**Detection of malware from the network traffic.**

**Digital forensics**

**CDAC, Noida**

**CYBER GYAN VIRTUAL INTERNSHIP PROGRAM**

**Submitted By:**

Abinash Dwibedi

Project Trainee, (Oct-Nov) 2024

**BONAFIDE CERTIFICATE**

This is to certify that this project report entitled “Detection of malware from the network traffic”. submitted to CDAC Noida, is a Bonafede record of work done by Abinash Dwibedi under my supervision from 04.10.2024 to 09.11.2023

(Signature) (Signature)

HEAD OF THE DEPARTMENT SUPERVISOR

**Declaration by Author(s)**

This is to declare that this report has been written by me/us. No part of the report is plagiarized from other sources. All information included from other sources have been duly acknowledged. I/We aver that if any part of the report is found to be plagiarized, I/we are shall take full responsibility for it.

Name of Author(S):\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Roll number: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**TABLE OF CONTENTS**

1. Introduction ............................................................ 1

1.1 Problem addressed ............................................... 1

1.1.1 ............................................................................ 3

1.1.2 ............................................................................ 5

1.2 Related literature .................................................. 7

1.2.1 ............................................................................ 7

1.2.2 ............................................................................ 9

1.2.3 ........................................................................... 10

**ACKNOWLEDGEMENT**

**Project Title**

**PROBLEM STATEMENT:**

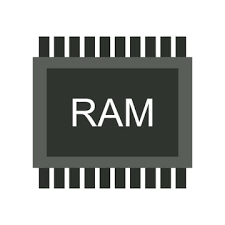
An attacker uses data hiding technique to embed securely a piece of malicious code/secret message/illicit image. Detection of the file with hidden multimedia content its extraction required.

Detection of malicious file hidden within the other file from the network traffic. Detection of the DanaBot malware used for persistence and stealing useful information from the network traffic.

Detection of the PikaBot malware used for ransomware from the network traffic.

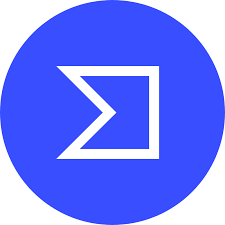
**Learning Objective**

* Detection of stegnographed file using network traffic/forensics tool/open source data hiding tool.
* Detection of malware used for ransomware attack
* Extraction of the hidden data.
* Finding all the traces from the victim machine.

