**恶意代码分析与防治技术实验报告**

**Lab6**

**学号：2013018 姓名：许健 专业：信息安全**

1. **实验内容**

完成课本Lab6的实验内容，编写Yara规则，并尝试IDA Python的自动化分析。

1. **问题解答**
2. Lab6-1
3. 由main函数调用的唯一子过程中发现的主要代码结构是什么？

**基本静态分析**：

InternetGetConnectedState函数，检查本地系统的Internet连接状态，当存在一个可用的Internet连接时，该函数返回1，否则返回0。

利用Windows Internet (WinINet) API，应用程序可以使用HTTP协议访问Internet资源。

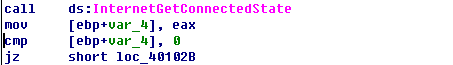


**基本动态分析**：

运行程序，打印“success:INnternet Connection”就退出了。

**高级静态分析**：

使用IDA Pro加载做完整分析，main函数调用sub\_401000，查看该函数的流程图。根据InternetGetConnectedState调用的结果，出现两条不同的路径。主要的代码结构是if语句。



1. 位于0x40105F的子过程是什么？

在函数中看到\_\_stbuf、\_\_ftbuf和文件描述符，经过比对确定位于0x40105F处的子例程是printf。

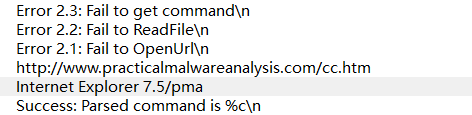
1. 这个程序的目的是什么？

该函数检测是否有一个可用的Internet连接。

1. Lab6-2
2. main函数调用的第一个子过程执行了什么操作？

**基本静态分析**：

从字符串推测该程序可能会打开一个网页，并解析一条指令。其中出现一个域名，可以当作网络特征来用。



导入表包含一些用于网络功能的Windows API。WinINet库提供了通过网络使用HTTP的一组简单的函数。



**基本动态分析**：

WinInet库通常使用HTTP，选择监听80端口。

可以捕获到对www.practicalmalwareanalysis.com的DNS请求，然后程序会使用一个URL请求网页。

**高级静态分析**：

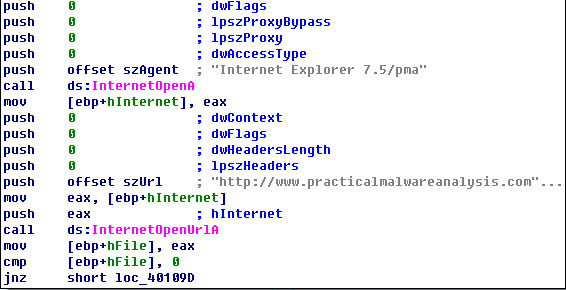
调用的第一个子过程与Lab6-1中的一样，检查是否存在可用的Internet连接。

1. 位于0x40117F的子过程是什么？

和之前一样，printf函数。

1. 被main函数调用的第二个子过程做了什么？

查看sub\_401040函数的graph view，可以看到一些字符串和域名，分析代码的行为

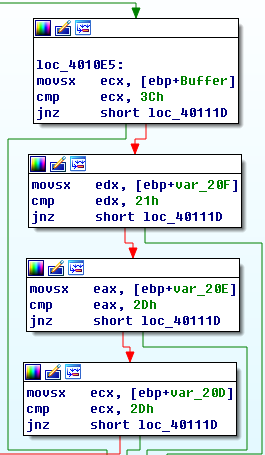


函数压入了一个值Internet Explorer 7.5/pma（HTTP协议里面的user agent），然后调用函数InternetOpenA函数，那这个值其实就传递给了InternetOpenA。

InternetOpenA的结果被赋给了hFile，并与0进行比较，如果它等于0函数会返回。否则hfile变量会传给InternetReadFile。hFile变量实际上是一个句柄——一种访问已经打开东西的途径，而这个句柄是用来访问URL的。

InternetReadFile从InternetOpenA打开的网页中读取内容。代码会将buffer和字符串<!--（HTML注释的开始部分）作检查，如果匹配上第5个字符就会被移到AL并从这个函数返回。

在网页中HTML注释不会被显示，但是可以通过查HTML页面源码看到，这种通过HTML注释隐藏指令的方法经常被攻击者用于向恶意代码发送指令，让恶意代码看起来是在访问一个正常的网页。



1. 在这个子过程中使用了什么类型的代码结构？

该子过程使用的结构是就是一个字符数组，InternetReadFile会将读到的字符流，写入这个数组中，然后一个一个的比较这个字符的头4个字符，如果等于<!--就将第5个字符用al返回。

