

UNIVERSITY OF SCIENCE AND TECHNOLOGY OF HANOI

Information and Communication Technology Department



REPORT PROJECT

Malware Analysis

Assignment 1

Student name : Phạm Phú Hưng
Student ID : BA12-081

Table Of Contents

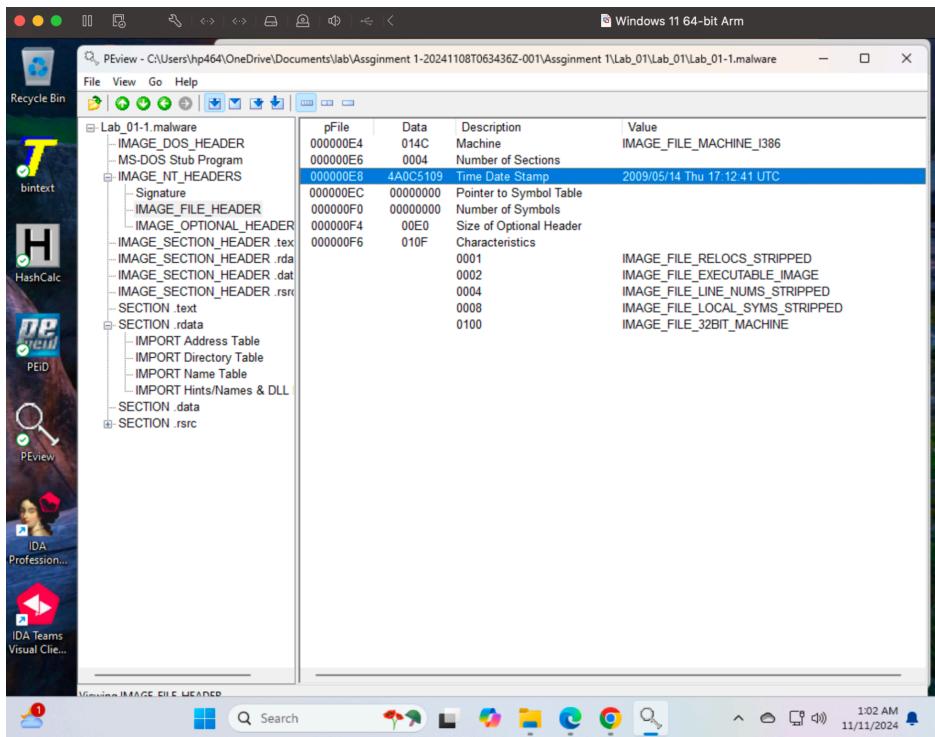
Malware Analysis	1
Assignment 1	1
I. LAB_01-01.malware	3
II. LAB_01-02.malware	10
IV. Lab_02-1.malware	18
V. Lab_02-2.malware	22

I. LAB_01-01.malware

- When was this file compiled?

Based on the information in PEview, the Time Date Stamp for Lab_01-1.malware is:

- 2009/05/14 Thu 17:12:41 UTC



- List a few imports or sets of imports and describe how the malware might use them

- Process and Thread Control (KERNEL32.dll)

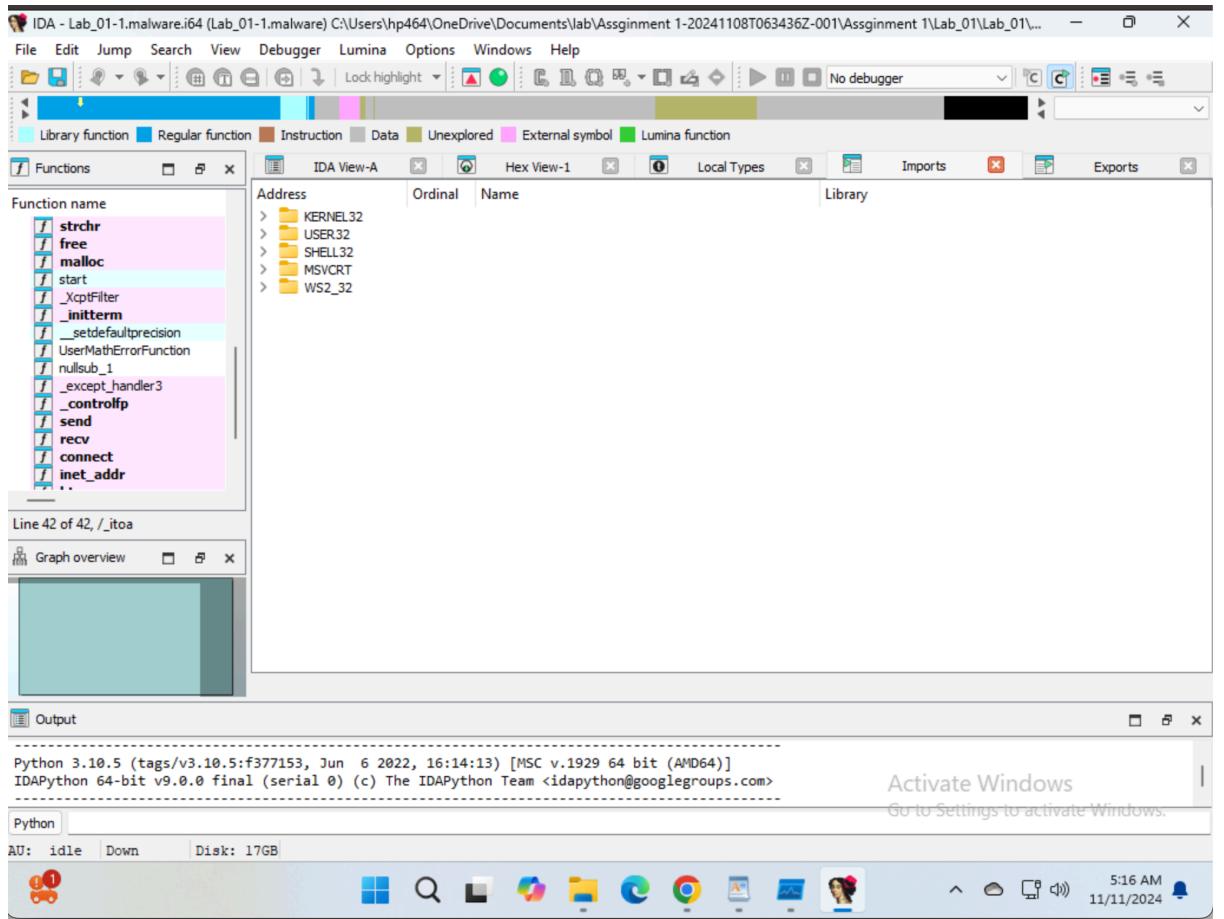
- CreateThread and CreateProcessA: These functions allow the malware to create new threads or processes, potentially to execute additional malicious code or manage multiple tasks simultaneously.
- TerminateProcess and ExitThread: These functions can be used to stop threads or processes, possibly to end operations or evade detection by terminating suspicious processes.

- Networking Functions (WS2_32.dll)

- connect, send, and recv: These functions are critical for establishing network connections and communication, likely to connect with a command-and-control (C2) server, send stolen data, or receive instructions.
- socket and WSASocket: These functions initialize network communication, setting up sockets and preparing the application for network operations.

c. Memory and String Management (MSVCRT.dll)

- malloc and free: For dynamic memory allocation and deallocation, allowing the malware to manage memory as needed for data storage.
- strchr, strnicmp, and sprintf: These string manipulation functions could be used to parse or format commands and data, useful in managing command strings or preparing data for network transmission.



3. What are a few strings that stick out to you and why?

- a. Network Connection: <http://www.ueopen.com/test.html>

Reason: This URL indicates a potential command-and-control (C2) endpoint that the malware might use to communicate with a remote server. The test.html page could serve as a way for the malware to check connectivity or retrieve commands. The presence of a URL in malware typically points to network-based functionality, such as receiving instructions from an attacker or exfiltrating data.

b. Command Execution: `cmd.exe` and `/c del`

Reason: The `cmd.exe` string, along with `/c del`, suggests that the malware might be executing system commands via the Windows Command Prompt. The `/c del` command specifically points to deletion actions, which the malware may use to delete temporary files or its own traces to avoid detection. This kind of command is often associated with malware that performs cleanup after executing its primary payload to avoid leaving evidence.

c. System Information Gathering: `GetComputerNameA`

Reason: This function call suggests that the malware might be gathering system information, specifically the computer name. Retrieving the computer name is a common reconnaissance technique in malware, used to identify the infected machine uniquely. The malware may send this information to its C2 server, allowing the attacker to track and differentiate infected systems.

