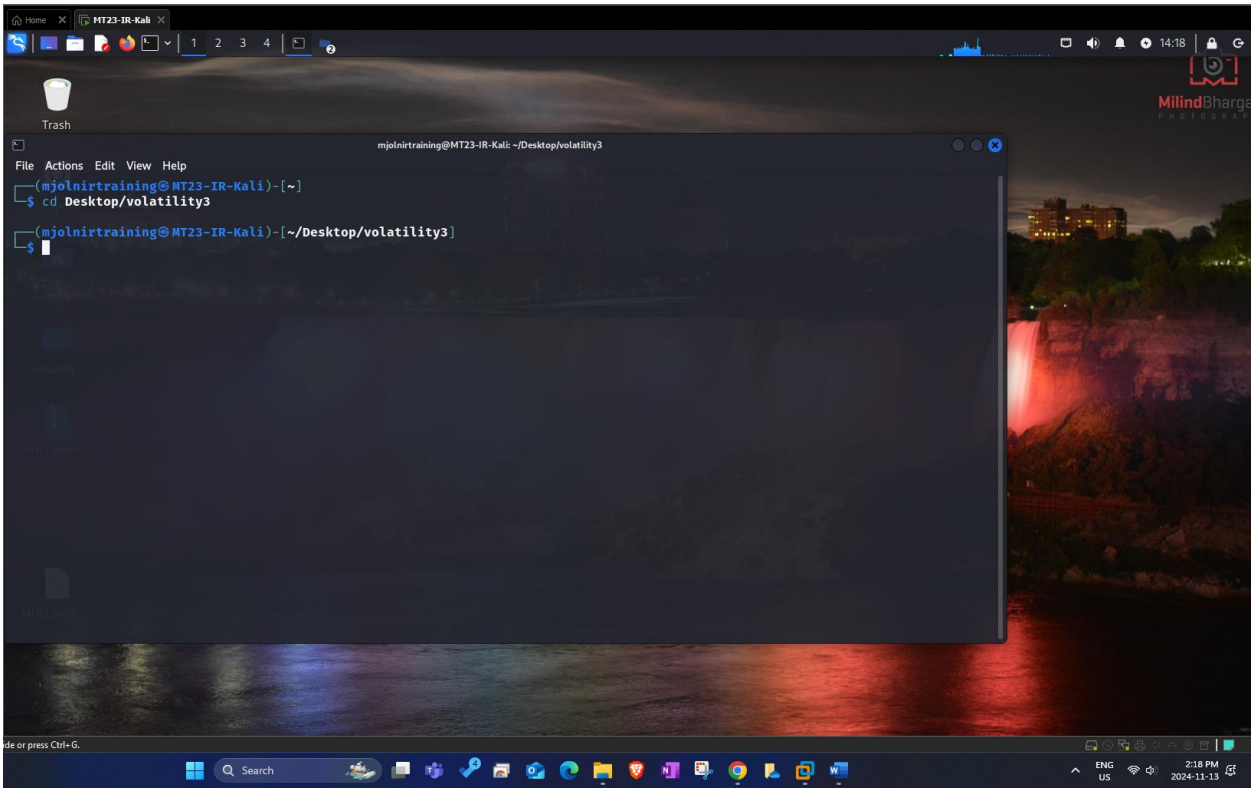
# MOHAMAD ALMASR 101167438

## DIGITAL FORENSICS&INCID. RESP - COMP 4071

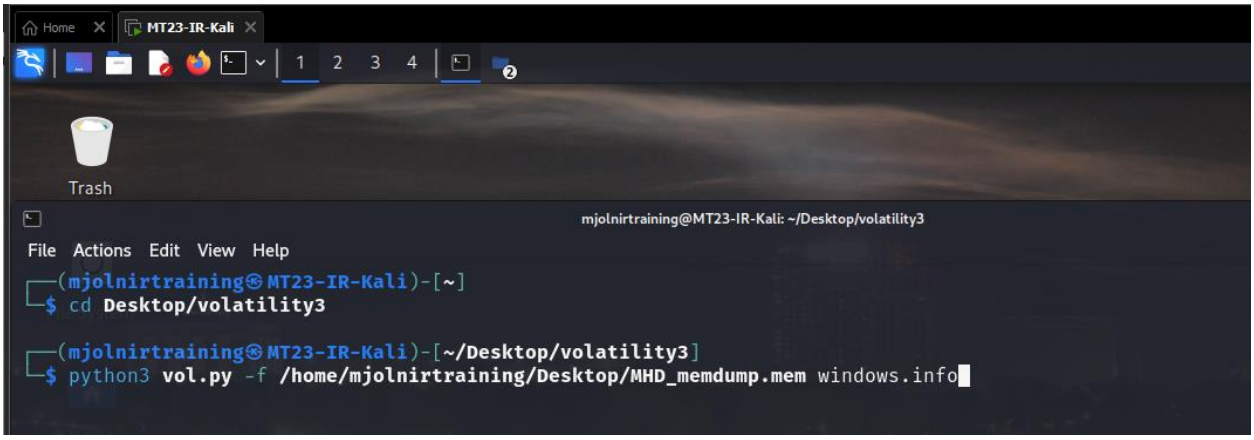
HOMEWORK LAB 3

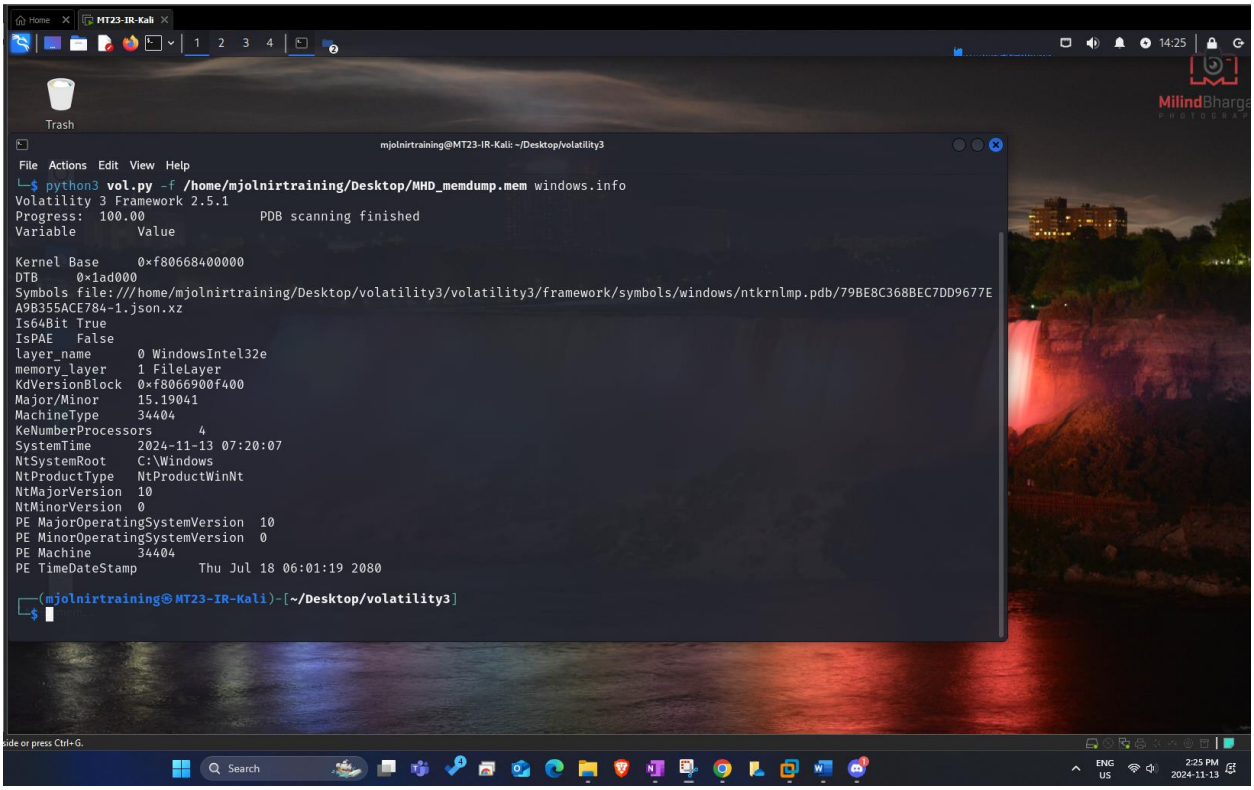
Move the Memory Dump: I started by transferring your memory dump file to Kali Linux, so I could work on it there.