1. 在这个程序中有任何基于网络的指示吗？

Internet Explorer 7.5/pma（HTTP协议里面的user agent）

从[www.practicalmalwareanalysis.com/cc.htm](http://www.practicalmalwareanalysis.com/cc.htm)下载网页

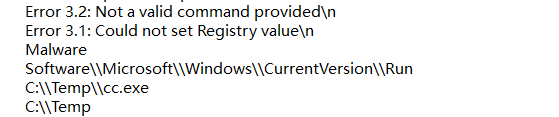
1. 这个恶意代码的目的是什么？

首先检查有无可用网络连接。恶意代码的目的是从恶意网址上获得一个要执行的远程命令，然后休眠一分钟，如果没有获得这个命令，则退出。

1. Lab6-3
2. 比较在main函数与实验6-2的main函数的调用。从main函数中调用的新的函数是什么？

**基本静态分析**：

多了一些与注册表相关的信息，可能将恶意程序添加到自启动选项。C:\\Temp\\cc.exe可能是一个有用的本地特征。



查看导入表也多了一些新函数，在注册表中插入信息。



**高级静态分析**：

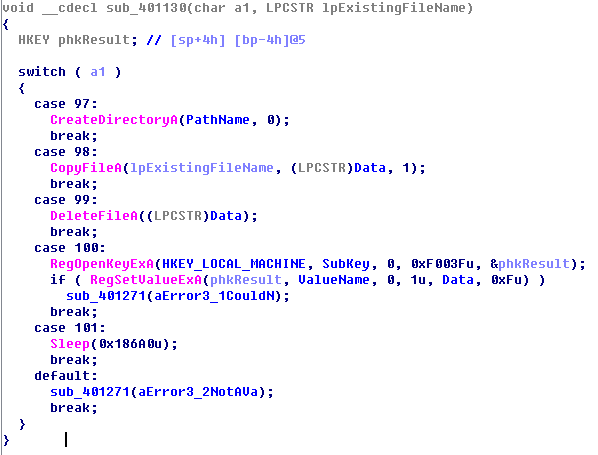
查看main函数调用图，多了sub\_401130函数，与修改注册表相关。

1. 这个新的函数使用的参数是什么？

参数是对当前运行程序名字字符串的引用和从HTML注释中解析出的指令字符

1. 这个函数包含的主要代码结构是什么？

Switch结构，是一个跳转表。



1. 这个函数能够做什么？

查看switch的每一个选项

A选项调用CreateDirectory，使用的参数是C:\\Temp

B选项调用CopyFile，目的文件是C:\\Temp\\cc.exe，源文件是当前程序

C选项调用DeleteFile，删除C:\\Temp\\cc.exe

D选项在注册表设置C:\\Temp\\cc.exe自启动

E选项休眠100秒

default选项打印“Error 3.2: Not a valid command provided”

1. 在这个恶意代码中有什么本地特征吗？

注册表键Software\Microsoft\Windows\CurrentVersion\Run\Malware

文件路径C:\Temp\cc.exe

1. 这个恶意代码的目的是什么？

会在本地创建一个C:\\Temp文件夹和创建一个C:\\Temp\\cc.exe文件，然后会在注册表中Software\Microsoft\Windows\CurrentVersion\Run中添加一个值C:\Temp\cc.exe。

