Task Level (Beginner)

Q1. Find all the ports that are open on the website http://testphp.vulnweb.com/

Introduction

This report will look closely at the open ports on the website http://testphp.vulnweb.com/. By finding out which ports are open, we can get a clearer picture of how secure the website is. This information is very valuable because it helps us understand how a website can be attacked and what measures can be taken to protect it.

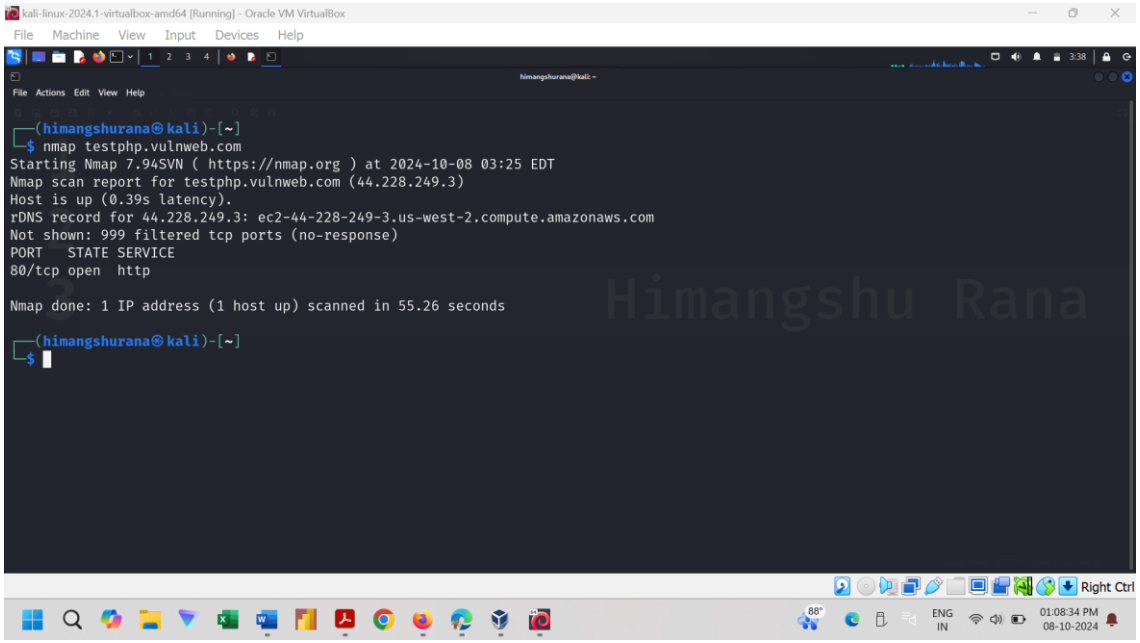
Tool and Technologies Used

For this project, we selected **Nmap** (Network Mapper) as our primary tool. Nmap is a powerful open-source tool used for network discovery and security auditing. It is widely used by network administrators and security professionals for several reasons:

- ☑ Versatility: Nmap can scan a range of IP addresses and perform various types of scans.
- ☑ Open Source: Being free to use, Nmap has a large community that regularly contributes to its development.
- ☑ Detailed Output: It provides comprehensive information about open ports and the services associated with them.

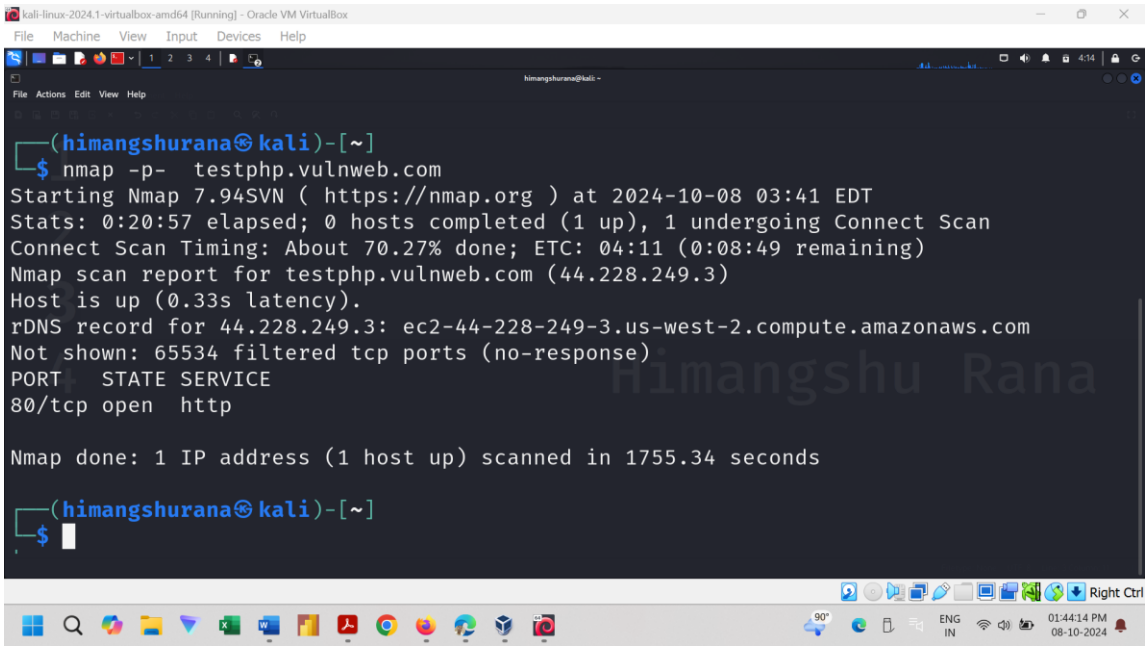
Steps to Reproduce

Step 1: Basic Scanning



Explanation: Performs a default scan on the target host, checking the most common 1,000 ports and providing basic information about any open ports.

Step 2: Scan All Ports



Explanation: Scans all 65,535 TCP ports. This is useful for finding less common open ports.

Results and Findings

In this report, a scan was conducted on the domain testphp.vulnweb.com to identify open ports. The domain is hosted on an Amazon AWS server, as indicated by the reverse DNS record, which shows the resolved IP address to be 44.228.249.3.

The scan was performed on a Kali Linux virtual machine, using the commands nmap testphp.vulnweb.com & nmap -p- testphp.vulnweb.com, which instructs Nmap to scan for all 65,535 TCP ports. The scan provides an overview of which ports are open, the services running on them, and any other significant information.

The Nmap scan revealed about the Open Ports:

- ☑ **Open Port**:
  - **Port 80 (HTTP)**: The only open port detected is port 80, which is used for HTTP services. This implies that the server is hosting a web service that communicates over the HTTP protocol, accessible via a standard web browser.

Severity

An open port 80 can be a security risk, scoring medium to high in vulnerability assessments, usually around 5 to 7 out of 10. It allows HTTP traffic, which attackers might exploit for unauthorised access or data theft. Securing it can reduce exposure to cyber threats effectively.

Impact

- ☑ **Lack of Encryption (HTTP Only)**: Port 80 uses HTTP, which doesn't encrypt data. This means that any information shared between the user and the website, like login details or personal information, could be intercepted by attackers.
- ☑ **Exposed Web Application**: The web service running on port 80 could potentially be vulnerable to various web-based attacks, including:
  - ☞ **Cross-Site Scripting (XSS)**
  - ☞ **SQL Injection**
  - ☞ **Cross-Site Request Forgery (CSRF)**

Mitigation Steps

To improve server security and reduce the risks of an open port 80, follow these recommended steps:

- **Switch to HTTPS**: HTTPS encrypts data sent between users and the server, keeping information private and more secure.
- **Conduct Web Application Security Testing**: Since port 80 is open, perform a thorough security test of the web application, including:
  - Penetration testing to identify any weak points.
  - Checking for vulnerabilities like cross-site scripting (XSS), SQL injection, and cross-site request forgery (CSRF).

Fix any issues found to ensure the web app is secure against attacks.

- **Review Firewall Rules Regularly**: Carry out routine audits of firewall settings to:
  - Ensure only necessary ports are open.
  - Confirm that closed ports are securely managed.

This limits entry points for attackers and reduces the chances of unauthorized access.

References

1. Nmap Documentation: https://nmap.org/docs.html
2. Common Vulnerabilities and Exposures (CVE): https://cve.mitre.org/cve/search_cve_list.html

Q2. Brute force the website http://testphp.vulnweb.com/ and find the directories that are present in the website.

Introduction

In this report, explore the process of identifying and brute-forcing hidden directories on the website http://testphp.vulnweb.com using the tool Gobuster. The primary objective of this project is to demonstrate how attackers can utilise directory brute-forcing techniques to discover unsecured or sensitive paths on a web server, which could potentially lead to information leakage or other security vulnerabilities.

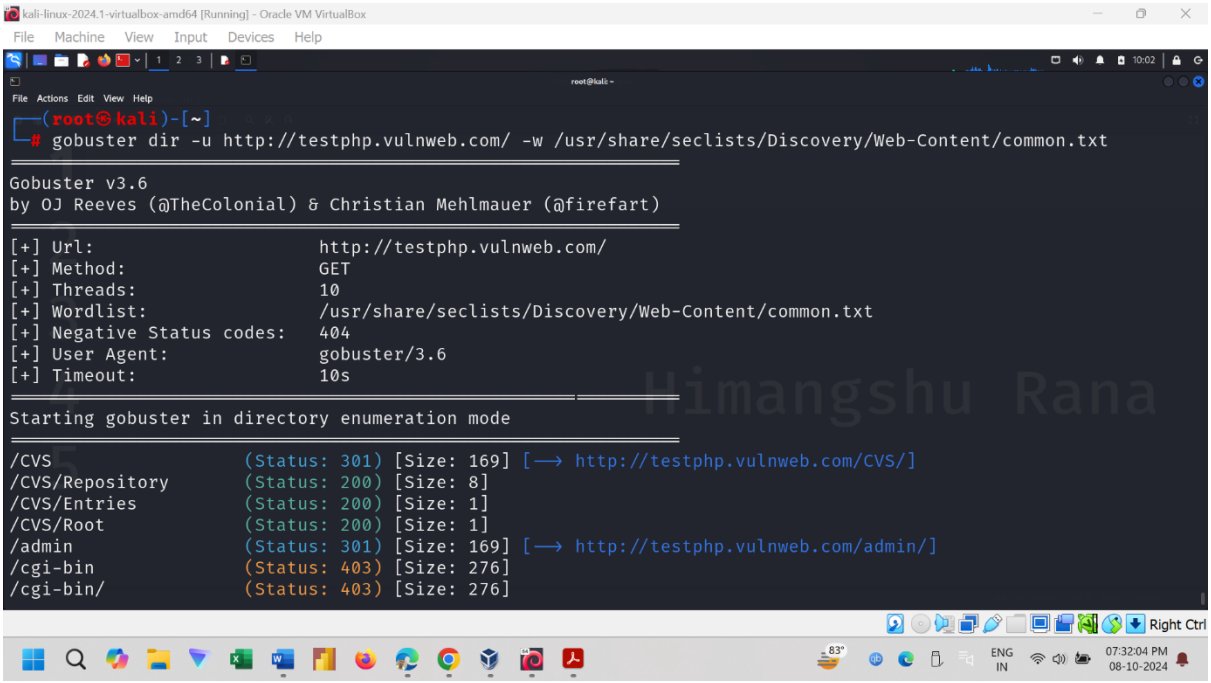
Technologies and Tools Used

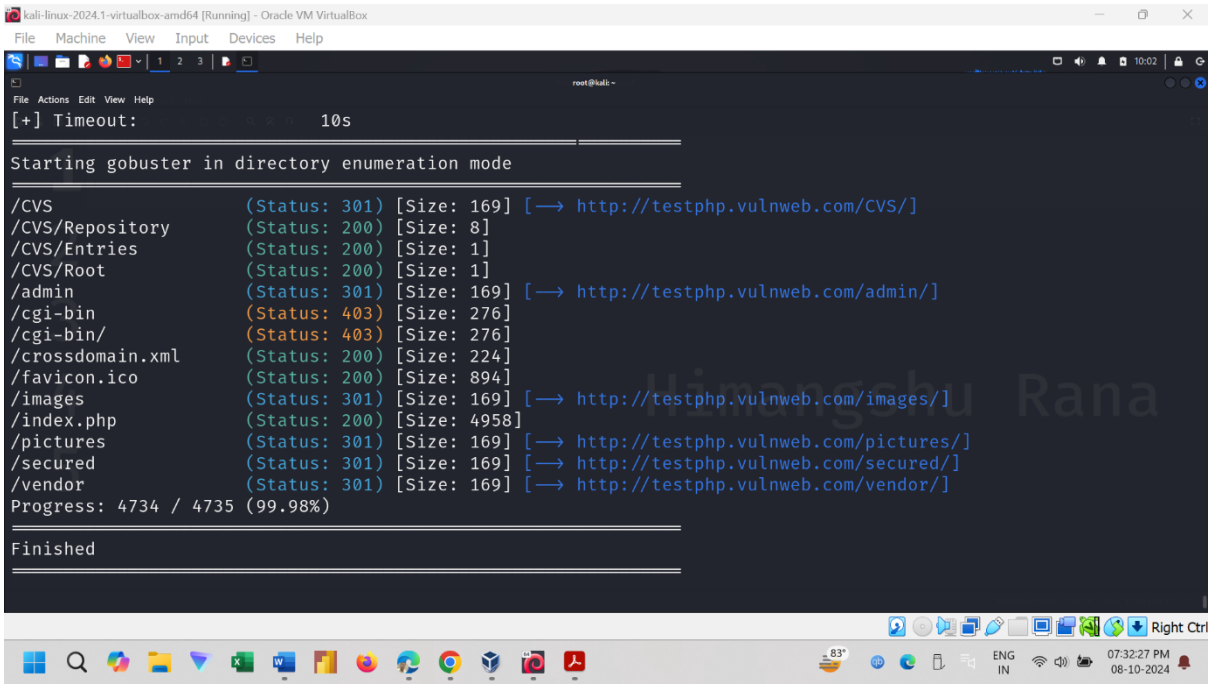
Brute forcing is a method of systematically guessing possible inputs such as passwords, directory names, or hidden file names until the correct one is found. When applied to directory discovery, brute-forcing entails sending multiple HTTP requests to the web server with different directory paths and checking the server's response to determine whether a directory exists

For this project, we used,

- ☑ **Gobuster:** Gobuster is an open-source tool widely used for brute-force attacks on web servers. It is capable of directory and file brute-forcing by sending HTTP requests and analysing the response status codes. These responses indicate whether a directory or file exists, or if it is restricted.
- ☑ **Wordlist (SecLists):** Gobuster requires a wordlist to test various directory and file names. We used a commonly used wordlist from SecLists, a compilation of useful wordlists for security testing.

Steps to Reproduce





Explanation: Gobuster initiated the process by sending HTTP requests for each directory name listed in the wordlist.

Analysis of Results

The Gobuster scan revealed several critical directories and files on testphp.vulnweb.com. While many of the directories serve routine purposes (such as images or CSS files), there are several sensitive directories and files that warrant further investigation.