File pos	Mem pos	ID	Text
A 00000000023E	00000000023E	0	0000000097D 00000040157D 0 L\$VHLD@
A 00000000026C	00000000026C	0	00000000995 000000401595 0 T\$V@0@
A 00000000029A	00000000029A	0	00000000B11 000000401711 0 D\$Qj
A 0000000002C6	0000000002C6	0	00000000C64 000000401864 0 D\$SV
A 0000000002F2	0000000002F2	0	00000000D05 000000401905 0 QRSSj
A 00000000031F	00000000031F	0	00000000DDC 00000040190C 0 D\$RP
A 00000000034B	00000000034B	0	00000000E17 000000401A17 0 T\$QR
A 000000000378	000000000378	0	00000000E09 000000401AD9 0 T\$4kP0@
A 000000000346	000000000346	0	00000000EF9 000000401AF5 0 L\$VQR
A 0000000003D3	0000000003D3	0	00000000F57 000000401B57 0 D\$SUWvh
A 000000000402	000000000402	0	00000000F76 000000401B76 0 T\$PQR
A 00000000042E	00000000042E	0	00000000F107 000000401C17 0 D\$x@
A 00000000045B	00000000045B	0	00000000F05 000000401C5C 0 T\$HjQR
A 00000000048B	00000000048B	0	00000000F182 000000401DB2 0 >uF
A 000000000484	000000000484	0	00000000F1F0 000000401DFD 0 XPVS
A 0000000004E0	0000000004E0	0	00000000F50 000000402110 0 ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789-/
A 000000000547	000000000547	0	00000000F6E2 0000004022E2 0 CloseHandle
A 000000000579	000000000579	0	00000000F6F0 0000004022F0 0 WaitForSingleObject
A 0000000005B3	0000000005B3	0	00000000F700 000000402306 0 CreateEventA
A 0000000005E6	0000000005E6	0	00000000F716 000000402316 0 ExitThread
A 000000000617	000000000617	0	00000000F724 000000402324 0 Sleep
A 000000000643	000000000643	0	00000000F72C 00000040232C 0 GetComputerNameA
A 00000000067A	00000000067A	0	00000000F740 000000402340 0 CreatePipe
A 00000000064B	00000000064B	0	00000000F74E 00000040234E 0 DisconnectNamedPipe
A 0000000006E5	0000000006E5	0	00000000F764 000000402364 0 TerminateProcess
A 00000000071C	00000000071C	0	00000000F770 000000402370 0 WaitForMultipleObjects
A 000000000759	000000000759	0	00000000F792 000000402392 0 TerminateThread
A 00000000078F	00000000078F	0	00000000F7A4 0000004023A4 0 CreateThread
A 0000000007C2	0000000007C2	0	00000000F7B4 000000402384 0 CreateProcessA
A 0000000007F7	0000000007F7	0	00000000F7C6 0000004023C6 0 DuplicateHandle
A 00000000082D	00000000082D	0	00000000F7D8 0000004023D8 0 GetCurrentProcess
A 000000000865	000000000865	0	00000000F7EC 0000004023EC 0 Readfile
A 000000000894	000000000894	0	00000000F7F8 0000004023F8 0 PeekNamedPipe
A 0000000008C8	0000000008C8	0	00000000F808 000000402408 0 SetEvent
A 0000000008F7	0000000008F7	0	00000000F814 000000402414 0 WriteFile
A 000000000927	000000000927	0	00000000F820 000000402420 0 SetProcessPriorityBoost

4. What happens when you run this malware? Is it what you expected and why?

When running this malware, it appears that it may attempt to:

Connects to 60.248.52.95, offers up a remote shell, then deletes itself

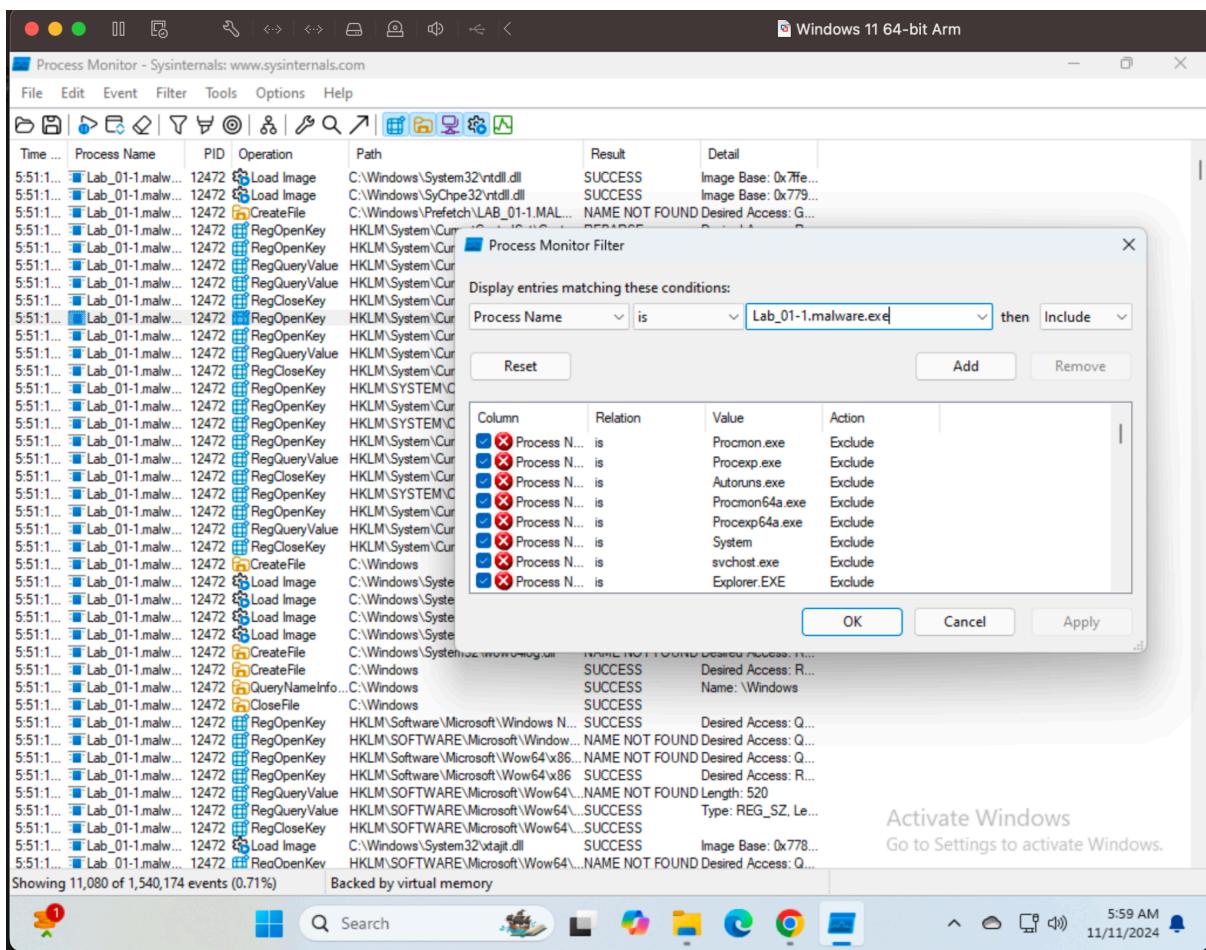
- Establish a Network Connection: The presence of the IP address `60.248.52.95:443` indicates that the malware might try to connect to this remote server over port 443 (typically used for HTTPS traffic), possibly to communicate with a command-and-control (C2) server for instructions or to exfiltrate data.
- Execute Commands: References to `cmd.exe` and the command `/c del` suggest that the malware might try to execute commands on the system. The use of `cmd.exe` could be for tasks like deleting files, modifying settings, or other command-line operations that help it maintain persistence or remove traces.
- Manipulate Threads and Processes: Imports like `CreateThread`, `CreateProcessA`, `ExitThread`, and `TerminateProcess` from KERNEL32 indicate that the malware may create or terminate threads and processes, possibly to launch new routines or manage its own components without crashing the main process.

5. Name a procmon filter and why you used it.

Filter Used: Process Name is "Lab_01-1.malware.exe"

Reason for Use: This filter was applied to isolate and monitor all activities initiated by the malware process Lab_01-1.malware.exe. By focusing exclusively on this specific process, we can examine the malware's behavior in a controlled manner, free from interference by other system processes. This approach is critical in malware analysis, as it allows us to clearly observe the malware's interactions with system resources, such as registry modifications, file manipulations, and network connections.

Using this filter helps reduce background noise, making it easier to identify potential malicious actions, persistence mechanisms, and system impact. This focused view is essential for accurately understanding the malware's intentions and possible effects on the host system.



