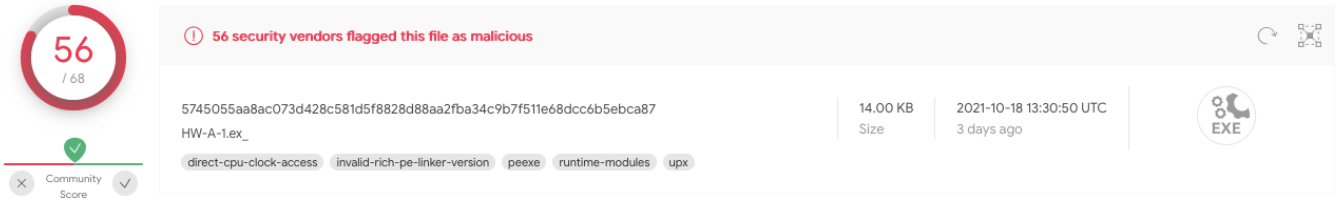
HW3 Combined

Analyzing HW-A-1.ex

HW-A-1.ex

1. Upload the file to https://www.virustotal.com. Does it match any existing antivirus definitions?

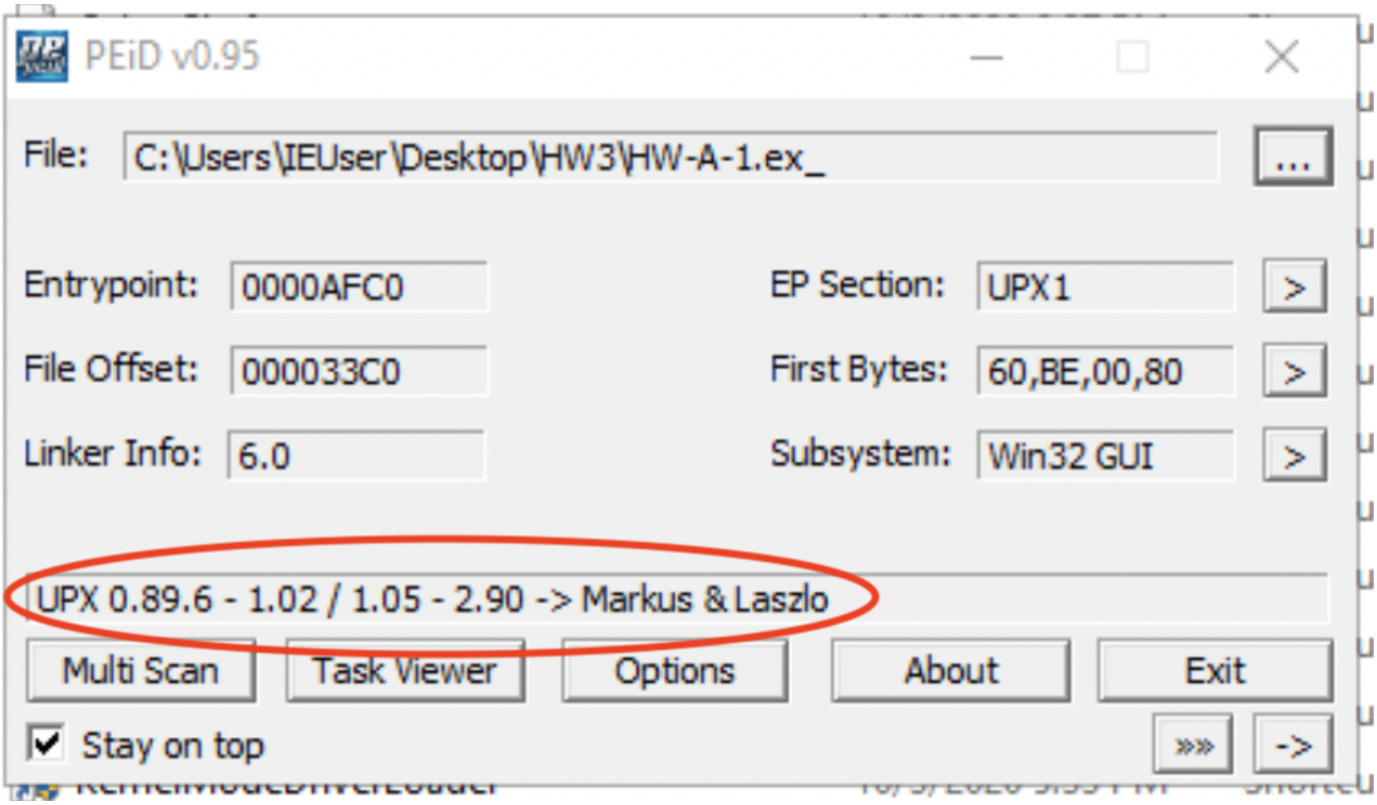
Yes, it matches 58/68 existing antivirus definitions.



DETECTION | DETAILS | RELATIONS | BEHAVIOR | COMMUNITY 1

Acronis (Static ML)	Suspicious	Ad-Aware	Trojan.Generic.15020151
AhnLab-V3	Trojan/Win32.Gen	Alibaba	Trojan:Win32/DocThief.d0928ec7
ALYac	Trojan.Generic.15020151	Antiy-AVL	Trojan/Generic.ASMalwS.14B3D7E
SecureAge APEX	Malicious	Arcabit	Trojan.Generic.DE53077
Avast	FileRepMalware	AVG	FileRepMalware
Avira (no cloud)	TR/Downloader.Gen	Baidu	Win32.Trojan.Agent.aqu
BitDefender	Trojan.Generic.15020151	BitDefenderTheta	Gen:NN.ZexaF.34218.amGfaulDCQk
Bkav Pro	W32.AIDetect.malware2	CAT-QuickHeal	Trojan.Docthief
Comodo	NetWorm.Win32.Koobface.FY@3sikb7	CrowdStrike Falcon	Win/malicious_confidence_100% (W)
Cylance	Unsafe	Cynet	Malicious (score: 100)
eGambit	Unsafe.AI_Score_99%	Elastic	Malicious (high Confidence)
Emsisoft	Trojan.Generic.15020151 (B)	eScan	Trojan.Generic.15020151
ESET-NOD32	Win32/Agent.WON	FireEye	Generic.mg.7722227cae5b9192
Fortinet	W32/Agent.WON!tr	GData	Trojan.Generic.15020151
Ikarus	Trojan.Win32.Agent	Jiangmin	Trojan/DocThief.a
K7AntiVirus	Trojan (0055e3dd1)	K7GW	Trojan (0055e3dd1)
Kaspersky	Trojan.Win32.DocThief.a	Kingsoft	Win32.Troj.DocThief.a.(kcloud)
Lionic	Trojan.Win32.DocThief.4!c	MAX	Malware (ai Score=100)
MaxSecure	Trojan.Malware.300983.susgen	McAfee	RDN/Generic.grp
McAfee-GW-Edition	BehavesLike.Win32.Generic.lc	Microsoft	Trojan:Win32/Occamy.C57
NANO-Antivirus	Trojan.Win32.ThreatHLLSIbased.dewzgf	Palo Alto Networks	Generic.ml
Panda	Trj/Genetic.gen	Sangfor Engine Zero	Suspicious.Win32.Save.a
SentinelOne (Static ML)	Static AI - Malicious PE	Sophos	Mal/Generic-S
Symantec	Trojan.Gen	TACHYON	Trojan/W32.DocThief.32768
Tencent	Win32.Trojan.Docthief.Agba	TrendMicro	Trojan.Win32.DOCTHIEF.A
TrendMicro-HouseCall	Trojan.Win32.DOCTHIEF.A	VBA32	Trojan.DocThief
VIPRE	Trojan.Win32.Generic!BT	Yandex	Trojan.GenAsa!Y8Q6MbldNgc
Zillya	Trojan.DocThief.Win32.2	ZoneAlarm by Check Point	Trojan.Win32.DocThief.a

2. Are there any indications that this file is packed or obfuscated? If so, what are these indicators? If the file is packed, unpack it if possible.



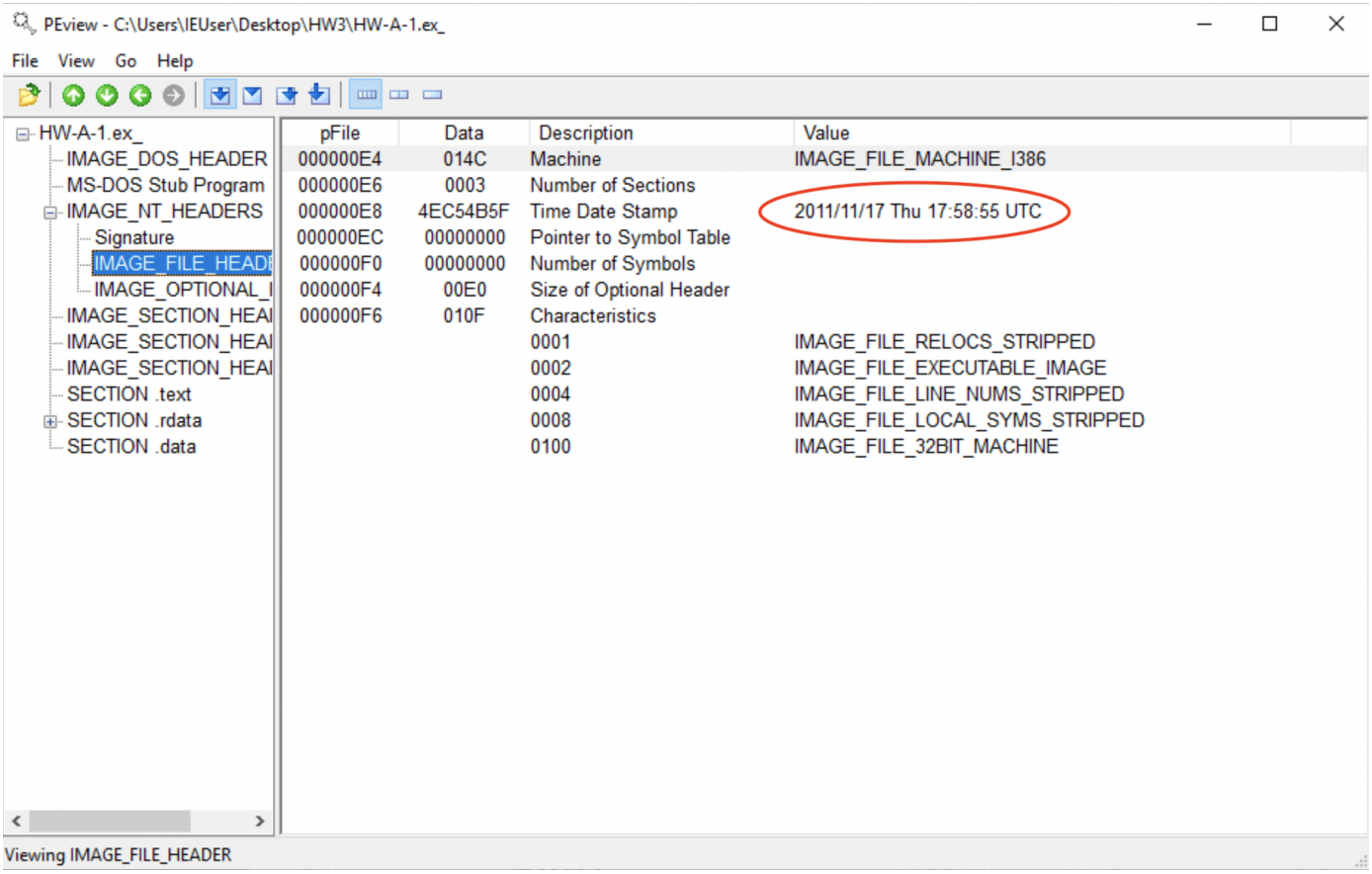
The PEiD has identified the program to be packed with UPX version 0.89 - 1.02 / 1.05-2.90.

To unpack, we use the command: "upx -d HW-A-1.ex_" downloaded from [UPX]

```
C:\Users\IEUser\Desktop\HW3>upx -d HW-A-1.ex_
                       Ultimate Packer for eXecutables
                          Copyright (C) 1996 - 2020
UPX 3.96w       Markus Oberhumer, Laszlo Molnar & John Reiser   Jan 23rd 2020

        File size         Ratio      Format      Name
   --------------------   ------   -----------   -----------
     32768 <-     14336   43.75%    win32/pe     HW-A-1.ex_

Unpacked 1 file.
```

3. When was this program compiled?

It was complied on 2011/11/17 Thu 17:58:55 UTC



4. Do any imports hint at this program's functionality? If so, which imports are they and what do they tell you?

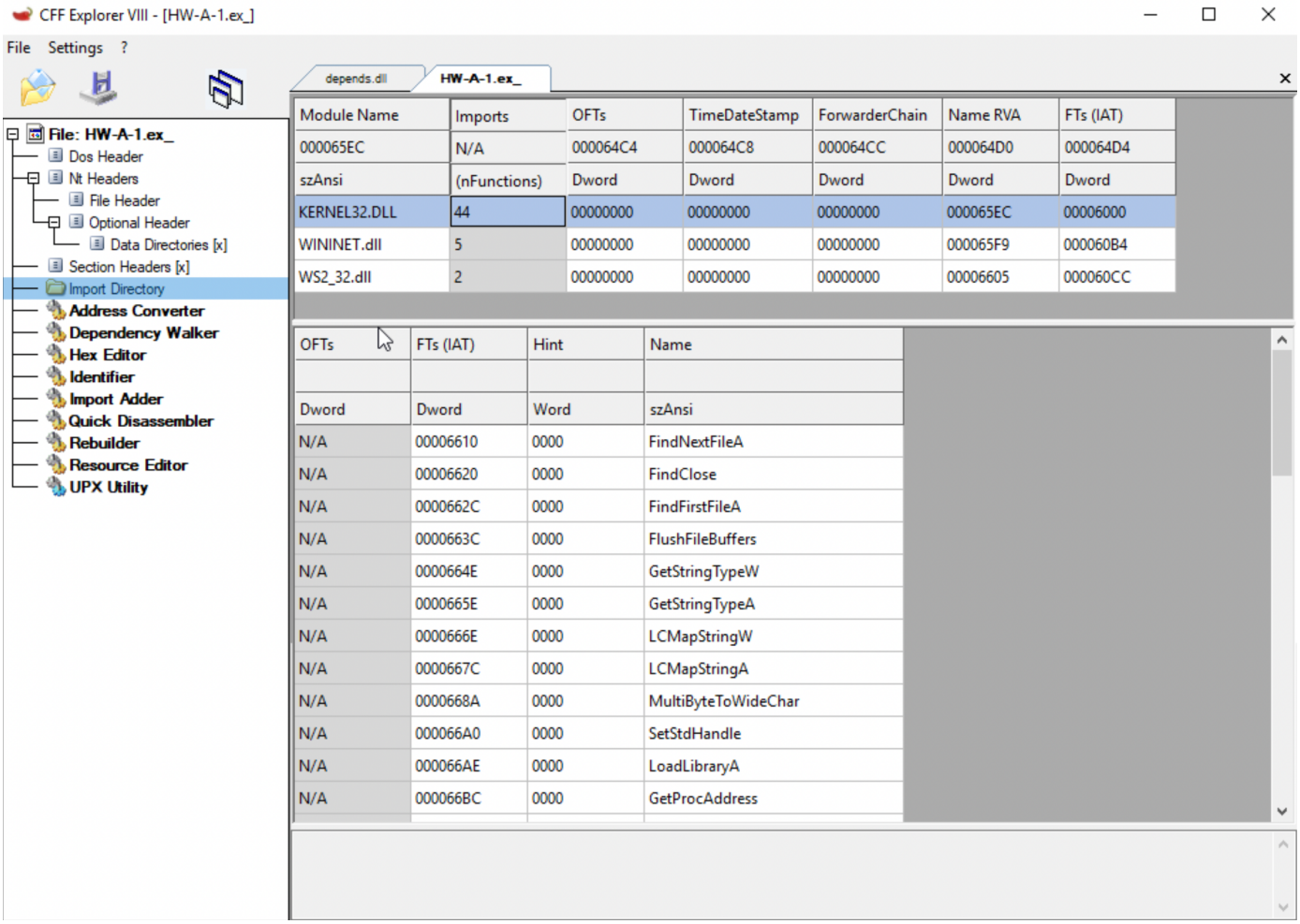
The DLL imported and their cooresponding function:

KERNEL32.DLL: manipulates memory, files, and hardware

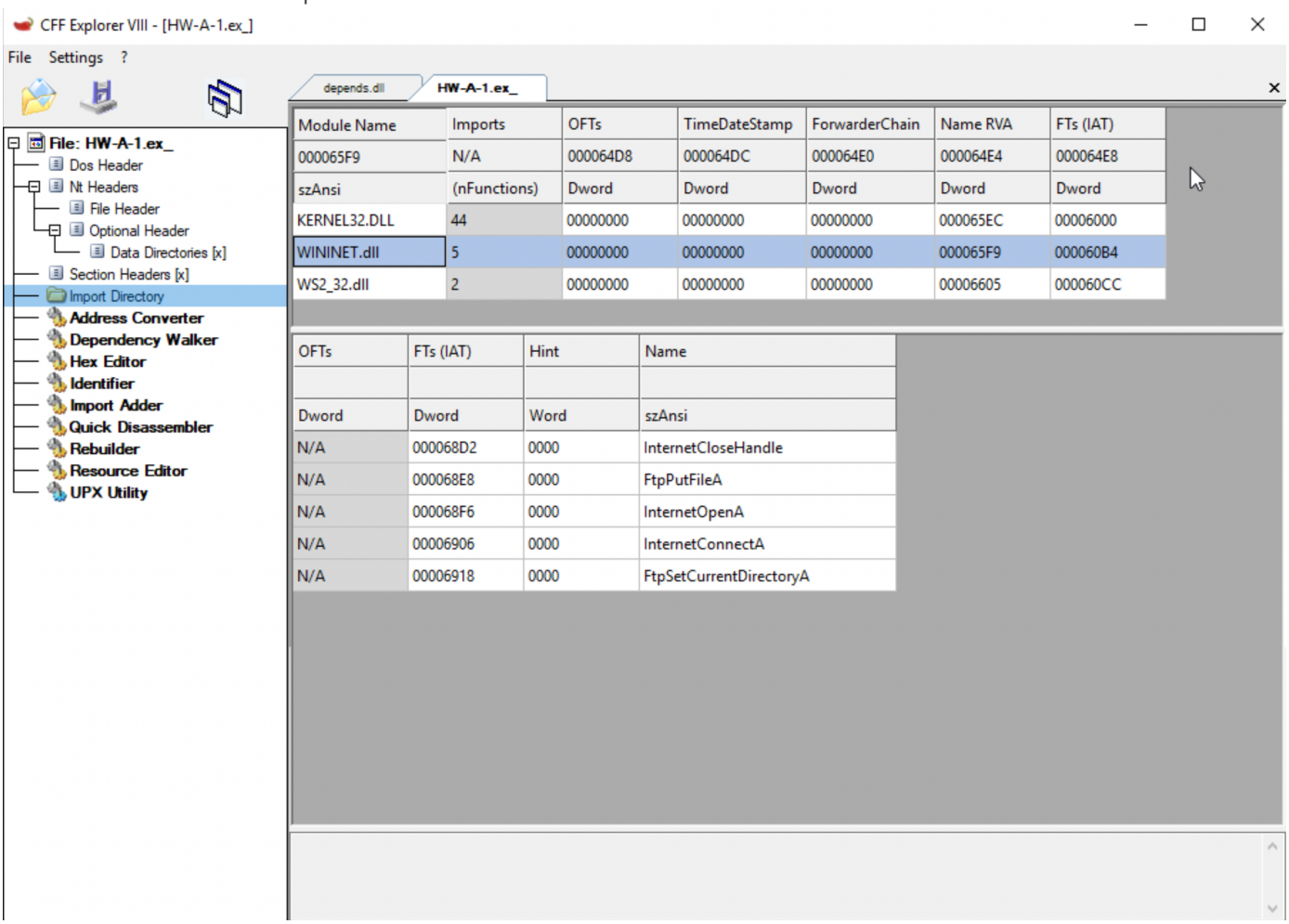
WININET.dll: Has networking function (ex: FTP, HTTP, NTP)

WS2_32.dll: Networking DLL --> the malware has network related task

Putting it together, it seems that the malware manipulates memory, file, or hardware then sends and recieve data from the attacker.



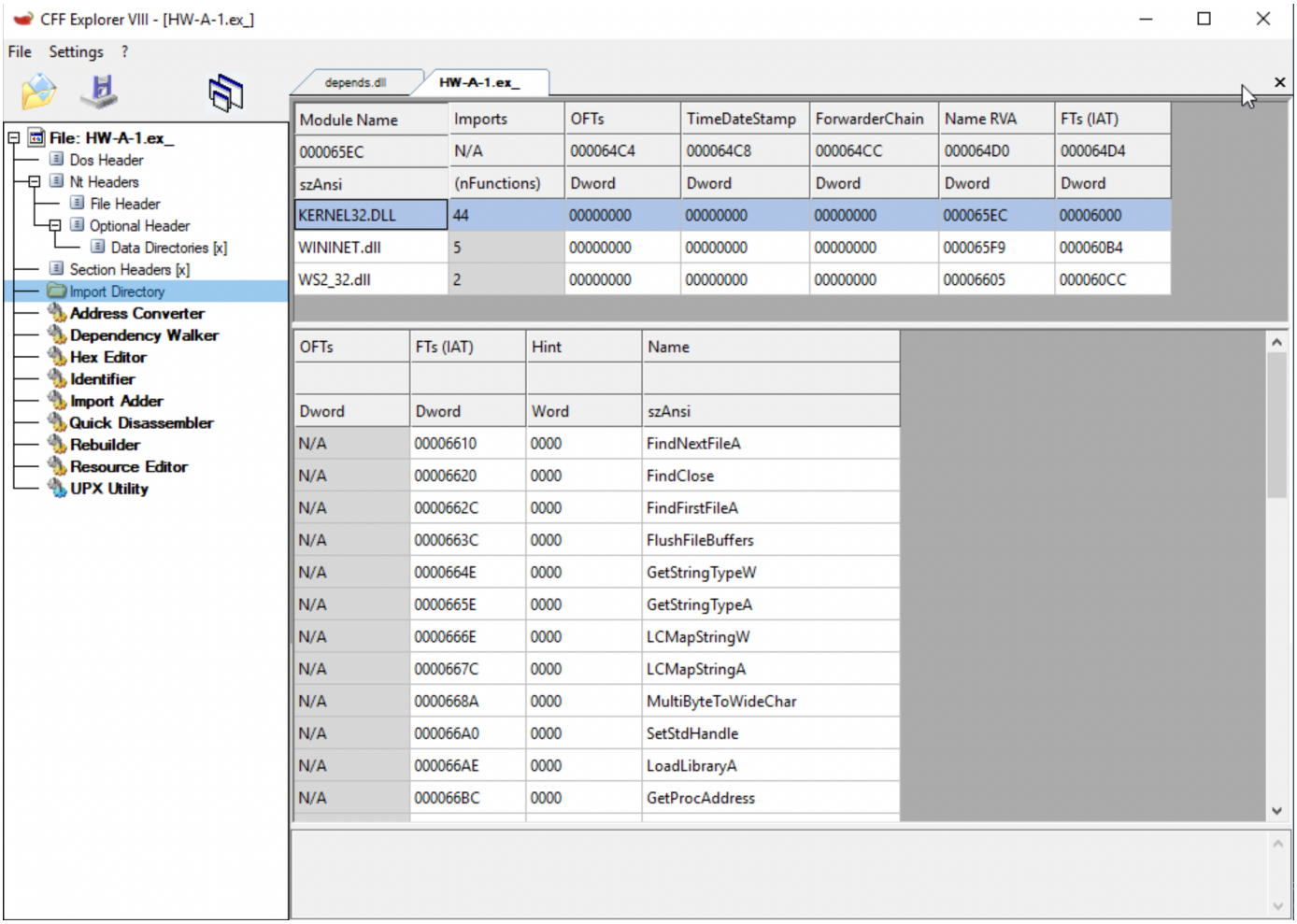
There are 5 functions imported from WININET.dll.

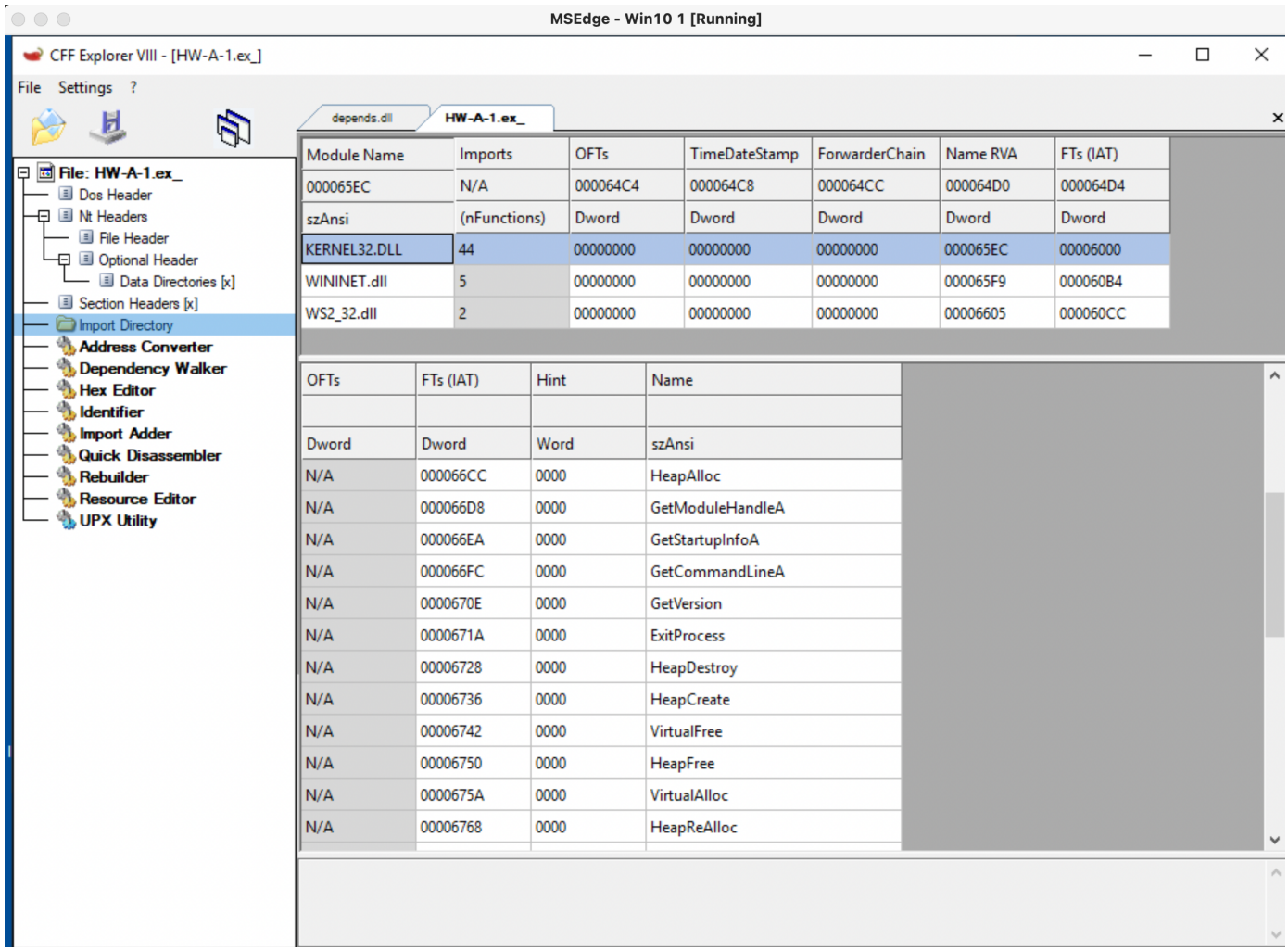


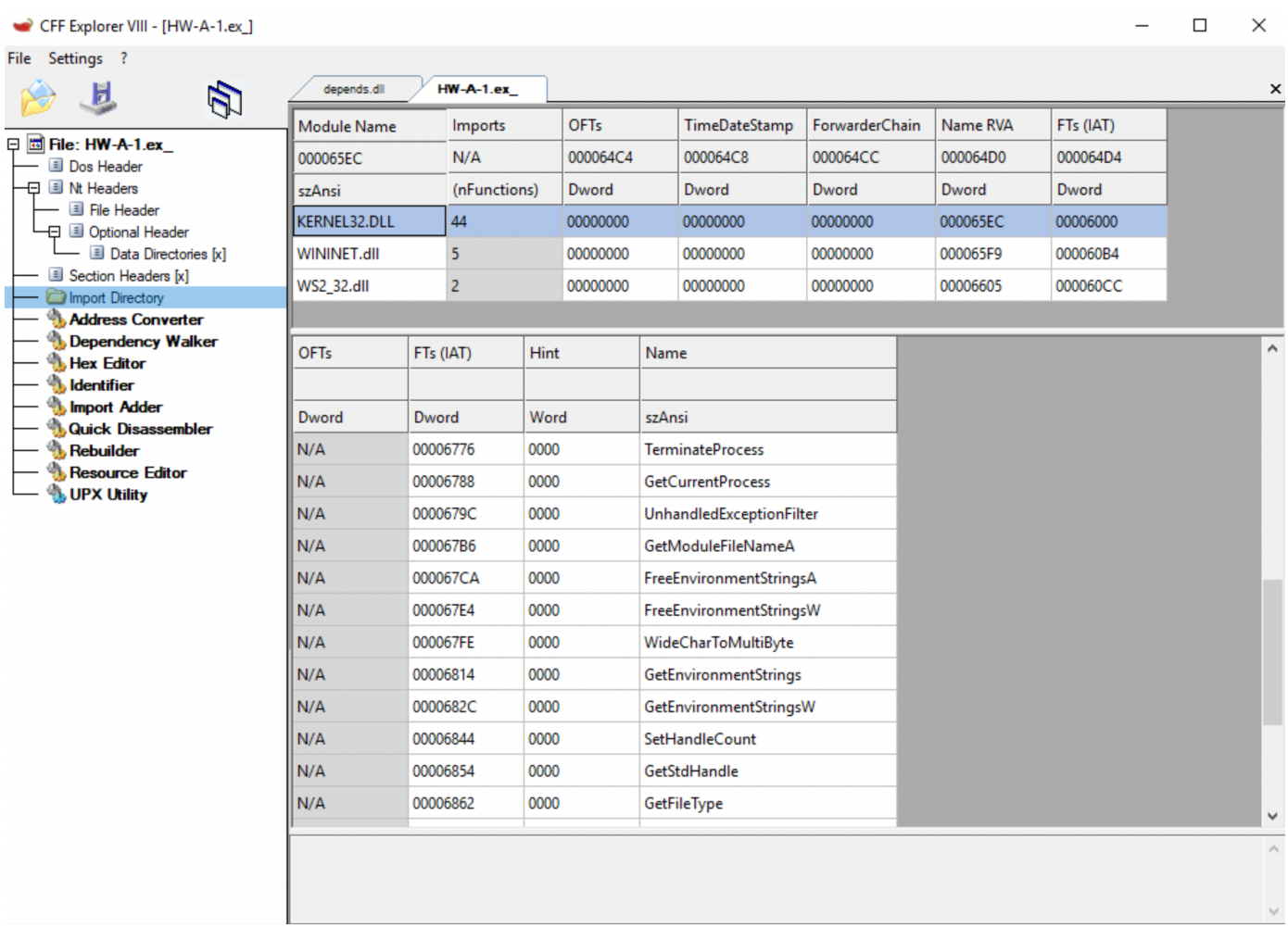
From the functions, it seems that the malware opens a FTP connection, put a malicious file, set the current directory into the malicious file.