- **CVS Directories**: The version control directories (**/CVS/** and its subdirectories) could be a significant security risk. These files might provide insight into the website's codebase and structure. If accessible, these directories could expose code vulnerabilities, misconfigurations, or outdated information that attackers could exploit.
- **Admin Directory**: The **/admin/** directory indicates the presence of an administrative interface. While the **301** status indicates a redirect, further testing should be performed to check for default credentials, weak login systems, or bypass techniques.
- **CGI-BIN Directory**: Although the **/cgi-bin/** directory is restricted, it should be regularly monitored for unauthorised access attempts. CGI scripts can be a vector for remote code execution attacks if not properly configured or secured.
- **Crossdomain.xml**: While not immediately concerning, the **crossdomain.xml** file should be reviewed to ensure that the website’s cross-domain policies are appropriately restrictive.

Severity

The severity of this attack is considered **high** because the exposure of hidden directories can lead to significant vulnerabilities. Critical directories, such as /admin/ or /backup/, if discovered, may expose the site's administrative login page, sensitive files, or backup data that could be exploited for further attacks.

- **Severity Level**: High
- **CVSS Score**: 7.5 (High)

Impact

☑ **Exposed Administration Pages**: Discovery of directories such as /admin/ can give attackers access to login pages for site management, providing an opportunity to attempt credential stuffing or brute force login attacks.

☑ **Sensitive Information Exposure**: Files or directories containing sensitive information such as database credentials, backup files, or configuration files might be exposed. For instance, /CVS/ and /crossdomain.xml/ were discovered, which could expose sensitive content.

☑ **Further Exploitation**: Once directories are identified, an attacker could investigate further for vulnerable scripts or misconfigurations that can be exploited. This could lead to remote code execution, data theft, or service disruption.

Mitigation Steps

To prevent brute force directory attacks, the following mitigation measures are recommended:

1. **Disable Directory Listing**: Ensure that directory listing is disabled on the web server. Attackers should not be able to see the contents of a directory. This can be done by modifying the server's configuration file.
2. **Restrict Access to Sensitive Directories**: Sensitive directories such as /admin/ and /cgi-bin/ should be restricted to authorised users only. Implement access control measures such as IP whitelisting, VPN access, or multi-factor authentication (MFA).
3. **Use Security Headers**: Implement HTTP security headers such as X-Frame-Options, X-XSS-Protection, Content-Security-Policy, and X-Content-Type-Options. These headers can provide additional protection against attacks by preventing browsers from misinterpreting or mishandling responses from the server.
4. **Implement Rate Limiting**: Rate limiting helps to prevent brute force attacks by limiting the number of requests that can be made to the server from a single IP address within a specific timeframe. This can be implemented at the server or application level to detect and mitigate brute force attempts.
5. **Utilise Web Application Firewalls (WAFs)**: A WAF can be deployed to detect and block suspicious activity, including directory enumeration and brute force attempts. WAFs can monitor requests and block those that match known attack patterns.

6. **Encrypt Sensitive Directories (HTTPS)**: Ensure that all sensitive directories and data transmissions occur over HTTPS. This will prevent attackers from intercepting traffic and gaining access to sensitive information.
7. **Security Testing and Auditing**: Regularly perform security audits and penetration testing on the website to uncover potential vulnerabilities before attackers can exploit them. Tools such as Gobuster should be used by the security team to check for any misconfigurations.

References

- ☑ OWASP. "Directory Indexing." Available at: https://owasp.org/www-project-top-ten/
- ☑ Gobuster Tool. GitHub: https://github.com/OJ/gobuster
- ☑ Apache Documentation on Directory Listing: https://httpd.apache.org/docs/2.4/mod/core.html#options
- ☑ Common.txt Wordlist from SecLists: https://github.com/danielmiessler/SecLists

Q3. Make a login in the website http://testphp.vulnweb.com/ and intercept the network traffic using wireshark and find the credentials that were transferred through the network.

Introduction

The objective of this project is to explore how network traffic interception can expose sensitive information like login credentials. By making a login attempt on a vulnerable website, http://testphp.vulnweb.com, and capturing the data with Wireshark, we examine how unencrypted credentials can be harvested. The focus is on identifying vulnerabilities associated with insecure transmission protocols and highlighting potential security risks.

Tools and Technologies Used

- ☑ **Wireshark**: Wireshark is a free and open-source network protocol analyser. It captures and analyses network traffic in real-time, making it an ideal tool for identifying potential security vulnerabilities, such as credential interception.
- ☑ **Kali Linux**: We used Kali Linux as the operating system for running Wireshark. Kali Linux is a penetration testing platform equipped with numerous security tools, including Wireshark.
- ☑ **Browser (Firefox)**: The login process was performed through a web browser, simulating a real user accessing the website.

Attack Overview

Attack Name: Man-in-the-Middle (MITM) Attack involving Network Traffic Interception.

In this scenario, a user's login information is sent to the website without encryption, which allows an attacker to intercept and capture sensitive data such as usernames and passwords. The attack targets websites that transmit data over HTTP, a non-secure communication protocol. Wireshark is employed to capture network traffic, allowing the attacker to analyse the data and extract credentials being sent from the client to the server.

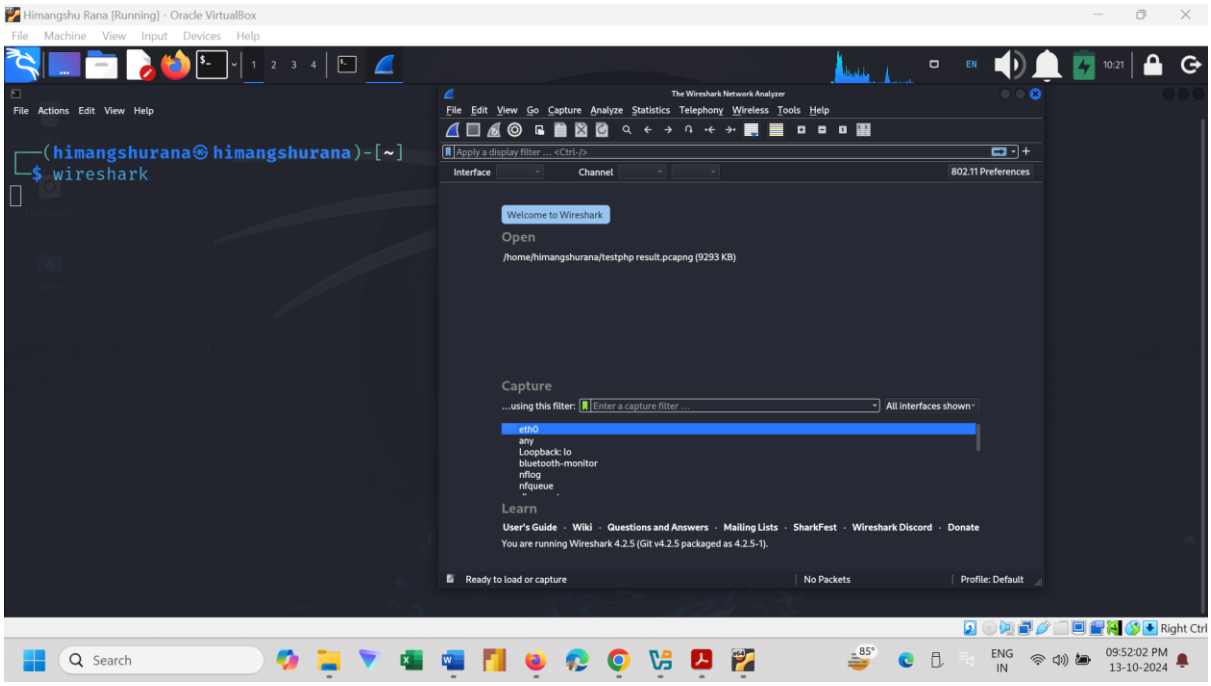
Severity

This vulnerability occurs because the website transmits sensitive information, such as login credentials, over an unencrypted channel (HTTP). This means the data can be easily intercepted by attackers, making it a serious security issue.

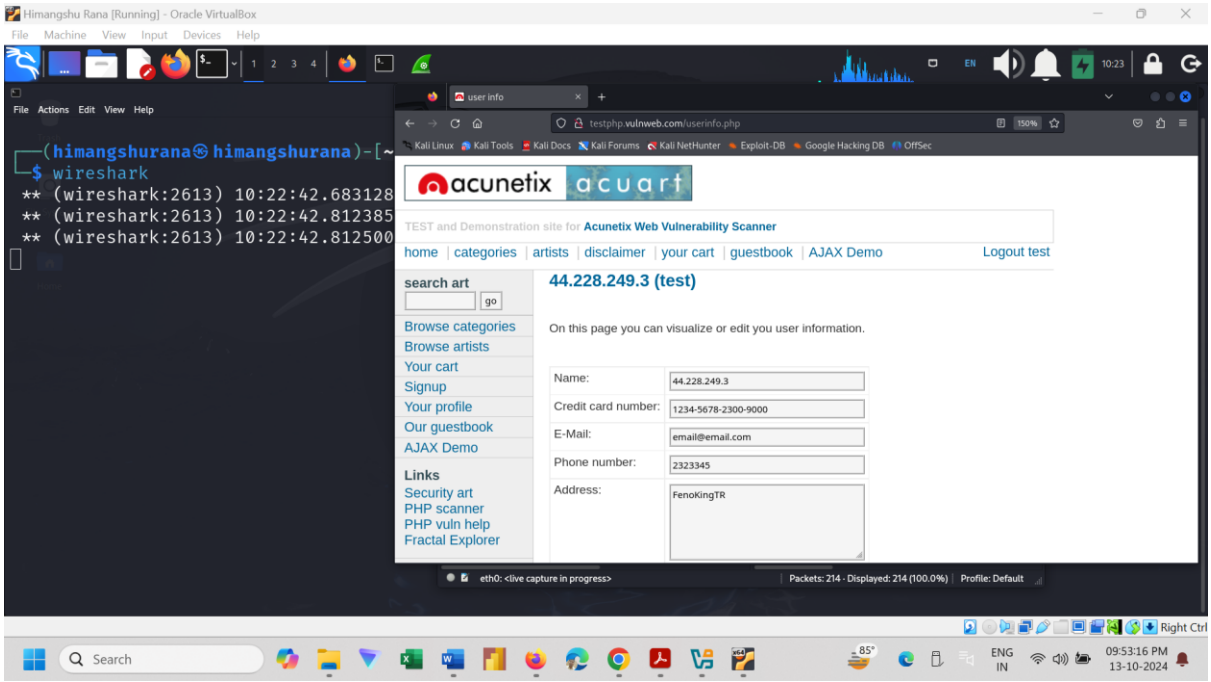
- **Score**: 7.5 (High)
- **Level**: High

Steps to Reproduce

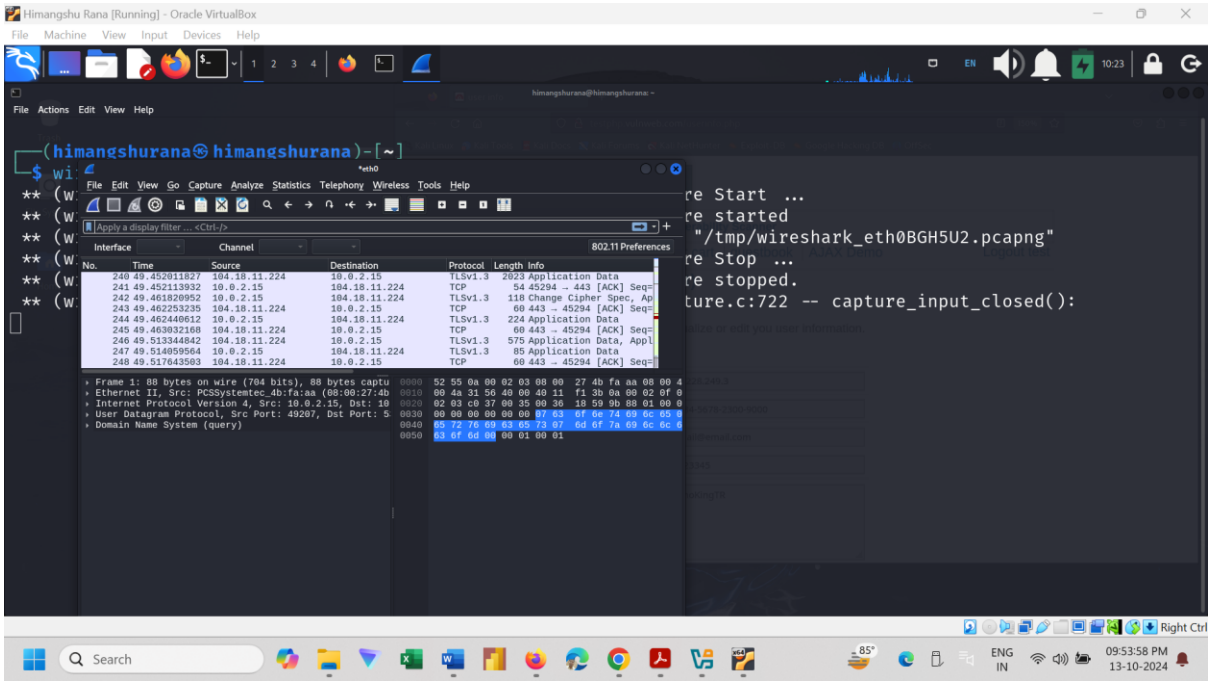
Step 1: Lunch Wireshark



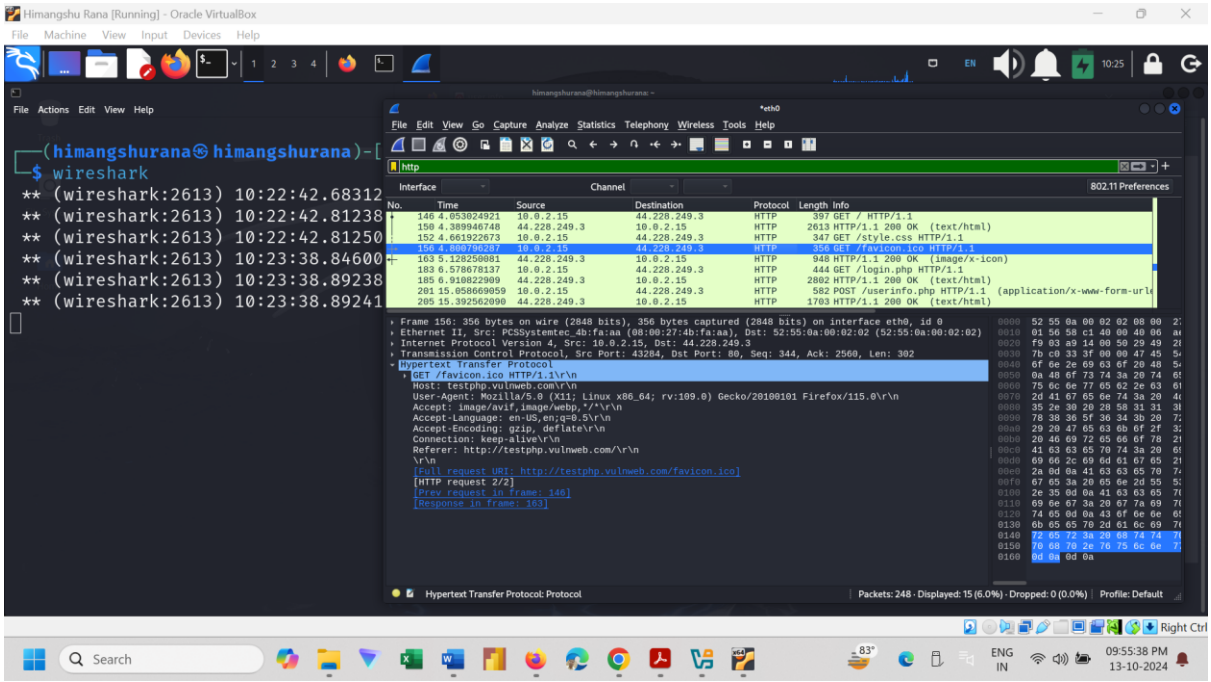
Step 2: Start capturing packets and Login to the Web Page