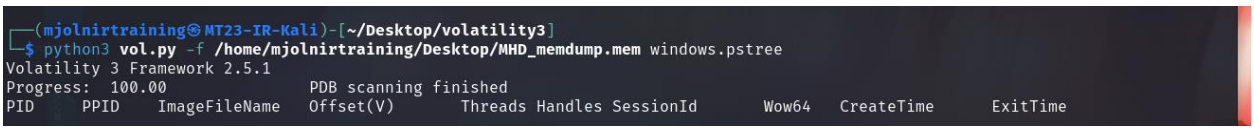
Check the Directory: I navigated to the directory where the memory dump is located to make sure it’s ready for analysis.



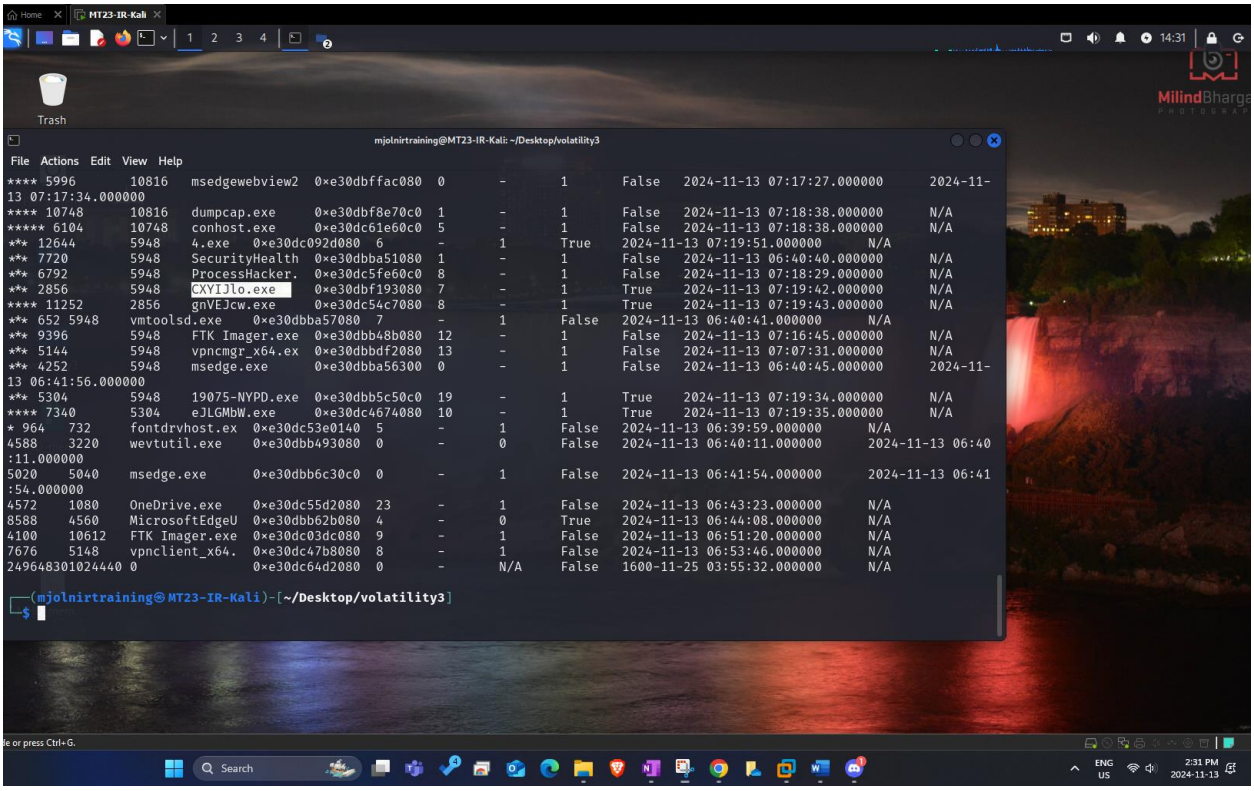
# MOHAMAD ALMASR 101167438

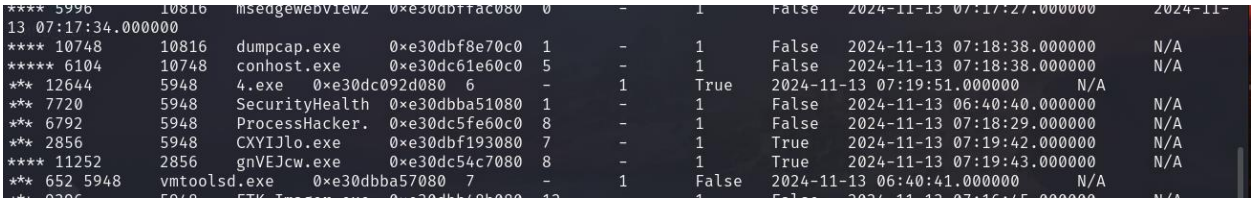


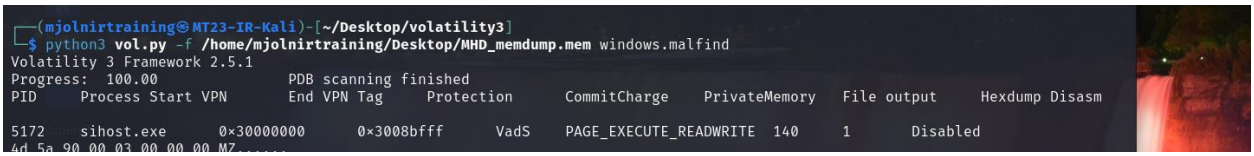



# MOHAMAD ALMASR 101167438

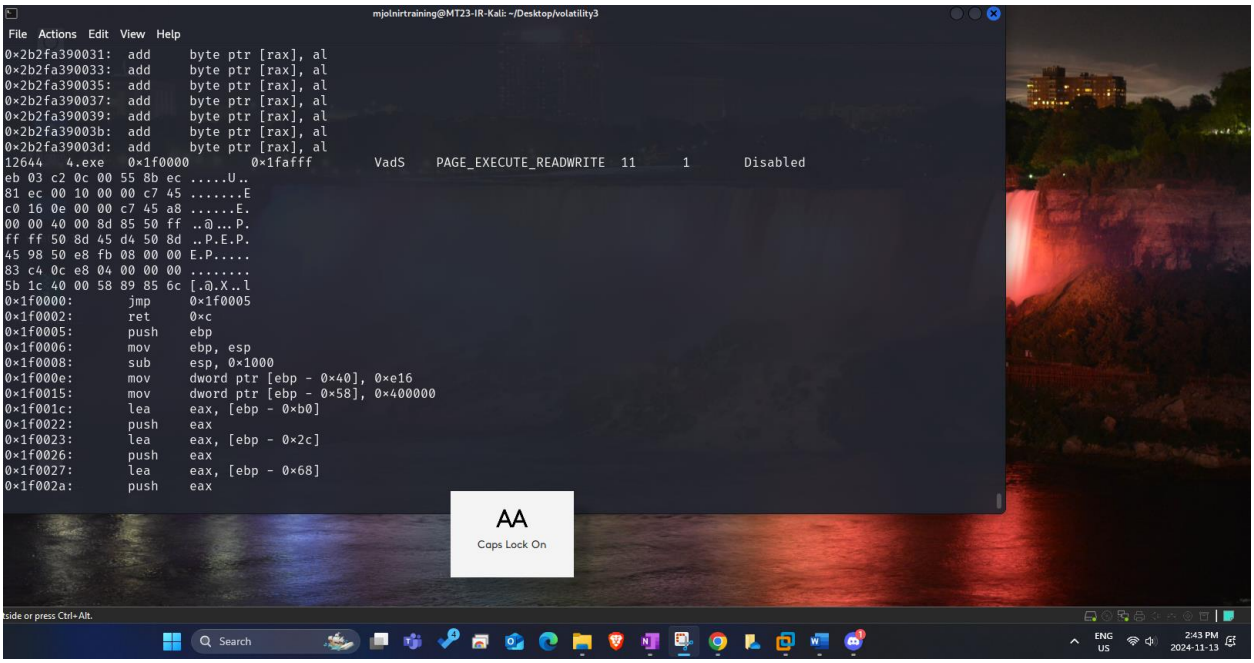


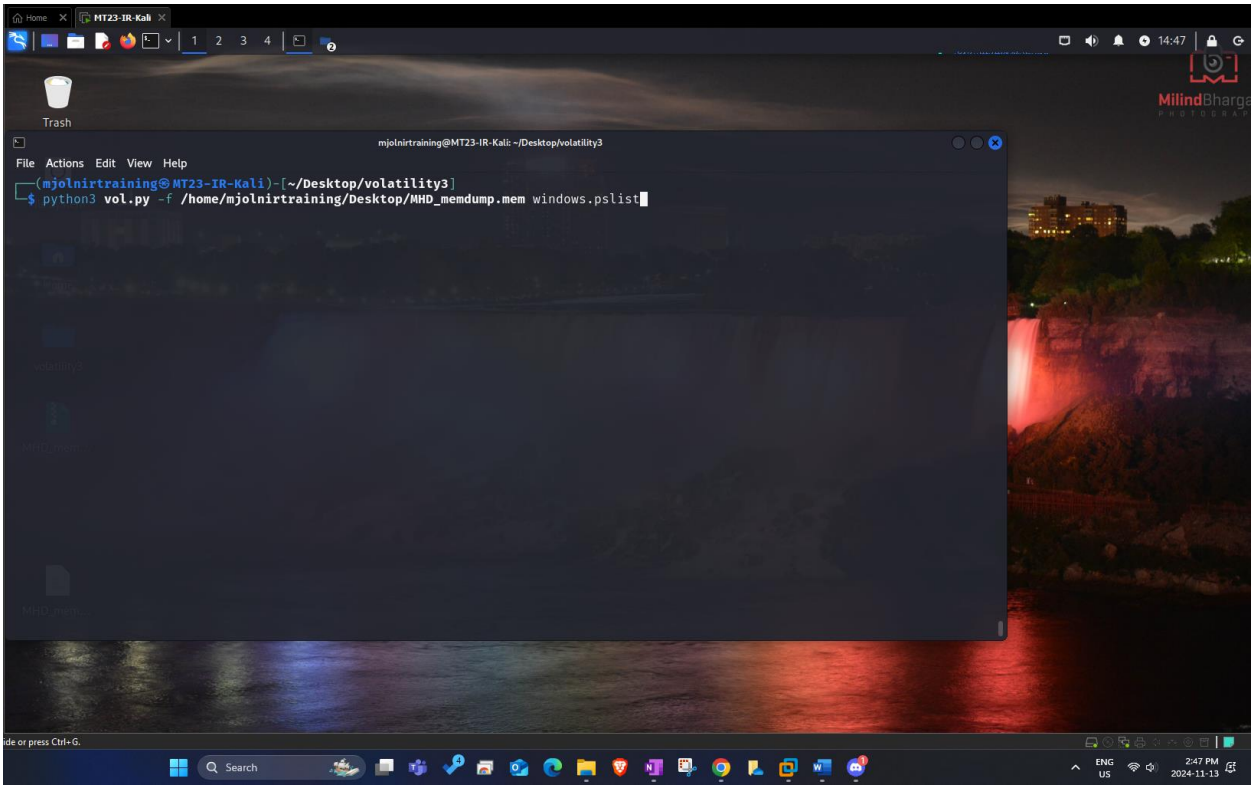