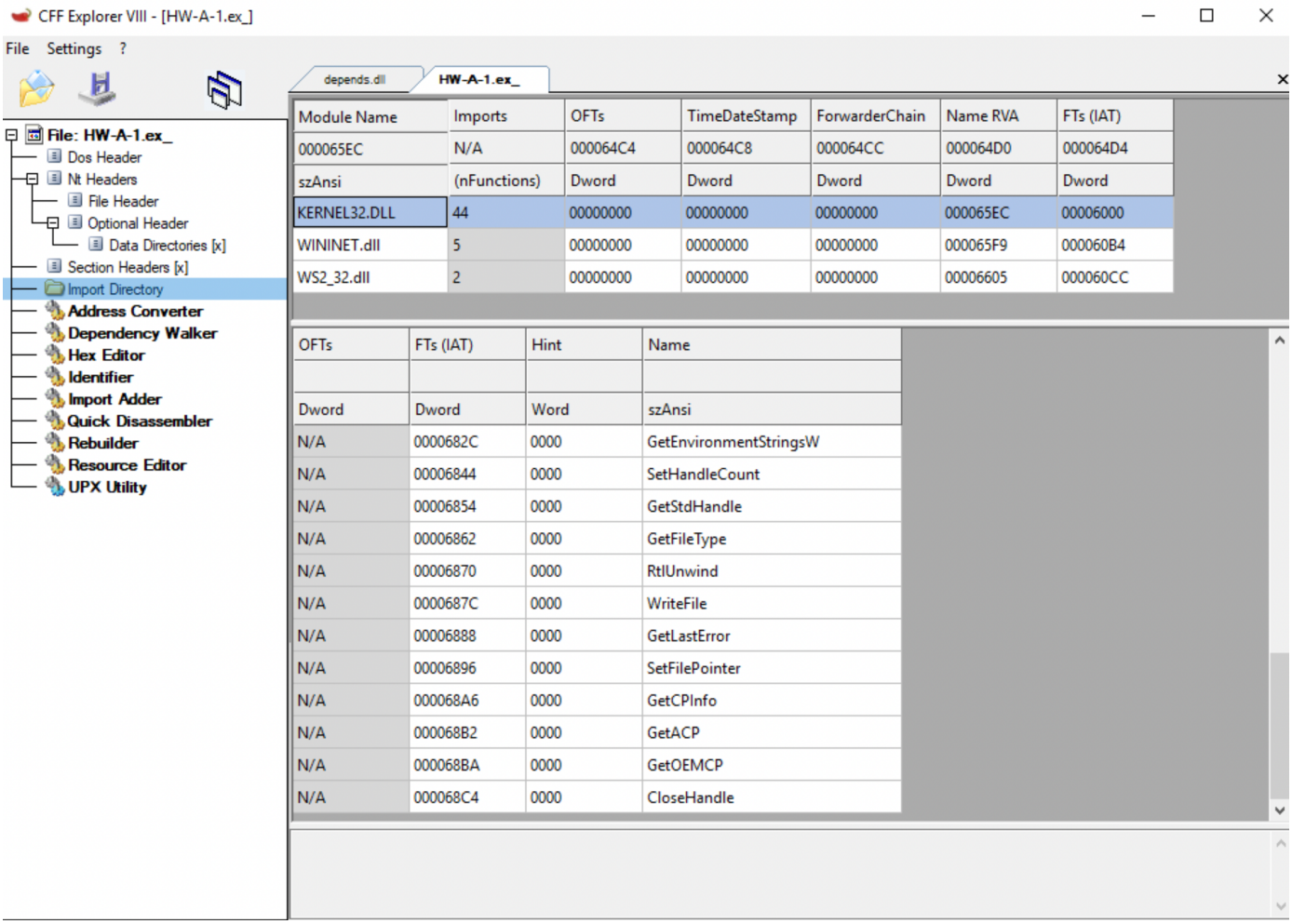
There are 2 functions imported from WS2_32.dll
This DLL helps assist with the creating and maintaing the FTP connection.

There are 44 functions imported from KERNEL32.DLL




MSEdge - Win10 1 [Running]
CFF Explorer VIII - [HW-A-1.ex_]
File Settings ?
depends.dll
HW-A-1.ex_
File: HW-A-1.ex_
Dos Header
Nt Headers
File Header
Optional Header
Data Directories [x]
Section Headers [x]
Import Directory
Address Converter
Dependency Walker
Hex Editor
Identifier
Import Adder
Quick Disassembler
Rebuilder
Resource Editor
UPX Utility
Module Name | Imports | OFTs | TimeDateStamp | ForwarderChain | Name RVA | FTs (IAT)
000065EC | N/A | 000064C4 | 000064C8 | 000064CC | 000064D0 | 000064D4
szAnsi | (nFunctions) | Dword | Dword | Dword | Dword | Dword
KERNEL32.DLL | 44 | 00000000 | 00000000 | 00000000 | 000065EC | 00006000
WININET.dll | 5 | 00000000 | 00000000 | 00000000 | 000065F9 | 000060B4
WS2_32.dll | 2 | 00000000 | 00000000 | 00000000 | 00006605 | 000060CC
OFTs | FTs (IAT) | Hint | Name
Dword | Dword | Word | szAnsi
N/A | 000066CC | 0000 | HeapAlloc
N/A | 000066D8 | 0000 | GetModuleHandleA
N/A | 000066EA | 0000 | GetStartupInfoA
N/A | 000066FC | 0000 | GetCommandLineA
N/A | 0000670E | 0000 | GetVersion
N/A | 0000671A | 0000 | ExitProcess
N/A | 00006728 | 0000 | HeapDestroy
N/A | 00006736 | 0000 | HeapCreate
N/A | 00006742 | 0000 | VirtualFree
N/A | 00006750 | 0000 | HeapFree
N/A | 0000675A | 0000 | VirtualAlloc
N/A | 00006768 | 0000 | HeapReAlloc


CFF Explorer VIII - [HW-A-1.ex_]
File Settings ?
depends.dll
HW-A-1.ex_
File: HW-A-1.ex_
Dos Header
Nt Headers
File Header
Optional Header
Data Directories [x]
Section Headers [x]
Import Directory
Address Converter
Dependency Walker
Hex Editor
Identifier
Import Adder
Quick Disassembler
Rebuilder
Resource Editor
UPX Utility
Module Name | Imports | OFTs | TimeDateStamp | ForwarderChain | Name RVA | FTs (IAT)
000065EC | N/A | 000064C4 | 000064C8 | 000064CC | 000064D0 | 000064D4
szAnsi | (nFunctions) | Dword | Dword | Dword | Dword | Dword
KERNEL32.DLL | 44 | 00000000 | 00000000 | 00000000 | 000065EC | 00006000
WININET.dll | 5 | 00000000 | 00000000 | 00000000 | 000065F9 | 000060B4
WS2_32.dll | 2 | 00000000 | 00000000 | 00000000 | 00006605 | 000060CC
OFTs | FTs (IAT) | Hint | Name
Dword | Dword | Word | szAnsi
N/A | 00006776 | 0000 | TerminateProcess
N/A | 00006788 | 0000 | GetCurrentProcess
N/A | 0000679C | 0000 | UnhandledExceptionFilter
N/A | 000067B6 | 0000 | GetModuleFileNameA
N/A | 000067CA | 0000 | FreeEnvironmentStringsA
N/A | 000067E4 | 0000 | FreeEnvironmentStringsW
N/A | 000067FE | 0000 | WideCharToMultiByte
N/A | 00006814 | 0000 | GetEnvironmentStrings
N/A | 0000682C | 0000 | GetEnvironmentStringsW
N/A | 00006844 | 0000 | SetHandleCount
N/A | 00006854 | 0000 | GetStdHandle
N/A | 00006862 | 0000 | GetFileType

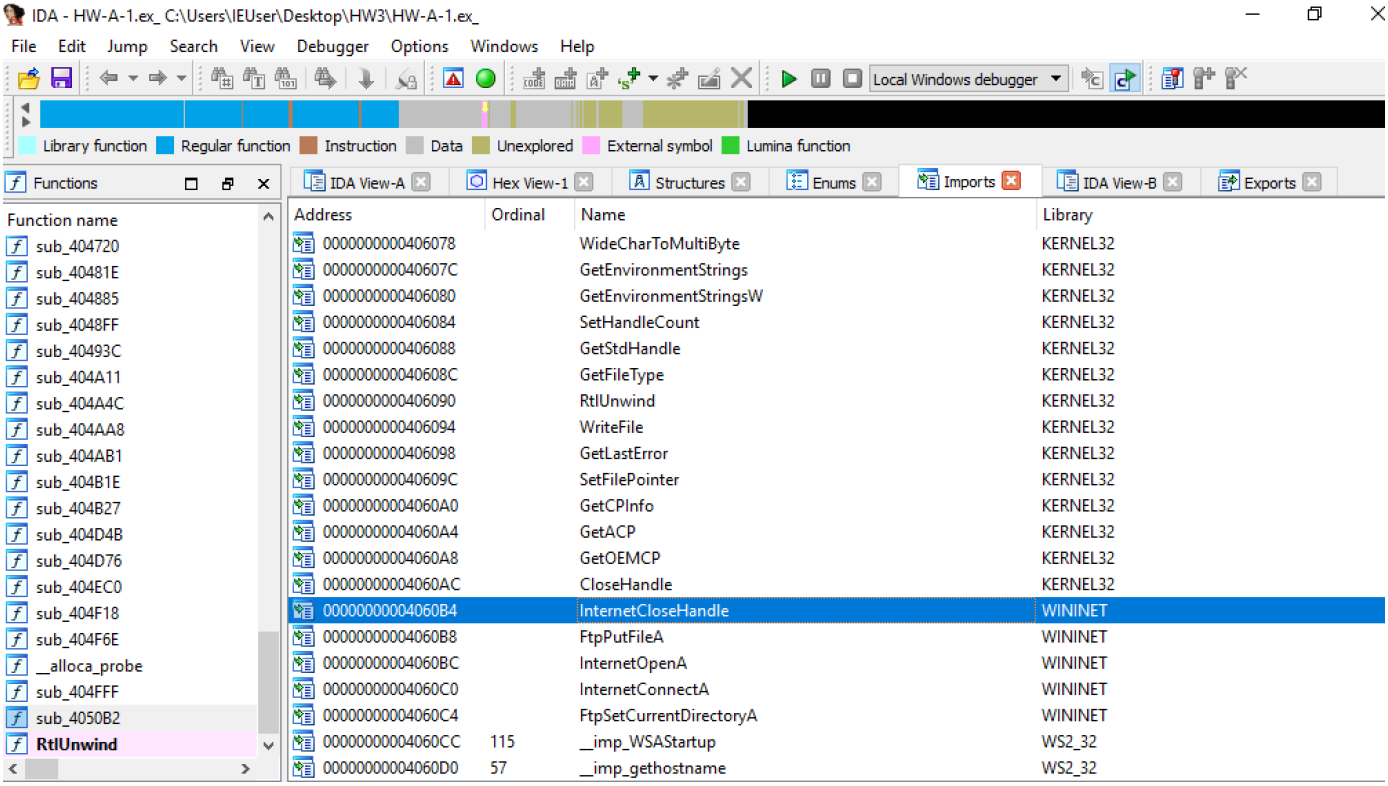


In the KERNEL32.DLL, we see the functions used to open and manipulate files including the FindFirstFile and FindNextFile. It indicates that the maleware searches through the filesystem, and it can open and modify files.

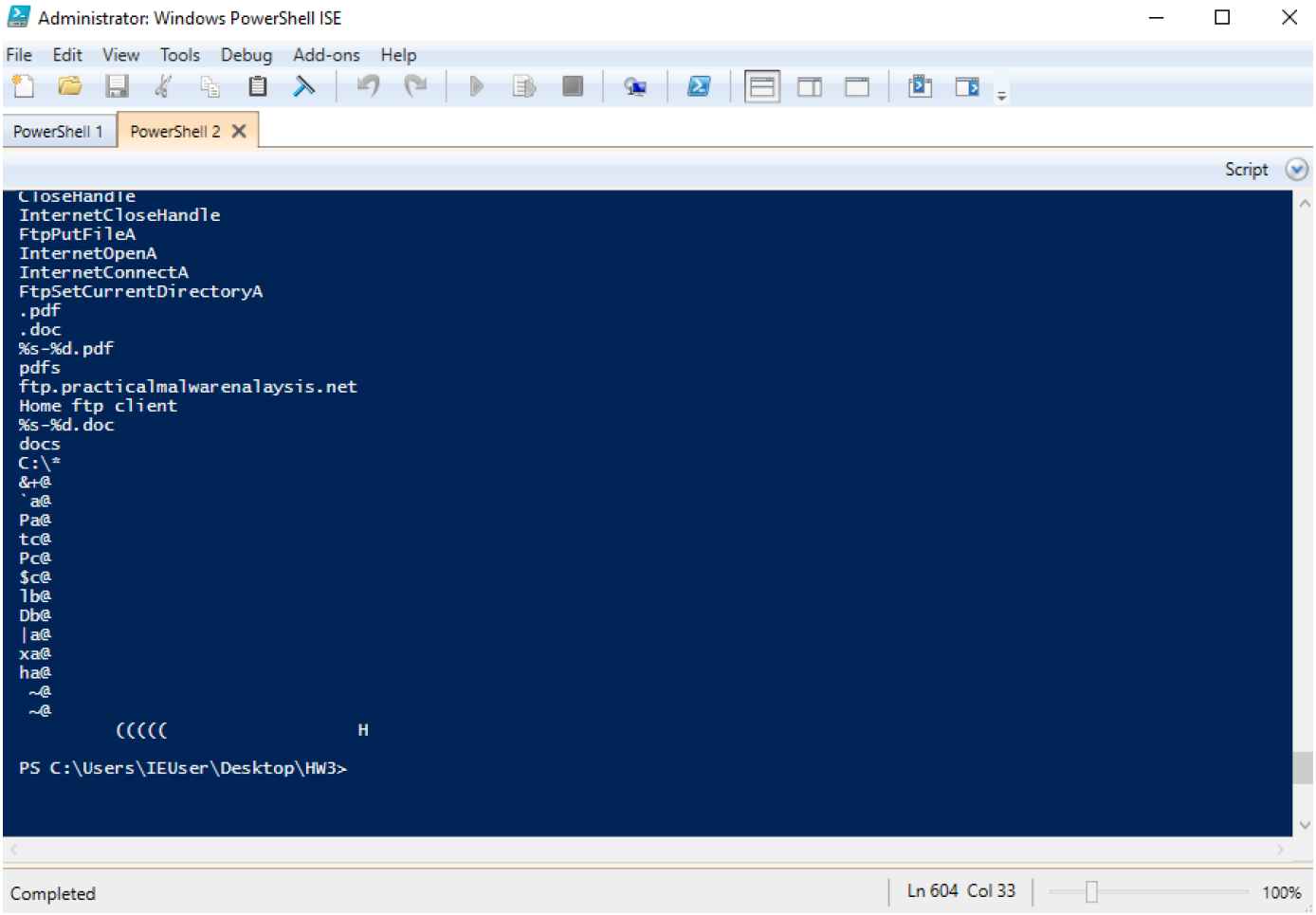
5. What host or network-based indicators could be used to identify this malware on infected machines?

It has WINNET.dll therefore it has networking functions.

We can check if there is an FTP connection on the infected machine.



From the command "strings HW-A-1.ex_" we can see that it calls for the string "ftp.practicalmalwareanalysis.net"

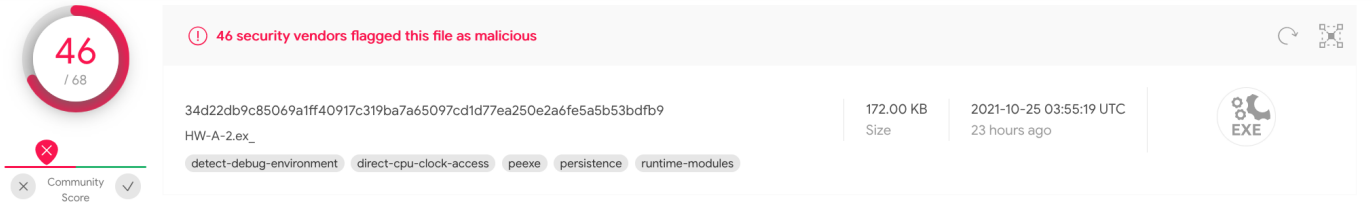


Analyzing HW-A-2.ex

Task 2: Answer the following questions after analyzing HW-A-2.exe

1. Upload the file to https://www.virustotal.com. Does it match any existing antivirus definitions?

Yes. It mataches 46/68 existing antivirus definition.

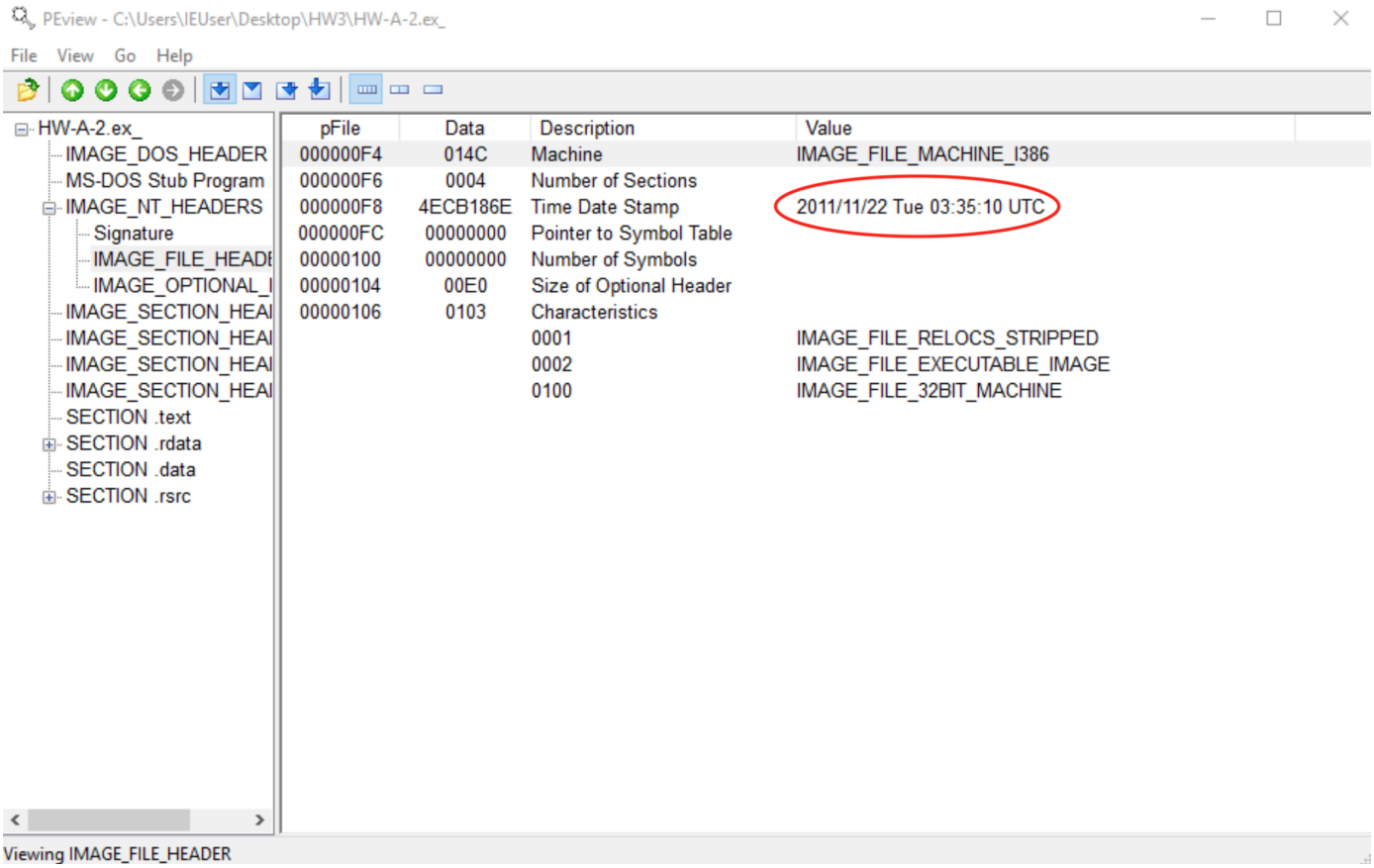


DETECTION | DETAILS | RELATIONS | BEHAVIOR | COMMUNITY 1

Ad-Aware	Trojan.GenericKD.2728766	AhnLab-V3	Trojan/Win32.Inject.C1318505
Alibaba	Trojan:Win32/Invader.39233a27	ALYac	Trojan.GenericKD.2728766
Antiy-AVL	Trojan/Generic.ASMalwS.B7022F	SecureAge APEX	Malicious
Arcabit	Trojan.Generic.D29A33E	Avast	Win32:Malware-gen
AVG	Win32:Malware-gen	Avira (no cloud)	TR/Dropper.Gen
BitDefender	Trojan.GenericKD.2728766	BitDefenderTheta	AI:Packer.DE75EAB320
Comodo	Malware@#ckh8u07jk971	CrowdStrike Falcon	Win/malicious_confidence_100% (W)
Cynet	Malicious (score: 100)	eGambit	Unsafe.AI_Score_97%
Elastic	Malicious (high Confidence)	Emsisoft	Trojan.GenericKD.2728766 (B)
eScan	Trojan.GenericKD.2728766	ESET-NOD32	A Variant Of Win32/Agent.WOK
FireEye	Generic.mg.d93adc942c53bdb9	Fortinet	PossibleThreat.SB!tr
GData	Trojan.GenericKD.2728766	Ikarus	Win32.SuspectCrc
K7AntiVirus	Trojan (0000000c1)	K7GW	Trojan (0000000c1)
Kingsoft	Win32.Heur.KVM003.a.(kcloud)	Lionic	Trojan.Win32.Invader.4!c
MAX	Malware (ai Score=100)	MaxSecure	Trojan.Malware.300983.susgen
McAfee	GenericR-EMO!D93ADC942C53	NANO-Antivirus	Trojan.Win32.Invader.dlbvjy
Palo Alto Networks	Generic.ml	Panda	Trj/CI.A
Sangfor Engine Zero	Trojan.Win32.Save.a	SentinelOne (Static ML)	Static AI - Malicious PE
Sophos	Generic ML PUA (PUA)	Symantec	ML.Attribute.HighConfidence
Tencent	Malware.Win32.Gencirc.114bec55	TrendMicro	TROJ_GEN.R002C0PJI21
TrendMicro-HouseCall	TROJ_GEN.R002C0PJI21	VBA32	Trojan.Invader
VIPRE	Trojan.Win32.Generic!BT	Yandex	Trojan.GenAsa!Bd6JlhQ2Vvo

2. When was this program compiled?

It was complied on 2011/11/22 Tue 03:35:10 UTC

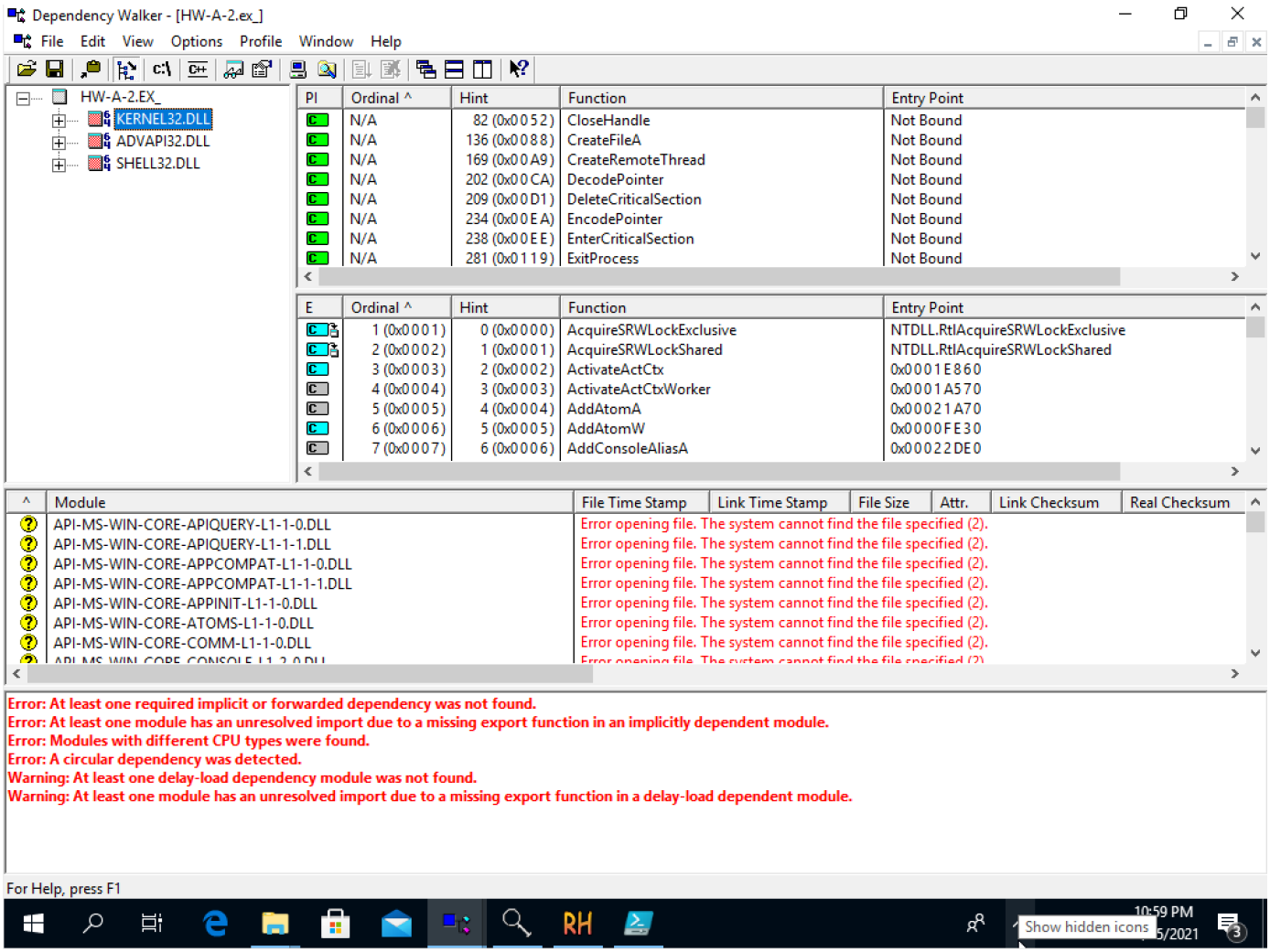


3. Do any imports hint at this program's functionality? If so, which imports are they and what do they tell you?

HW-A-2.ex_

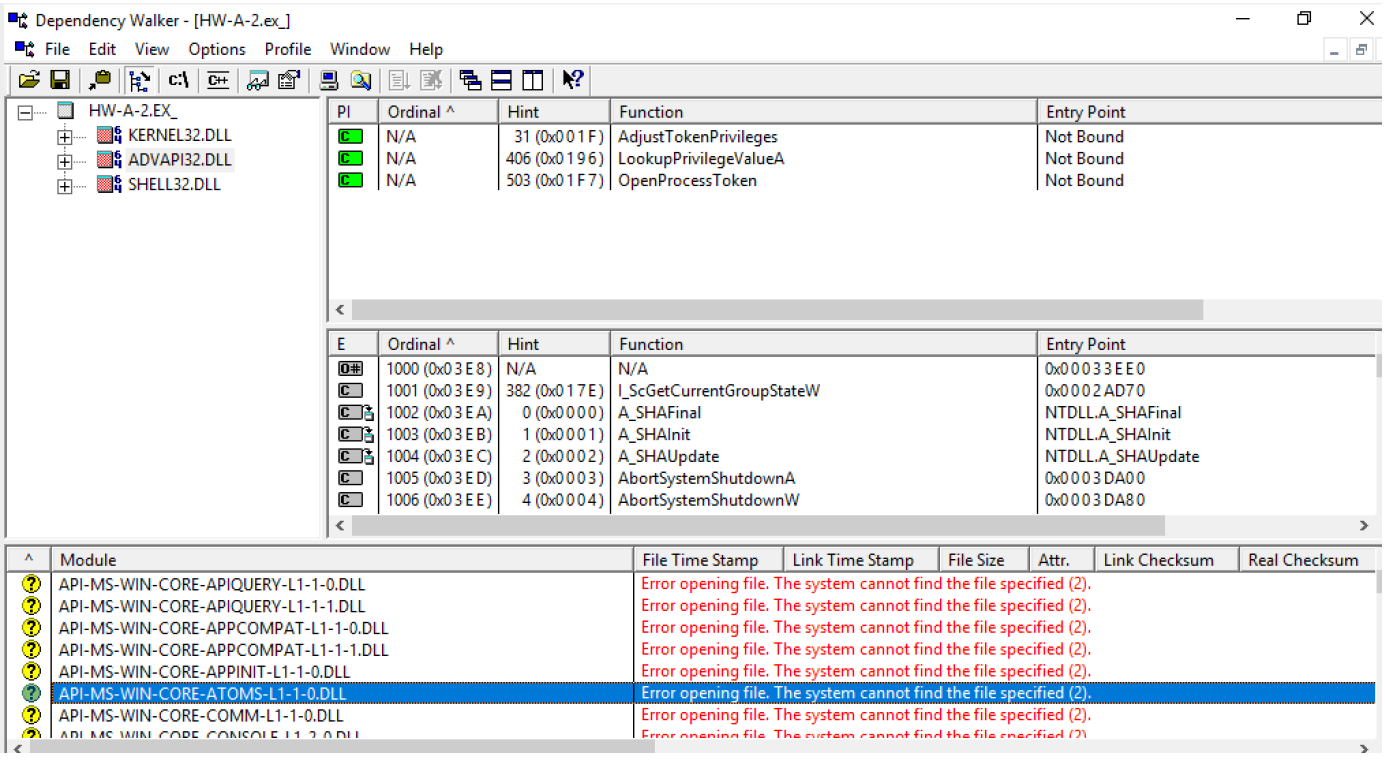
Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.dll	72	00007C3C	00000000	00000000	00007EB6	00006010
ADVAPI32.dll	3	00007C2C	00000000	00000000	00007F08	00006000
SHELL32.dll	1	00007D60	00000000	00000000	00007F26	00006134

The KERNEL32.dll import tells me that the malware manipulates memory, files, and hardware.

