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

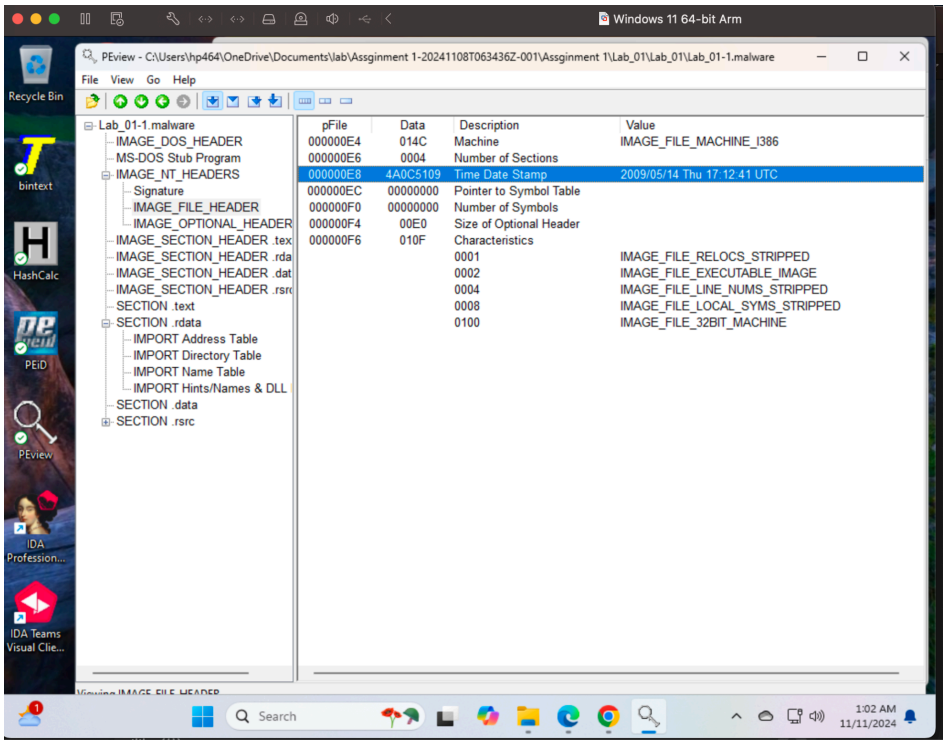
Malware Analysis **1**
Assignment 1 **1**
I. LAB_01-01.malware 3
II. LAB_01-02.malware 10
IV. Lab_02-1.malware 18
V. Lab_02-2.malware 22

I. LAB_01-01.malware

1. 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



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

a. Process and Thread Control (KERNEL32.dll)

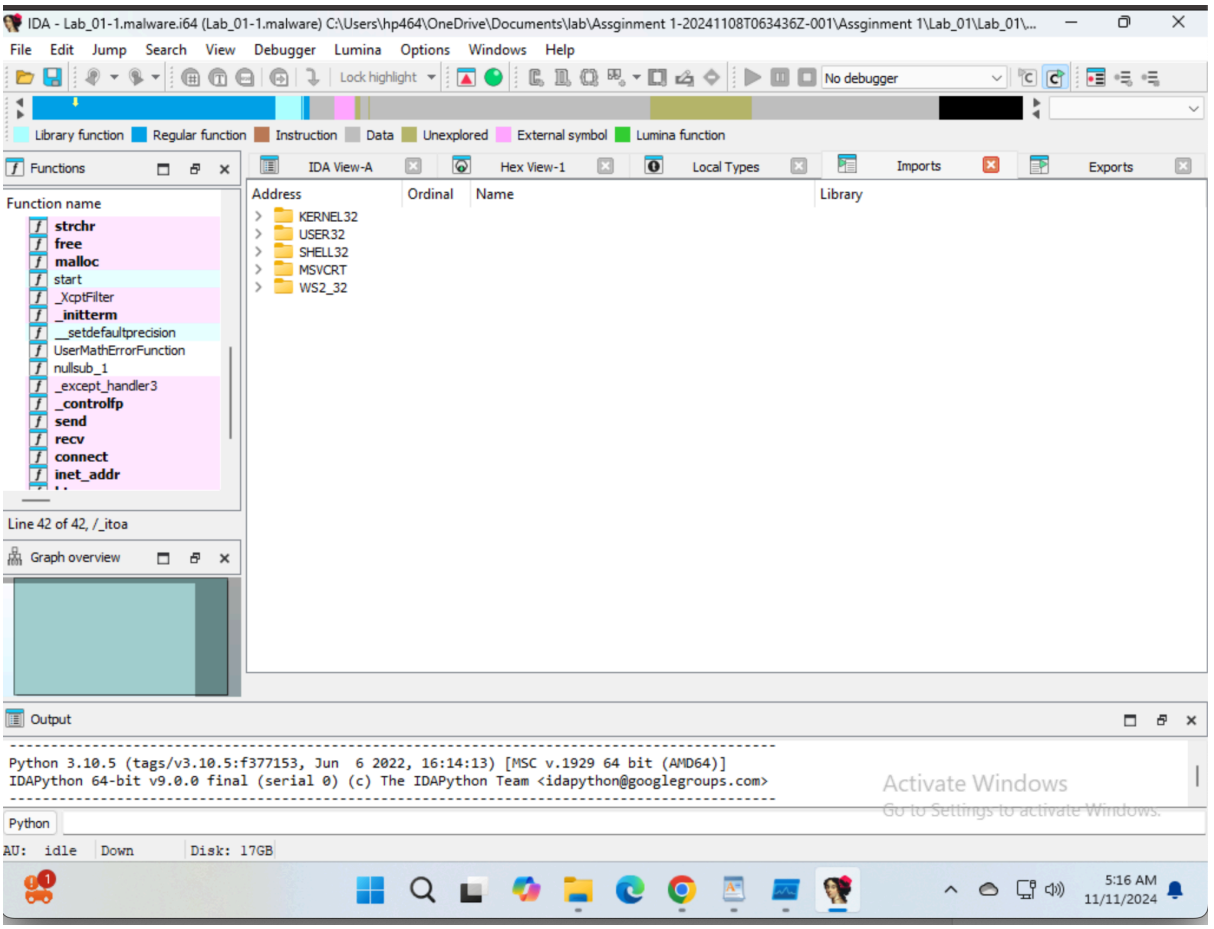
- CreateThread and CreateProcessA: These functions allow the malware to create new threads or processes, potentially to execute additional malicious code or manage multiple tasks simultaneously.
- TerminateProcess and ExitThread: These functions can be used to stop threads or processes, possibly to end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

b. 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 WSAStartup: These functions initialize network communication, setting up sockets and preparing the application for network operations.

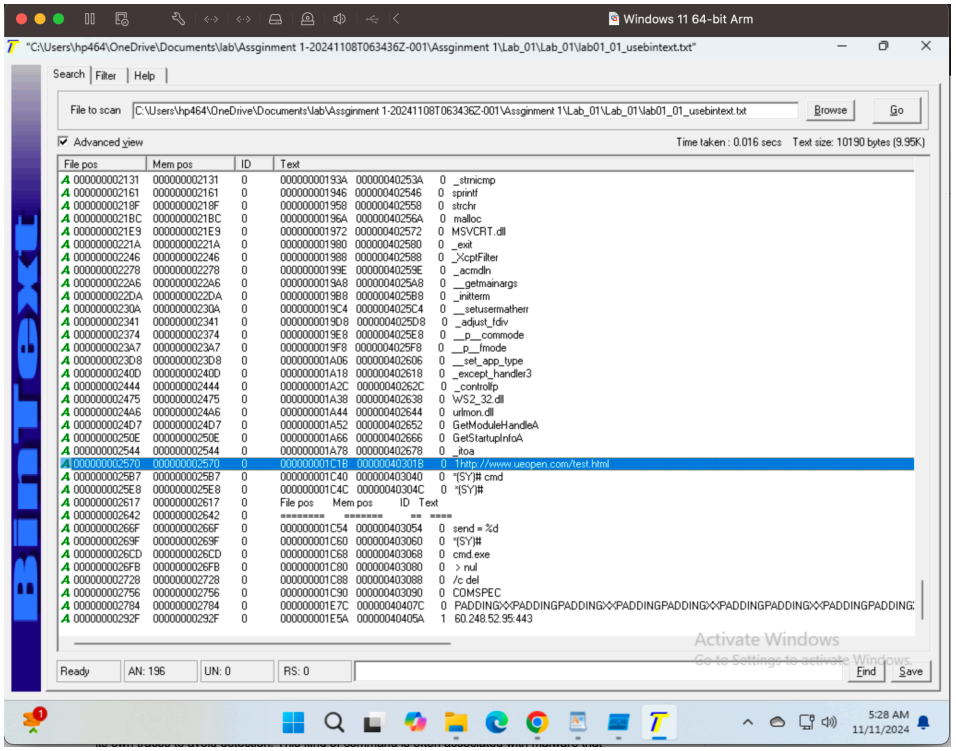
c. Memory and String Management (MSVCRT.dll)

- 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.