Runs the malfind plugin, which is used to detect injected code in memory

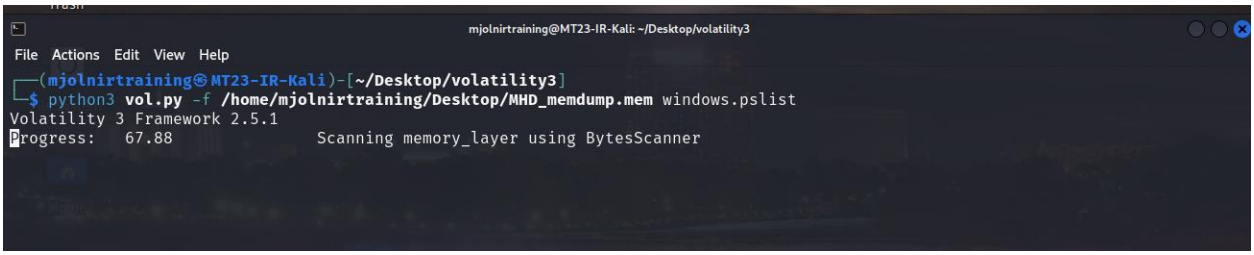


# MOHAMAD ALMASR 101167438



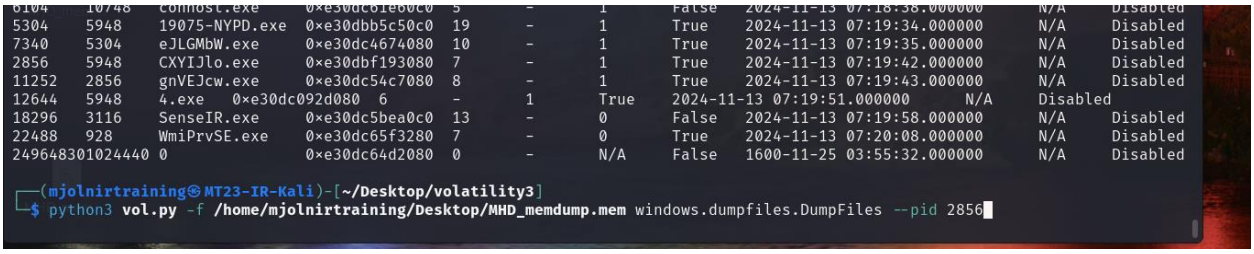



# MOHAMAD ALMASR 101167438

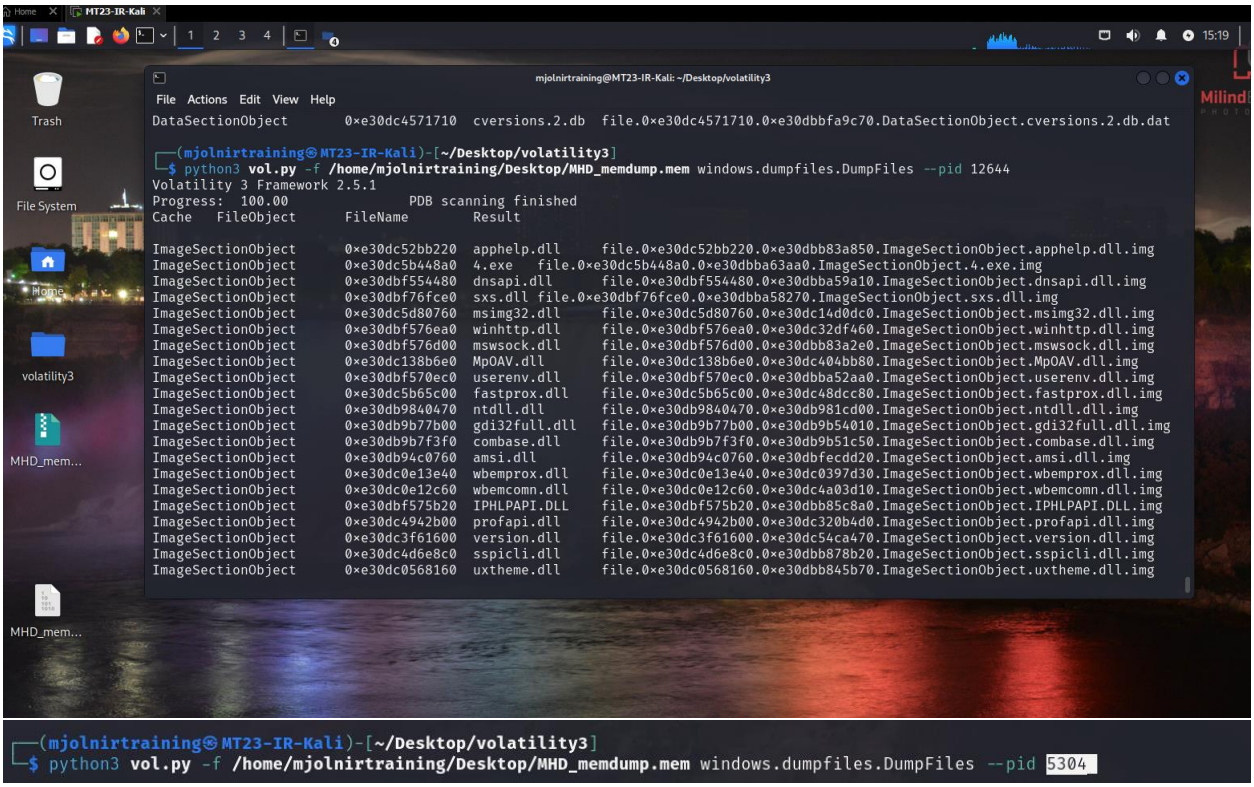



Run Volatility for System Info: Using the Volatility tool, I ran the windows.info command to gather basic information about the system in the memory dump. This gave US a quick overview of the operating system and version.