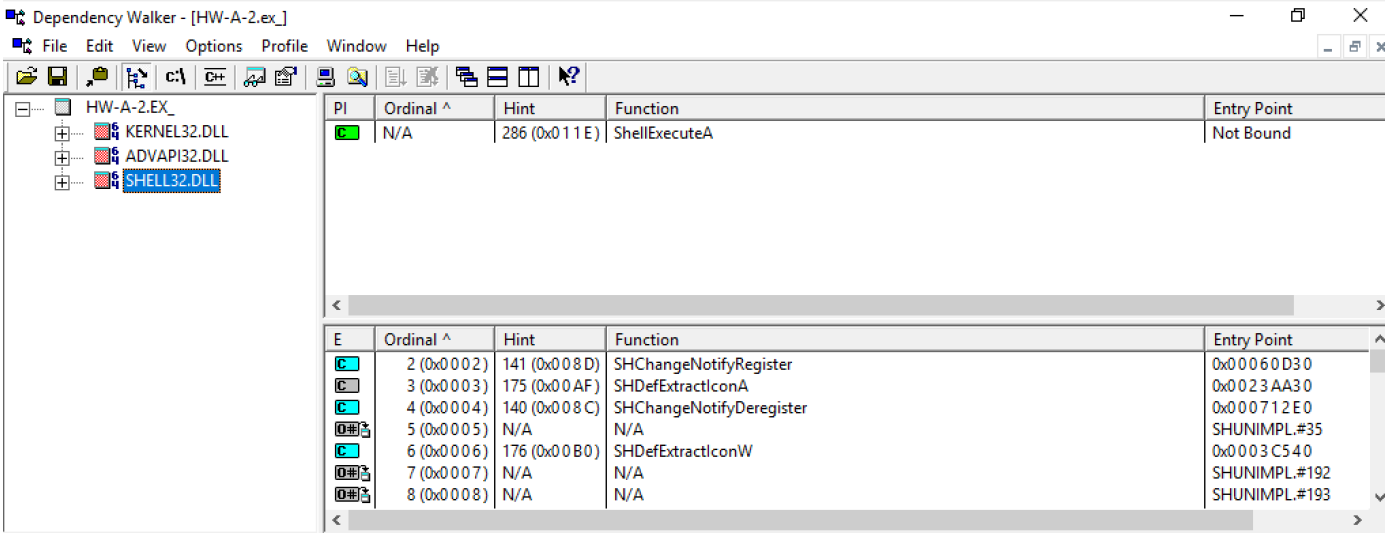


The import functions tells us that the malware loads data from resource section (LoadResoruce, FindResource) and writes a file to disk (CreateFile, WriteFile)

The ADVAPI32.dll import tells me that the malware is trying to access advanced core Windows components such as the Service Manager and Registry. We can assumes that it tries to access protected files by adjusting to admin special permissions.

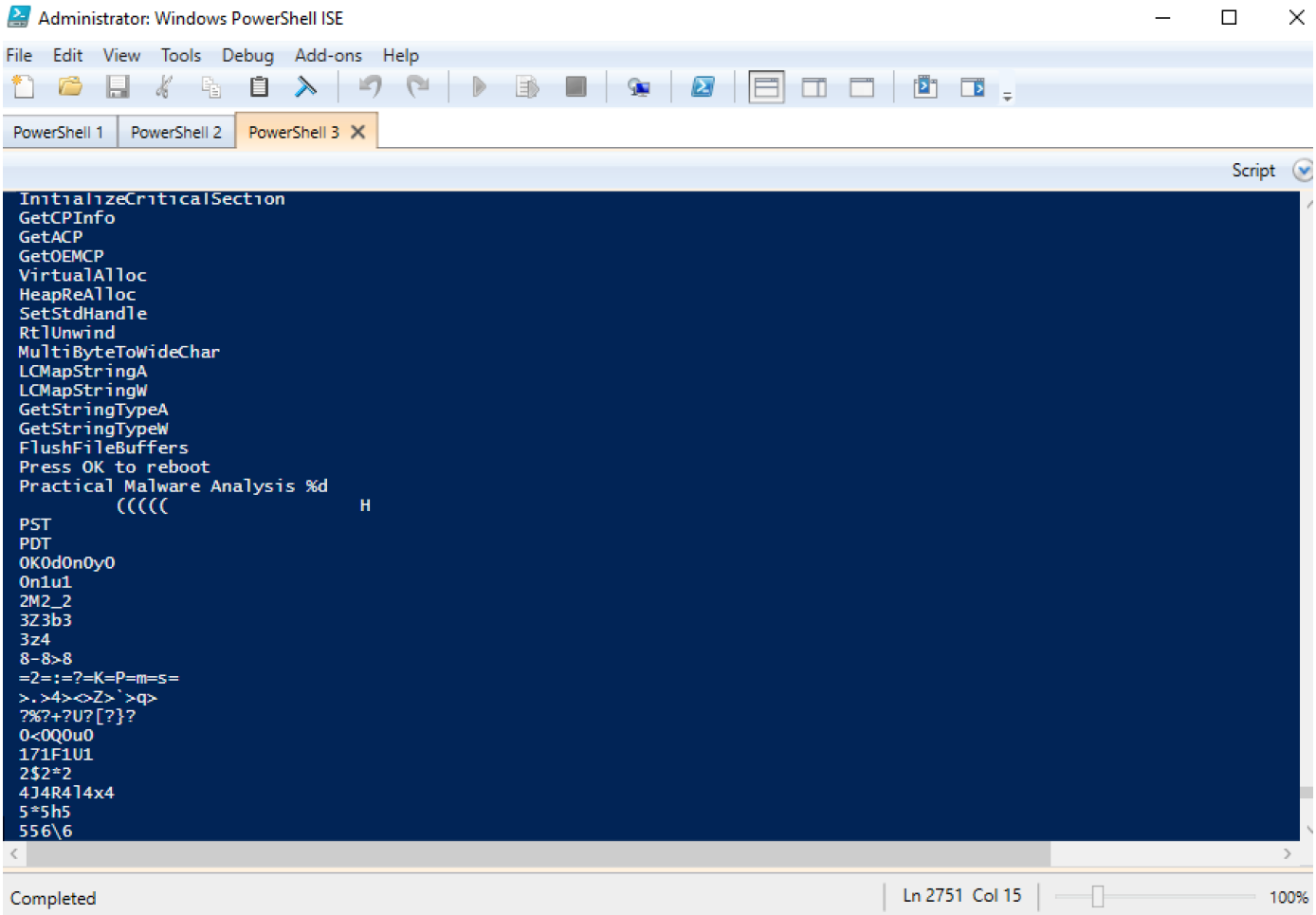


The SHEL32.dll import tells me that the malware is trying to open a powershell which can be used to open webpages and files. It executes teh program using the function ShellExecuteA.



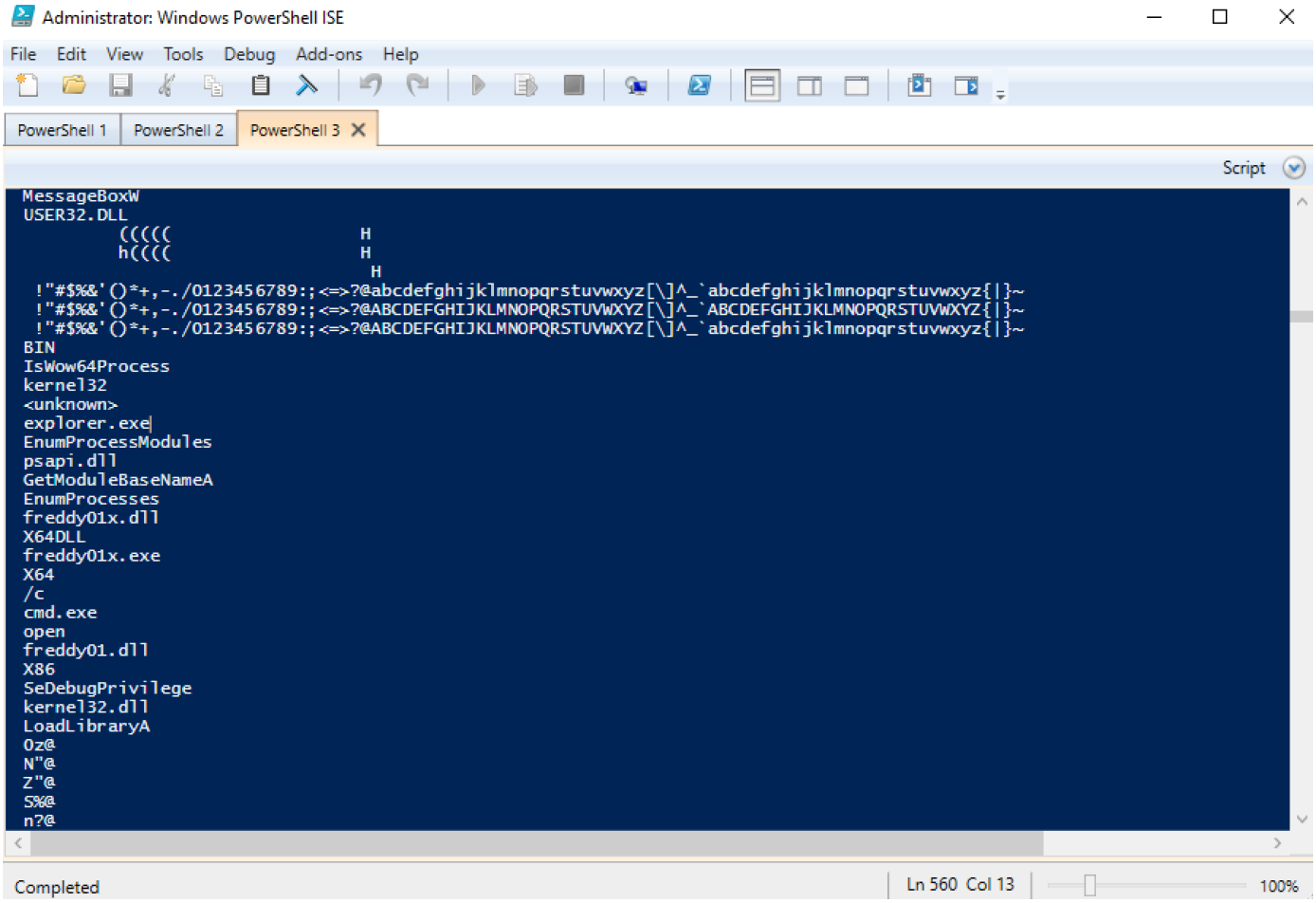
4. What host- or network-based indicators could be used to identify this malware on infected machines?

The malware attempts to reboot the computer with the attacker having admin privilege.



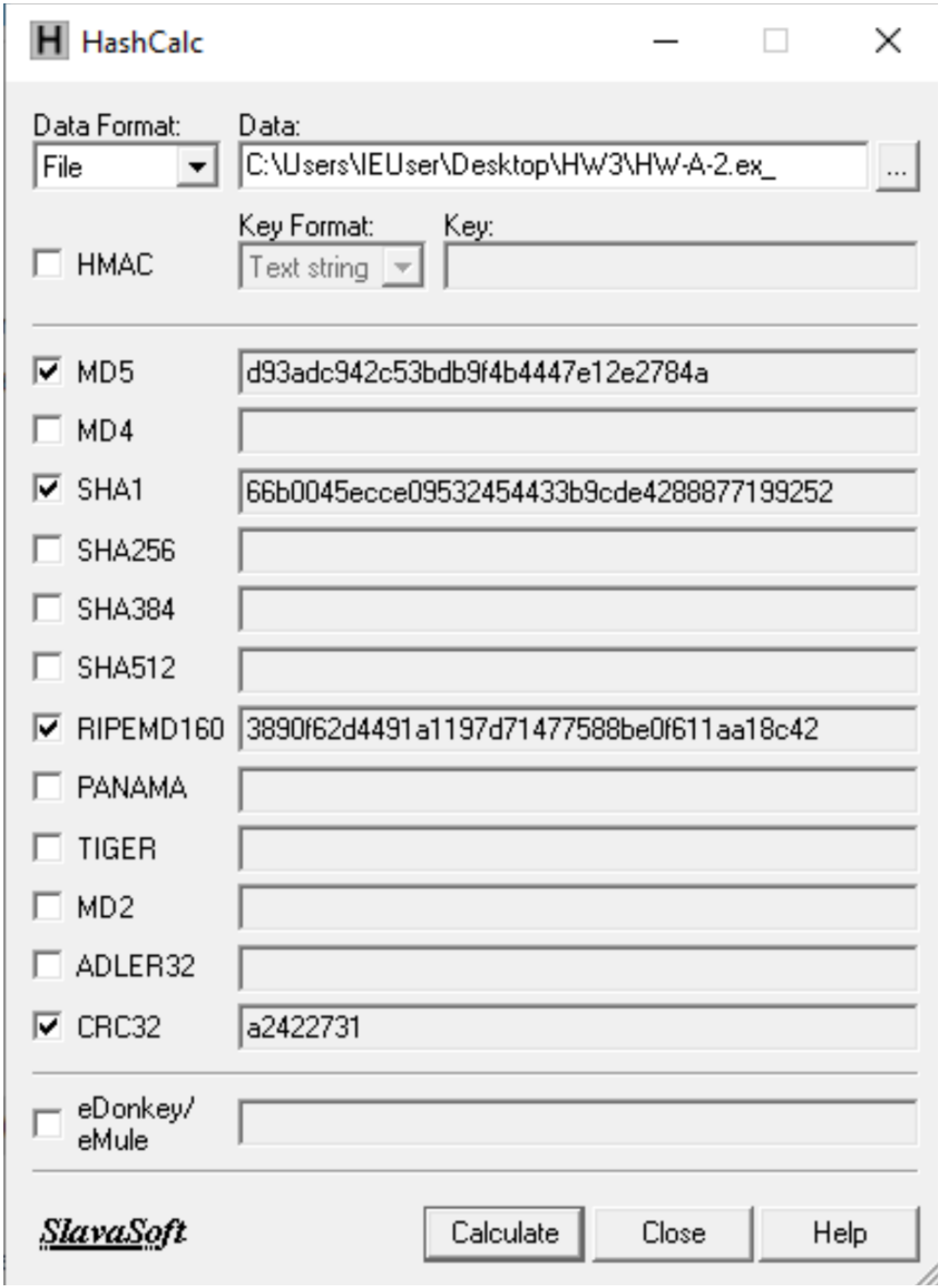
In order to do this the malware runs dll modules that is personally writen including freedy01x.dll and

attempts to execute it as seen with freedy01x.exe.



We can check if freedy01.dll is running on the infected machine.

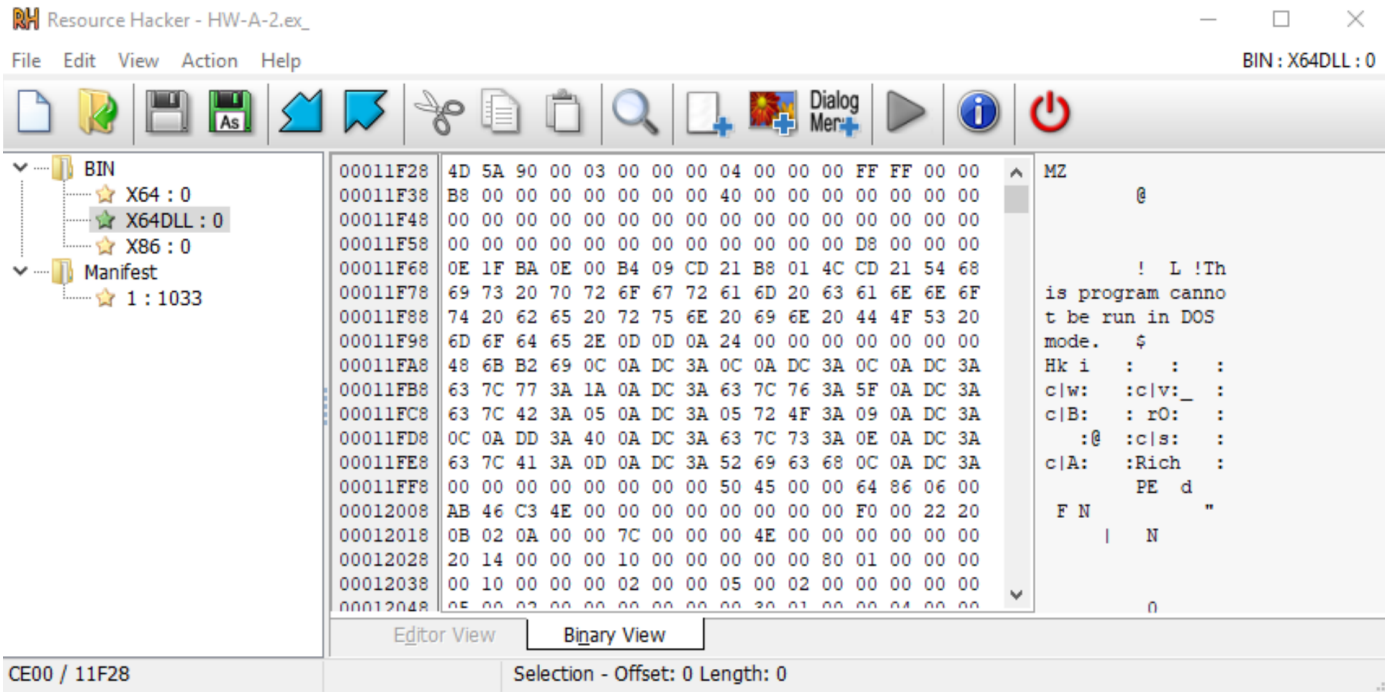
5. This file has multiple resources in its resource section. What are their respective MD5 or SHA hashes? (Resources should be in BIN format)


HashCalc
Data Format:
File
Data:
C:\Users\IEUser\Desktop\HW3\HW-A-2.ex_
...
HMAC
Key Format:
Text string
Key:
MD5
d93adc942c53bdb9f4b4447e12e2784a
MD4
SHA1
66b0045ecce09532454433b9cde4288877199252
SHA256
SHA384
SHA512
RIPEMD160
3890f62d4491a1197d71477588be0f611aa18c42
PANAMA
TIGER
MD2
ADLER32
CRC32
a2422731
eDonkey/
eMule
SlavaSoft
Calculate
Close
Help

```
PS C:\Users\IEUser\Desktop\HW3> Get-FileHash -Algorithm MD5 .\HW-A-2.ex_

Algorithm       Hash
---------       ----
MD5             D93ADC942C53BDB9F4B4447E12E2784A
```

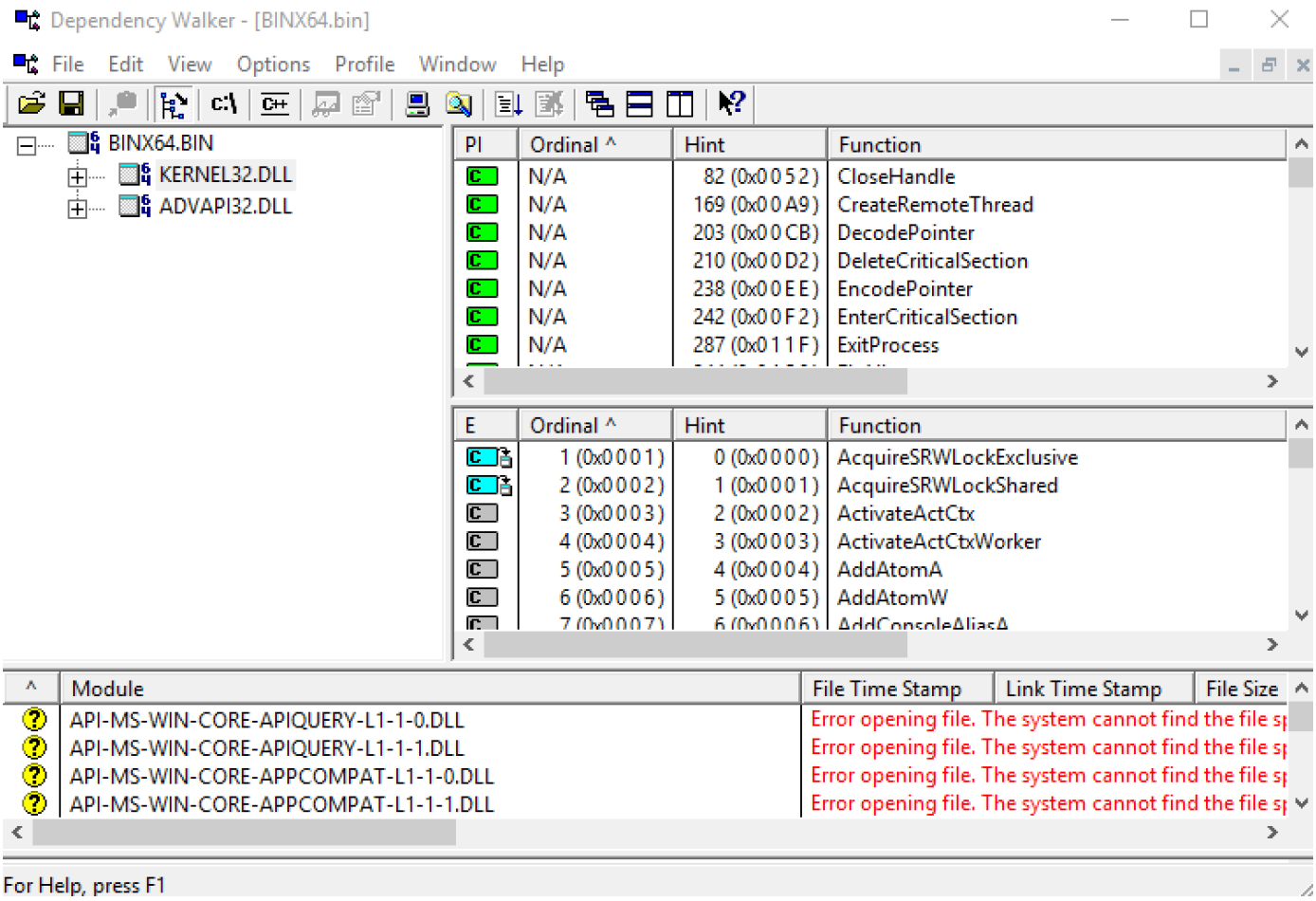
```
PS C:\Users\IEUser\Desktop\HW3> Get-FileHash -Algorithm SHA1 .\HW-A-2.ex_

Algorithm       Hash
---------       ----
SHA1            66B0045ECCE09532454433B9CDE4288877199252
```

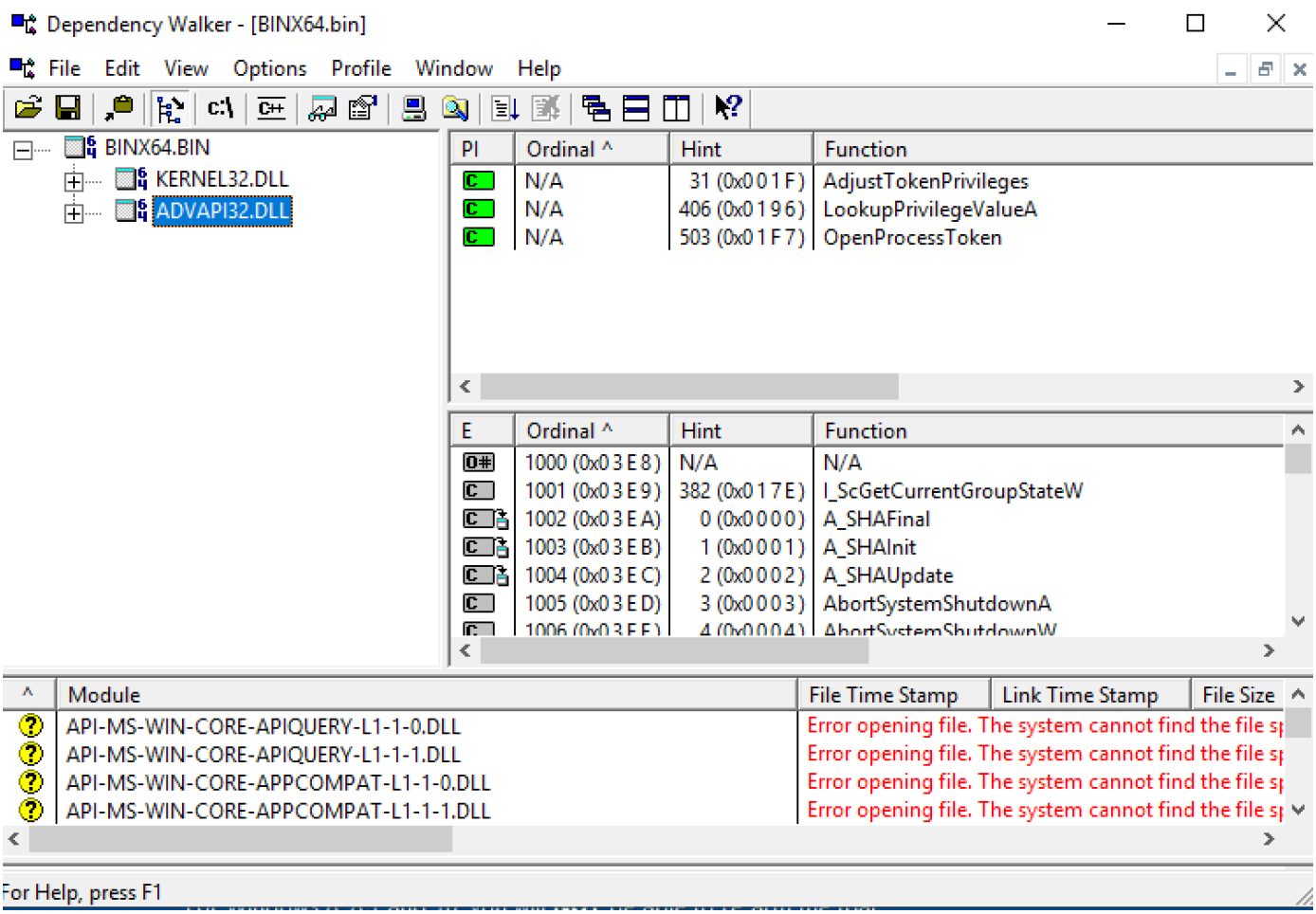
6. What are the differences between the resources?



