实验 4 程序静态分析技术基础

实验目的

掌握程序静态分析技术，能够使用工具初步分析恶意代码。

实验前提

请安装虚拟机 winXPenSP3 调试环境。

本节课所用到的文件均存放在windXPenSP3 调试环境虚拟机 C:\workspace\PracticalMalwareAnalysis-Labs\Practical Malware Analysis Labs\BinaryCollection 目录的Chapter_1L中。

注意：请在分析样本前，在C:\workspace\ 下建立一个tmp临时文件夹，存放要分析的文件。因为某些分析过程可能会破坏原文件。

实验内容

- 1.分析Lab01-01.exe和Lab01-01 .dll。
- 2.分析Lab01-02.exe文件。
- 3.分析La601-03.exe文件。
- 4.分析Lab01-04.exe文件。

实验步骤

一.分析Lab01-01.exe和Lab01-01 .dll。

1.将文件上传至http: //www.VirusTotal.com分析并查看报告。

2.查看这些文件，回答它们是什么时候编译的?

> 提示：这里可以使用 PEview 工具，浏览PE信息 IMAGE_NT_HEADERS->IMAGE_FILE_HEADER->Time Date Stamp字段。这个字段中的值可以告诉我们该文件于何时被编译。

> 这两个文件的编译时间如果距离很近，说明它们是一起生成的，可能存在相互关联关系。

3.检查文件并判断这两个文件是否被混淆处理或加壳处理？

> 提示：1.使用strings工具，可知这两个文件中存在着大量可读字符串；2.两个文件的导入表（函数）数量很少，说明是小程序；3.文件都有适当大小的节（.text,.data,...）；4.使用PEiD工具，显示未被加壳。综上，可知他们是未被混淆和加壳的小程序。

4.这两个文件是否有导入函数能显示出这些恶意代码是做什么的? 有哪些导入函数?

> 提示：使用Dependency Walker工具检查这两个文件；msvcrt.dll导入的一些函数，这些函数通常是被每个exe所包含的，因为他们是exe程序的包装代码。kernel32.dll导入的函数中有打开和操作文件的函数，有 FindFirstFile、FindNextFile函数，这说明这个exe程序可能会搜索文件，打开并修改。从字符串“.exe”分析，可猜测该恶意代码会搜索某些可执行程序。导入WS2_ 32.dll中的函数，这些函数提供了联网功能。

5.是否有任何其他文件或基于主机的迹象，让你可以在受感染系统上查找?

> 提示：使用strings工具检查字符串，发现在Lab01-01.exe中有kerne132.dll（请注意，文件kernel32.DLL用数字1代替了字母1），这是为了看起来像是系统文件kernel32.dll。这个文件可以用来在主机作为恶意代码感染的迹象进行搜索。后续分析可以根据这一点查看代码污染线索。

6.是否有基于网络的迹象，可以用来发现受感染机器上的这个恶意代码?

> 提示：使用strings工具检查字符串，Lab01-01.dll文件中包含有私网子网IP地址：127.26.152.13的字符串，预示着这个程序是为了教学目的而不是真正恶意目的而编制的，如果这是真正的恶意代码，这个IP地址应该是可路由的公网IP地址，它会是一个很好的基于网络的恶意代码感染迹象，可以用来识别这个恶意代码。

> 提示：查看Lab01-01.dll的导入表(使用Dependecy walker工具)和字符串列表(使用strings工具)，导入了WS2_32.dll的一些函数。这些导入函数是按照序号进行导入的，但不清楚是那些函数。

> 提示：在kernel32.dll中导入的函数有 CreateProcess 和 Sleep ，这两个函数在后门程序中常见，他们若在exec与sleep字符串结合使用时，需要特别关注。exec字符串可能是通过网络给后门发送指令，然后通过 CreateProcess 创建一个进程。Sleep 字符串可能用于命令后门程序进入休眠状态。

7.你猜这些文件的目的是什么?

> 提示：.dll可能是一个后门，运行态可能名为kerne132.dll ；.exe可能是用来安装和运行DLL文件的。

二.分析**Lab01-02.exe**文件。

1.将Lab01-02.exe文件上传至http: //www.VirusTotal.com进行分析并查看报告。文件匹配到了已有的反病毒软件特征吗?

2.是否有这个文件被加壳或混淆的任何迹象?如果是这样，这些迹象是什么?如果该文件被加壳，请尽可能进行脱壳。

> 提示：使用PEview[具打开这个文件，最明显的迹象是名为UPX0、UPX1和UPX2的节，明显是由UPX进行加壳后恶意代码程序的节名称。

> 提示：使用PEiD查壳，当然结果可能不准确或未知。

> 提示：可以注意到恶意代码拥有一个过小的导入表，而且第一个节.UPX0，虚拟大小为0x4000，而原始数据大小却为O。UPX0是长度最大的节，标记为可一执行的，因此其中很可能包含了原始的未加壳代码。

> 提示：https://upx.github.io 可下载UPX脱壳软件。使用命令 `upx -d <原有壳文件名> -o <脱壳后新文件名>`

3.有没有任何导入函数能够暗示出这个程序的功能?如果有，是哪些导入函数，它们会告诉你什么?

> 提示：脱壳之后，我们开始审查脱壳恶意代码的导入表和字符串列表。这个恶意代码的导入函数都是从kernel32.dll和msvcrt.dll，而这些函数几乎被每个程序都导入，所以它们能够告诉我们关于这个恶意代码的信息很少。从wininet.dll导入的函数告诉我们，这个恶意代码会进行联网操作(Internetopen和工nternetOpenURL)，从advapi32.dll的导入函数(CreateService)告诉我们这个代码会创建一个服务。

4.哪些基于主机或基于网络的迹象可以被用来确定被这个恶意代码所感染的机器?