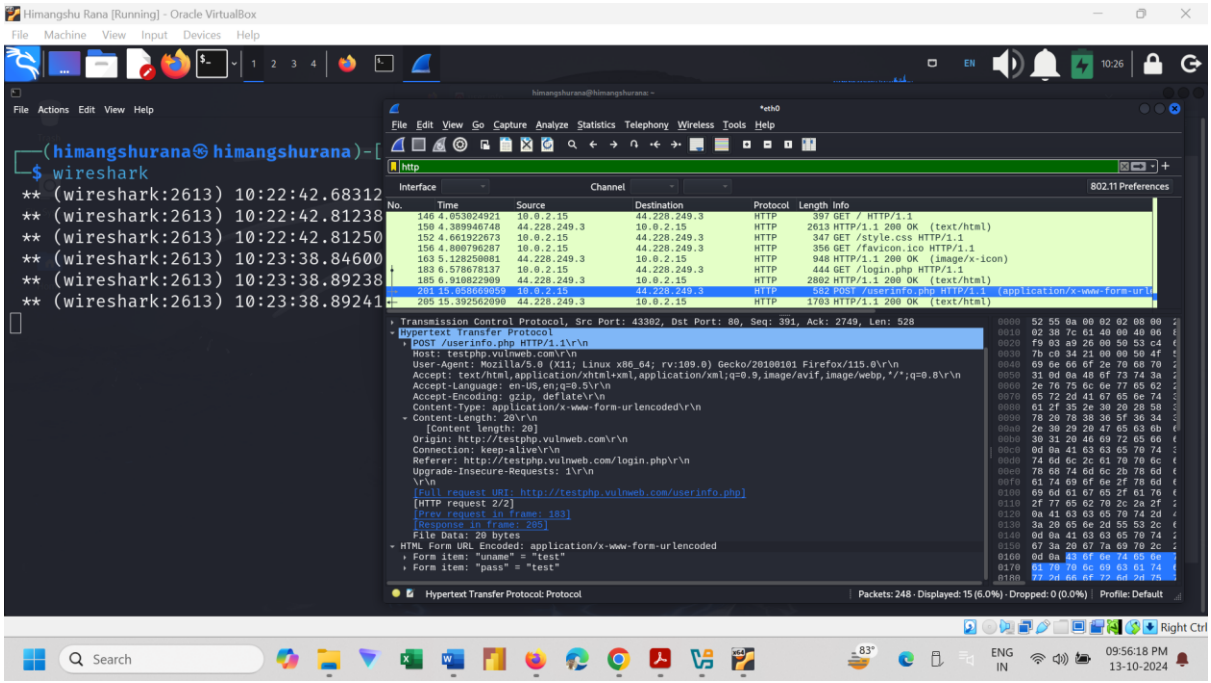
Step 3: Stop Capturing



Step 4: Filter HTTP Traffic



Step 5: Find the Login Request



Explanation: Look for the HTTP POST request that corresponds to login attempt. In the login request, the credentials being sent in the request body. Since the website uses HTTP (not HTTPS), the credentials will be transmitted in plain text.

Impact

The captured credentials could allow an attacker to:

1. Gain unauthorized access to user accounts.
2. Compromise the website's security, leading to data breaches.
3. Launch further attacks (such as account takeover or privilege escalation).

If this vulnerability is exploited, it may result in significant data loss, privacy violations, or identity theft.

Mitigation Steps

☑ **Use HTTPS (TLS/SSL)**: Implement HTTPS to ensure that all data transmitted between the user and the server is encrypted. This prevents attackers from intercepting the data.

☑ **Enforce Strong Authentication Mechanisms**: Use multi-factor authentication (MFA) to add an additional layer of security, reducing the likelihood of an attacker gaining unauthorized access.

☑ **Regular Audits and Penetration Testing**: Conduct regular security audits and penetration tests to identify and resolve vulnerabilities before attackers exploit them.

☑ **User Awareness and Training**: Educate users on the risks of using public networks to access sensitive websites, encouraging them to avoid logging in on unsecured connections.

8. References

1. OWASP: Man-in-the-Middle (MITM) Attack - https://owasp.org/www-community/attacks/Man-in-the-middle
2. Wireshark User Guide - https://www.wireshark.org/docs/wsug_html_chunked/
3. SSL/TLS Best Practices - https://www.ssl.com/article/best-practices-tls-ssl/

Task Level (Intermediate)

Q1. A file is encrypted using Veracrypt (A disk encryption tool). The password to access the file is encrypted in a hash format and provided to you in the drive with the name encoded.txt. Decode the password and enter in the vera crypt to unlock the file and find the secret code in it. The veracrypt setup file will be provided to you.

Introduction

This report documents the process of decrypting a file encrypted using VeraCrypt, a popular disk encryption tool. The password to access the encrypted file was provided in an encoded format within a file named encoded.txt, stored on the drive. The aim of this project was to decode the password, unlock the VeraCrypt file, and retrieve a secret code hidden inside.

Tools and Technology

- ☑ **VeraCrypt**: A free and open-source encryption software used for securing data on disk drives.
- ☑ **Hash Decryption Tool**: Tools such as hashcat was used to decode the password from its hash format.
- ☑ **Text Editor**: To view and manipulate the encoded password file (encoded.txt).

Attack Overview

Attack Name: Dictionary Attack on the hashed password.

The password was provided in a hash format inside the encoded.txt file. To unlock the VeraCrypt file, we needed to first decode the hashed password. A Dictionary Attack was applied to decode the password. These attacks try to guess the password by testing multiple combinations of words or characters.

- ☑ **Dictionary Attack**: This method involves testing passwords from a predefined list of common passwords or phrases until the correct one is found.

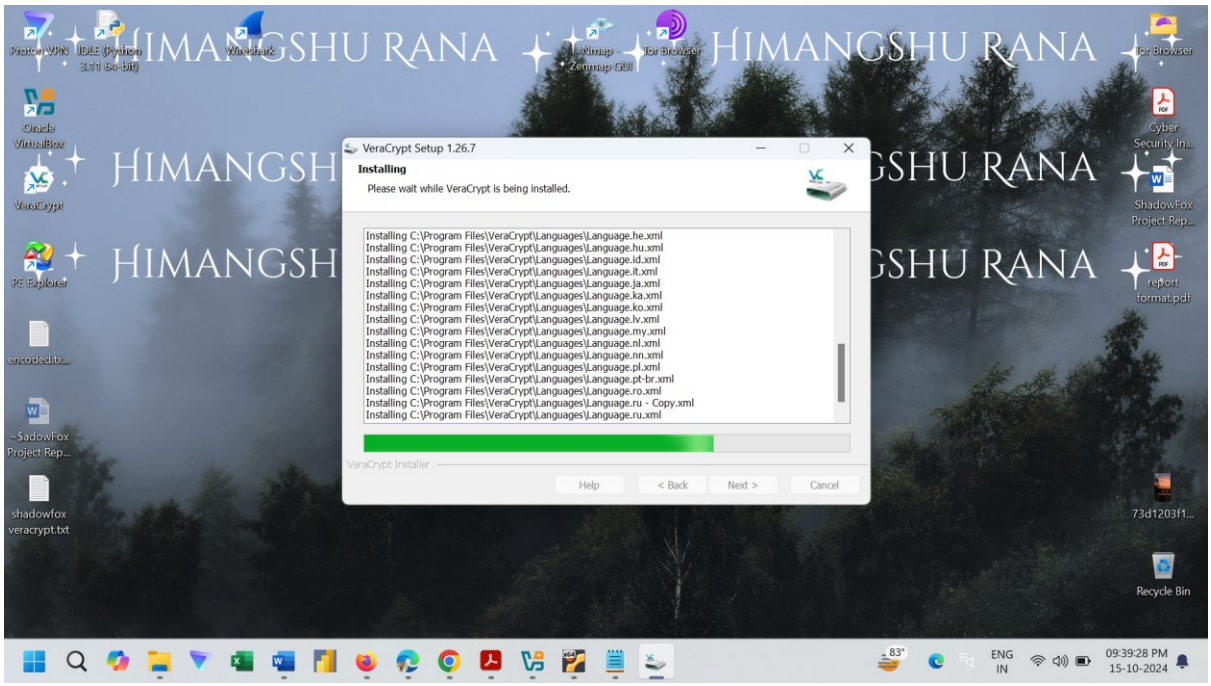
Severity

☑ **CVSS Score**: 7.5 (High)
☑ **Level**: High

This high severity is due to the sensitive nature of encrypted files. If the password is weak or exposed in a hashed form, an attacker can potentially recover the password using brute force or dictionary attacks, compromising confidential information.

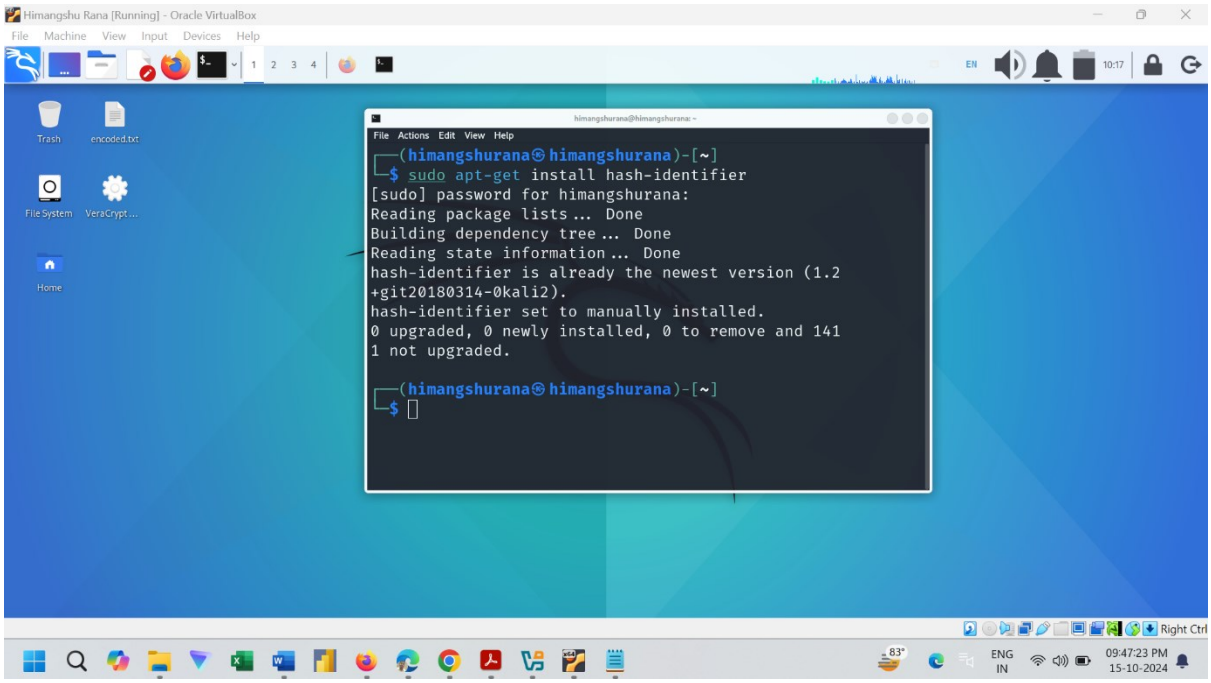
Steps to Reproduce

Step 1: Setup VeraCrypt:

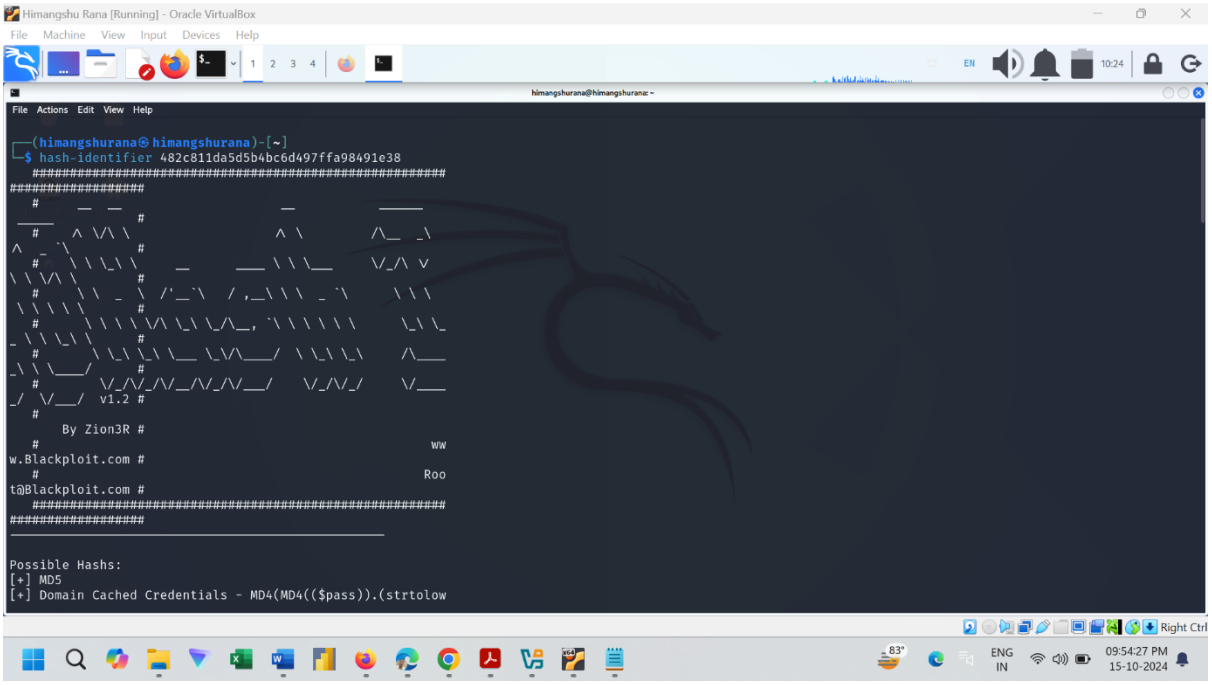


Step 2: Decode the Password:

1. **Identify the Hash Type**
   - ☑ **To identify Hash Type, install hash-identifier**

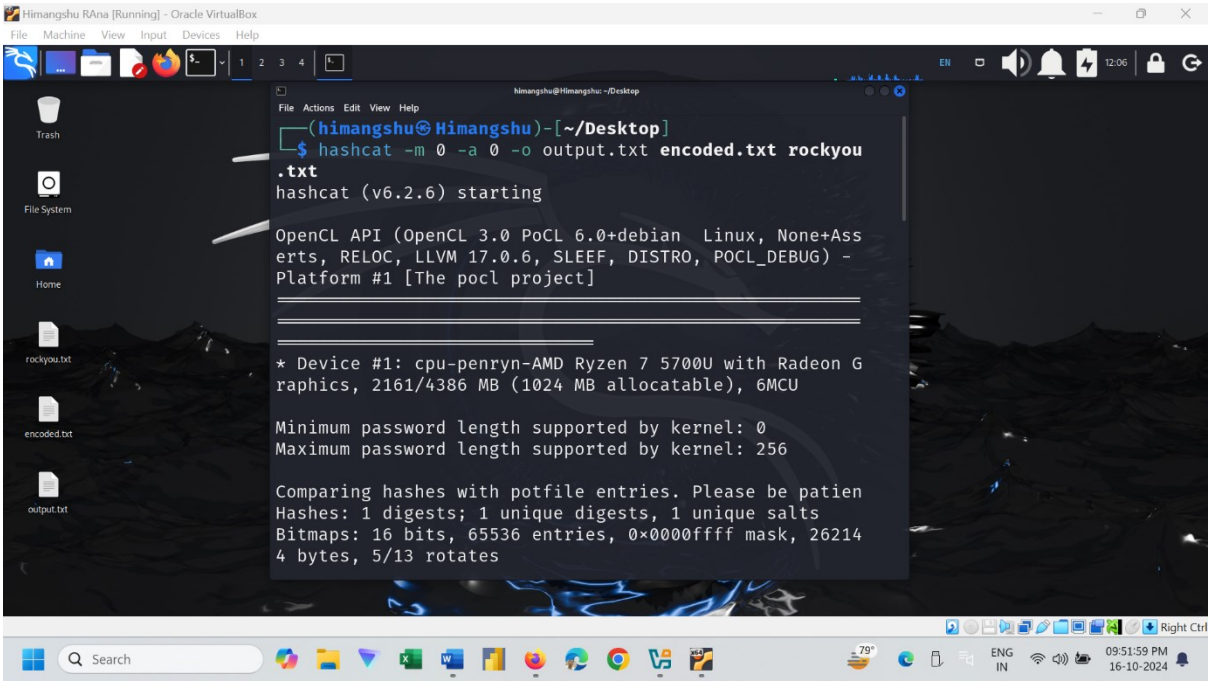


☑ Now Finding the Hash Type

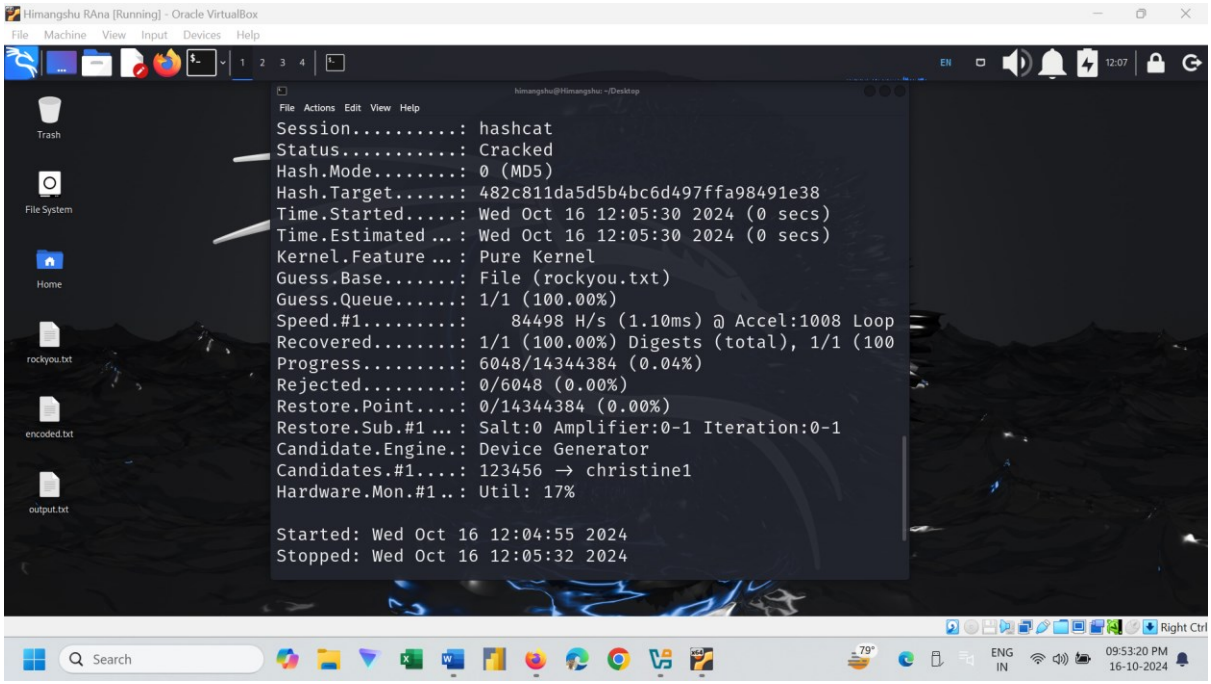


1. Run Hashcat for Decode the Password

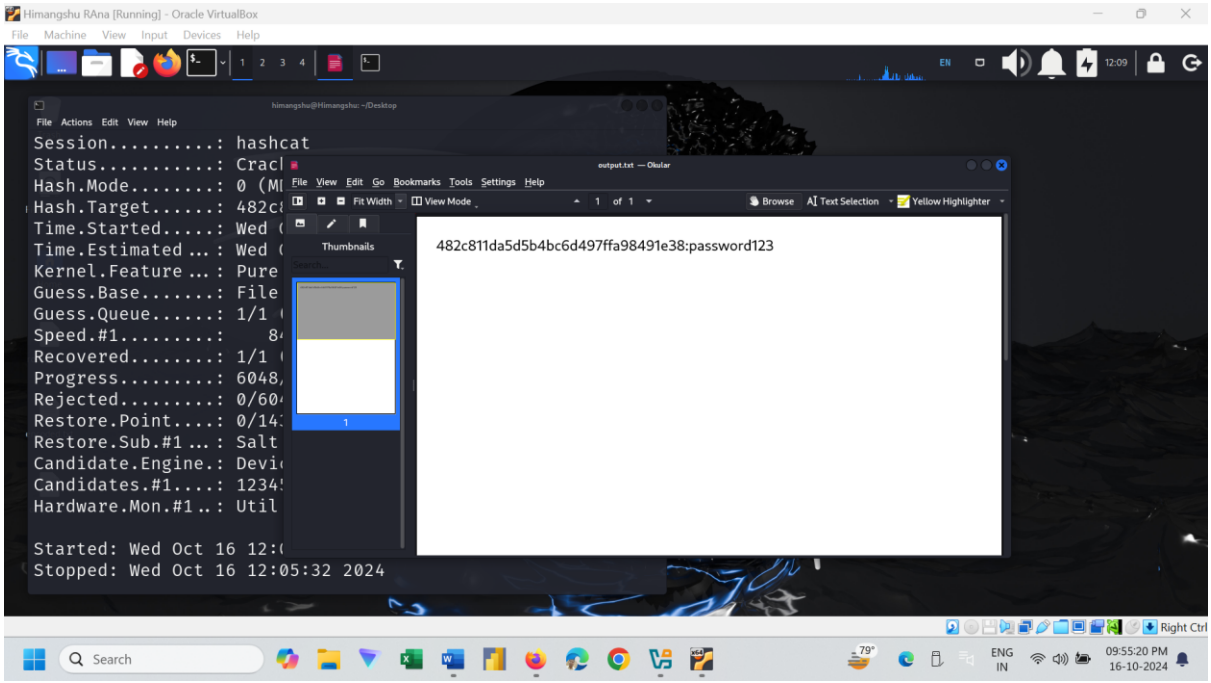
☑ Run the Command in the terminal



☑ Finally Hashcat cracked the Hash

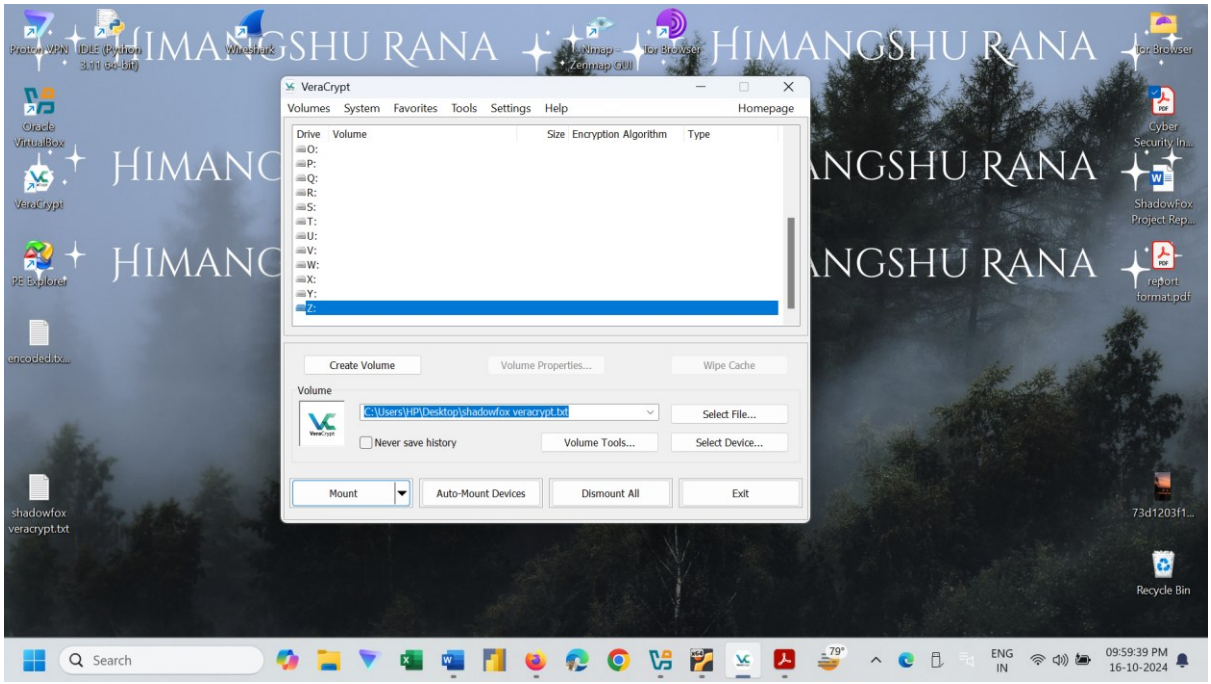


☑ The Password is: password123

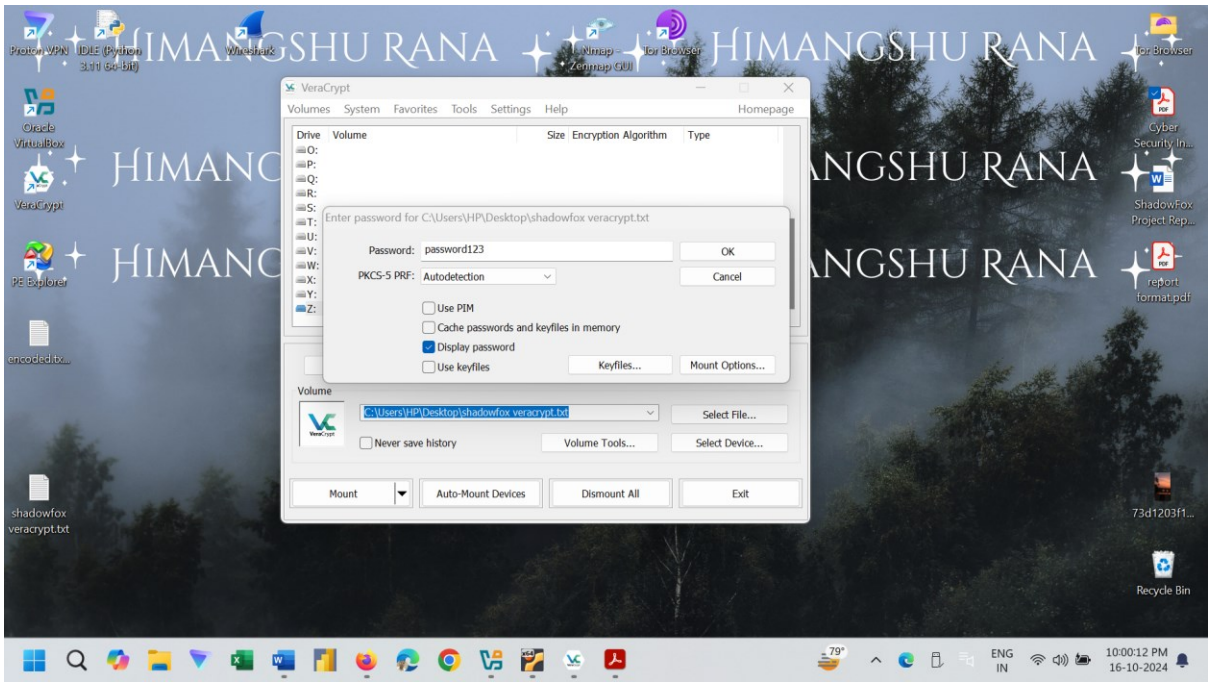


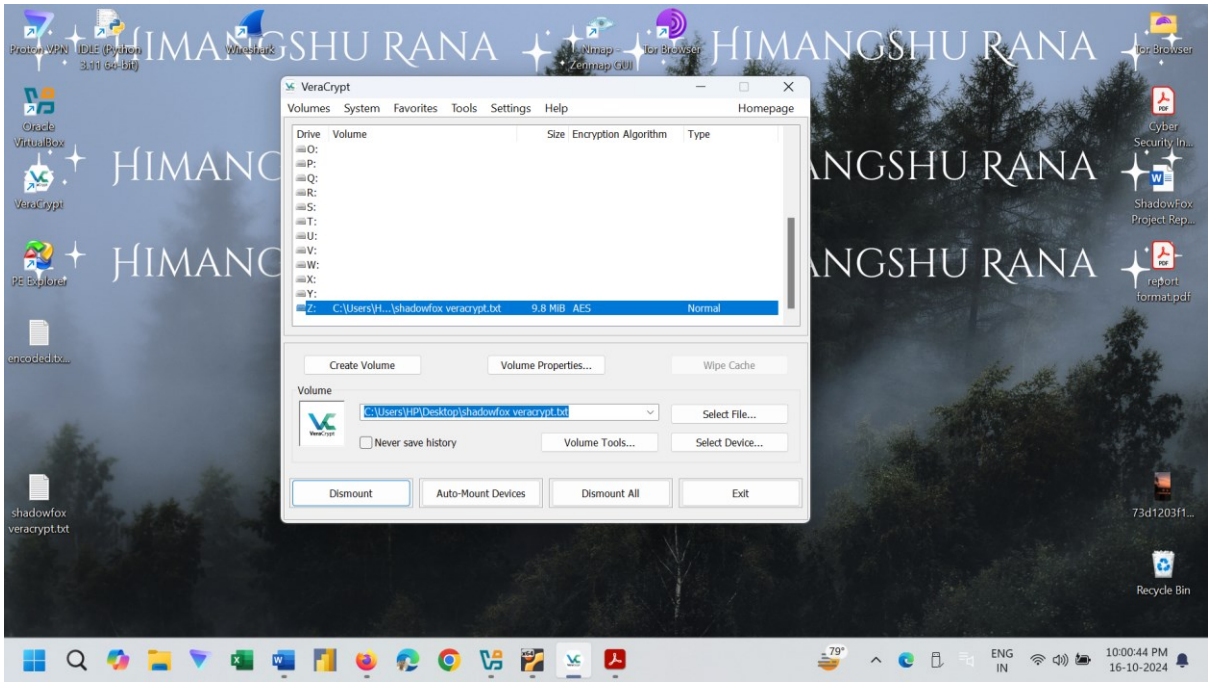
Step 3: Unlock and Access the Encrypted File:

☑ Load the File in Veracrypt

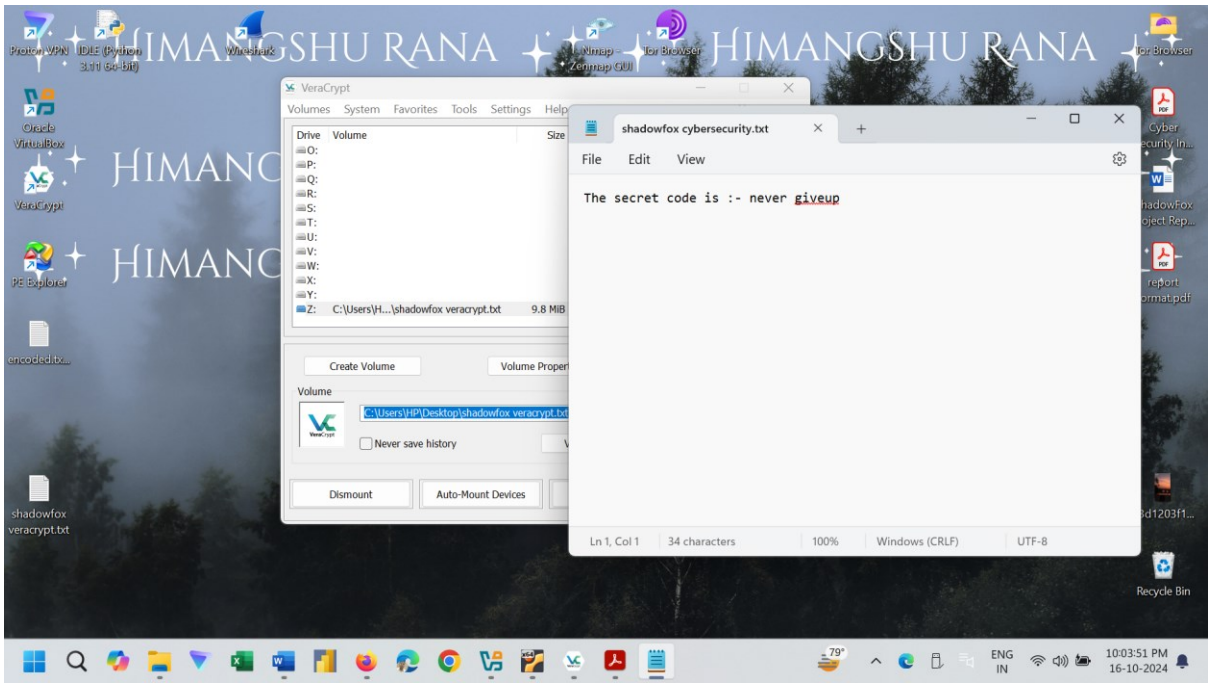


☑ Unlock the File





☑ Find the Secret Code : never giveup



Impact

Successfully decrypting the password and accessing the encrypted file shows the vulnerability of hashed passwords, particularly if a weak hash algorithm or password is used. If an attacker gains access to such information, they could retrieve sensitive or classified data, leading to data breaches or financial loss.

Mitigation Steps

- ☑ **Use Stronger Hash Algorithms**: Implement stronger hashing algorithms like bcrypt or Argon2 to avoid quick decryption.
- ☑ **Complex Passwords**: Encourage the use of long, complex passwords that resist dictionary and brute force attacks.
- ☑ **Salting**: Apply salting to passwords before hashing to make each password unique and harder to crack.

References

- VeraCrypt Official Documentation: https://www.veracrypt.fr/en/Documentation.html
- Hashcat: https://hashcat.net/hashcat/
- CrackStation Hash Decryption Tool: https://crackstation.net/
- Common Vulnerability Scoring System (CVSS) v3.1: https://www.first.org/cvss/

Q2. An executable file of veracrypt will be provided to you. Find the address of the entry point of the executable using PE explorer tool and provide the value as the answer as a screenshot.

Introduction

This report outlines the process of finding the entry point address of a VeraCrypt executable using the PE Explorer tool. VeraCrypt is an open-source encryption software widely used for securing data. Analysing executable files for potential vulnerabilities is essential in understanding and mitigating security risks. The focus of this report is to locate the entry point of the executable file, which is a key part of understanding how a program is initiated in memory.

Tools and Technology

- ☑ **VeraCrypt Executable File:** The executable provided is a VeraCrypt file that needs to be examined.
- ☑ **PE Explorer:** This tool is used to explore Portable Executable (PE) files, providing insights into their structure, headers, and entry point details.
- ☑ **Operating System:** Windows-11 operating system for running PE Explorer and analysing the executable file.

Attack Overview

Attack Name: PE Header Manipulation Attack.

In a PE Header Manipulation Attack, attackers alter the entry point of an executable to redirect the execution to malicious code. By manipulating the PE headers, an attacker can make the system execute unintended code when the program runs. Understanding the entry point helps security analysts detect such attacks. The entry point of an executable is critical in executing the first instructions of a program, and tampering with it can lead to security exploits.

Severity

CVSS Score: 7.5 (High) The attack could allow malicious code to run before any legitimate program functionality starts, potentially giving attackers control over the system.

Impact Level: High

Confidentiality Impact: High

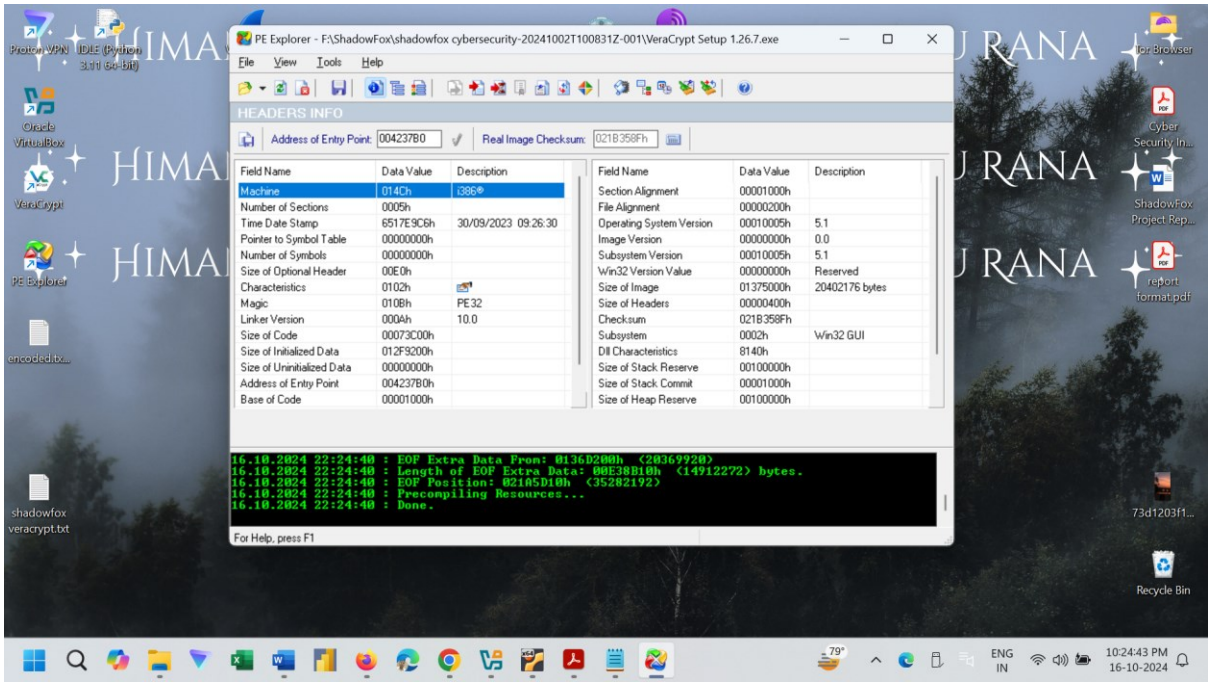
Integrity Impact: High

Availability Impact: Medium

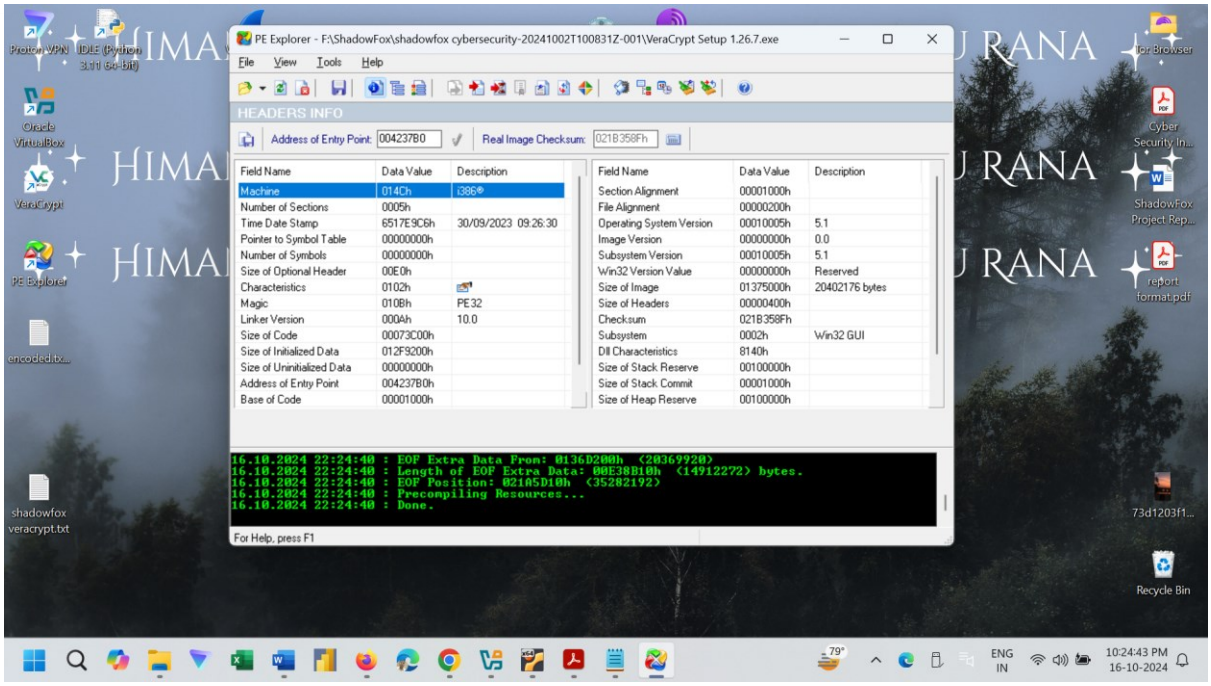
Steps to Reproduce

Step 1: Install PE Explorer:

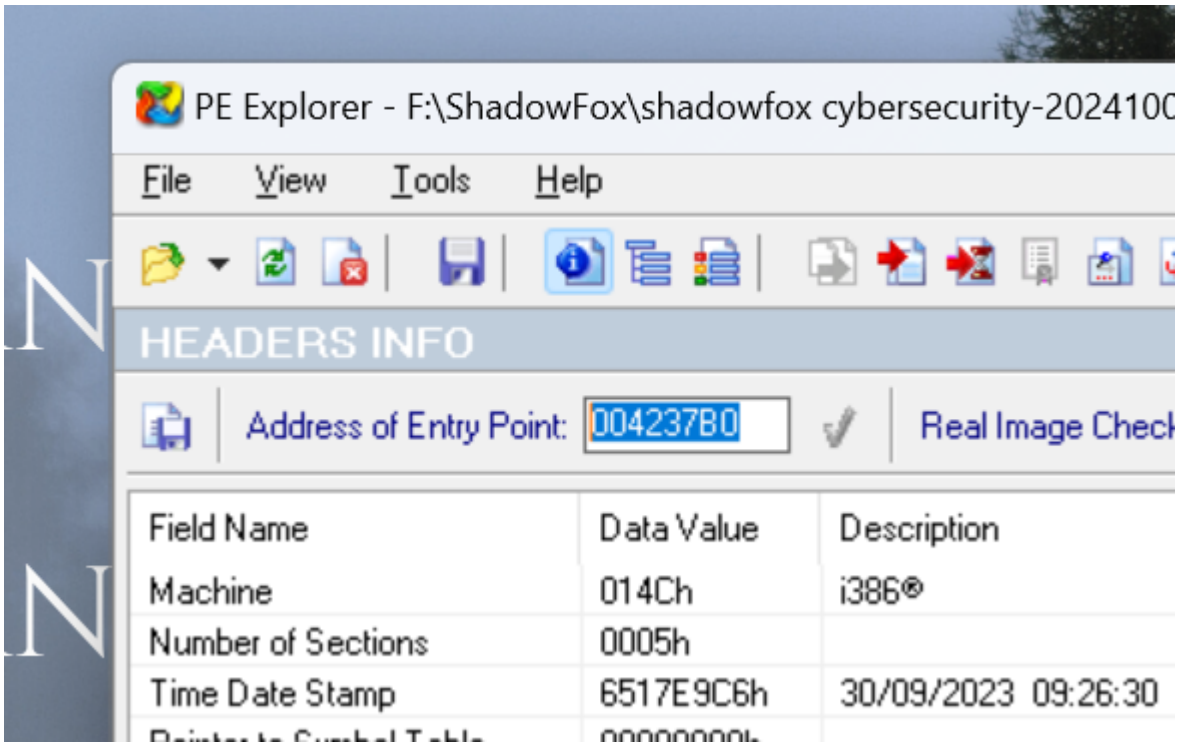
Step 2: Load the VeraCrypt Executable:



Step 3:Navigate to the PE Header:



Step 4: Locate the Entry Point Address: 004237B0



Impact

Finding the entry point is a key step in understanding the flow of a program. If the entry point is manipulated, attackers can gain unauthorised access or execute malicious code. In this case, the report demonstrates how security teams can use tools like PE Explorer to inspect binaries and identify potential vulnerabilities.