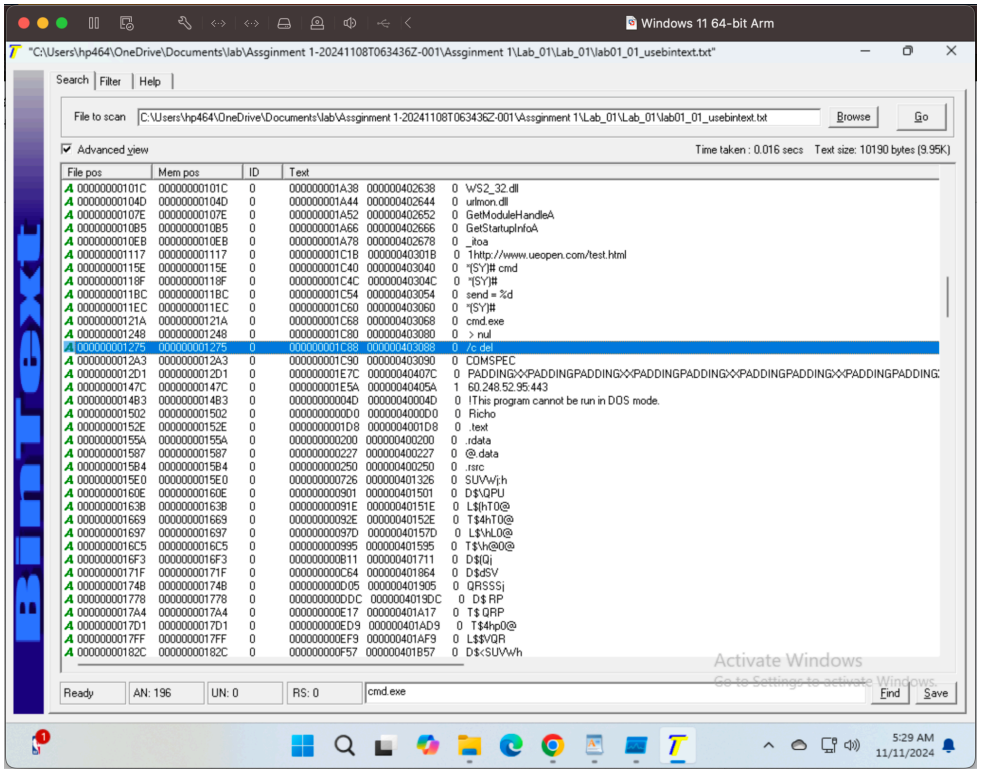
3. What are a few strings that stick out to you and why?
   a. Network Connection: http://www.ueopen.com/test.html

      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.



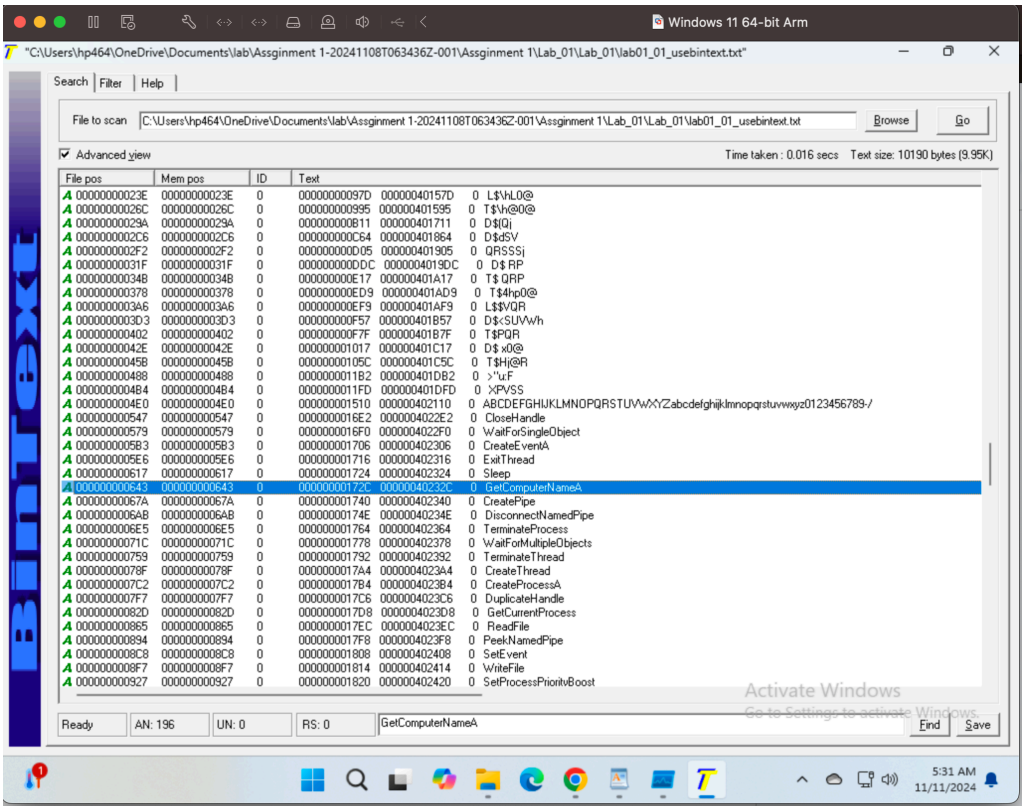
b. Command Execution: cmd.exe and /c del

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.



