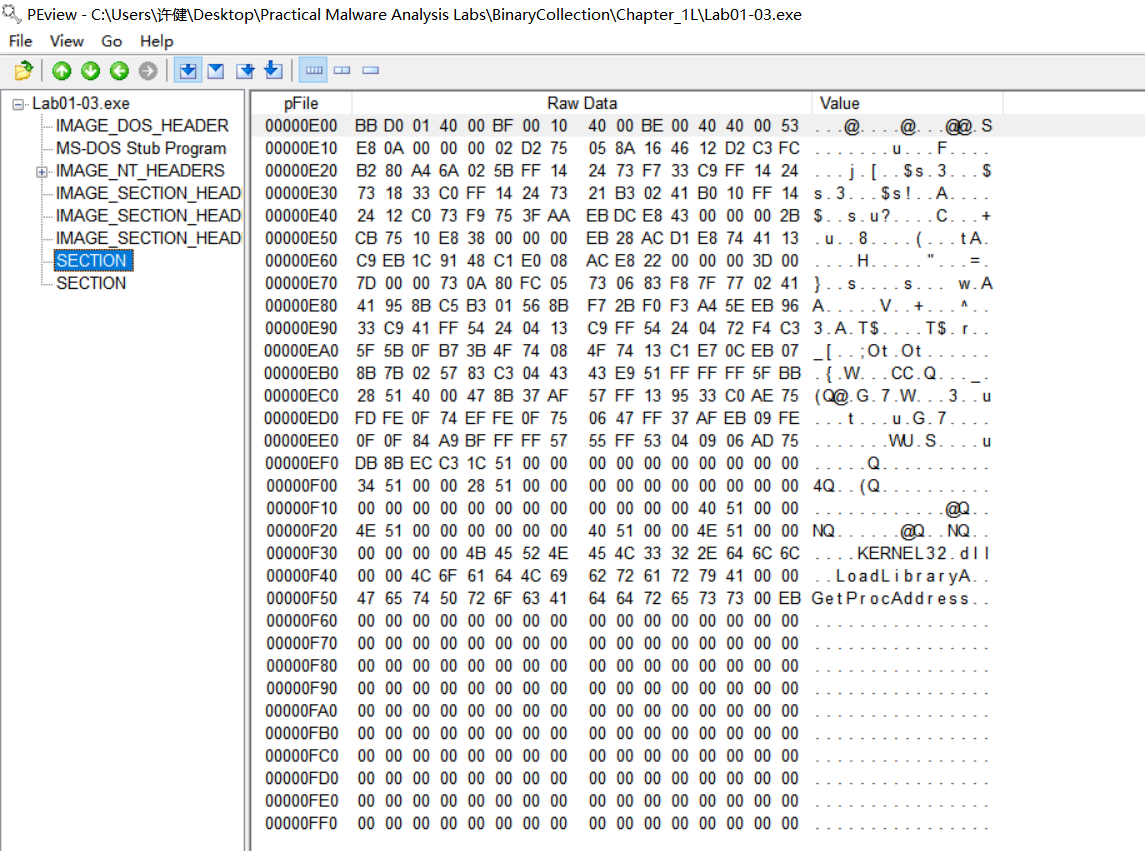
因此想要检查被恶意代码感染的主机，可能的思路是注意名为Malservice的服务以及通过http://www.malwareanalysisbook.com的网络流量。