**APPROACH:**

Technology Used :-

1. Wireshark

2. hashmyfiles

3. Virus Total Website

4. ram capturer

5. volatility

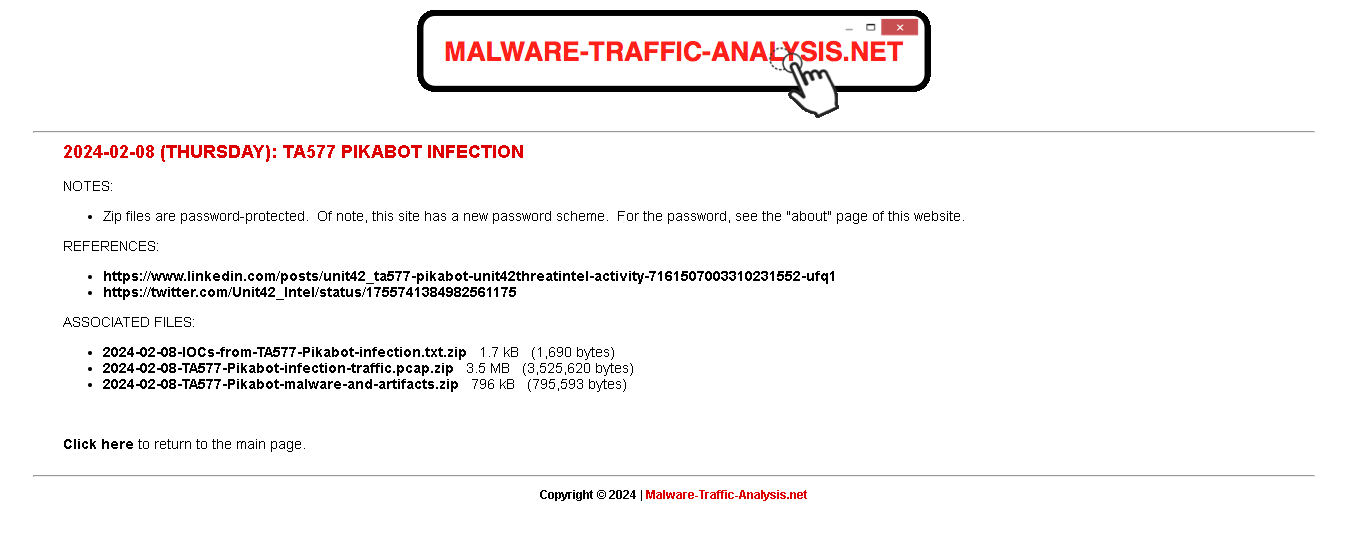
6. steghide

**IMPLEMENTATION:**

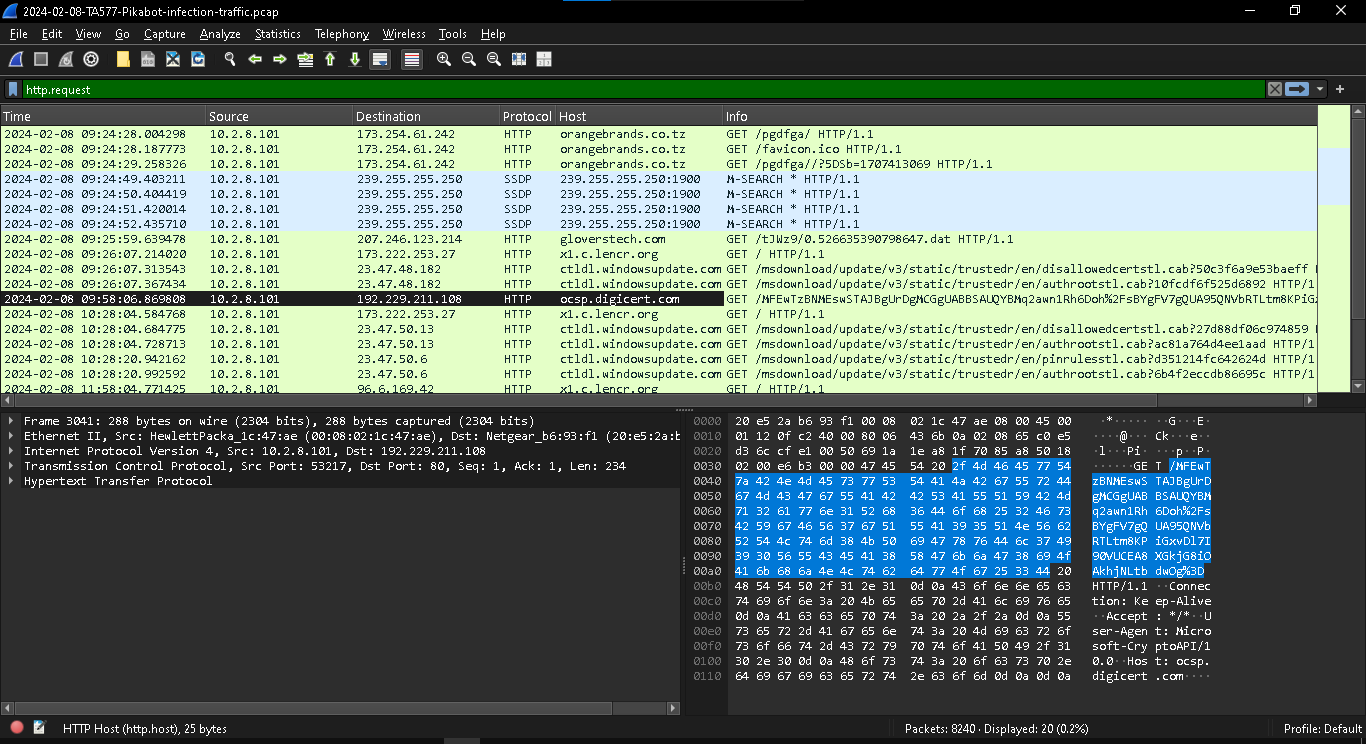
step 1

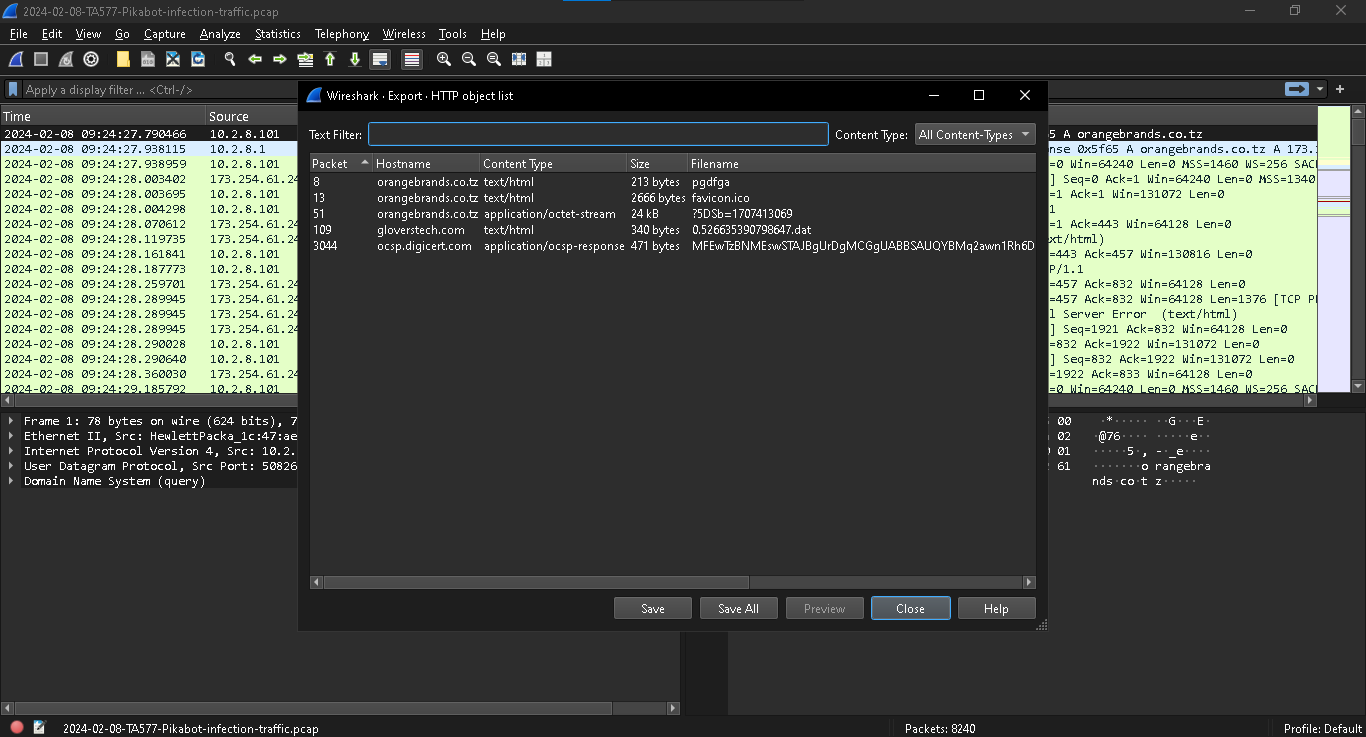
I did the process in 3 steps

I started by downloading a sample PikaBot pcap file from the Malware Traffic Analysis website. This pcap file contains network traffic data, which may include hidden malicious payloads.

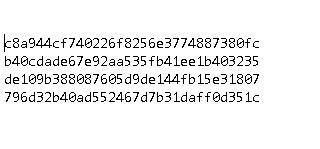
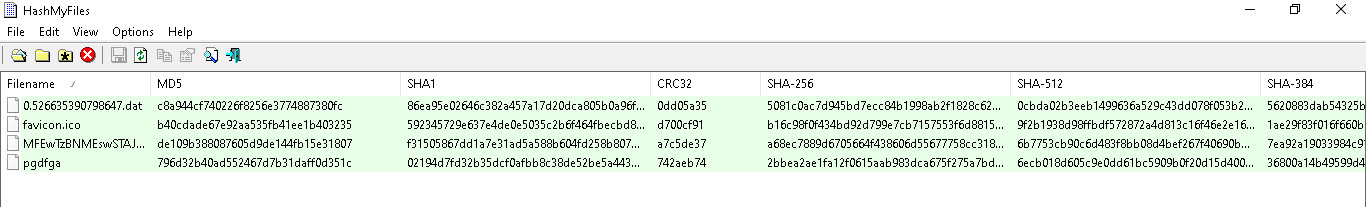


Next, I opened the file in **Wireshark** and navigated to File > Export Objects > HTTP. From there, I extracted all the embedded objects (files) from the traffic and stored them in a designated folder on my local machine. Although **Network Miner** is another tool that could also be used for this task, I opted for Wireshark to keep the process simpler.





To analyze the extracted files for potential malware, I used **HashMyFiles** on a Windows machine to compute cryptographic hashes (MD5, SHA-1, and SHA-256) of each file. This was done to avoid uploading the actual files directly to VirusTotal, which could have resulted in the leakage of confidential data. I then uploaded the individual file hashes to **VirusTotal** to check if any of the files were flagged as malicious.