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

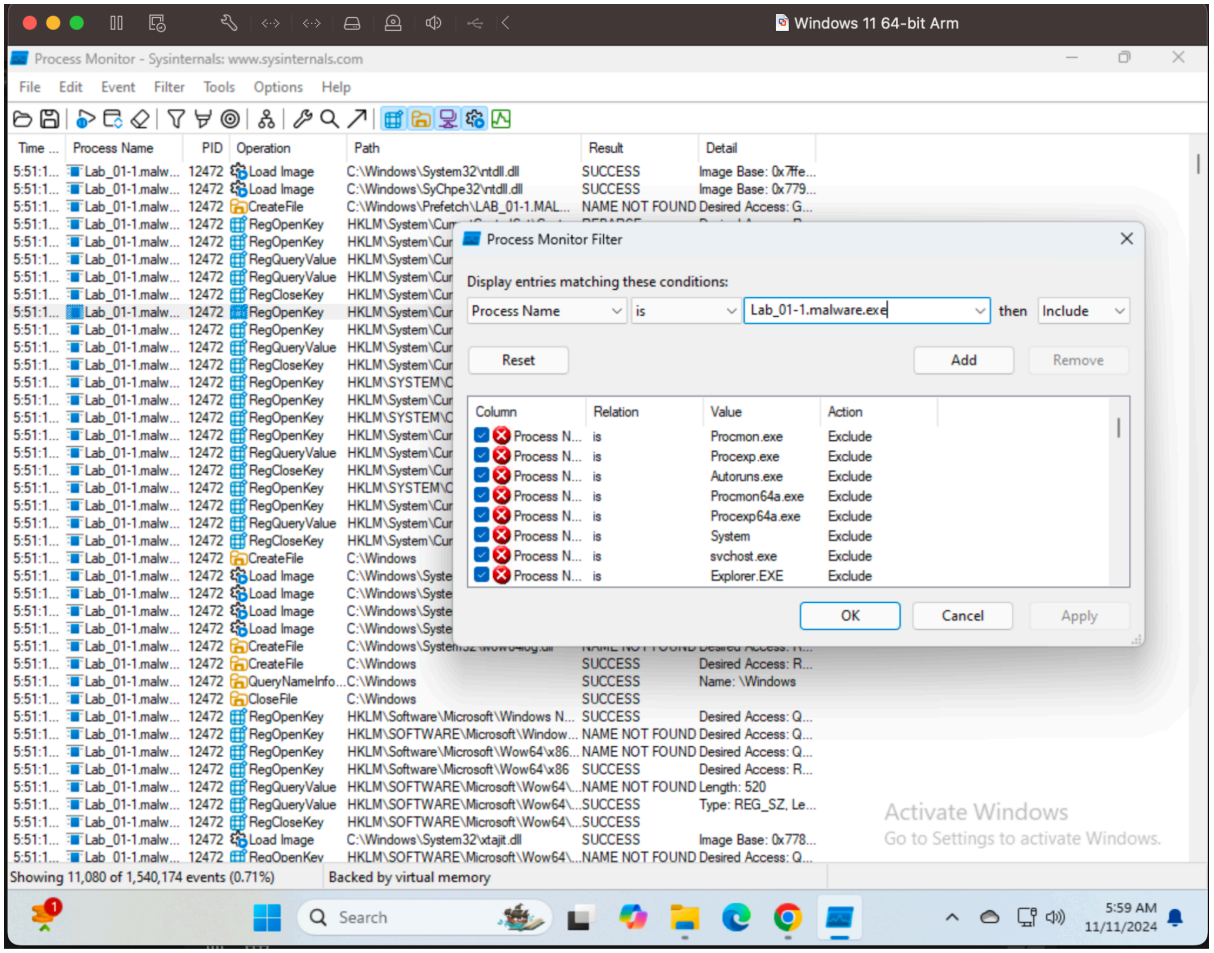
a. Establish a Network Connection: The presence of the IP address 60.248.52.95:443 indicates that the malware might try to connect to this remote server over port 443 (typically used for HTTPS traffic), possibly to communicate with a command-and-control (C2) server for instructions or to exfiltrate data.
b. 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.
c. 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?

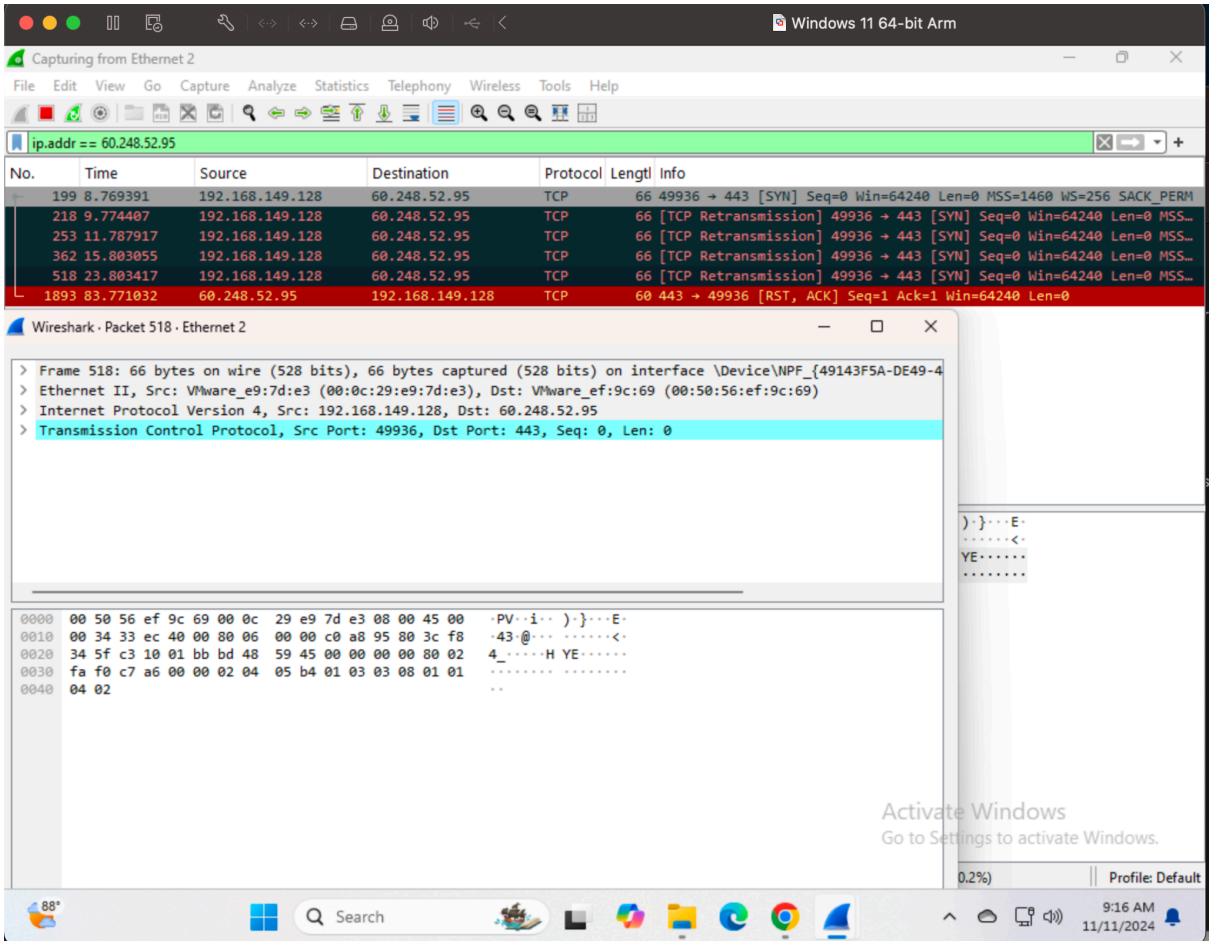
a. URLs:
- http://www.ueopen.com/test.html
  - This URL likely serves as a command-and-control (C2) endpoint or may be used to download additional instructions or exfiltrate data.

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.



8. 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

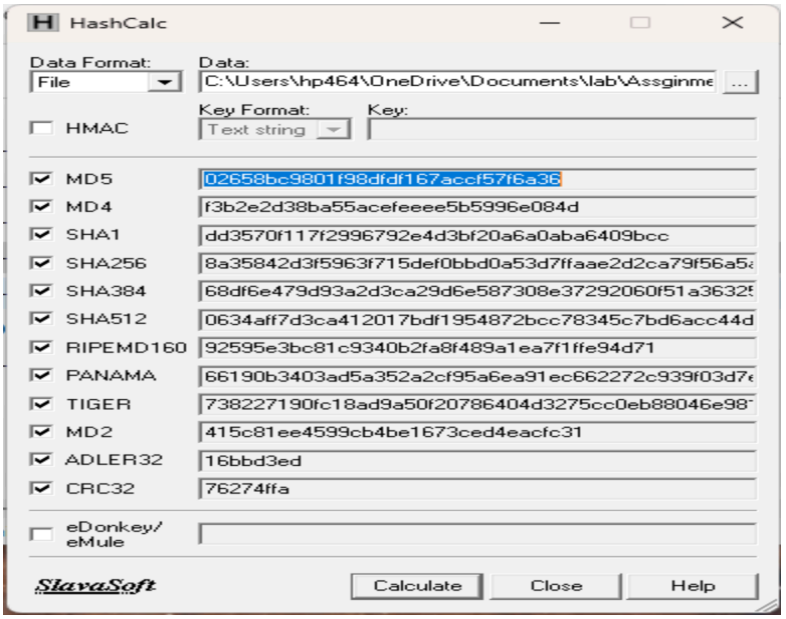
9. What do you think is the purpose of this malware?

   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?