Mitigation Steps

- Code Signing: Ensure that the executable file is digitally signed. This helps verify its authenticity and detect any tampering.
- Security Audits: Regularly audit software binaries and check their PE headers for any modifications.
- Antivirus and Anti-malware: Use up-to-date antivirus software to scan and protect against malicious executables.
- Integrity Checks: Implement hash verification to ensure the integrity of the executable file before execution.

References

- PE File Format Documentation: https://docs.microsoft.com/en-us/windows/win32/debug/pe-format
- VeraCrypt Official Website: https://www.veracrypt.fr/en/Home.html
- PE Explorer Tool: https://www.heaventools.com/PE-explorer.htm
- Common Vulnerability Scoring System (CVSS): https://www.first.org/cvss/

Q3. Create a payload using Metasploit and make a reverse shell connection from a Windows machine in your virtual machine setup.

Introduction

In this project, we explore how to create a malicious payload using Metasploit, a popular penetration testing framework, and establish a reverse shell connection. The reverse shell allows an attacker to gain control over a remote machine. For this demonstration, the attack is performed on a Windows 7 machine set up within a virtual environment.

Tools and Technology

☑ **Metasploit Framework**: A powerful penetration testing tool that provides exploit modules and payloads for security testing.

☑ **Windows 7**: The target operating system in this scenario.

☑ **VirtualBox**: Virtualisation platforms used to create a controlled testing environment.

☑ **Kali Linux**: The attacker's machine, equipped with Metasploit, running in a virtual environment.

☑ **msfvenom**: A Metasploit tool used to generate the malicious payload.

☑ **Reverse Shell**: A shell session in which the target system initiates a connection to the attacker’s machine.

Attack Overview

Attack Name: Reverse Shell Attack

The reverse shell attack exploits the target system by creating a payload that, once executed on the target machine (Windows 7), connects back to the attacker's machine. This connection allows the attacker to execute commands on the compromised system remotely.

Attack Process:

- The attacker uses msfvenom to generate a malicious payload that creates a reverse shell.
- The payload is delivered to the target machine using apache2.
- Once the payload is executed on the target, it opens a connection to the attacker's machine, granting control of the target system.

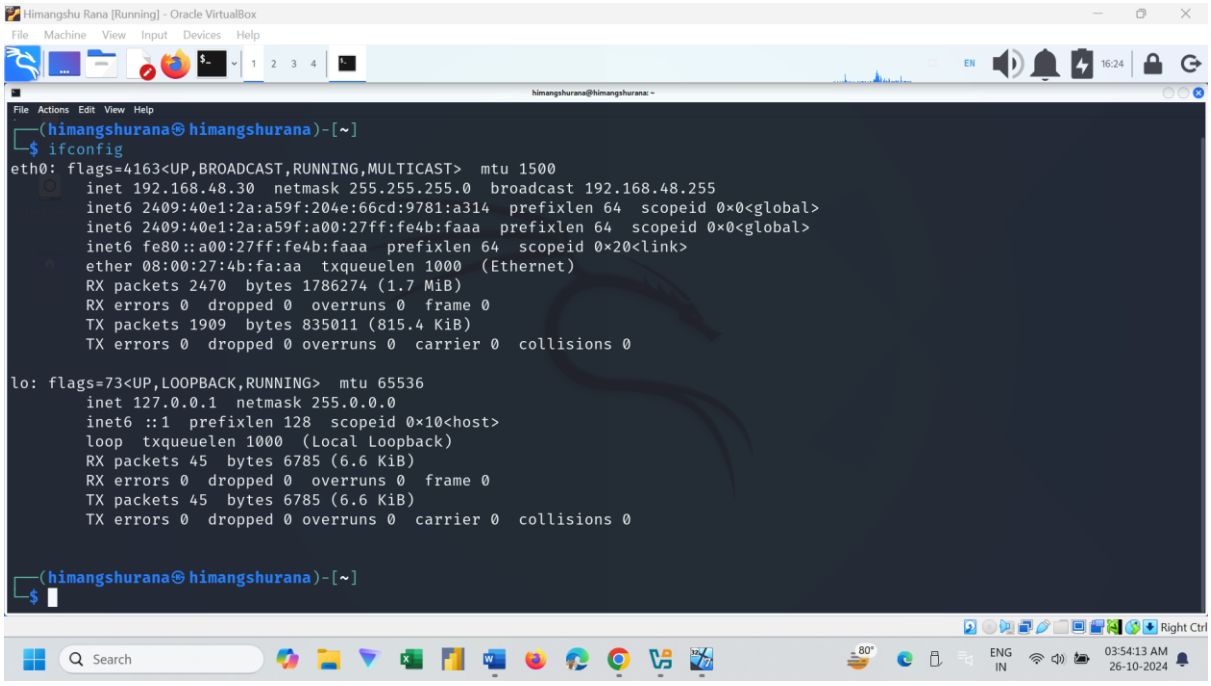
Severity

- ✓ **Score**: 8.2 (High)
- ✓ **Severity Level**: Critical

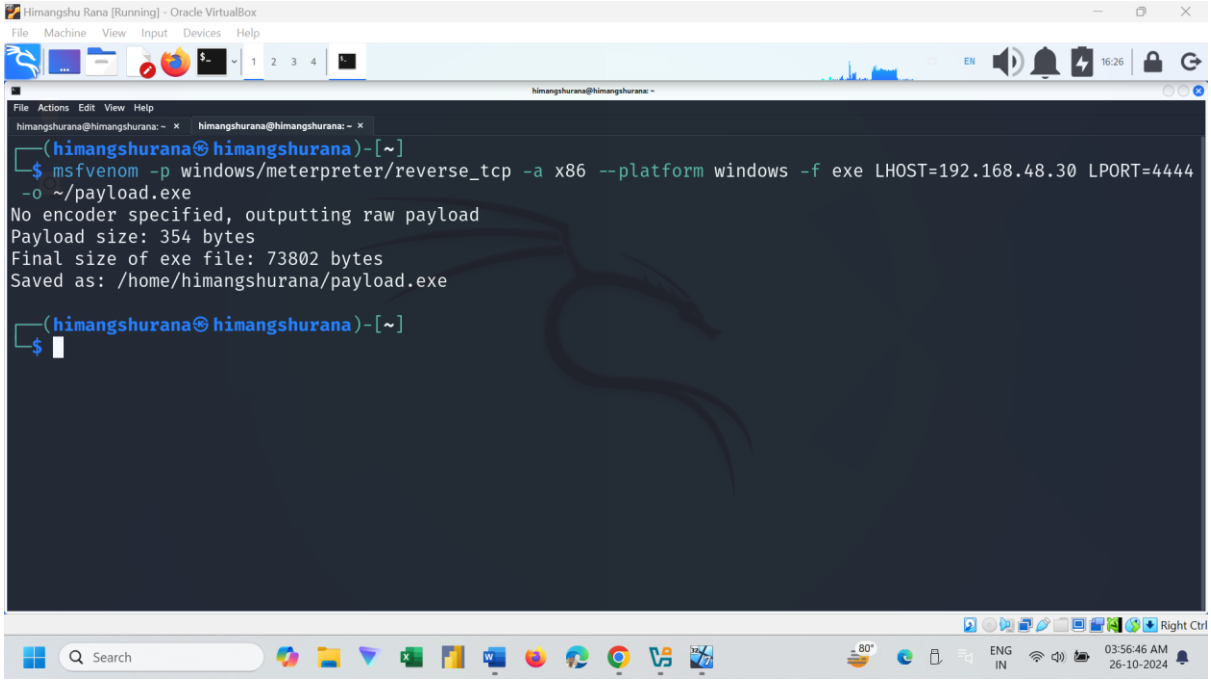
The high severity of this attack stems from the fact that once the reverse shell is established, the attacker has full control over the compromised system. This could lead to data theft, system compromise, or further exploitation.

Steps to Reproduce

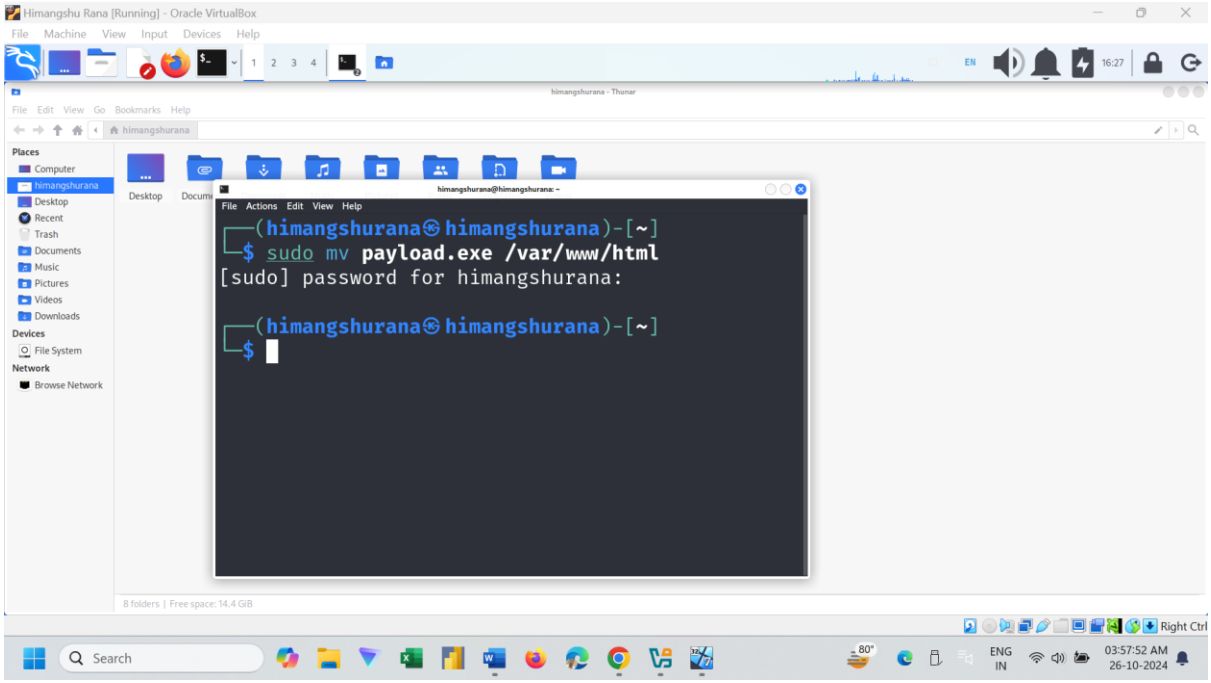
Step 1: Open the Kali Linux virtual machine and note its IP address (e.g., 192.168.48.30).



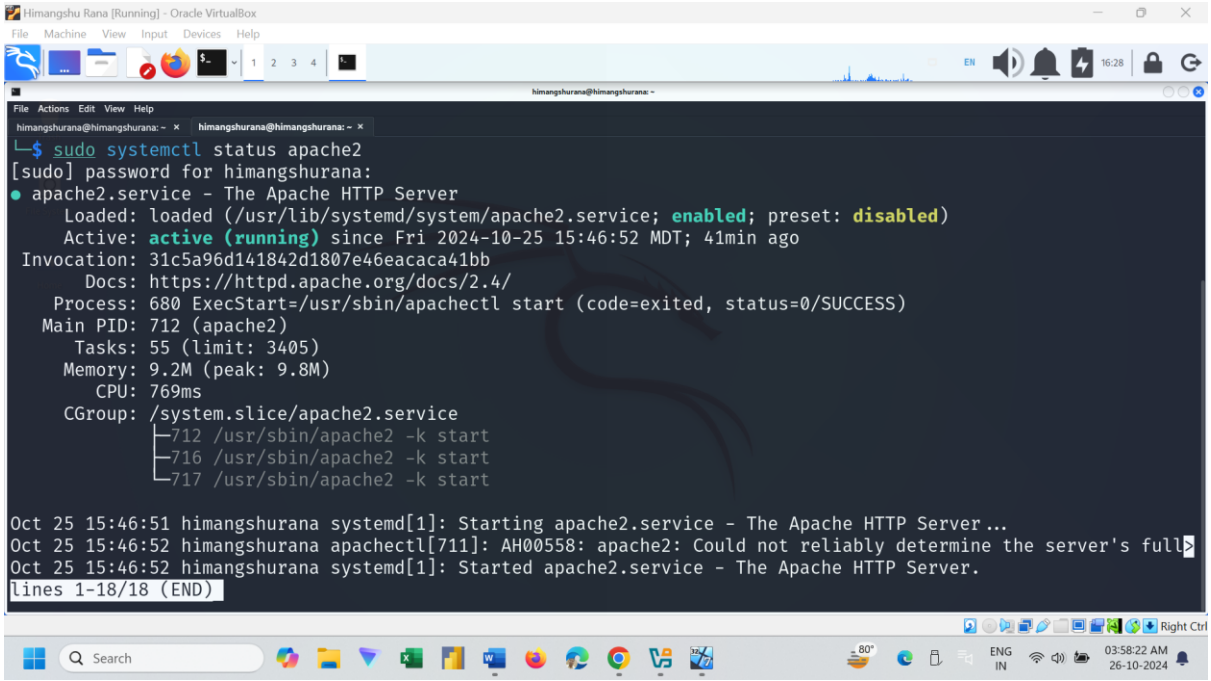
Step 2: execute the “msfvenom” script to create a standalone payload as an executable file.