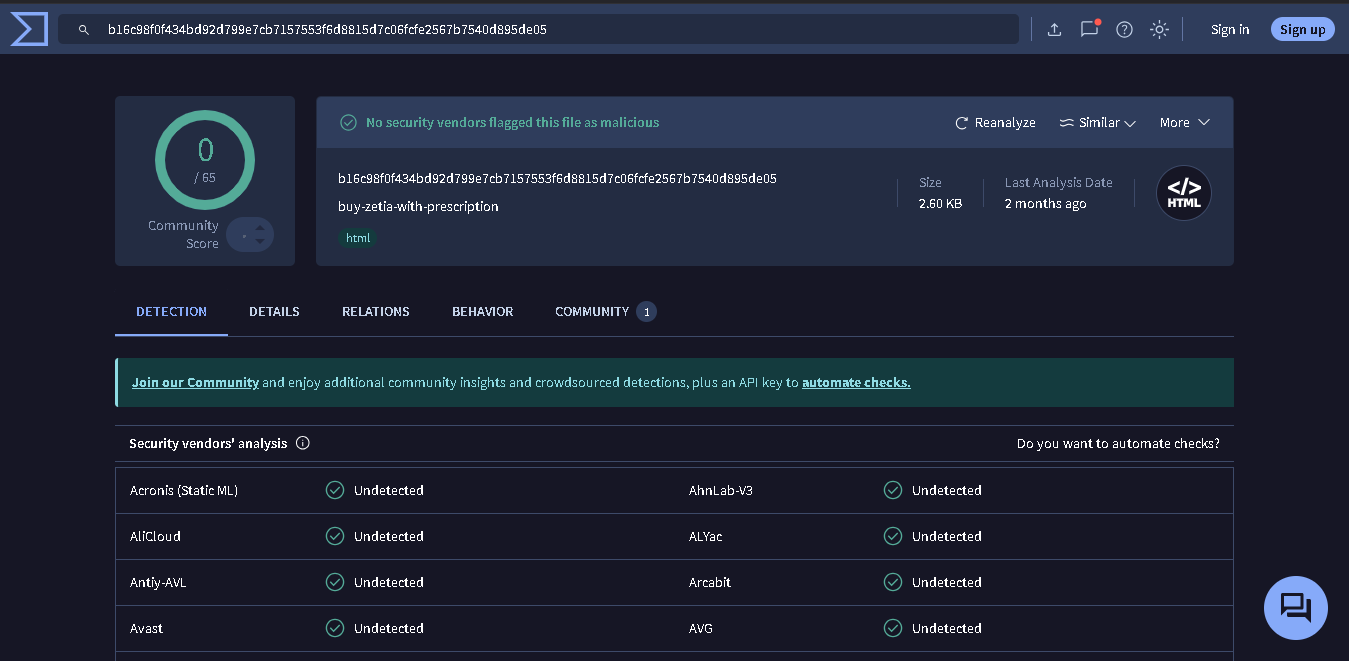
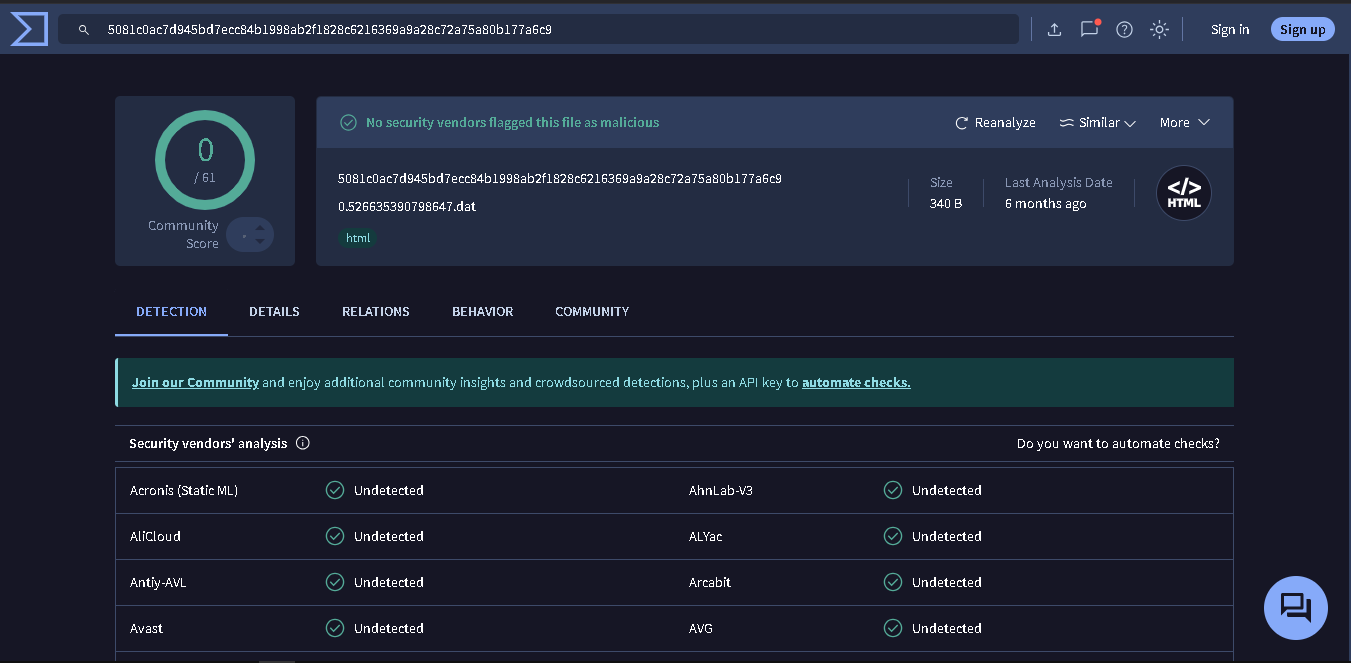