6. Are there any host based signatures? (Files, registry keys, processes or services, etc). If so, what are they?

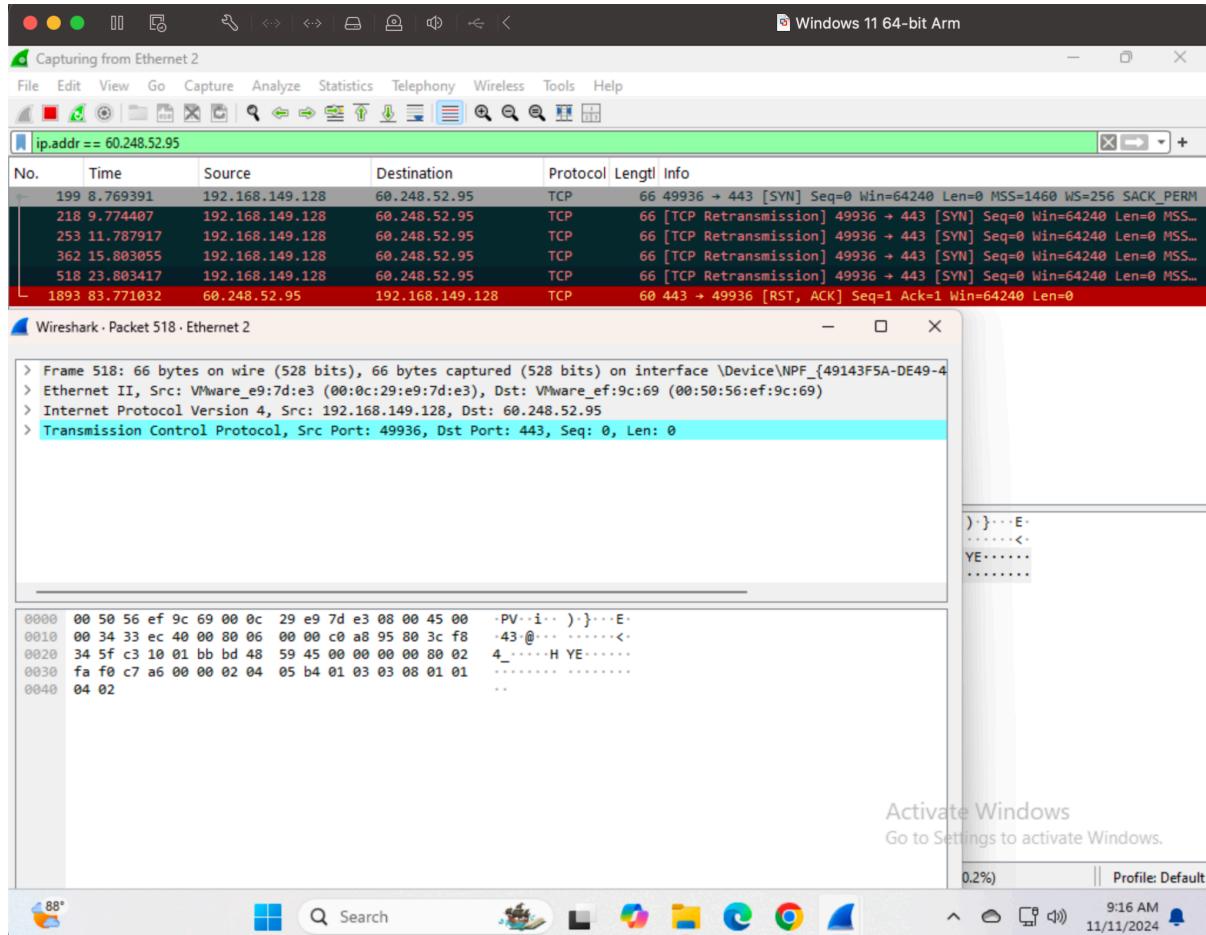
a. Files and Executables:

- Lab_01-1.malware.exe:
 - Path: C:\Users\hp464\OneDrive\Documents\lab\Assignment\Lab_01-1.malware.exe
 - Operation: Process Start with PID 2828.
 - This executable accesses several system files and resources, loading key Windows libraries.
 - Prefetch Files:

- Access to prefetch file C:\Windows\Prefetch\LAB_01-1.MALWARE.EXE-DB05513F.pf suggests recent execution or system pre-caching, which is common for identifying recent processes on a system.
- b. System DLLs Loaded:
 - DLLs commonly associated with Windows core functionality and compatibility were loaded:
 - C:\Windows\System32\ntdll.dll
 - C:\Windows\SyChpe32\ntdll.dll
 - C:\Windows\System32\wow64.dll, wow64base.dll, wow64win.dll, xtajit.dll
 - These indicate attempts to leverage system APIs and compatibility libraries, possibly to evade detection or for compatibility with different architectures.
- c. Registry Keys:
 - The process accessed several registry keys:
 - System and Session Configuration:
 - > HKLM\System\CurrentControlSet\Control\Nls\CodePage\ACP
 - > HKLM\System\CurrentControlSet\Control\Session Manager
 - Wow64 and Compatibility Layer:
 - > HKLM\Software\Microsoft\Wow64\x86
 - > HKLM\SOFTWARE\Microsoft\Wow64\x86\xtajit\Lab_01-1.malware.exe
 - > HKLM\Software\Microsoft\Windows NT\CurrentVersion
 - User-Specific Keys:
 - > HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer
 - > HKCU\Software\Microsoft\Windows NT\CurrentVersion
 - Accessing these keys suggests modifications to compatibility settings, session management, and user-specific configurations, which could be persistence mechanisms.
- d. Network and API Configuration:
 - The process accessed network-related registry keys, such as:
 - HKLM\System\CurrentControlSet\Control\NetworkProvider
 - These entries could indicate attempts to manipulate or observe network settings, relevant in cases where malware modifies network configurations.
 -

7. Are there any network based signatures? (URLs, packet contents. etc) If so, what are they?
 - a. URLs:
 - <http://www.ueopen.com/test.html>
 - This URL likely serves as a command-and-control (C2) endpoint or may be used to download additional instructions or exfiltrate data.
 - b. IP Address and Port:
 - 60.248.52.95:443
 - The IP address with port 443 suggests a potential HTTPS connection. This IP could be a C2 server that the malware attempts to connect to for secure communication, making it harder to intercept data without SSL decryption.
 - c. Packet Contents (Inferred from Strings):
 - Command Execution:
 - cmd.exe: This indicates that the malware might execute system commands and potentially send the output back to the C2 server.
 - Data Transmission:
 - send = %d: This suggests that the malware may transmit specific data or status information to the server. It might be reporting execution results or status updates.
 - d. Observations from Wireshark (Connection Attempts):

In the Wireshark screenshot, repeated SYN packets indicate that the malware is trying to establish a TCP connection with 60.248.52.95 on port 443, which is typical for HTTPS communication. However, the RST packets suggest that the connection was unsuccessful, possibly due to the server being unreachable or a network defense mechanism.



- Is there anything that impeded your analysis? How so? How might you overcome this?

The file's self deletion was a nuisance. This can be overcome by keeping a separate copy, or by NOP'ing the delete call

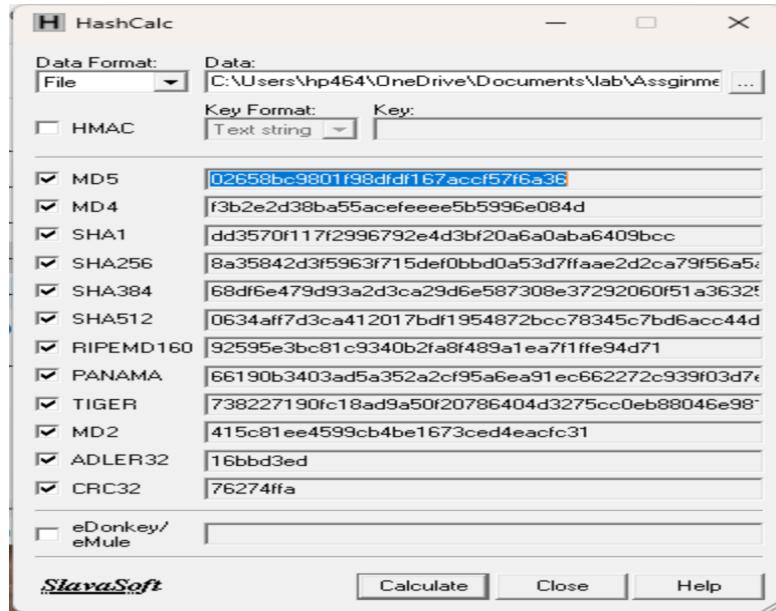
- What do you think is the purpose of this malware?

The purpose of this malware appears to be to act as a backdoor, allowing the attacker remote access to the compromised system. By running from unusual directories like OneDrive and SyChpe32, it tries to avoid detection while maintaining persistence. The presence of suspicious registry activity and file operations in Prefetch suggests it might be designed to re-execute after system restarts. Ultimately, it likely provides the attacker with the ability to control the system remotely without the user's knowledge.

II. LAB_01-02.malware

1. What is the md5sum? What of interest does VirusTotal Report?

The md5sum for the file shown in the screenshots is [02658bc9801f98dfd167accf576a36](#).