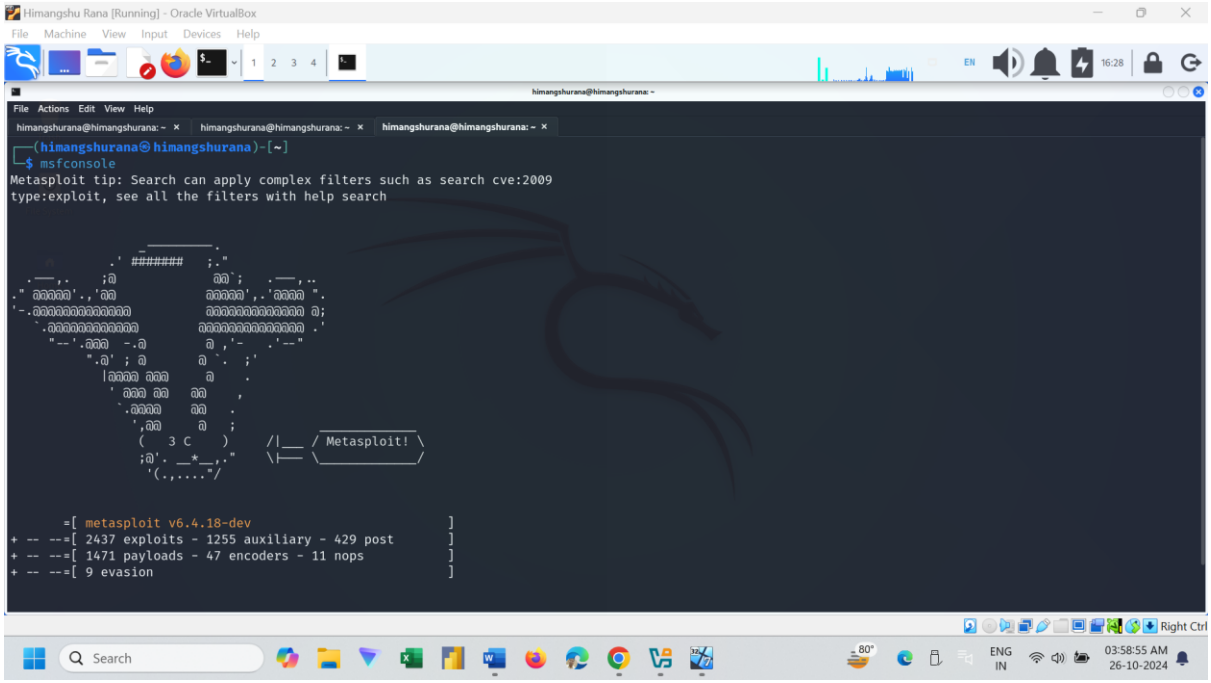
Step 3: Move the Payload in "/var/www/html"



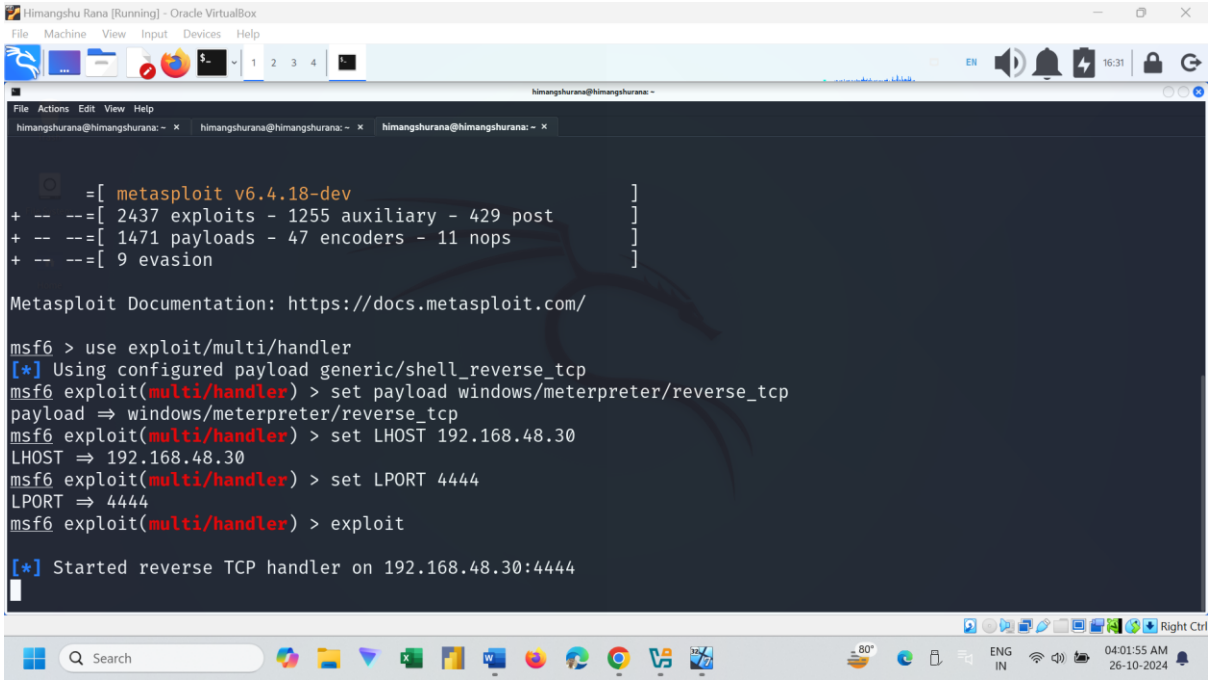
Step 4: Create a web file server using apache2



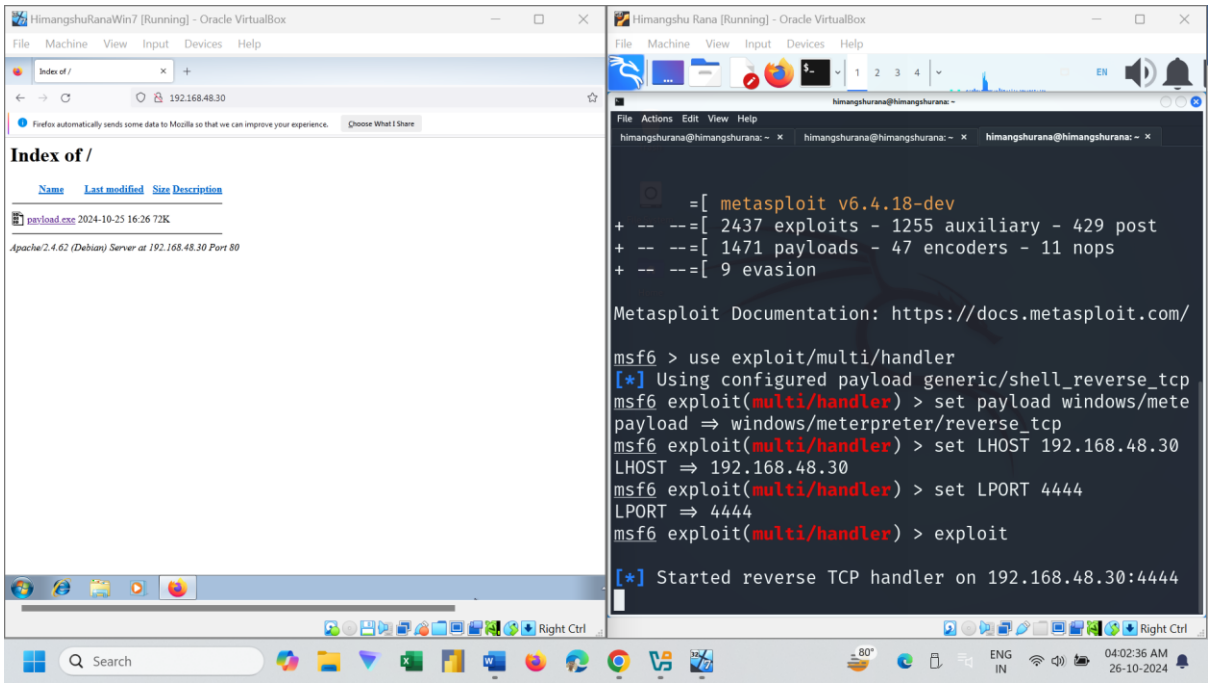
Step 5: Start the Metasploit Framework console



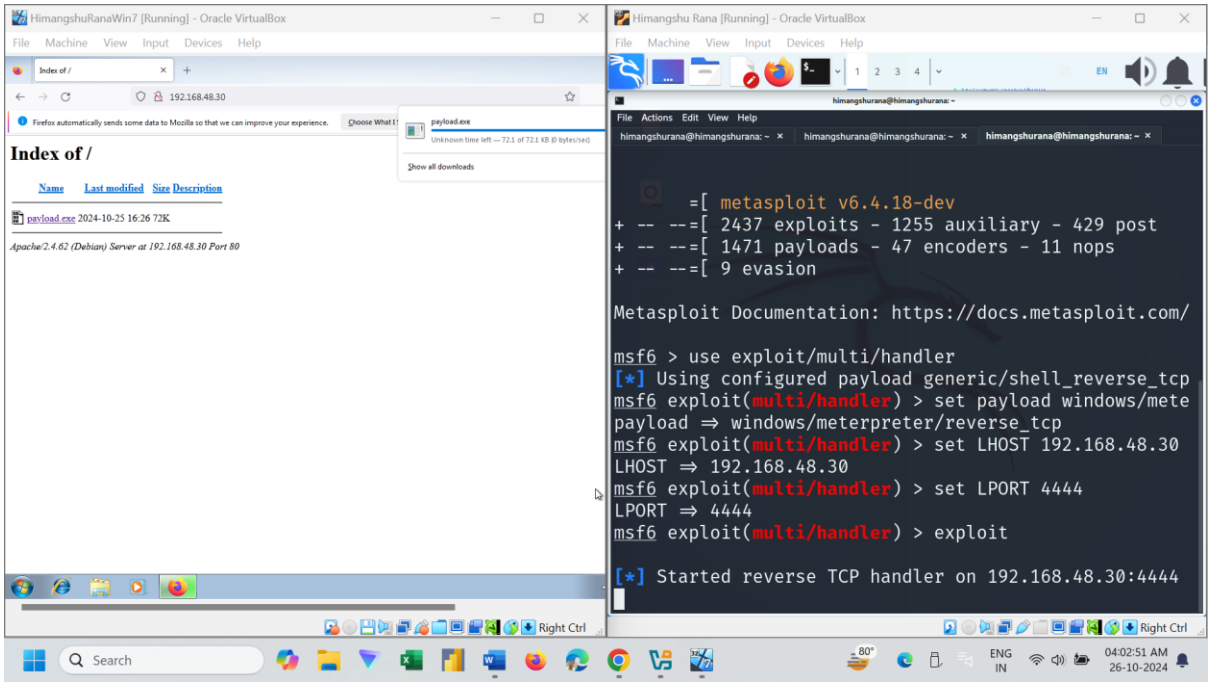
Step 6: Configure it to start the server on the meterpreter & waiting for the payload connection.



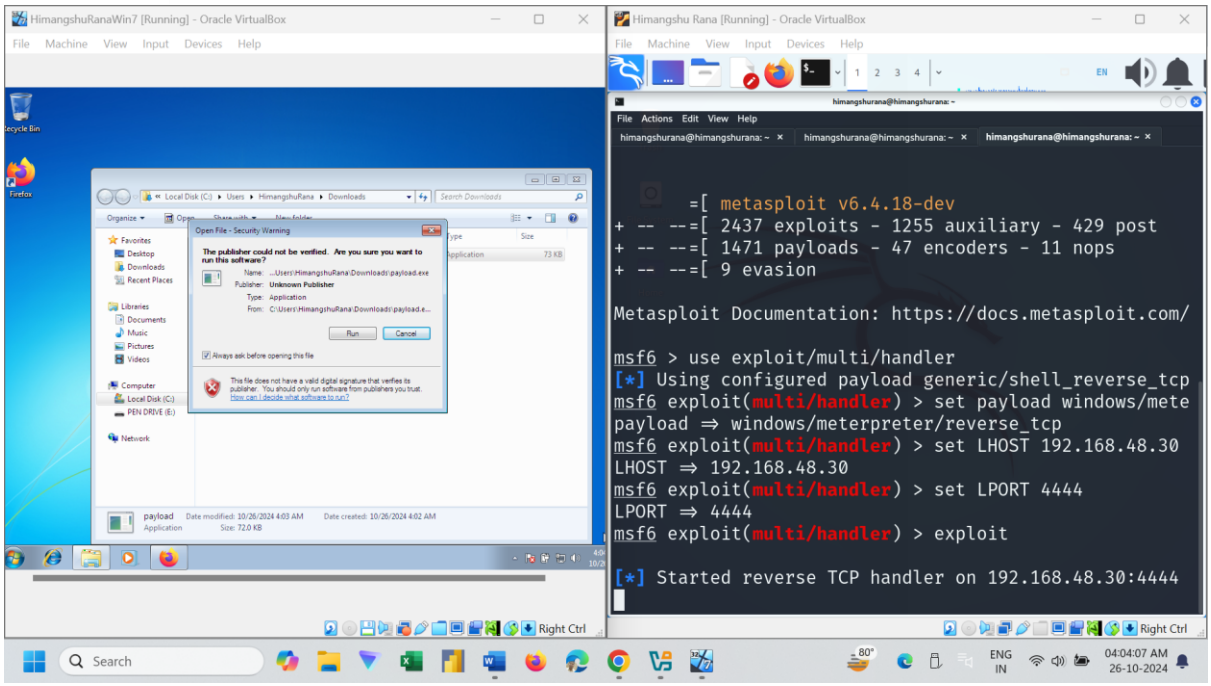
Step 7: Open Firefox on the victim machine & type the IP address of the Kali machine (e.g., 192.168.48.30).



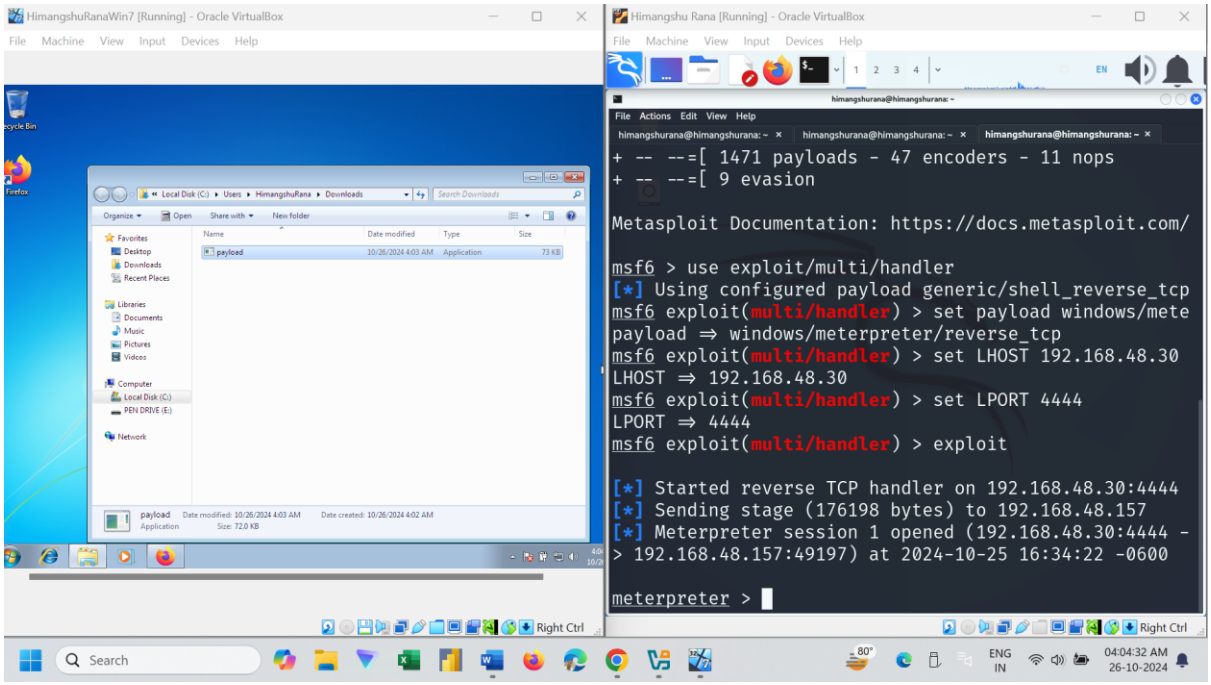
Step 8: Access the HTTP web server and locate the payload.exe file & download the payload.exe.