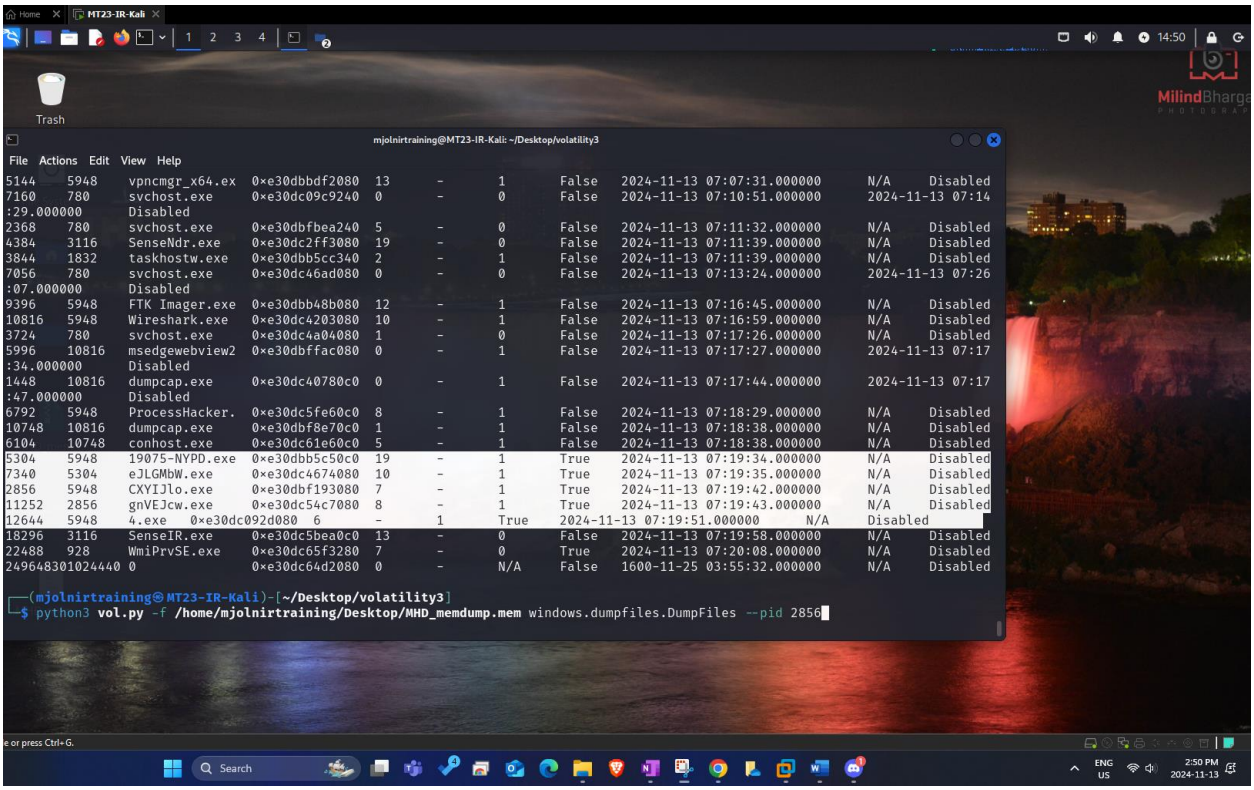
List Processes pslist: Next, I used Volatility's windows.pslist to list all active processes in the memory dump. This helped US decide which process (PID) to analyze.



# MOHAMAD ALMASR 101167438

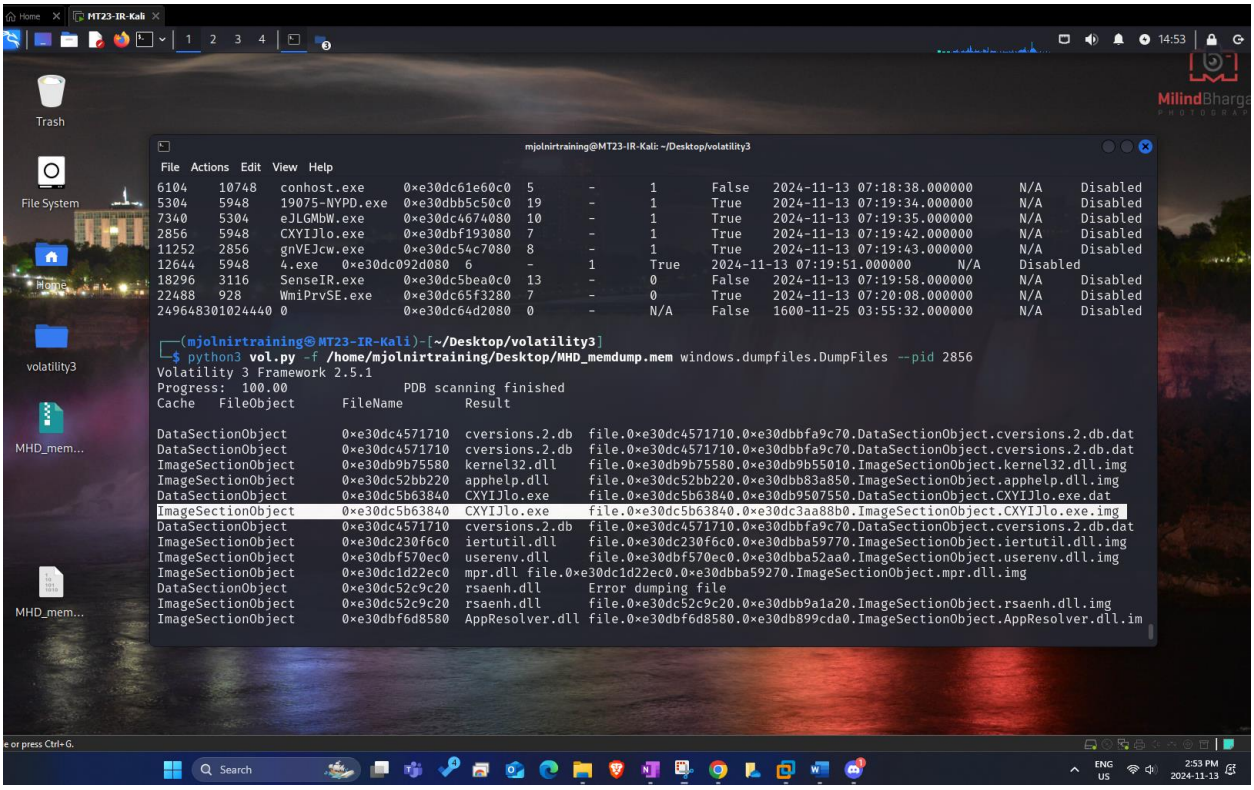



Next, I used Volatility's windows.pslist to list all active processes in the memory dump. This helped US decide which process PID to analyze.



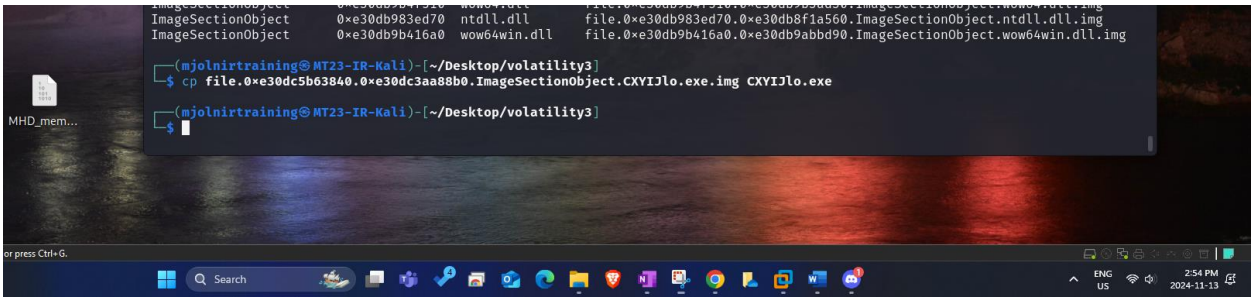
# MOHAMAD ALMASR 101167438

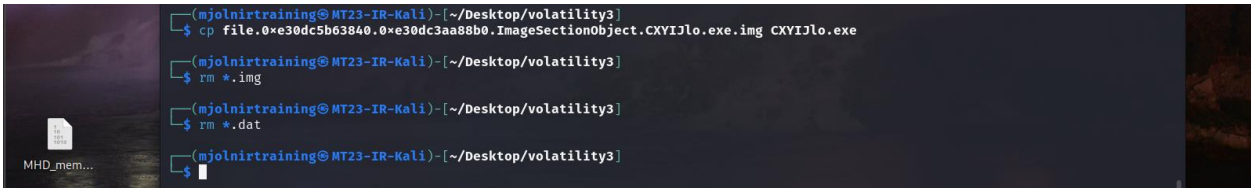
Select and Dump Process: With a specific PID chosen, I used windows.dumpfiles to extract the file associated with that process. This step gave US a copy of the file in memory