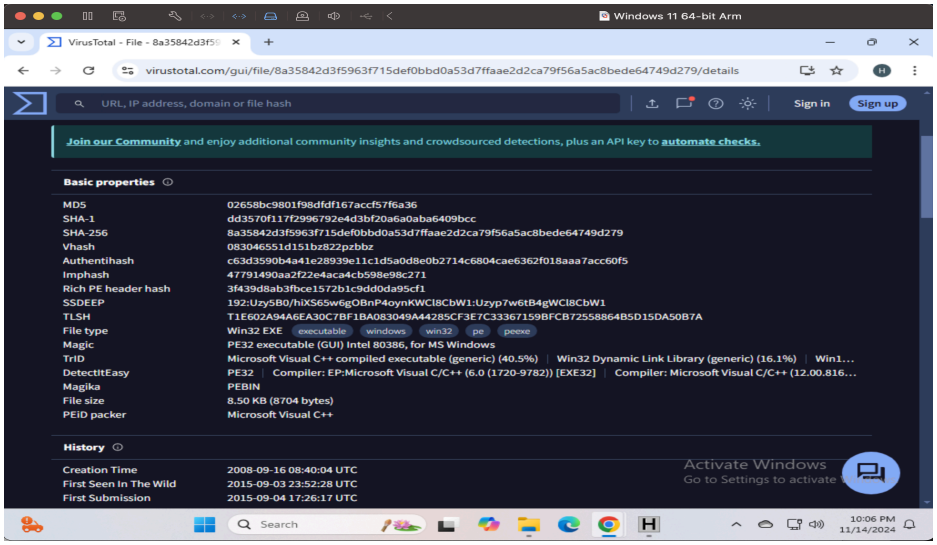
The md5sum for the file shown in the screenshots is 02658bc9801f98dfd167accf576a36.



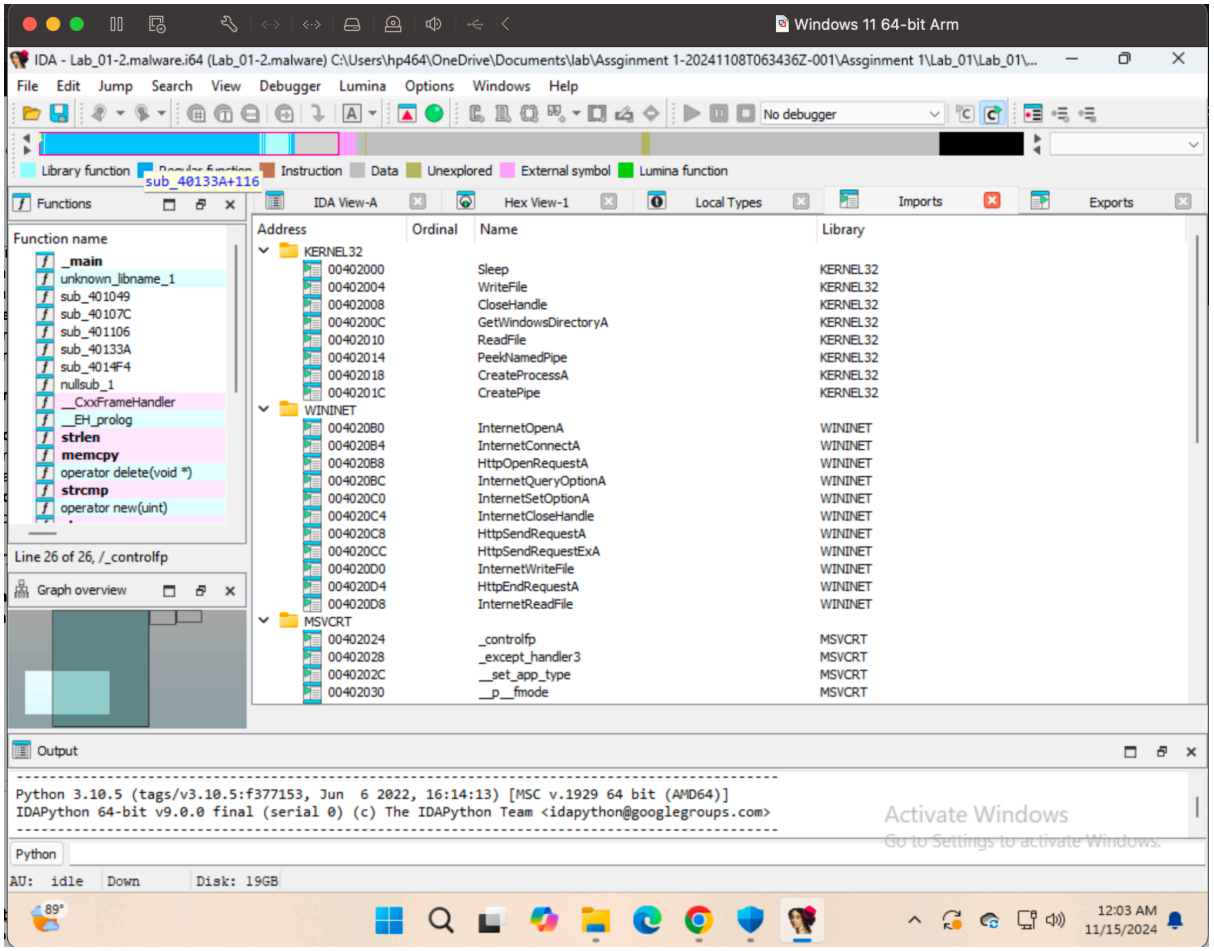
The VirusTotal report indicates the following details of interest:

a. SHA-1: dd3570f117f2996792e4d3bf20a6a0aba6409bcc
b. SHA-256: 8a35842d3f5963f715def0bbd0a53d7ffaa2d2ca79f56a5ac8bede64749d279
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?
   a. 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.
   b. wuauclt.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.
   c. 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.

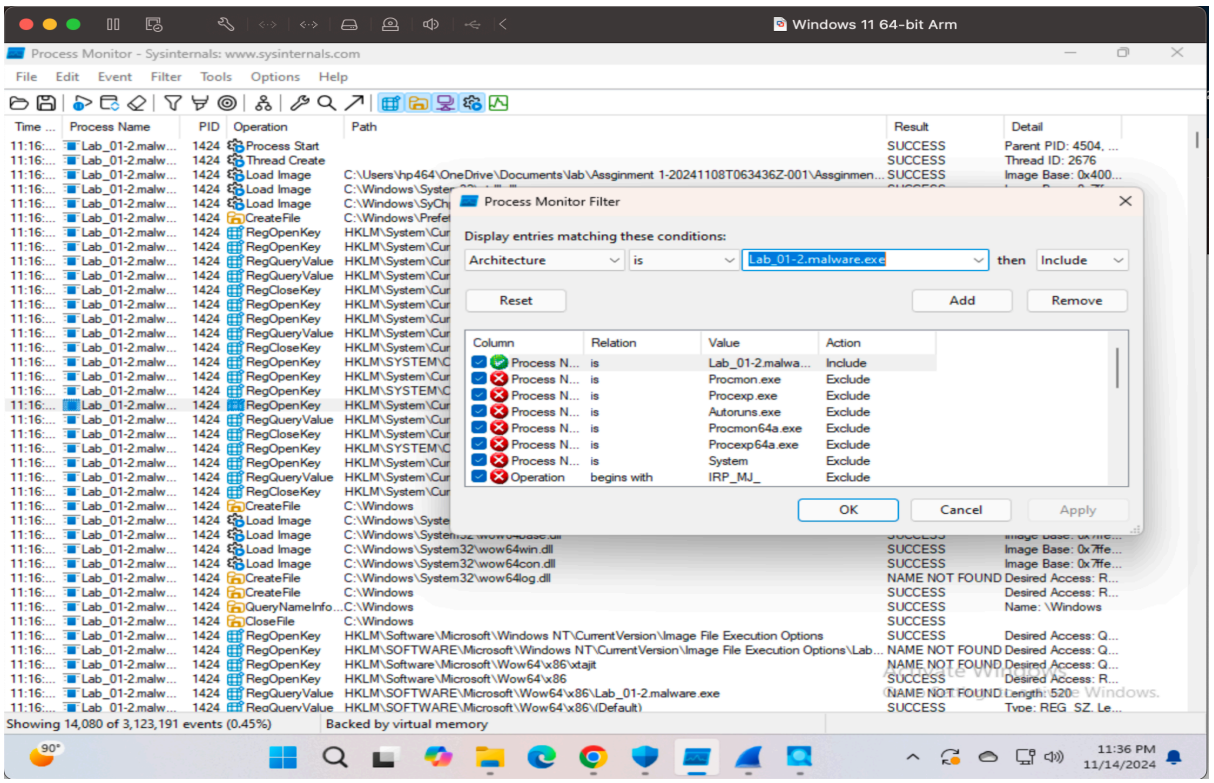
4. What happens when you run this malware? Is it what you expected and why?