BIN has 3 resources sectuib x64, x64DLL, and x86.

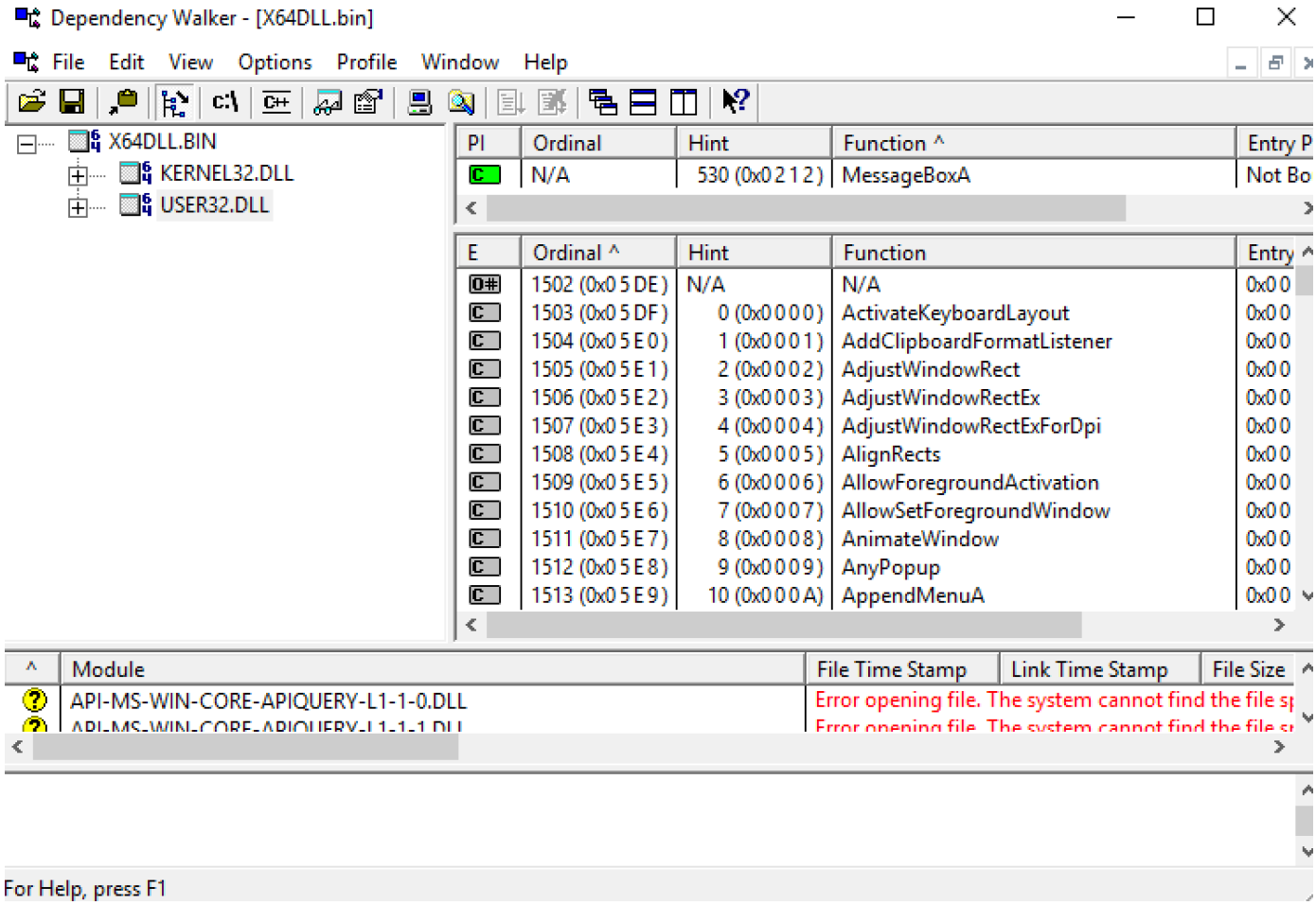
x64 imports KERNEL32.DLL and ADVAPI32.DLL.



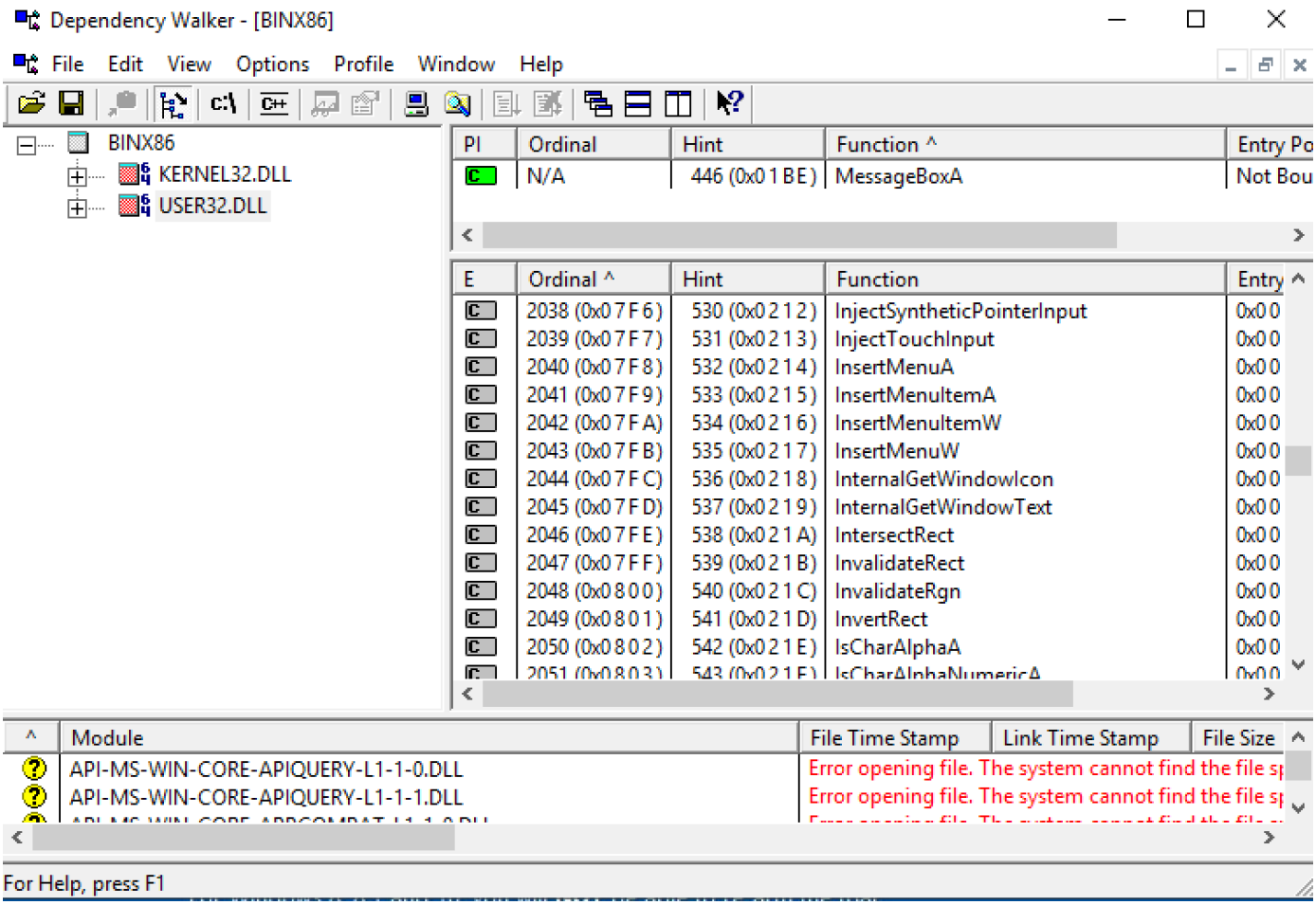


The most important things x64 does is that it adjusts the privillege of the malware to have admin access.

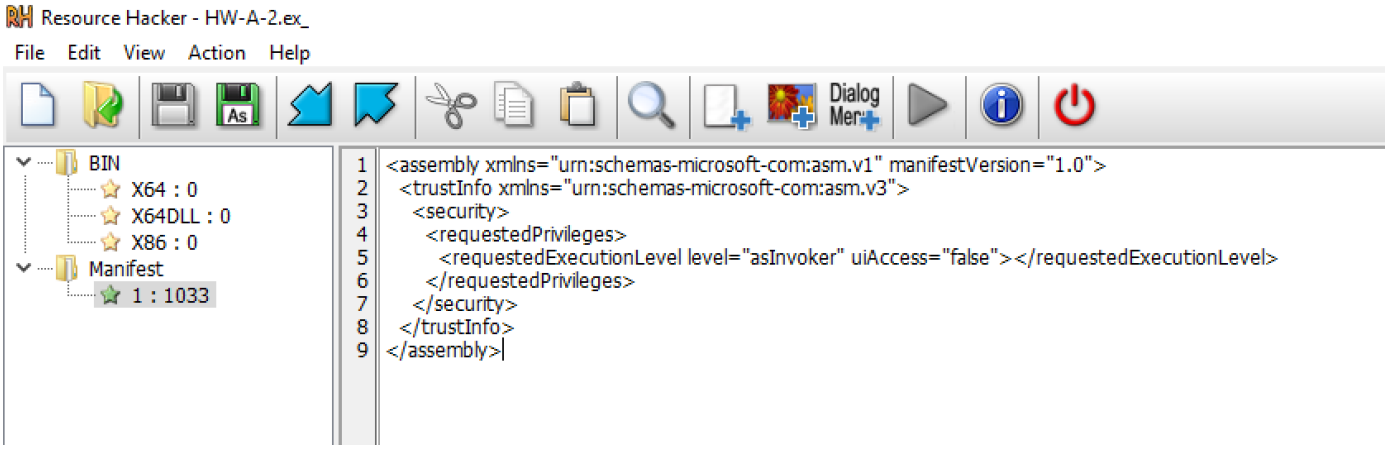
x64DLL imports two DLLs, KERNEL32.DLL and USER32.DLL. The USER32.DLL has functions such as GetKeyboardLayout, GetMouseMovePointsEx, indicating that it is probably a passive keylogger. Additionally, the MessageBoxA Displays a modal dialog box that contains a system icon, a set of buttons, and a brief application-specific message, such as status or error information.It stores the data from keylogging in a file indicated by the functions in KERNEL32.DLL containtaing functions such as CreateFileW, WriteFile.



x86 imports two DLLs, KERNEL32.DLL and USER32.DLL. However, unlike x64DLL, x86 injects malicious information into the system through functions such as InjectSynthticPointInput, InjectTouchInput.



Manifest has 1 resource. It requests privilege "asInvoker" without UI access.

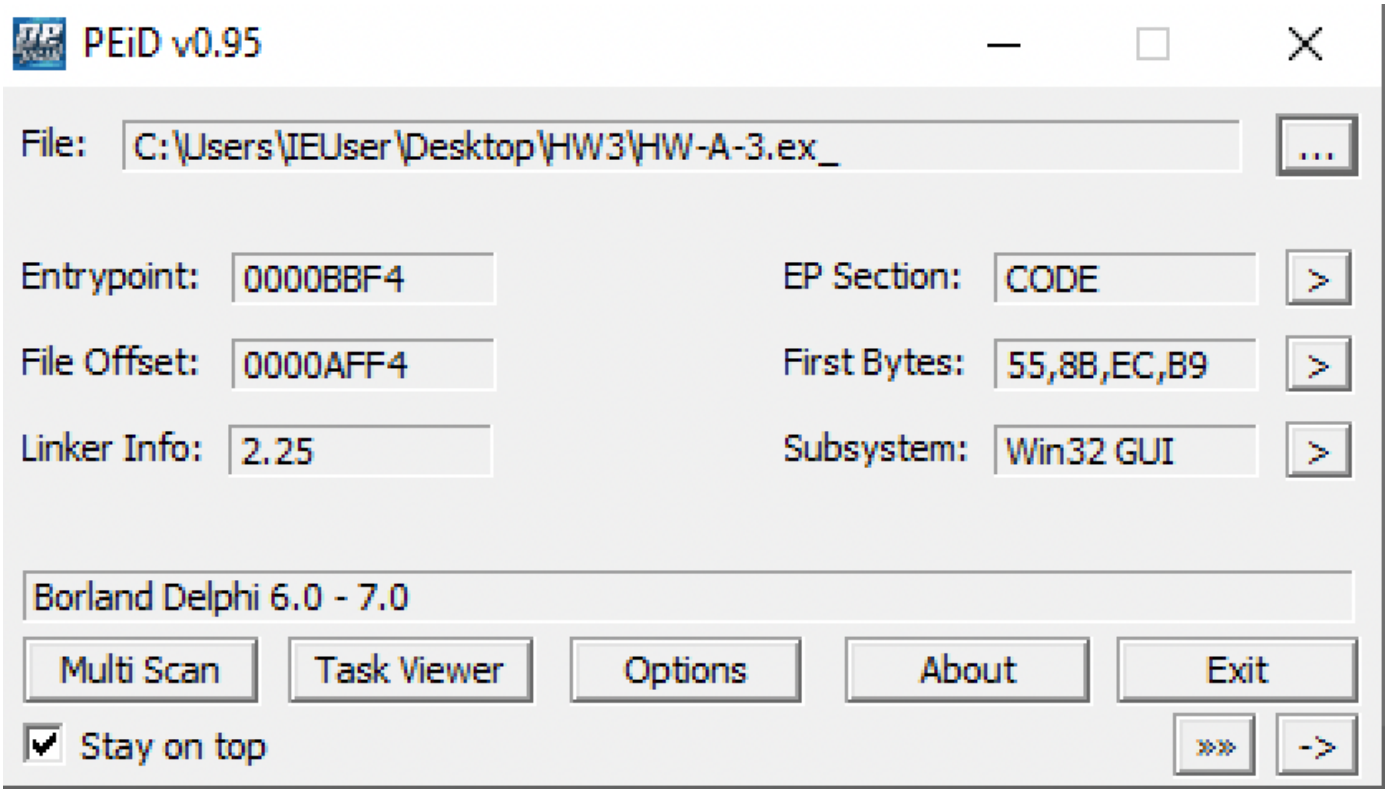


Analyzing HW3-A-3.ex_

Task 3: Answer the following questions after analyzing HW-A-3.exe

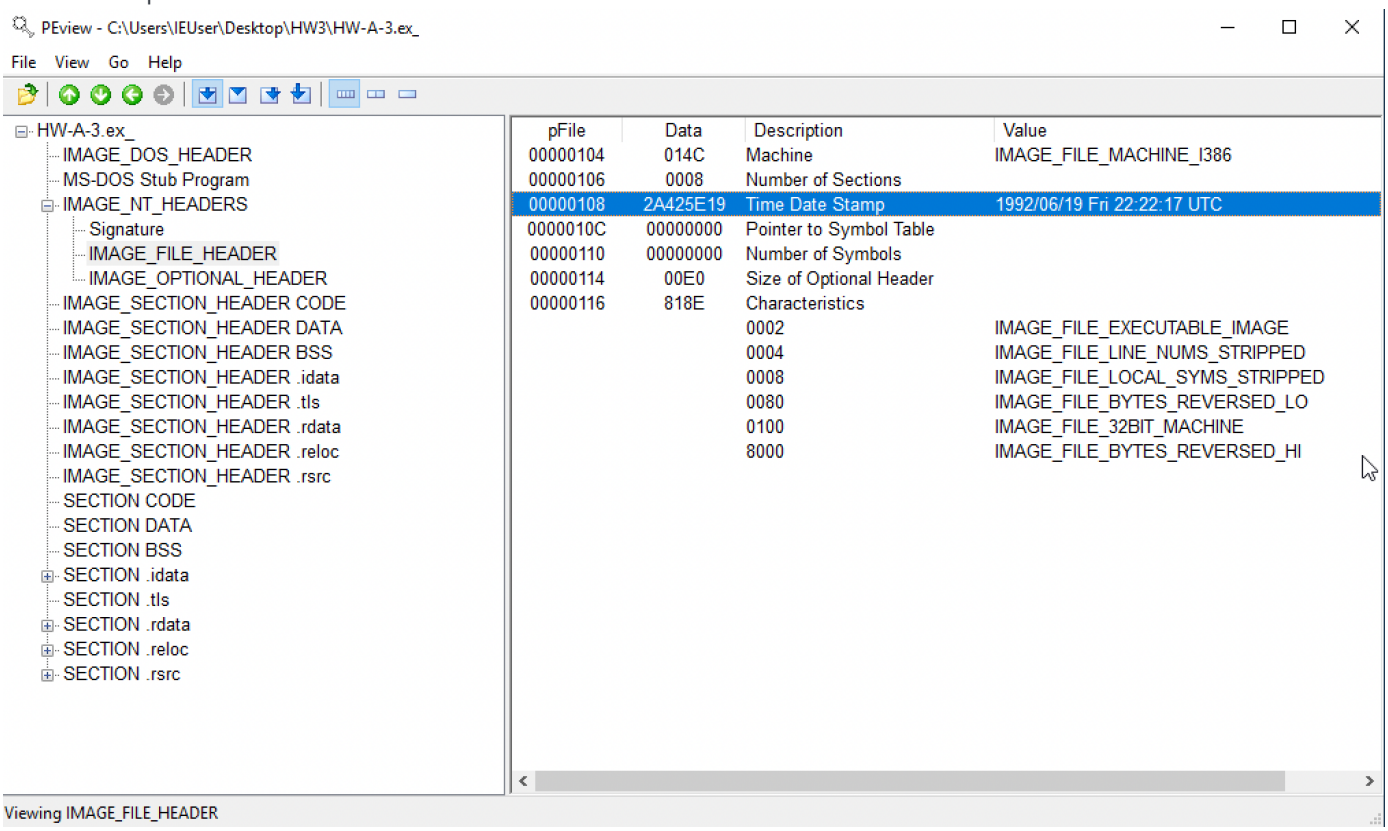
1. Are there any indications that this file is packed or obfuscated?

No, it is not packed. The program is written in Borland Delphi version 6.0-7.0.



2. When was this program compiled?

It was complied on 1992/06/19 Fri 22:22:17 UTC

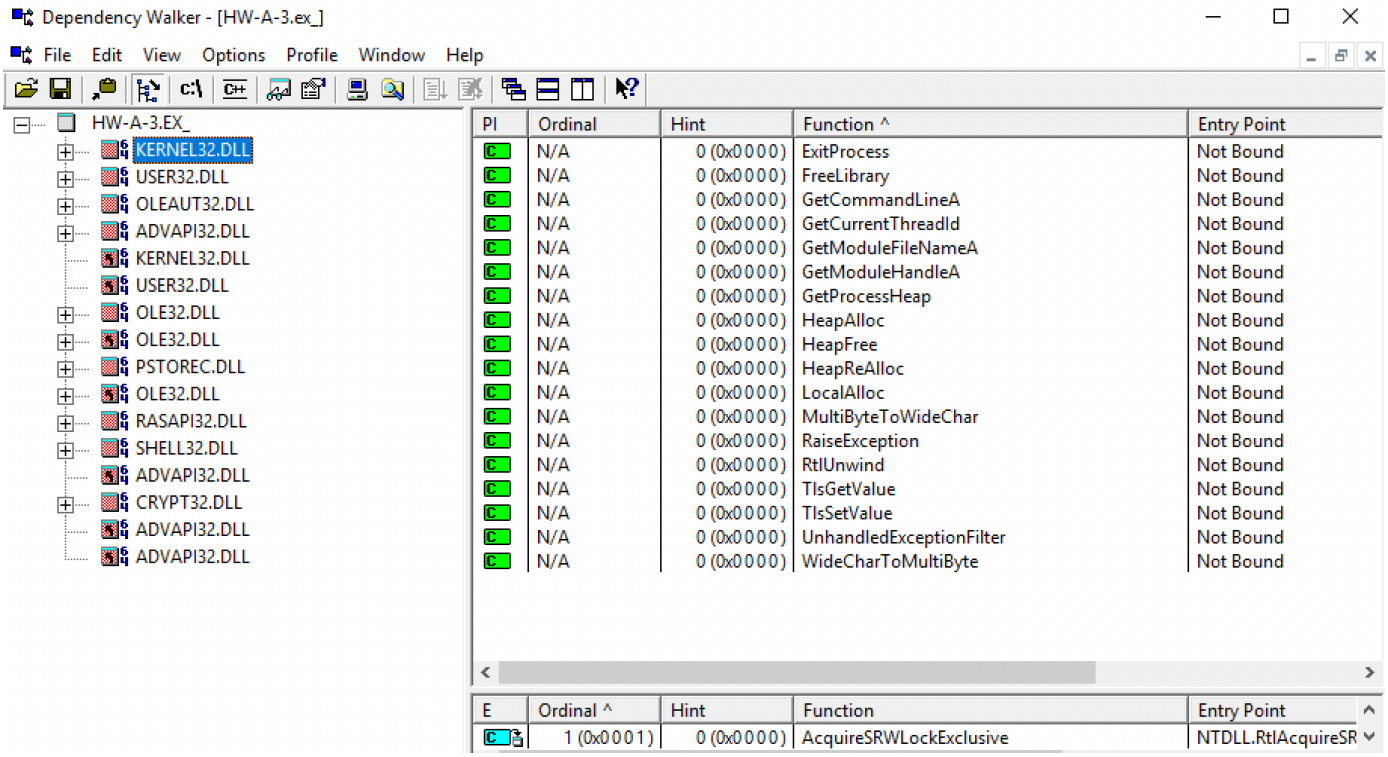


3. Do any imports hint at this program's functionality? If so, which imports are they and what do they tell you?

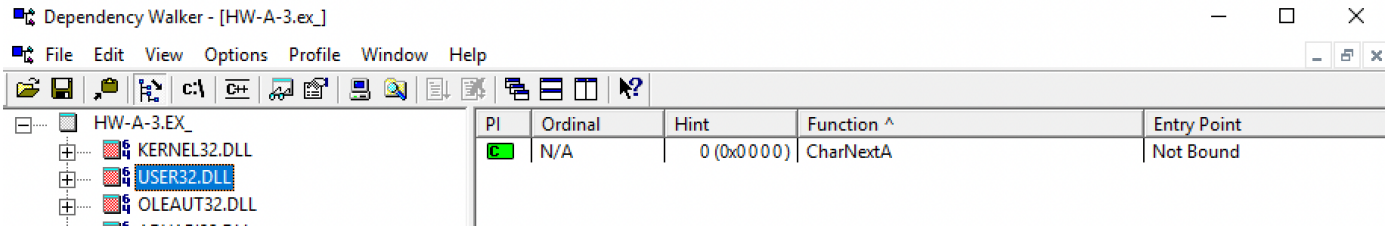
Program's overall functionality: steal user's credentials and password and send it back to the attack on remote device.

Detailed Analysis:

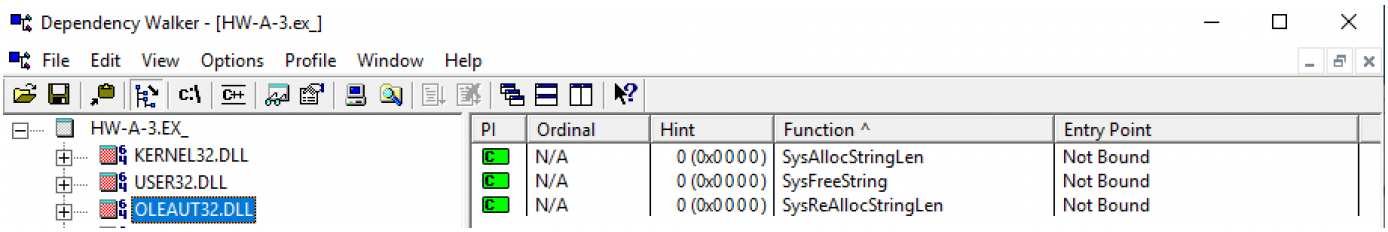
KERNEL32.DLL is the most basic import to manipulate files.



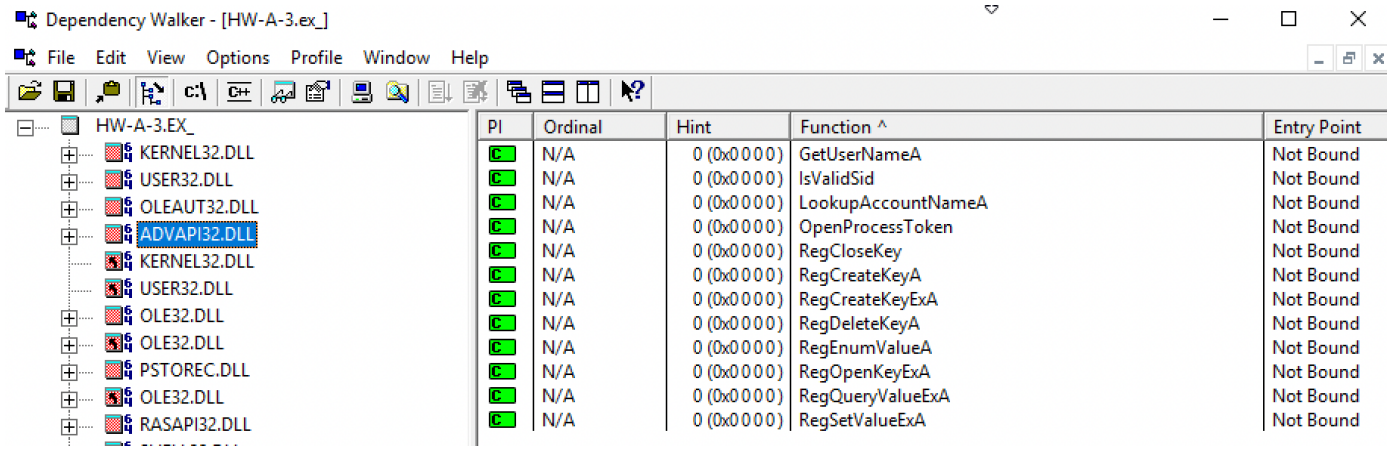
CharNextA from USER32DLL gets the next character input from USER



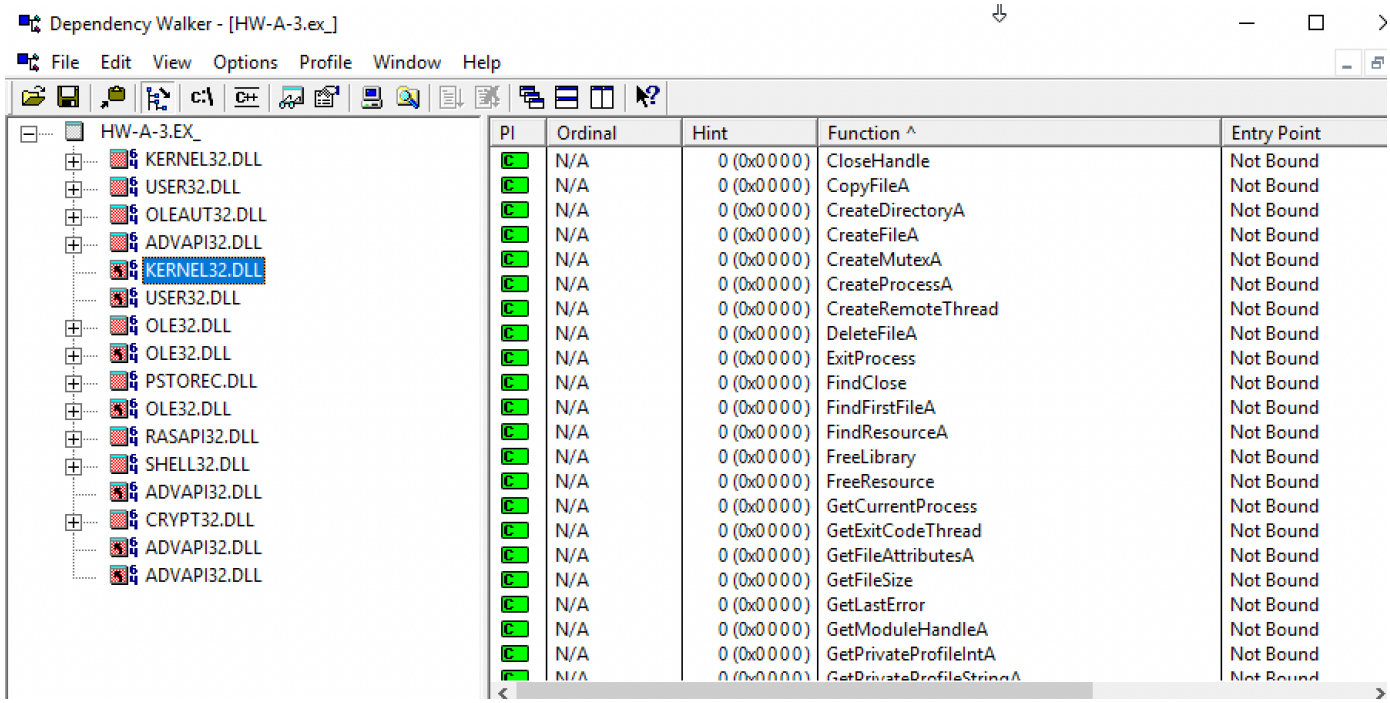
OLEAUT32.DLL allows applications to handle files and information created by other applications. It indicates that the malware is talking to some software.