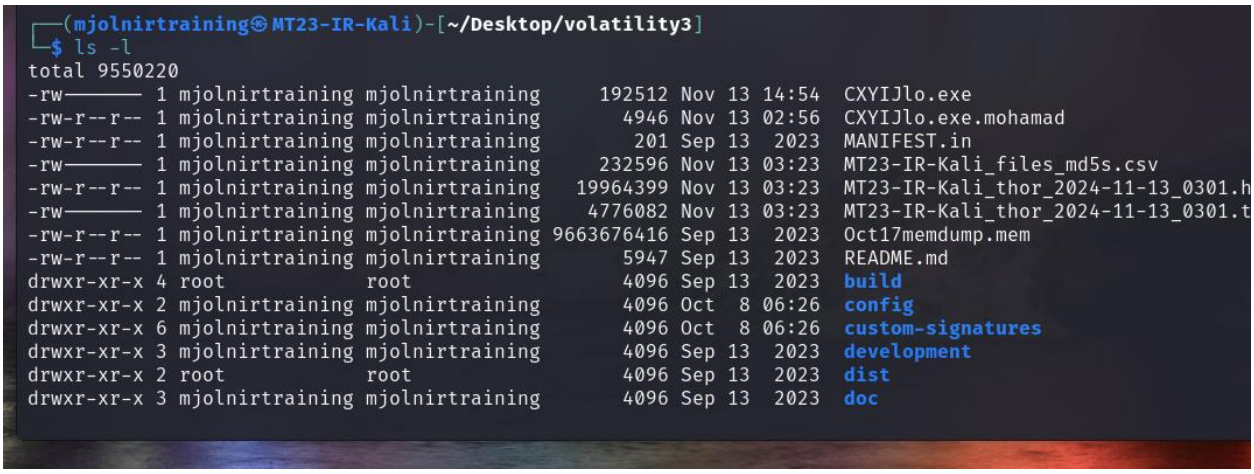

Copy and Extract Dumped File: I copied the dumped file and saved it in a new location. This made it easier to manage and examine the file separately.

# MOHAMAD ALMASR 101167438

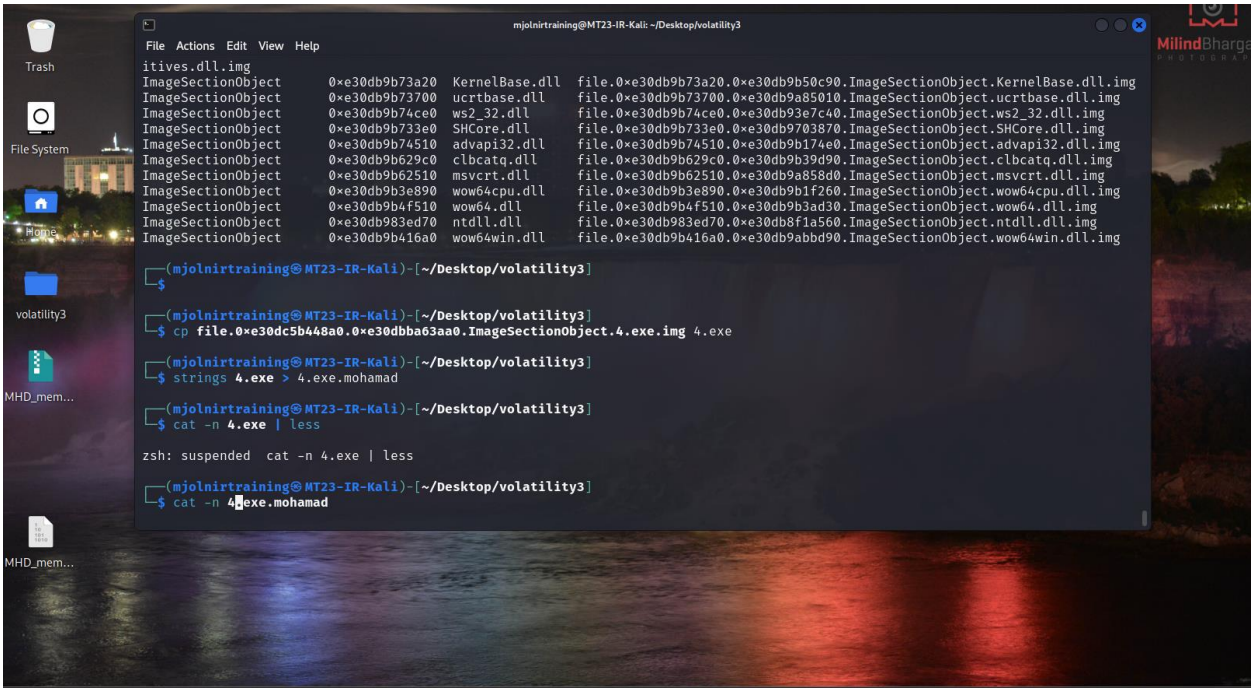






# MOHAMAD ALMASR 101167438




Analyze with Strings Command: To look for readable text within the dumped file, I used the strings command. This helped US identify any suspicious patterns, keywords, or clues within the file.

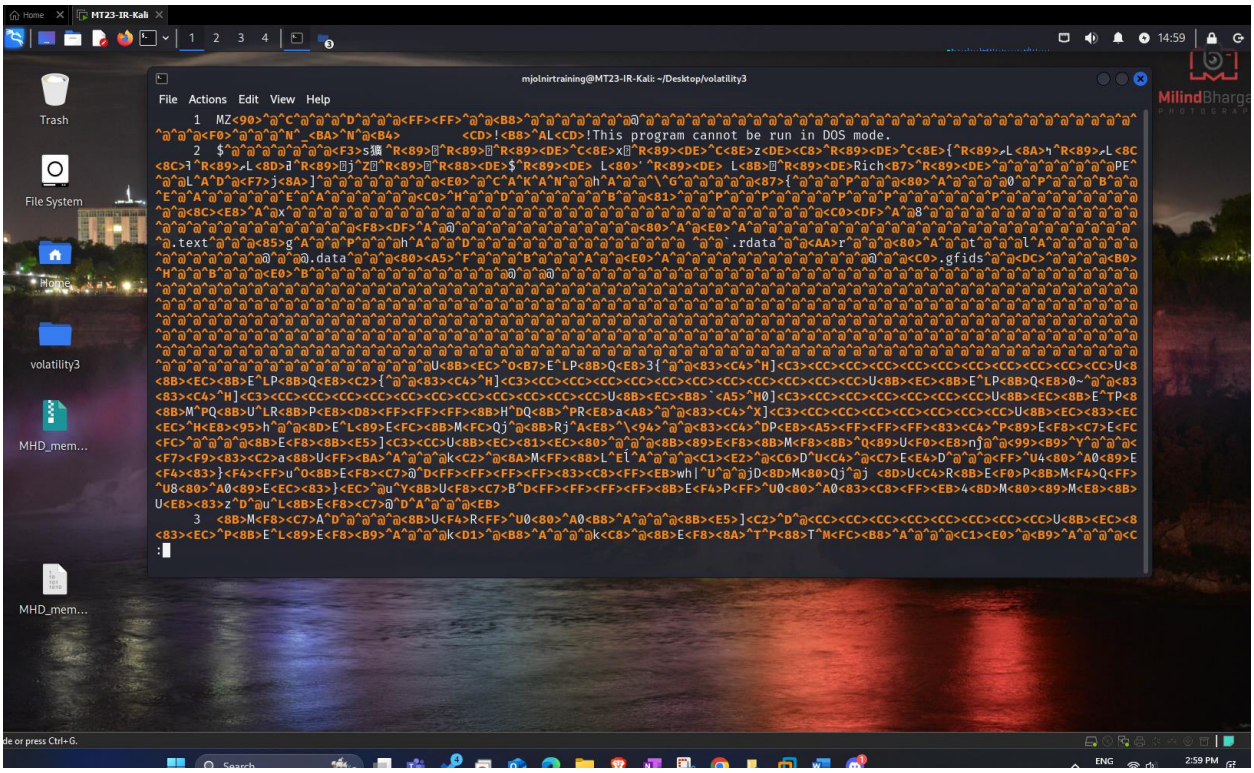
Save Strings Output: I saved the strings output with ny name filename . This way, i could keep track of it and refer back to it later.