   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.



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

Files:

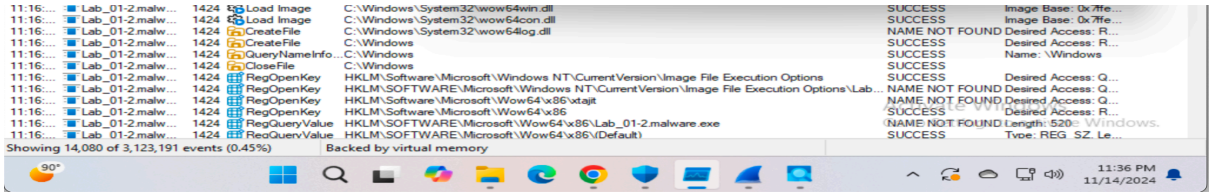
- C:\Windows\Prefetch\LAB_01-2.MALWARE.EXE-0835F...: The presence of this prefetch file name indicates that the executable Lab_01-2.malware.exe has been run on the system. Monitoring the Prefetch folder for unusual or suspicious executables can help detect the malware.
- C:\Windows\System32\wow64log.dll: 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.



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:

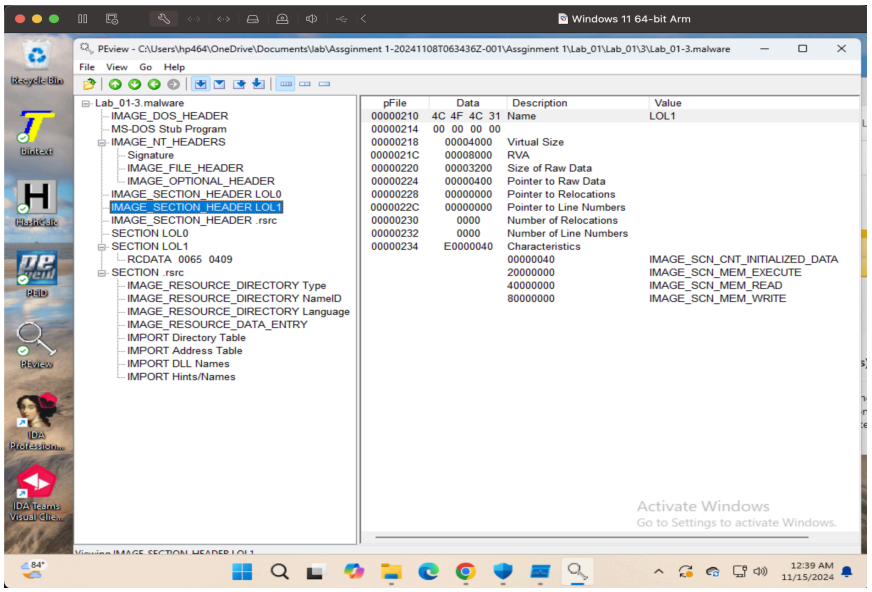
   - Packet Contents: The logs show Length: 0 for the packets, suggesting that these might be keep-alive signals or that the actual data is encrypted, making it challenging to interpret packet contents directly from the log.
8. Is there anything that impeded your analysis? How so? How might you overcome 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?

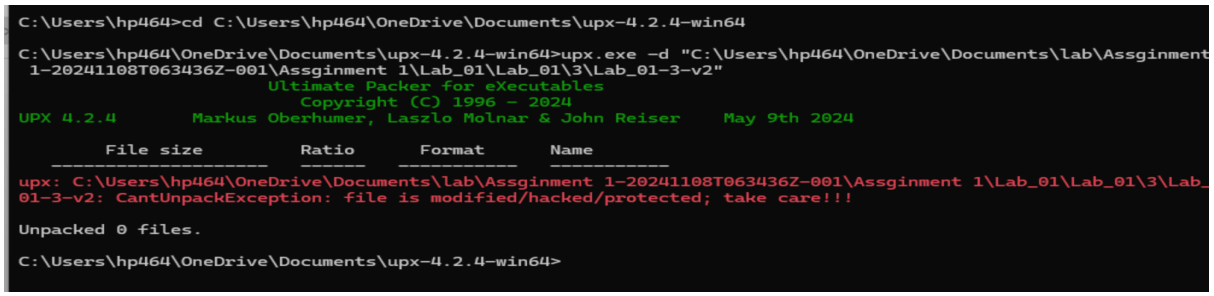
The malware acts as a backdoor, enabling remote access to the infected system. Its purpose is to allow the attacker to execute commands, manipulate files, and maintain persistence while remaining hidden, likely for data theft, espionage, or further network compromise.

III. Lab_01-3.malware

1. 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.



2. Are you able to unpack it? Why or why not?



3. 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.

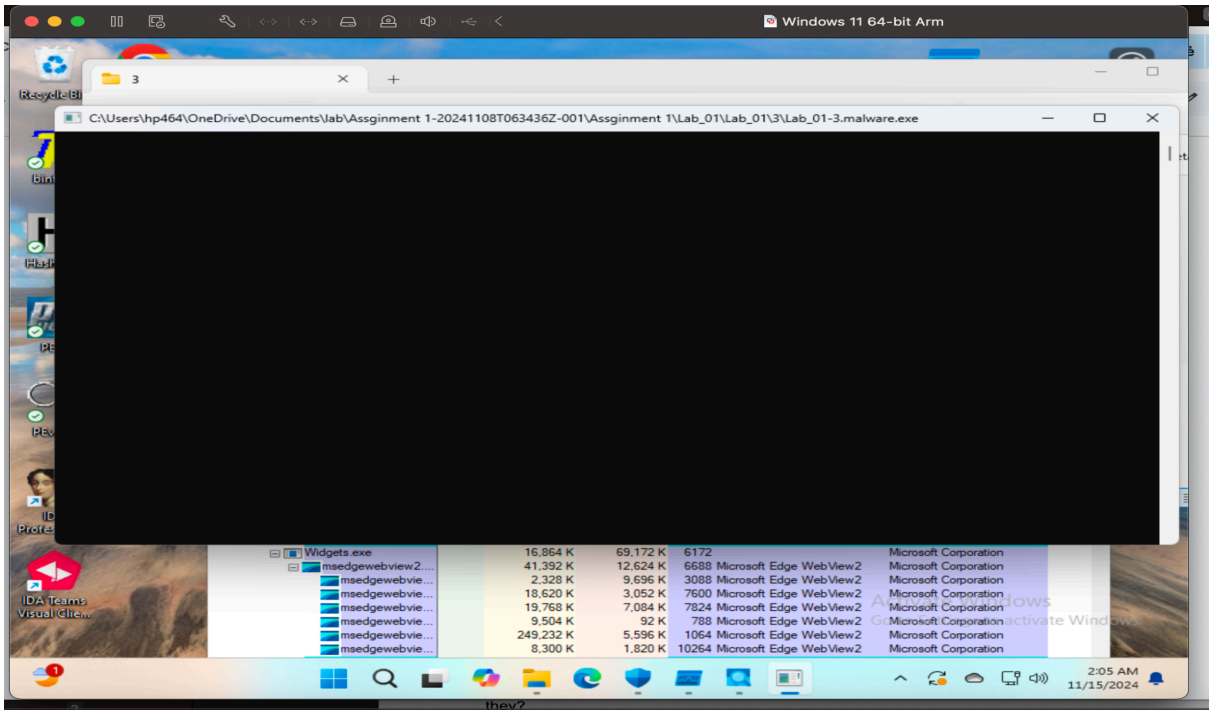
b. "http://%s"