1. **Lab1-3**

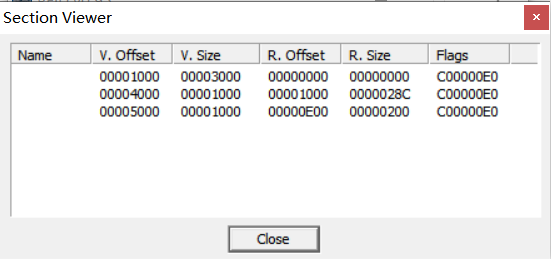
72个反病毒引擎中有66个将文件标记成恶意代码，并且分析出是用FSG加壳的。



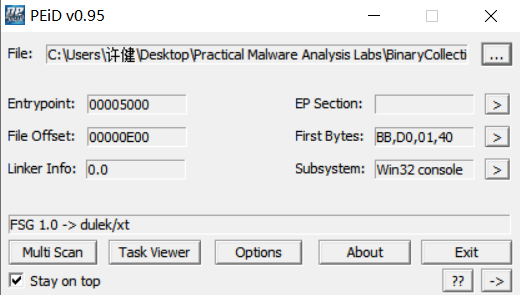
使用PEview看不到节名



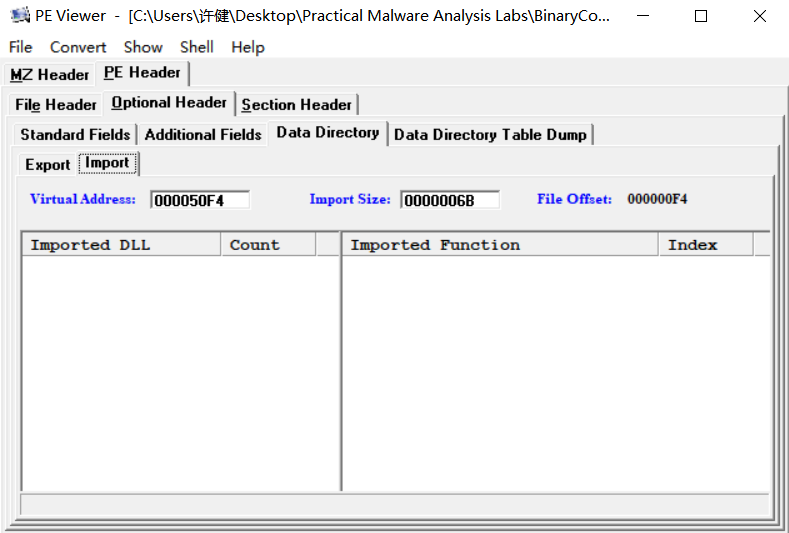
首节的虚拟大小为0x3000，但是原始数据大小却为0



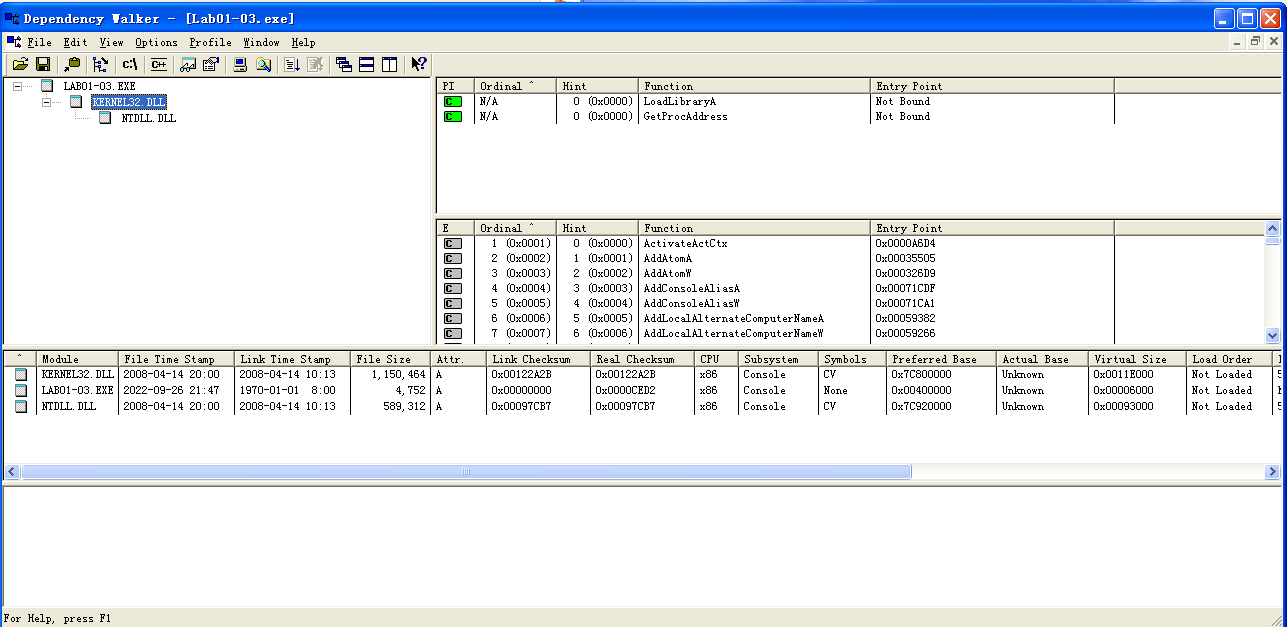
PEiD将加壳器标识为FSG 1.0 -> dulek/xt