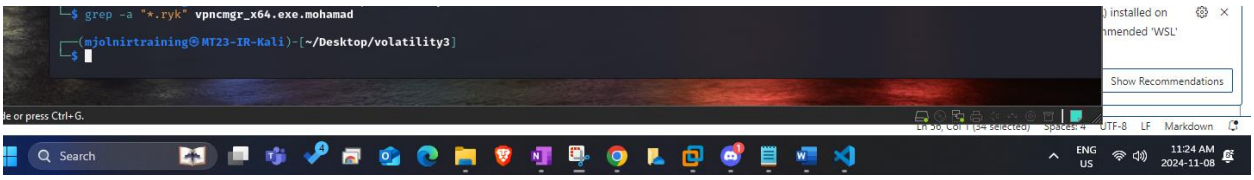
Open File with cat Command: Using cat, i viewed the contents of the saved strings output file. This allowed us to review the extracted text from the dumped file easily.


cat this to chose strings for yara rules

# MOHAMAD ALMASR 101167438

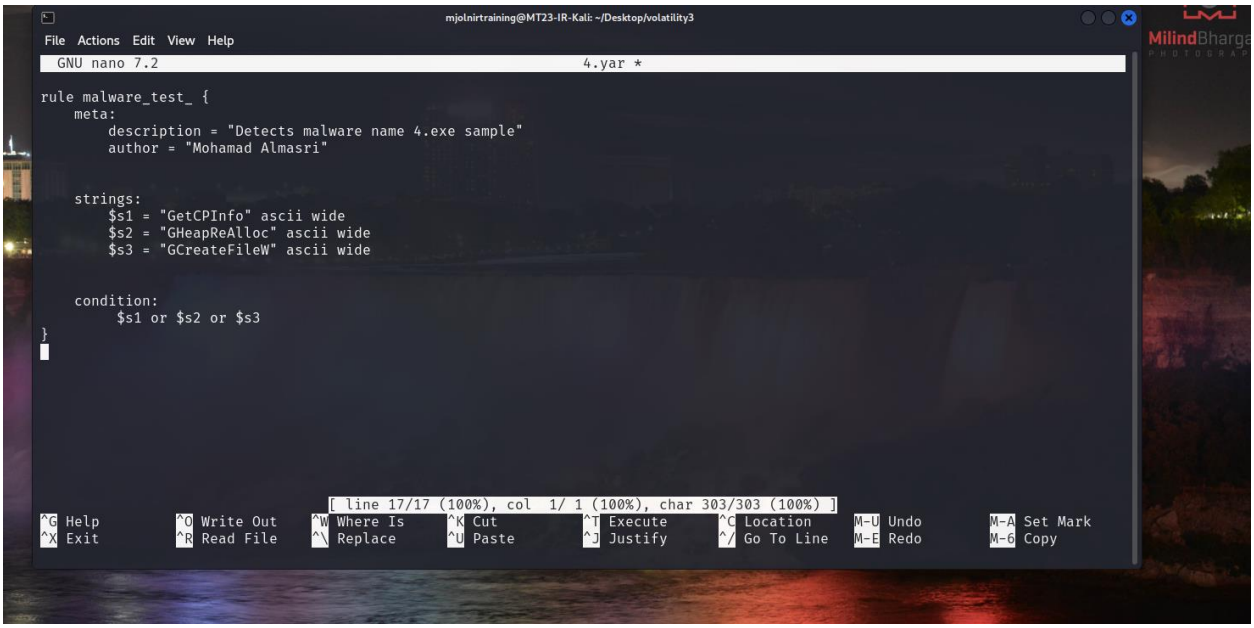
**Review Output with cat**:



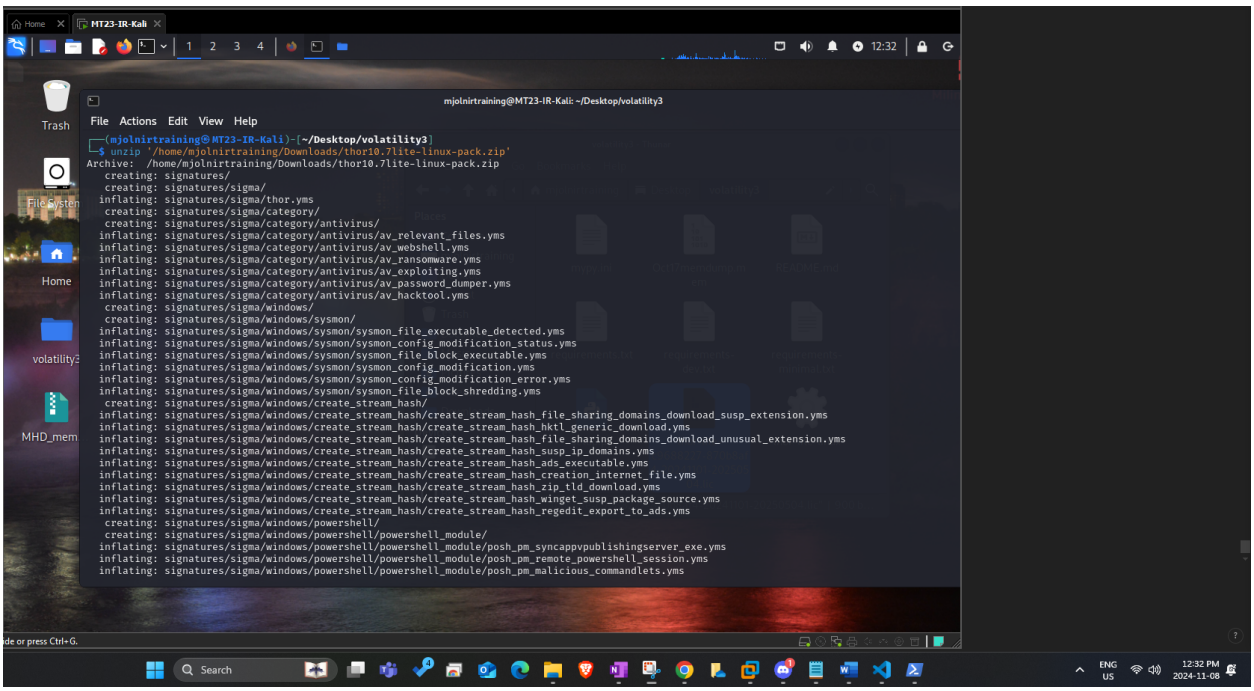

Create YARA Rules: After analyzing the file, I created YARA rules that would help detect patterns or behaviors identified in the memory dump. These rules are tailored to look for specific types of malware.