- 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

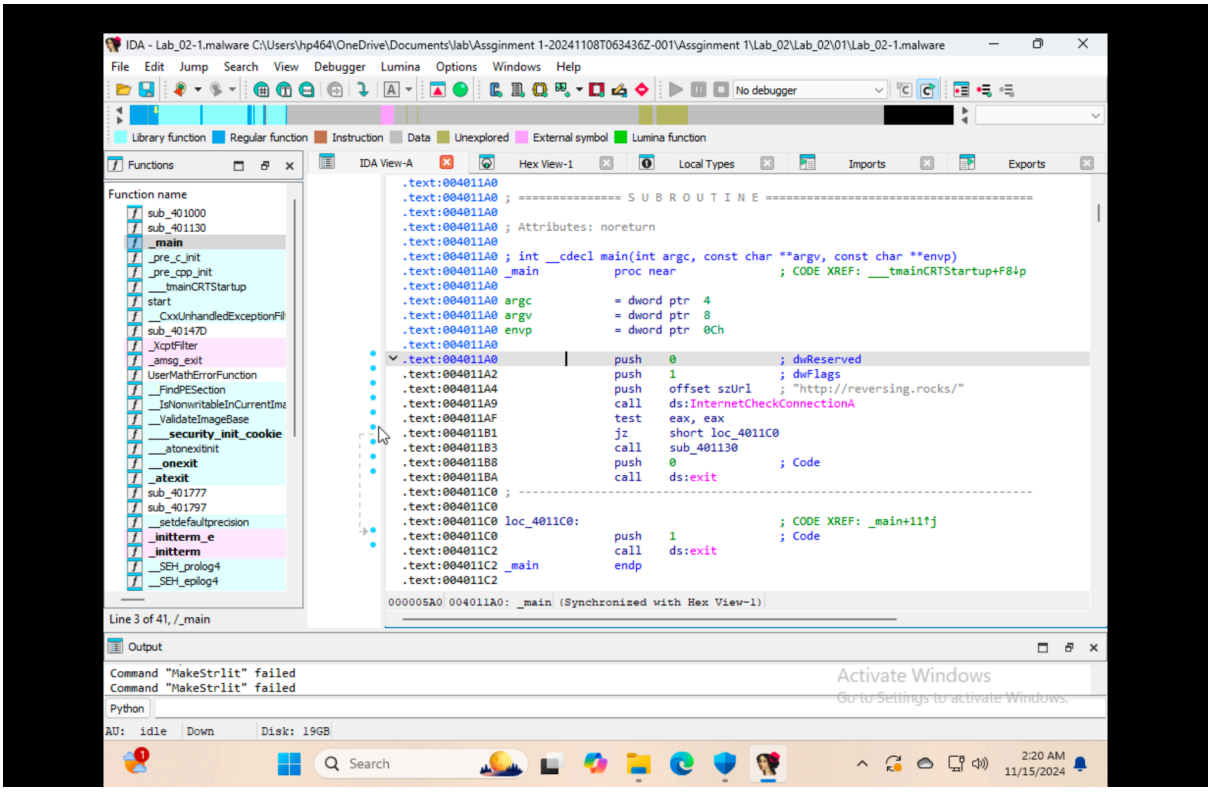
steps. Additionally, this functionality might serve as part of a larger modular malware framework, where further payloads are delivered depending on the system identified. Without further investigation, the malware's broader objectives remain unclear.

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.



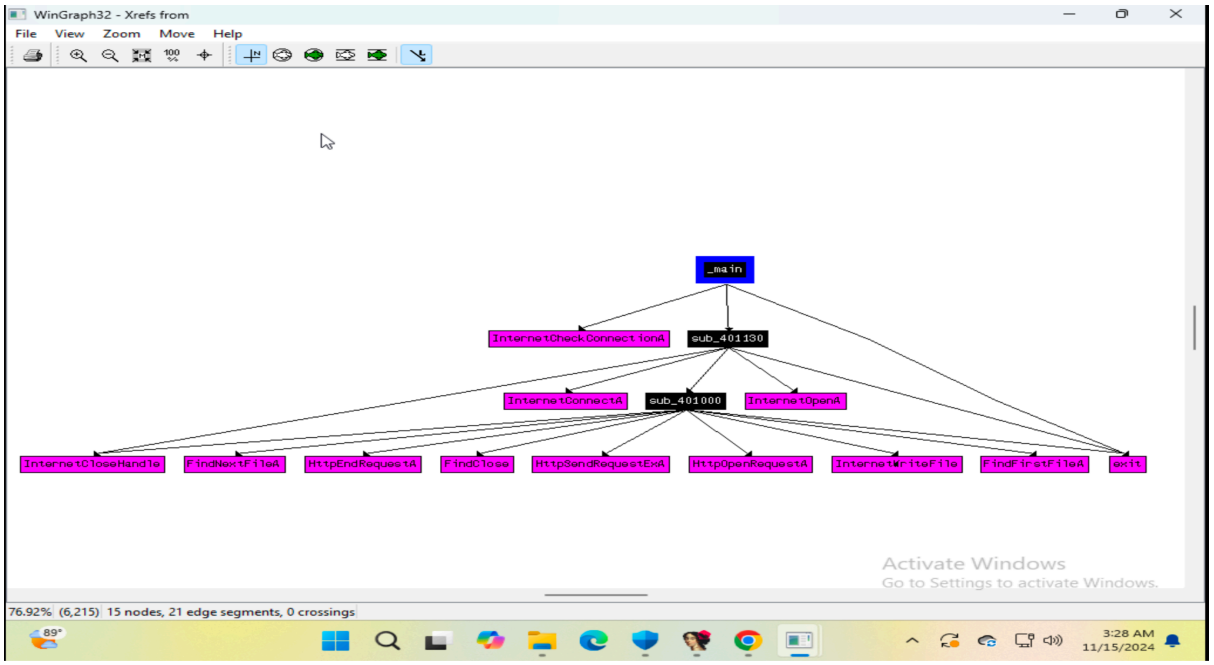
b. What does this function do?

- 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?

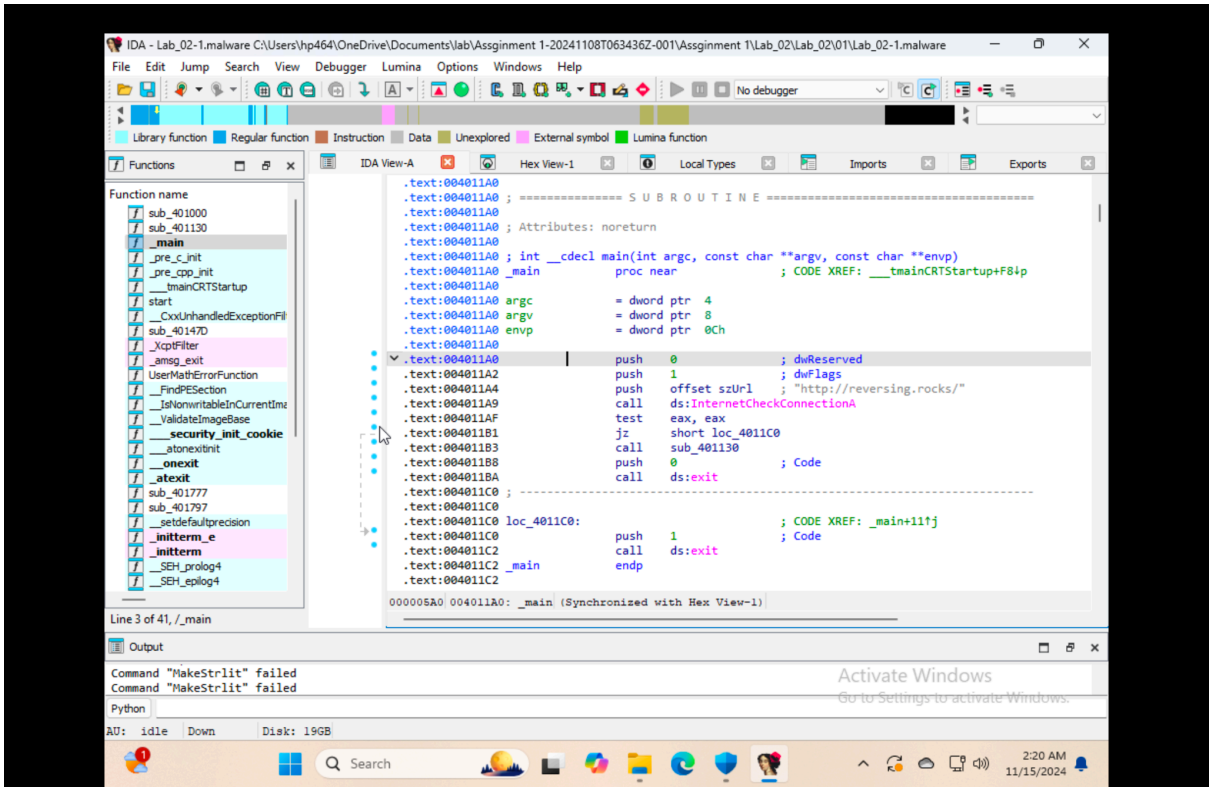
- API Call:
  - The function makes a call to the Windows API InternetCheckConnectionA, which is commonly used to test connectivity to a specific URL.
- Control Flow:
  - Conditional branching is performed using test eax, eax and jz (jump if zero) to handle the results of the API call:
    - If the test 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.