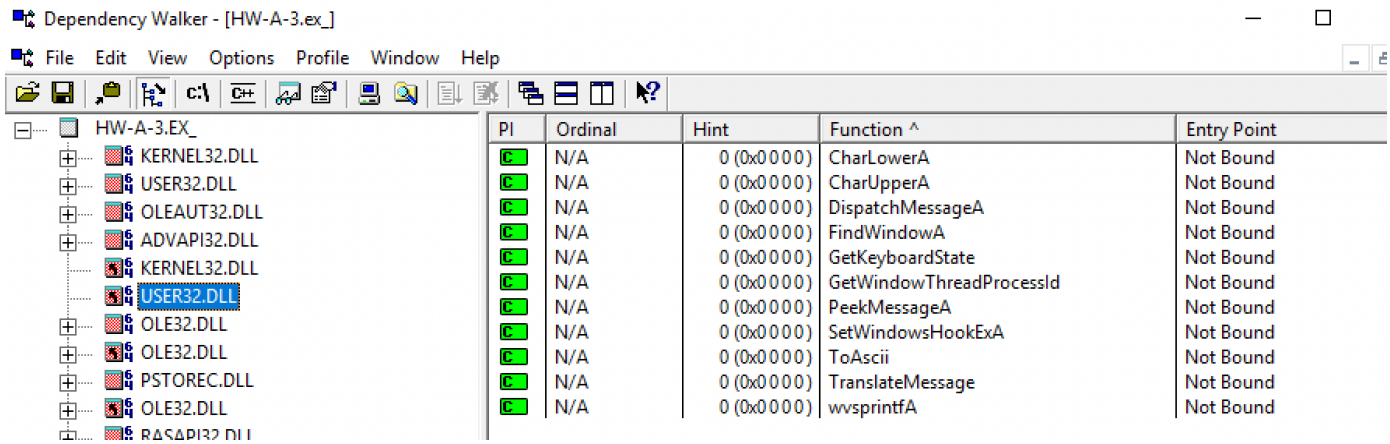
ADVAPI32.DLL: We can assumes that it tries to access protected files by adjusting to admin special permissions.



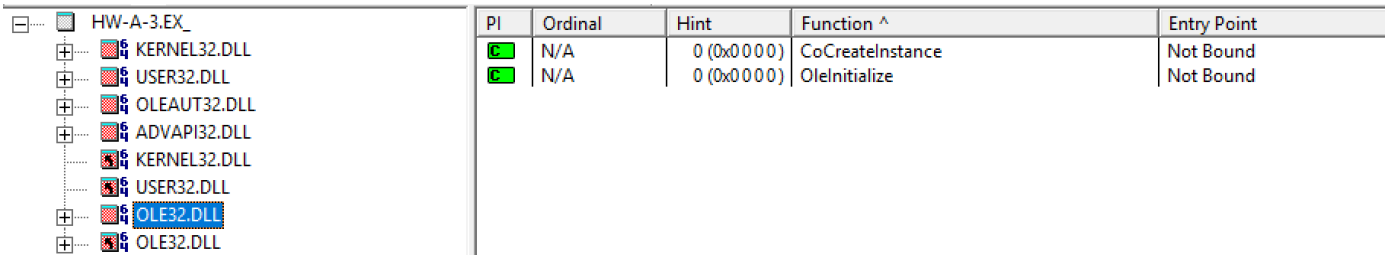
KERNEL32.DLL Creates a directory(CreateDirectoryA) and is reading, writing, executing(CreateProcessA, FindFirstFileA, FindResourceA) resources



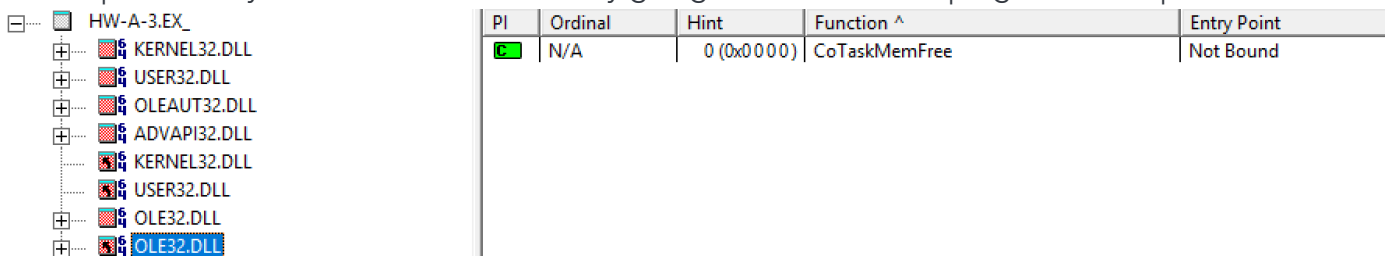
Most likely evesdropping(PeekMessageA) on the keyboard (GetKeyboardStateA) input from user.



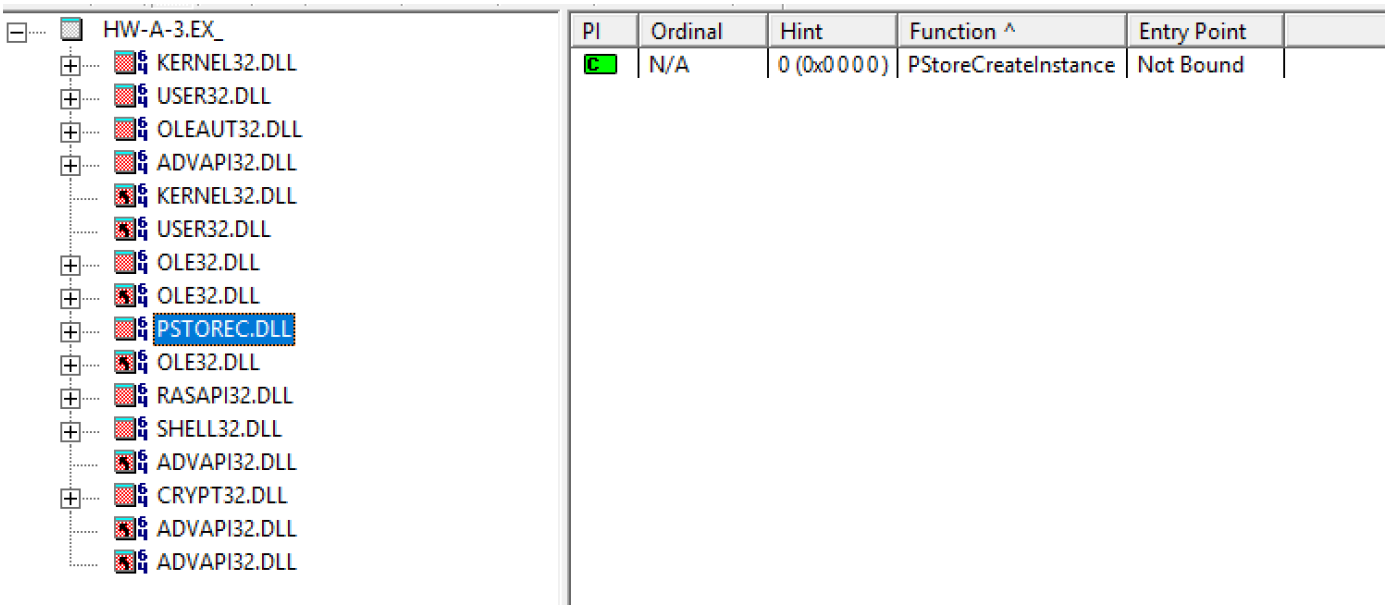
CoCreateInstance is used to access to COM functionality. Possibly to launch internet explorer and access an address



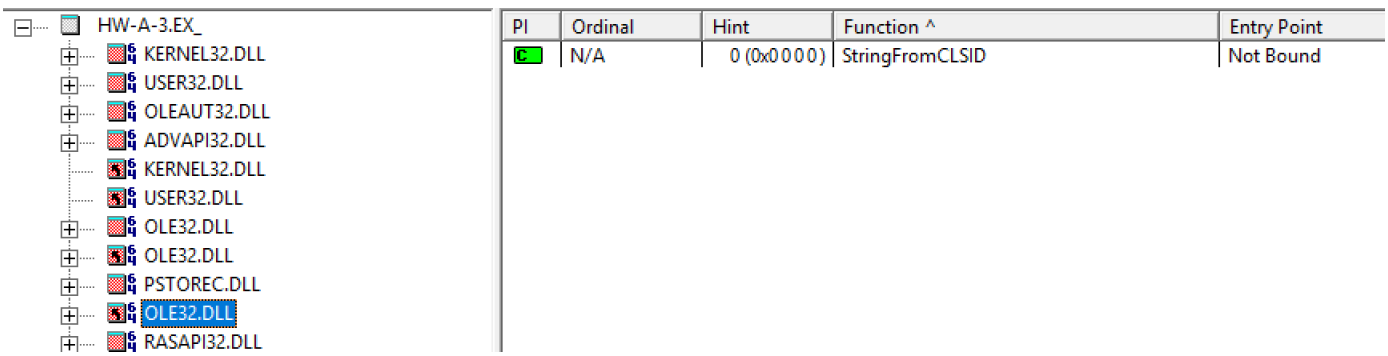
Frees up a memory block. Malware most likely going to execute some program at this place



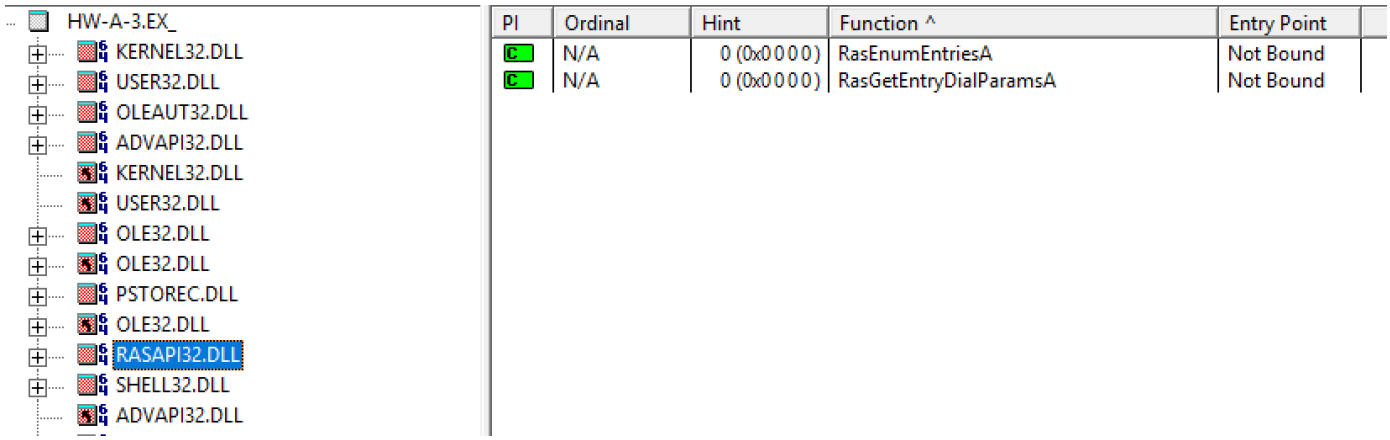
Access the legacy Windows data store, Pstore. If successful a call to the PStoreCreateInstance function to get the pointer to protected storage will be made to allow calls to functions that enumerate and retrieve any stored credentials.



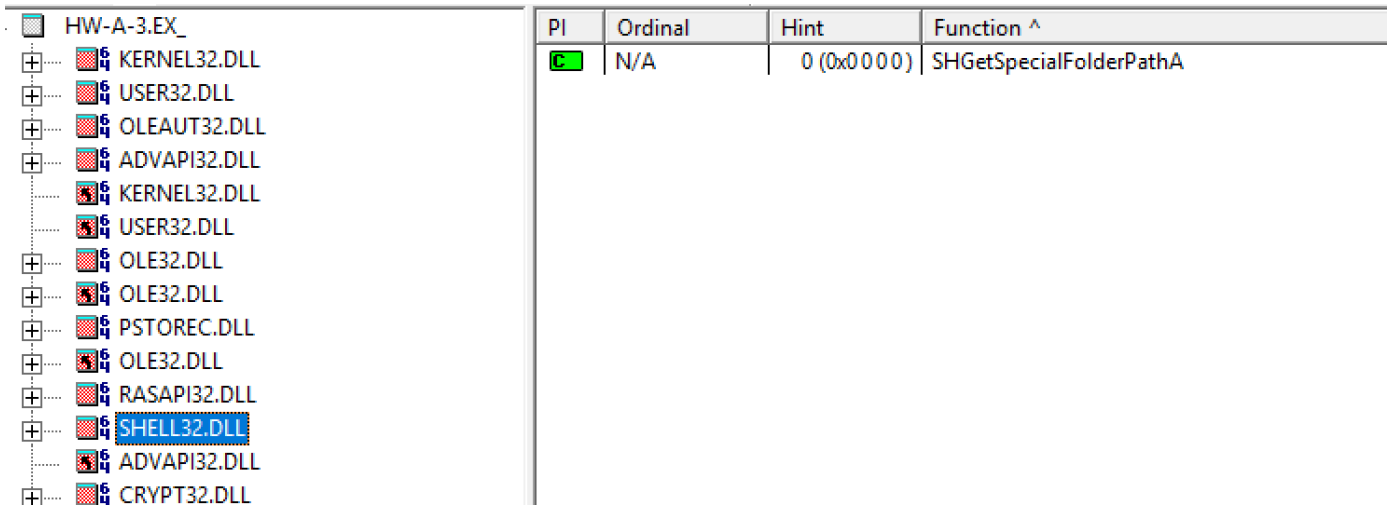
Converts CLSID into a string of printable characters. Allows the malware to run an application on Windows.



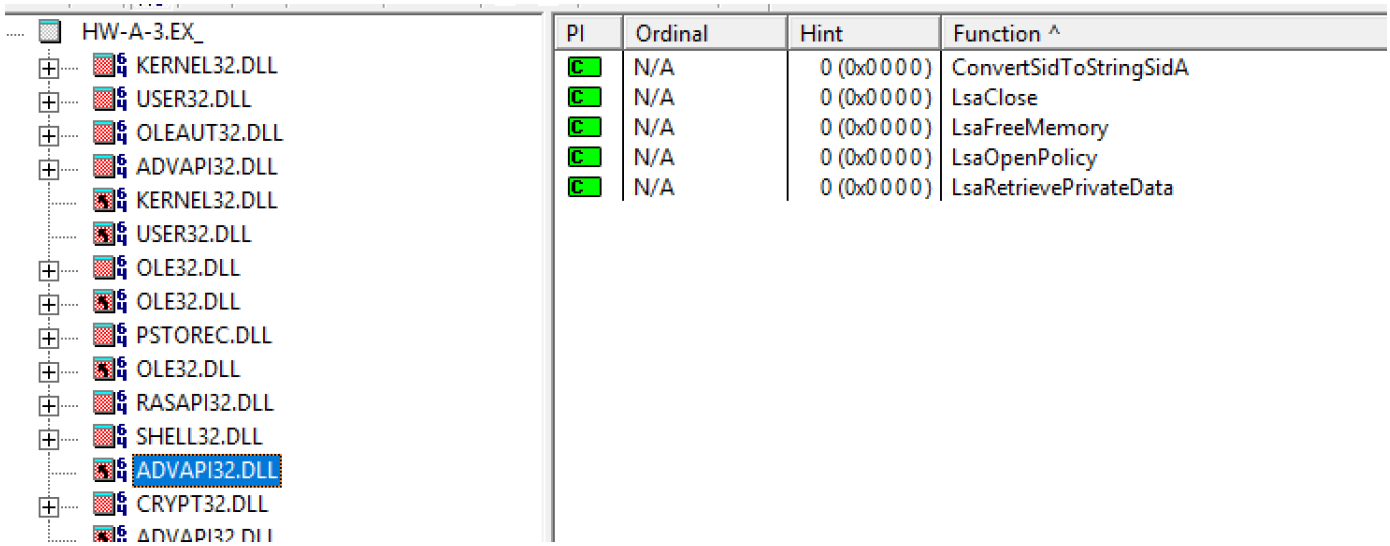
RAS is a remote access service. RasEnumEntriesA indicates a that the malware is using communication module to connect to the attacker's machine



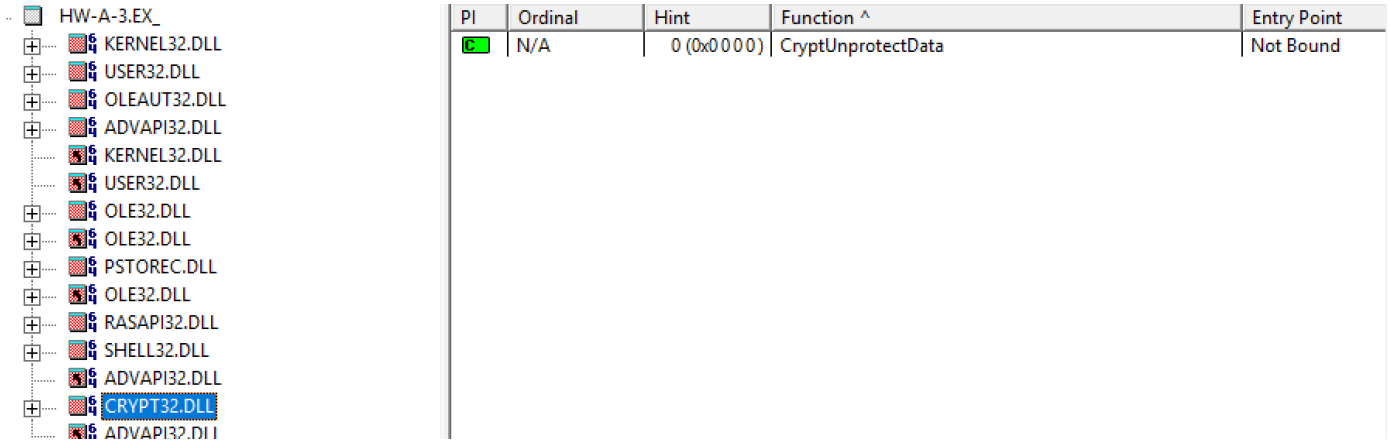
CSIDLs is passed the following functions as arguments: SHGetSpecialFolderPathA. The purpose is for malware to access important folders.



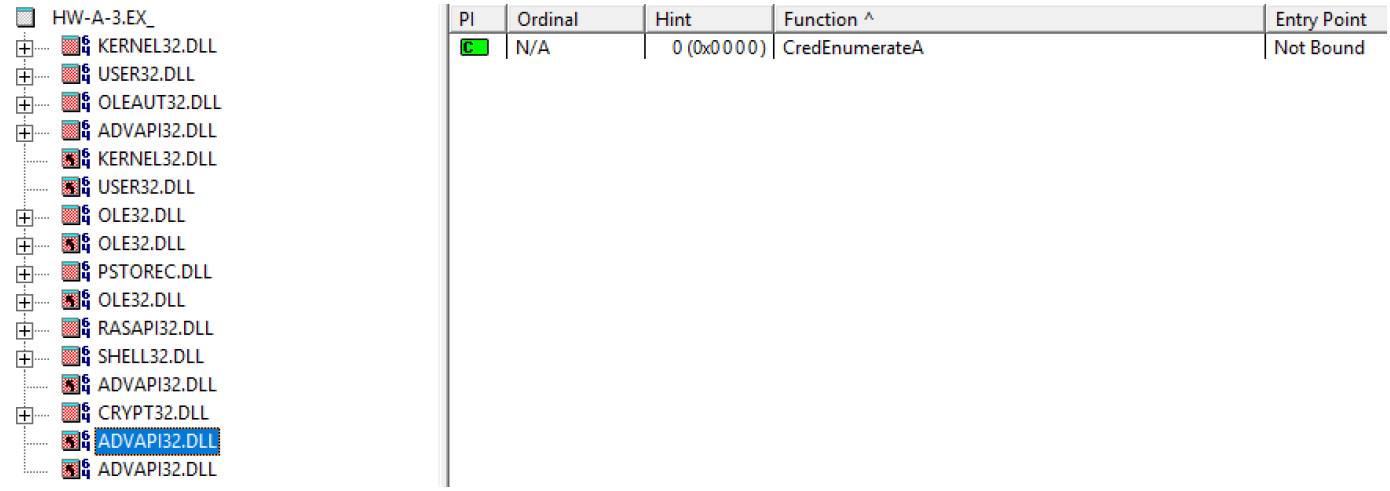
The malware retrives private data(LSARetrievePrivateData) which could be things like default password.



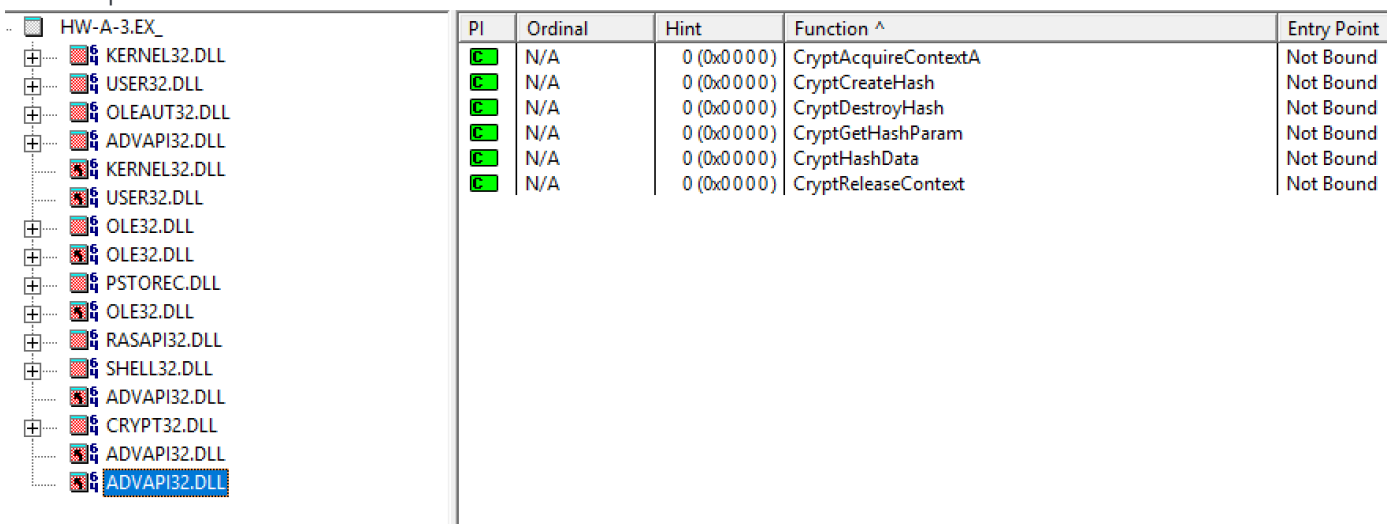
Malware steals credentials and cookies



CredEnumerateA API to identify and steal saved credentials.

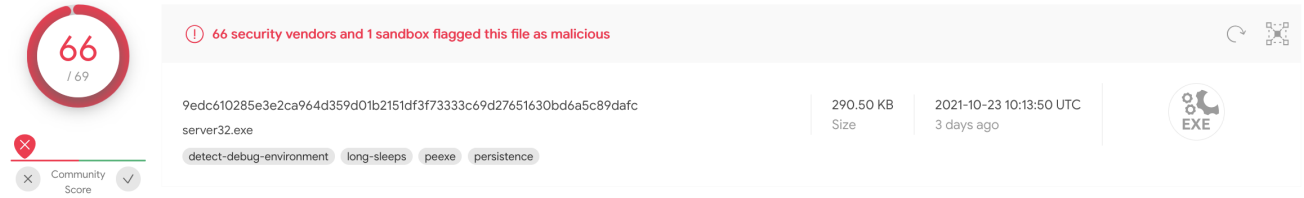


Clean up afterwards.



4. What can you observe using Basic Static Analysis techniques?

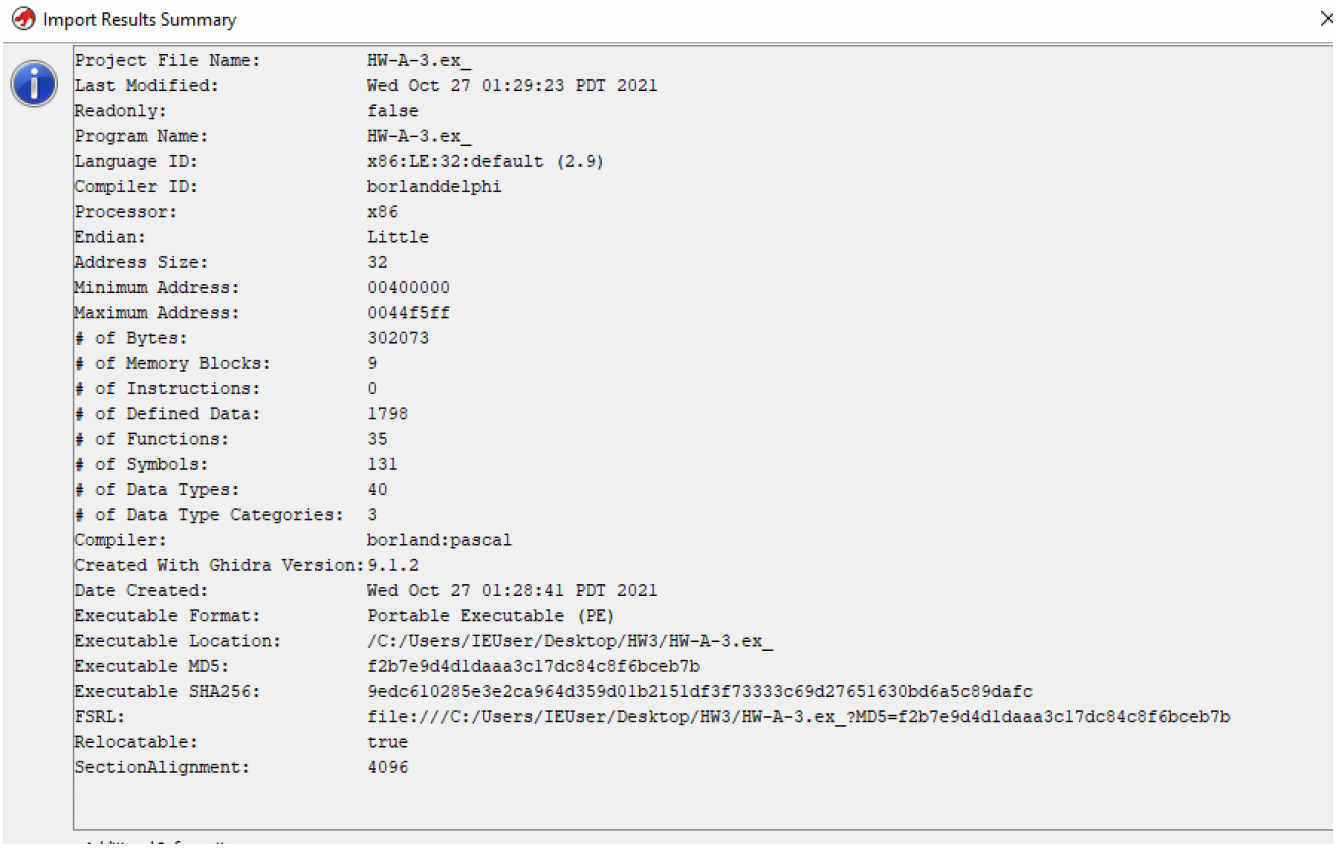
1. Matches 66/69 existing antivirus definition



2. Hashing, MD5 Executeable

```
f2b7e9d4d1daaa3c17dc84c8f6bceb7b
```

3. A few important information including date created, number of functions, symbols, and language used to write the program:

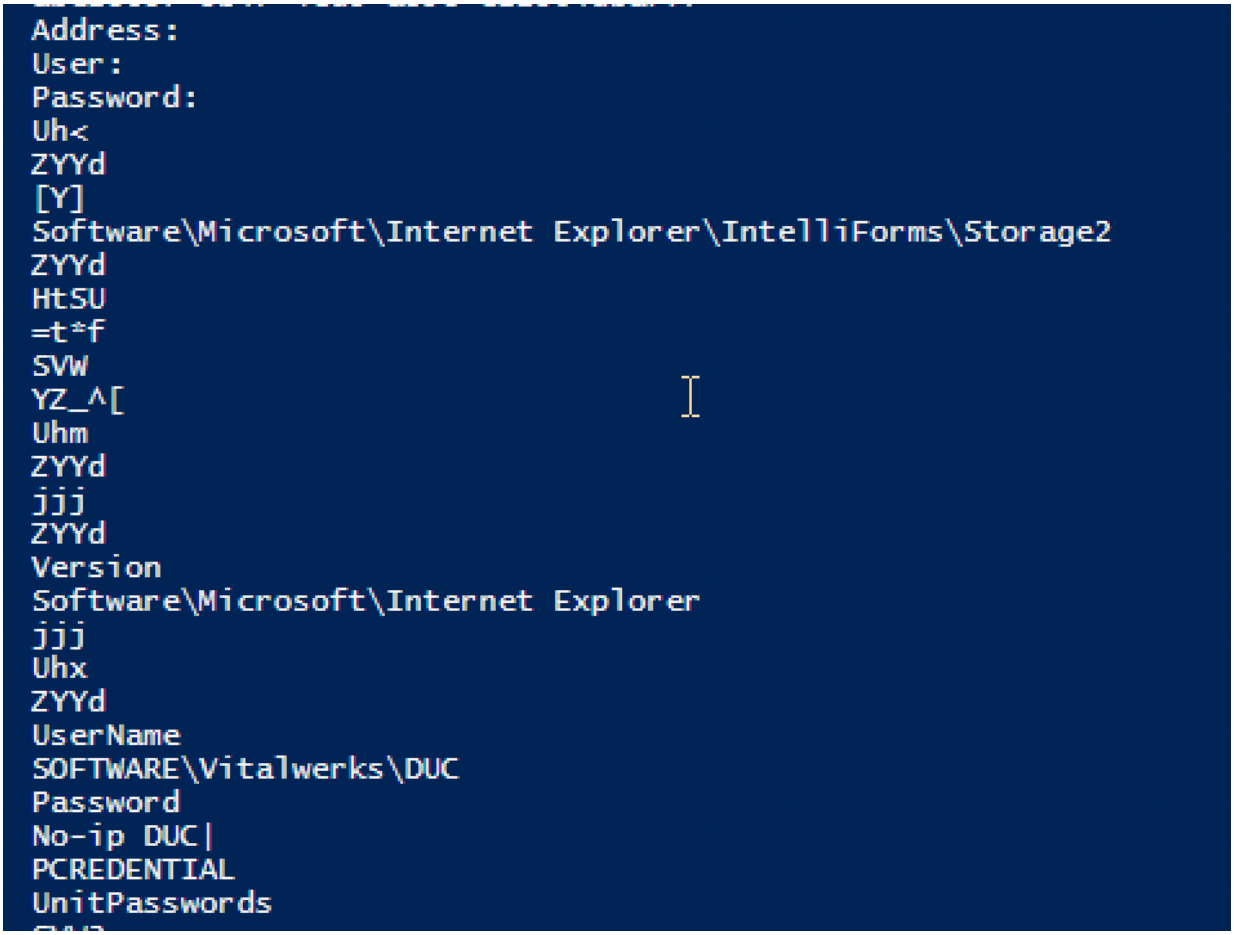


4. Its not packed.

5. 