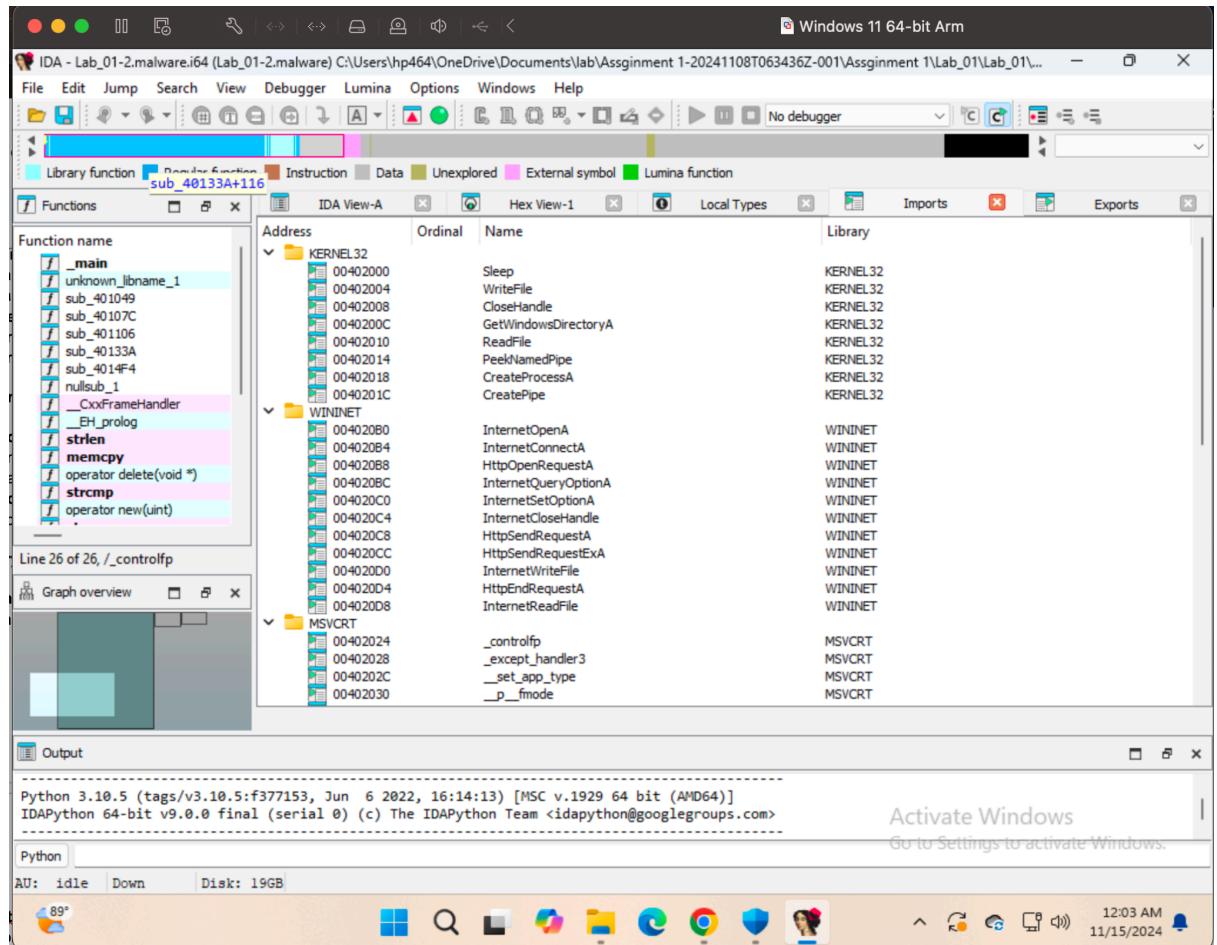


The VirusTotal report indicates the following details of interest:

- a. SHA-1: [dd3570f117f2996792e4d3bf20a6a0aba6409bcc](#)
- b. SHA-256: [8a35842d3f5963f715def0bbd0a53d7ffaae2d2ca79f56a5ac8bede64749d279](#)
- c. File type: PE32 executable (GUI) for Microsoft Windows.
- d. File size: 8.50 KB (8704 bytes).
- e. Compiler: Microsoft Visual C++.
- f. Detection: The file has been marked as suspicious/malicious by various antivirus engines on VirusTotal.
- g. History: The file's creation time is recorded as 2008-09-16 08:40:04 UTC, with its first appearance in the wild on 2015-09-03.

This information suggests that the file is a potentially malicious Windows executable, likely packed or obfuscated, given the use of PEID packer and its small size.

2. List a few imports or sets of imports and describe how the malware might use them.
 - a. File and Process Management (KERNEL32.dll)
 - CreateFileA and WriteFile: These functions allow the malware to create and write to files, which could be used to store stolen data or configuration information.
 - CreateProcessA and CreateThread: These functions enable the malware to launch new processes or threads, potentially to execute additional payloads or manage various malicious activities concurrently.
 - CloseHandle and TerminateProcess: These functions can be used to close handles and terminate processes, allowing the malware to stop its own processes if it detects analysis or to shut down other security-related processes for stealth.
 - b. Internet Communication (WININET.dll)
 - InternetOpenA and InternetConnectA: These functions open internet connections, potentially to establish communication with a remote command-and-control (C2) server.
 - HttpOpenRequestA and HttpSendRequestA: These functions allow the malware to send HTTP requests, which could be used to exfiltrate data, receive commands, or download additional payloads.
 - InternetCloseHandle: This function closes the internet connection, helping the malware to clean up network connections after communication to avoid detection.
 - c. Memory and String Manipulation (MSVCRT.dll)
 - malloc and free: These functions handle dynamic memory allocation and deallocation, allowing the malware to allocate memory for storing data as needed during execution.
 - strcpy and strcat: These functions enable the malware to copy and concatenate strings, useful for constructing file paths, commands, or URLs dynamically.
 - memset and memcmp: These functions are used to initialize or compare memory, which can aid in data management and control flow within the malware's operations.



3. What are a few strings that stick out to you and why?

- 69.25.50.10

This is an IP address embedded in the binary, which may be the address of a command-and-control (C2) server. Malware often uses hardcoded IP addresses to establish communication with remote servers for further instructions or data exfiltration.

- wuauctl.exe

This is the name of a legitimate Windows Update process. Malware sometimes references or masquerades as legitimate processes to evade detection, potentially attempting to blend in with normal system activity or to run unnoticed.

- D-o-w-n-l-o-a-d-f-i-l-e and U-p-l-o-a-d-f-i-l-e

These strings indicate functionality for downloading and uploading files, which suggests that the malware has capabilities to retrieve additional malicious payloads or exfiltrate data from the infected machine. This is typical of malware with remote access or data-stealing purposes.

4. What happens when you run this malware? Is it what you expected and why?

Nothing appears on screen. In the background it is attempting to connect to 69.25.50.10, but fails. If it succeeds it offers a remote shell.

5. Name a procmon filter and why you used it.

Filter Used: Process Name is "Lab_01-2.malware.exe"

Reason for Use: This filter helps isolate and focus on the activities specifically performed by the malware file ([Lab_01-2.malware.exe](#)). By filtering out other processes, you can more easily analyze the malware's behavior, including registry changes, file operations, network connections, and other interactions with the system. This targeted filtering reduces noise in the logs, making it easier to identify malicious or suspicious actions initiated by the malware itself.

The screenshot shows the Process Monitor application window from Sysinternals. The main pane displays a list of events with columns for Time, Process Name, PID, Operation, Path, Result, and Detail. A significant number of events are listed, mostly for the process 'Lab_01-2.malware.exe' with PID 1424. The 'Operation' column shows various system calls like Process Start, Thread Create, Load Image, RegOpenKey, and CreateFile. The 'Result' column indicates most events were successful. The 'Detail' column provides specific details for each event, such as Parent PID and Image Base.

A modal dialog titled 'Display entries matching these conditions:' is open in the center. It contains a search bar with the text 'Architecture is Lab_01-2.malware.exe then Include'. Below the search bar is a table with four columns: Column, Relation, Value, and Action. The table lists several conditions with their corresponding actions: 'Process N...' is 'Lab_01-2.malware.exe' (Include), 'Process N...' is 'Procmon.exe' (Exclude), 'Process N...' is 'Proexp.exe' (Exclude), 'Process N...' is 'Autoruns.exe' (Exclude), 'Process N...' is 'Procmon64a.exe' (Exclude), 'Process N...' is 'Proexp64a.exe' (Exclude), 'Process N...' is 'System' (Exclude), and 'Operation begins with IRP_MJ_ (Exclude). At the bottom of the dialog are OK, Cancel, and Apply buttons.

6. Are there any hostbased signatures? (Files, registry keys, processes or services, etc). If so, what are they?

Files:

- [C:\Windows\Prefetch\LAB_01-2.MALWARE.EXE-0835F...](#): The presence of this prefetch file name indicates that the executable [Lab_01-2.malware.exe](#) has been run on the system. Monitoring the [Prefetch](#) folder for unusual or suspicious executables can help detect the malware.
- [C:\Windows\System32\wow64log.dll](#): The malware attempts to access this file, which is missing on the system ([NAME NOT FOUND](#)). However, repeated attempts to access non-existent system DLLs may indicate malicious behavior.

Registry Keys:

- **HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options**: The malware accesses this registry path, which is commonly used to set debugger paths or manipulate execution options. Monitoring this path can reveal potential interference with security or system settings.
- **HKLM\System\CurrentControlSet\Control\Session Manager** and **HKLM\System\CurrentControlSet\Control\Nls\CodePage**: The malware repeatedly queries and sets values in these registry paths, which could indicate attempts to modify session management or code page settings, potentially to manipulate environment settings.

Processes:

- **Lab_01-2.malware.exe**: This process name is specific to this malware sample. Any process with this name would be highly indicative of malicious activity.