> 提示：在检查字符串列表时，我们看到www.malwareanalysisbook.com，这可能是InternetopenURL 函数中所打开的URL，以及Malservice字符串，这可能是所创建的服务名称。虽然我们还不能肯定这个程序会做什么，但我们已经发现一些线索，能够帮助我们在网络上和系统里来搜索这个恶意代码。

三.分析La601-03.exe文件。

1.将Lab01-03.exe文件上传至http: //www.VirusTotal.com进行分析并查看报告。文件匹配到了已有的反病毒软件特征吗?

2.是否有这个文件被加壳或混淆的任何迹象?如果是这样，这些迹象是什么?如果该文件被加壳，请进行脱壳。

> 提示：当我们使用PEview 打开文件时，一些迹象表明这个文件是加壳的。首先是文件的竹并没有名字。此外，首节的虚拟大小为0x3000，但原始数据大小却为0。我们运行PEiD工具进行确认，它

> 将加壳器标识为FSG 1.0->dulek /xt。

3.有没有任何导入函数能够暗示出这个程序的功能?如果是，是哪些导入函数，它们会告诉你什么?

> 提示：为了确认这个文件是被加壳的，我们使用PEview搜索.rdata,但却没有发现任何节里有导入表。一个没有导入表的可执行文件是极为罕见的。

> 提示：PEview无法正常处理这个文件，使用Dependency Walker打开这个文件，看到它拥有一个导入表，但其中只有两个函数:LoadLibrary和GetProcAddress。加壳文件往往只有这两个导入函数，这进一步表明了这个文件是被加过壳的。

> 提示：可以尝试使用UPX来对这个文件进行脱壳，但我们知道这个文件是由FSG进行加壳的，而不是UPX。目前无法脱壳，所以本题暂时无法回答本体和后续问题。

4.有哪些基于主机或基于网络的迹象，可以被用来确定被这个恶意代码所感染的机器?

四.分析Lab01-04.exe文件。

1.将Lab01-04.exe文件上传至http: //www.VirusTotal.com分析并查看报告。文件匹配到了已有的反病毒软件特征吗?

2.是否有这个文件被加壳或混淆的任何迹象?如果是这样，这些迹象是什么?如果该文件被加壳，请进行脱壳，如果可能的话。

> 提示：PEview没有迹象表明这个文件是加壳或是混淆的。

3.这个文件是什么时候被编译的?

> 提示：这里可以使用 PEview 工具，浏览PE信息 IMAGE_NT_HEADERS->IMAGE_FILE_HEADER->Time Date Stamp字段。这个字段中的值可以告诉我们该文件于何时被编译。但是这个数值是可以被伪造的。

4.有没有任何导入函数能够暗示出这个程序的功能?如果是，是哪些导入函数，它们会告诉你什么?

> 提示：从 advapi32.dll导入的函数，表明程序在做一些与权限有关的事情。导入函数WinExec和WriteFile，以及virusTotal.com的结果，告诉我们这个程序会写一个文件到磁盘上，然后执行它。导入函数还有一些是用于从文件的资源节中读取信息的。从advapi32.dll导入的函数告诉我们，程序做了一些与权限有关的事情。我们可以假设它试图访问使用了特殊权限进行保护的文件。从kenael32. dll的导入函数告诉我们这个程序从资源节中装载数据(LoadResource, FindResource和SizeofResource)，并写一个文件到磁盘上(CreateFile和WriteFile)，接着执行一个磁盘上的文件(WinExec)。我们也可以猜测这个程序将文件写入到系统目录，因为它调用了GetWindowsDirectory函数。

5.有哪些基于主机或基于网络的迹象，可以被用来确定被这个恶意代码所感染的机器?

> 提示：通过检查字符串，我们可以看到了字符串 www.malwareanalysisbook.com/updater.exe 。通过检查字符串，我们看到了www.malwareanalysisbok.com/updater.exe, 这可能是保存下载恶意代码的网络位置。我们还看到，字符串 \system32\wupdmgr.exe， 结合GetWindowsDirectory函数调用，这表明恶意代码在 C:\Windows\system32\wupdmgr.exe 位置创建或者修改了一个文件。

6.这个文件在资源段中包含一个资源。使用Resource Hacker工具来检查资源，然后抽取资源。从资源中你能发现什么吗?

> 提示：现在我们已经有比较充分的证据，知道这个恶意文件下载了一个新的恶意代码。我们已经知道它是从哪个位置下载恶意代码的，并且可以猜到它会将下载到的恶意代码存储在什么位置，我们现在唯一还没明白的怪异现象，就是这个程序看起来并没有在访问任何网络函数。这个恶意代码样本最有趣的就是它的资源节，当我们使用Resource Hacker工具打开这个恶意代码时，我们只看到了一个资源。Resource Hacker工具识别这个资源的类型是二进制，说明是任意的二进制数据，但我们查看数据时，它的绝大部分都是无意义的，唯一例外的是字符串!This program cannot be run in DOS mode,这个字符串是在所有PE文件开始处的DOS头部中都会包含的错误提示。我们可以认为这一资源其实是在Lab01-04.exe资源节中存储的另一个可执行文件。这种技术在恶意代码中使用得非常普遍。

> 提示：在Resource Hacker继续分析，我们单击Action->Save resource as binary file, 存储完资源后，我们使川PEview来打开这个文件，分析内嵌的文件。通过查看导入表，我们看到嵌入文件在访问一些网络函数，它调用了URLDownloadToFile,一个恶意下载器普遍使用的函数。它还调用了WinExec函数，可能执行了下载到的文件。