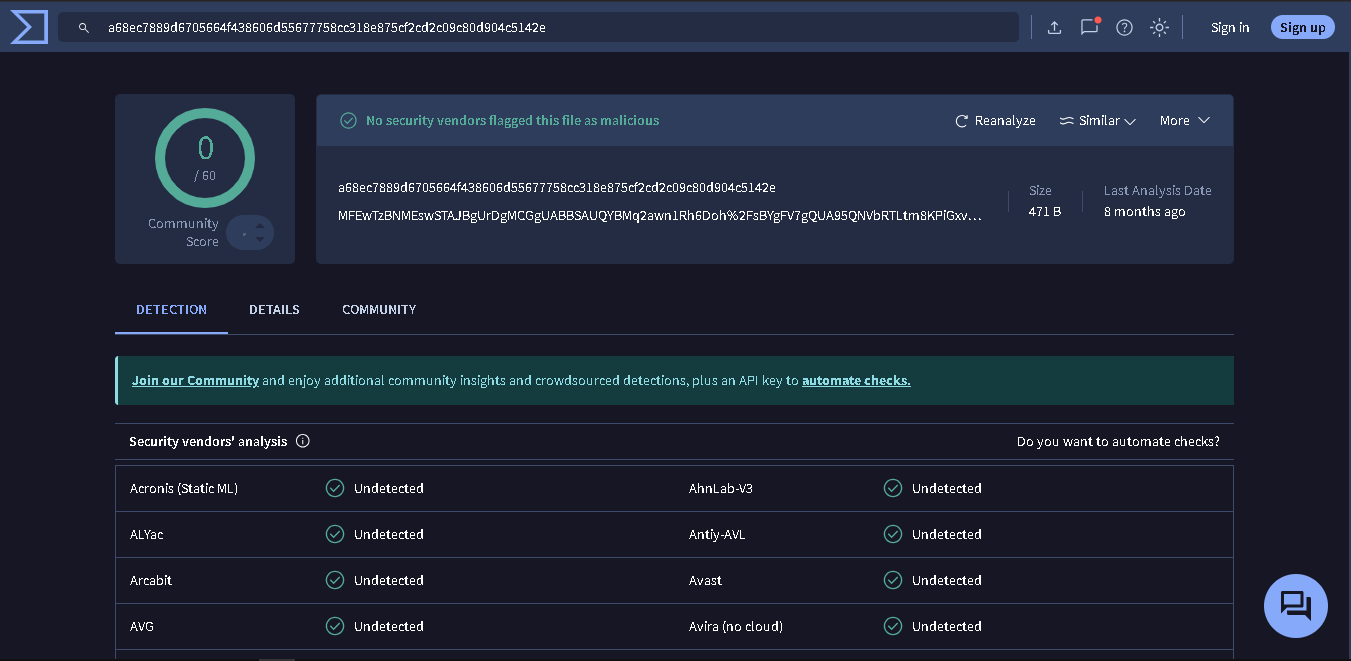
one file (796d32b40ad552467d7b31daff0d351c) was identified as infected, I returned to Wireshark and examined the network traffic to identify the source of the malicious file. I focused on tracking the IP address from which the infected file originated, recording the details for further analysis.