Time	Process	Action	Path	Result	Image Base
11:16...	Lab_01-2.malw...	1424 Load Image	C:\Windows\System32\wow64win.dll	SUCCESS	0x7ffe...
11:16...	Lab_01-2.malw...	1424 Load Image	C:\Windows\System32\wow64con.dll	SUCCESS	0x7ffe...
11:16...	Lab_01-2.malw...	1424 CreateFile	C:\Windows\System32\wow64log.dll	NAME NOT FOUND	Desired Access: R...
11:16...	Lab_01-2.malw...	1424 CreateFile	C:\Windows\System32\kernel32.dll	NAME NOT FOUND	Desired Access: R...
11:16...	Lab_01-2.malw...	1424 QueryNameInfo...	C:\Windows	SUCCESS	Name: \Windows
11:16...	Lab_01-2.malw...	1424 CloseFile		SUCCESS	
11:16...	Lab_01-2.malw...	1424 RegOpenKey	HKEYLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options	SUCCESS	Desired Access: Q...
11:16...	Lab_01-2.malw...	1424 RegOpenKey	HKEYLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\Lab...	NAME NOT FOUND	Desired Access: Q...
11:16...	Lab_01-2.malw...	1424 RegOpenKey	HKEYLM\Software\Microsoft\Wow64\vtB5\vtajit	NAME NOT FOUND	Desired Access: Q...
11:16...	Lab_01-2.malw...	1424 RegOpenKey	HKEYLM\Software\Microsoft\Wow64\vtB5\vtajit\Lab_01-2.malware.exe	SUCCESS	Desired Access: R...
11:16...	Lab_01-2.malw...	1424 RegQueryValue	HKEYLM\Software\Microsoft\Wow64\vtB5\vtajit\Lab_01-2.malware.exe	SUCCESS	Desired Access: R...
11:16...	Lab_01-2.malw...	1424 RegQueryValue	HKEYLM\Software\Microsoft\Wow64\vtB5\vtajit\Lab_01-2.malware.exe\Default	(NAME NOT FOUND)	Length: 520 Windows.
Showing 14,080 of 3,123,191 events (0.45%) Backed by virtual memory					

7. Are there any network based signatures? (URLs, packet contents. etc) If so, what are they?

IP Address and Port:

- Remote IP Address: **69.25.50.10**
- Port: **https** (likely using port 443 for encrypted communication) The malware attempts multiple TCP connections to the IP address **69.25.50.10** over HTTPS. This IP address could be a command-and-control (C2) server where the malware connects to receive instructions or exfiltrate data.

Connection Patterns:

- Frequent Reconnects: There are multiple instances of **TCP Reconnect** and **TCP Disconnect** operations to the same IP address, indicating persistent attempts to establish or maintain a connection. This suggests that the malware may be designed to maintain a steady communication link with the C2 server, even if interrupted.

Network Operation Details:

- Packet Contents: The logs show **Length: 0** for the packets, suggesting that these might be keep-alive signals or that the actual data is encrypted, making it challenging to interpret packet contents directly from the log.

8. Is there anything that impeded your analysis? How so? How might you overcome this?

No, though more information could have been made available if 69.25.50.10 was up

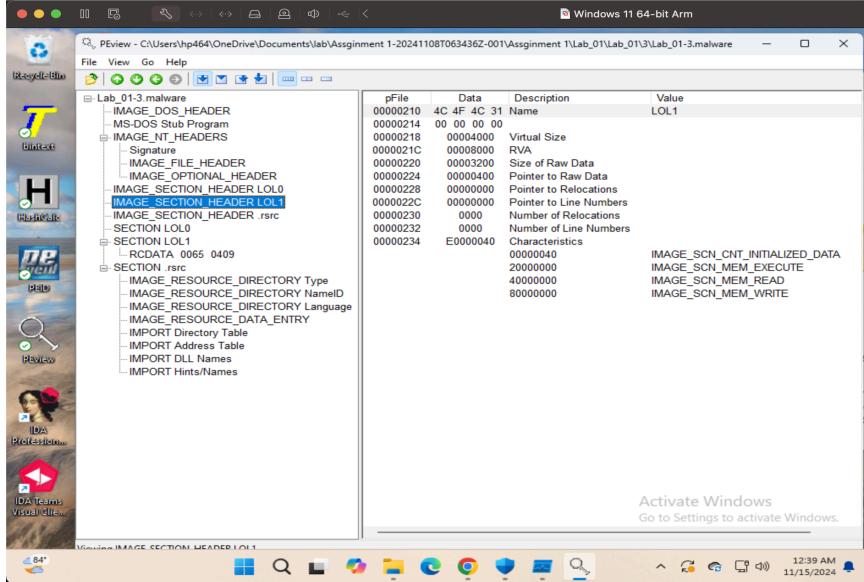
9. What do you think is the purpose of this malware?

The malware acts as a backdoor, enabling remote access to the infected system. Its purpose is to allow the attacker to execute commands, manipulate files, and maintain persistence while remaining hidden, likely for data theft, espionage, or further network compromise.

III. Lab_01-3.malware

- Are there any indications that this malware is packed? What are they? What is it packed with?

There are very few visible strings and imports, and the VirtualSize is significantly larger than the Size of Raw Data, suggesting it may be packed with UPX.



- Are you able to unpack it? Why or why not?

```
C:\Users\hp464>cd C:\Users\hp464\OneDrive\Documents\upx-4.2.4-win64
C:\Users\hp464\OneDrive\Documents\upx-4.2.4-win64>upx.exe -d "C:\Users\hp464\OneDrive\Documents\lab\Assignment 1-20241108T063436Z-001\Assignment 1\Lab_01\Lab_01-3-v2"
    Ultimate Packer for eXecutables
    Copyright (C) 1996 - 2024
UPX 4.2.4      Markus Oberhumer, Laszlo Molnar & John Reiser      May 9th 2024
File size      Ratio      Format      Name
-----
upx: C:\Users\hp464\OneDrive\Documents\lab\Assignment 1-20241108T063436Z-001\Assignment 1\Lab_01\Lab_01-3-v2: CantUnpackException: file is modified/hacked/protected; take care!!!
Unpacked 0 files.
C:\Users\hp464\OneDrive\Documents\upx-4.2.4-win64>
```

- What are a few strings that stick out to you and why?

a. "InternetOpenA"

- Relevance: This API indicates that the program interacts with the internet. It is commonly used in malware for initiating HTTP/HTTPS connections to communicate with command-and-control (C2) servers.

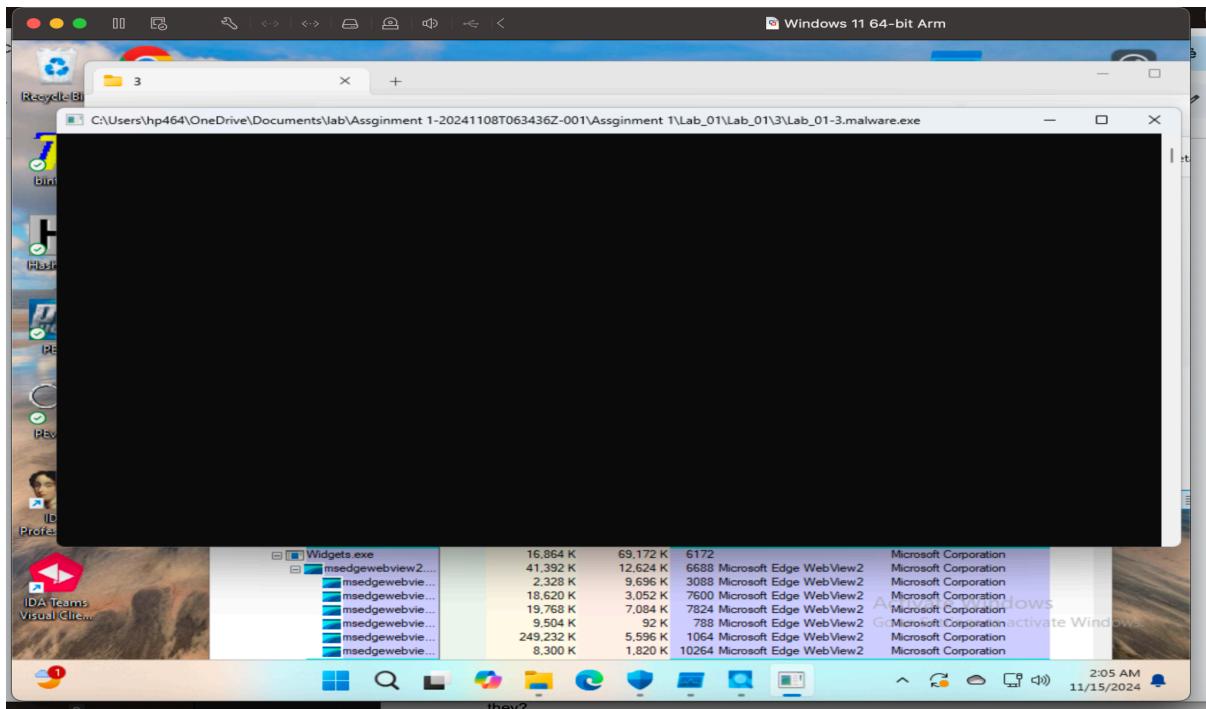
b. "<http://%s>"

- Relevance: This string is a placeholder for dynamically constructed URLs. It suggests that the program may use runtime-resolved URLs for network operations, potentially for downloading files, exfiltrating data, or contacting a C2 server.

c. "WININET.dll"

- Relevance: This library is used for internet-related functions, including HTTP requests and file transfers. Its presence confirms potential web interaction, often associated with malware trying to connect to external servers.