1. Lab6-4
2. 在实验6-3和6-4的main函数中的调用区别是什么？

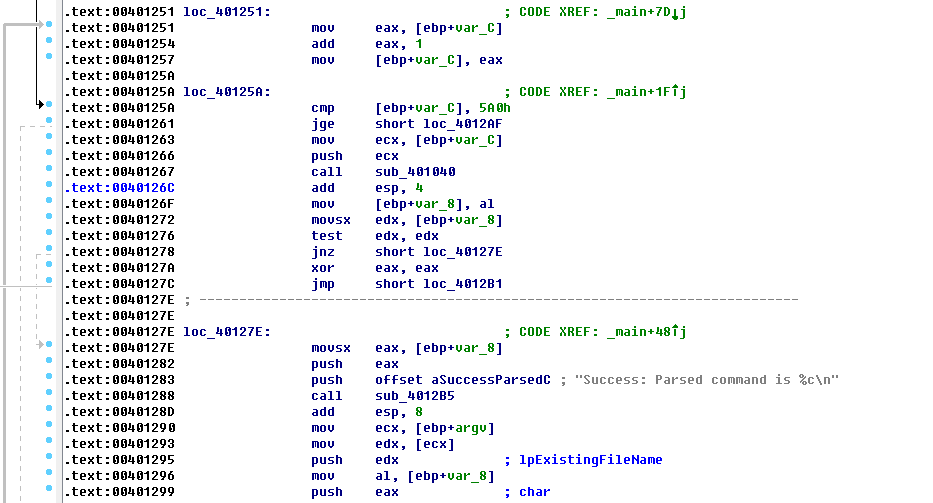
**基本静态分析**：

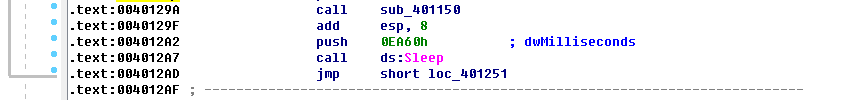
该程序会动态生成User-Agent



**高级静态分析**：

查看main函数的graph view，出现了循环。



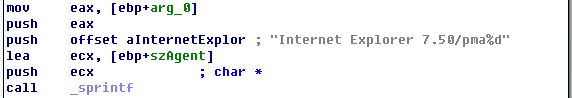


1. 什么新的代码结构已经被添加到main中？

在main函数里增加了循环结构

1. 这个实验的解析HTML的函数和前面实验中的那些有什么区别？

增加了一个参数，arg\_0是从main函数中传入的计数器（var\_C）。会调用sprintf函数来对输出进行格式化，也就是会改变User-Agent的值。



1. 这个程序会运行多久？（假设它已经连接互联网）？

局部变量var\_c用于循环计数，初始化为0，如果var\_C大于或等于0x5A0（1440），循环就会终止，每次循环都会休眠一分钟，该过程一共会持续1440分钟，也就是24小时。

1. 在这个恶意代码中有什么新的基于网络的迹象吗？

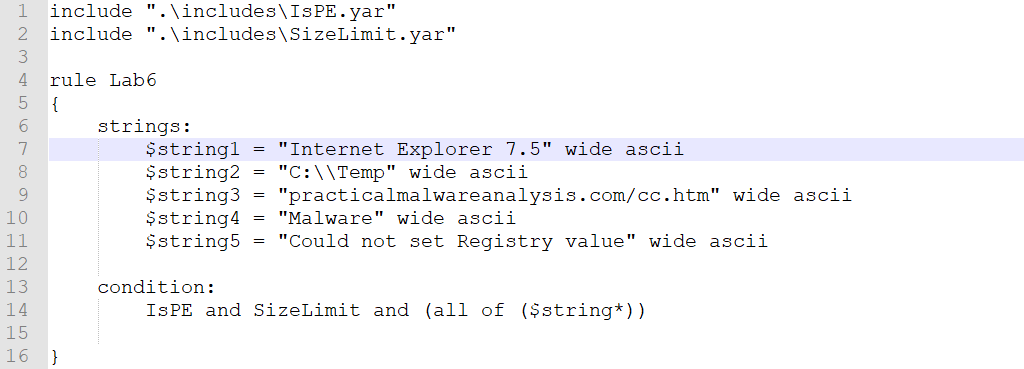
新的基于网络的迹象就是这个程序运行次数会写在User-Agent里面，我们可以通过这个来确定这个程序已经运行了多少分钟多少次了。

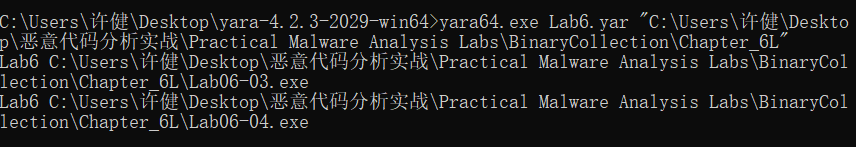
1. 这个恶意代码的目的是什么？

首先检查有无可用网络连接。恶意代码的目的是从恶意网址上获得一个要执行的远程命令，然后休眠一分钟，如果没有获得这个命令，则退出。（增加了for循环部分）

1. **编写Yara规则**

由于Lab6 1-4实验循序渐进，所以针对具有恶意代码行为的Lab6-3和Lab3-4编写Yara规则检测，检测用的string都是之前分析提到的特征字符串。





1. **编写IDAPython脚本**

编写一个帮助识别循环的IDA Python脚本，循环结构最后一条指令一般都是一条往回的跳转指令，我们的目标是识别出这样的指令。编写脚本如下：

def IsJmp(addr):

ls = ['JB','JNB','JE','JNE','JBE','JA','JP','JNP','JL','JNL','JNG','JG','JMP','JMPE']

op = GetMnem(addr)

for j in ls:

if op.upper() == j:

return 1

global Seg

for seg in Segment()：

if SegName(seg) == '.text':

addr = seg

break

for addr in range(Seg,SegEnd(Seg):

if IsJmp(addr) == 1:

tmp = GetDisasm(addr)

if int(tmp[-6:],16) < addr:

print("循环跳转指令：0x%x" % addr)