使用PE Viewer显示没有与导入表有关的信息



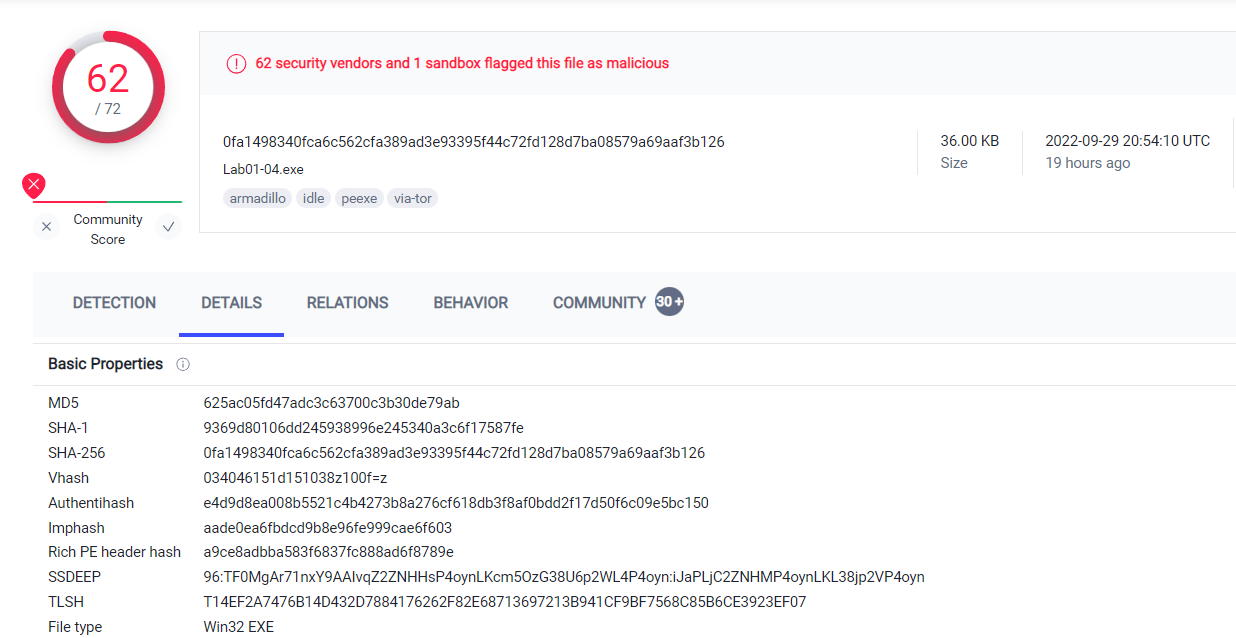
使用Dependency walker打开Lab1-3.exe，只能看到导入函数LoadLibrary和GetProcAddress。一般加壳软件只有这两个导入函数，进一步说明这个文件是被加过壳的。



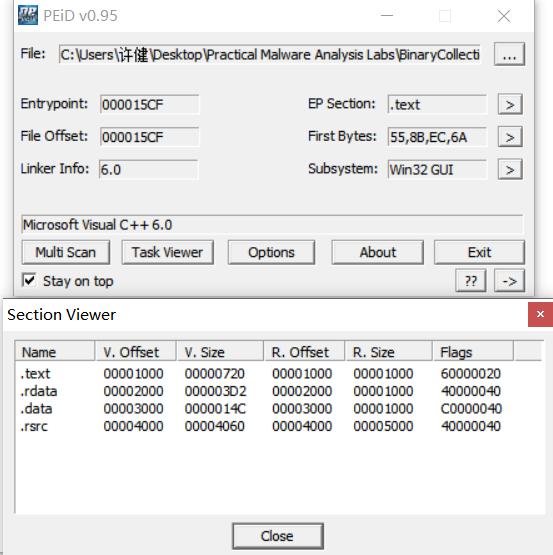
由于文件是用FSG进行加壳的，目前我们无法对其手工脱壳，因此也就无法寻找基于主机或基于网络的迹象来确定被恶意代码感染的机器。