4. What happens when you run this malware? Is it what you expected and why?



5. Are there any hostbased signatures? (Files, registry keys, processes or services, etc). If so, what are they?

No

6. Are there any network based signatures? (URLs, packet contents, etc) If so, what are they?

The network-based signatures of this malware include the URL it communicates with, which can be used to identify its Command-and-Control (C&C) server. Additionally, the malware's user agent string provides another signature that can be leveraged to detect its activity on the network. These indicators can be used in intrusion detection systems (IDS) or firewalls to identify and block malicious traffic.

7. Is there anything that impeded your analysis? How so? How might you overcome this?
Cannot be decompressed

8. What do you think is the purpose of this malware?

The purpose of this malware cannot be definitively determined without conducting a more detailed analysis. Based on its behavior of reporting the hostname to the attacker, it appears to be gathering initial reconnaissance data. This could be used to identify high-value targets within a network, determine the system's role or importance, or assist the attacker in deciding on the next

steps. Additionally, this functionality might serve as part of a larger modular malware framework, where further payloads are delivered depending on the system identified. Without further investigation, the malware's broader objectives remain unclear.

IV. Lab_02-1.malware

1. Main Function

a. What is the address of `main`?

- The address of the `main` function is `0x004011A0`, as seen in the disassembly.

```
IDA - Lab_02-1.malware C:\Users\hp464\OneDrive\Documents\lab\Assignment 1-20241108T063436Z-001\Assignment 1\Lab_02\Lab_02\01\Lab_02-1.malware
File Edit Jump Search View Debugger Lumina Options Windows Help
Library function Regular function Instruction Data Unexplored External symbol Lumina function
Functions IDA View-A Hex View-1 Local Types Imports Exports
Function name
sub_401000
sub_401130
_main
pre_c_int
pre_cpp_init
__tmainCRTStartup
start
__CxuUnhandledExceptionFilter
sub_40147D
XcpFilter
amsq_exit
UserMathErrorFunction
FindESection
IsNonwritableInCurrentImage
ValidateImageBase
security_init_cookie
__atexitntrnt
__onexit
__atexit
sub_401777
sub_401797
_setdefaultprecision
__initterm_e
__ninitterm
__SEH_prolog4
__SEH_epilog4
Line 3 of 41, /_main
Output
Command "MakeStrlit" failed
Command "MakeStrlit" failed
Activate Windows
Go to Settings to activate Windows.
AU: idle Down Disk: 19GB
Python
2:20 AM 11/15/2024
.text:004011A0 ; ===== S U B R O U T I N E =====
.text:004011A0 ; Attributes: noreturn
.text:004011A0 .text:004011A0 _main proc near ; CODE XREF: __tmainCRTStartup+F8+P
.text:004011A0 .text:004011A0 argc = dword ptr 4
.text:004011A0 argv = dword ptr 8
.text:004011A0 envp = dword ptr 0Ch
.text:004011A0
.push 0 ; dwReserved
.push 1 ; dwFlags
.push offset szUrl ; "http://reversing.rocks/"
call ds:InternetCheckConnectionA
test eax, eax
jz short loc_4011C0
call sub_401130
push 0 ; Code
call ds:exit
.text:004011A0 ;
.text:004011C0 loc_4011C0: push 1 ; CODE XREF: _main+11+j
.text:004011C0 call ds:exit
.text:004011C2 _main endp
.text:004011C2
000005A0 004011A0: _main (Synchronized with Hex View-1)
```

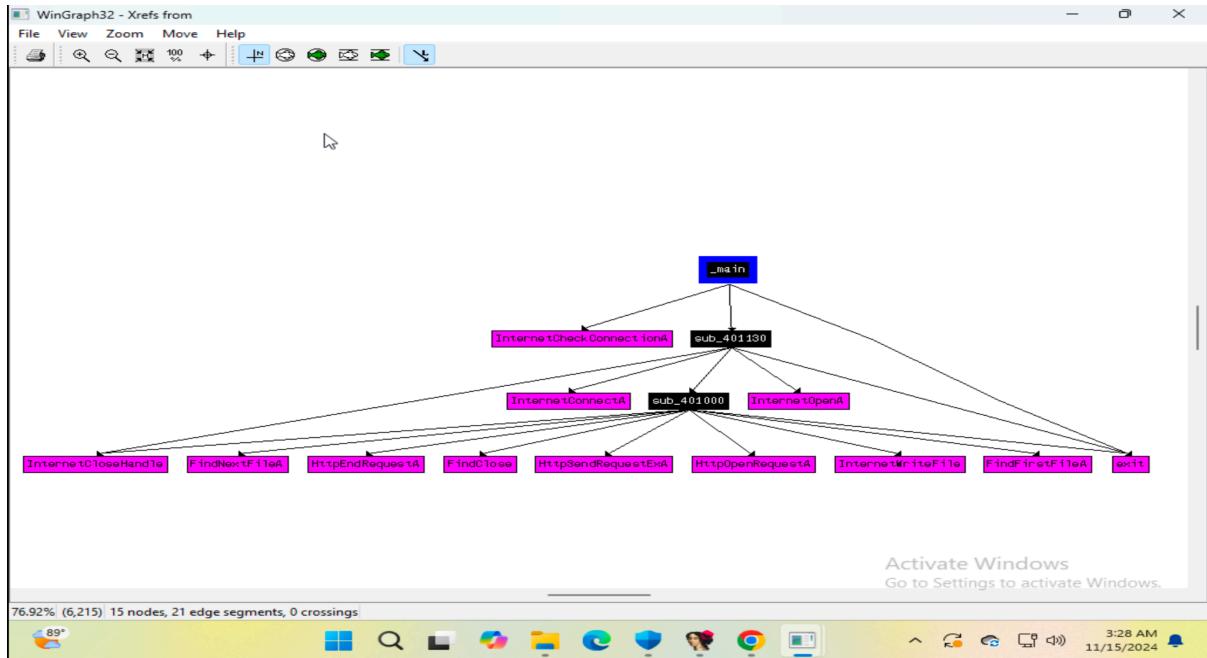
b. What does this function do?

- Logic and Purpose:
 - The `main` function verifies internet connectivity to a specific URL (<http://reversing.rocks/>) using the Windows API function `InternetCheckConnectionA`.
 - If the internet connection test fails, the program exits with a status code of 1.
 - If successful, the function transitions into `sub_401130`, which likely contains the malware's core functionality.

i. What code constructs are used in this function?

- API Call:
 - The function makes a call to the Windows API `InternetCheckConnectionA`, which is commonly used to test connectivity to a specific URL.
- Control Flow:
 - Conditional branching is performed using `test eax, eax` and `jz` (jump if zero) to handle the results of the API call:
 - If the test fails, it jumps to `loc_4011C0`, where it calls `exit(1)`.
 - Otherwise, it proceeds to `sub_401130`.
- Exit Calls:

- The function explicitly terminates the program using the `exit` function, with status codes 0 or 1 depending on the outcome of the connection test.



ii. Are there any interesting strings? If so, what are they?

- `http://reversing.rocks/`:**
 - This hardcoded string is used as the target URL for the connectivity test. It is suspicious and serves as a key Indicator of Compromise (IOC).
 - Its presence suggests potential malicious activity, such as:
 - Verification of network conditions before proceeding with a payload.
 - Interaction with a Command and Control (C2) server.

The screenshot shows the IDA Pro interface with the assembly view open. The main window displays the assembly code for the `_main` function. The code includes several calls to Windows API functions, notably `InternetConnectA` and `InternetCloseHandle`. The assembly code is color-coded according to the IDA Pro conventions:

- Text (green):** `sub_401000`, `sub_401130`, `_main`, `_pre_crt_init`, `_pre_cpp_init`, `_tmainCRTStartup`, `start`, `_CxxUnhandledExceptionFilter`, `sub_40147D`, `_XopFilter`, `_amsg_exit`, `UserMathErrorFunction`, `_FindPESection`, `_IsNonwritableInCurrentImage`, `_ValidateImageBase`, `__security_init_cookie`, `_atexitbnt`, `_onexit`, `_atexit`, `sub_401777`, `sub_401797`, `_setdefaultprecision`, `_initterm_e`, `_initterm`, `_SEH_prolog4`, `_SEH_epilog4`.
- Instruction (blue):** `int _cdecl main(int argc, const char **argv, const char **envp)`.
- Data (grey):** `proc near`, `CODE XREF: __tmainCRTStartup+F84p`.
- Unexplored (yellow):** `push 0`, `push 1`, `push offset szUrl`, `call ds:InternetCheckConnectionA`, `test eax, eax`, `jz short loc_4011C0`, `push 0`, `call ds:exit`.
- External symbol (pink):** `push 0`, `push 1`, `push offset szUrl`, `call ds:InternetCheckConnectionA`, `test eax, eax`, `jz short loc_4011C0`, `push 0`, `call ds:exit`.
- Lumina function (purple):** `push 0`, `push 1`, `push offset szUrl`, `call ds:InternetCheckConnectionA`, `test eax, eax`, `jz short loc_4011C0`, `push 0`, `call ds:exit`.