# MOHAMAD ALMASR 101167438

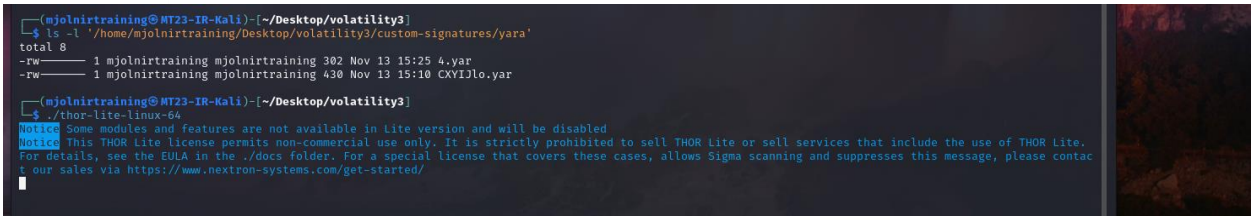



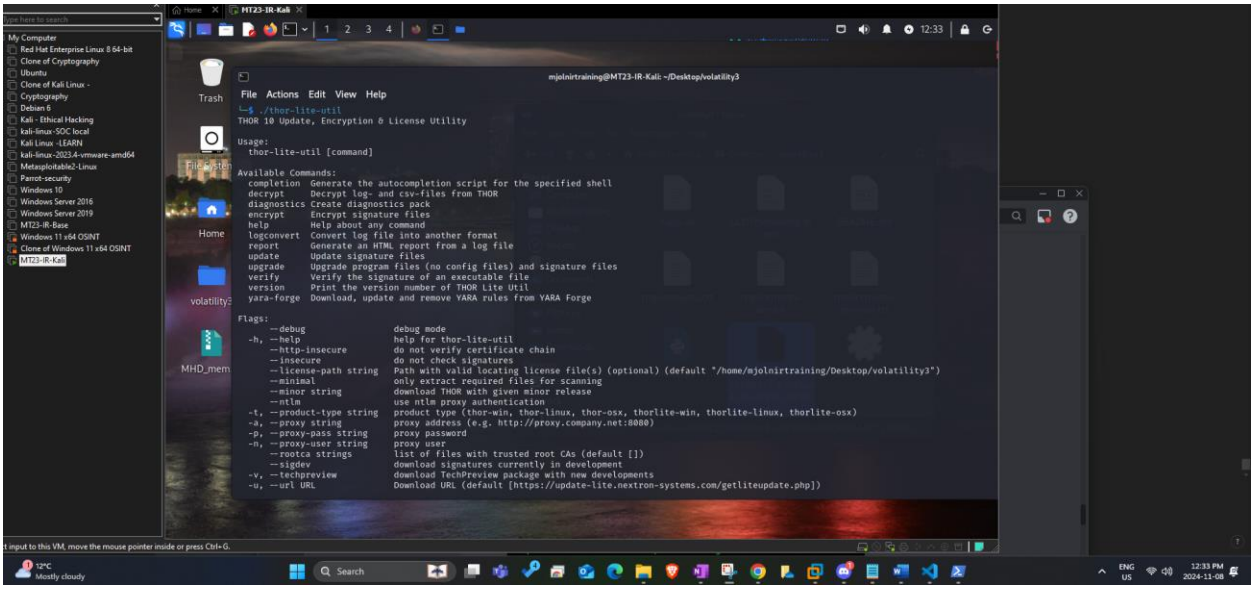
# MOHAMAD ALMASR 101167438

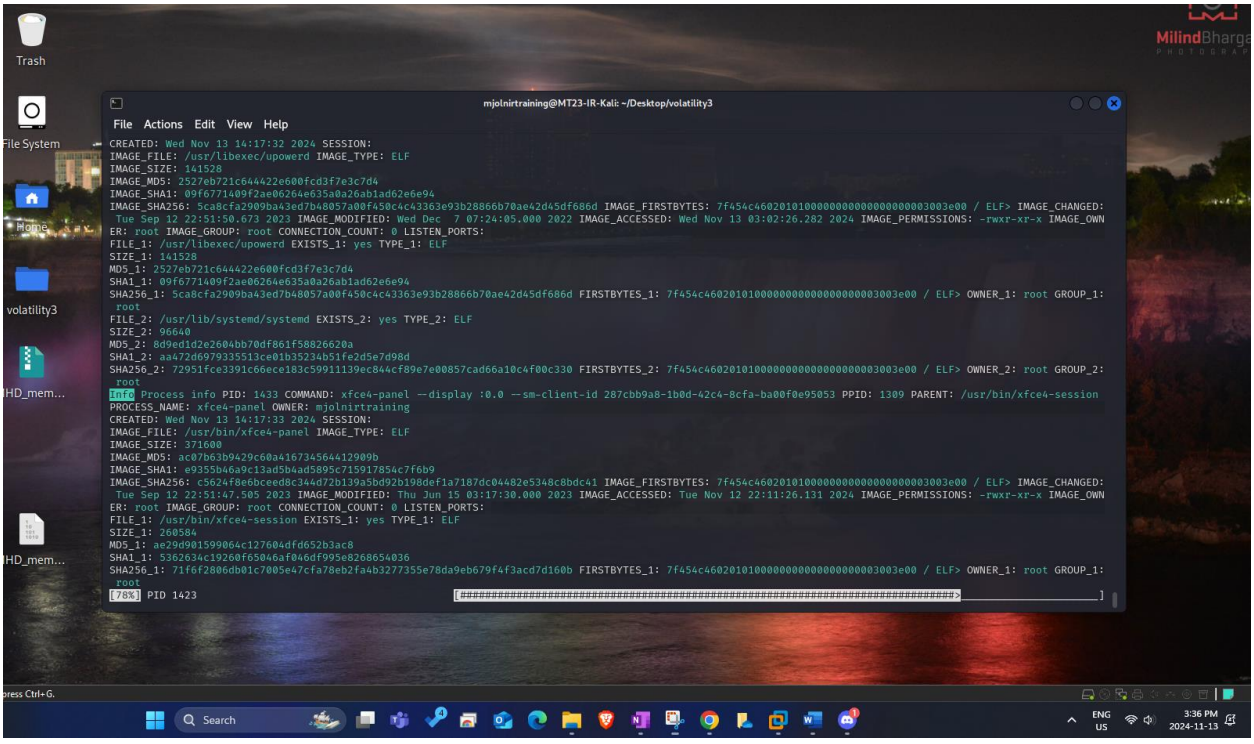

I downloaded THOR Lite a scanning tool which allowed US to use it for checking the memory dump