1. **Lab1-4**

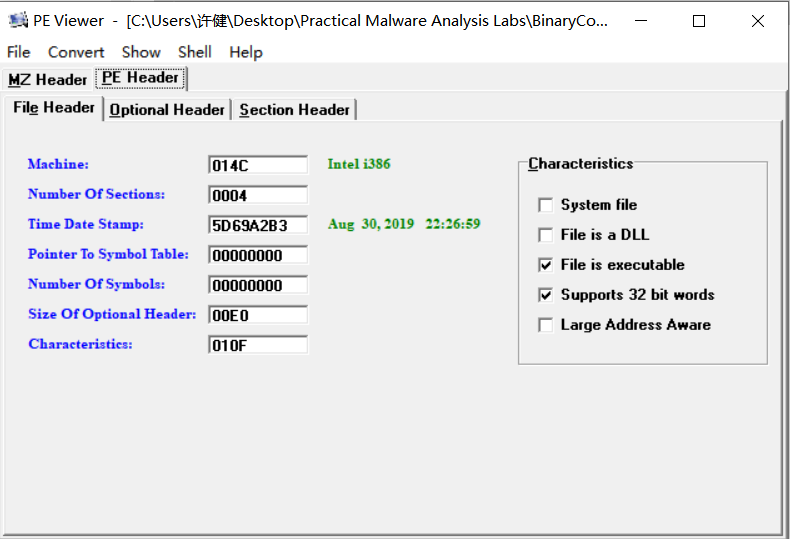
72个反病毒引擎中有62个将文件标记成恶意代码



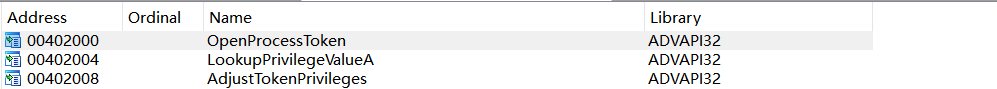
使用PEiD检测可以看到正常的节信息，没有显示迹象表明这个文件是加壳或是混淆的。



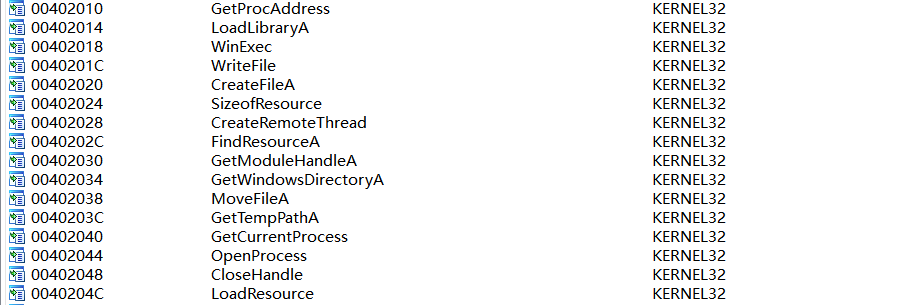
打开PEViewer.，通过时间戳可以看到文件是2019年8月30号编译的，显然是伪造的，我们不能确认这个文件究竟是什么时候编译的。



从advapi.dll导入的函数表明程序在做一些与权限有关的事，可能是访问一些受保护的文件。



kernel32.dll的导入函数说明程序从资源节中装载数据(LoadResource、FindResource和SizeofResource)，并写一个文件到磁盘(CreateFile和WriteFile)，接着执行一个磁盘上的文件(WinExec)。