    1. Interesting String

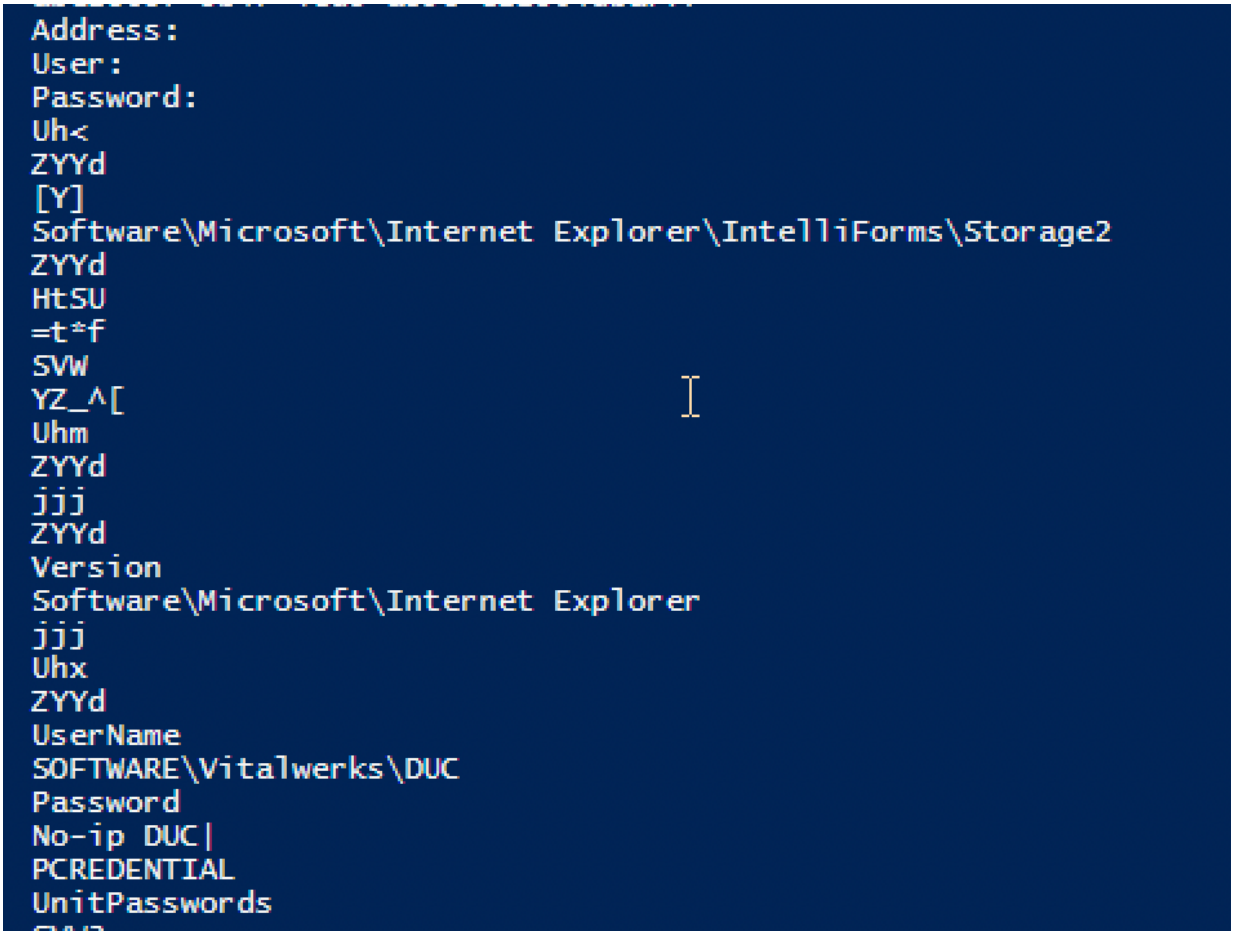
1. Calls for internet explorer: software\microsoft\internet explorer



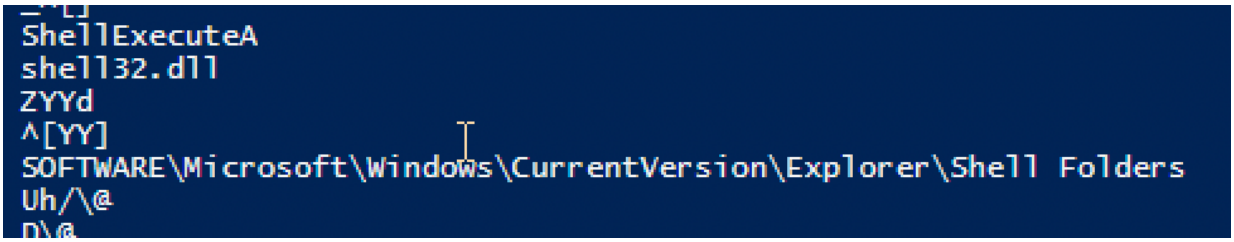
2. Injects a library : OUnitInjectLibrary

```
Base64
KWindows
System
SysInit
UTypes
,uIE7_decode
UnitDiversos
TlHelp32
gUnitServerUtils
wcrypt2
KuURLHistory
sActiveX
3Messages
CryptApi
uRASReader
IEpasswords
Pstoreclib
SPSTORECLib_TLB
'unitStartup
UnitComandos
deleteUnit
UnitPasswords
EditSvr
UnitInstalacao
UUnitSettings
RUnitVariaveis
6UnitSandBox
OUnitInjectLibrary
```

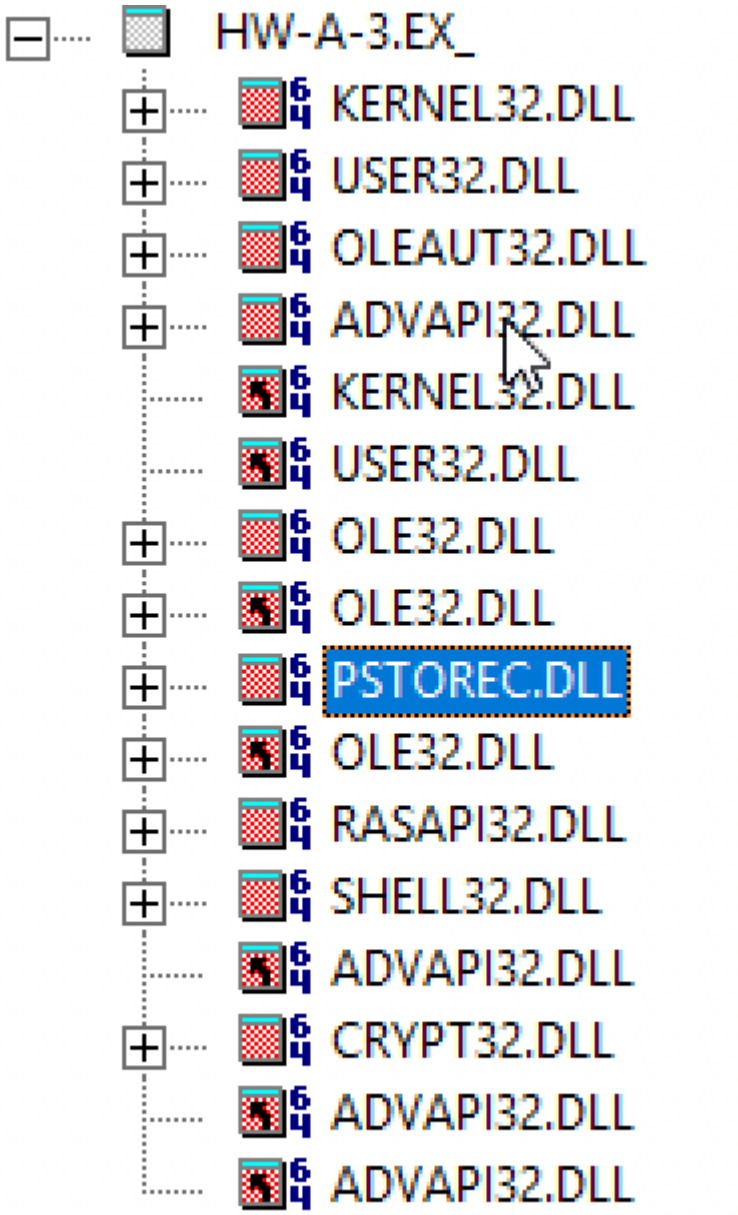
3. Program asks for User: Password: and Address:



4. Looks for shell:



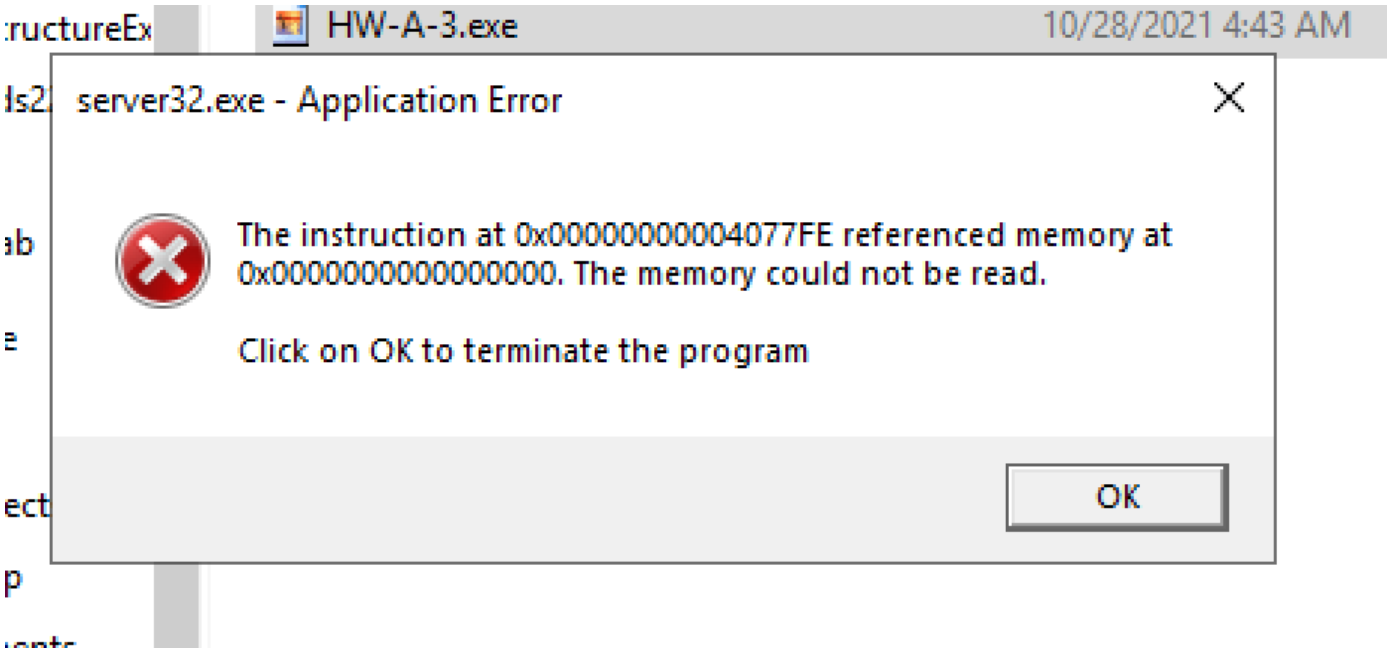
6. Import Functions were analysed in detail in question 4.



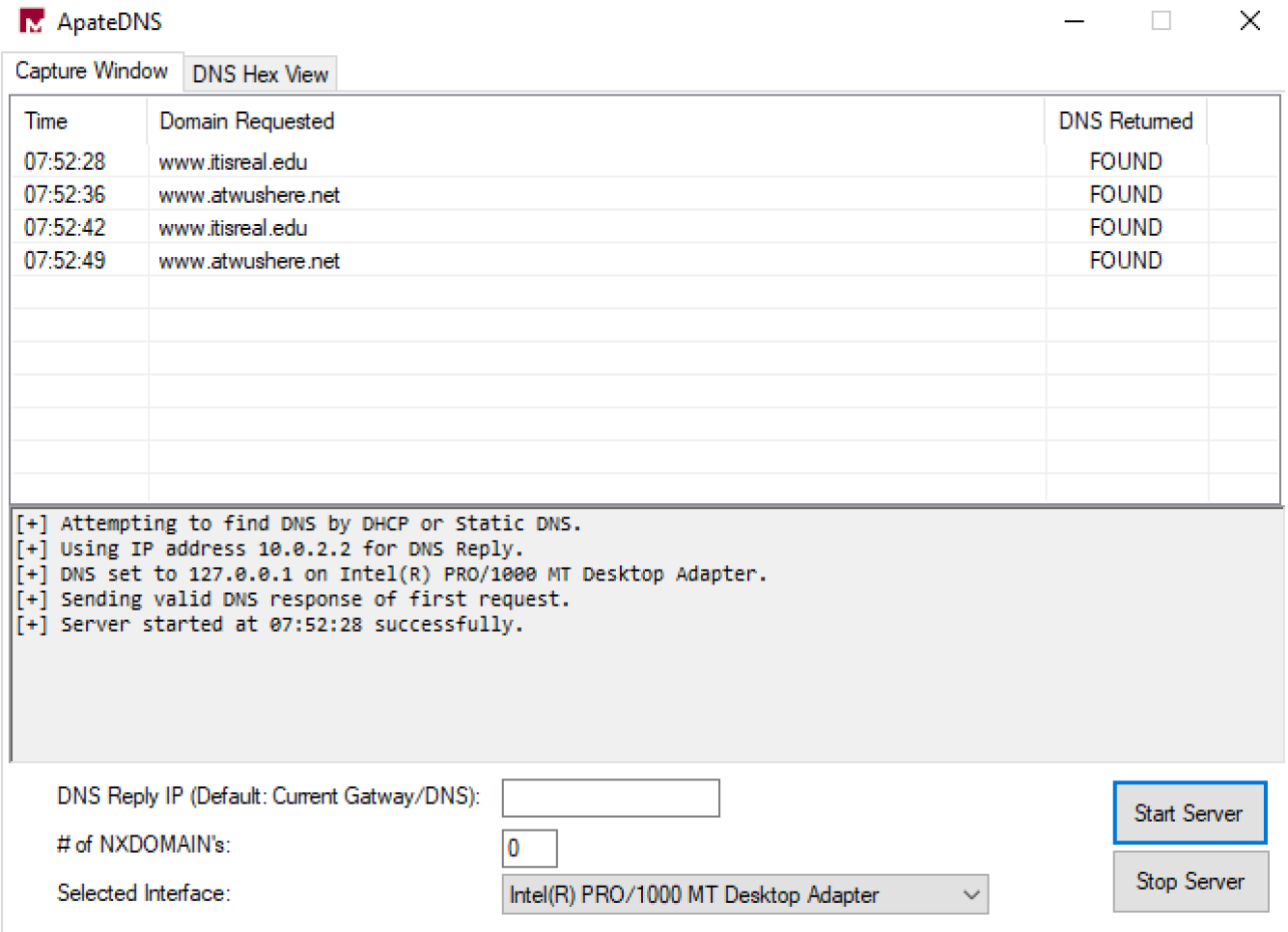
1. summary: steal user's credentials and password and send it back to the attack on remote device connected via internet explorer.

5. What do you observe through dynamic analysis?

Running the program shows "Application Error"



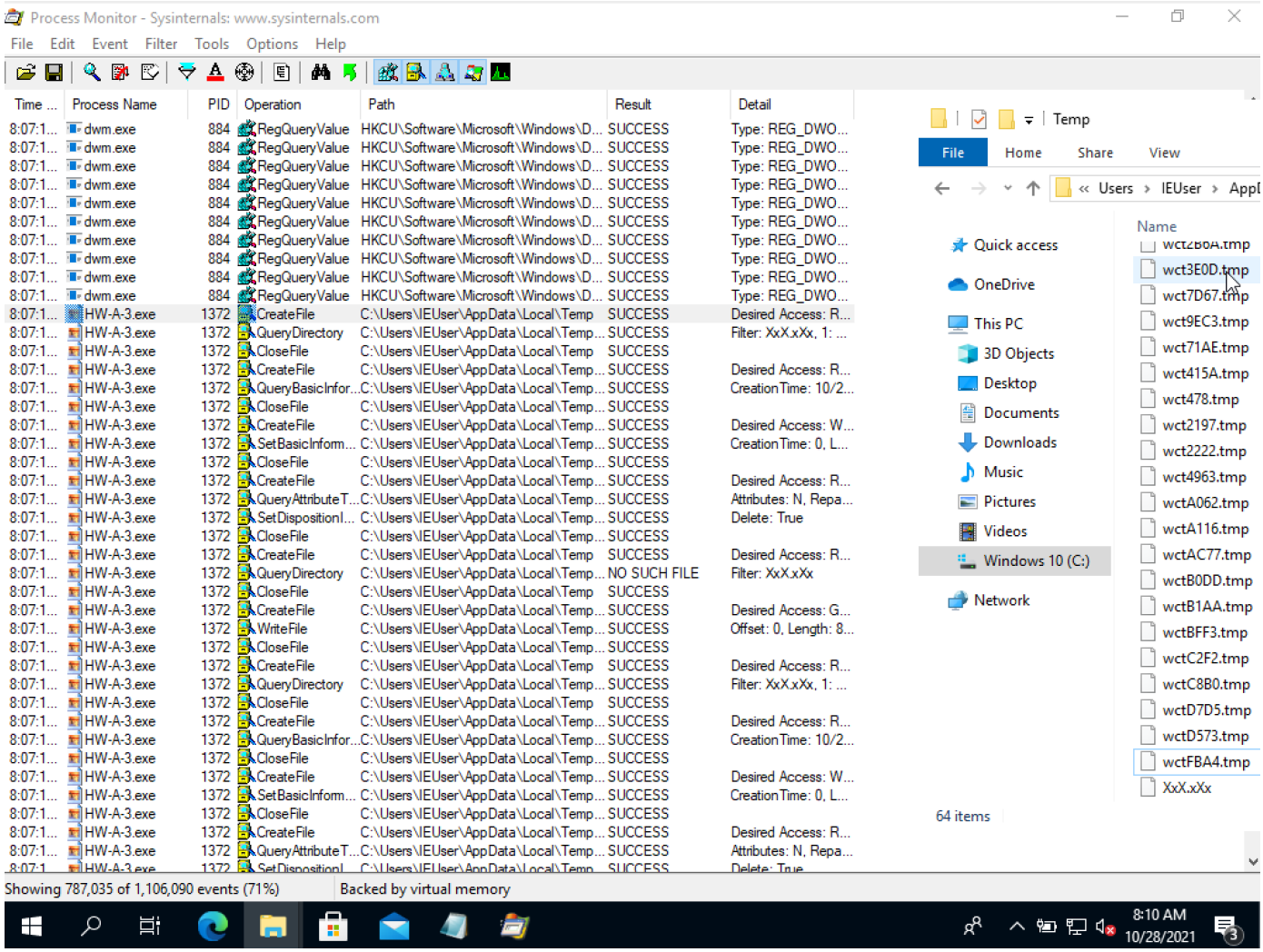
1. Examine ApateDNS to see if DNS requests were performed.
   Using ApateDNS, we see that DNS requests are preformed to "www.isitreal.edu" and "www.atwushere.net" every few seconds.



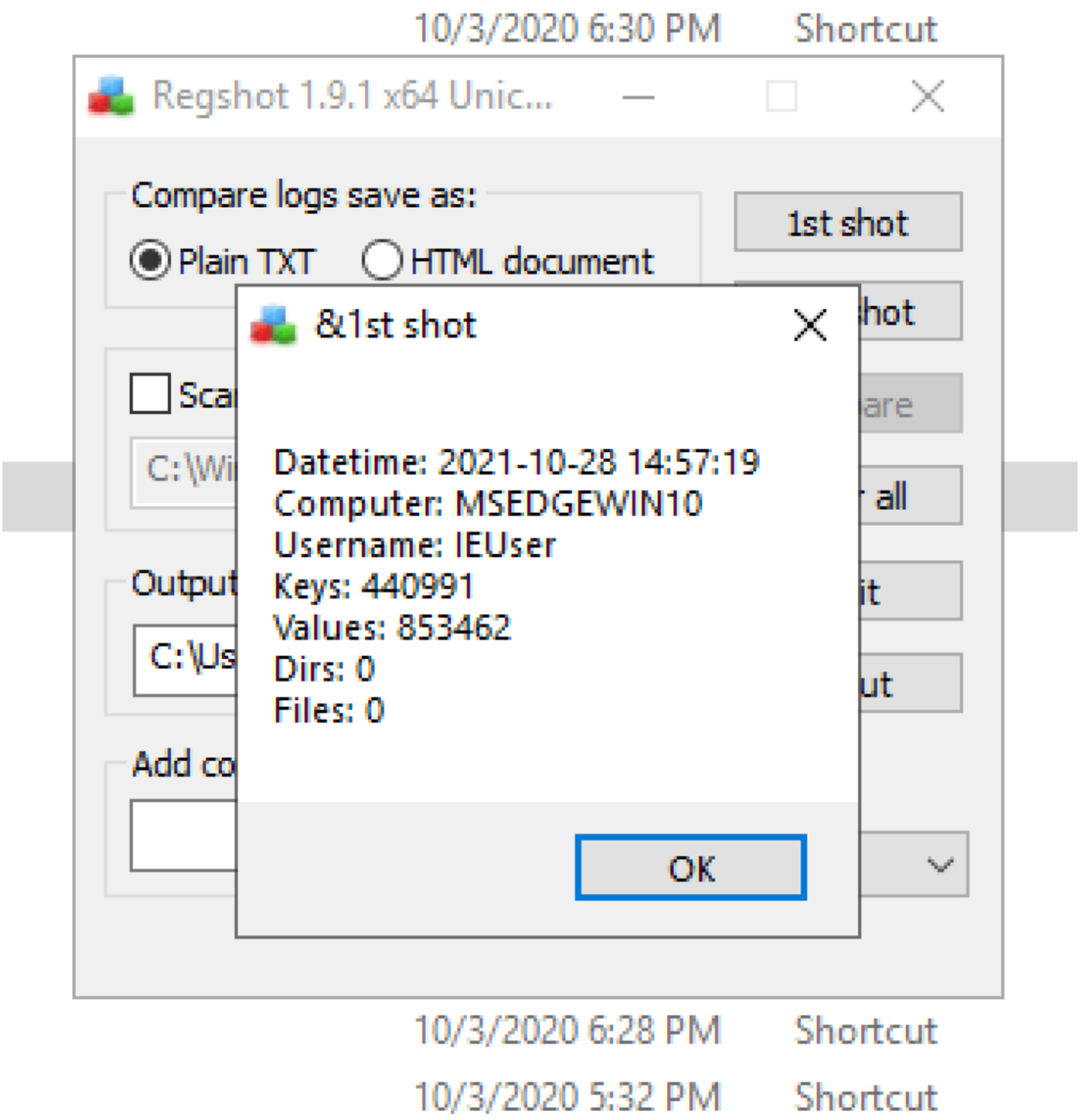
2. Review the procmon results for file system modifications.

Analysing with procmon, we see that the malware creates a lot of files in C:\Users\IEUser\AppData\Local\Temp. It attempts to read the file XxX.xXx specifically which is used to

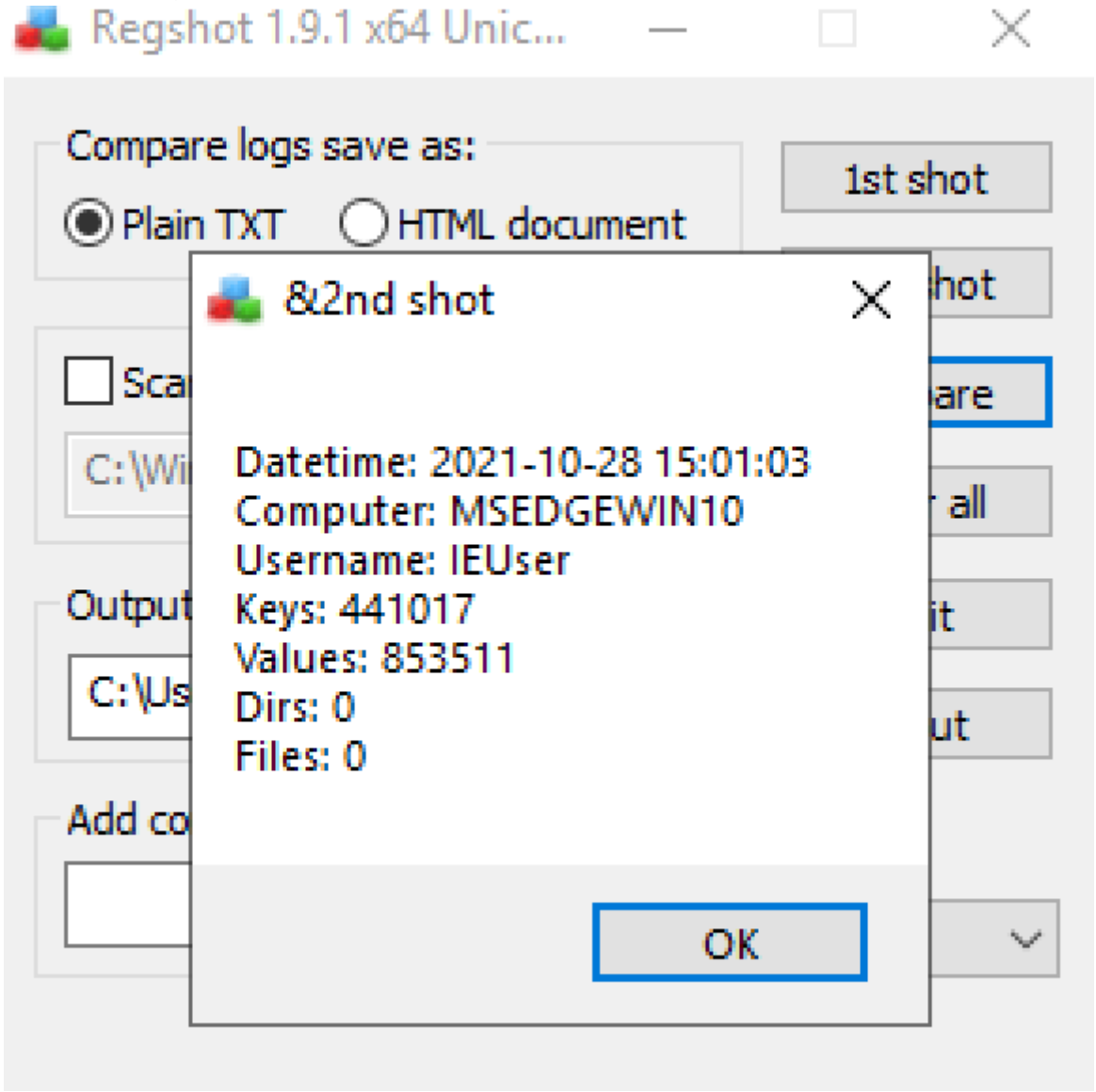
block the mouse.