2. Looking at the subroutine a 0x00401130:

a. Arguments to `InternetConnectA` and Their Meaning:

- `hInternet` (EDI): Handle to the WinINet session (from `InternetOpen`).
- `IpszServerName` (`reversing.rocks`): Server name to connect to.
- `nServerPort` (66): Server port (custom port 66).
- `IpszUserName` (0): No username for authentication.
- `IpszPassword` (0): No password for authentication.
- `dwService` (3): Service type (`INTERNET_SERVICE_HTTP`).
- `dwFlags` and `dwContext` (0): No flags or context.

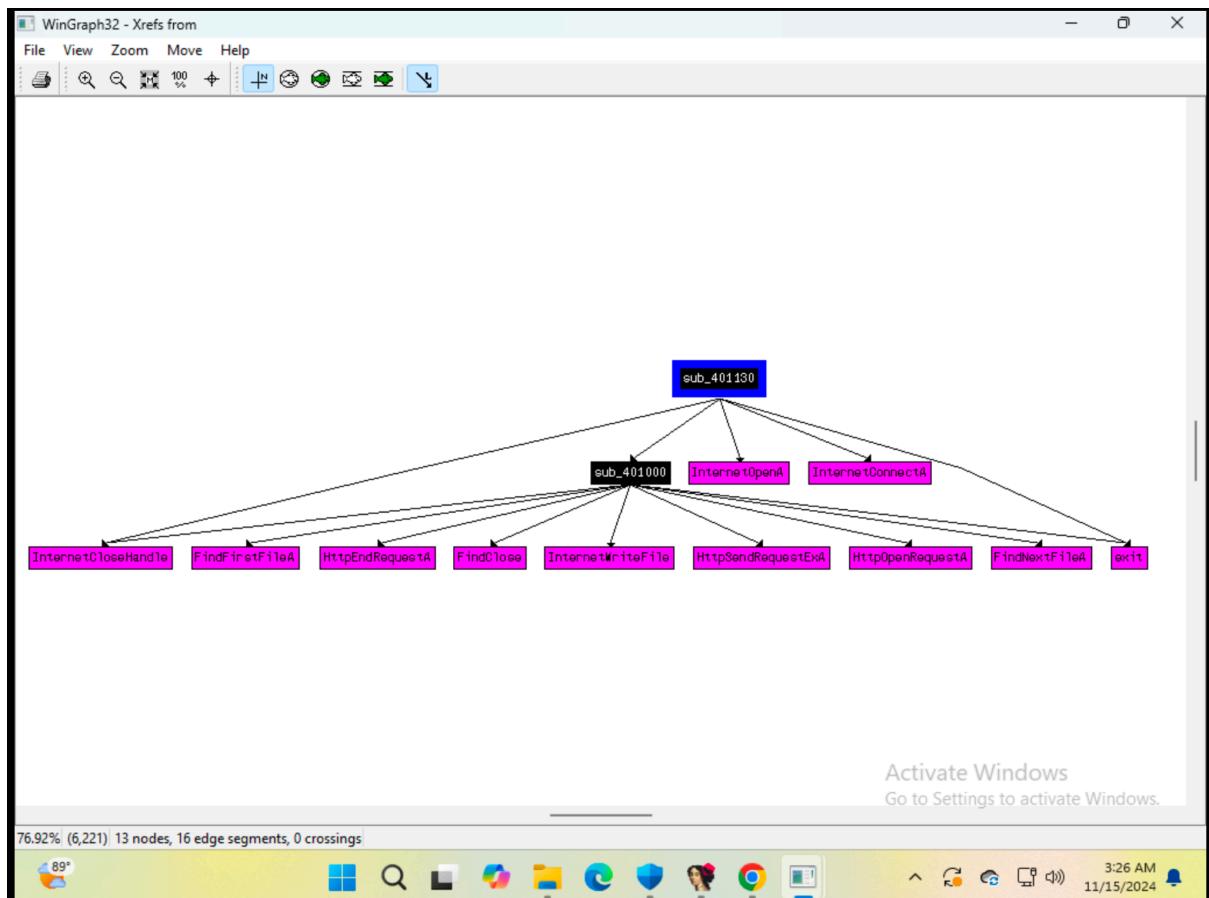
b.What Does This Function Do?

- Establishes an HTTP connection to "reversing.rocks" on port 66 using `InternetConnectA`.
- If the connection fails, exits the program (`exit(1)`).
- If successful, calls `sub_401000` (likely sends/receives data).
- Cleans up the connection using `InternetCloseHandle`.

i. Code Constructs Used:

- API Calls: `InternetConnectA`, `InternetCloseHandle`.
- Conditional Branching: Tests the result of `InternetConnectA` (test `esi, esi`) and exits on failure.

- Function Call: Executes `sub_401000` for further network operations.



3. Looking at the subroutine at 0x00401000:

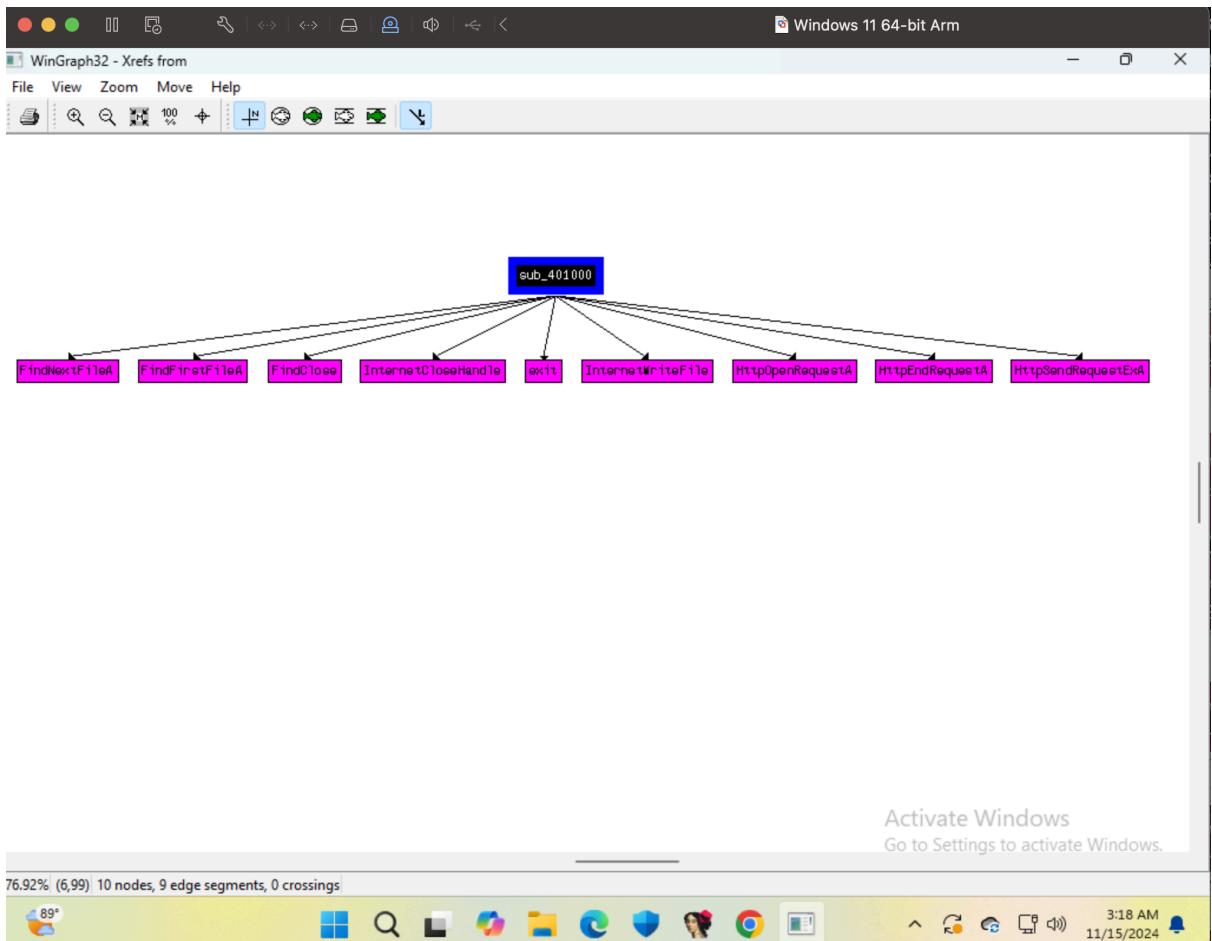
a. Code Constructs:

- Loops to iterate files.
- Conditional checks for success/failure.
- API calls to handle files and HTTP requests.

b. Imported Functions:

- `FindFirstFileA`, `FindNextFileA` (file enumeration).
- `HttpOpenRequestA`, `HttpSendRequestExA` (HTTP requests).
- `InternetWriteFile` (send data).

- `HttpEndRequestA`, `InternetCloseHandle`, `FindClose` (cleanup).



c. What It Does:

- Enumerates filenames in the directory.
- Sends filenames to [reversing.rocks](#) via HTTP POST.
- Finalizes and cleans up network and file handles.

4. What does this malware do?

The malware attempts to connect to the creator's site and then exfiltrate files from the local drive to his server. It then closes the connection and quits.