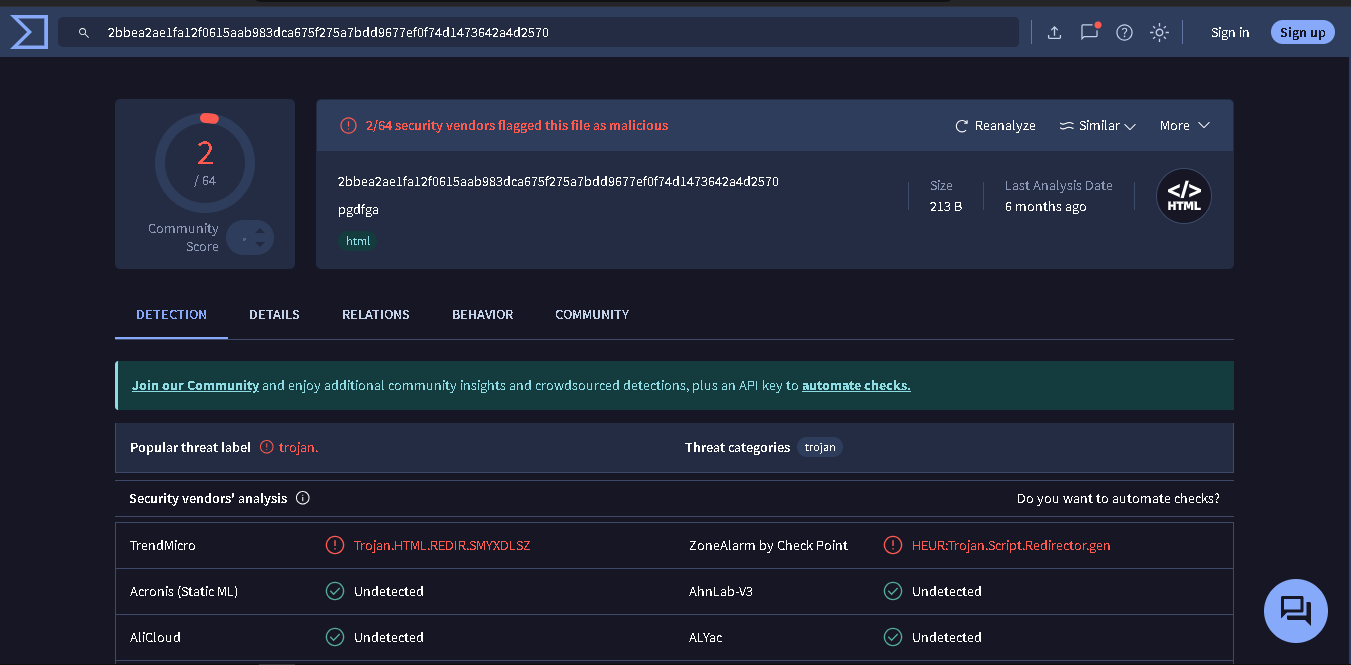
infected site url - orangebrands.co.tz

infected site ip - 173.254.61.242

infected machine ip – 10.2.8.101



****

****

step 2

In this step, I performed **steganographic analysis** to simulate the process of detecting hidden data within an image file. Initially, I was unable to find any hidden files within the existing data, so I decided to create a scenario where I could embed a text file within an image.

To do this, I used **Steghide**, a popular tool for steganography, which allows hiding data inside various file types such as images or audio files.

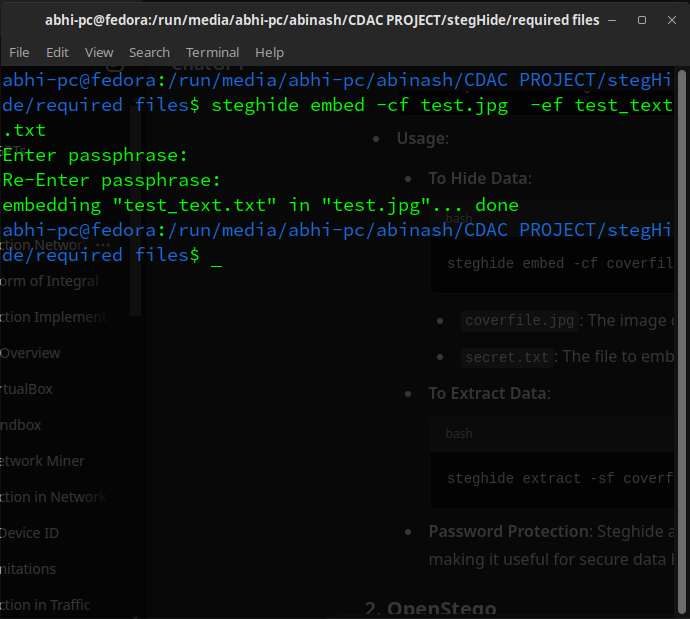
I executed the following command to hide a text file (test\_text.txt) inside an image file (test.jpg):

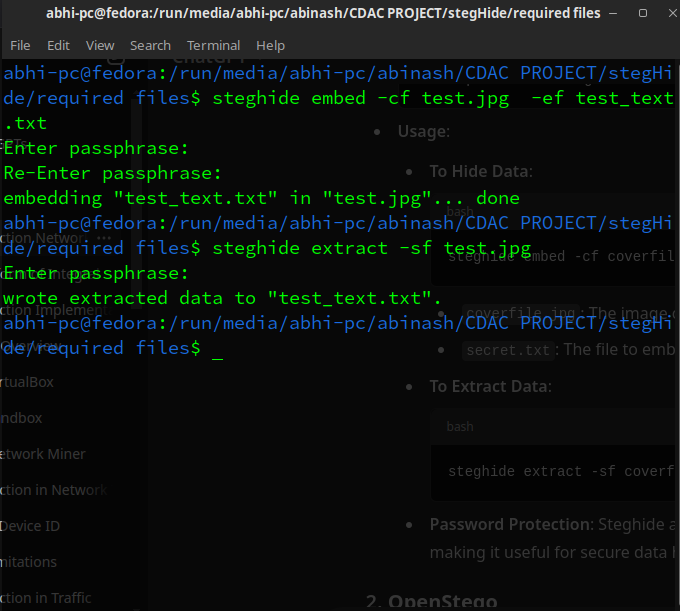
→ steghide embed -cf test.jpg -ef test\_text.txt

This command embeds the content of test\_text.txt within the test.jpg image, making it imperceptible to the naked eye, but retrievable using the same tool.