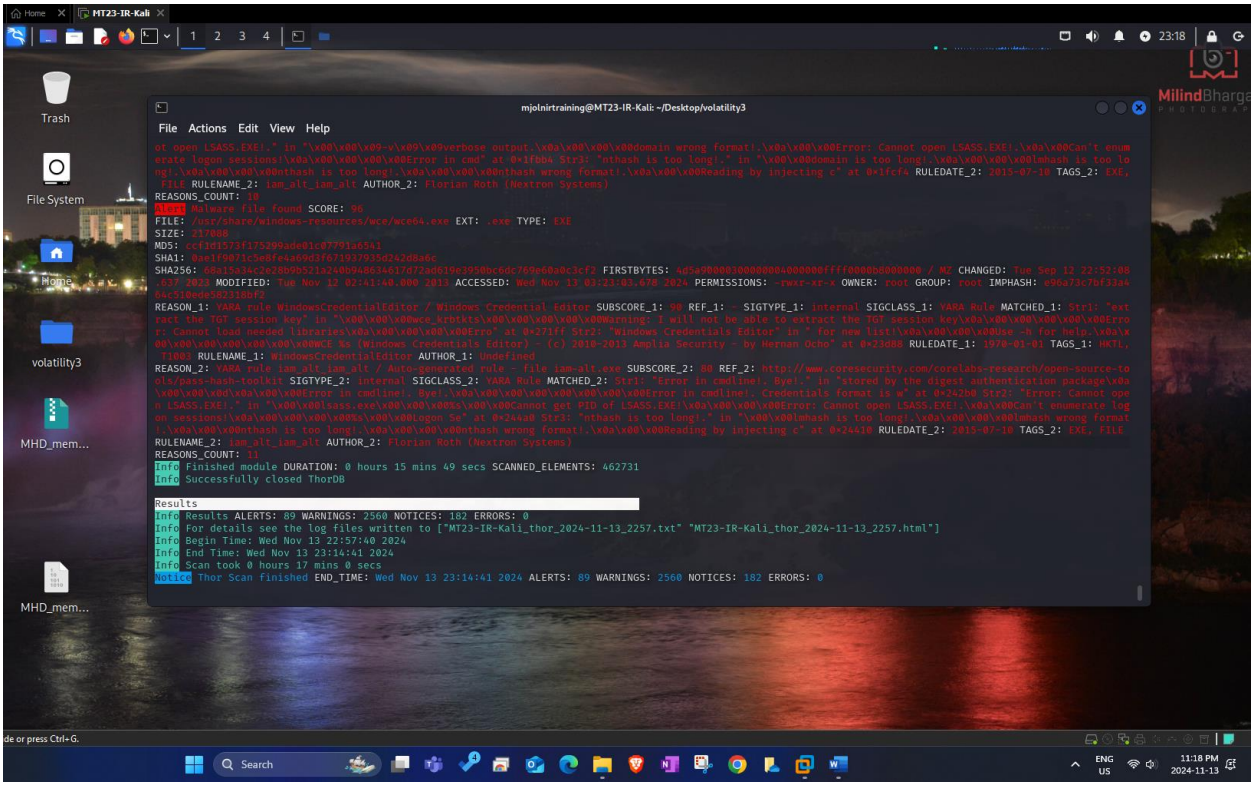
# MOHAMAD ALMASR 101167438

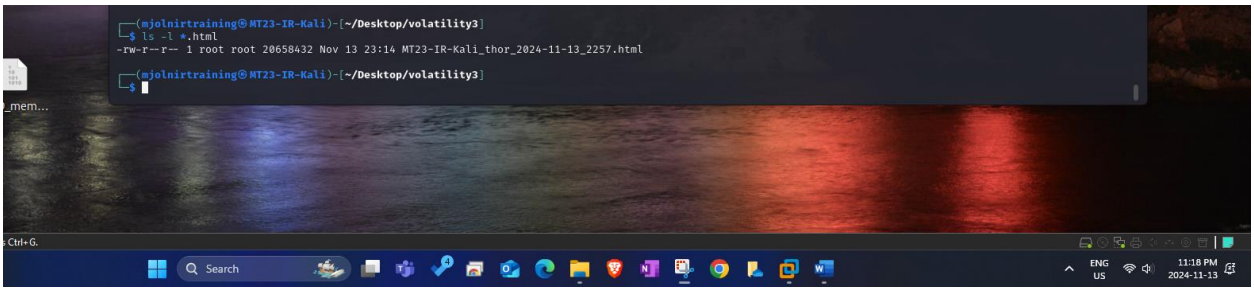


# MOHAMAD ALMASR 101167438



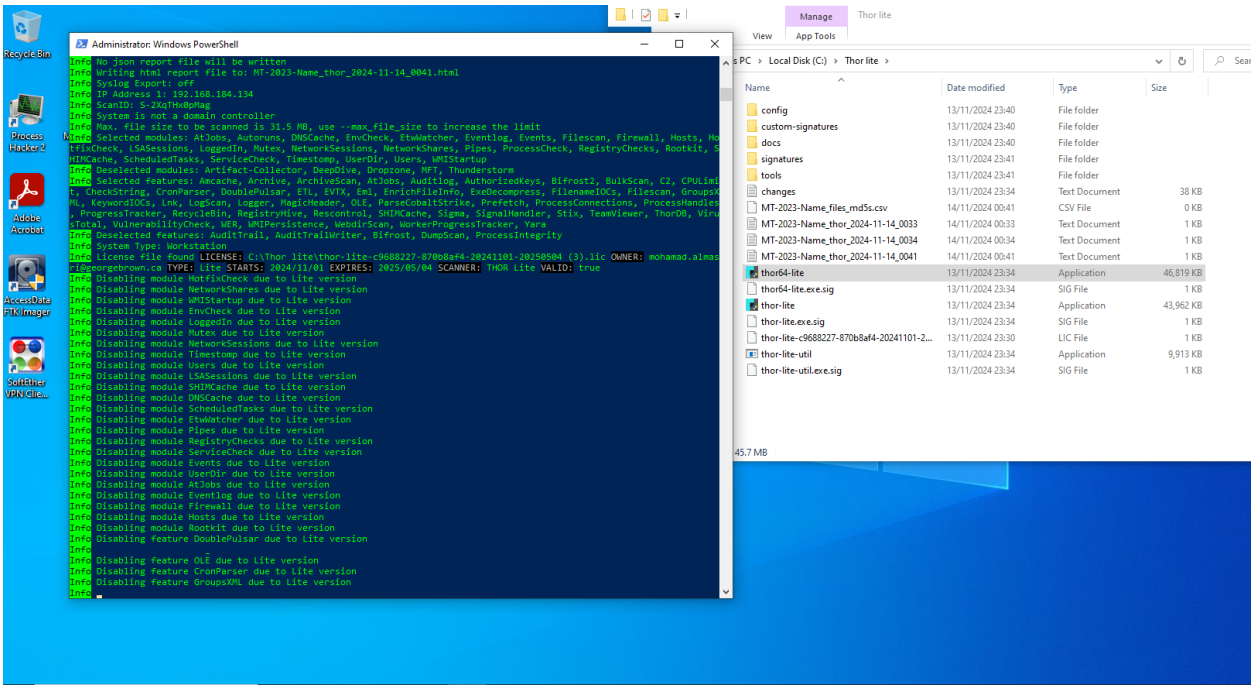

**Run THOR Lite and Generate a Report**: Finally, I ran THOR Lite with thor-lite-linux-64 and generated a report to see any detections.

THOR Lite produced an HTML report showing any detections.

# MOHAMAD ALMASR 101167438

# ON WINDOWS:

**DOWLOAD Thor lite and extract it then copy the license**

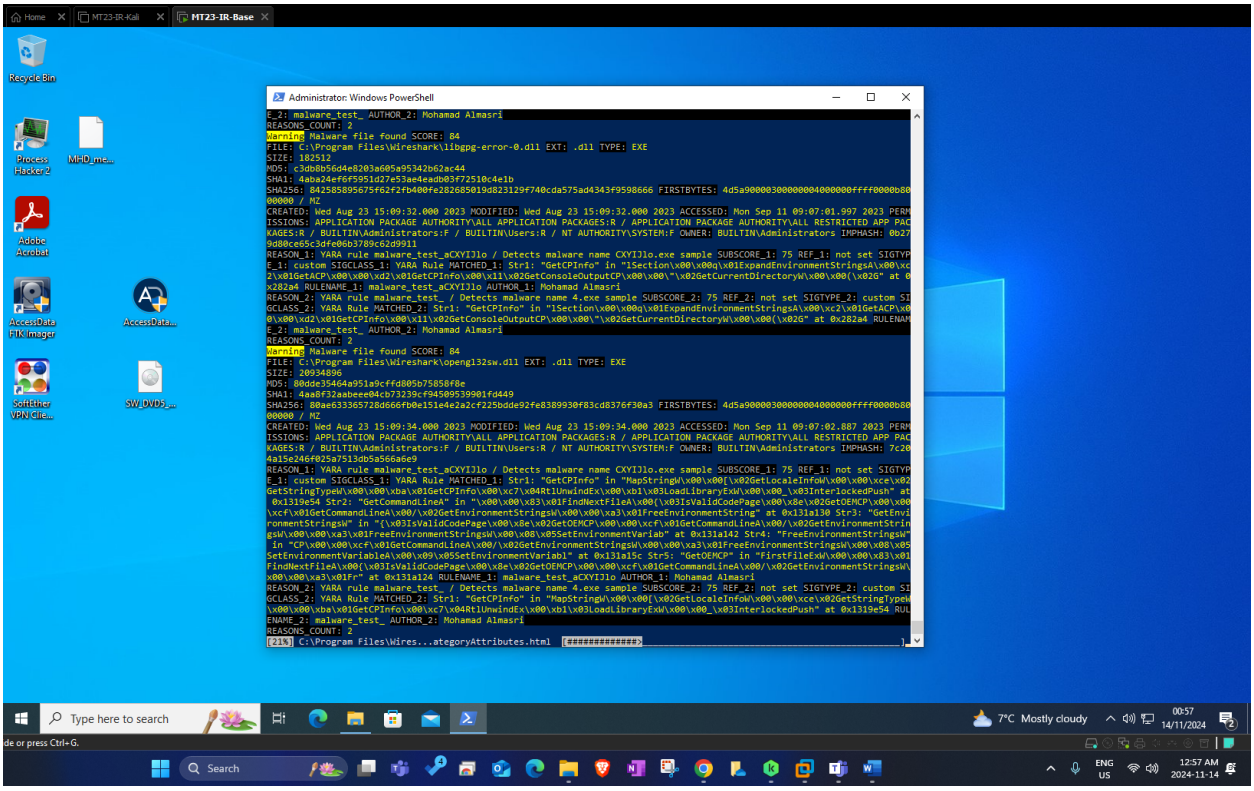


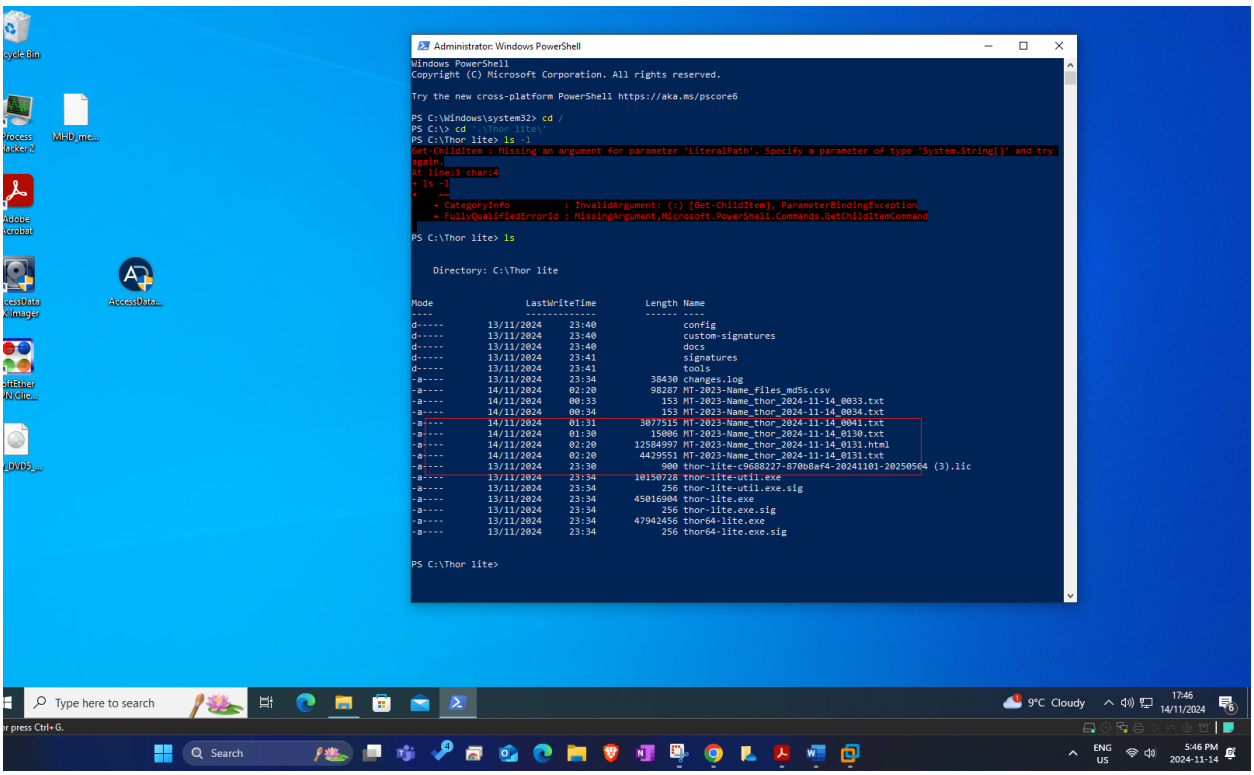

THOR Lite is scanning the system files and registry entries

THOR Lite is applying YARA rules, identifying the files as potentially suspicious, and giving them scores based on the findings.

Detailed information about the file's location, creation date, and matching YARA rule is displayed.

# MOHAMAD ALMASR 101167438



# MOHAMAD ALMASR 101167438

```
-a----        14/11/2024     02:20       12584997 MT-2023-Name_thor_2024-11-14_0131.html
-a----        14/11/2024     02:20        4429551 MT-2023-Name_thor_2024-11-14_0131.txt
```