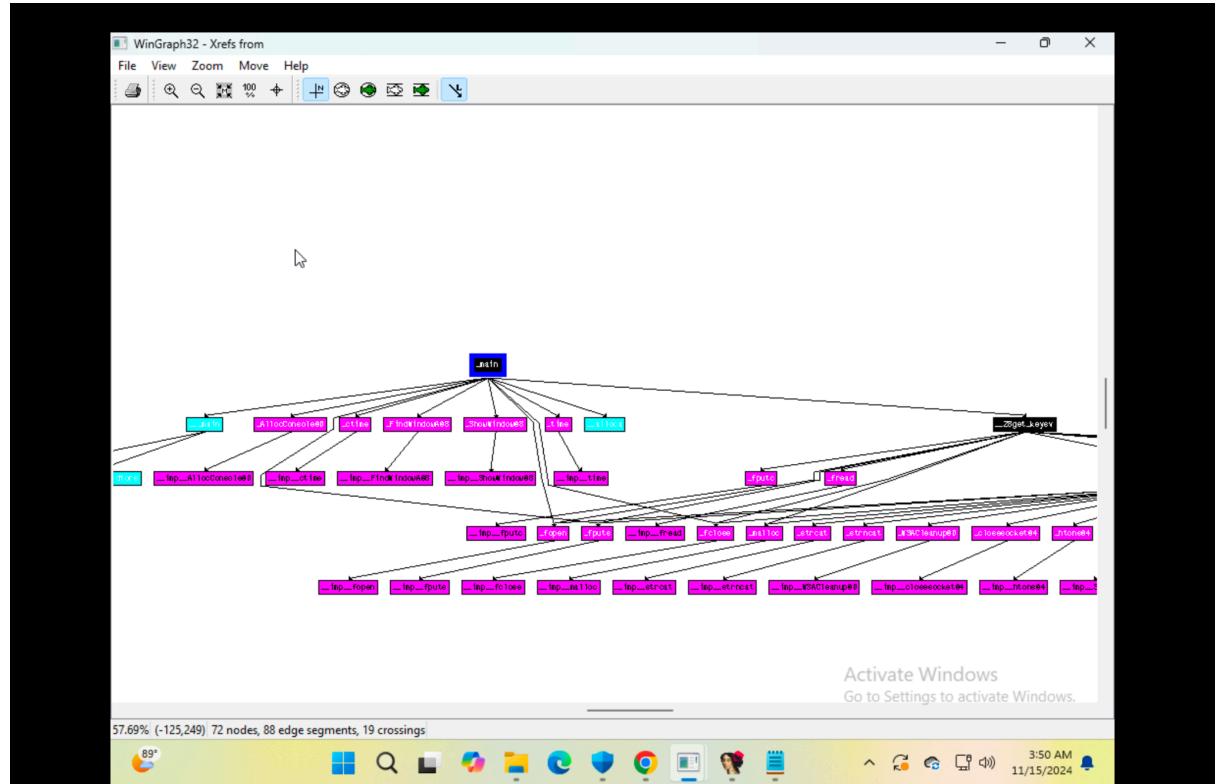
V. Lab_02-2.malware

1. Main function:

a. Imported Functions:

- `AllocConsole()`: Creates a new console window.
- `FindWindowA()`: Finds a window by its class and name.
- `ShowWindow()`: Controls window visibility.
- `fopen()`: Opens a file for reading/writing.
- `time()`: Gets the current system time.
- `fputs()`: Writes a string to a file.
- `ctime()`: Converts time to a readable format.
- `fclose()`: Closes the file.

- `get_keys()`: Likely logs keystrokes.



b. Interesting Strings:

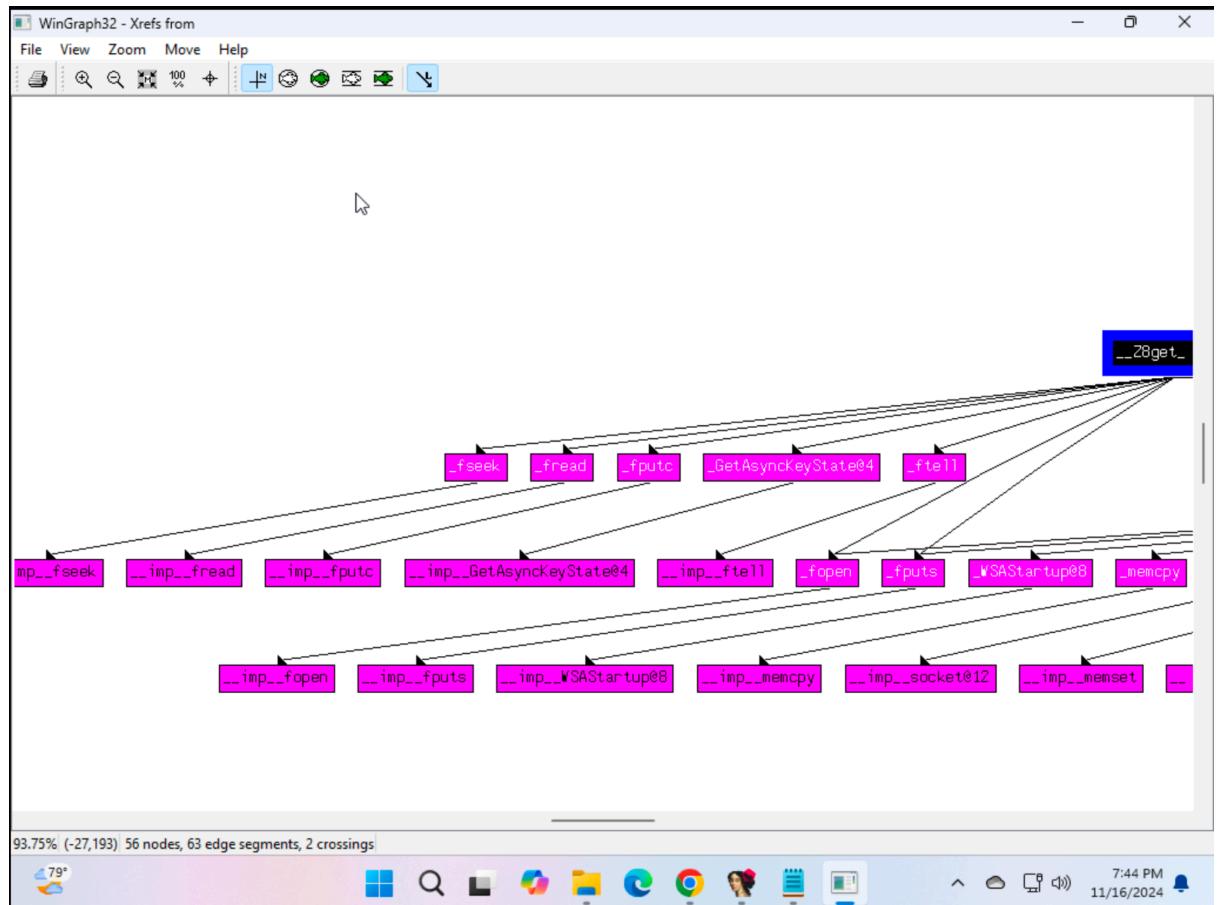
- `"\\Windows\\lzwindowlz.av"`: File path used for logging.
 - `"Started logging: "`: Log message.
 - `"ConsoleWindowClass"`: Standard window class name.

2. Looking at the subroutine at 0x0040135C:

a. Imported Functions:

- `_fopen`, `_fputs`, `_fclose`: File operations.
 - `_GetAsyncKeyState`: Checks key state.
 - `_malloc`: Allocates memory.
 - `_WSAStartup`, `_socket`, `_connect`, `_recv`, `_send`: Networking functions.
 - `_time`, `_ctime`: Time functions.
 - `_exit`, `_Sleep`: Control flow and exit.

- `_strcpy`, `_strncat`, `_strlen`: String operations.



b. Code Constructs:

- Conditional jumps (`jne`, `jg`, `jz`): Control flow.
- Indirect jump (`jmp eax`): Dynamic function call based on runtime conditions.
- Loops: Iterates through keypresses and actions.

3. What does this malware do?

a. What does this malware do?

- Keylogging: It logs keypresses using `GetAsyncKeyState` and stores them in a file.
- File Handling: It creates and writes to "`\WINDOWS\lzwindow1z.av`", logging keystrokes.
- Networking: It connects to a remote server, sends/receives data using socket functions.
- Mailing: It uses `MailIt` to potentially send data via email.

i. Proposed Signatures:

- File path signature: Monitor for "\\WINDOWS\\lzwindowlz.av" in `fopen()` calls.
- Keylogging signature: Look for `GetAsyncKeyState` and specific keypresses (e.g., `ENTER`, `SHIFT`).
- Network activity: Detect `socket`, `connect`, `send`, `recv` calls.
- Email communication: Track suspicious email addresses like "unknown-g@hotmail.co.uk".
- Persistence: Look for `Sleep()` calls indicating delays.

ii. File Creation:

- Yes, the sample creates the file \\WINDOWS\\lzwindowlz.av to log keystrokes, storing the data for later access or exfiltration.