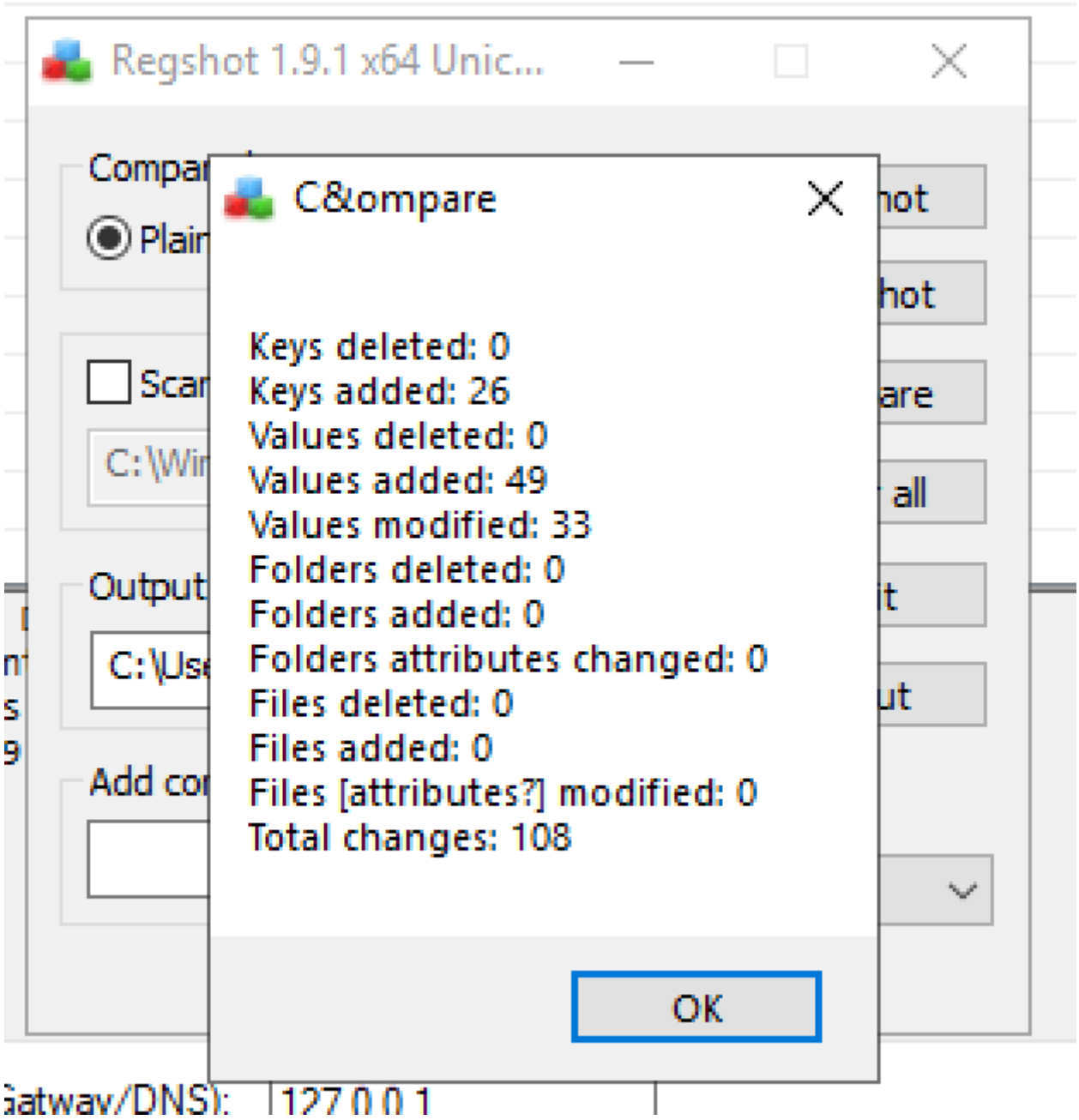
3. Compare the two snapshots taken with Regshot to identify changes.



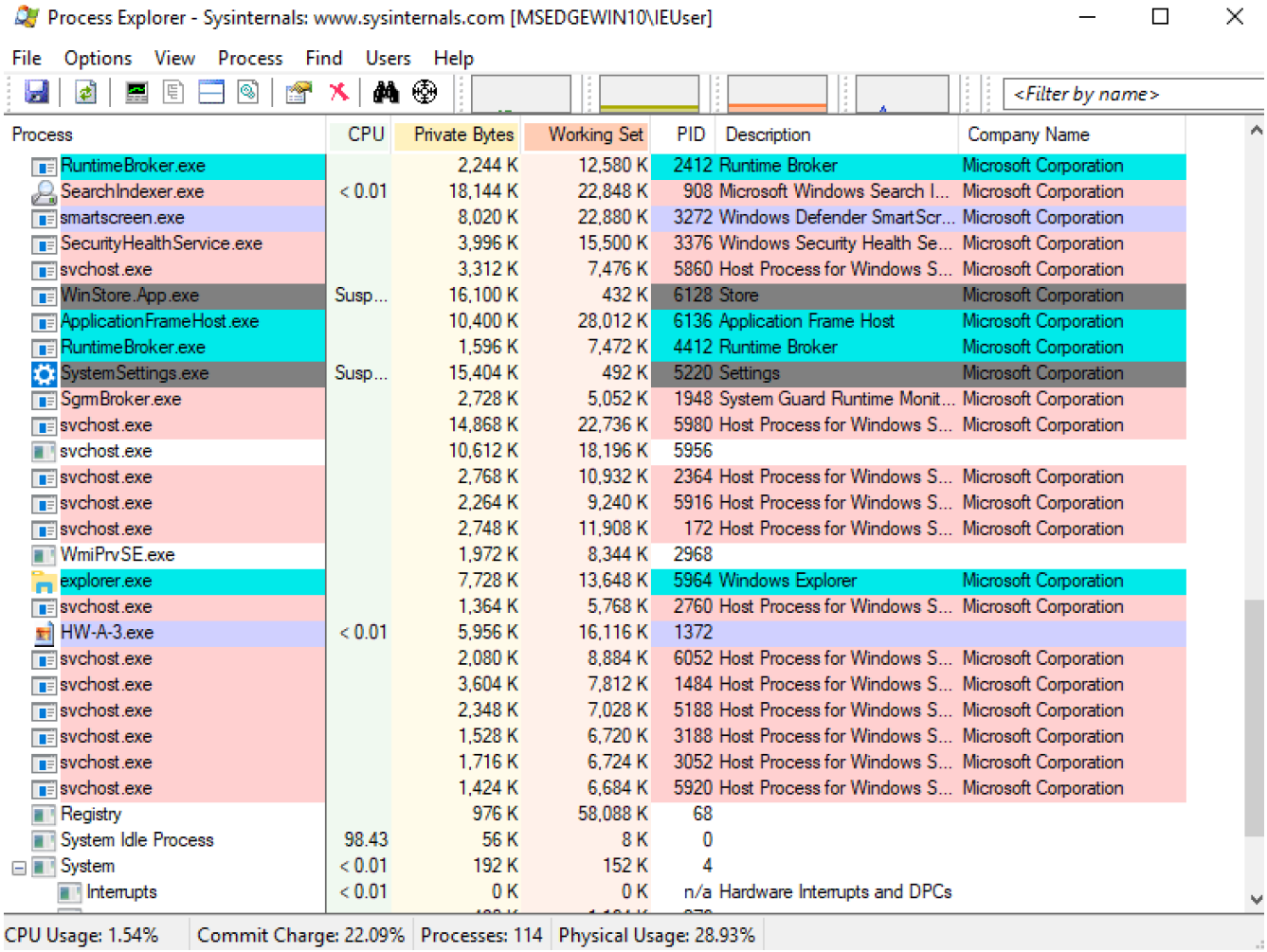
After installing FakeNet



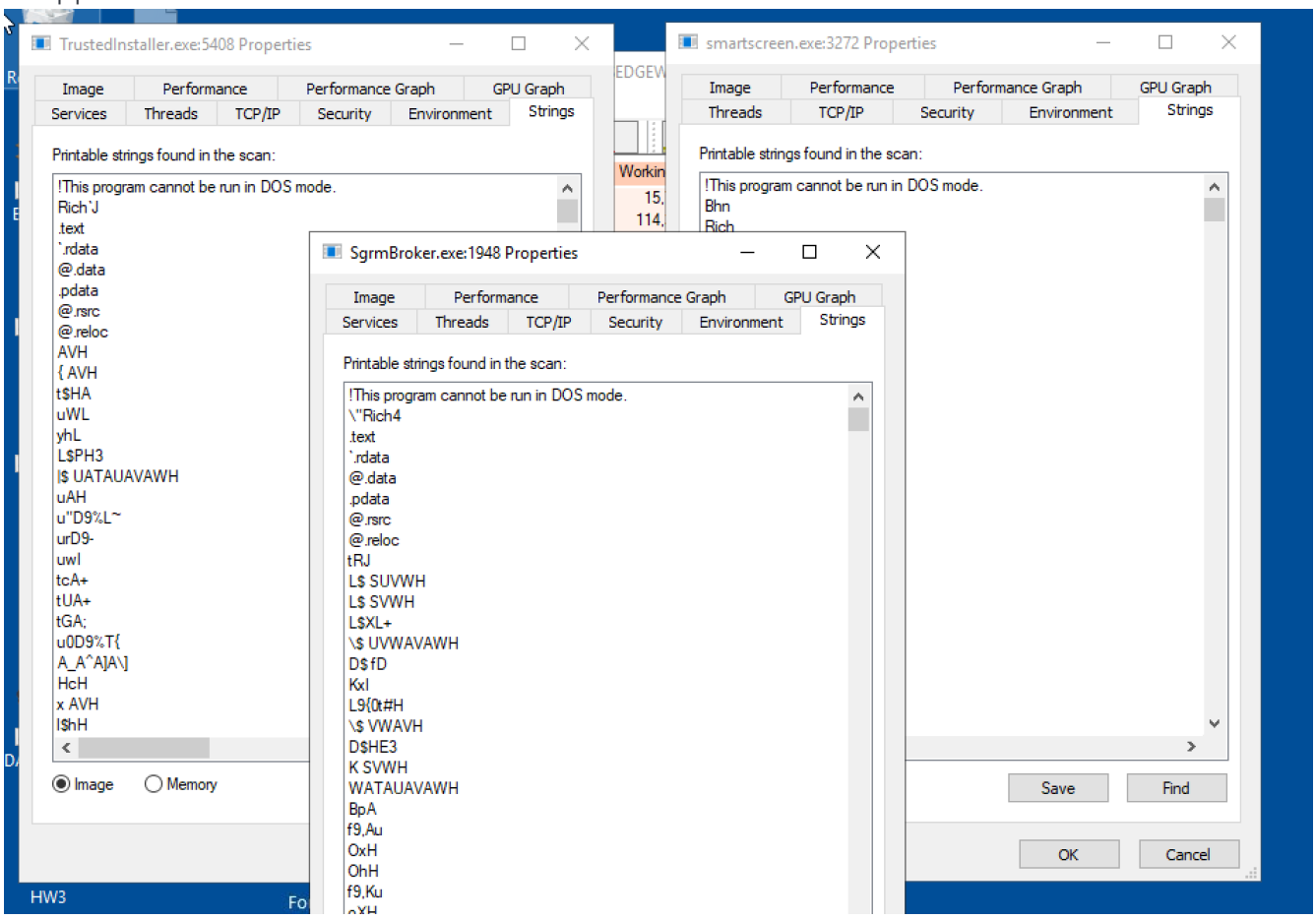
Comparing the two there were several keys added, values deleted, and values added.

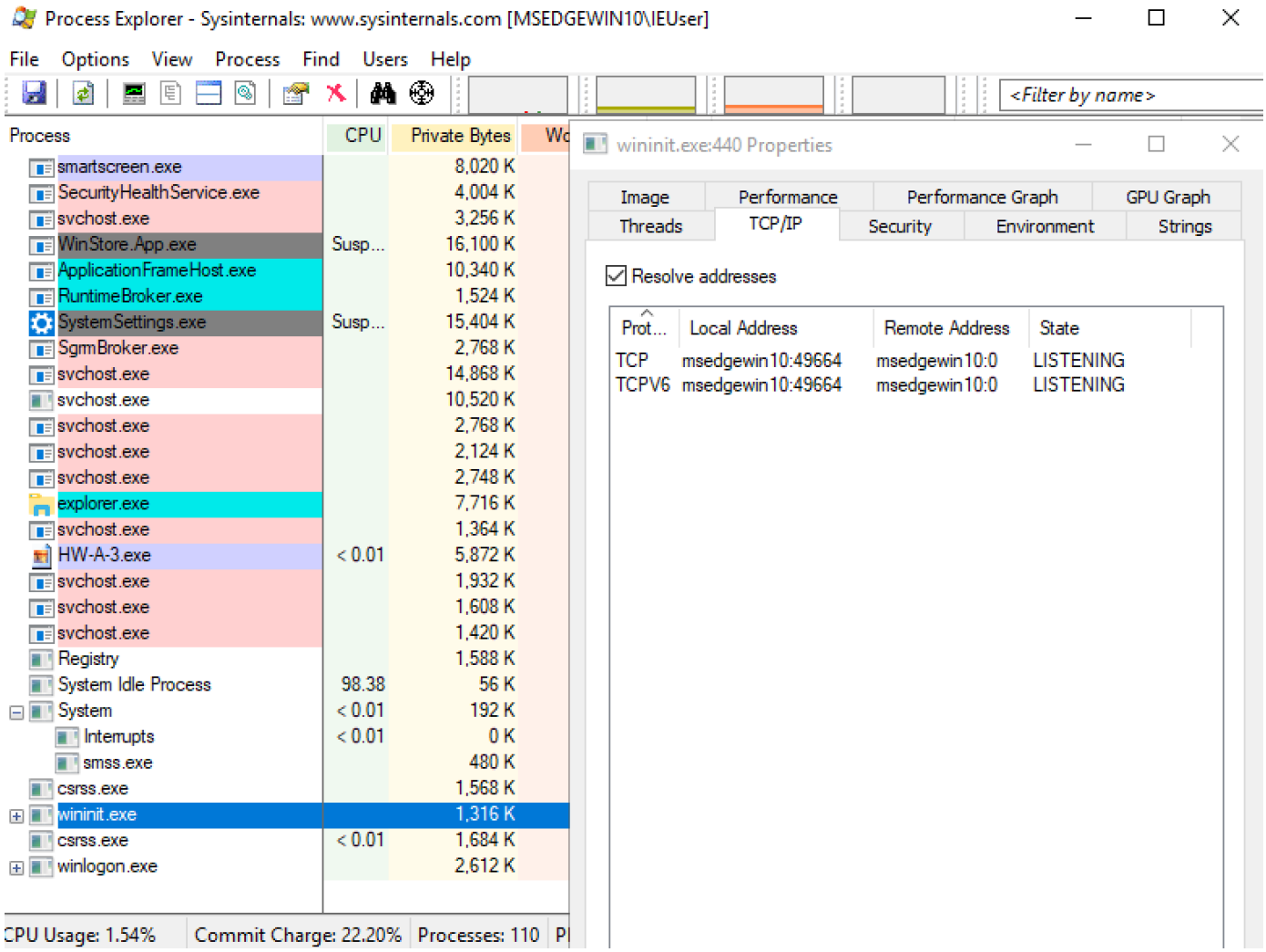


4. Use Process Explorer to examine the process to determine whether it creates mutexes or listens for incoming connections



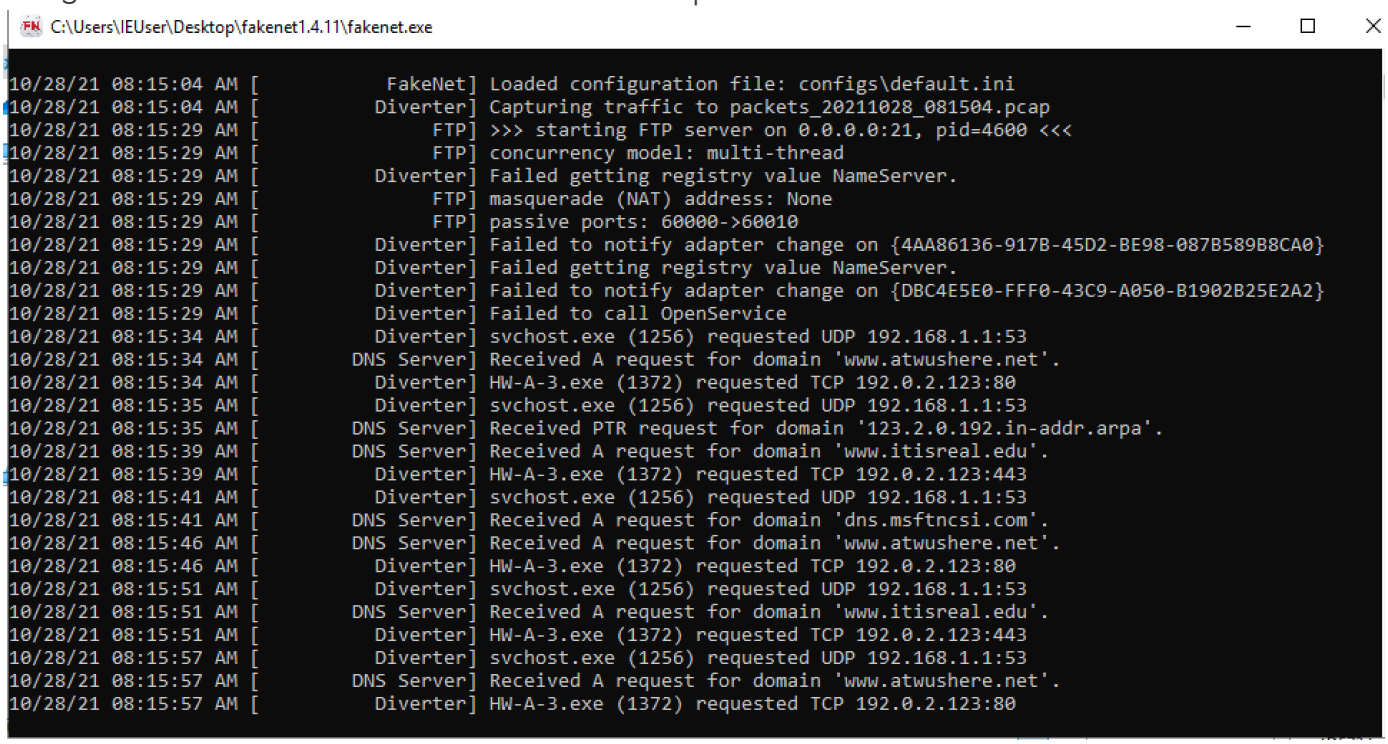
There are several instances of orphaned svchost.exe which are suspicious and unusual. They start new executable files such as TrustedInstaller.exe, SGrmBroker.exe, smartscreen.exe. Which disappears after some time.





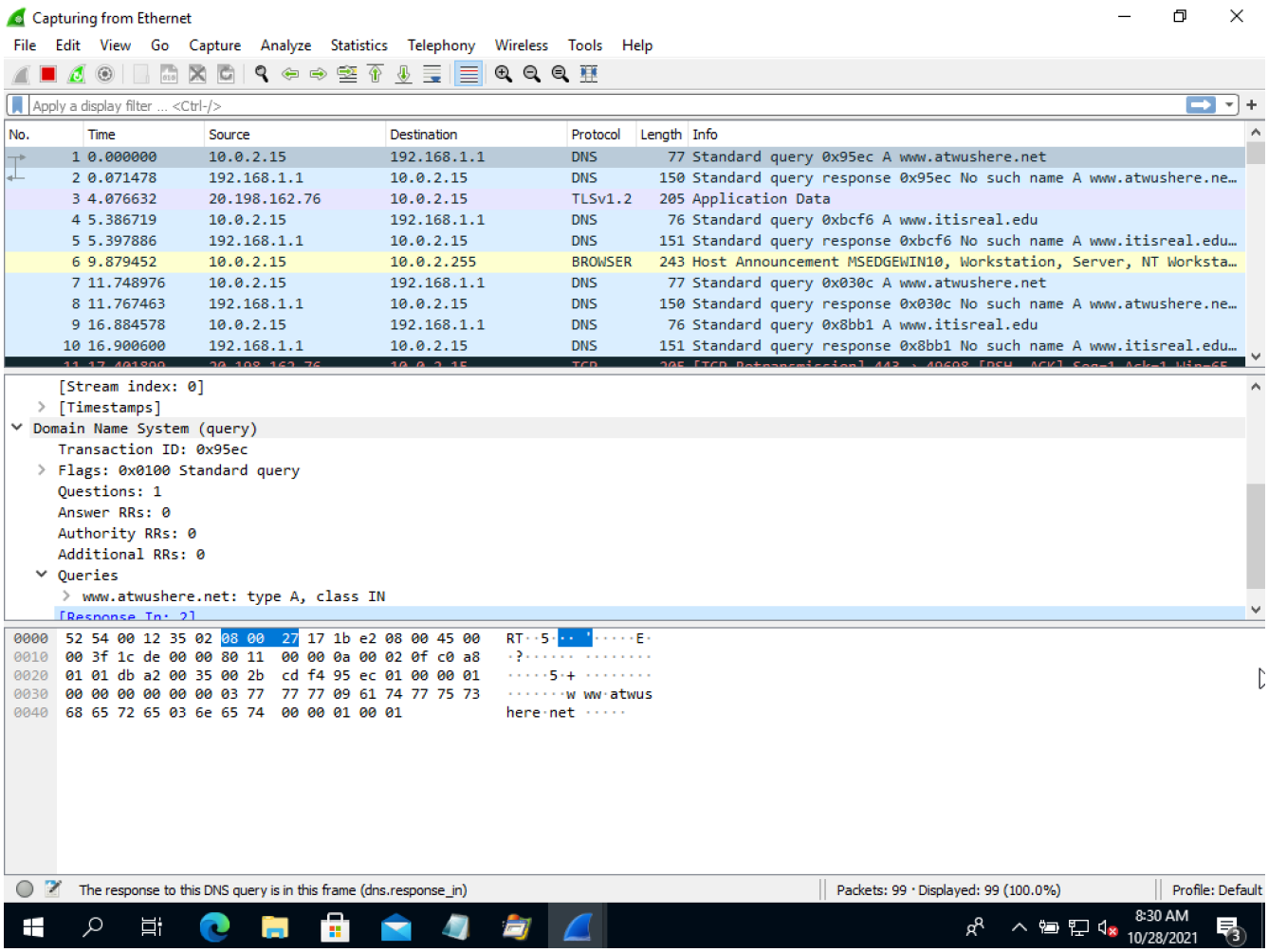
It uses wininit.exe to snoop on the user. As evident by the TCP/IP connection

Using FAKENET we can see that the malware attempts to establish TCP connection to 192.0.2.123



6. Review the Wireshark capture for network traffic generated by the malware.

   The malware opens as evident in "Application Data"



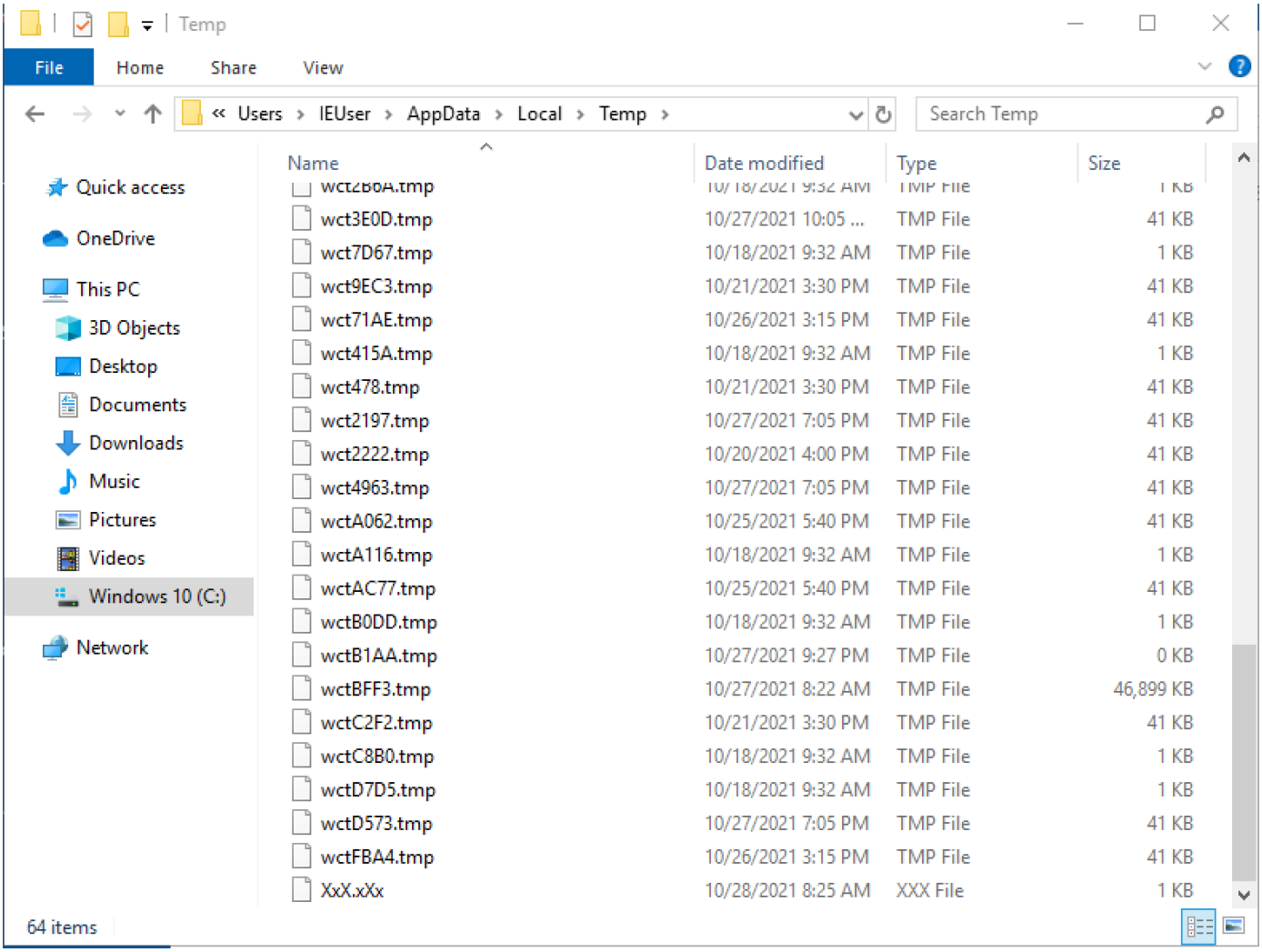
1. It sends a standard DNS request to www.itisreal.edu destination IP 192.168.1.1
2. Then it attempts to open a browser, in this case is "MSEDGEWIN10"
3. Then it sends another DNS request www.atwushere.net.