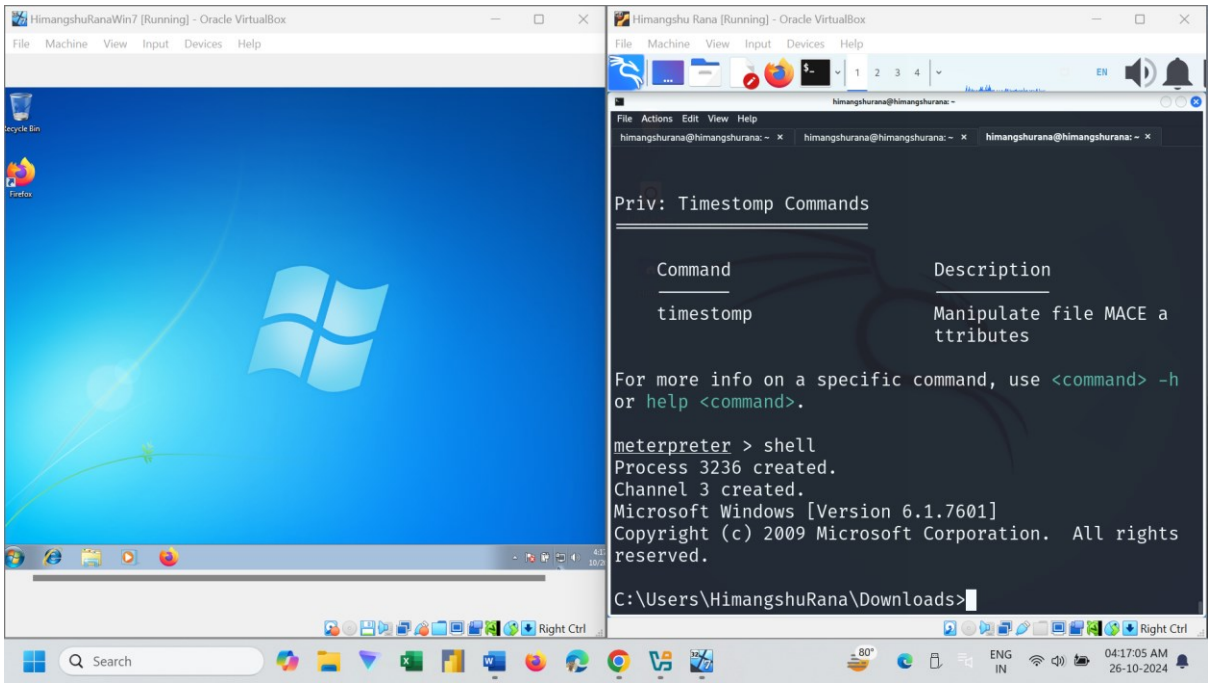
Step 9: Run the payload.exe



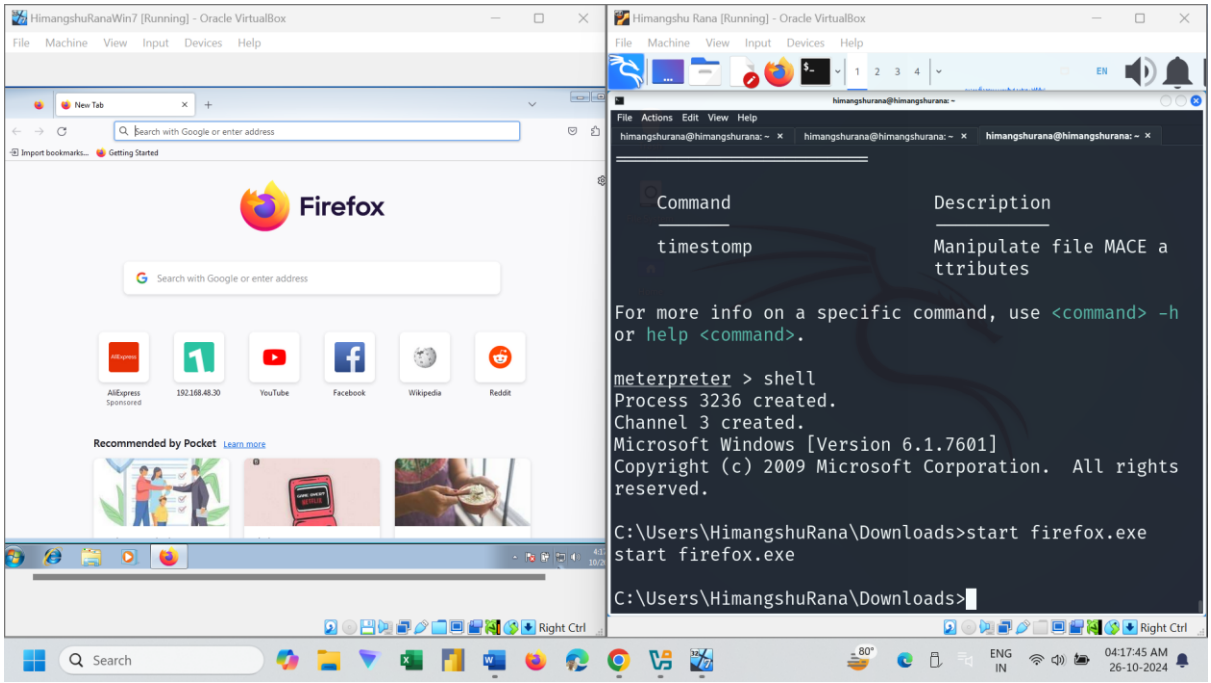
Step 10: After executing the payload, the Kali machine receives a request and creates a Meterpreter session with the Windows victim machine.



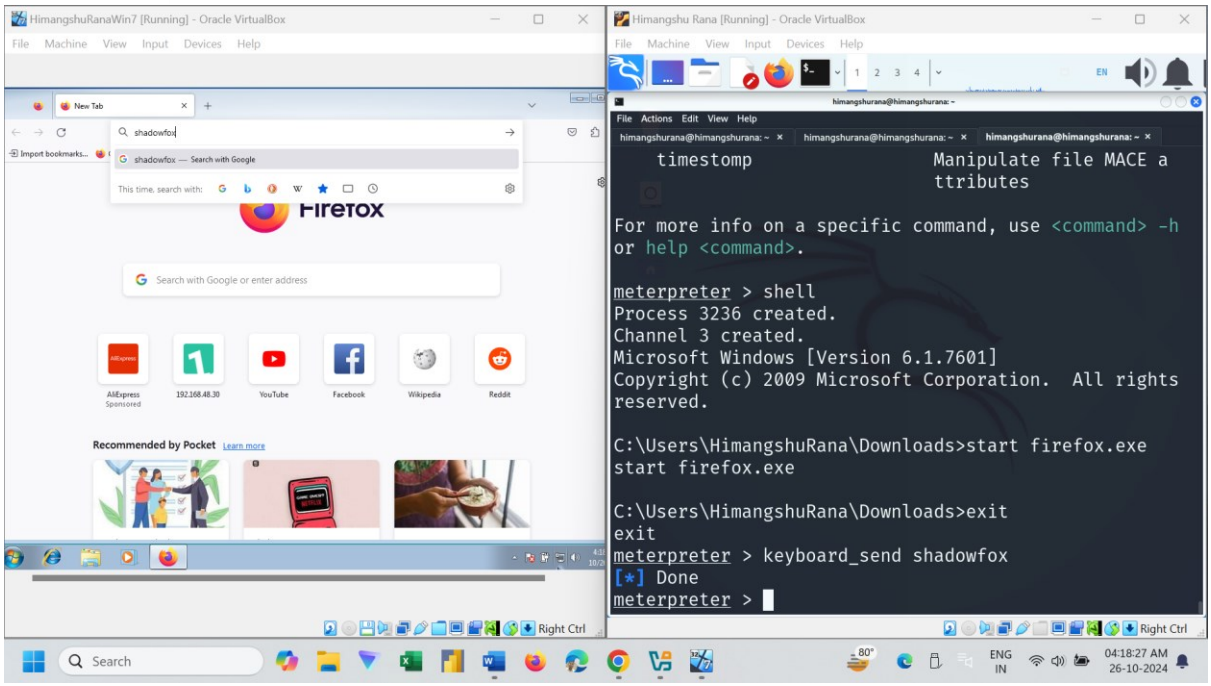
Step 11: Execute a reverse shell attack by typing “shell” in the Meterpreter session.



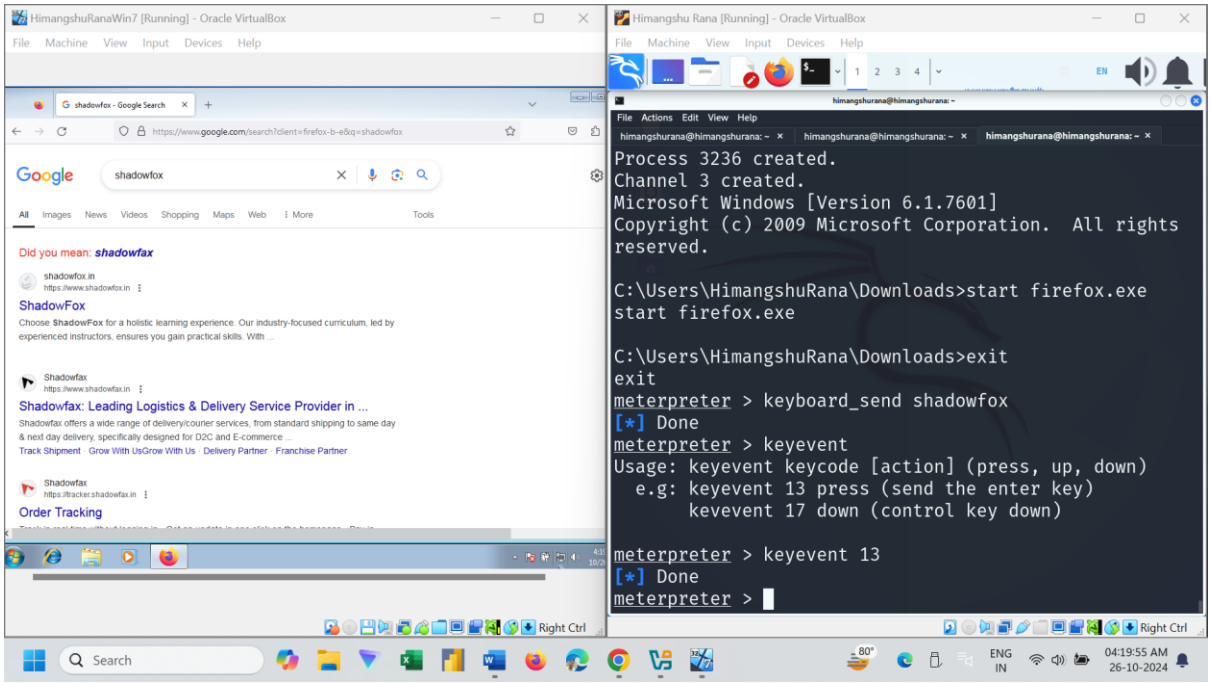
Step 12: Open Firefox



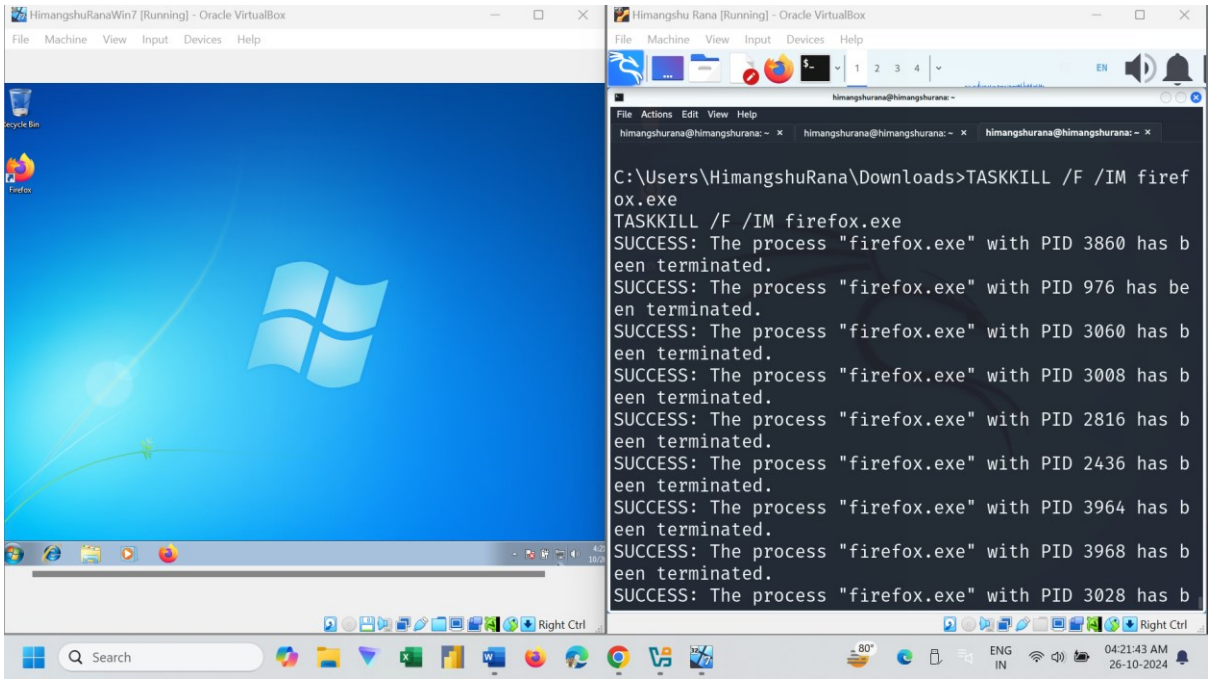
Step 13: Type Shadowfox in Firefox



Step 14: Press Enter



Step 15: Close Firefox



Impact

Once the reverse shell connection is established, the attacker gains significant control over the target machine. This may lead to:

- Unauthorised access to sensitive data
- Execution of arbitrary commands
- Installation of additional malware or spyware
- Potential to compromise an entire network, depending on the access level of the compromised system

Mitigation Steps

- **Regular Software Updates**: Ensure that all systems, including Windows, are up to date with the latest security patches.
- **Antivirus and Anti-malware Software**: Use up-to-date antivirus software that can detect and block malicious payloads.
- **Network Monitoring**: Implement network monitoring tools to detect unusual outbound connections that could indicate a reverse shell.
- **Limit User Privileges**: Restrict the use of administrative privileges and reduce the attack surface by implementing the principle of least privilege.
- **Security Awareness Training**: Train users to recognise phishing emails and avoid downloading or executing suspicious files.
- **Firewalls and Intrusion Detection Systems (IDS)**: Configure firewalls to block outbound traffic from unknown or suspicious sources and use IDS to detect reverse shell activity.

References

- Offensive Security. "Metasploit Unleashed - The Official Metasploit Guide." https://www.offensive-security.com/metasploit-unleashed/
- Rapid7. "Metasploit Framework." https://www.metasploit.com/