2. Looking at the subroutine a 0x00401130:
    a. Arguments to InternetConnectA and Their Meaning:
        - hInternet (EDI): Handle to the WinINet session (from InternetOpen).
        - lpszServerName (reversing.rocks): Server name to connect to.
        - nServerPort (66): Server port (custom port 66).
        - lpszUserName (0): No username for authentication.
        - lpszPassword (0): No password for authentication.
        - dwService (3): Service type (INTERNET_SERVICE_HTTP).
        - dwFlags and dwContext (0): No flags or context.

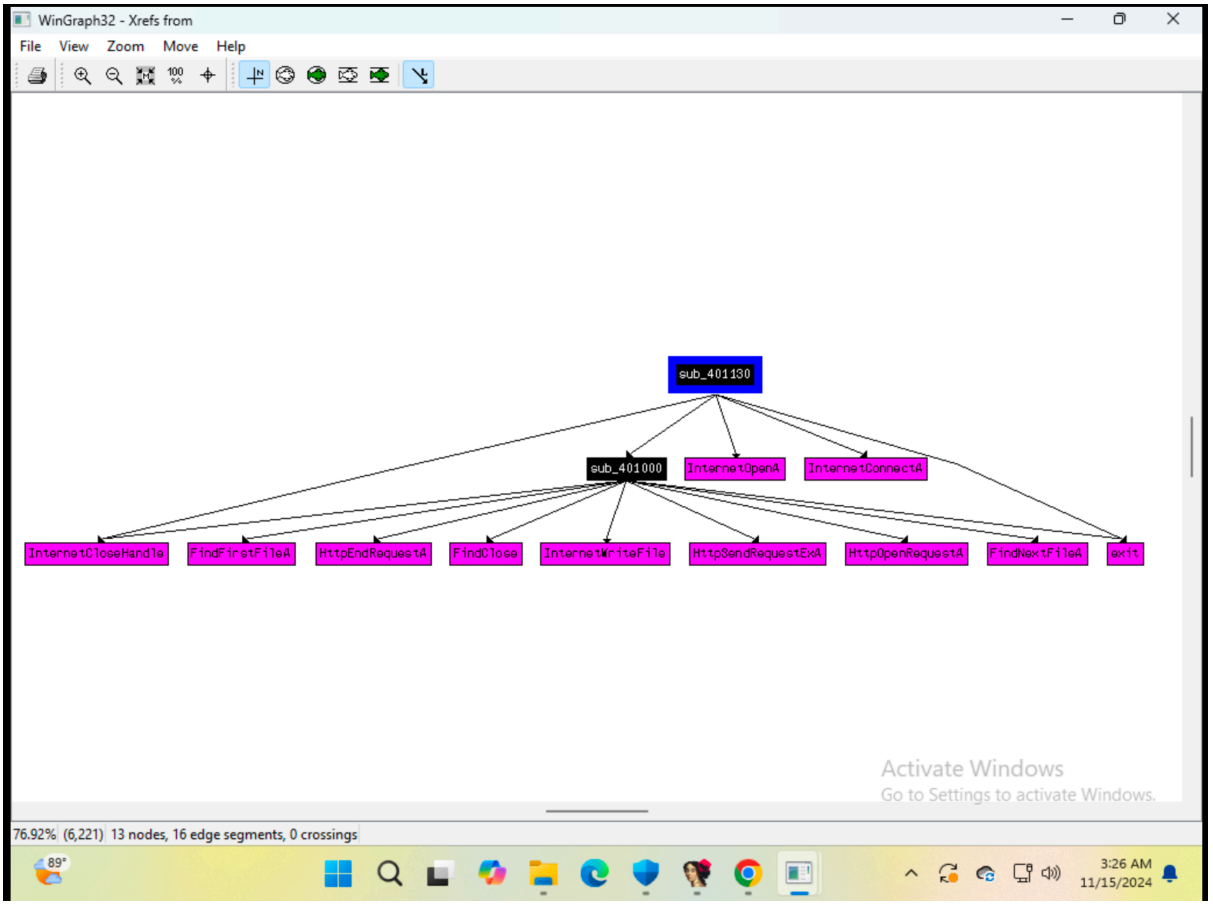
    b.What Does This Function Do?

        - 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.



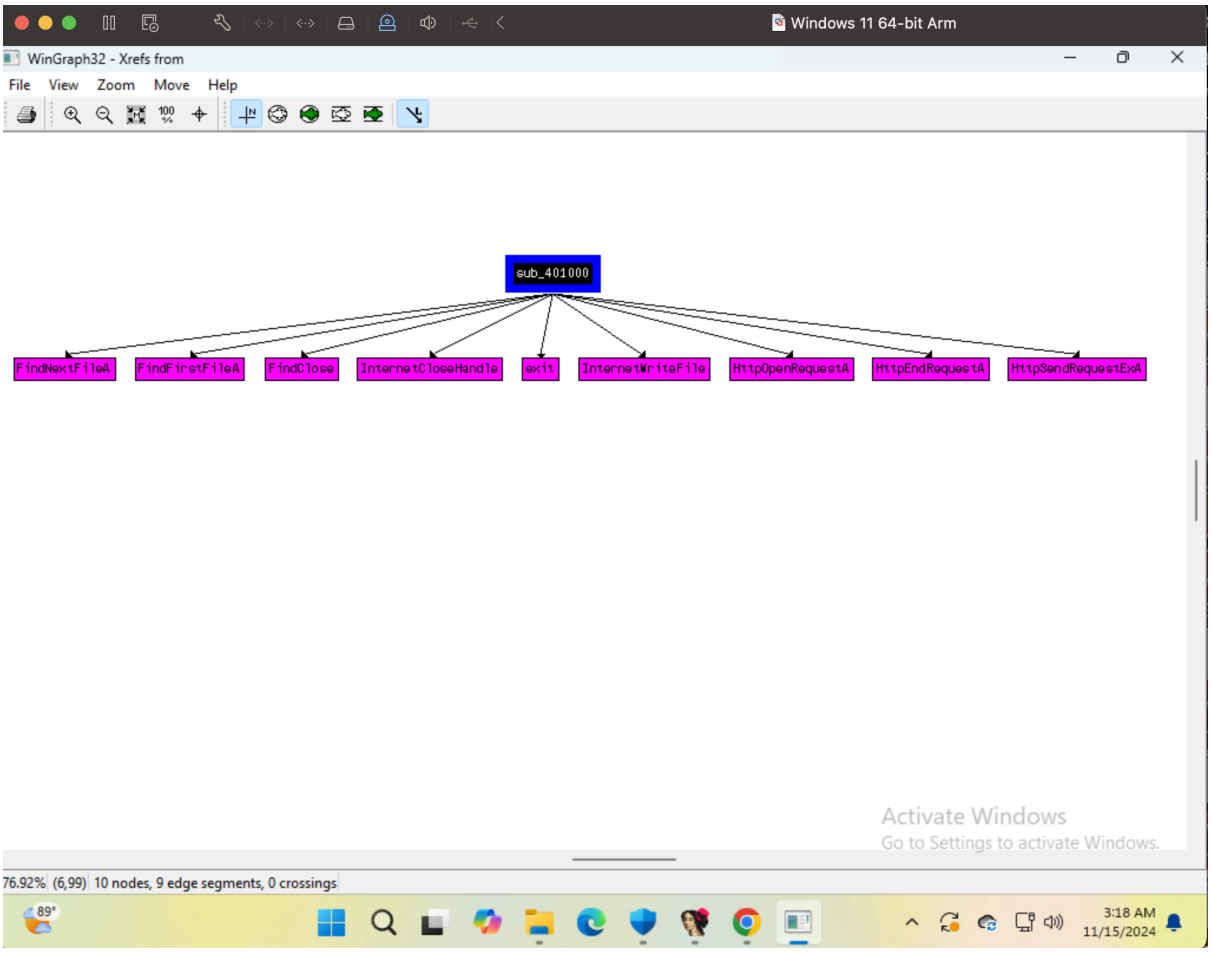
3. Looking at the subroutine at 0x00401000:

   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:

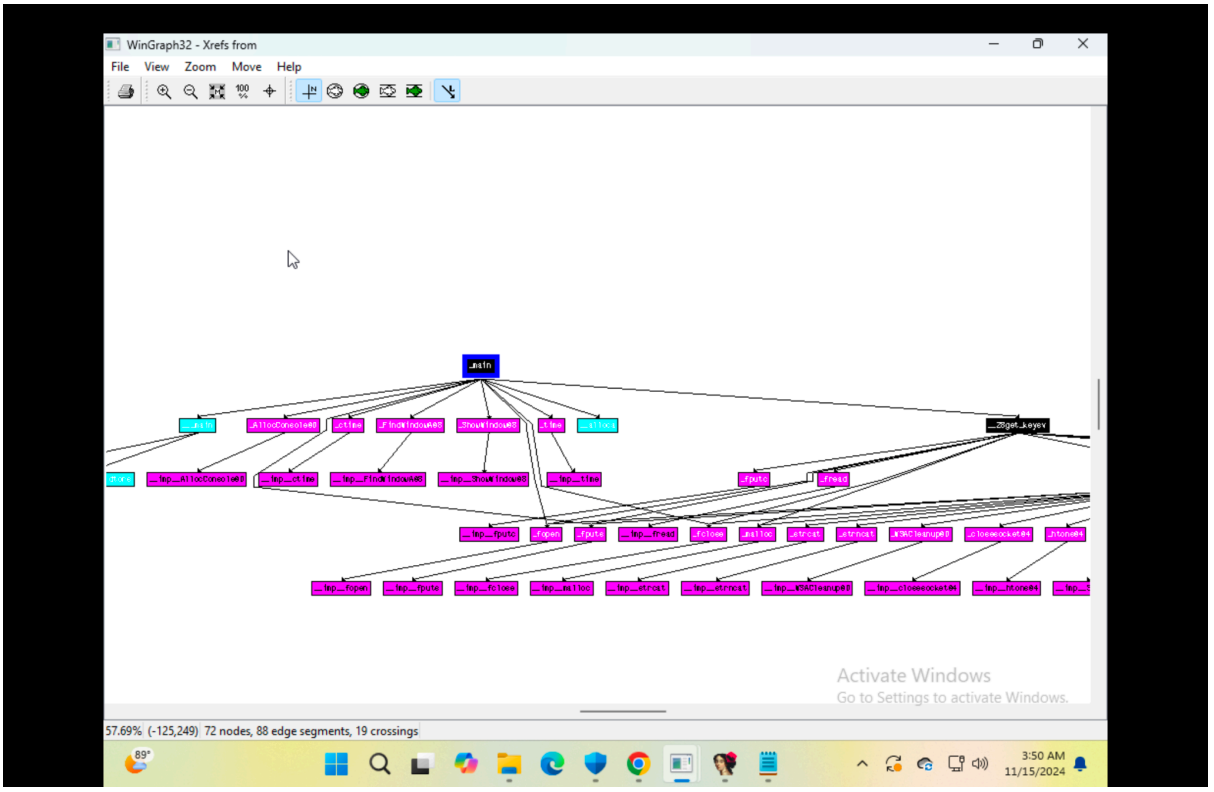
- Enumerates filenames in the directory.
- Sends filenames to reversing.rocks via HTTP POST.
- Finalizes and cleans up network and file handles.

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:

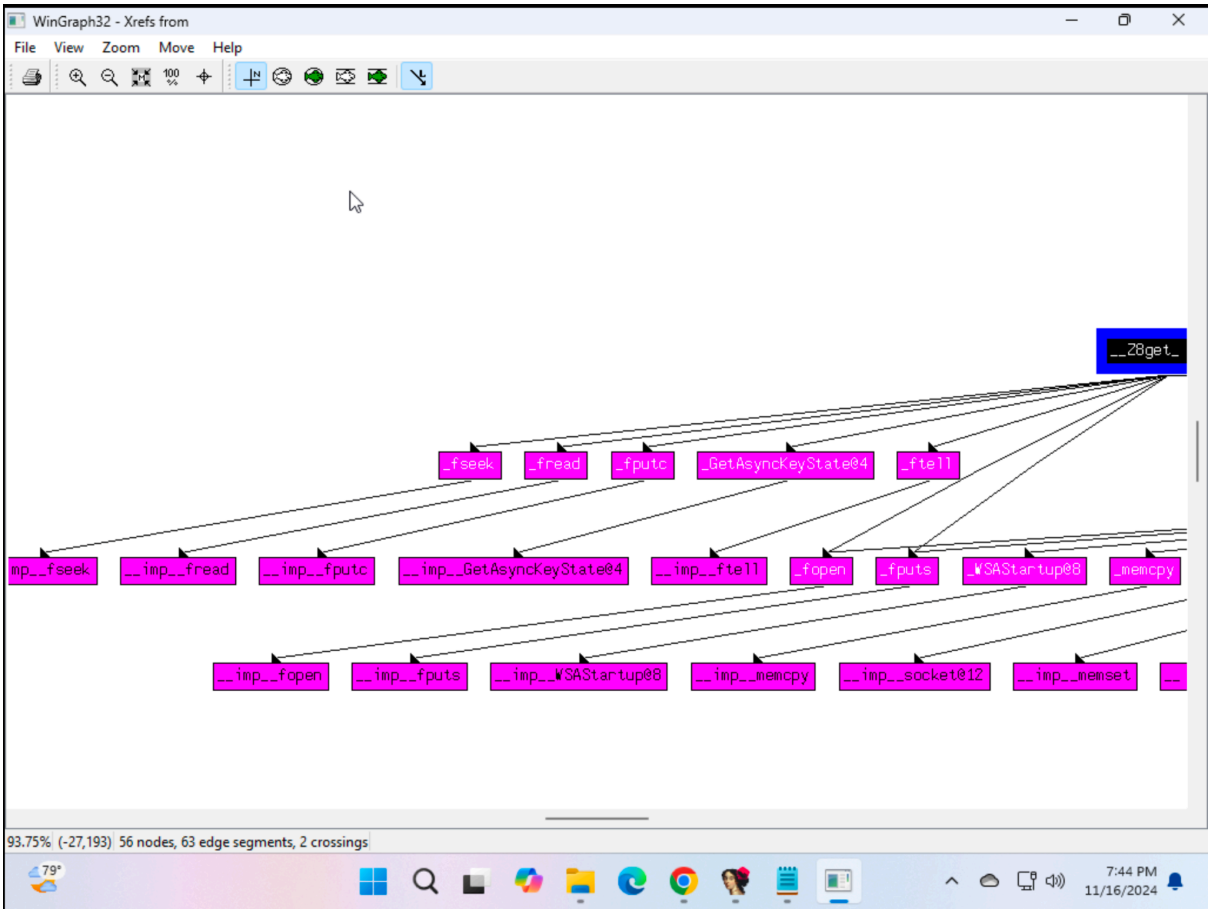
- "\\WINDOWS\\lzwindowlz.av": File path used for logging.
- "Started logging: ": Log message.
- "ConsoleWindowClass": Standard window class name.

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\\lzwindowlz.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.