To extract the hidden text file, I then used the following command:

→ steghide extract -sf test.jpg

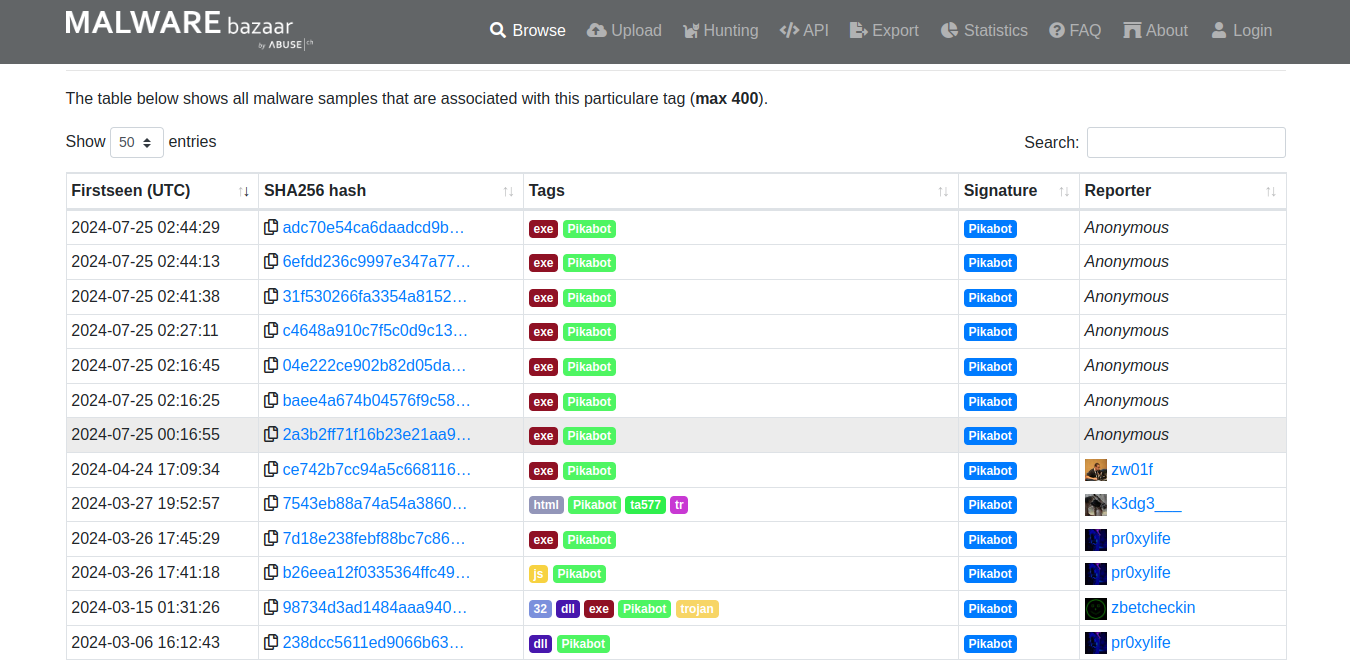
****

****

step 3

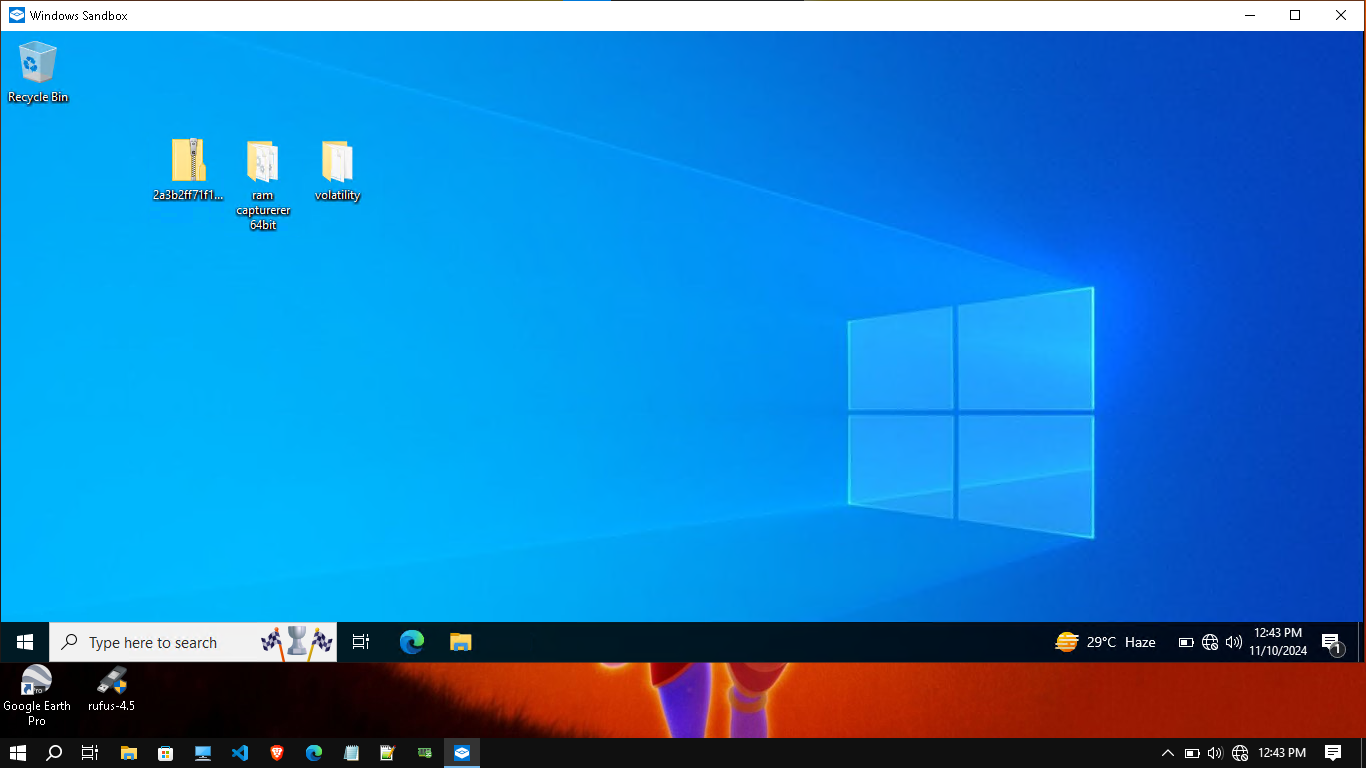
For this step, I focused on analyzing the behavior of the **PikaBot** malware sample in a controlled environment using memory analysis techniques.

I started by downloading the **PikaBot executable** (pikabot.exe) from **MalwareBazaar**, a repository of known malware samples.

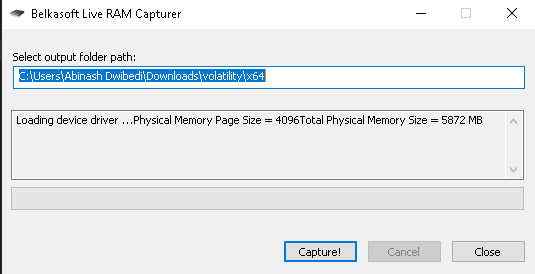
****

Next, I created a secure environment using **Windows Sandbox** to run the malware. This isolated environment ensures that any malicious actions by the malware do not affect the host machine.

I copied the necessary tools into the sandbox, including **Volatility** (for memory analysis) and **RAM Capturer** (for capturing memory dumps).



Before running the PikaBot executable, I launched RAM Capturer to capture the system's initial memory state (pre-execution). I saved the memory dump from this initial state for comparison later.



Then, I executed the PikaBot executable inside the sandbox. After running the malware, I once again used RAM Capturer to capture the memory state, storing the second dump (post-execution).

Finally, I opened the memory dumps in Volatility to analyze the differences between the memory states before and after running the PikaBot malware. This comparison allowed me to observe any changes in system behavior, such as new processes, network connections, or modifications to the memory, which could indicate malicious activity.

**CONCLUSION & RECOMMENDATIONS:**

In this project, I successfully demonstrated techniques for detecting malware, specifically **PikaBot**, from network traffic. The key steps included extracting files from network traffic, performing steganographic analysis to detect hidden data, and conducting memory analysis to observe the behavior of the malware.

**LIST OF REFERENCES:**

Malware traffic analysis :- <https://youtu.be/3t1BNAavrlQ?si=vhTI8LVYWH6Me8Te>

Malware bazar :- https://bazaar.abuse.ch/

Malware traffic analysis :- https://www.malware-traffic-analysis.net/2024/02/08/index.html

Wireshark Documentation :- https://www.wireshark.org/docs/

Steghide :- https://github.com/StegHigh/steghide

VirusTotal :- https://www.virustotal.com/gui/