Summary: The malware attempts to communicate whether the infected computer is real and if it is real establishes a connection to www.itisreal.edu and sends the location information via www.atwushere.net. Once confirmed it's real, it download files and attempts to decrypt sensitive user data.

```
154 201.238953   10.0.2.15        20.189.173.20    TCP        54 49952 → 443 [ACK] Seq=214 Ack=2921 Win=64240 Len=0
155 201.239093   20.189.173.20    10.0.2.15        TCP      1514 443 → 49952 [ACK] Seq=2921 Ack=214 Win=65535 Len=1460 [TCP seg…
156 201.239094   20.189.173.20    10.0.2.15        TLSv1.2    76 Server Hello, Certificate, Server Key Exchange, Server Hello D…
157 201.239124   10.0.2.15        20.189.173.20    TCP        54 49952 → 443 [ACK] Seq=214 Ack=4403 Win=64240 Len=0
158 201.242274   10.0.2.15        20.189.173.20    TLSv1.2   212 Client Key Exchange, Change Cipher Spec, Encrypted Handshake M…
159 201.242499   20.189.173.20    10.0.2.15        TCP        60 443 → 49952 [ACK] Seq=4403 Ack=372 Win=65535 Len=0
160 201.487429   20.189.173.20    10.0.2.15        TLSv1.2   105 Change Cipher Spec, Encrypted Handshake Message
161 201.523967   10.0.2.15        20.189.173.20    TLSv1.2  1154 Application Data
162 201.524210   20.189.173.20    10.0.2.15        TCP        60 443 → 49952 [ACK] Seq=4454 Ack=1472 Win=65535 Len=0
163 201.524501   10.0.2.15        20.189.173.20    TCP      2974 49952 → 443 [ACK] Seq=1472 Ack=4454 Win=64189 Len=2920 [TCP se…
```

6. List the potential host-based indicators of this malware.

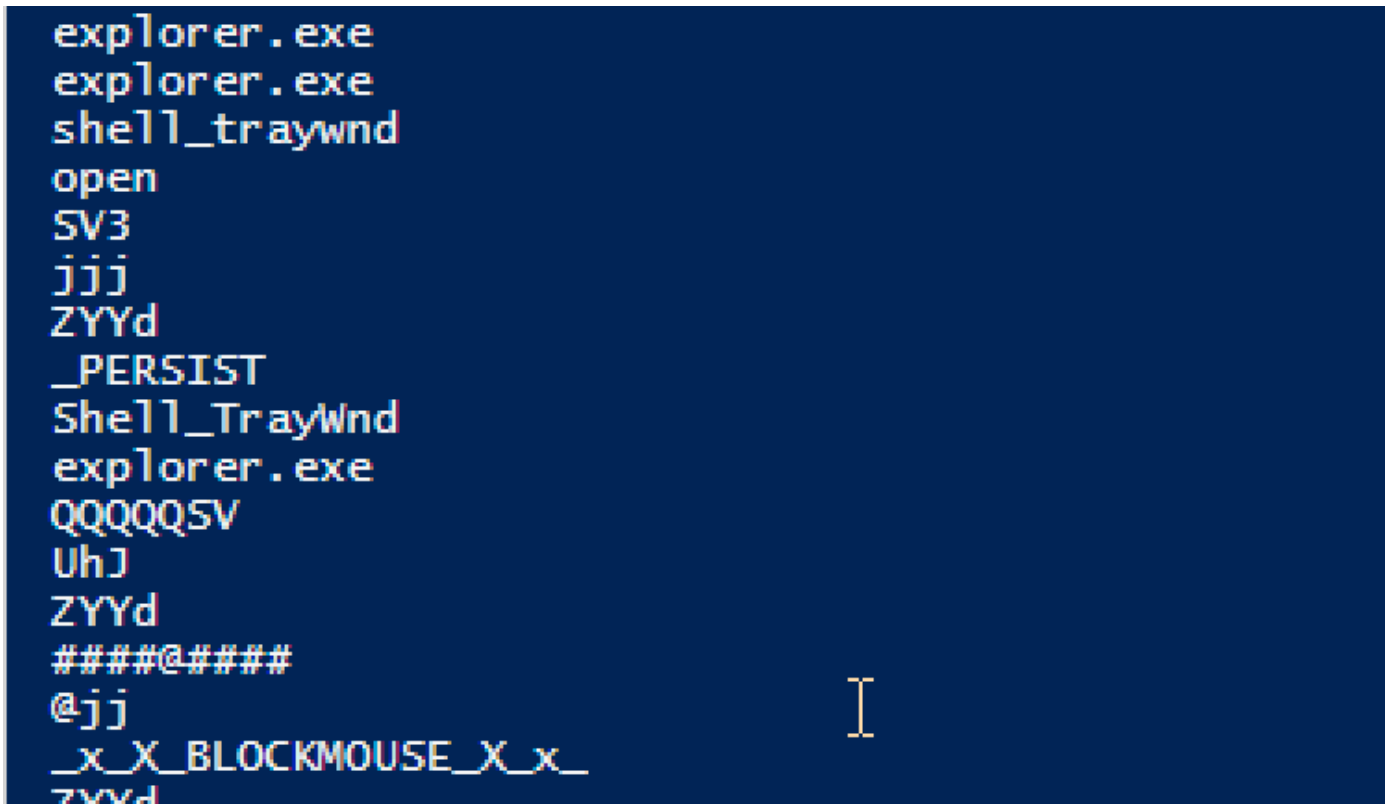
Can check if the file XxX.xXx is found in Users>IEUser>AppData>Local>Temp



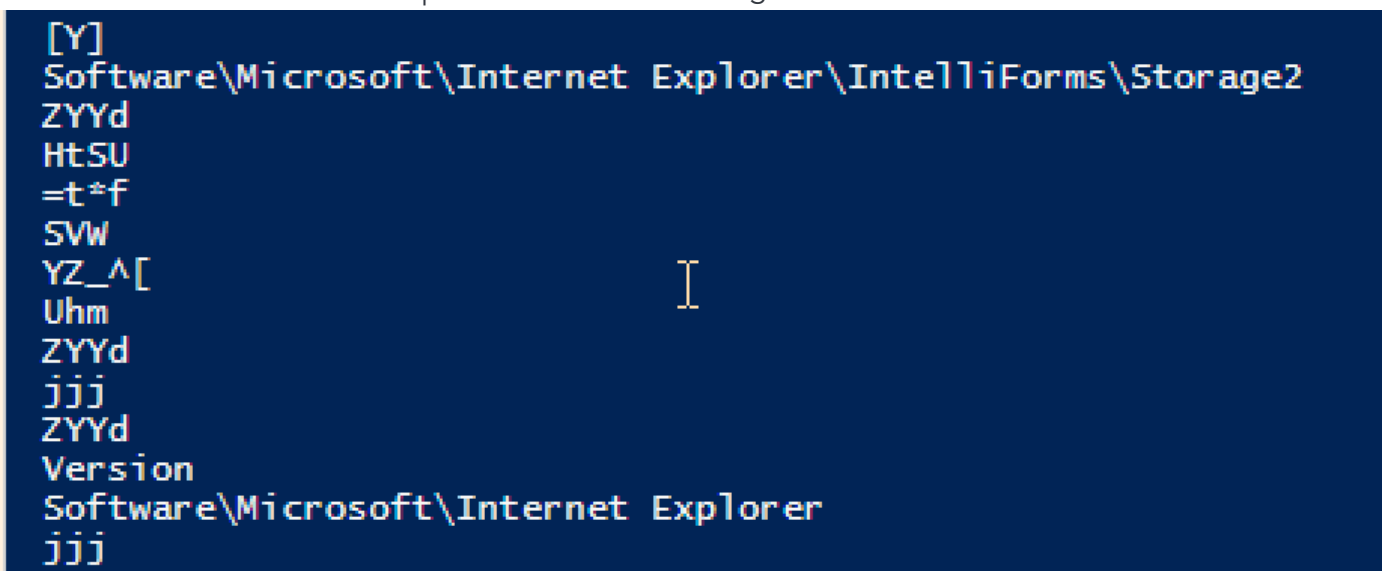
We can check if the infected computer reboots and if the mouse becomes blocked.

explorer.exe

x_X_BLOCKMOUSE_X_x



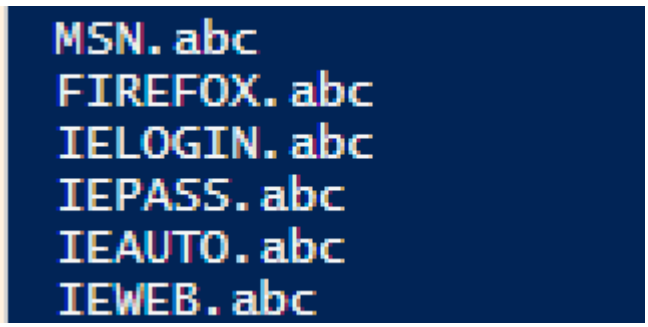
\Software\Microsoft\Internet Explorer\IntelliForms\Storage2



IELOGIN.abc

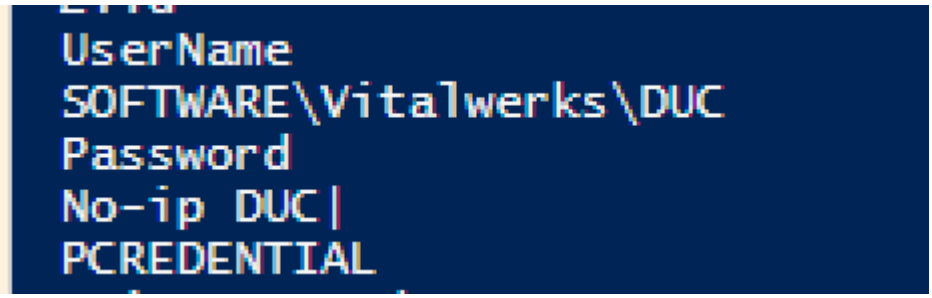
IEPASS.abc

IEAUTO.abc



We can also check if the following software is running on the infected comptuer.

SOFTWARE\Vitalwerks\DUC



7. List the potential network-based indicators of this malware.

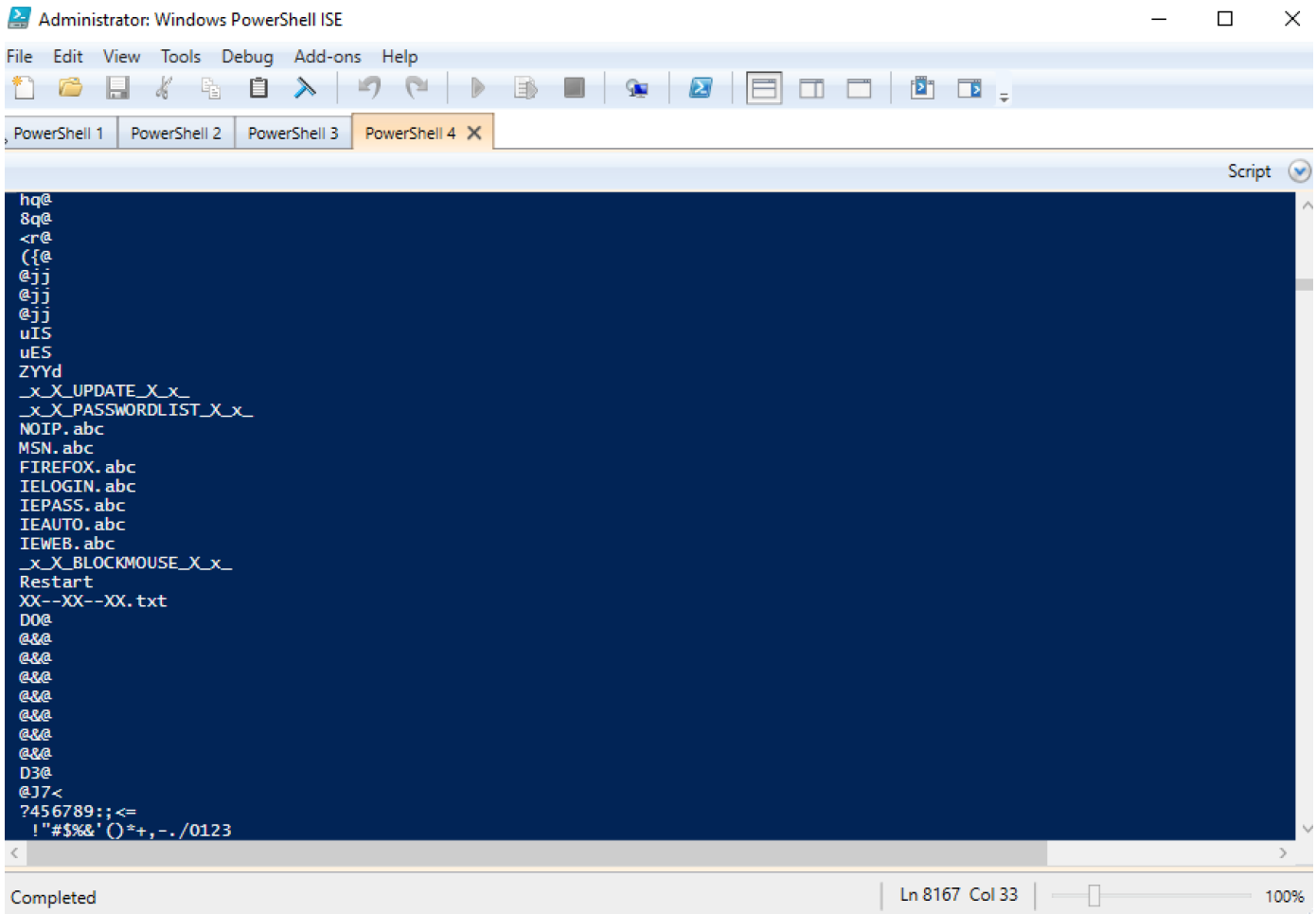
Can check if the infected computer has a TCP connection to 192.0.2.123 at port 443

Can check if the infected computer has TCP connection to 192.0.2.123 at port 80

Can check if the infected computer has a UDP connection to 127.0.0.1 at port 53

```
svchost.exe (1256) requested UDP 127.0.0.1:53
Received A request for domain 'www.itisreal.edu'.
HW-A-3.exe (1372) requested TCP 192.0.2.123:443
nc.exe (1848) requested TCP 10.0.2.15:49934
svchost.exe (1256) requested UDP 127.0.0.1:53
Received A request for domain 'www.atwushere.net'.
HW-A-3.exe (1372) requested TCP 192.0.2.123:80
```

FIREFOX.abc



\Mozilla\Firefox

```
mozcrt19.dll
sqlite3.dll
nspr4.dll
plc4.dll
plds4.dll
nssutil3.dll
softokn3.dll
nss3.dll
NSS_Init
NSSBase64_DecodeBuffer
PK11_GetInternalKeySlot
PK11_Authenticate
PK11SDR_Decrypt
NSS_Shutdown
PK11_FreeSlot
userenv.dll
GetUserProfileDirectoryA
\Mozilla\Firefox\
profiles.ini
Path
Profile0
\signons3.txt
\signons2.txt
\signons1.txt
\signons.txt
##$$
(unnamed value)
(unnamed password)
ZYYd
```

\Software\Microsoft\Internet Explorer

```
=t*t
SVW
YZ_^[
Uhm
ZYYd
jjj
ZYYd
Version
Software\Microsoft\Internet Explorer
jjj
Uhx
ZYYd
```

Microsoft\Network\Connections\pbk\rasphone.pbk

```
RasDialParams!
L$_RasDefaultCredentials#0
Microsoft\Network\Connections\pbk\rasphone.pbk
WVS
[^_
ZYYd
^[YY]
rasapi32.dll
```

\Internet Explorer\iexplore.exe

http\shell\open\command

```
http\shell\open\command
.exe
EXE
\Internet Explorer\iexplore.exe
QSVW
```

8. How many different domains do you think the malware can connect to? What are those domains?

2

```
svchost.exe (1256) requested UDP 127.0.0.1:53
Received A request for domain 'www.itisreal.edu'.
HW-A-3.exe (1372) requested TCP 192.0.2.123:443
nc.exe (1848) requested TCP 10.0.2.15:49934
svchost.exe (1256) requested UDP 127.0.0.1:53
Received A request for domain 'www.atwushere.net'.
HW-A-3.exe (1372) requested TCP 192.0.2.123:80
```

www.itisreal.edu

www.atwshere.net

PE File Format

1. Output the following to standard output:

a. Identify the file type as DLL or EXE or SYS regardless of the file's extension.

```
def get_file_type(pe):
    if pe.is_exe():
        return "EXE"
    if pe.is_dll():
        return "DLL"
    if pe.is_driver():
        return "SYS"
    return "UNKNOWN"
```

b. Total number of imported DLLs.

```
def count_dlls(pe):
    i = 0
    for entry in pe.DIRECTORY_ENTRY_IMPORT:
        i = i+1
    return i
```

c. Total number of imported functions.

```
def count_fns(pe):
    i = 0
    for entry in pe.DIRECTORY_ENTRY_IMPORT:
        for func in entry.imports:
            i = i+1

    return i
```

d. Output the compile time.

```
def get_compile_time(pe):
    return pe.FILE_HEADER.dump_dict()['TimeDateStamp']['Value'].split('[')[1][:-1]
```

2. Alert the user if the entry point of the code is not in a section with the name ".text", ".code", "CODE", or "INIT".

```
def get_valid_section_entry(pe):
    valid_sections = [".text",".code","CODE","INIT"]
    entry = ""
    valid = False
    for section in pe.sections:
        if section.Name.decode().rstrip('\x00') == valid_sections[0]:
            entry = entry + valid_sections[0] + " "
            valid = True
        if section.Name.decode().rstrip('\x00') == valid_sections[1]:
            entry = entry + valid_sections[1] + " "
            valid = True
        if section.Name.decode().rstrip('\x00') == valid_sections[2]:
            entry = entry + valid_sections[2] + " "
            valid = True
        if section.Name.decode().rstrip('\x00') == valid_sections[3]:
            entry = entry + valid_sections[3] + " "
            valid = True

    if valid == False:
        return "Entry point NOT VALID."
    return entry
```

3. Automatically use the PEiD database that comes with pefile to identify packers. Confirm that this works with UPX. Output the detection to standard output.

```
def signature_match(pe, PEiD_database):
    signatures = peutils.SignatureDatabase(data=PEiD_database)
    matches = signatures.match_all(pe, ep_only=True)
    result = ''

    if matches != None:
            for match in matches:
                m = ','.join(match)
                result  = result+m
            return result
    else:
        return None
```

4. Calculate and output the entropy for each section. Alert the user when you suspect that a section maybe packed or compressed.

```
packers_sections = {
        #The packer/protector/tools section names/keywords
        '.aspack': 'Aspack packer',
        '.adata': 'Aspack packer/Armadillo packer',
        'ASPack': 'Aspack packer',
        '.ASPack': 'ASPAck Protector',
        '.boom': 'The Boomerang List Builder (config+exe xored with a single byte key 0x77)',
        '.ccg': 'CCG Packer (Chinese Packer)',
        '.charmve': 'Added by the PIN tool',
        'BitArts': 'Crunch 2.0 Packer',
        'DAStub': 'DAStub Dragon Armor protector',
        '!EPack': 'Epack packer',
        'FSG!': 'FSG packer (not a section name, but a good identifier)',
        '.gentee': 'Gentee installer',
        'kkrunchy': 'kkrunchy Packer',
        '.mackt': 'ImpRec-created section',
        '.MaskPE': 'MaskPE Packer',
        'MEW': 'MEW packer',
        '.MPRESS1': 'Mpress Packer',
        '.MPRESS2': 'Mpress Packer',
        '.neolite': 'Neolite Packer',
        '.neolit': 'Neolite Packer',
        '.nsp1': 'NsPack packer',
        '.nsp0': 'NsPack packer',
        '.nsp2': 'NsPack packer',
        'nsp1': 'NsPack packer',
        'nsp0': 'NsPack packer',
        'nsp2': 'NsPack packer',
        '.packed': 'RLPack Packer (first section)',
        'pebundle': 'PEBundle Packer',
        'PEBundle': 'PEBundle Packer',
        'PEC2TO': 'PECompact packer',
        'PECompact2': 'PECompact packer (not a section name, but a good identifier)',
        'PEC2': 'PECompact packer',
        'pec1': 'PECompact packer',
        'pec2': 'PECompact packer',
        'PEC2MO': 'PECompact packer',
```

```
    'PELOCKnt': 'PELock Protector',
    '.perplex': 'Perplex PE-Protector',
    'PESHiELD': 'PEShield Packer',
    '.petite': 'Petite Packer',
    'petite': 'Petite Packer',
    '.pinclie': 'Added by the PIN tool',
    'ProCrypt': 'ProCrypt Packer',
    '.RLPack': 'RLPack Packer (second section)',
    '.rmnet': 'Ramnit virus marker',
    'RCryptor': 'RPCrypt Packer',
    '.RPCrypt': 'RPCrypt Packer',
    '.seau': 'SeauSFX Packer',
    '.sforce3': 'StarForce Protection',
    '.spack': 'Simple Pack (by bagie)',
    '.svkp': 'SVKP packer',
    'Themida': 'Themida Packer',
    '.Themida': 'Themida Packer',
    'Themida ': 'Themida Packer',
    '.taz': 'Some version os PESpin',
    '.tsuarch': 'TSULoader',
    '.tsustub': 'TSULoader',
    '.packed': 'Unknown Packer',
    'PEPACK!!': 'Pepack',
    '.Upack': 'Upack packer',
    '.ByDwing': 'Upack Packer',
    'UPX0': 'UPX packer',
    'UPX1': 'UPX packer',
    'UPX2': 'UPX packer',
    'UPX!': 'UPX packer',
    '.UPX0': 'UPX Packer',
    '.UPX1': 'UPX Packer',
    '.UPX2': 'UPX Packer',
    '.vmp0': 'VMProtect packer',
    '.vmp1': 'VMProtect packer',
    '.vmp2': 'VMProtect packer',
    'VProtect': 'Vprotect Packer',
    '.winapi': 'Added by API Override tool',
    'WinLicen': 'WinLicense (Themida) Protector',
    '_winzip_': 'WinZip Self-Extractor',
    '.WWPACK': 'WWPACK Packer',
    '.yP': 'Y0da Protector',
    '.y0da': 'Y0da Protector',
}

packers_sections_lower =  {x.lower(): x for x in packers_sections.keys()}
```

```
def get_entropy(section):
    section_name = section.Name.decode().rstrip('\x00')

    # Your code here
    entropy = section.get_entropy()

    suspect_packed = False
    print("\t ", section_name, ": ", entropy, end=" ")

    if section_name.lower() in packers_sections_lower.keys():
        suspect_packed == True
        print("Suspect packed")
    else:
        print()
    return
```

7. Alert the user when there is a zero sized section.

```
def is_zero_sized(section):
    data = section.get_data()
    try:
        # newbytes' count() takes a str in Python 2
        count = data.count("\0")
    except TypeError:
        # bytes' count() takes an int in Python 3
        count = data.count(0)

    if count == len(data):
        return True

    return False
```

8. Compare the PE Optional Header checksum with the actual checksum. Alert the user when

they don't match up.

```
def verify_checksum(pe):
    return pe.verify_checksum()
```

9. If there is a resource section, dump the first resource (of any type) to a file on disk.

```
def dump_rsrc(pe, dump_path):
    # If no resource section, return false;
    ret = {}
    if not hasattr(pe, 'DIRECTORY_ENTRY_RESOURCE'):
        return False
    # Otherwise, dump the first one to dump_path and return True.
    else:
        #For all possible resources
        for resource_type in pe.DIRECTORY_ENTRY_RESOURCE.entries:
            if resource_type.name is not None:
                name = "%s" % resource_type.name
            else:
                name = "%s" % pefile.RESOURCE_TYPE.get(resource_type.struct.Id)
            if name == None:
                name = "%d" % resource_type.struct.Id

            if hasattr(resource_type, 'directory'):

                #Try
                # resource_entry = resource_type.directory.entries[0]
                # resource_entry_directory = pe.parse_resource_entry(resource_entry.struct.OffsetToDirectory)
                # resource_entry_data = pe.parse_resource_data_entry(resource_entry.struct.OffsetToData)
                # print(resource_entry_data)
                # print(resource_entry_directory)

                ##Manual Loop through first resource
                resource_id = resource_type.directory.entries[0]
                if hasattr(resource_id, 'directory'):
                    for resource_lang in resource_id.directory.entries:
                        data = pe.get_data(resource_lang.data.struct.OffsetToData, resource_lang.data.struct.Size)
                        lang = pefile.LANG.get(resource_lang.data.lang, '*unknown*')
                        sublang = pefile.get_sublang_name_for_lang(resource_lang.data.lang, resource_lang.data.sublang )
                        ret[0] = (name, resource_lang.data.struct.OffsetToData, resource_lang.data.struct.Size, lang, sublang)

        #print(ret)
        f = open(dump_path, "a")
        f.write(str(ret))
        f.close()
        return True
```

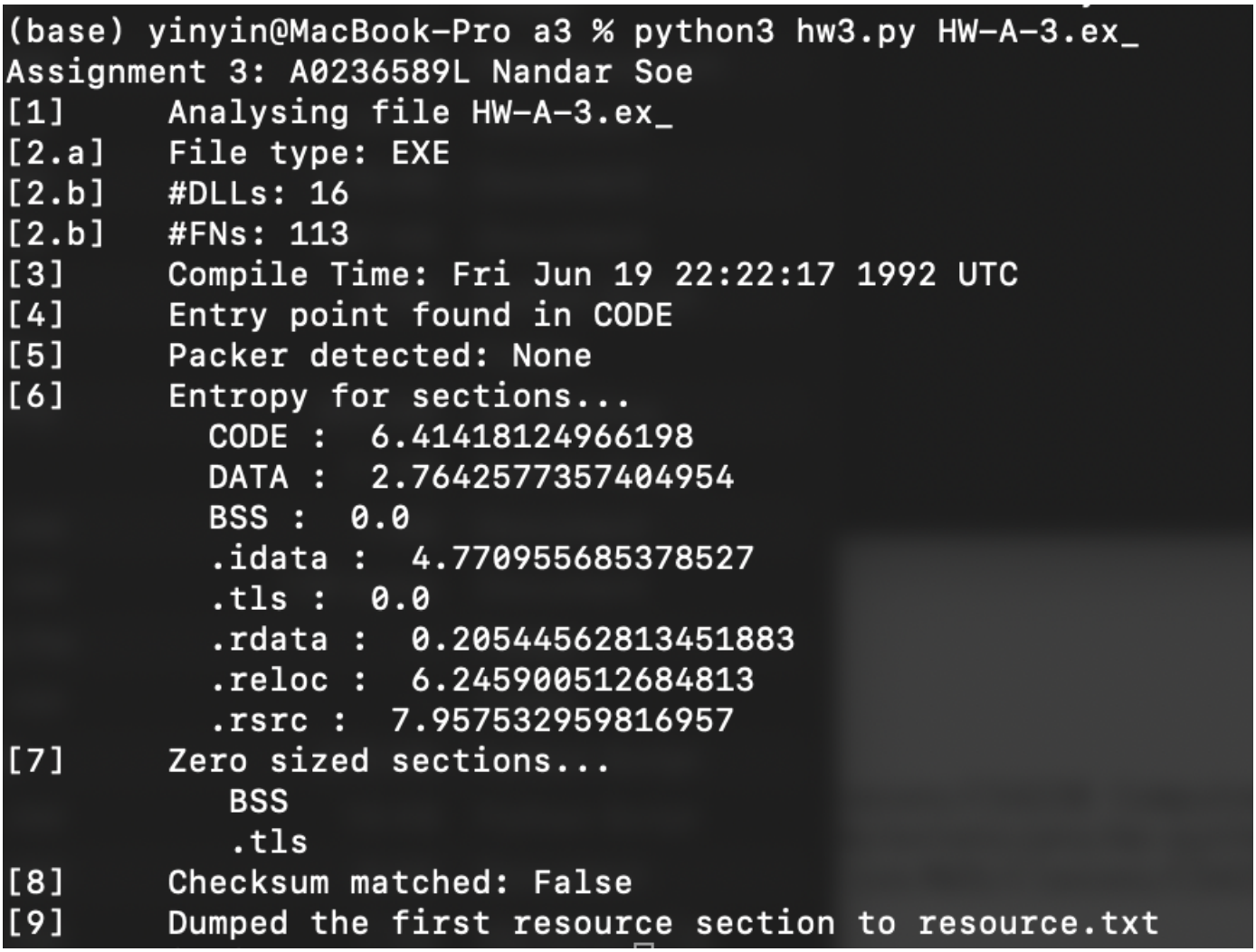
Results:

```
(base) yinyin@MacBook-Pro a3 % python3 hw3.py HW-A-1.ex_
Assignment 3: A0236589L Nandar Soe
[1]     Analysing file HW-A-1.ex_
[2.a]   File type: EXE
[2.b]   #DLLs: 3
[2.b]   #FNs: 8
[3]     Compile Time: Thu Nov 17 17:58:55 2011 UTC
[4]     Entry point found in Entry point NOT VALID.
[5]     Packer detected: UPX 2.90 [LZMA] -> Markus Oberhumer, Laszlo Molnar & John Reiser
[6]     Entropy for sections...
          UPX0 :  0.0 Suspect packed
          UPX1 :  7.8104290904126 Suspect packed
          UPX2 :  2.296234245718198 Suspect packed
[7]     Zero sized sections...
           UPX0
[8]     Checksum matched: False
[9]     No resource section has been found
```

```
(base) yinyin@MacBook-Pro a3 % python3 hw3.py HW-A-2.ex_
Assignment 3: A0236589L Nandar Soe
[1]     Analysing file HW-A-2.ex_
[2.a]   File type: EXE
[2.b]   #DLLs: 3
[2.b]   #FNs: 76
[3]     Compile Time: Tue Nov 22 03:35:10 2011 UTC
[4]     Entry point found in .text
[5]     Packer detected: None
[6]     Entropy for sections...
          .text :  6.443155266626159
          .rdata :  4.792233566888121
          .data :  2.4906229575121626
          .rsrc :  5.306569289651679
[7]     Zero sized sections...
          No zero sized section found.
[8]     Checksum matched: False
[9]     Dumped the first resource section to resource.txt
```

resource.txt

```
{0: ('RT_MANIFEST', 186664, 346, 'LANG_ENGLISH', 'SUBLANG_ENGLISH_US')}
```


(base) yinyin@MacBook-Pro a3 % python3 hw3.py HW-A-3.ex_
Assignment 3: A0236589L Nandar Soe
[1] Analysing file HW-A-3.ex_
[2.a] File type: EXE
[2.b] #DLLs: 16
[2.b] #FNs: 113
[3] Compile Time: Fri Jun 19 22:22:17 1992 UTC
[4] Entry point found in CODE
[5] Packer detected: None
[6] Entropy for sections...
CODE : 6.41418124966198
DATA : 2.7642577357404954
BSS : 0.0
.idata : 4.770955685378527
.tls : 0.0
.rdata : 0.20544562813451883
.reloc : 6.245900512684813
.rsrc : 7.957532959816957
[7] Zero sized sections...
BSS
.tls
[8] Checksum matched: False
[9] Dumped the first resource section to resource.txt

gd