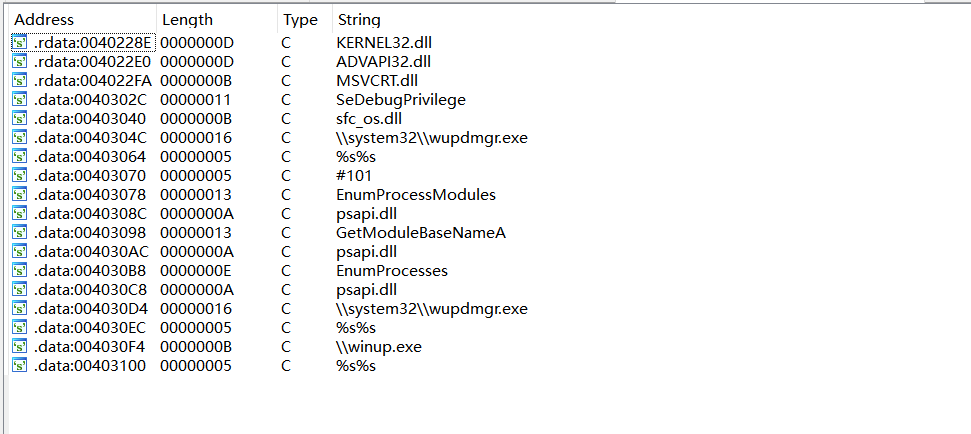


在Strings window中得到的可疑字符串包括：

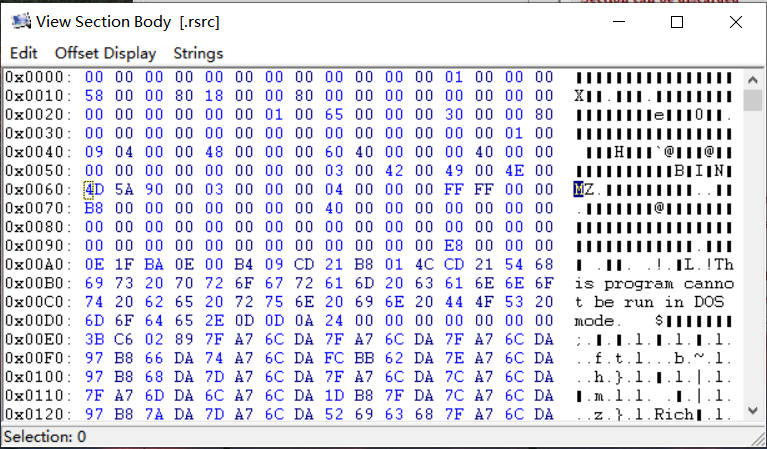
\system\wupdmgr.exe程序可能在这个位置创建或修改文件

<http://www.practicalmalwareanalysis.com/updater.exe额外恶意代码的下载地址>

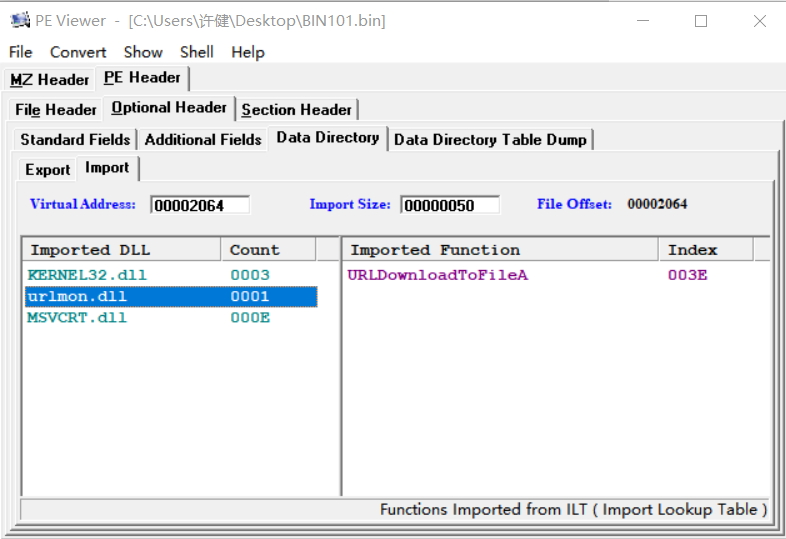
因此这个恶意代码可能会下载一个新的恶意代码并存放在\system\wupdmgr.exe处



同时我们在PE文件的.rsrc节发现一个有趣的字符串“!This program cannot be run in DOS mode.”(还有熟悉的MZ头4D5A)，这个字符串是在所有PE文件开始处的DOS头部中都会含有的错误消息，因此我们可以推断该文件的资源节中存储了另外一个可执行程序。



我们尝试用Resource Hacker将资源存储为二进制文件，然后用PE Viewer打开这个文件，查看导入表，可以看到URLDownloadFileA函数，说明嵌入文件访问了一些网络函数，恶意下载了一些东西并使用WinExec函数执行。



1. **实验感想**

**由于对PE文件结构以及Windows系统函数的不理解，所以借助着实验文档对Lab1的四个恶意代码小实验进行了实操，在自己亲自动手分析恶意代码的时候，我发现扎实的基础和对底层知识的理解是有多么重要。同时通过实验我对于PE文件的格式有了更深的了解，并熟练了一些常用的工具的使用。我的感悟是在今后的学习中要注意细节，吃透知识点并进行实操演练，在不借助参考文档的情况下即可自主分析出恶意代码的攻击原理，那样才能达到一个基本满意的水平。**